UNITED STATES
DEPARTMENT OF THE INTERIOR

Earthquake Risk Analysis Using Numerical and Stochastic Models of Time-Dependent Strain Fields

Albert Smith
Earth Sciences Board
University of California
Santa Cruz, CA 95064

USGS CONTRACT NO. 14-08-0001-G-399
Supported by the EARTHQUAKE HAZARDS REDUCTION PROGRAM

OPEN FILE NO. 80-1155

U.S. Geological Survey
OPEN FILE REPORT

FINAL TECHNICAL REPORT

Earthquake Risk Analysis Using Numerical and Stochastic Models of Time-Dependent Strain Fields

Sponsored by the U. S. Geological Survey
No. 14-08-0001-G-399

Contractor: University of California, Santa Cruz

Principal Investigator: Albert T. Smith

U.S.G.S. Project Officer: Dr. Jack F. Evernden

Effective Date: October 15, 1976

Expiration Date: October 14, 1979

Submitted: March 12, 1980

SUMMARY OF FINAL TECHNICAL REPORT

14-08-0001-G399

Earthquake Risk Analysis Using Numerical and Stochastic Models of Time-dependent Strain Fields

Albert T. Smith
Earth Sciences Board
University of California
Santa Cruz, CA. 95064

March 10, 1980

In this research we have developed a solution method to treat the potential mechanisms causing the Palmdale uplift. In this report I outline the progress, consider defects in the approach of other investigators, and emphasize the necessary features to explain the Southern California uplift. The problem is not resolved as yet; however, the work does offer some corstraints.

Previous models of the Southern California uplift suffer from inappropriate assumptions and numerical errors. Using the post-seismic deformations for the 1946 Nankaido earthquake, we indicate the origin of these difficulties and develop the requirements for a model in Southern California. First, the viscoelastic half-space used by Thatcher and Rundle(1979) for the asthenosphere gives unsatisfactory models. In addition, numerical instabilities invalidate their solution using an elastic bulk modulus. A constant lambda and viscoelastic bulk modulus are also inappropriate for the asthenosphere. Finally, lateral heterogeneities strongly effect the deformations and invalidate any simple analytical models if compared to detailed geodetic data.

Using an improved geodetic data set, the 1946 Nankaido earthquake does provide constraints for the viscosity structure of the asthenosphere.

(1) A 30 to 40 km thick elastic lithosphere.

(2) A thin, 20 km thick low-viscosity channel with an effective viscosity of 10**21 poise.

(3) A higher viscosity mesosphere with an approximate viscosity of 10**21 poise or greater.

(4) A zone or layer of low effective viscosity within the asthenosphere adjacent to the subducting slab. Within

the zone the viscosity is less than 2.x10**19 poise.

These results offer important constraints for models of Southern California. They suggest that both Kosloff's(1977) and Rundle's(1979) solutions for the southern California uplift are incorrect: The first ignores relaxation in the asthenosphere and assumes an elastic problem, while the second model contains numerical instabilities and incorrectly assumes a half-space with a viscoelastic bulk modulus. These assumptions will strongly effect the resulting deformations.

Our proposals and reports (Smith, 1977, 1978, 1979) suggest a combination of the following mechanisms generating the unusual pattern of vertical and horizontal movements in Southern California:

(1) Complex fault geometry represented by the San Andreas fault and branch faults.

(2) Stress relaxation in the asthenosphere originating from a low-viscosity channel and lateral heterogeneities.

(3) Coupling of aseismic slip between faults and fault segments.

Kosloff (1977) has treated the first of these for the elastic case, while the final mechanism adopts greater importance in light of Mavko and Stuart's(1979) solution for coupling of aseismic creep between the San Andreas and Calaveras faults. The time scale of the vertical and horizontal deformations, however, require including the second process, stress relaxation in the asthenosphere.

To allow solution of a general problem, the finite element method was chosen as the solution method. A three-dimensional code has been developed to include the following conditions:

(1) General, linear viscoelastic media.

(2) Nonlinear constitutive relation along the fault.

(3) Complex fault configurations including intersecting thrust and strike-slip faults.

(4) Lateral heterogeneities within the crust and asthenosphere.

To differentiate the effect of mantle relaxation from fault creep and to demonstrate the utility of the computer code for three dimensional problems, we solve a simple model of a strike slip fault within a layered media. In our

computations asthenospheric stress relaxation cannot account for the deformations occurring in southern California; complex fault geometries and fault creep are also necessary. Relaxation, however, will perturb the deformations by at least 5 centimeters.

To allow direct comparison to observed geodetic data, we are now constructing a model of southern California which includes aseismic fault creep or instabilites, complex fault geometries, and stress relaxation. Using the approach of Mavko and Stuart(1979), the solutions will place constrairts on the contribution of each factor.

TABLE OF CONTENTS

1. INTRODUCTION

The origin of the Palmdale uplift remains a perplexing and unresolved problem. The research in this grant provides techniques to resolve the question and to suggest alternative theories for the origin. In this report I will outline the progress, present the solution methods and programs, and give suggestions for future work. The problem is not resolved as yet; however, the work does offer some constraints. Development of solution methods proved to be more formidable than anticipated.

The Southern California uplift, if we assume its existence, has been attributed to two mechanisms: The first proposal viewed the uplift as the interaction of complex fault geometries (Kosloff, 1977). Using an elastic model, the region was modelled as a plate with vertical deformation constrained to zero at its base. The faults were modelled as weak elements within the plate. When the edges are displaced to simulate the motion of the North American and Pacific Plates, these weak zones were allowed to creep. The results suggest the importance of complex fault geometries, specifically, the bend in the San Andreas fault and related branch faults.

This simple finite element model suffers, however, from a number of major assumptions. The crust is locked at its base; an unrealistic assumption in light of the low viscos-

ity of the asthenosphere (see later sections). The effects of thrust faults in the Transverse ranges is ignored. Finally, creep processes assumed along the fault are simplified. The first assumption will have significant effects on the vertical deformations.

A recent model by Rundle(1979) considers the deformation as a consequence of a very low angle thrust fault. Again the model suffers from a number of significant defects: Numerical instabilities have been discovered in his solution method for an elastic layer over a viscoelastic half-space. For a low angle thrust these instabilities will be particularly severe. In addition, a viscoelastic half space underlying the elastic crust is a poor approximation; I will demonstrate this in a later section.

Although each of these solutions have specific defects, each mechanism may be significant for the final deformations in Southern California (Smith, 1977, 1978, 1979). As we have emphasized in our original proposals, the most probable mechanism generating the uplift is a combination of the following processes:

(1) Complex fault geometry represented by the San Andreas fault and branch faults.

(2) Stress relaxation in the asthenosphere originating from a low-viscosity channel.

(3) Coupling of aseismic slip between faults and fault segments.

The final mechanism adopts greater importance in light of Mavko and Stuart's (1979) solutions for coupling of aseismic creep between the San Andreas and Calaveras faults.

Assuming these rather complex interactions, our objective is to resolve their significance and develop a solution method which incorporates them. One very useful approach would be similar to Mavko and Stuart's (1979) for aseismic creep, but extend the solution to an elastic half-space in Southern California. The time scales in Southern California are, however, longer than Northern California which may invalidate the half-space assumption. Instead, a viscoelastic asthenosphere may be required to model the large scale deformations. This necessity will be shown in a later section. A series of faults within a plate overlying a viscoelastic half-space would be a better assumption. Two approaches are now obvious: One uses Rundle(1978) solution for a strike-slip fault within an elastic layer overlying a viscoelastic half-space; the other utilizes a finite element model.

To allow solution of a general problem, the finite element method was chosen for the time dependent problems. Our code will allow the following problems:

(1) General viscoelastic media for the model. The simplest properties, however, will be chosen for medium; an elastic lithosphere and a Maxwellian response for the shear modulus within the asthenosphere.

(2) Nonlinear constitutive relation along the fault. Laboratory evidences indicates that the response of a fault zone to stress may be very complex. The dependence probably involves the strain history and the confining pressure as important parameters. The finite element model will incorporate a general relation along the fault through convolutions of the step function response for each segment of the fault interface. This requires modelling the fault as an internal boundary.

(3) Complex fault configurations. As an example, the Southern California region involves a complex group of intersecting thrust and strike slip faults. This requires careful treatment of boundary conditions at the intersection of branch faults with the San Andreas system.

(4) Lateral heterogeneities within the crust. The complex geology in Southern California as reflected in the topography may be a significant element influencing the strain fields.

Our ability to incorporate each of these factors in the computation is an advantage; however, three dimensional

finite element models are notoriously complex and require extensive computer facilities. Consequently, there are advantages in the approach taken by Rundle, Mavko, and Stuart with analytical-numerical models. The finite element models cannot replace these, particularly for simple models of strike slip faults in an elastic layered media. For viscoelastic media and for complex geometries with lateral heterogeneities, the finite element method is the only suitable approach.

In the following sections I will outline the computer code for the finite element method and demonstrate its applicability to a problem of thrust faults. The solutions for surface deformations associated with the 1946 Nankaido earthquake will constrain the viscosity of the asthenosphere. This work is an extension of an earlier interpretation using additional data and models (Smith, 1974a,b, 1976). These results are then applied to a simple strike slip fault to indicate the contribution of stress relaxation to horizontal and vertical surface deformations.

2. VISCOSITY OF THE ASTHENOSPHERE

Large shallow earthquakes at island arcs such as Japan and Alaska are attributed to convergence and subduction of lithospheric plates. The majority of mechanisms fit neatly into the framework provided by new global tectonics: The earthquakes occur as a megathrust between the continental and underthrusting oceanic lithosphere (McKenzie & Parker, 1967; Isacks, et al, 1968; Plafker, 1972). The subduction process requires motion of the lithospheric plate over a viscous asthenosphere if strain accumulation is to occur at the subduction zone. In this section the interaction of the lithosphere and the asthenosphere will be constrained using geodetic data following major dip slip earthquakes. The resulting information, the effective viscosity of the asthenosphere, is a critical parameter for modelling Southern California. Unlike the conclusions of Thatcher and Rundle(1979), the models suggest that asthenospheric stress relaxation provides a simple explanation for the post-seismic deformations.

The sequence of crustal deformations associated with a major, shallow thrust earthquake such as the 1946 Nankaido earthquake have particular significance. These movements reduce to four stages: secular, pre-seismic, seismic, and post-seismic (Scholz, 1972). The secular strain accumulation and seismic deformation conform to the classical elastic rebound theory of Reid(1910). The rapid crustal

movements which characterize preseismic and post-seismic deformations offer the best constraints on faulting processes and interactions between the lithosphere and asthenosphere.

Post seismic deformations have either the same or the reverse sense of movement as the earthquake phase; the sense of movement depends on the particular earthquake (Scholz, 1972). The 1966 Parkfield earthquake appears as a buried fault: the post-earthquake movements may correspond to creep along the surface extension of a buried fault (Scholz, et al, 1969). The post-seismic movements would then have the same direction as the earthquake and approach its inferred offset. Thatcher (1975) suggests this behavior for the 1906 San Francisco earthquake except along deeper segments. Barker (1976) notes, however, that fault creep cannot be resolved from stress relaxation in the asthenosphere.

Another group of earthquakes do not as readily lend themselves to post-seismic fault creep; instead, asthenospheric stress relaxation provides a simple explanation. These earthquakes include the 1946 Nankaido earthquake illustrated in Figure 1, 1923 Kanto, the 1964 Niigata, and the 1964 Alaskan earthquakes (Brown, 1977). To resolve the contribution of asthensopheric stress relaxation to post-seismic deformation, the geodetic data from the 1946 Nankaido earthquake will constrain simple models of Southwest Japan. The model must contain the essential assumption, a

subducting lithosphere and stress relaxation in the asthenosphere, while ignoring unwarranted or unresolvable complications. The approach taken uses a linear viscoelastic medium for the asthenosphere and lithosphere. Initial elasticity and time-dependent relaxation are present, but it avoids further material and numerical complications inherent in nonlinear stress-strain relaxation. Finally, a novel approach in the Laplace domain using the finite element method results in a very tractable model for the time dependent problem.

The results indicate that asthenospheric stress relaxation will explain the post-seismic deformations and also provide details of the viscosity structure which are unresolved by the analysis of post-glacial rebound. These results differ significantly from those of Thatcher and Rundle (1979) as a result of their simplifying assumptions: no lateral heterogeneities and a viscoelastic half-space for the asthenosphere. Both depth and lateral variations in the time-dependent properties must be incorporated for modelling the deformations. This is not to underestimate the importance of fault creep, but to emphasize the need to incorporate realistic geometries and properties of the fault and medium.

Using the geodetic data from the 1946 Nankaido earthquake, the models place the following constraints on the lithosphere and asthenosphere:

(1) A 30 to 40 km thick elastic lithosphere.

(2) A thin, 20 km thick asthenosphere with an effective viscosity of 1 or 2 x 10**20 poise.

(3) A higher viscosity mesosphere with an approximate viscosity of 10**21 poise or greater.

(4) A zone or layer of low effective viscosity within the asthenosphere adjacent to the subducting slab. Within the zone the viscosity is less than 2 x 10**19 poise.

In the following sections I will discuss the approach to this problem and the data constraining each conclusion.

2.1. Variational Formulation and the Finite Element Method

The earth exhibits a complex structure along island arcs: a dipping fault plane often separating continental and oceanic lithospheres. This real earth domain can include geometrical and material inhomogeneities, and irregular boundaries. A solution strategy for the time-dependent strains from faulting must be capable of handling these problems. Analytical solutions for a fault are essentially limited to a half- space (Roseman and Singh, 1973a,b). Approximate solutions for a layered media are possible using the method of images and including only the first or higher order terms (Nur and Mavko, 1974; Rundle and Jackson, 1977). Hybrid analytical solutions with propagator matrices are

possible for a linear viscoelastic model, but are also limited to a layered media (Barker, 1976). This may be suitable for a strike-slip fault, however, island arcs do not correspond to a layered media. Numerical solutions appear as the only technique that holds any promise for even a simple subducting lithosphere. Two general strategies immediately suggest themselves: finite element method and finite differences.

The finite element method solves the variational problem, that is, minimizing an energy functional or similar expression over subregions and then combining into a banded symmetrical matrix. The engineering sciences have provided the impetus for its development, for it is ideally suited to elliptical boundary value problems as encountered in elasticity theory (Zienkiewicz, 1977). For these problems the method is fast and has flexible resolution. But time dependence introduces problems. One could discretize in time and solve; however, this normally requires a two-point boundary value problem. Time represents an initial value domain. It is at this stage that finite differences is usually introduced together with all its difficulties.

The finite difference technique has been the usual solution strategy for time dependent problems (Richtmeyer and Morton, 1967). It is also notorious for its idiosyncrasies: slow and unstable convergence, and difficulties at discontinuities. If we use an integral technique for

generating the difference equations (Smith and Toksoz, 1972), and then step in time, one does better. Yet this is basically equivalent to a finite element solution in space and finite differences in time (e.g. Zienkiewicz, et al, 1968; Bischke, 1974). This is better than finite differences in time and space, but not by much considering the bookkeeping needed during the solution.

An alternate method uses an approximate Laplace transform inversion and applies the finite element method to the Laplace domain of the variational-evolutionary principle. First developed by Schapery (1962) for the Rayleigh-Ritz method, the technique immediately limits one to linear viscoelasticity, actually a very desirable assumption. Adey and Brebbia (1973) suggested its extension to the finite element method since the strategy overcomes the problems of stability and error propagation found in the finite difference technique. Indeed, it is applicable to any problem formulated as a Stieljes integral having a well-behaved kernal.

Appendix A contains a complete description of the approach and the development of the evolutionary principle. The solution method allows the generation of a series of "Green's" functions for each segment of the fault. This, however, imposes certain constraints on the solution method. These problems and alternative approaches are outlined in the appendix A, and the computer code is given in appendix

C.

2.2. Post-Seismic Deformation for the 1946 Nankaido Earthquake

Characteristics of the 1946 earthquake have been extensively discussed in Fitch and Scholz (1972), Smith (1974), and Thatcher and Kundle (1979). In the context of this report, I would like to emphasize certain aspects of the data set which have significant consequences for the mode of post-seismic deformation occurring in Southwest Japan. Special attention will be directed towards the following problems: wavelength and locations of post-seismic deformation; time scale of post-seismic deformation; time scale for tilts; and migration of maximum post-seismic uplift.

Figure 2 illustrates the post-seismic deformations from 1946 to 1960 on the island of Shikoku. The leveling data was supplied by Ando (1976) and tied to the mean sea-level changes occurring at the island (Tsumura, 1970). The results allow a satisfactory correction for oceanographic effects; consequently, the hinge line is constrained within a few centimeters. The form of the deformation allows an approximate two-dimensional model. Important constraints are offered by the location of the hinge line, position and width (70 km) of maximum uplift, and extent of subsidence far from the fault. The uplift of 20 to 30 cm occurs during a period of 13 years. The comparison of this leveling

data with similar data from 1964 to 1971 will yield additional constraints.

The vertical movements in Shikoku given in Figure 3 demonstrate a continuation of the post-seismic deformation. Again the leveling data is tied to the mean sea level changes (Crustal Activity Research Office, 1972). The deformation rate is now slower and the location of maximum uplift has migrated north by 10 or 20 km. The hinge line may also have a slight northward migration. The width of maximum deformation remains approximately 70 km. Figure 4 illustrates both the vertical movements of Shikoku and adjacent Honshu during the same time interval, 1964 to 1971. The pattern of deformation is repeated on the Kii peninsula which is also adjacent to the Nanki trough. Again subsidence occurs inland from the hinge line.

The previous diagrams confirm post-seismic deformation for at least 20 years; however, additional constraints are necessary to resolve the time scale. Tilt data can provide this information. Tilting at Muroto peninsula in figure 5 has commonly provided one such data set (Okada and Nagata, 1953). Using a series of leveling lines at the promotory, the tilts suggest very rapid tilting towards the trough. The decay time is in the order of one year. This is actually a rather crucial set of data for many investigators. Both Fitch and Scholz (1972), and Thatcher and Rundle (1979) argue in favor of fault creep as the responsible mechanism.

It is not possible to satisfy the tilt data and the previous vertical movements of Shikoku using a simple model of relaxation in the asthenosphere (Smith, 1974, 1976). Additional information is need to constrain the time scale for the tilts across Shikoku; Figure 6 gives this important information.

The figure illustrates tilts from three leveling lines across Shikoku and compares them to a water-tube tiltmeter at Matsuyama (Earthquake Research Institute, 1975). Each set of observations fits a similar decay rate for the tilts, approximately 10 years. This rate is an order of magnitude longer than the decay at Muroto peninsula and, consequently, suggests two separate mechanisms for the deformations. The next section will propose a solution.

2.3. Models for Island Arcs

Specific structural features characterize an island arc system: The features are dominated by a thrust fault separating the descending oceanic lithosphere from the slalnd arc or continental plate. A distinct low velocity and low Q zone underlies both the oceanic plate and island arc (Utsu, 1967). These are also the essential features that dominate the finite element models.

Figure 7 illustrates the finite element mesh together with these essential regions. The dominant, time-dependent interaction is one between the asthenosphere and

lithosphere; however, other factors are important: gravity perturbs the time-dependent deformations; and the length of the descending lithosphere influences possible modes of deformation. Even more significant are the thicknesses of the lithosphere and asthenosphere, and the relative dimensions of the fault.

The simplest model and the one most closely approaching the analytical solutions use a single elastic layer overlying a Maxwellian viscoelastic half-space. There has been recent confusion, however, on the appropriate parameters to use for the viscoelasticity. Thatcher and Rundle(1979) assume elastic behavior for lambda and Maxwellian behavior for the shear modulus. The bulk modulus will then be a standard solid with relaxation in volume. Figure 8 compares this assumption to a pure elastic bulk modulus. The resulting deformations are significantly different, particularly if one is comparing the model to observations at some distance from the fault zone.

Comparing the finite element solution in Figure 8b to Thatcher and Rundle's (1979) analytical solution in figure 9 indicates a discrepancy. Near the fault zone numerical instabilities are apparent for the post-seismic deformations computed with Rundle's analytical method; the oscillations are particularly strong when the bulk modulus is assumed to be elastic. These oscillations are not present in the finite element solutions. In addition, the finite element

method has been verified in Appendix B using analytical solutions. Finally, models with different grid structures and decay times all yield consistent results for equivalent problems. Clearly, a numerical problem exists in their approximations; its origin lies in the assumptions for the derivation of the "Green's" function (Rundle, 1978). Lambda is required to be constant in all the layers. Consequently, a viscoelastic shear modulus requires the bulk modulus k to be viscoelastic. Assuming an elastic bulk modulus will give the numerical instability and the incorrect solution. In addition, the image method gives an incorrect boundary condition at the interface between the layers; stress relaxation will again accentuate this discrepancy. Incorrect solutions will result for both strike-slip faults and thrust faults. Unless another derivation of the Green's function is used, this problem cannot be corrected.

There is little evidence to suggest viscoelastic behavior for the bulk modulus. Significant relaxation has only been observed for the shear modulus in laboratory experiments. Other phenomena such as dilatancy have been modelled using relaxation of the bulk modulus; however, Thatcher and Rundle (1979) did not assume this process or justify the magnitude of relaxation. Consequently, our models will assume an elastic behavior for the bulk modulus (Smith, 1974).

Another discrepancy is the relaxation time adopted by Thatcher and Rundle (1979). First, they define the relaxation time tau = 2 / where is the viscosity of the asthenosphere, and represents the shear modulus for a Maxwellian model. This represents the relaxation time for an elastic layer overlying a viscoelastic half-space, not the asthenosphere itself (see McConnell, 1965). For Maxwellian response in shear, the correct relaxation time is tau = / (Christensen, 1971, using equations 1.32 and 1.41). Both approaches, however, give the same viscosity for the asthenosphere since our tau = 4. would be equivalent to time 2. in Thatcher and Rundle (1979).

Figure 8 also illustrates the importance of the fault depth D to lithospheric thickness H. When the fault extends through only part of the lithosphere, the post-seismic deformation near the fault is subsidence, while at greater distances uplift occurs. However, when the fault fully extends thru the lithosphere as in Figure 10, the region of uplift dominates, and the pattern of deformation becomes strongly assymmetrical. These general features have also been observed for earthquakes. Both the 1964 Niigata earthquake and the 1923 Kanto earthquake are characterized by a zone of post-seismic subsidence near the fault and little uplift at greater distances. For at least the 1964 Niigata earthquake, this is consistence with the faulting mechanism; a shallow thrust fault extending to perhaps 10 or 20 km

depth. On the other hand, the 1946 earthquake is dominated by post-seismic uplift similar to figure 10a. The non-dimensional parameter, D/H, primarily controls then the shape of the profile. This confirms the rough analytical models of Nur and Mavko (1974) and the solutions of Thatcher and Rundle (1979).

Gravity represents an additional perturbation to the surface deformations. Figure 11 from Smith (1974) compares two finite elment models, one with gravity and one without, when the gravitational potential has been introduced into the variational principle according to Appendix A. The deformations are for two times, zero or elastic and tau = 12.6 after the dislocation. The gravitational effect is evident for the post seismic phase and only slight for the elastic deformations. When the asthenosphere relaxes, the gravitational restoring force becomes relatively more important; the asthenosphere approaches a buoyant fluid. The general form of deformation remains the same; however, accurate comparison to geodetic data requires the inclusion of the gravitational potential.

2.4. Viscosity Structure

Analytical models of Thatcher and Rundle (1979) assume a viscoelastic half-space underlying the lithosphere; however, both the vertical and lateral viscosity structure are crucial for the deformations (Smith, 1974). Island arcs

represent a gross deviation from a simple elastic layer over a half-space: the subducting slab and regions of high seismic attenuation suggest a complex structure. An example of the possible consequences is illustrated in Figure 12. Figures 12a and b demonstrate deviations from a viscoelastic half-space. Both contain a low viscoity channel underlying the lithosphere. For the thicker channel of 50 km in 12b, we now observe subsidence at distances greater than 200 km from the fault instead of further uplift as in figure 10. This is a crucial difference. When a thinner 25 km channel is introduced as in 12a, the width of uplift substantially decreases. The thickness of the lithosphere and the relative thickness of the low-viscosity channel determines the minimum wavelength as in the problem of post-glacial rebound (McConnell, 1965, figure 6) and the relative thickness of the low-viscosity channel.

If a descending slab is included in the model, further deviations occur in the vertical deformation. The hinge point shifts slightly towards the fault while a tail develops in the region of uplift. The width of maximum uplift, however, remains about the same with only a nominal decrease in amplitude. Yet these deviations are important in the final models.

The volcanic zones in island arcs and models of convection suggest the existence of partial melt near the fault zone within the asthenosphere. A region of low viscosity

will effect the resulting surface deformations (Wahr and Wyss, 1979); however, the effect of the asthenosphere must still be included in the models. Figure 12d combines both a low-viscosity channel and a zone of lower viscosity near the fault. The results are very significant: Near the fault rapid deformation corresponds to relaxation in the low-viscosity zone; a broad region of uplift continues by relaxation in the asthenosphere; and finally migration of the hinge line and maximum uplift occurs in response to their coupling. These are the essential characteristics observed in the tilts and deformations of the 1946 Nankaido earthquake.

2.5. A Model for the 1946 Nankaido Earthquake

A synthesis of the previous models suggests a hypothesis for the origin of the post-seismic deformations. A channel of low effective viscosity with an inclusion of still lower effective viscosity near the fault. Since the zone is near the fault tip, its low effective viscosity may originate from a power law relation between effective viscosity and stress (eg. Slade, et al, 1979). The data is unable to resolve the distinction.

Figure 13 summarizes the approximate deformations associated with the complete model. These will serve as an example of scaling. The model uses the same 30 degree dip for the slap; tsunami data suggests that 20 to 25 degrees

may be more appropriate (Ando, 1975). These results are not signficantly effected by a small change in dip. In order to fit the narrow width of vertical uplift, a thin asthenosphere is required. The width also places constraints on the corresponding thickness of the lithosphere. The time scales of post-seismic deformation require inclusion of secular strain accumulation which is illustrated in figure 12c. This is modelled using slow reverse creep along the fault; the rate corresponds to the observed strain accumulation. Figure 12c corresponds to a recurrence interval of tau = 20. for time nondimensionalized by the asthenospheric relaxation time. This secular deformation is then added to the post-seismic movements to give figures 12a,b, and d.

With secular accumulation the post-seismic uplift spans a width equal to approximately twice the lithospheric thickness. When only a low-viscosity channel is present as in figure 13a and b, the characteristic time constant for uplift is approximately 2 to 3 times the relaxation time of the asthenosphere. Neither the hinge line nor the location of maximum uplift migrates. Using only one effective viscosity, simple relaxation cannot explain the observations.

Inclusion of a low-viscosity zone near the fault tip introduces an additional relaxation time. In figure 13d the effective viscosity of the zone is one-fifth of the asthenospheric relaxation time. The resulting tilts near

the fault occur with this characteristic relaxation time, while at greater distances asthenospheric relaxation dominates the vertical movements. The relative contribution of each depends on the proportion of the asthenosphere occupied by the low-viscosity zone. In addition, the maximum uplift and hinge line migrate away from the fault. This model contains the features required by the post-seismic vertical movements; our problem is now to scale these general models to our specific case.

The wavelength of the deformation gives the best constraint to the thickness of the lithosphere. Using 70 km as the observed width of postseismic uplift implies a thickness of 35 km for the lithosphere and approximately 15 to 20 km for the low-viscosity channel or asthenosphere. Scaling from the 1 meter dislocations in the model to an average slip of 3 meters suggested for the 1946 Nankaido earthquake (Ando, 1976; Thatcher and Rundle, 1979) scales the amplitude of the uplift for each dimensionless time. From figure 13d, 20 to 30 cm of maximum uplift would occur at tau = 2.; this uplift occurs within the interval of 14 to 18 years implying a relaxation time of 8 years and an effective viscosity of 1.5 x 10**20 poise. The resulting tilts across Shikoku are similar to the previous observations in figure 6.

The zone of lower effective viscosity within the asthenosphere accounts for the rapid tilts at the Muroto promitory. Relaxation in this region generates a narrow

zone of rapid uplift and tilting. The uplift is analogous to an extension of the fault into the asthenosphere; consequently, the length of the zone along the fault determines the details of the uplift. For example, at a distance of 150 km in figure 13d, the model predicts initial postseismic subsidence followed by uplift. This has been observed at tidal stations (Tsumura, 1971). To reproduce the tilts at Muroto requires an effective viscosity less than 2.x10**19 poise. With this scaling the model faithfully duplicates the characteristics of the observed postseimic and secular deformations for the 1946 earthquake.

3. APPLICATIONS TO SOUTHERN CALIFORNIA TECTONICS

The previous sections have emphasized the importance of the viscosity structure as a function of depth, lateral heterogeneities on the deformations, and the fault geometry. These factors also have significance for modelling the deformation mechanisms in southern California. Hadley and Kanamori's (1977) analysis of crustal structure in the Transverse Ranges suggest a complex structure. Models of the deformation have either assumed an elastic crustal and asthenospheric structure (Kosloff, 1977), or a viscoelastic half-space underlying a simple elastic layer (Rundle, 1979). The previous sections in this report suggest that neither model is adequate to represent the vertical and horizontal deformations. In addition, Rundle's (1979) model of the Palmdale bulge as a low angle thrust fault suffers from numerical errors and a viscoelastic bulk modulus; the errors will be particularly severe for a low-angle thrust fault. The computational method outlined in this report can treat these problems with the addition of fault creep; its primary difficulty is adequate computer resources.

To differentiate the effect of mantle relaxation from fault creep and to demonstrate the utility of the computer code for three-dimensional problems, we developed a simple model of a strike slip fault within a layered media. Similar problems have been studied by other investigators (Savage and Prescott, 1978; Barker, 1976). Under specific

circumstances, they conclude that stress relaxation may be important during the earthquake cycle. In our computations, asthenospheric stress relaxation cannot account for the deformations occurring in southern California; complex fault geometries and fault creep are also necessary.

A surface view of the 3D finite element mesh is illustrated in figure 14. The thickness of the lithosphere and asthenosphere are 50 and 100 km, respectively. A 5 meter, strike-slip dislocation is applied to the fault through the thickness of the lithosphere. The length of the fault is approximately 4 times its width. The fault is free to slip in the vertical (Z) direction in response to strains introduced by the dislocation strike-slip dislocation. In analytical models this feature is difficult to achieve. Although a simple model, it is adequate to define the general deformations along the fault.

The vertical deformations are illustrated in figure 15 for the post-seismic and seismic deformations. For the vertical seismic deformations in figure 15b, the maximum uplift and subsidence is in the order of 50 cm near the fault tip. The large vertical movement is in response to the stress concentration at the crack tip and the vertical free slip condition on the fault plane. The top figure, 15a, illustrates the post-seismic movement at tau = 2. when the vertical free slip condition is retained along the fault. This is not a realistic assumption, but it will not

dramatically alter the results. Further uplift and subsidence occurs near the tip; however, it now extends over a much broader region. The maximum post-seismic movement is approximately 15 centimeters.

The next figure illustrates post-seismic movements on an exagerated scale. At time tau = 1. in figure 16, one again observes large vertical offsets near the fault tip. This occurs in response to stress relaxation in the asthenosphere at the crack tip. In addition, uplift and subsidence occur at distances corresponding to a few lithospheric thicknesses H. The maximum uplift is approximately 10 to 15 cm.

In southern California adjacent to the San Andreas fault, the elastic thickness of the lithosphere may be as little as 20 to 30 km. This is not an unwarranted assumption considering the recent tectonics and the previous results for Japan. Scaling the problem in figure 16 to a lithospheric thickness of 20 km instead of 50 km, the fault length is now 80 to 100 km and the thickness of the asthenosphere is 40 km. If we conservatively assume an effective viscosity of 10**20 poise for the asthenosphere, figure 16 corresponds to approximately 5 years after the earthquake. For slip of 2 meters along the fault, the maximum vertical deformation would then be 8 cm. If this slip occurs as buried creep along the fault, the broad pattern of post-seismic deformation would be similar. We might infer,

then, that slip dislocation or creep along a simple strike-slip fault gives insufficient vertical movement to explain the large scale vertical movements in southern California.

These models , however, suggest alternate approaches. A bend in the fault will generate additional concentrations of strain (Kosloff,1977); relaxation will be greatst in these zones. Complex fault geometries also concentrate strain and deformation (Kosloff, 1977) ; we would also expect post-seismic deformation to concentrate in these localities. Unfortunately, for these complex geometries it is not clear how the post-seismic deformation will evolve when couple to creep instabilites along the fault. We are now constructing a model of southern California in order to allow direct comparison to observed geodetic data.

4. STOCHASTIC MODELLING

Earthquake migration patterns have been observed by many researchers (eg. Mogi, 1968; Kelleher and others, 1970 to 1976), but their observations have always been qualitative. This, coupled with the apparent randomness of the data, often make their interpretations questionable; the direction of the migration depending in part on the bias of the researcher (Kelleher, 1972; Delsemme and Smith, 1979).

In the previous grant, we developed techniques to quantitatively establish the migration pattern of large earthquakes. These techniques consist of (1) a new sampling procedure (see Appendix I), (2) two-dimensional spectral analysis, and (3) a new stacking method to reinforce the common features before large earthquakes (see Appendix II). The spectral analysis was applied to both the space-time diagrams and to the stacked patterns.

In the last six months we considered spectral analysis in more detail in order to obtain estimates of the confidence bands around the computed spectra. In addition, we started preliminary work on the stochastic analysis of larger earthquakes using the Box-Jenkins methodology.

4.1. Estimating confidence bands

Confidence bands would be easy to compute, were it not for the finiteness of the fourier transform and the smooth-

ing requirements (Rayner, 1971; Beauchamp and Yuen, 1979).

A finite length of data in the time domain can be interpreted as looking at the entire data, both past and future, through a limited window. This window is rectangular since it begins abruptly at the beginning and ends abruptly too at the end of the recorded data. Unfortunately a rectangular window becomes a sinc function

$$(\sin \pi t / \pi t)$$

when transformed into the frequency domain. This means that each computed frequency is the convolution of the "true" transform and the sinc function.

The sinc function is not a very desirable function since it is highly oscillatory and converges slowly. Its one advantage is a very sharp central peak. What is needed is a window in the time domain that would transform into the frequency domain with a sharp central peak but with no oscillatory sidelobes. Actual desirable windows will be compromises between these two features. Since the oscillatory sidelobes are due to the sharp discontinuities at each end of the data, better windows are produced by tapering the ends of the data toward zero. There are many ways to produce this effect (e.g. Bartlett, Parzen, Hamming,...) but only one is in very common use: the Hanning window. It has a fairly narrow central peak with small oscillatory sidelobes converging rapidly. Another reason why it is popular, is its ease of computation in both the time and

frequency domain. This means that the window can be applied either before or after calculating the fourier transform of the data. This first window will reduce the leakage caused by the sampling and the finite length of the data.

In addition to this, it is desirable to average values in the computed spectrum in order to reduce the variance, even though it entails a decrease in resolution. The weighting function used to smooth the spectrum is also called a window. This second window has a bandwidth describing the amount of averaging it effects on the spectrum. If we have a data series of length L, the frequency interval will be 1/L. A window with bandwidth B will average over

$$B / (1/L) = BL$$

values. 2BL is then the number of independent values entering in the final estimate of the computed spectrum. The factor 2 is caused by the complex nature of the transform, i.e. two numbers (a real and an imaginary) are used to compute the spectrum. The product 2BL is called the number of degrees of freedom in the computed spectrum.

The concept of a bandwidth is difficult to define quantitatively. For a detailed description, see Parzen (1961). Some examples of bandwidth for some common windows are given in Beauchamp and Yuen (1979).

Now we can estimate the confidence bands around the computed spectrum. The fourier transform of the original data is the sum of a large number of random variables (i.e. the sum of x(t) exp(iwt) over all t) therefore this sum will be approximately Gaussian, as can be proved with the central limit theorem. The final computed spectrum is the sum of square of 2BL values. Recall that Chi-square(n) is defined as the sum of square of n independent Gaussian variables each with zero mean and unit variance. Therefore the final computed spectrum will behave as a Chi-square variable with 2BL degrees of freedom. This distribution can be found in practically any introductory statistics textbook.

4.2. Stochastic Modelling Using Box-Jenkins

Earthquakes have a strong stochastic nature, otherwise the forecasting problem would have been resolved long ago (Lomnitz, 1966). The observed earthquake sequences can be considered as a realization of some geophysical process. From this realization one can deduce the basic properties of the stochastic process such as means, moments,... But the major purpose in time series analysis is to construct molel(s) that have properties similar to that of the geophysical process generating the earthquakes.

Box and Jenkins present a logical method to do just this (Anderson, 1975; Box and Jenkins, 1976; Robinson and Silvia, 1979). It is an iterative approach to model build-

ing: First, a general class of model is chosen to fit the problem at hand. Second, an initial model is chosen from the general class. This is done using autocorrelation and partial-autocorrelation functions, to determine the smallest possible number of parameters for an adequate representation (parsimony principle). Third, the model is fitted to the data and the parameters are estimated. Fourth, the residuals between the model and the real time series are computed and tested to discover possible lack of fit. This diagnostic test either accepts or rejects the model. If rejected, the steps are iterated until a suitable model is found. If accepted, forecasting becomes possible.

So far no mention has been made of the kind of models to consider for geophysical data. Box and Jenkins limit themselves to stationary processes. According to Wold (1954), stationary processes can be decomposed into an autoregressive term and a moving average term. This decomposition theorem suggests that little is gained by considering nonlinear models, provided that the series is indeed stationary.

For earthquakes it is likely that stationary is satisfied, since the driving force of plates is probably constant over short geological time periods (Kagan and Knopoff, 1976). But even if some series are non-stationary, it is possible to transform them by repeatedly differencing them until they do become stationary.

Earthquakes may be thought of as a point process (Snyder, 1975). In order to use the Box-Jenkins method, the point process must be converted to a time series. One way to do it, is to use the sampling procedure we developed earlier for the two-dimensional spectral analysis (see Appendix I for more details). The purpose of the sampling was to filter out the high frequencies in order to be able to use a limited number of samples. Since this restriction is not directly applicable to the Box-Jenkins method, a simpler sampling procedure may be used.

Rather than dealing with each earthquake individually, the Box-Jenkins method excels at modelling statistical processes, so that it becomes more efficient to group earthquakes in larger samples. We are investigating three related series: (1) the number of earthquakes above a certain magnitude, (2) the energy released by these earthquakes, and (3) the equivalent magnitude corresponding to the energy released by the earthquakes. Present work suggest sampling intervals of the order of months to years.

To accomplish the goals described above, we developed an extensive computer code. About two-thirds of the basic routines for the univariate case have been written and tested. Some of the routines follow Box and Jenkins' own algorithms as described in their books: Time series analysis: forecasting and control (1976). Five main programs are required.

(1) Earthquake sampling routine (ESAMP). This routine converts an earthquake catalog into suitable time series. The list of earthquakes can be edited in terms of geographic locations, magnitude range, and depth range. The sampled time series is stored on disk (or on tape kt1) for later input.

(2) Univariate stochastic model identification (USID). This routine accepts the data from routine ESAMP. It transforms and differences the data if desired. It computes means, variances, autocovariances, autocorrelations, and partial-autocorrelations. The program plots all needed results on the line printer. By examining the autocorrelations and partial-autocorrelations, a class of models is chosen. In addition, the routine displays histograms of the time series to check their gaussian distribution. The autocovariances are stored on disk (or on tape kt2) for later processing.

(3) Univariate stochastic model preliminary estimation (USPE). This routine accepts the autocovariances from routine USID, and from them, computes preliminary estimates of the parameters for the class of models selected.

(4) Univariate stochastic model estimation (USES). The initial parameters computed by routine USPE may be

refined using a non-linear least-square optimization method, due to Marquardt (1963). Diagnotic checks are also furnished to determine the appropriateness of the model chosen. This routine has not yet been implemented.

(5) Univariate stochastic model forecasting (USFO). Once the least-square estimates of the parameters of the model have been found, and that the model has been checked as appropriate using routine USES, forecasting is possible. The routine inputs the sampled time series from ESAMP, finds the generalized ARMA parameters, then forecast the series from any origin. The time series, with the forecast and its upper and lower probability limits are plotted on the line printer. The results are also stored on disk (or on tape kt3) for later plotting on an X-Y plotter.

All routines have the option of seasonal differencing if needed. That option has not yet been tested. Some of the work still to be done includes the non-linear least-square algorithm, and the plotting routines needed to drive the Tektronix X-Y plotter. A non-linear least-square FORTRAN program exists in Bevington (1969) and could probably be adopted. No difficulties are anticipated for the plotting routines due to our familiarity with the plotter.

Once the programs have been completed and tested, univariate forecasting will be possible. We expect that univariate forecasting will not be entirely adequate to predict earthquakes directly. However it may be appropriate for forecasting the amount of energy released, or the number of earthquakes in a given year.

This tool can be used for many other geophysical time series. By fitting stochastic models to series known only from their observations, it will be possible to compare the observations to the forecast values. Anomalies will stand out as being unpredictable, and those anomalous precursory events can then be described in terms of the probability limit of the stochastic process.

4.3. Appendix I: Sampling technique

Two-dimensional fourier transforms require a lot of samples since the number of samples varies with the square of the number of samples on one side. Therefore in order to be efficient, the number of samples must be minimized.

Sampling earthquakes, however present special problems. Earthquakes are a point process, with each point representing the time and epicenter of one event. Each event is a spike with essentially instantaneous rise time. A spike contains a uniform frequency distribution; however, a bandlimited signal is needed to avoid aliasing.

To solve these difficulties, we have followed a technique described first by French and Holden (1971). Their method involves several steps which are to be simultaneously executed. Although they describe the solution for a one-dimensional case, their method can be extended to more than one dimension. In two-dimensions, the steps are: (1) Truncate the data to a finite record. (2) Convolve each earthquake or spike with amplitude A with a function that is an ideal low-pass filter (Brigham, 1974):

$$A^2 \frac{\sin(\pi t / dt)}{\pi t} \frac{\sin(\pi x / dx)}{\pi x}$$

where t denotes time and x represents space with respective Nyquist frequencies of 1/2dt and 1/2dx. (3) Truncate the data again with the same window used in step (1). (4) Sample the data with sampling rate of 1/dt and 1/dx. (5) Remove the mean to prevent the appearance of a DC spike in the final spectrum.

In addition to these steps, the space-time diagrams are extended with zeroes in order to increase the resolution of the final spectra. This is done prior to step (2) to avoid truncation of the ideal low-pass filter and, thereby, minimize bias. A Hanning split-cosine bell taper is then applied at the edges of the sampling grid to reduce leakage, as described earlier.

4.4. Appendix II: Stacking technique

In order to improve the signal to noise ratio of the two-dimensional spectra, we developed a stacking technique. Each earthquake has a unique history of earthquakes preceding it. The previous earthquakes can be described using the earthquake as origin. In two-dimensions we would consider (1) the time intervals and (2) the distances between the previous earthquakes and the earthquake. We can plot the pattern before each earthquake on the same diagram, using the same origin. The relative time intervals and distances are then distributed in a similar manner, and any common features will appear as clusters of points.

Instead of considering just one realization of the stochastic process leading to an earthquake, we are considering several realizations. These realizations are grouped into one diagram so that it becomes possible to describe the ensemble average of the stochastic process.

Since we are dealing with a historical record of finite length, care must be taken in selecting the earthquakes to be stacked. Each of them must have the same record length. For example, in order to examine 20 year long patterns in a data set starting in 1897, the first stackable earthquake occurs in 1918.

When considering two-dimensions, the space-axis extending along a seismic belt will also be finite. In this case

edge effects are unavoidable. But if the density of earthquakes is constant along the seismic belt, these effects are symmetrical and will be equivalent at each end of the space-axis, thereby cancelling each other.

FIGURE CAPTIONS

FIGURE 1

Vertical movements across Shikoku. The profiles are from Fitch and Scholz (1971). Distance is measured from their inferred position of the fault within the Nanki trough.

(1) The top figure illustrates the secular deformations prior to the 1946 Nankaido earthquake. Notice the position of the hinge line.

(2) Seismic deformation associated with earthquake.

(3) Post-seismic deformation until 1964. Maximum uplift occurs in the region of seismic subsidence.

(4) Sum of seismic and post-seismic movement.

FIGURE 2

Vertical post-seismic movements on Shikoku from 1946 to 1960 from Ando (1976). The levelling data is tied to the variations in mean sea level (Tsumura, 1971). The resulting levelling data and tidal datum at stations around Shikoku are consistent to within approximately 5 cm.

FIGURE 3

Vertical movements in Shikoku from 1964 to 1971. The leveling data is tied to mean sea level changes (Crustal Activity Research Office, 1972).

FIGURE 4

Vertical movements in Shikoku and adjacent Honshu from approximately 1964 to 1971. (Crustal Activity Research Office, 1973).

FIGURE 5

Tilting at Muroto promitory. Notice the secular deformation prior to the earthquake in 1946 and the rapid postseismic movement following the coseismic tilt. The decay time is in the order of one year. (Okada and Nagata, 1953; Fitch and Scholz, 1971)

FIGURE 6

Tilts across Shikoku using leveling data with a comparison to the water tube tilt meter. The shape of the curves are similar and suggest a decay time in the order of 10 years. (Earthquake Research Institute, 1975)

FIGURE 7

Finite element mesh used for figures 8, 10, 12, and 13. Small modifications adapt the mesh to a simple half-space solution, a layered structure, or a subducting slab.

FIGURE 8

Vertical post-seismic deformation for fault dipping 30 degrees and D/H=.75 (fault extending through 3/4 of lithosphere). The thickness of the lithosphere is arbitrarily taken as 50 km, and overlies a Maxwellian viscoelastic half-space. The relaxation time for the shear modulus is nondimensionalized to one within the half-space. The inset gives the configuration of the lithosphere and fault using the horizontal distance scale. The intersection of the fault with the surface corresponds to zero distance if aligned with the distance scale. Fault slip is one meter.

(1) Model for an elastic bulk modulus k.

(2) Deformations when lambda is elastic and the bulk modulus is viscoelastic. This case corresponds to figure 1 in Thatcher and Rundle (1979).

FIGURE 9

Figure 1a from Thatcher and Rundle (1979). Coseismic vertical displacement (t=0, solid line) and subsequent viscoelastic response (t=2*tau); dotted line, compressibility of half-space held constant; dashed line, lambda fixed) due to slip on a fault rupturing three quarters of an elastic plate of thickness H overlying a viscoelastic (Maxell) half-space. Vertical uplift is normalized by the reverse fault slip, and distance perpendicular to the fault, y, is in multiples of the lithospheric thickness H. Fault dip is 30 degrees.

FIGURE 10

Similar to figure 8, except the fault fully extends through the lithosphere.

(1) Deformations for an elastic bulk modulus.

(2) Lambda held fixed and the bulk modulus becomes viscoelastic.

FIGURE 11

Perturbation introduced by gravity on the vertical surface deformations. The geometry is again shown in the upper left inset using the bottom distance scale. The fault slip is now 10 m with linear decay between the filled circles.

The descending lithosphere penetrates to 600 km and dips at 45 degrees. The comparison is for two nondimensional times, 0. and 12.6. The solid line represents the solution with gravity; the dashed line is the solution without gravity. Thus gravity introduces an additional restoring force, thereby reducing the post-seismic deformations.

FIGURE 12

Effects of viscosity structure on post-seismic deformation.

(1) top left (a). A thin low viscosity channel overlying a higher viscosity half-space with relaxation time 10.

(2) bottom left (b). A thicker low viscosity channel, otherwise similar to the previous.

(3) top right (c). A short subducting slab with a thin low-viscosity channel.

(4) bottom right (d). Similar to previous but with zone of lower viscosity within channel (darkened region). Viscosity is one-fifth of channels (relaxation time is .2).

FIGURE 13

Post-seismic and coseismic vertical deformations when secular deformation is included. The models correspond to a thin low-viscosity channel and a subducting slab. The relaxation times are nondimensionalized by the relaxation time of the channel.

(1) top left (a). Post-seismic vertical deformation with the secular deformation in (3) or (c).

(2) bottom left (b). Coseismic plus post-seismic deformation.

(3) top right (c). Secular deformation using model in (a) and assuming recurrence interval as 20 relaxation times. Deformations along upthrown side (oceanic) are not modelled properly using normal slip on fault.

(4) bottom right (d). Post-seismic deformation for zone of lower viscosity (one-fifth) embedded in the channel. Rapid relaxation causes the uplift to migrate from the fault.

FIGURE 14

Surface view of 3D finite element mesh. A strike-slip fault is located in the center and extends through the lithosphere. The thickness of the lithosphere is 50 km and it overlies a 100 km thick asthenosphere. A 5 meter dislo-

cation is applied to the fault. Element numbers are also shown.

FIGURE 15

Vertical deformations for seismic (bottom figure) and post-seismic (top figure) for the previous model. The vertical scale is in centimeters while the horizontal scale is in kilometers. The post-seismic plot is made at tau = 2.0.

FIGURE 16

Post-seismic deformations for model in figure 14 at tau = 1.0. Vertical scale is again in centimeters.

Figure 2

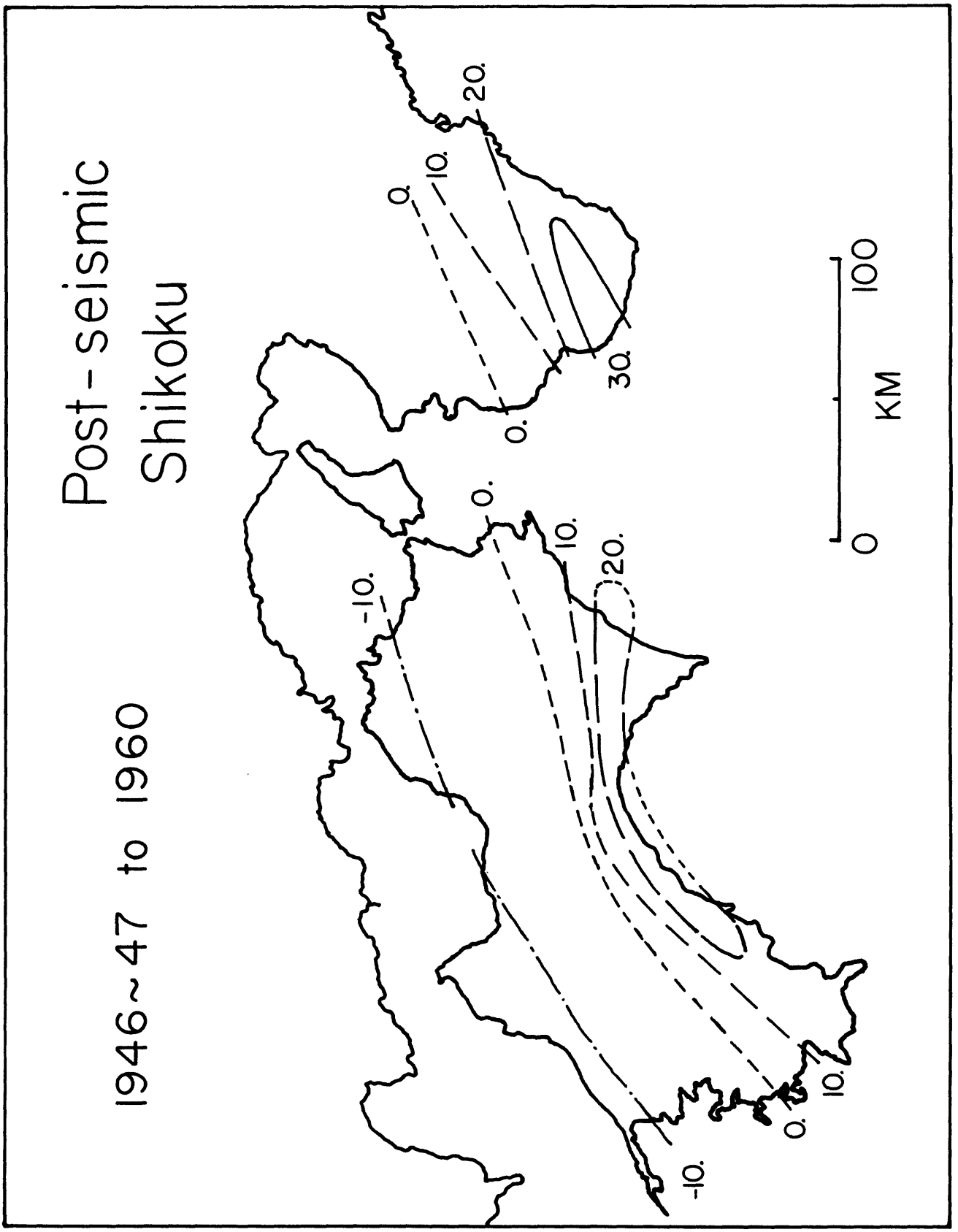
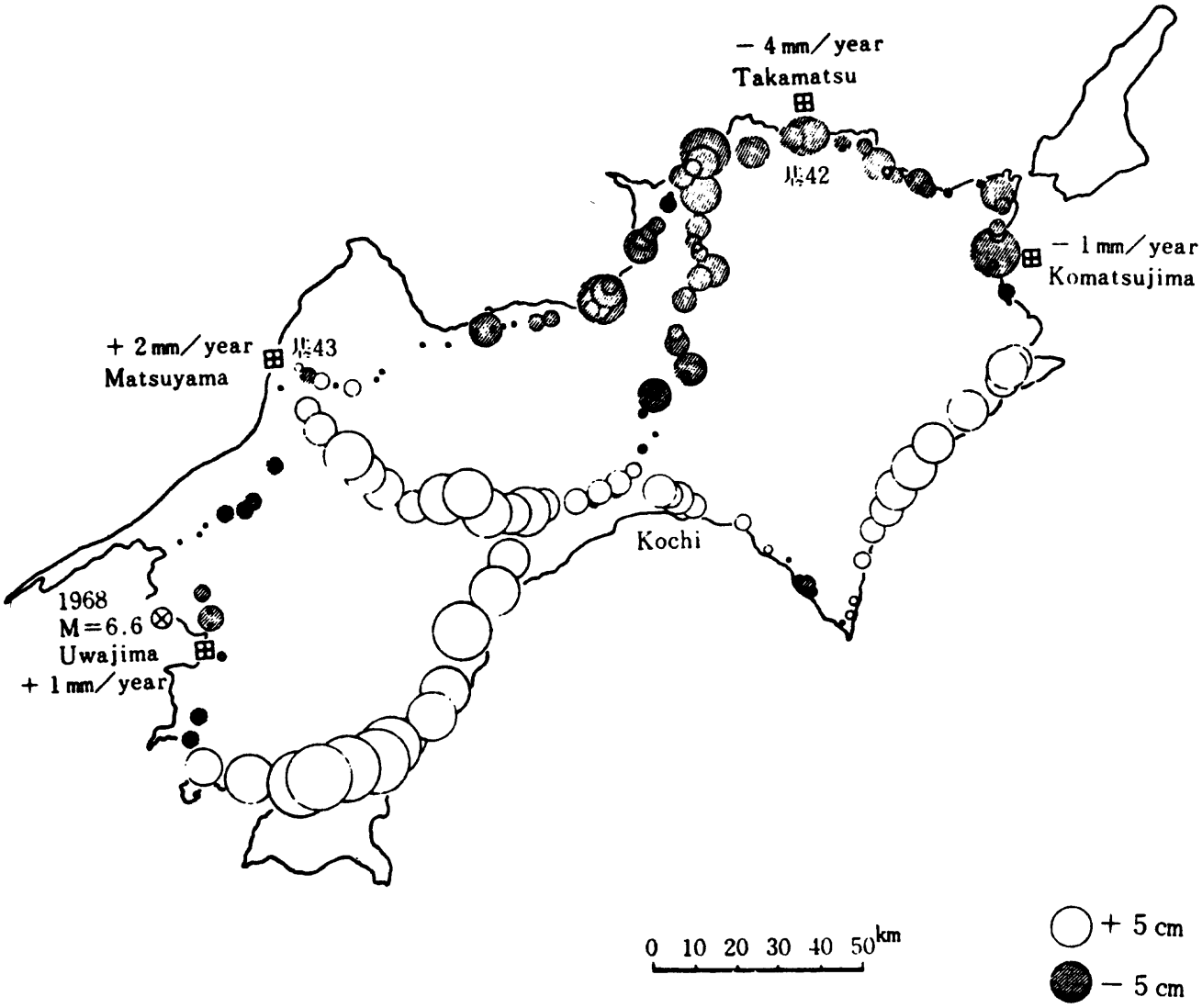


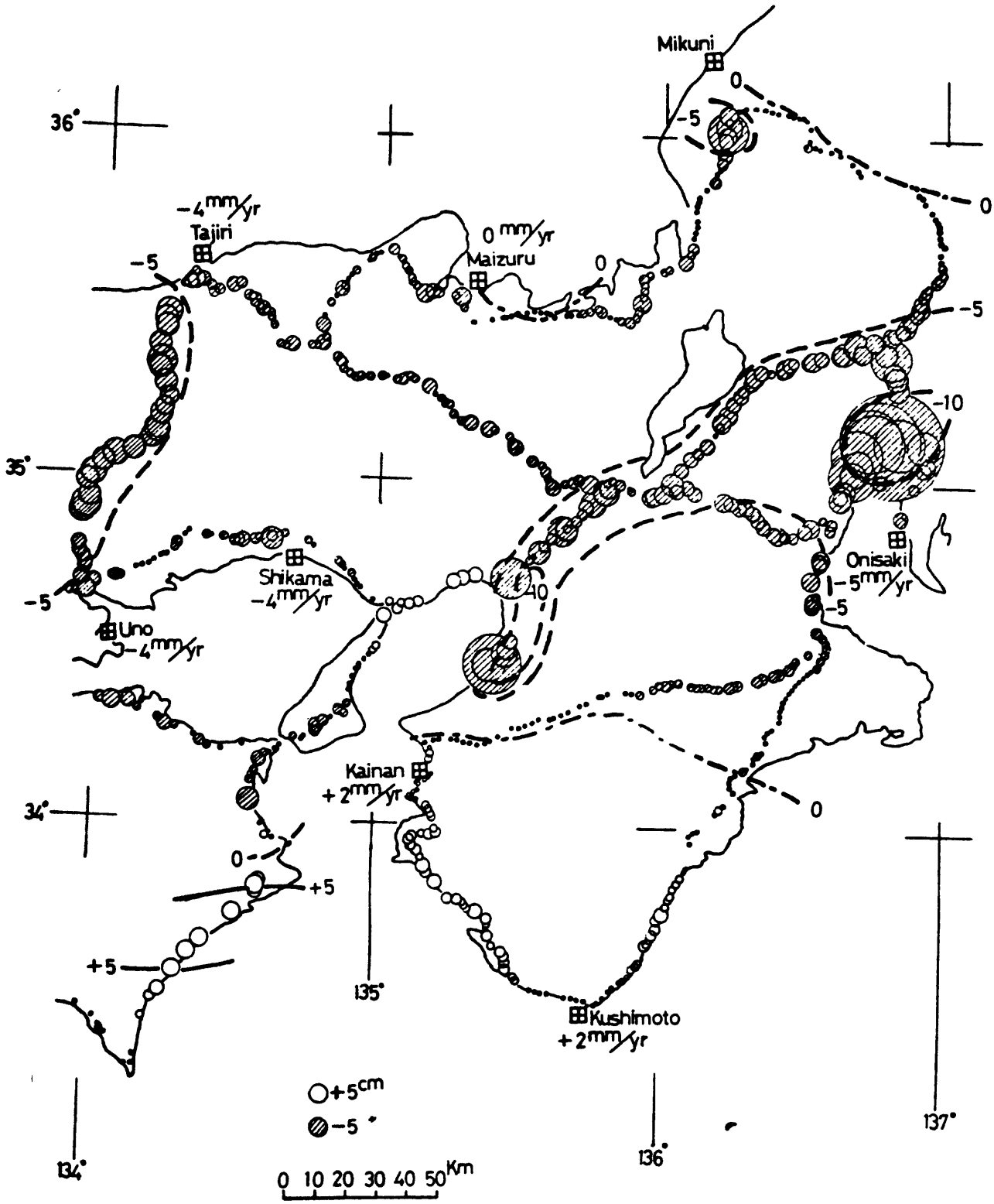
Figure 3



第1図　四国地方の上下変動（ 1964～1971 ）

Fig. 1　Vertical movements in Shikoku district (1964 - 1971)

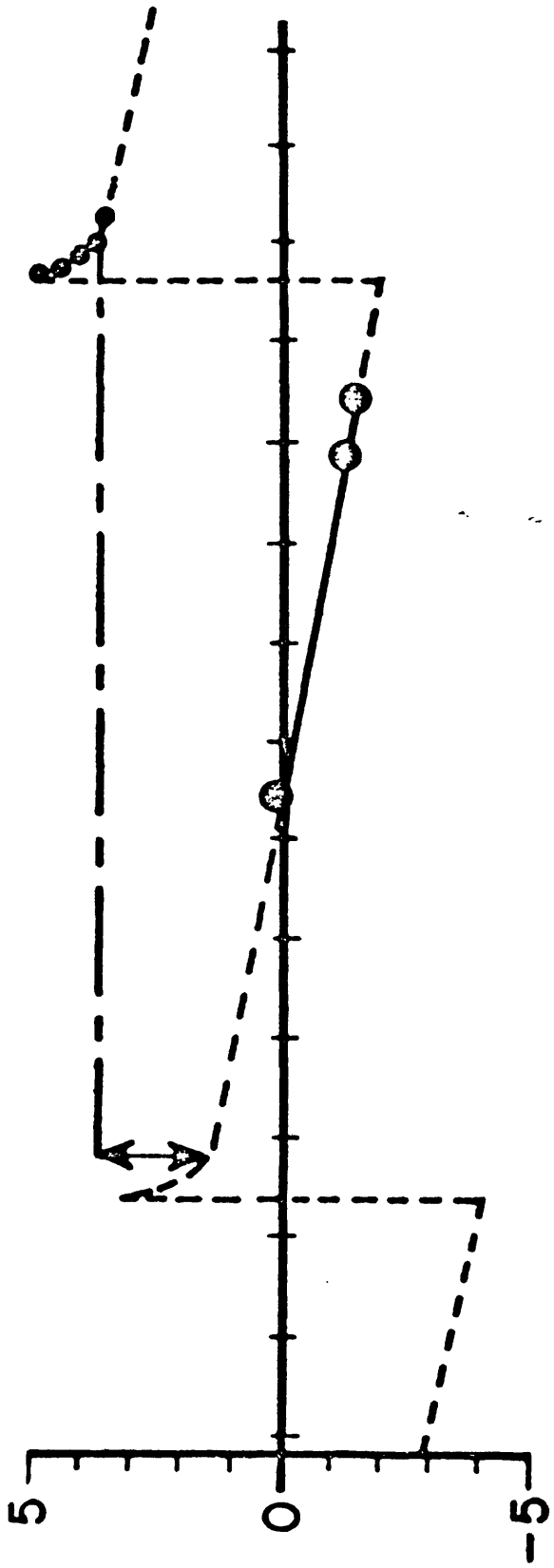
Figure 4

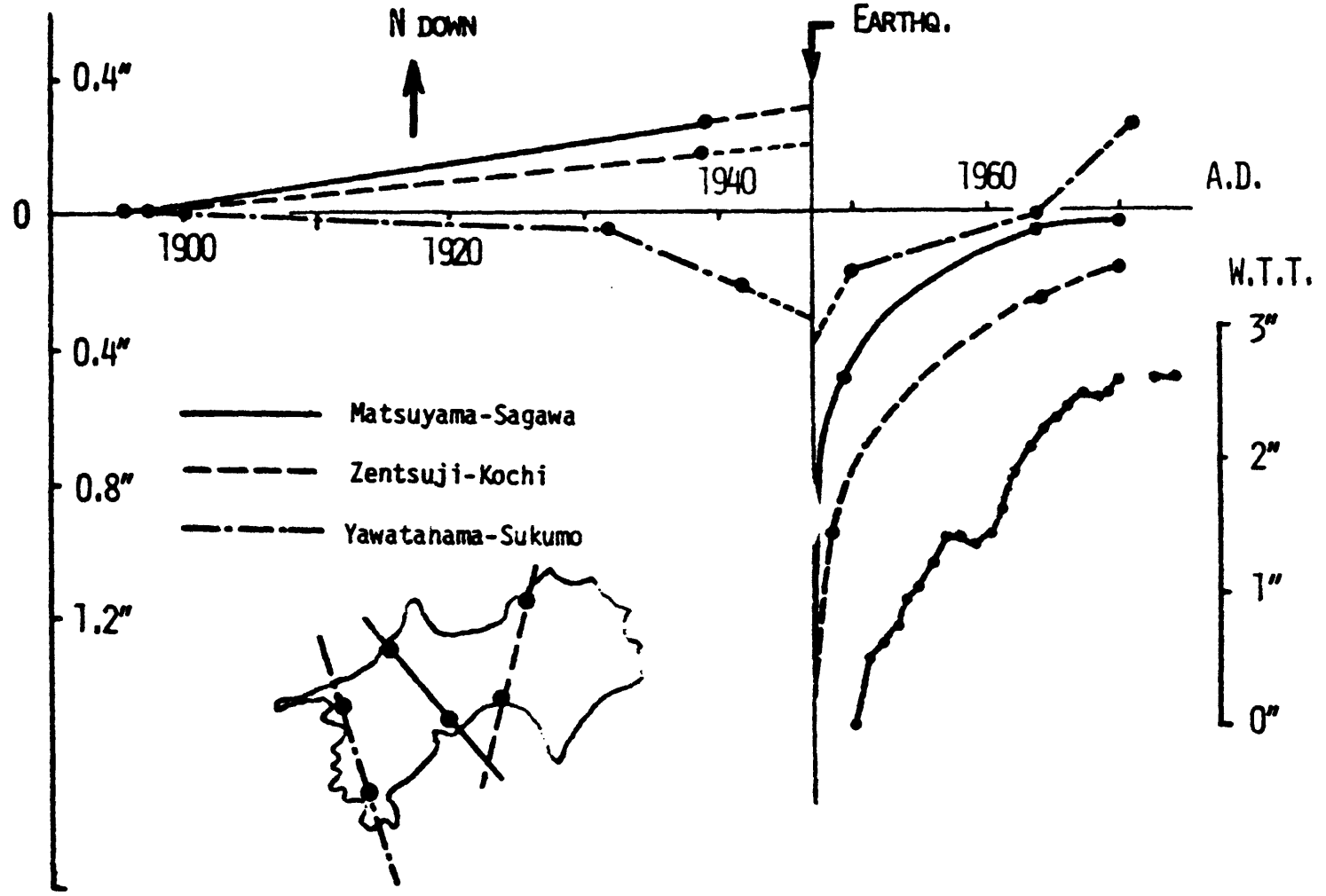


第3図　近畿地方の上下変動（1964～68－1970～72）

Fig. 3　Vertical movements in Kinki district (1964-68 - 1970-72)

Figure 5





第3図 水準測量による四国西部の広域傾斜変動と比較した松山における水管傾斜計観測結果。

Fig. 3 Vertical land movement in the western part of Shikoku as compared with the water-tube tiltmeter observations at Matsuyama.

Figure 6

Figure 7

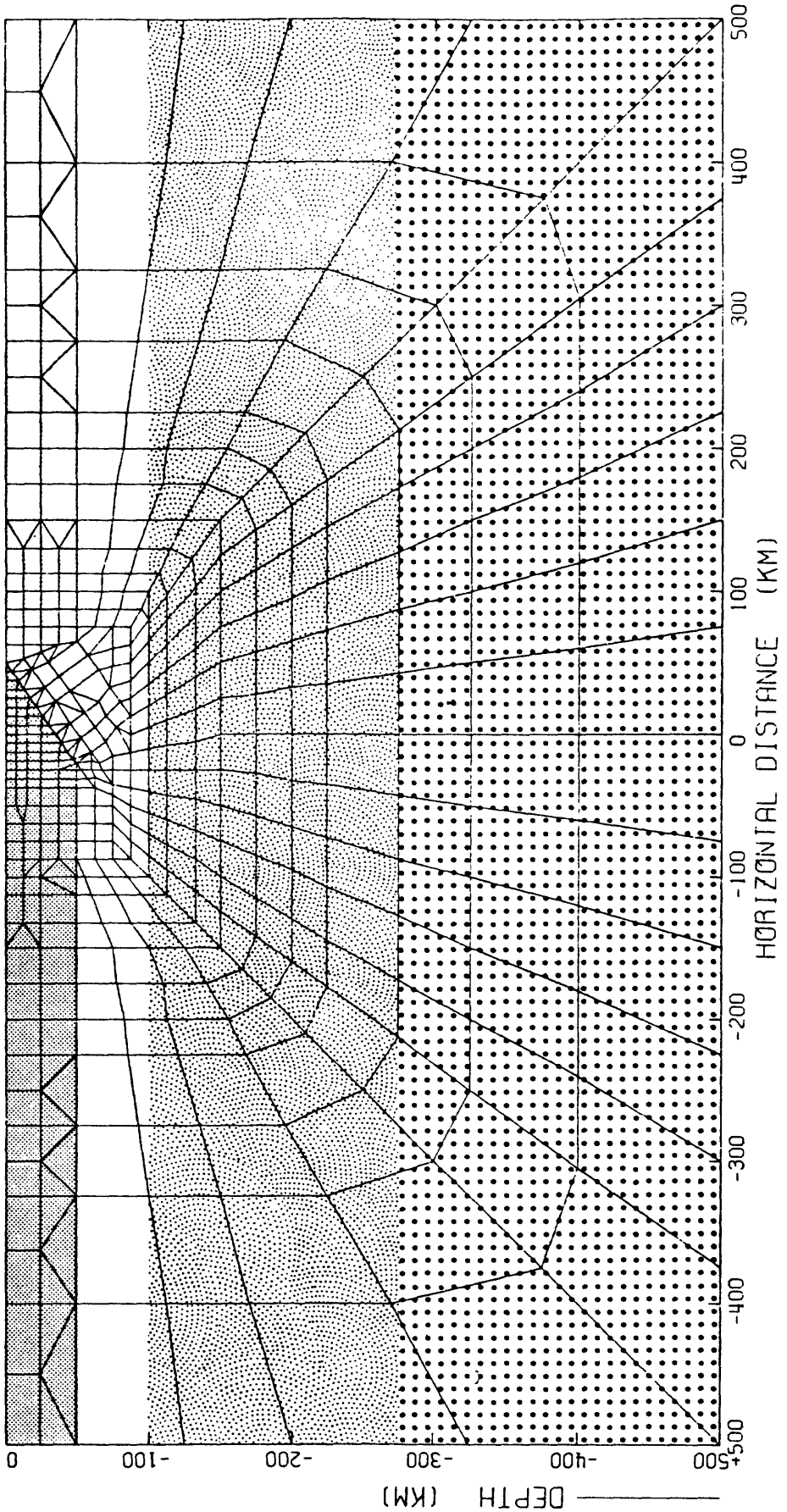
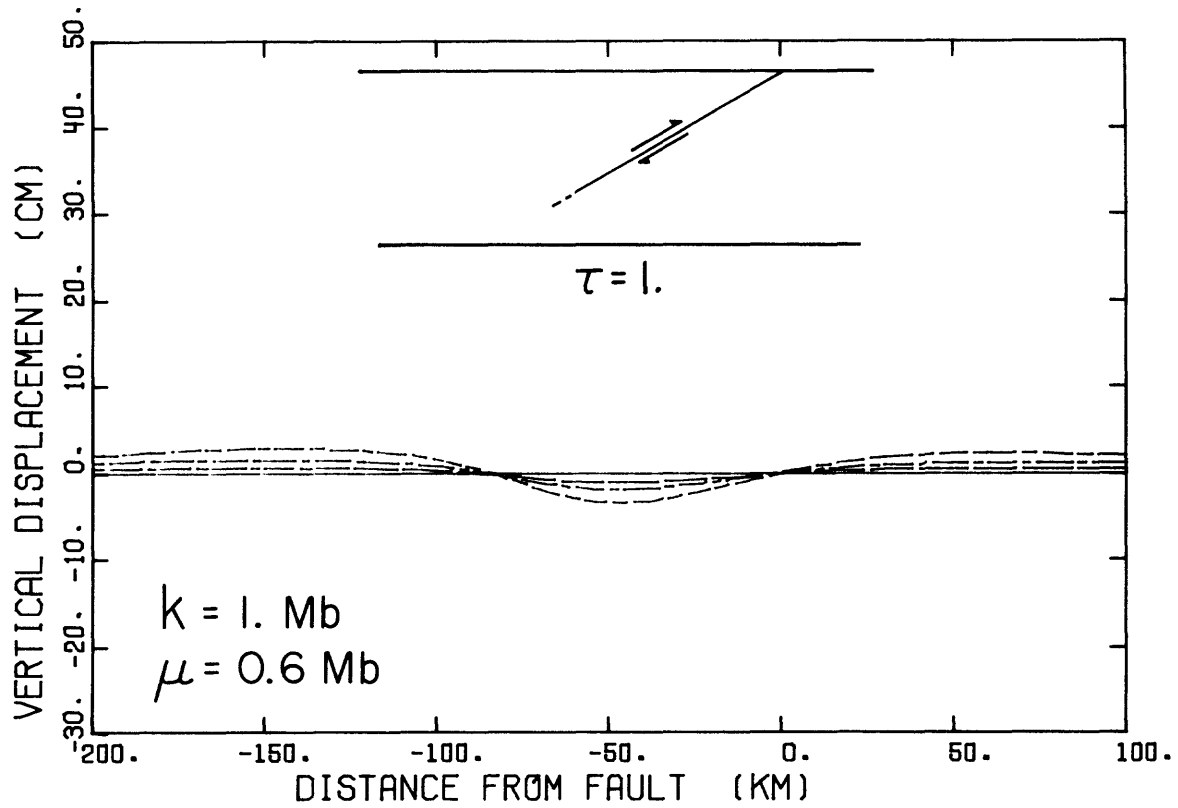

HORIZONTAL DISTANCE (KM)
500
400
300
200
100
0
-100
-200
-300
-400
-500
DEPTH (KM)
0
-100
-200
-300
-400
-500

Figure 8


τ = 1.
k = 1. Mb
μ = 0.6 Mb
VERTICAL DISPLACEMENT (CM)
50.
40.
30.
20.
10.
0.
-10.
-20.
-30.
-200.
-150.
-100.
-50.
0.
50.
100.
DISTANCE FROM FAULT (KM)

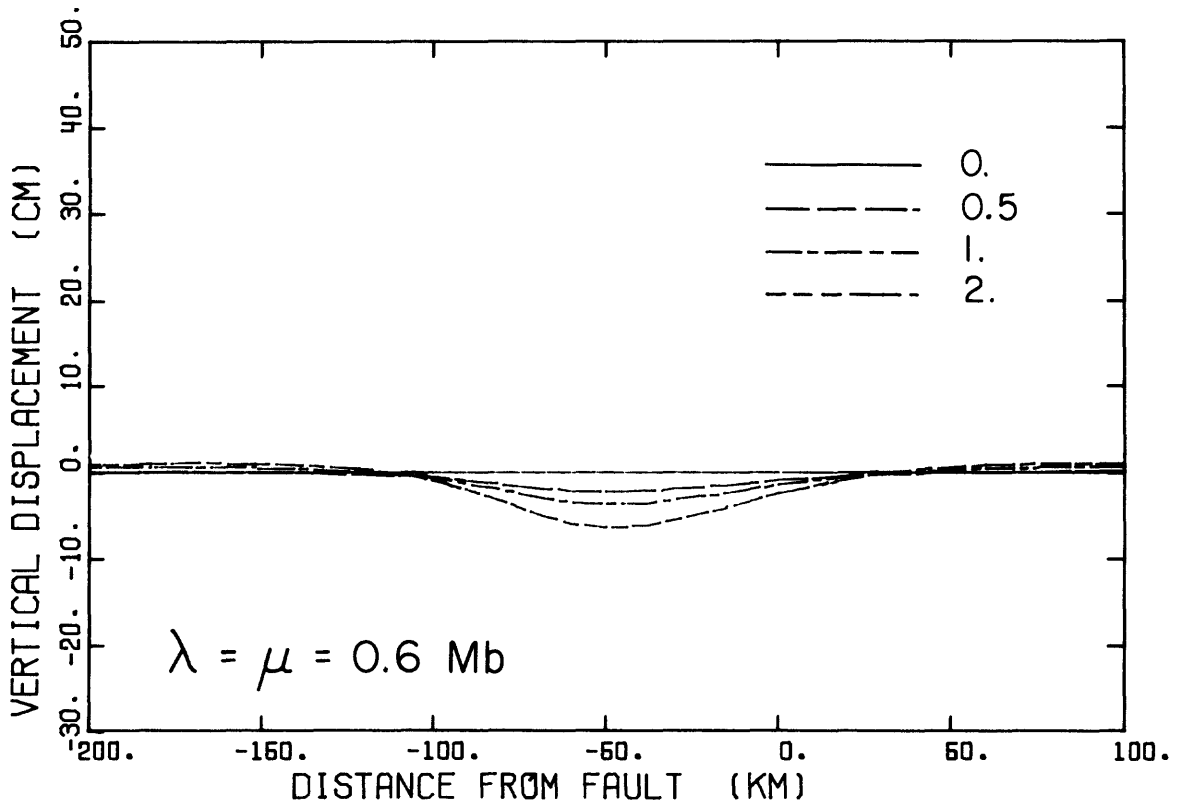

0.
0.5
1.
2.
λ = μ = 0.6 Mb
VERTICAL DISPLACEMENT (CM)
50.
40.
30.
20.
10.
0.
-10.
-20.
-30.
-200.
-150.
-100.
-50.
0.
50.
100.
DISTANCE FROM FAULT (KM)

Figure 9

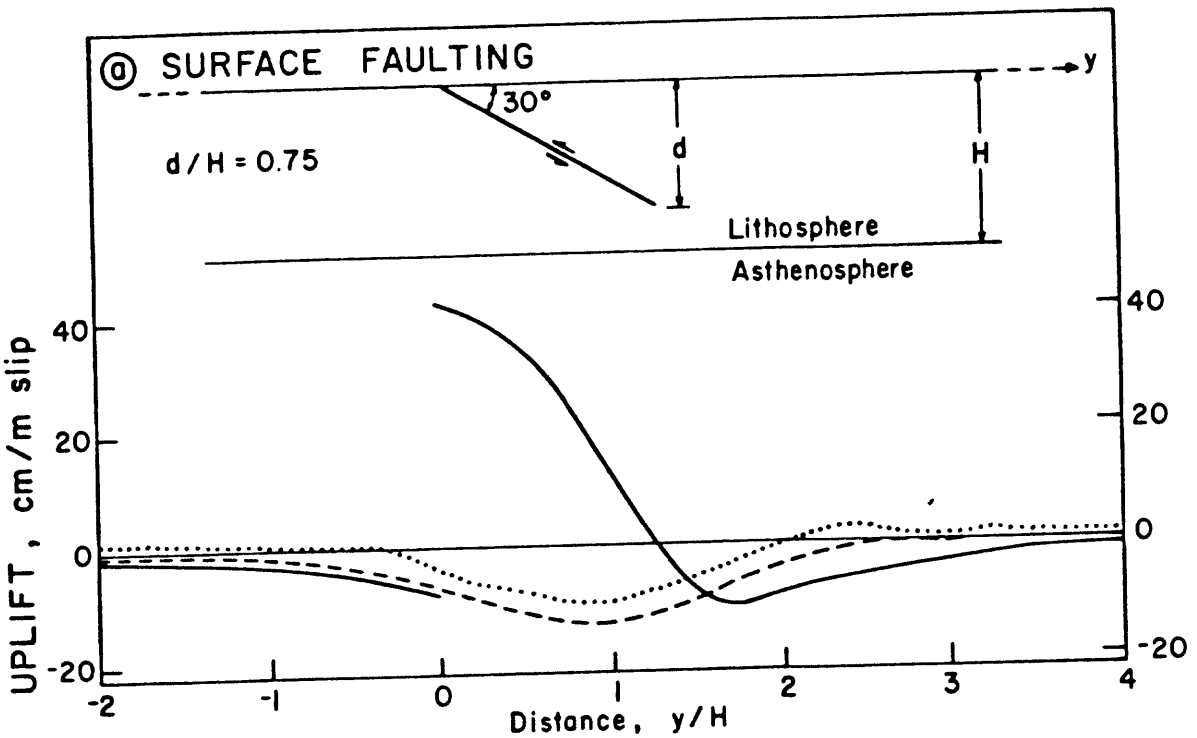
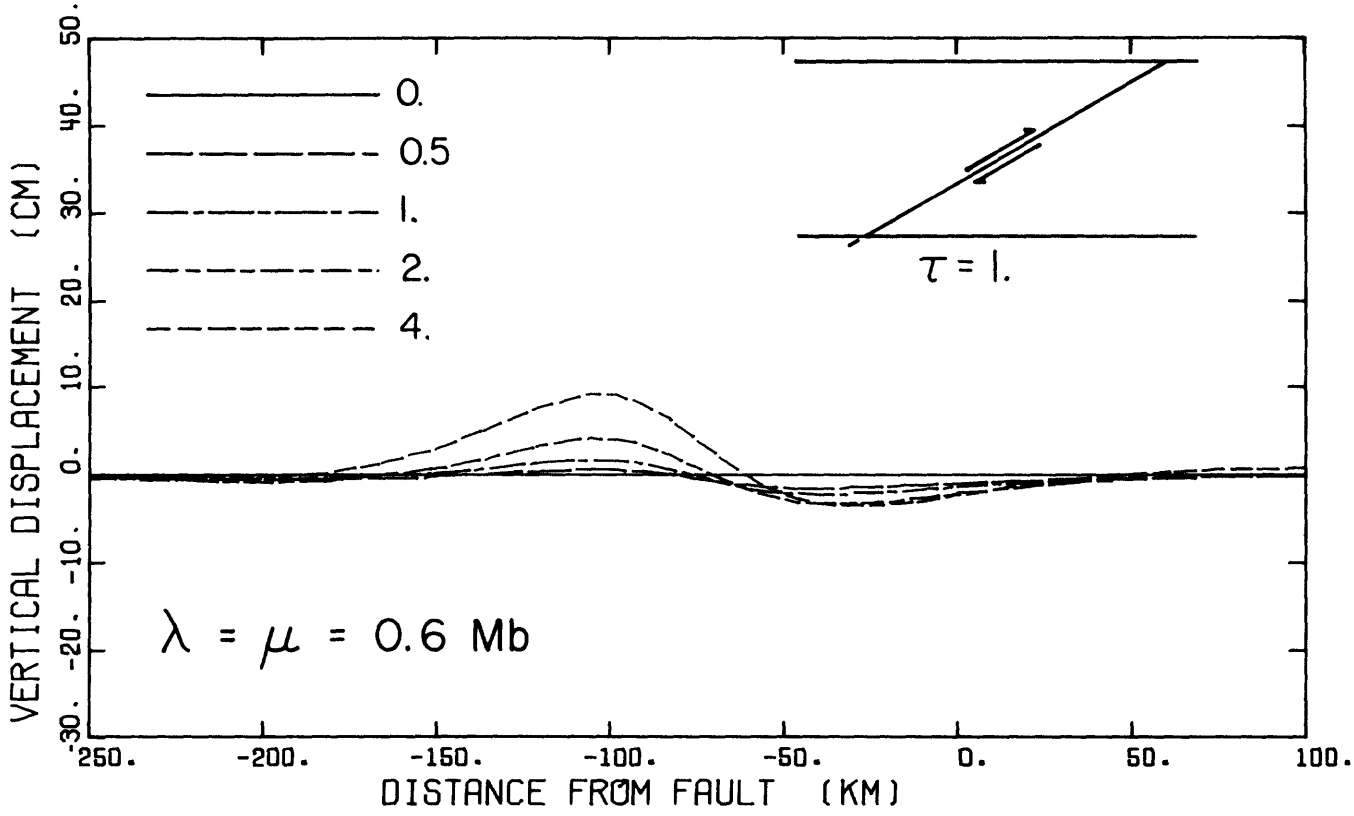

@ SURFACE FAULTING
30°
y
d/H = 0.75
d
H
Lithosphere
Asthenosphere
UPLIFT, cm/m slip
40
20
0
-20
-2
-1
0
1
2
3
4
Distance, y/H

Figure 10


0.
0.5
1.
2.
4.
τ = 1.
λ = μ = 0.6 Mb
VERTICAL DISPLACEMENT (CM)
50.
40.
30.
20.
10.
0.
-10.
-20.
-30.
-250.
-200.
-150.
-100.
-50.
0.
50.
100.
DISTANCE FROM FAULT (KM)

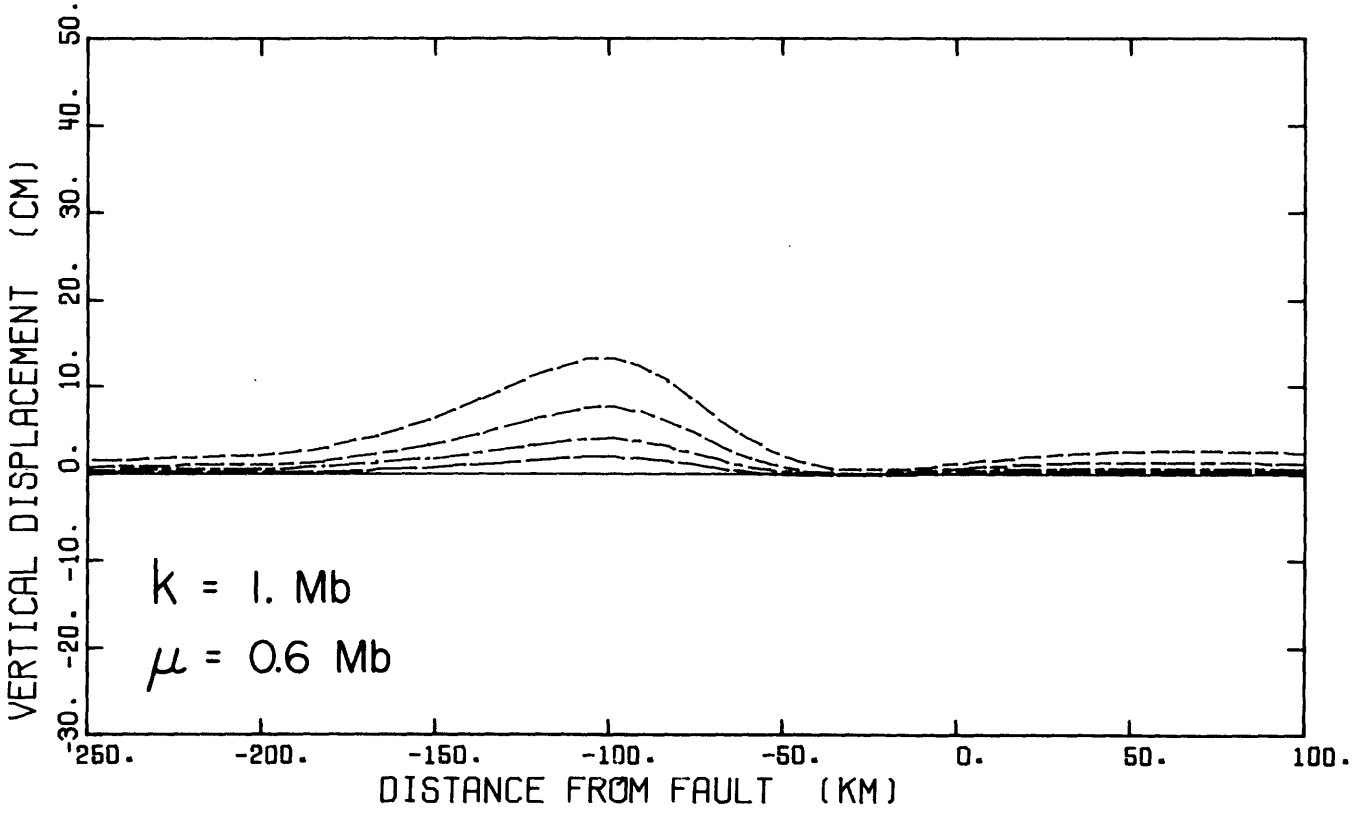

k = 1. Mb
μ = 0.6 Mb
VERTICAL DISPLACEMENT (CM)
50.
40.
30.
20.
10.
0.
-10.
-20.
-30.
-250.
-200.
-150.
-100.
-50.
0.
50.
100.
DISTANCE FROM FAULT (KM)

Figure 11

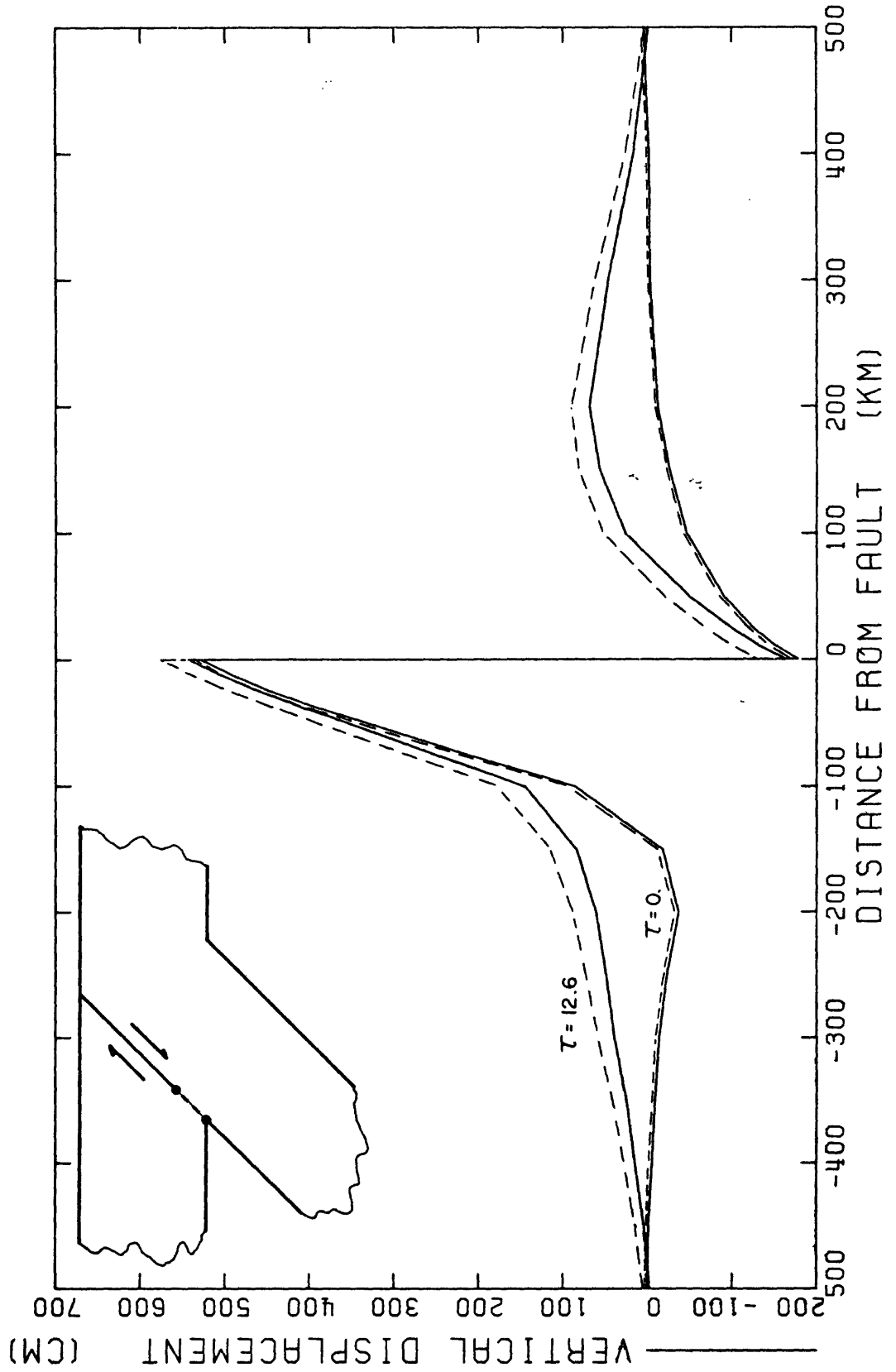

VERTICAL DISPLACEMENT (CM)
-200 -100 0 100 200 300 400 500 600 700
DISTANCE FROM FAULT (KM)
500 400 300 200 100 0 -100 -200 -300 -400 -500
τ = 12.6
τ = 0.

Figure 12

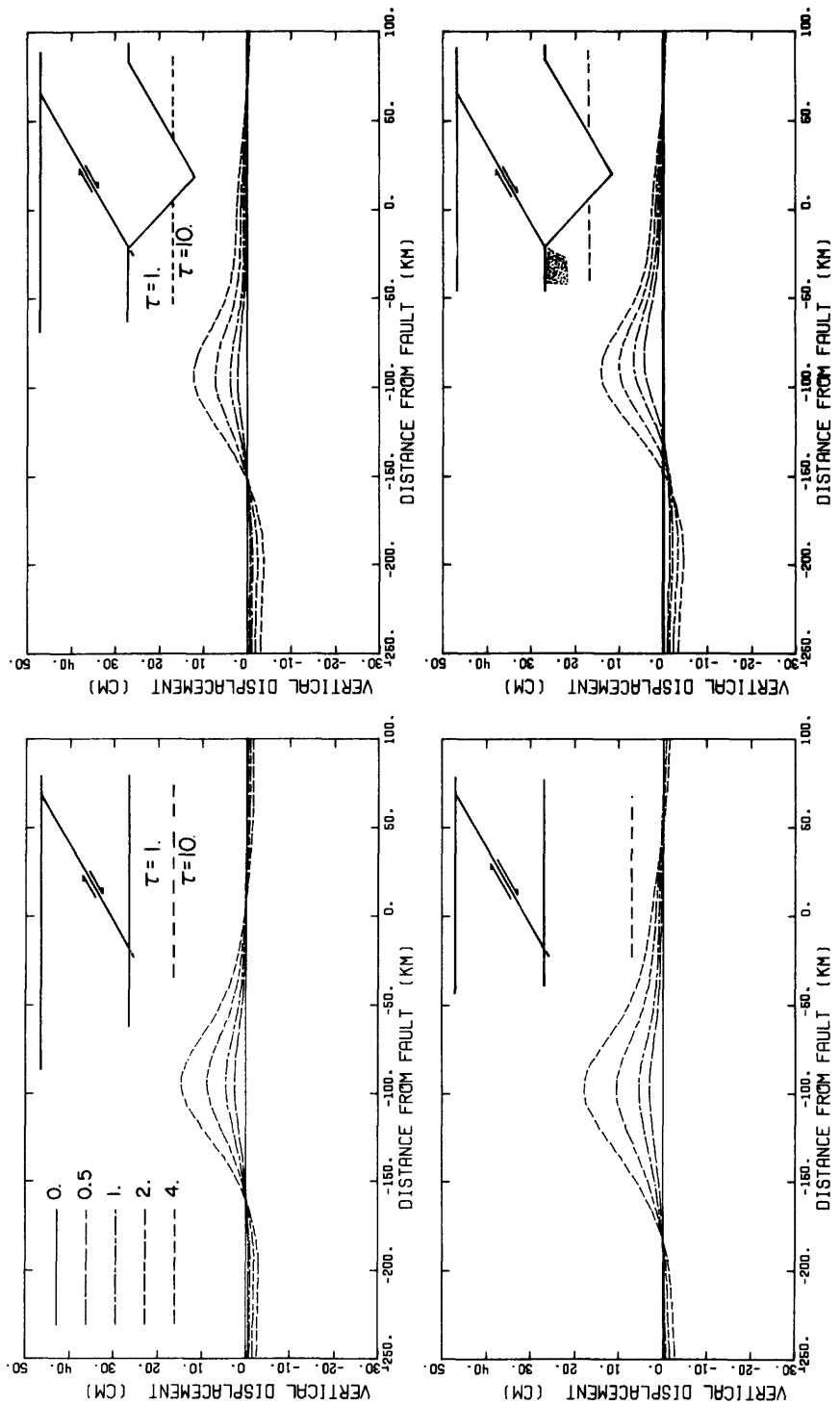

VERTICAL DISPLACEMENT (CM)
DISTANCE FROM FAULT (KM)
0.
0.5
1.
2.
4.
τ = 1.
τ = 10.

Figure 13

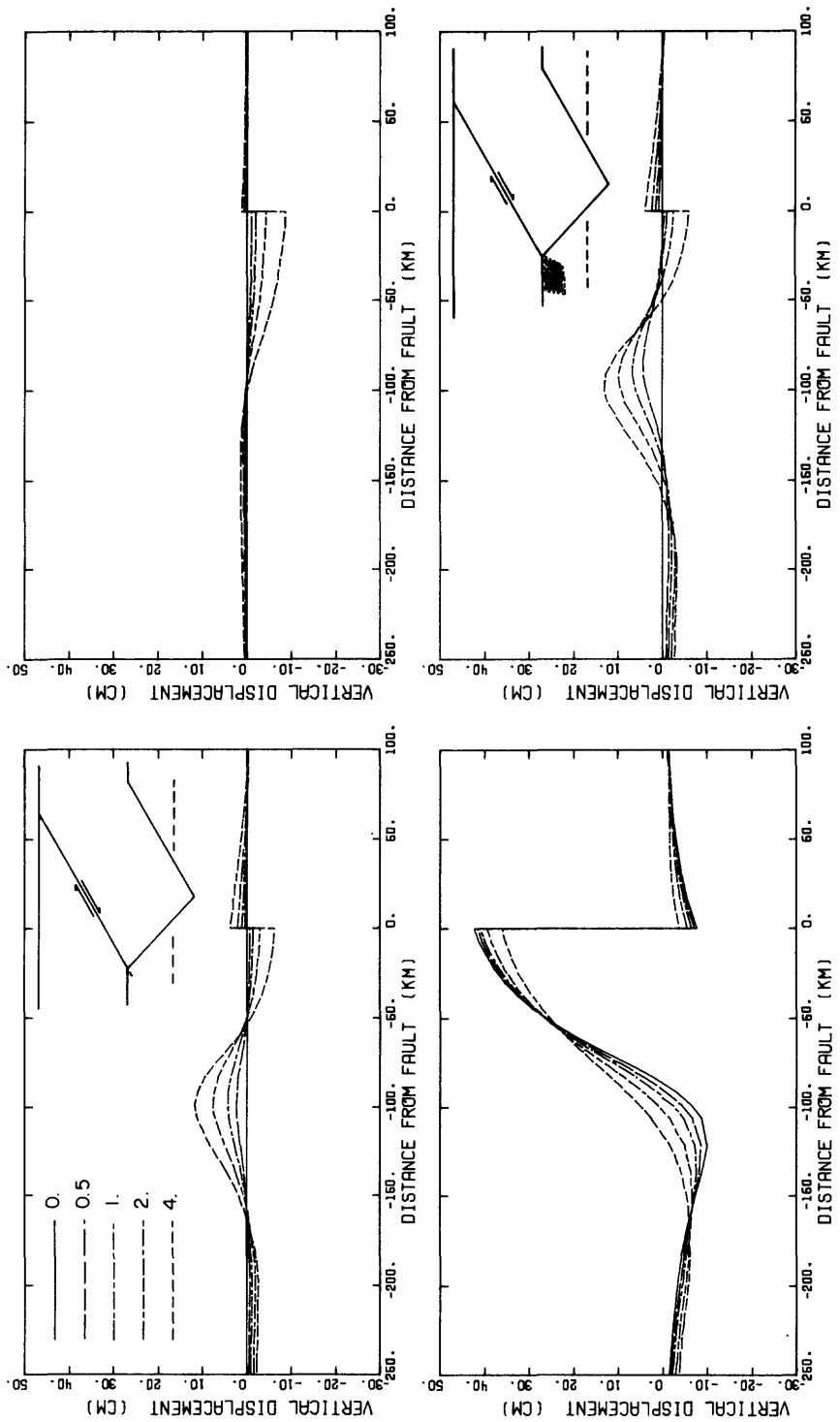


Figure 14

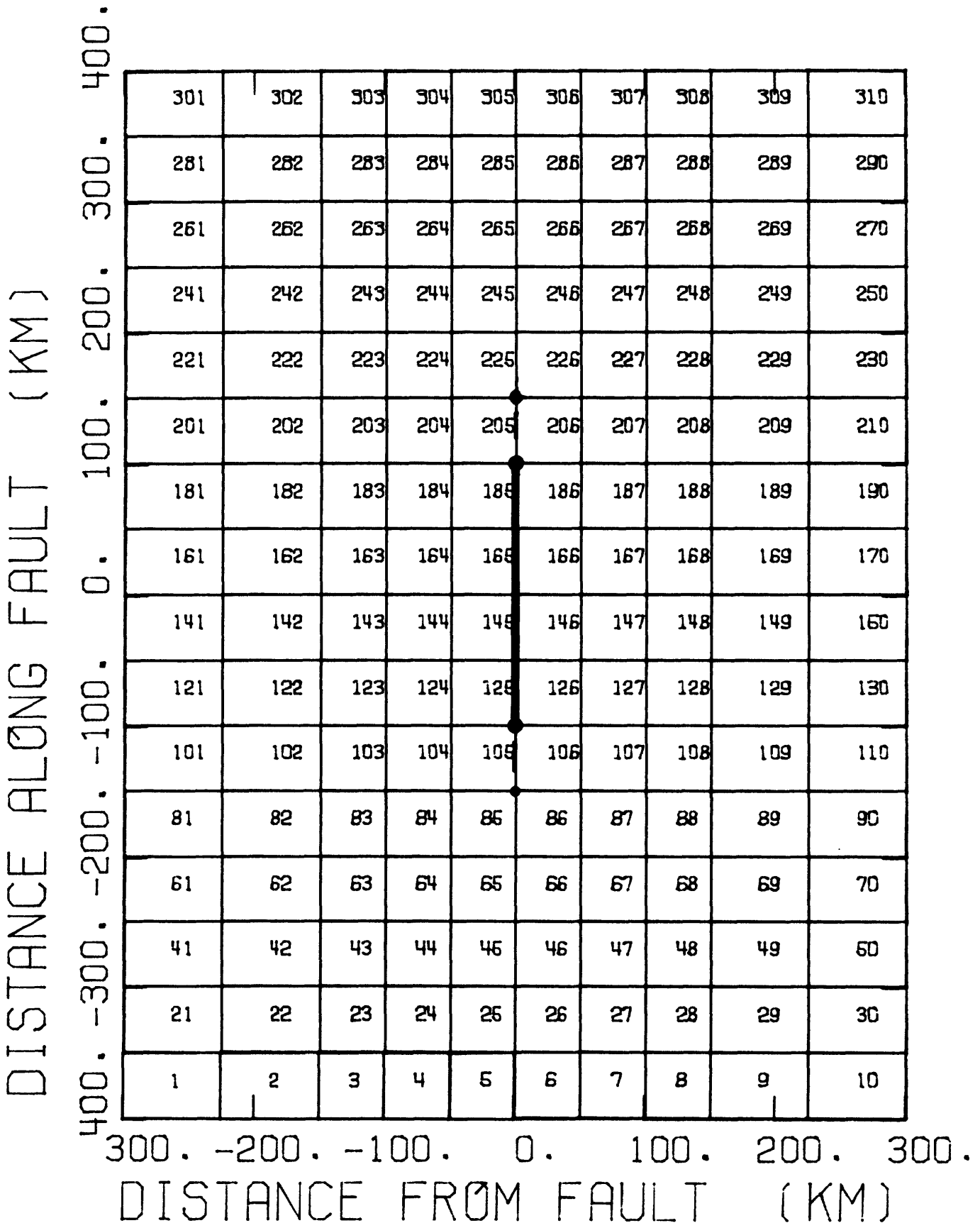

DISTANCE ALONG FAULT (KM)
400. 300. 200. 100. 0. -100. -200. -300. -400.
301 302 303 304 305 306 307 308 309 310
281 282 283 284 285 286 287 288 289 290
261 262 263 264 265 266 267 268 269 270
241 242 243 244 245 246 247 248 249 250
221 222 223 224 225 226 227 228 229 230
201 202 203 204 205 206 207 208 209 210
181 182 183 184 185 186 187 188 189 190
161 162 163 164 165 166 167 168 169 170
141 142 143 144 145 146 147 148 149 160
121 122 123 124 125 126 127 128 129 130
101 102 103 104 105 106 107 108 109 110
81 82 83 84 85 86 87 88 89 90
61 62 63 64 65 66 67 68 69 70
41 42 43 44 46 46 47 48 49 60
21 22 23 24 26 26 27 28 29 30
1 2 3 4 5 6 7 8 9 10
300. -200. -100. 0. 100. 200. 300.
DISTANCE FROM FAULT (KM)

Figure 15

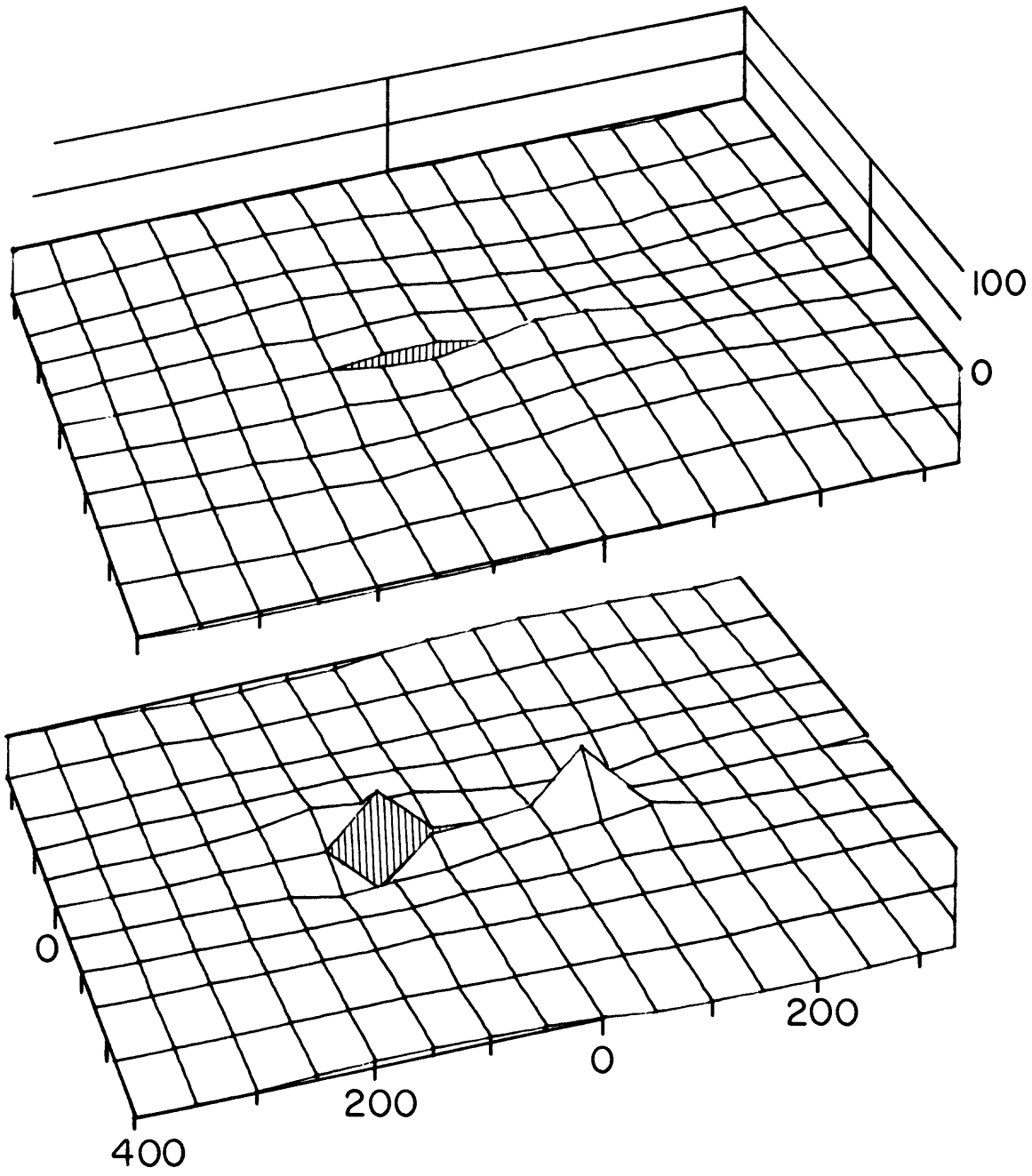
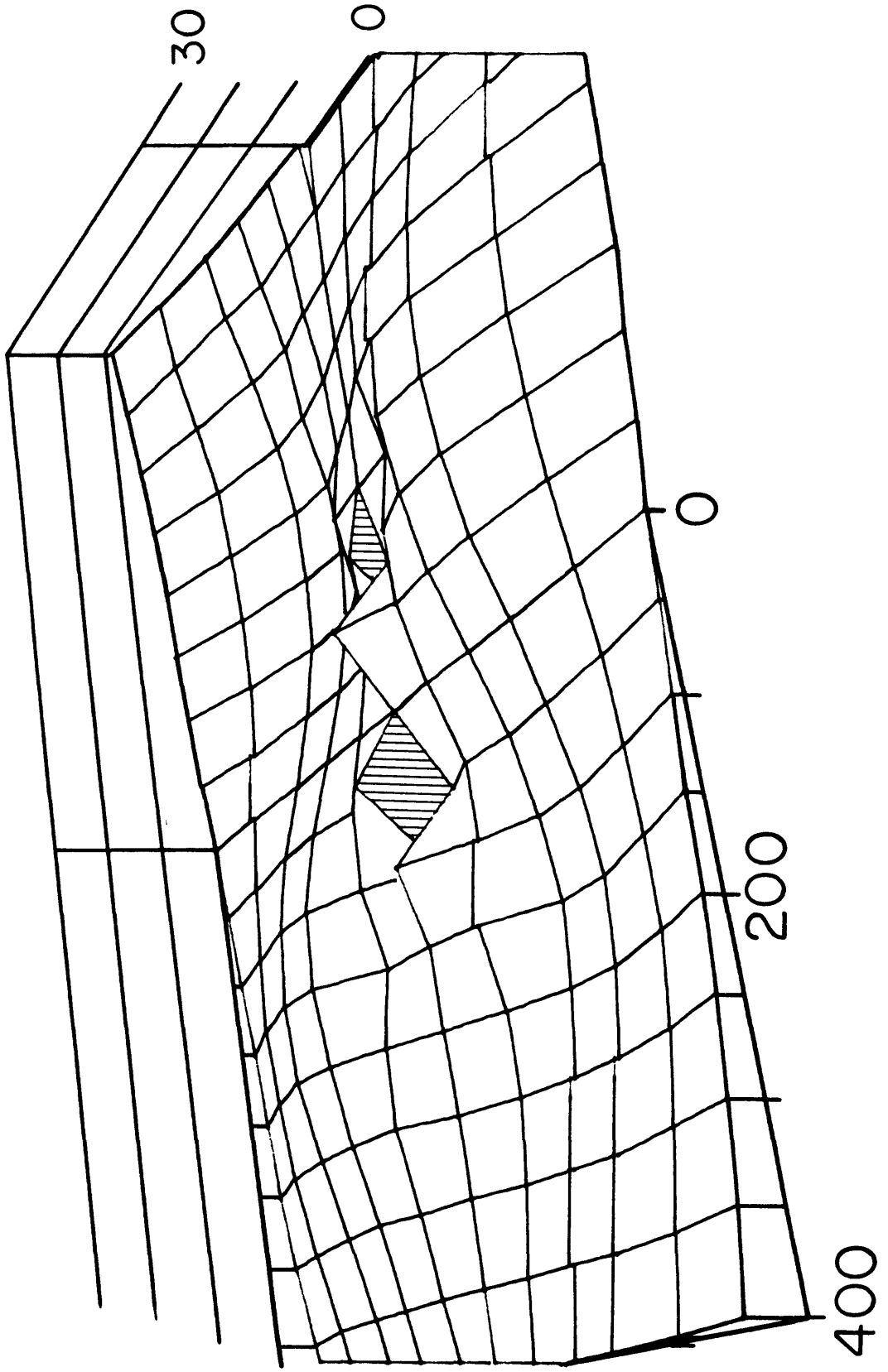


Figure 16



Adey, R. A. and Brebbia, C. A. 1973. Efficient method for solution of viscoelastic problems. ASCE, EMG, 1119-1127. Preprint.

Anderson, O. D. 1975. Time series analysis and forecasting the Box-Jenkins approach. Butterworths, Boston.

Ando, M. 1975a. Source mechanism and tectonic significance of historical earthquakes along the Nankai trough, Japan. Tectonophysics, 27, 119-140.

Ando, M. 1975b. Long-duration faulting in the 1946 Wankaido earthquake. EOS, Trans. Amer. Geophys. On., 56, 1067.

Ando, M. 1976. Personal communication.

Argyris, J. H. 1954. Energy theorems and structural analysis, Aircraft Eng., 26, 347-356 (Oct.), 383-387, 394 (Nov.).

Barker, T. G. 1976. Quasi-static motions near the fault zone. Geophys. J. R. Astr. Soc., 45, 689-705.

Beauchamp, K. G., Yuen, C. K. 1979. Digital methods for signal analysis. George Allen and Unwin, London.

Bevington, P. R. 1969. Data reduction and error analysis for the physical sciences. McGraw-Hill Book Co., New York.

Bilby, B. A. and J. D. Eshelby, 1968. Dislocations and the theory of fracture. In Fracture, ed. by H. Liebowitz, Academic Press, N. Y., Vol. I, pp. 99-182.

Biot, M. A. 1958. Linear thermodynamics and the mechanics of solids. Third U. S. Natl. Congress of Appl. Mech. 1-18.

Biot, M. A. 1965. Mechanics of incremental deformations, Wiley, 504pp.

Biot, M. A. 1970. Variational principles in heat transfer. Clarendon, 185pp.

Bischke, R. E. 1974. A model of convergent plate margins based on the recent tectonics of Shikoku, Japan. J. Geophys. Res., 79, 4845-4857.

Box, G.E.P. and Jenkins, G.M. 1976. Time Series Analysis: forecasting and control San Francisco. Holden-Day.

Brigham, E. O. 1974. The fast fourier transform. Englewood Cliffs: Prentice-Hall.

Brown, L. D., R. E. Reilinger, S. R. Holdahl, and E. I. Balazes 1977. Post seismic crusted uplift near Anchorage Alaska. J. Geophys. Res., 82, 3369.

Christensen, R. M. 1971. Theory of viscoelasticity: An introduction. Academic Press, 245pp.

Courant, R. 1943. Variational methods for the solution of problems of equilibrium and vibrations. Bull. Amer. Math. Soc., 49, 1-23.

Crustal Activity Research Office, Geographical Survey Institute 1972. Vertical movements in Shikoku District. Report Coord. Comm. Earthquake Pred., 7, 86.

Crustal Activity Research Office, Geographical Survey Institute 1973. Vertical movements in Kinki District. Report Coord. Comm. Earthquake Pred., 9, 104.

Delsemme, J., Smith, A. T. 1979. Spectral analysis of earthquake migration in South America. Pure and Applied Geophys. v. 117, No. 5.

Desai, C. S. and J. F. Abel 1972. Introduction to the finite element method. Van Nostrard, 462pp.

Earthquake Research Institute, Matsugama Geophys. Obs. 1975. Crustal movements in Matsugama District. Report Coord. Comm. Earthquake Pred., 14, 95.

Fitch, T. J. and C. H. Scholz 1971. Mechanism of underthrusting in southwest Japan: A model of convergent plate interactions. J. Geophys. Res., 76, 7260-7292.

French, A. S. and Holden, A. V. 1971. Alias-free sampling of Neuronal spike trans Kybernetik, 8, 5, 165-171.

Fung, Y. C. 1965. Foundations of solid mechanics. Prentice-Hall, 525pp.

Gelfand, I. M. and S. V. Fomin 1963. Calculus of variations. Prentice-Hall, 232pp.

Gurtin, M. E. 1963. Variational principles in the linear theory of viscoelasticity, Arch. Rational Mech. Anal., 13, 179-191.

Hadley, D. and Kanamori, H. 1977. Seismic structure of the transverse ranges. Preprint.

Hai-Chang, Hu 1955. On some variational principles in the theory of elasticity and the theory of plasticity. Sc. Sinica, 4, 33.

Isacks, B., J. Oliver, and L. Sykes 1968. Seismology and the new global tectonics. J. Geophys. Res., 73, 5855-5899.

Jungels, P. H. and G. A. Frazier 1973. Finite element analysis of the residual displacements for an earthquake rupture: Source parameters for the San Fernando earthquake. J. Geophys. Res., 78, 5062-5083.

Kagan, Y. and Knopoff, L. 1976. Statistical search for non-random features of the seismicity of strong earthquakes. Phys. Earth Planet Int., 12, 291-318.

Kelleher, J. A. 1970. Space-time seismicity of the Alaska-Aleutian siesmic zone. J. Geophys. Res., 75, 29, 5745-5756.

Kelleher, J. A. 1972. Rupture zones of large South American earthquakes and some predictions. J. Geophys. Res. 77, 2087-2103.

Kelleher, J., Sykes, L., Oliver, J. 1973. Possible criteria for predicting earthquake locations and their application to major plate boundaries of the Pacific and the Caribbean. J. Geophys. Res., 78, 14, 2547-2585.

Kelleher, J., Savino, J., Rowlett, H., McCann, W. 1974. Why and where great earthquakes occur along island arcs. J. Geophys. Res., 79, 32, 4889-4899.

Kelleher, J., Savino, J. 1975. Distribution of seismicity before large strike slip and thrust-type earthquakes. J. Geophys. Res., 80, 2, 260-271.

Kelleher, J., and McCann, W. 1976. Buoyant zones, great earthquakes, and unstable boundaries of subduction. J. Geophys. Res., 81, 26, 4885-4896.

Kosloff, D. 1977. Numerical simulations of tectonic processes in Southern California. Geophys. J. R. astr. Soc., 51, 487-501.

Lee, E. H., J. R. M. Radok, and W. B. Woodward 1959. Stress analysis for linear voscoelastic materials. Trans. Soc. Rheology, 3, 41-59.

Love, A.E.H., 1944. Mathematical theory of elasticity, 4th ed., Dover, 643pp.

McCowan, D. W., P. Glover, and S. S. Alexander 1974. An inversion technique for the static finite element method

(FEM). Trans. Am. Geophys. Union, 55, 353.

McConnell, R. K. 1965. Isostatic adjustment in a layered earth. J. Geophys. Res., 70, 5171-5188.

McKenzie, D. P. and R. L. Parker 1967. The North Pacific: An example of tectonics on a sphere. Nature, 216, 1276-1280.

Marquardt, D. W. 1963. An algorithm for least squares estimation of non-linear parameters. J. of the Soc. Ind. Applied Math. 11, 431.

Mavko, G. and Stuart, W. D. 1979. A model study of interacting faults in the Hollister area. (abstract) EOS, Trans. Am. Geophy. Un, 60, 952.

Mogi, K. (1968). Migration of seismic activity. Bull. Earthq. Res. Inst., 46, 53-74.

Nur, A. and Mavko, G., 1974. Post-seismic viscoelastic rebound. Science, 183, 204-206.

Okada, A. and T. Nagata 1953. Land deformation of the neighborhood of Muroto Point offer the Nankaido great earthquake in 1946. Bull. Earthquake Res. Inst., Tokyo Univ., 31, 169-178.

Orringer, O. and S. E. French, 1972. FEABL, AFOSR TR72-2228, Dept. Aeron. and Astro. Mass. Inst. of Tech.

Parzen, E. 1961. Mathematical considerations in the estimation of spectra. Technometrics 3, 67-190.

Plafker, G. 1972. Alaskan Earthquake of 1964 and Chilean earthquake of 1960: Implications for arc tectonics. J. Geophys. Res., 77, 901-925.

Rayner, J. N. 1971. An introduction to spectral analysis. Pion Limited, London.

Reid, H. F. 1910. The mechanics of the earthquake. In: The California Earthquake of April 18, 1906. Rept. State Earthquakes Invest. Comm., Carnegie Inst., Washington, D. C., 192pp.

Richtmyer, R. D. and K. W. Morton 1967. Difference methods for initial-value problems. Interscience, 2nd ed.

Robinson, E. A., Silvia, M. T. 1979. Digital foundations of time series analysis: vol. 1 - the Box-Jenkins approach. Holden-Day, San Francisco.

Rosenman, M. and S. J. Singh 1973a. Quasi-static strains and tilts due to faulting in a viscoelastic half-space. Bull. Seism. Soc. Am., 63, 1737-1752.

Rosenman, M. and S. J. Singh 1973b. Stress relaxation in a semi-infinite viscoelastic earth model. Bull. Seismol. Soc. Am., 63, 2145-2154.

Rundle, J. B. 1978. Viscoelastic crustal deformation by finite, quasi-static sources. J. Geophys. Res., 83, 5937-5945.

Rundle, J. B. 1979. Modeling of horizontal crustal strain in Southern California during 1971-1978 (abstract). Trans. Am. Geophy. Un., EOS, 60, 810.

Rurdle, J. B. and D. D. Jackson 1977. A kinematic viscoelastic model of the San Francisco earthquake of 1906. Geophys. J. Reg. Astron. Soc., 50, 441-458.

Savage, J. C. and Prescott, W. H. 1978. Asthenosphere readjustment and the earthquake cycle. J. Geophys. Res., 83, 3369-3376.

Schapery, R. A. 1961. Approximate methods of transform inversion for viscoelastic stress analysis. Proceedings of the Fourth U. S. Nat. Cong. of Appl. Mech., 1075-1085.

Schapery, R. A. 1962. Irreversible thermodynamics and variational principles with applications to viscoelasticity. Ph.D. thesis, California Institute of Technology.

Schapery, R. A. 1964. On the time dependence of viscoelastic variational solutions. Quart. Appl. Math., 22, 207-215.

Scholz, C. 1972. Crustal movements in tectonic areas. In: E. F. Savarensky and T. Rikitake (ed.) Forerunner of Strong Earthquakes, Tectonophysics, 14, 201-217.

Scholz, C. H., M. Wyss, and S. W. Smith 1969. Seismic and aseismic slip on the San Andreas fault. J. Geophys. Res., 74, 2049-2069.

Shimazaki, K. 1974. Pre-seismic crustal deformation caused by an underthrusting oceanic plate, in eastern Hokkaido, Japan. Phys. Earth Planet. Int., 8, 148-157.

Slade, M. A., H. J. Melash, and A. Raefsky 1979. Non-Newtonian post-seismic motions: a source of Chandler Wobble? (abstract). EOS (Trans. Am. Geophys. Un.), 60, 879.

Smith, A. T. 1974a. Time-dependent strain accumulation and release at island arcs. (abstract), EOS, Trans. Amer. Geophys. Un., 55, 427.

Smith, A. T. 1974. Time-dependent strain accumulation and release at island arcs: implications for the 1946 Nankaido earthquake. Ph.D. thesis, Mass. Inst. of Tech., Cambridge, Nov.

Smith, A. T. 1976. Post-seismic fault creep or stress-relaxation following the 1946 Nankaido earthquake? Trans. Am. Geophys. Un., 57, 290.

Smith, A. T. 1977. Earthquake risk analysis using numerical and stochastic models of time-dependent strain fields. U. S. Geological Survey, Semi-Annual Technical Report, Grant No. 14-08-0001-G399, Oct.

Smith, A. T. 1978. Earthquake risk analysis using numerical and stochastic models of time-dependent strain fields. U. S. Geological Survey, Semi-Annual Technical Report, Grant No. 14-08-0001-G399, Oct.

Smith, A. T. and M. N. Toksöz 1972. Stress distribution beneath island arcs. Geophys. J. R. Astr. Soc., 29, 289-318.

Snyder, D. L. 1975. Random Point Processes. Wiley, New York.

Strang, G., Fix, G. J. 1973. An analysis of the finite element method. Prentice-Hall, Englewood Cliffs.

Thatcher, W. 1975. Strain accumulation and release mechanism of the 1906 San Francisco earthquake. J. Geophys. Res., 80, 4862-4872.

Thatcher, W. and J. B. Rondle 1979. A model for the earthquake cycle in underthrust zones. J. Geophys. Res., 84, 5540-5556.

Tsumura, K. 1970. Investigation of mean sea level and its variation along the coast of Japan (Part 2). Jour. Geod. Soc. Japan, 16, 239-275.

Turner, M. J., R. W. Clough, H. C. Martin, and L. P. Topp, 1956. Stiffness and deflection analysis of complex structures. J. Aeron. Sci., 23, 805-823.

Utsu, T. 1967. Anomalies in seismic wave velocities and attenuation associated with a deep earthquake zone (I), J. of Faculty of Science, Hok. Univ., Japan, Ser. VII, 3, 1-25.

Wahr, J. and M. Wyss 1979. Interpretation of post-seismic deformation - with a viscoelastic relaxation model. Preprint.

Washizu, K. 1955. On the variational principles of elasticity and plasticity. Rept. 25-18, Cont. N5ORI-07833, Mass. Inst. of Techn. March.

Wilkinson, J. H. and C. Reinsch 1971. Linear Algebra. Springer-Verlag, Berlin, 438pp.

Wold, H. 1954. A study in the analysis of stationary time zones. Almquist & Wiksell: Stockholm.

Zienkiewicz, O. C., M. Watson, and I. P. King 1968. A numerical method of visco-elastic stress analysis. Int. J. Mech. Sci., 10, 807-827.

Zienkiewicz, O. C. 1971. The finite element method in engineering science, 521 pp. McGraw-Hill, London.

Zienkiewicz, O. C. 1977. The finite element method. McGraw-Hill, London, 787pp.

APPENDICES

APPENDIX A .1

Implementation of time-dependent, finite element method

APPENDIX A2 outlines a finite element method for linear time-dependent problems; now the implementation of this scheme introduces new problems. Flexibility is needed for the method: elastic solutions, gravity, faults, and inversions should be available. Each finite element solution must yield both displacement and stress solutions in the time-domain. These requirements place tremendous demands upon input-output control within the programming. All these require careful implementation to give an efficient and useful computational strategy.

These requirements are obvious in the solution method. Each finite element solution in the Laplace domain demands a separate factoring of the stiffness matrix; each element's contribution involves an integral of the transformed relaxation function $\bar{\underset{\sim}{G}}$

$$\underset{\sim}{K} = \sum_{n} \frac{1}{2} \int_{V} \underset{\sim}{\phi}^{T} \underset{\sim}{E}^{T} \bar{\underset{\sim\sim}{G}} \underset{\sim}{E} \underset{\sim}{\phi} dv \qquad (A.1)$$

where the notation conforms to Chapter 2. For each Laplace time the transformed displacement $\bar{\underset{\sim}{q}}$ are the solution to

$$\underset{\sim}{K}\ \bar{\underset{\sim}{q}} = \bar{\underset{\sim}{Q}} \qquad (A.2)$$

where the boundary conditions and loads $\bar{Q}$ start at time zero (section 2.3). The inversion to the time-domain using collocation or least-squares requires the transformed displacements $\bar{q}$ for each reduced time. If stresses are necessary, we also save the stress-strain matrix for each reduced Laplace time and element. This involves enormous data retention. Once the finite element solutions are available in the Laplace domain, least-squares fits the displacements $\bar{q}$ or element stresses to the transformed exponential series:

$$[\bar{q}]_{P=1/\gamma_j} = \left[\sum_{i=1}^{n} \frac{S_i}{1+\gamma_i P}\right]_{P=1/\gamma_j} \qquad j = 1,n \qquad (A.3)$$

where

$$q = \sum_{i=1}^{n} S_i (1 - e^{-t/\gamma_i}) \qquad (A.4)$$

in the time domain (section 2.6). Notice that all initial displacements or stresses correspond to a step function at time zero. The fitting implies access to many solutions in the Laplace domain. Inversions multiply this requirement: each fault segment needs its own time-dependent solution. Fortunately, multiple factoring is unnecessary for the stiffness matrix K; rather, forward-backward substitution yields each solution when intermediate results are available. Again efficient data management is paramount.

Figure A.1

Outline of system structure for finite element computations and inversion. Three systems are represented:
i) The finite element computations (FEM) include SETUP, SOLUTION, and INVERSION stages. Three direct-access files on disk represent intermediate storage modes. For the solution program, gravity, the Laplace times for the solutions, and the problem type (elastic or viscoelastic) must be specified. The subsequent inversion to the time domain involves either the stress of displacement when the series times and, if necessary, a constant flow term are selected.
ii) The second system plots the displacements and stresses for models derived from either inversions or specific FEM problems. The program accesses the mesh configuration stored on disk and plots at desired times using the series approximation (equation 2.6.9).
iii) Given an initial model and the constraining data, the inversion stage generates new models together with the resolution and errors for each. The first program formulates the problem into an eigenvalue problem and the resulting work files are stored on direct-access disk. The second solves the problem, while the last generates new models and determines the resolution and errors for a specific number of eigenvalues. These models may now serve as a new initial model.

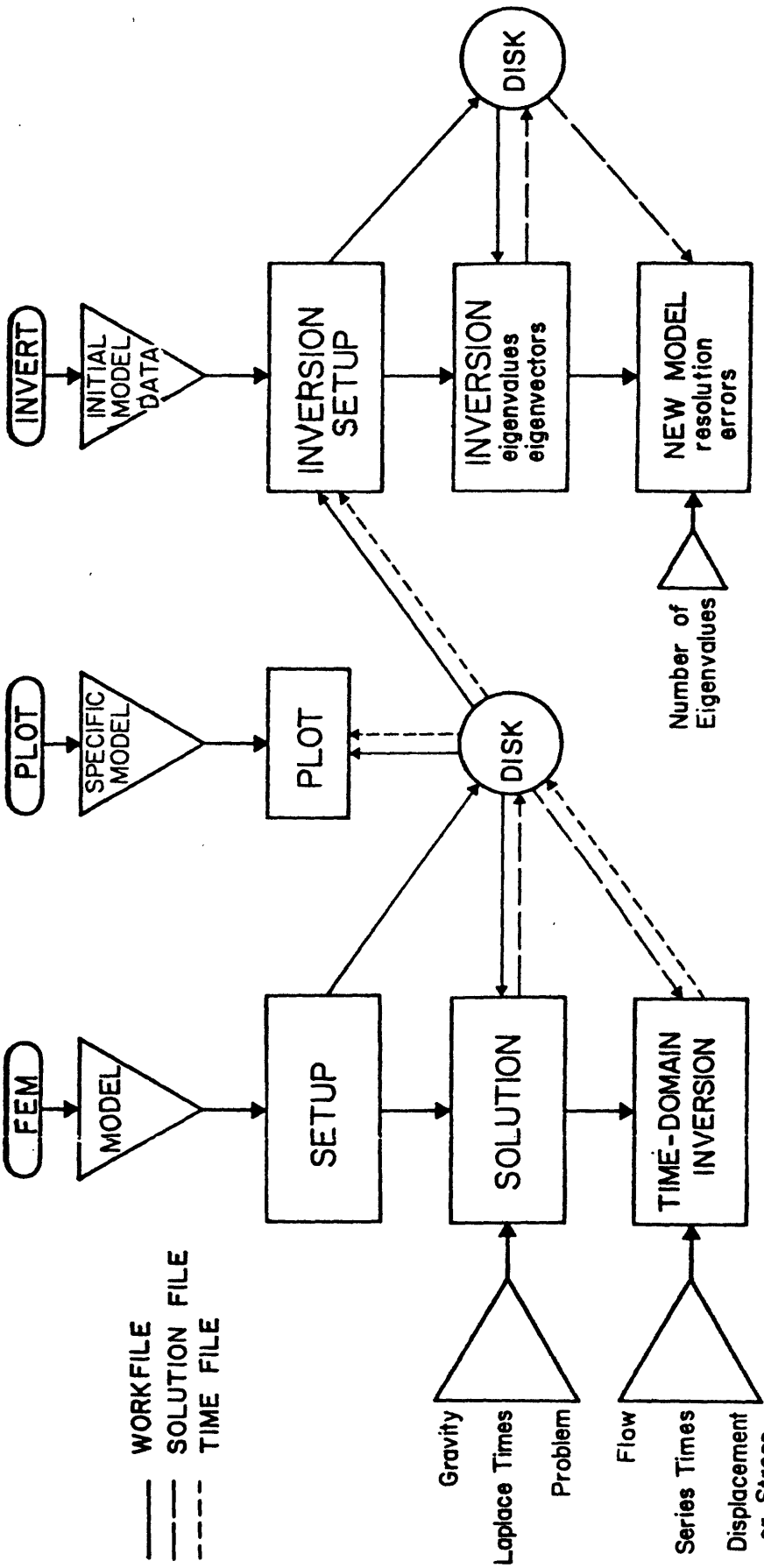

FEM
MODEL
SETUP
SOLUTION
TIME-DOMAIN INVERSION
Gravity
Laplace Times
Problem
Flow
Series Times
Displacement or Stress
PLOT
SPECIFIC MODEL
PLOT
DISK
INVERT
INITIAL MODEL DATA
INVERSION SETUP
INVERSION eigenvalues eigenvectors
NEW MODEL resolution errors
DISK
Number of Eigenvalues
WORKFILE
SOLUTION FILE
TIME FILE

Fig. A.1

On-line core storage is crucial. Adopting a scheme used by Orringer and French (1972), we store each band of the stiffness matrix beginning with the first non-zero entry. A significant savings in space results compared to the maximum bandwidth mode. Subroutines from the FEABLE system (Orringer and French, 1972) have been incorporated with modifications for setting up the problem, factoring the matrix using Cholesky's algorithm, and subsequent solution using forward-backward substitution. These provide an efficient solution scheme when we retain the whole stiffness matrix in core.

Figure A.1 shows the system structure adopted for the viscoelastic solution. A crucial element is sufficient versatility to allow solution of various problems: elastic, viscoelastic, and inversion for both displacements and stresses. Information must be accumulated for each. For example, the stress-strain matrix for each Laplace time and element must be saved for computing time-dependent stresses. To accomplish these tasks, we subdivide the problem into the following operations:

I. SETUP assigns locations to the storage areas (i.e. stiffness matrix, solution-force vector, bookkeeping variables, etc.) based upon the dimensions of the problem, and reads and stores the element information. This is independent of the type of solution, whether elastic or viscoelastic; it

only depends on the grid structure. All this is stored on a disc work-file together with the element information, material constants, node locations, densities, boundary conditions, and other assorted information. Access to the disk file occurs one track at a time to reduce input-output operations.

II. SOLUTION constructs the stiffness matrices and solves them for the specified problem. For an elastic solution just one factoring is necessary while a visco-elastic solution requires half a dozen or more separate assembling and factoring operations. Each factoring may involve more than one solution if a generalized matrix inversion is necessary. All the input is accessed from the disc work-file, while all output is stored on a disc solution-file. This dramatically decreases input-output expense when combined with an additional buffer variable located in a read-write subroutine. Large blocks of input or output may be transferred thereby reducing the input-output operations to the computer.

III. TIME DOMAIN INVERSION reads the displacement solutions for each Laplace time and inverts using the series approximation for either displacement or stress. The resulting coefficients are stored on disc file. Once

these solutions are available, various modes of processing can be executed including plotting and constructing the variational parameters for the inversion solutions.

IV. DATA INVERSION

The generalized matrix inverse allows a simple inversion scheme for the fault displacements. We access a file containing the time inverted coefficients for the displacements and introduce a starting model. SETUP fixes the viscosity ratios for the asthenosphere and mantle; however, scaling allows us to vary the initial viscosity in the asthenosphere. In the starting model we constrain this initial viscosity and define the fault segments for the inversion. The resulting coupling and variational coefficients together with appropriate weighting and errors define the coefficient matrix for the inversion.

Now the first of two routines multiplies the coefficient matrix by its transpose and factors into a tridiagonal matrix using Householder's method, solves for the p largest eigenvalues using bisection, and finds the eigenvectors. With the results stored on disc, the second reoutine selects the number of eigenvalues and generates the solution, variances, resolution matrix, and weighting matrices for the next iteration. The structure facilitates alterations and new models without solving a new finite element model.

APPENDIX A.2

FINITE ELEMENT METHOD AND LINEAR VISCOELASTICITY

2.2 Variational Problem for Linear Viscoelasticity

Essential to the finite element problem is a variational formulation of the problem: the solution technique is based on the minimization of an integral-differential operator. For boundary value problems as in the theory of elasticity, these variational principles are relatively straightforward and assert that a function u (displacement or stress) satisfies such a problem if and only if the given functional (i.e. potential or complementary energy) is stationary at u. However, viscoelasticity requires an "evolutionary" principle: a functional which is minimized along a trajectory in time.

Several derivations of variational theorems in quasi-static viscoelasticity are possible. Schapery (1962) has used Biot's thermodynamic theory to deduce one such principle.

This theorem imposes certain limitations on the class of problem. Instead, a simple generalization of the elasto-static variatonal principle due to Gurtin (1963) using Stieljes convolutions will be proven. Later we will review Schapery's assumptions for they bear upon the numerical technique.

For reference the differential equation defining the quasi-static viscoelastic boundary value problem are (Christensen, 1971)

$$\varepsilon_{ij} = 1/2(u_{i,j}+u_{j,i}) \text{ assuming infinitesimal strain and } u_{i,j} = \frac{\partial u_i}{\partial x_j}$$

$$\sigma_{ij} = \int_0^t G_{ijk\ell}(t-\tau) \frac{\partial \varepsilon_{k\ell}(\tau)}{\partial \tau} d\tau \qquad (2.2.1)$$

or

$$\sigma_{ij} = G_{ijk\ell} * d\varepsilon_{k\ell}$$

using the Stieljes convolution form of the constituent relation. Note that σ_{ij} is the stress tensor and ε_{ij} is the strain tensor.

$$\sigma_{ij,j} + F_i = 0 \quad \text{for equilibrium relation,} \qquad (2.2.2)$$

and

$$\sigma_{ij}n_j = S_i \quad \text{on } B_\sigma \qquad (2.2.3)$$

$$u_i = \Delta_i \quad \text{on } B_u \qquad (2.2.4)$$

where S_i and Δ_i are the prescribed stresses and displacements on the boundary B when n_j is the normal vector and F_i is the body force. The kernals $G_{ijk\ell}(t)$ are mechanical properties of the material and are termed relaxation functions.

The object is to derive a functional π whose variation $\delta\pi$ vanishes and yields equations (2.2.2) - (2.2.4). We now define the functional π according to Gurtin (1963):

$$\pi = \int_V [1/2G_{ijk\ell} * d\varepsilon_{ij} * d\varepsilon_{k\ell} - \sigma_{ij} * d\varepsilon_{ij} - (\sigma_{ij,j}+F_i) * du_i]dv + \int_{B_u} [\sigma_i * d\Delta_i]da + \int_{B_\sigma} [(\sigma_i - S_i) * du_i]da \qquad (2.2.5)$$

where F_i, Δ_i, and S_i are given quantities and each variable is dependent upon the appropriate coordinates. Again we let $\sigma_i = \sigma_{ij}n_j$ on the boundary. π is the correct functional if the Euler equations give the proper field equations and boundary conditions. Thus it is necessary that the first variational $\delta\pi$ of the functional π in equation (2.2.5) vanish if and only if the field equations and boundary conditions given by equations (2.2.2) - (2.2.4) are satisfied. The variational principle is then a generalization to visco-elasticity of the Hu-Washizu theorem in elasticity (Hai-Chang, 1955; Washizu, 1955).

Suppose we take the variations in the histories $u_i(\tau)$, $\varepsilon_{ij}(\tau)$, and $\sigma_{ij}(\tau)$ to prove this statement; that is

$$\begin{aligned} & u_i(\tau) + \delta u_i(\tau)\alpha \\ & \varepsilon_{ij}(\tau) + \delta\varepsilon_{ij}(\tau)\alpha \\ & \sigma_{ij}(\tau) + \delta\sigma_{ij}(\tau)\alpha \end{aligned} \qquad (2.2.6)$$

where α is some small real number and $\delta u_i(\tau)$, $\delta\varepsilon_{ij}(\tau)$, and $\delta\sigma_{ij}(\tau)$ are sufficiently smooth, arbitrary functions. The

general variation of the function is desired (Gelfand and Fomin, 1963). The first variation of π becomes

$$\delta\pi = \int_V [G_{ijk\ell} * d\varepsilon_{k\ell} * d\delta\varepsilon_{ij} - \sigma_{ij} * d\delta\varepsilon_{ij} - \delta\sigma_{ij} * d\varepsilon_{ij} - (\sigma_{ij,j} + F_i) * d\delta u_i - \delta\sigma_{ij,j} * du_i] dv \qquad (2.2.7)$$

$$+ \int_{B_u} [\delta\sigma_i * d\Delta_i] da + \int_{B_\sigma} [(\sigma_i - S_i) * d\delta u_i + \delta\sigma_i * du_i] da$$

Here the commutative property of Stieljes convolutions ($G_{ijk\ell} * d\delta\varepsilon_{ij} * d\varepsilon_{k\ell} = G_{ijk\ell} * d\varepsilon_{k\ell} * d\delta\varepsilon_{ij}$), and the symmetric relation ($G_{ijk\ell} = G_{k\ell ij}$) based on the symmetry of the stress and strain tensors are used (Christensen, 1971). Following Gurtin's development the term $\int_{B_u} [\delta\sigma_i * du_i] da$ is subtracted from the integral over B_u and added to the last term of the B_σ integral. Using Green's theorem, the first variation results:

$$\delta\pi = \int_V (G_{ijk\ell} * d\varepsilon_{k\ell} - \sigma_{ij}) * d\varepsilon_{ij} - (\sigma_{ij,j} + F_i) * d\delta u_i$$

$$- (\varepsilon_{ij} - 1/2 u_{i,j} - 1/2 u_{j,i}) * d\sigma_{ij}] dv$$

$$+ \int_{B_\sigma} [(\sigma_i - S_i) * d\delta u_i] da + \int_{B_u} [(\Delta_i - u_i) * d\delta\sigma_i] da \qquad (2.2.8)$$

To satisfy $\delta\pi = 0$ for arbitrary $\delta\sigma_i$, δu_i, and $\delta\varepsilon_{ij}$ each integral must separately vanish. This requires that the integrand of each equal zero giving us the field equation and boundary conditions, equations (2.2.2) - (2.2.4).

Since a 'displacement' approximation will be used with the finite element method, a simplified variation principle is more useful:

$$\pi_d = \int_V [1/2 G_{ijk\ell} * d\varepsilon_{ij} * d\varepsilon_{k\ell} - F_i * du_i] dv - \int_{B_\sigma} (S_i * du_i) da \tag{2.2.9}$$

Only the displacements are varied subject to the constraint imposed by the displacement boundary conditions (2.2.4). This variational principle is then analogous to stationary potential energy in elasticity and the proof proceeds as before (Christensen, 1971, sec. 5.4).

An operational form of the variational principle may now be obtained if the Laplace transform of π_d is taken (Schapery, 1962):

$$\bar{\pi}_d = \int_0^\infty e^{-pt} \pi_d(t) dt \tag{2.2.10}$$

obtaining

$$\bar{\pi}_d = \int_V [1/2 \bar{G}_{ijk\ell} * \overline{d\varepsilon}_{ij} * \overline{d\varepsilon}_{k\ell} - \overline{F_i} * \overline{du_i}] dv - \int_{B_\sigma} (\bar{S}_i * \overline{du_i}) da \tag{2.2.11}$$

where the bar denotes the Laplace transform of the function. The variation of $\bar{\pi}_d$ derives the Laplace transformed Euler equations. These are completely analogous to the Euler equations in the time domain. Using this operational variational principle we can easily compute the displacement

solution in the Laplace domain given the transformed relaxation function $\bar{G}_{ijkl}$ for the material behavior and the transformed boundary conditions. Only the solution must be inverted to the time domain. These are the topics of the next sections.

2.3 Finite Element Method for Operational Variational Principles

The finite element method is very similar to the Rayleigh-Ritz-Galerkin technique. Each begins with a functional or variational principle and minimizes it for an approximating function u. This in turn leads to the Euler equations for the system. While finite difference schemes approximate the derivatives after minimization in these Euler or field equations, both the finite element and Rayleigh-Ritz methods start with the actual variational principle, but with one significant difference: the Rayleigh-Ritz-Galerkin strategy chooses a finite number of trial functions $\phi_1 \ldots \phi_N$ which span the whole domain and satisfy the boundary conditions for the variational principle. The finite element method, on the other hand, subdivides the domain into 'elements'. Within each element a simple, complete sequence of trial functions interpolates the function u. Compatibility with essential boundary conditions is easily achieved for the trial functions within each regular element, as opposed to the Rayleigh-Ritz method when irregular boundaries over the whole domain must be satisfied by the trial functions. Each subdivision

or element of the finite element domain can then be assembled with the others into discrete algebraic equations having "nice" banded properties, a significant advantage for efficient numerical solution. It is this simplicity of the trial or interpolation functions which gives the finite element method its power.

Courant (1943) first proposed the equivalent of the finite element method with a piecewise application of the Ritz method to the St. Venant torsion problem. Engineering problems, however, provided the real motivation for its development (Argyris, 1954; Turner, et al., 1956), while its relation to the Rayleigh-Ritz-Galerkin principles was only later recognized and used to advantage. It is not my intention here to outline the whole theory, for an excellent mathematical text is available (Strang and Fix, 1973). Instead the emphasis is placed upon the solution of the quasi-static viscoelastic problem using the operational formulation.

Common to all finite element problems is the variational formulation of the equations. These can be developed by using the principle of 'virtual work' for elasticity theory (Fung, 1965, sec. 10.7), by introducing the thermodynamics of the system (Biot, 1965, 1970), or by appealing to various mathematical operations as the Galerkin method (Strang and Fix, 1973, sec. 2.3). It is fundamental to remember, however, that the variational principle determines the boundary

conditions satisfied by the trial or interpolation functions for each element. The variational statement represents the primary physical principle, and the differential equation and boundary conditions are only a secondary consequence. Care must be taken then to preserve the proper boundary conditions within the variational principle. The results will be obvious as the finite element approximation emerges from the variational principle.

Let us begin with the simplified displacement variational principle developed in the previous section for viscoelasticity. Although the operational form in the Laplace domain is used for the derivation, the results are exactly analogous to the elasticity problem. The transformed operational variational principle is given by equation (2.2.11) if $\varepsilon_{ij} = 1/2(u_{i,j}+u_{j,i})$:

$$\bar{\pi} = \int_V [1/2\bar{G}_{ijk\ell} * \overline{d\varepsilon}_{ij} * \overline{d\varepsilon}_{k\ell} - \bar{F}_i * \overline{du}_i]dv - \int_{B_\sigma} (\bar{S}_i * \overline{du}_i)da \tag{2.3.1}$$

when the integrations are carried over the volume V for the region and surface area B_σ for the applied tractions S_i. The bar denotes the Laplace transform of the function. Associated with this operational variational principle are the following Euler equations and boundary conditions in the Laplace domain:

$$\bar{\sigma}_{ij,j} + \bar{F}_i = 0 \tag{2.3.2}$$

$$\bar{\sigma}_{ijn_j} = \bar{S}_i \quad \text{on } B_\sigma \tag{2.3.3}$$

Now the approximation requires an interpolation function for u applied to each element, then summing the contributions of the elements.

The interpolation function for the subregion must satisfy certain conditions in displacement: first, rigid body motions must be possible. This is a consequence of the essential boundary conditions. Second, the admissible space of the derivatives in the variational principle must be represented in the interpolation. Finally, the set of functions should be complete between the lowest and highest degree of approximation. With the variational principle (2.3.1) the interpolation of u must contain a constant displacement term u_o and at least the linear term to represent the first derivative. Polynomial interpolation such as Hermite splines give the best and most complete representation. A discussion of this problem may be found in Strang and Fix (1973); here it will suffice to assume a polynomial representation is optimum.

Let us represent the function (i.e. displacement) u as a sum of the basis or interpolation functions ϕ_j. If the problem is discretized, nodal parameters q_j can be associated with each element. Each of these q_j is the value at a given node z_j of either the function itself or one of its derivatives. Thus one has for trial function v^h

$$q_j = D_j v^h(z_j) \qquad (2.3.4)$$

when D_j is the differential operator of order j. For each

element specific nodes are assigned for the approximation as illustrated in Figure 2.1. Now a trial function ϕ_j is assigned to each nodal parameter q_j having the following property: At node z_j, $D_j\phi_j$ equals 1, while at all other nodes the interpolation function is zero. One has

$$D_i\phi_j(z_i) = \delta_{ij} \tag{2.3.5}$$

Within the element the displacement trial function then becomes

$$u^h = \sum q_j\phi_j \tag{2.3.6}$$

This represents a local basis within one element.

If the variational principle is now considered as the sum of each element's contribution, one arrives at the following form after introducing matrix notation:

$$\bar{\pi} = \sum_n \{\int_V [1/2\overline{d\varepsilon}^T \underset{\sim}{G}\ \overline{d\varepsilon} - \overline{du}^T \bar{F}]dv - \int_{B_\sigma} (\overline{du}^T \underset{\sim}{\bar{S}})da\} \tag{2.3.7}$$

The interpolation basis function can be represented by

$$\underset{\sim}{\overline{du}} = \underset{\sim}{\phi}\underset{\sim}{\bar{q}} \tag{2.3.8}$$

and

$$\underset{\sim}{\overline{d\varepsilon}} = \underset{\sim}{E}\ \underset{\sim}{\overline{du}} = \underset{\sim}{E}\underset{\sim}{\phi}\underset{\sim}{\bar{q}} \tag{2.3.9}$$

where $\underset{\sim}{E}$ represents the strain operator, $\underset{\sim}{\phi}$ is the interpolation function, and $\underset{\sim}{\bar{q}}$ represents the nodal values. The operator $\underset{\sim}{E}$ depends upon the coordinate system and particular element.

Figure 2.1

Configuration for representative elements. Nodes Z_j define each element, where the values q_j approximate the trial function $\not{V}$ at node Z_j.

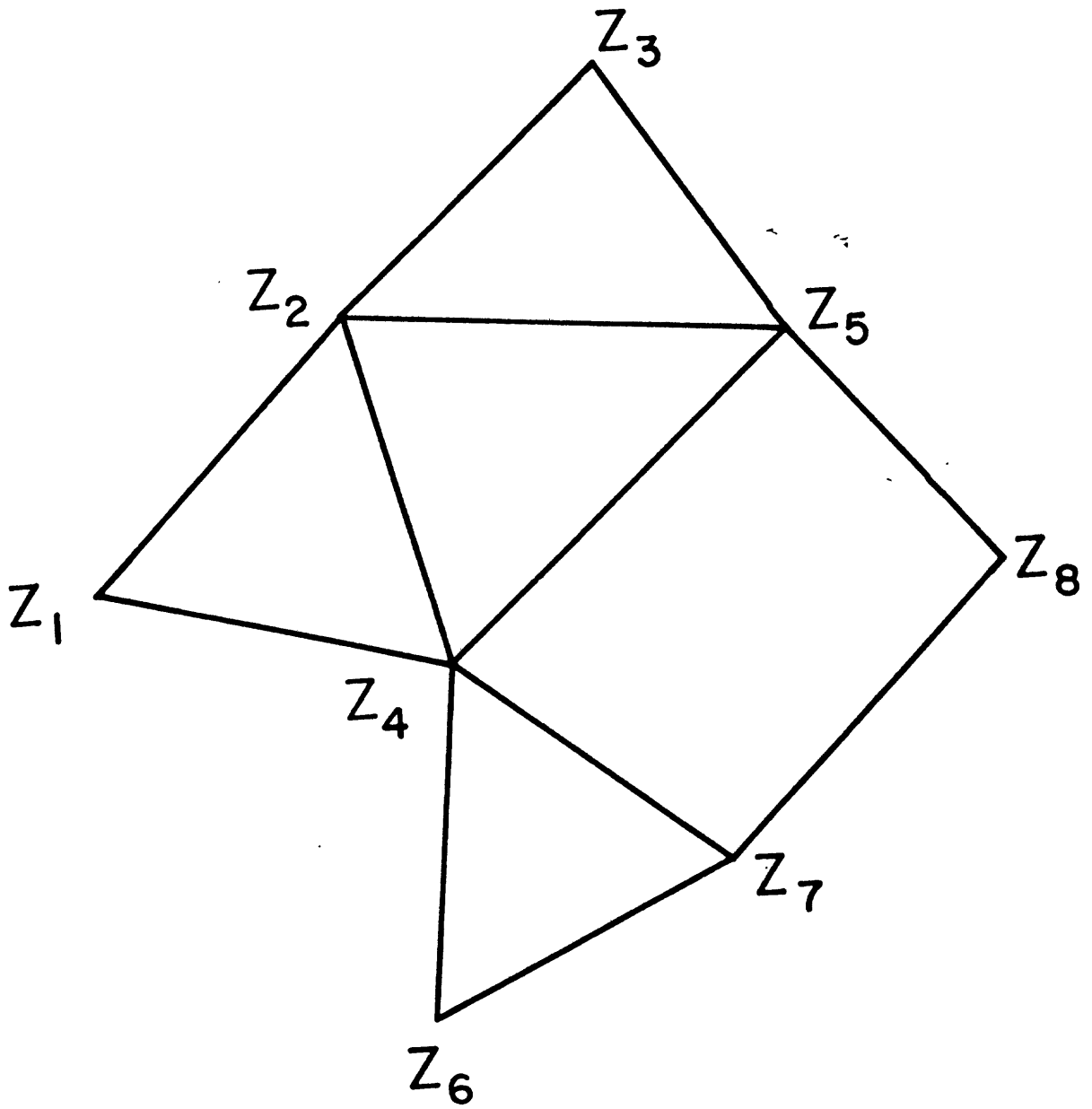


Fig. 2.1

for e.l. element

Substituting these into the variational principle yields

$$\bar{\pi} = \sum_{n} (\bar{q}^T k q - \bar{q}^T \bar{Q}^e) \tag{2.3.10}$$

where

$$k = 1/2 \int_V \phi^T E^T \bar{G} E \phi dv \tag{2.3.11}$$

$$\bar{Q}^e = \int_V \phi^T \bar{F} dv + \int_{B_\sigma} \phi^T \bar{S} da \tag{2.3.12}$$

Here $\bar{q}$ have been removed from the integrals since they represent the nodal values. All the elements may now be summed obtaining a matrix of the form

$$\bar{\pi} = \bar{q}^T K \bar{q} - \bar{q}^T \bar{Q} \tag{2.3.13}$$

where

$$K = \sum_{n} k \tag{2.3.14}$$

$$\bar{Q} = \sum_{n} \bar{Q}^e \tag{2.3.15}$$

The quadratic form for $\bar{\pi}$ is minimized when

$$K \bar{q} = \bar{Q} \tag{2.3.16}$$

Thus we need only solve this matrix equation.

The solution technique is improved if we observe certain properties of the matrix K: first, K is banded if care is taken when numbering or ordering the displacements q. Each position in K represents the coupling between two nodal

values; an entry will generally occur if the nodes are adjacent to one another. Thus adopting a uniform and sequential ordering scheme for $\underset{\sim}{q}$ drastically reduces the bandwidth of $\underset{\sim}{K}$ and its storage requirements. Second, $\underset{\sim}{K}$ is symmetric and positive-definite since $\bar{G}_{ijk\ell}$ are always positive. Under these circumstances Cholesky factorization and forward-backward substitution provides an efficient solution strategy for the problem (see Wilkinson and Reinsch, 1971, Part 1).

As one might have forseen, the elements and their basis functions have a crucial position in the method. The degree of the approximation dictates the minimum possible bandwidth, but also the accuracy of the approximation. For example, if a linear approximation to the displacement u is used on a triangular element, stress will be constant throughout the element (hence the name Constant Strain Triangle), and six degrees of freedom are required for a two-dimensional element. The error for the displacement is proportional to h^2 where h is the characteristic dimension of the element (Strang and Fix, 1973, chapter 1). Introducing a quadratic approximation and six nodes or twelve degrees of freedom on the triangular element vastly improves the accuracy. One now has a linear approximation in stress and h^4 error in displacement; however, the bandwidth is increased with twelve degrees of freedom for one element giving a corresponding increase in solution time (Strang and Fix, 1973, sec. 1.9). A trade-off exists then

between the order of the approximation and the solution time. It may be more expedient to use more three node than six node triangular elements to achieve a desired accuracy for the displacements. The most desirable element for stress approximation is not obvious for most circumstances, since the twelve degree of freedom triangular element can make an immense difference (Desai and Abel, 1972, sec. 6.2). For the viscoelastic computations both six node triangular and four node isoparametric quadrilaterals also improve the stress approximation. Each problem, then, poses different conditions requiring careful thought for the grid structure.

The singularity introduced by the fault (i.e. crack tip) strongly effects the element configuration. The convergence error associated with the singularity depends on the dimension of the element and not the order of the approximation within the element (Strang and Fix, 1973, chapter 8). Unless a special element containing a singularity function is introduced, the grid must be more refined near the crack tip. For these computations I have opted for increasing the number of elements rather than using a singularity function. Arbitrary displacements are necessary along the fault which implies singularity functions for each element along the whole fault. This complication would significantly effect the bandwidth and computational speed, negating the effort required for development and incorporation of these singularity elements.

2.4 Modified Variational Principle for Hydrostatic Stress

Gravity introduces prestressing into the media which can be reduced from the variational principle. Biot (1965) has extensively treated this problem; his development will be cited in this section. The finite element method introduces special problems since the symmetry of the banded matrix must be maintained. The terms representing hydrostatic prestressing, however, retain a bilinear form allowing a simple transformation to a real symmetric quadratic matrix.

When the hydrostatic component of the initial stress is removed from the variational principle and only residual or deviatoric stress is expressed, two additional terms are introduced which represent the work of the buoyancy forces (Biot, 1965, sec. 3.6):

$$Y = \rho_f X_j u_j e + 1/2\ X_j \frac{\partial \rho_f}{\partial x_i} u_i u_j \qquad (2.4.1)$$

Here ρ_f denotes the fluid density of the media producing the hydrostatic stress; X_j is the j component of the body force; e is the dilatation; and u are the displacement components. The first term, $\rho_f X_j u_j e$, expresses a buoyancy effect arising from a change of volume, while the second, $1/2\ X_j \frac{\partial \rho_f}{\partial x_i} u_i u_j$, represents a buoyancy generated from the displacement and the density gradient. The latter term appears as an elastic force proportional to the displacement directed normally to the equipotential surfaces. The modulus is proportional to

the product of the body force magnitude and the density gradient; hence, depending on the sign, it may be stabilizing or destabilizing.

To incorporate these terms within the finite element method, they must be represented in a symmetric quadratic form. We can represent the dilatation e using our nodal approximation in the finite element method (see section 2.3)

$$e = \underset{\sim}{E}\underset{\sim}{\phi}\underset{\sim}{q} \qquad (2.4.2)$$

where $\underset{\sim}{u} = \underset{\sim}{\phi}\underset{\sim}{q}$ for the displacements and $\underset{\sim}{E}$ represents the strain-displacement operator. Thus a bilinear form also results for Y:

$$Y = \underset{\sim}{q}^T\underset{\sim}{\alpha}^T(\rho\underset{\sim}{X})^T\underset{\sim\sim}{E}\underset{\sim\sim}{\alpha}\underset{\sim}{q} + 1/2\underset{\sim}{q}^T\underset{\sim}{\alpha}^T\frac{\partial\rho_f}{\partial\underset{\sim}{x}}^T\underset{\sim\sim}{X}\underset{\sim\sim}{\alpha}\underset{\sim}{q} \qquad (2.4.3)$$

Any real bilinear matrix $\underset{\sim}{A}$ can be transformed to a real symmetric quadratic

$$\underset{\sim}{x}^T\underset{\sim\sim}{C}\underset{\sim}{x} \qquad (2.4.4)$$

if

$$\underset{\sim}{C} = 1/2[\underset{\sim}{A} + \underset{\sim}{A}^T] \qquad (2.4.5)$$

This formulation, then, readily accommodates initial hydrostatic prestressing within the media.

2.5 Fault Zone in Finite Elements

An earthquake fault zone is analogous to a crack or internal boundary condition within the media. Different boundary conditions are possible along each face of the

crack: a stress boundary implies a constant stress drop while a displacement boundary signifies a Somigliana dislocation (Bilby and Eshelby, 1968). In this section a method is indicated which incorporates either boundary condition along a fault so long as linearity assumptions are maintained for the deformation.

For simplicity let us consider a displacement dislocation along a crack or fault represented by $\Delta u = u^{+}-u^{-}$. This internal boundary can be represented by two adjacent nodes with a prescribed displacement between the nodes. Each node is then free to deform within the media; yet their location relative to one another defines the dislocation. Figure 2.2 illustrates this configuration. To implement this strategy using the finite element method requires little additional effort if the coordinate system of the variational principle is transformed along the fault. We must express the absolute coordinates along one fault interface in terms of the opposing face. Assuming the elements have been assembled into a stiffness matrix, we have the following variational principle (see section 2.3):

$$\pi = \underset{\sim}{q}^{T}\underset{\sim\sim}{K}\underset{\sim}{q} - \underset{\sim}{q}^{T}\underset{\sim}{Q} \qquad (2.5.1)$$

where $\underset{\sim}{q}$ are the nodal parameters or generalized coordinates, $\underset{\sim}{K}$ is the stiffness matrix, and $\underset{\sim}{Q}$ represents the nodal forces. If the matrix is partitioned into sections

Figure 2.2

Representation for fault (or crack) within finite element region. Considering a pair of nodes, we define the u_2 coordinate in terms of u_1 ($u_2 = u_1 + \Delta u$) along the internal boundary.

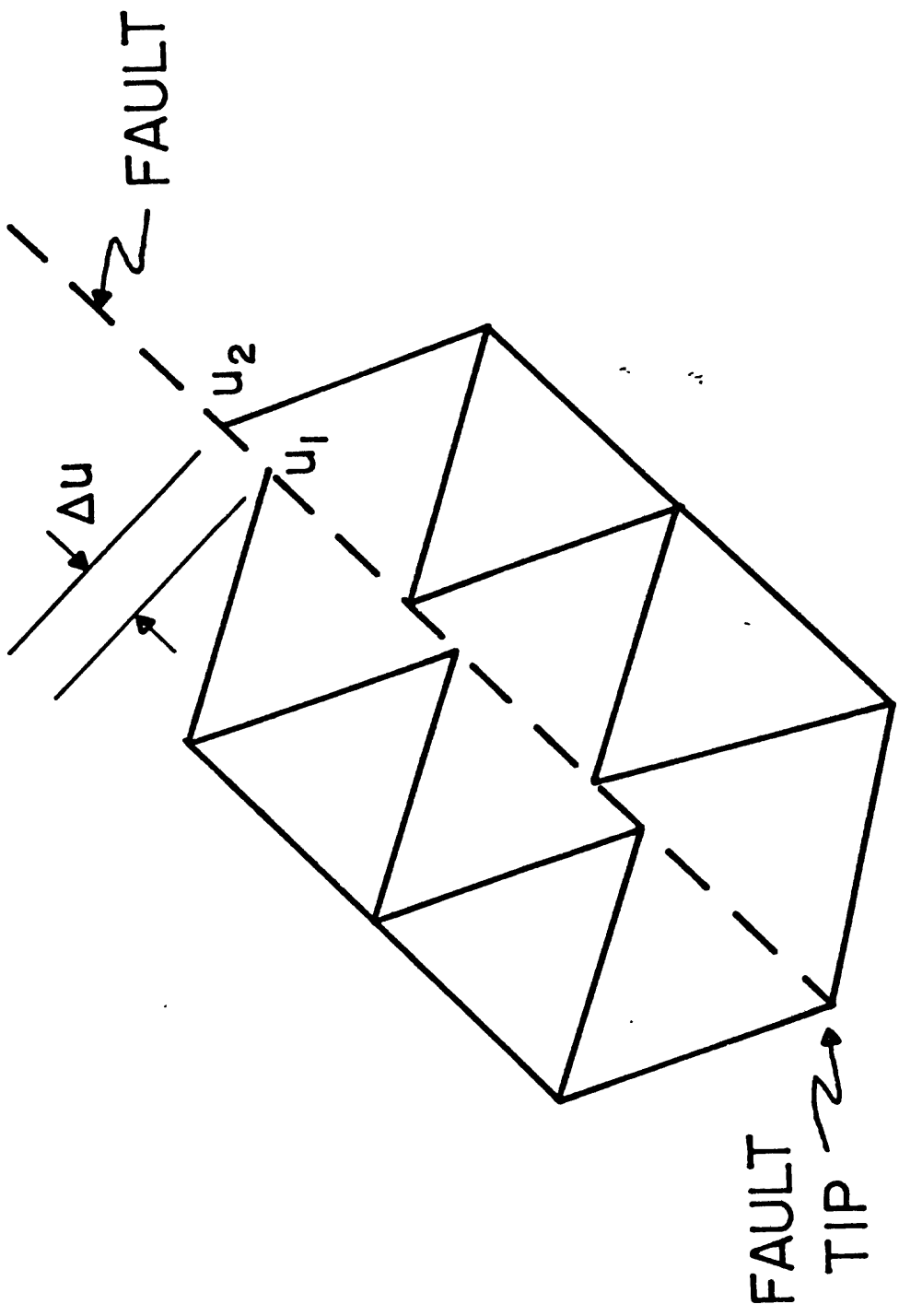


Fig. 2.2

$$\pi = \begin{bmatrix} q_\alpha & q_\beta & q_\gamma \end{bmatrix} \begin{bmatrix} K_{\alpha\alpha} & K_{\beta\alpha} & K_{\gamma\alpha} \\ K_{\beta\alpha} & K_{\beta\beta} & K_{\gamma\beta} \\ K_{\gamma\alpha} & K_{\gamma\beta} & K_{\gamma\gamma} \end{bmatrix} \begin{bmatrix} q_\alpha \\ q_\beta \\ q_\gamma \end{bmatrix} - \begin{bmatrix} q_\alpha & q_\beta & q_\gamma \end{bmatrix} \begin{bmatrix} Q_\alpha \\ Q_\beta \\ Q_\gamma \end{bmatrix} \tag{2.5.2}$$

where α are coordinates in the continuum, β are coordinates along one fault interface, and γ are the coordinates along the opposing fault interface. Introducing a new coordinate where

$$q_\gamma = q_\beta + \Delta \ , \tag{2.5.3}$$

we obtain

$$\pi = \begin{bmatrix} q_\alpha & q_\beta & \Delta \end{bmatrix} \left[\begin{array}{c|c|c} K_{\alpha\alpha} & K_{\beta\alpha} + K_{\gamma\alpha} & K_{\gamma\alpha} \\ \hline K_{\beta\alpha} + K_{\gamma\alpha} & K_{\beta\beta} + K_{\gamma\gamma} + 2K_{\beta\gamma} & K_{\gamma\beta} + K_{\gamma\gamma} \\ \hline K_{\gamma\alpha} & K_{\gamma\beta} + K_{\gamma\gamma} & K_{\gamma\gamma} \end{array}\right] \begin{bmatrix} q_\alpha \\ q_\beta \\ \Delta \end{bmatrix} - \begin{bmatrix} q_\alpha & q_\beta & \Delta \end{bmatrix} \begin{bmatrix} Q_\alpha \\ Q_\beta + Q_\gamma \\ Q_\gamma \end{bmatrix} \tag{2.5.4}$$

Δ now represents the dislocation along the internal boundary independent of the coordinates; it can either be a free parameter or prescribed along the interface.

Introducing this coordinate change simplifies the computations and properly simulates the dislocation. Jungels and Frazier (1973) express the fault dislocation in the force or inhomogeneous vector $\underset{\sim}{Q}$. This representation complicates the bookkeeping associated with assembling the elements. Each element location and submatrix adjacent to the fault must be retained in addition to any rows and

columns of the $\underset{\sim}{K}$ matrix for successive application of the fault displacements. If instead the fault is prescribed in the absolute coordinate frame (i.e. left in q), the problem is completely wrong. Rigid body motions of the crack or fault are then impossible. McCowan, Glover, and Alexander (1974) and Shimazaki (1974) are among those that have committed this error. The coordinate change appears then as the most efficient strategy to incorporate the fault.

2.6 Inversion to the Time Domain

Essential to the use of operational variational methods for transient problems is the final inversion to the time domain. A strategy is needed giving accuracy comparable to solutions in the Laplace domain, but unwarranted accuracy would be inefficient. Thus using Fast Fourier transforms are very inefficient for they require too many solutions and exact evaluation. Schapery (1962) developed, instead, a technique based upon the thermodynamic properties of linear viscoelasticity: apart from steady flow, the time dependence of all coordinates is given by a series of decaying exponentials (Biot, 1958). Schapery went on to prove uniform convergence for a finite sum of decaying exponentials and to demonstrate its power with the Rayleigh-Ritz method. But recognition of its potential usefulness to the finite element method is due to Adey and Brebbia (1973).

To prove this property of time dependence for the coordinates, we must first indicate certain properties of the kernal $G_{ijk\ell}$ in the Stieljes convolution or 'hereditary integral' (equation (2.2.1)). Using the thermodynamics we can prove the following assertions (Christensen, 1971):

I $$G_{ijk\ell}(t) \geq 0 \qquad (2.6.1)$$

from the requirement of non-negative work.

II $$\frac{\partial}{\partial t} G_{ijk\ell}(t) \leq 0 \qquad (2.6.2)$$

from the requirement of non-negative dissipation or increasing entropy.

III $$\frac{\partial^2}{\partial t^2} G_{ijk\ell}(t) \geq 0 \qquad (2.6.3)$$

if we require a fading memory behavior.

These conditions are fulfilled if we assume a prony series representation:

$$G_\alpha(t) = \sum_{n=1}^{N} G_\alpha^n \, e^{-t/\tau_{\alpha n}} + G_\alpha^\circ \qquad (2.6.4)$$

Steady flow introduces a differential operator into the kernal. Biot (1958) has proven equation (2.6.4) represents the solution of the normal coordinates for a hereditary material. A concise treatment of this proof may be found in Fung (1965, sec. 13.5).

Once the form of the kernal $G_{ijk\ell}$ has been established, the Laplace transform gives the operational moduli (Biot, 1958; Schapery, 1962):

$$\bar{G}_{ijk\ell} = \sum_n \frac{pG^n_{ijk\ell}}{p+1/\tau_n} + G^\circ_{ijk\ell} + pG^s_{ijk\ell} \qquad (2.6.5)$$

where τ_n are the relaxation times and p is the Laplace transform variable. Each matrix is symmetric, real, and positive semi-definite; i.e.

$$G^\circ_{ijk\ell} \, , \; G^s_{ijk\ell} \, , \; G^n_{ijk\ell} \geq 0 \qquad (2.6.6)$$

but the matrix made up of the sum is positive definite:

$$[\sum_n G^n_{ijk\ell} + G^\circ_{ijk\ell} + G^s_{ijk\ell}]e_{ij}e_{k\ell} > 0 \qquad (2.6.7)$$

If the operational moduli $G_{ijk\ell}$ is now substituted into the operational variational principle, one can now prove the following conjecture (Schapery, 1962, 1964):

$$q = \sum_n S^n(1-e^{-t/\gamma_n}) + S^\circ + S^s t \qquad (2.6.8)$$

where γ_n are the relaxation times for the series. We have assumed the relaxation times are independent of position within any one element and a finite number of degrees of freedom are used for q. These assumptions are completely consistent with the finite element method.

This relationship suggests a very simple procedure to calculate the time-dependent displacements (or stress) from our operational-variational principle. Provided that the undisturbed linear visccelastic body is subject to prescribed loads and displacements which are step functions of time

applied at time zero, we can denote the stress or displacement response by

$$\psi(t) = \psi^{\circ}+\psi^{s}t + \Delta\psi(t) \qquad (2.6.9)$$

where ψ° and ψ^{s} are constant with respect to time and $\Delta\psi(t)$ is the transient response:

$$\Delta\psi(t) = \int_0^{\infty}\phi(\tau)e^{-t/\tau}d\tau \qquad (2.6.10)$$

$\phi(\tau)$ represents the temporal spectral distribution function of the variable τ. The function may consist of discrete frequencies represented by Dirac delta functions, ie.

$$\phi(\tau) = \sum_{i=1}^{n}\phi_i\ \delta(\tau-\tau_i) \qquad (2.6.11)$$

One obtains then the series representation as in equation (2.6.4):

$$\Delta\psi(t) = \sum_{i=1}^{n}\phi_i\ e^{-t/\tau_i} \qquad (2.6.12)$$

Representing the displacements or stress response by equation (2.6.9) for $\psi(t)$ and using $\Delta\psi(t)$ implies that the Laplace transform $\bar{\psi}(t)$ has singularities only on the non-positive real p axis, and that all poles are simple, except at the origin where a double pole is possible (Schapery, 1962, 1964).

The simplicity of the Laplace transform of $\psi(t)$ suggests that a reasonable approximation to the displacement solution is possible using collocation or least-squares (Schapery, 1961, 1962). If the transient response is given by the

prony series

$$\Delta\psi_D = \sum_{i=1}^{n} S_i\left(1 - e^{-t/\gamma_i}\right) \qquad (2.6.13)$$

where γ_i are prescribed relaxation times, the unspecified coefficients S_i are readily calculated by minimizing the total squared error between the actual displacements $\Delta\psi$ and the calculated displacement $\Delta\psi_D$. This total squared error is

$$E^2 = \int_0^\infty [\Delta\psi - \Delta\psi_D]^2 dt \qquad (2.6.14)$$

with the minimization yielding

$$-1/2\ \frac{\partial E^2}{\partial S_i} = 0 = \int_0^\infty [\Delta\psi - \Delta\psi_D] e^{-t/\gamma_i} dt \quad i=1,\ldots,n \qquad (2.6.15)$$

Collocation results in n relations between the Laplace transform of $\Delta\psi$ and $\Delta\psi_D$ evaluated at $1/\gamma_i$:

$$\Delta\bar{\psi}_D(p)_{p=1/\gamma_i} = \Delta\bar{\psi}(p)_{p=1/\gamma_i} \quad i=1,\ldots,n \qquad (2.6.16)$$

Substituting the Laplace transform for $\Delta\psi_D$ and multiplying by p, we obtain a convenient form for the variational solutions:

$$[p\Delta\bar{\psi}(p)]_{p=1/\gamma_j} = \sum_{i=1}^{n} \frac{S_i}{1 + \gamma_i p}_{p=1/\gamma_j} \quad j=1,\ldots,n \qquad (2.6.17)$$

Additional p for evaluation allows calculating ψ° and ψ^s in the series (2.6.9), or using a least-squares solution

to (2.6.17). $p\Delta\psi(p)$ is just the finite element solution using the operational-variational principle for linear viscoelasticity. Thus the coefficients S_i when substituted into the series (2.6.13) and (2.6.9) immediately yield the time-dependent solution.

The total squared error involves both the accuracy of fitting the series to the calculated displacements and the numerical error introduced when the transformed displacements are calculated with the finite element method. Consequently, if an approximate inversion has been obtained with n terms and it is desirable to reduce the squared error by using additional terms, it will often be necessary to evaluate the transform $\Delta\bar{\psi}(p)$ and $\Delta\bar{\psi}_D(p)$ with increased accuracy. When the transforms are calculated with enough numerical accuracy, the total squared error (2.6.14) indicates that, if they are evaluated sufficiently close for $0 \leq p \leq \infty$, the error of the time-dependent approximation is arbitrarily small.

2.7 Boundary Conditions

Application of the finite element method to a problem normally occurring in a half-space entails artificial boundaries within the media. A slice of the media is taken rather than the half-space. In elastostatics we can appeal to the similarity between a fault within a half-space and Saint-Venants

principle: the application of a system of forces statically equivalent to zero force and zero couple to a small part of a body's surface produces strains that are of negligible magnitude at distances which are large compared with the linear dimensions of the part (Fung, 1965). A fault, however, does introduce couples; thus, we must carefully investigate its effect, particularly for time-dependence.

Two approaches to simulate the boundary of a half-space suggest themselves. The first follows from the flexibility afforded by the elements: we grade the element dimensions for the desired resolution. Near the fault the elements are fine, while along the boundary far from the fault, the elements increase in size. This maximizes the distance between the boundary and fault for a given number of unknowns. But this is nothing new.

An alternate strategy simulates the half-space on the boundary. This is analogous to Boussinesq and Cerruti's problem in elasticity: we desire the relation between displacement and force for the surface of an elastic or viscoelastic half-space. Since the two dimensional analogue of Boussinesq's solution does not tend to zero at infinity (Love, 1944 art. 150), we examine the behavior using an empirical relation. Boussinesq and Cerruti's problems indicate a linear relation between surface displacements u and load P for a half-space (Fung, 1965, sec. 8.8).

$$P = \alpha u \qquad (2.7.1)$$

We can postulate a similar relation for the boundary of the region to simulate the infinite continuum. Along the diagonal of the stiffness matrix an additional constant α for the elastic support relates the displacement and reaction. This is equivalent to altering the variational problem for the new boundary condition. A term $1/2\alpha_{ij}u_iu_j$ enters the variational principle along the surface specified by these boundaries.

To estimate the best value for α , we compare an analytic solution for a fault within a half-space to the finite element method. Another comparison uses a second finite element model constructed by reducing the size of the original model. Varying the boundary conditions then allows a best fit for the reduced model to the original. Using this analogy to a half-space, a noticeable improvement occurs for the reduced model.

APPENDIX B

TEST PROBLEM: Pressurization of a viscoelastic cylinder with an elastic case.

We desire a problem with an analytic solution for comparison to the numerical method. A problem often encountered in engineering literature is the pressurization of an infinite viscoelastic cylinder enclosed by an elastic case. Using the correspondence principle of viscoelasticity and the elastic solution, we can compute an analytic solution. The strategy is similar to the operational form of the variational principle, except the Laplace transform of the Euler equations reduce to equations analogous to the elastic problem (Christensen, 1971). Thus elastic solutions are applicable when the elastic moduli are replaced by the corresponding transformed viscoelastic moduli. We then transform the operational solution in the Laplace domain to the time domain. Using this approach, Lee et al. (1959) have calculated the stresses for an externally reinforced viscoelastic cylinder. These results allow direct comparison with our finite element solution.

The numerical solution uses the method previously outlined: finite element solutions in the Laplace domain and a series inversion usiny collocation to the time domain. In addition, a finite element solution using

stepping in the time domain is available (Zienkiewicz, et al., 1968). Assuming a Maxwellian viscoelastic behavior in shear and elastic behavior for the bulk modulus, the constituent relations are:

$$G_{ijk\ell}(t) = [k(t) - \frac{2}{3}\mu(t)]\,\delta_{ij}\delta_{k\ell} + \mu(t)[\delta_{ik}\delta_{j\ell} + \delta_{i\ell}\delta_{jk}] \quad (B.1)$$

where $k(t)$ is the bulk modulus relaxation function and $\mu(t)$ is the shear modulus relaxation function. Therefore one has

$$\sigma_{ij} = \delta_{ij}\int_0^t [k(t-\tau) - \frac{2}{3}\mu(t-\tau)]\frac{\delta\varepsilon_{kk}(\tau)}{\delta\tau}\,d\tau + 2\int_0^t \mu(t-\tau)\frac{\delta\varepsilon_{ij}(\tau)}{\delta\tau}\,d\tau \quad (B.2)$$

where the relaxation functions for our case are

$$k(t) = K \quad (B.3)$$

and

$$\mu(t) = \mu_0 e^{-t/\tau} \quad (B.4)$$

For the thin elastic case, these moduli are set to

$$K = 3.778\times10^7 \text{ lb/in}^2;\ \mu_0 = 1.1525\times10^7 \text{ lb/in}^2;\ \tau = 1. \quad (B.5)$$

The relaxation time τ is in arbitrary units. For the internal viscoelastic cylinder these take the values

$$K = 1\times10^5 \text{ lb/in}^2;\ \mu_0 = .375\times10^5 \text{ lb/in}^2;\ \tau = 10^4 \quad (B.6)$$

Thus the case is essentially elastic.

The element configuration adopted for the problem is given in Figure B.1 . The elements are all constant strain triangles. The boundaries simulate a full cylinder; free slip is specified along the edge of each quadrant. The interior is then given an outward, arbitrary pressure of one unit. Finally, the inversion of stresses to the time domain uses five reduced times.

Figure B.2 illustrates the principle stresses for two times, zero and ten, plotted on the element array. The zero time is obscured by time ten except where a tensional hoop stress occurs along the inner boundary. The ends of the tensional stress are denoted by small asterisks superimposed on the isotropic stress at time ten. The stiffer elastic case receives the brunt of the tensional hoop stresses as the shear modulus relaxes within the viscoelastic cylinder. These results correspond precisely to both our intuition and the analytic solution.

The results for the radial stress, σ_{rr}, are given in Figure B.3 with a comparison to the analytic solution of Lee, et al. (1959). The results correspond very favorably when the individual elements are properly averaged for each radial distance. The elements, constant strain triangles, do not give exceptionally accurate results since stress is constant throughout the element. Better results occur when adjacent elements are averaged to obtain an approximation

Figure B.1

Finite element net for viscoelastic cylinder with elastic case. Using plane strain, one quadrant simulates the full cylinder when slip boundary conditions are applied to the faces. The origin then represents the center of revolution. Shaded elements correspond to the visco-elastic medium; the surrounding, thin elastic case conforms to the outer, unshaded elements. The inner arrows represent the pressurization of the interior at time zero.

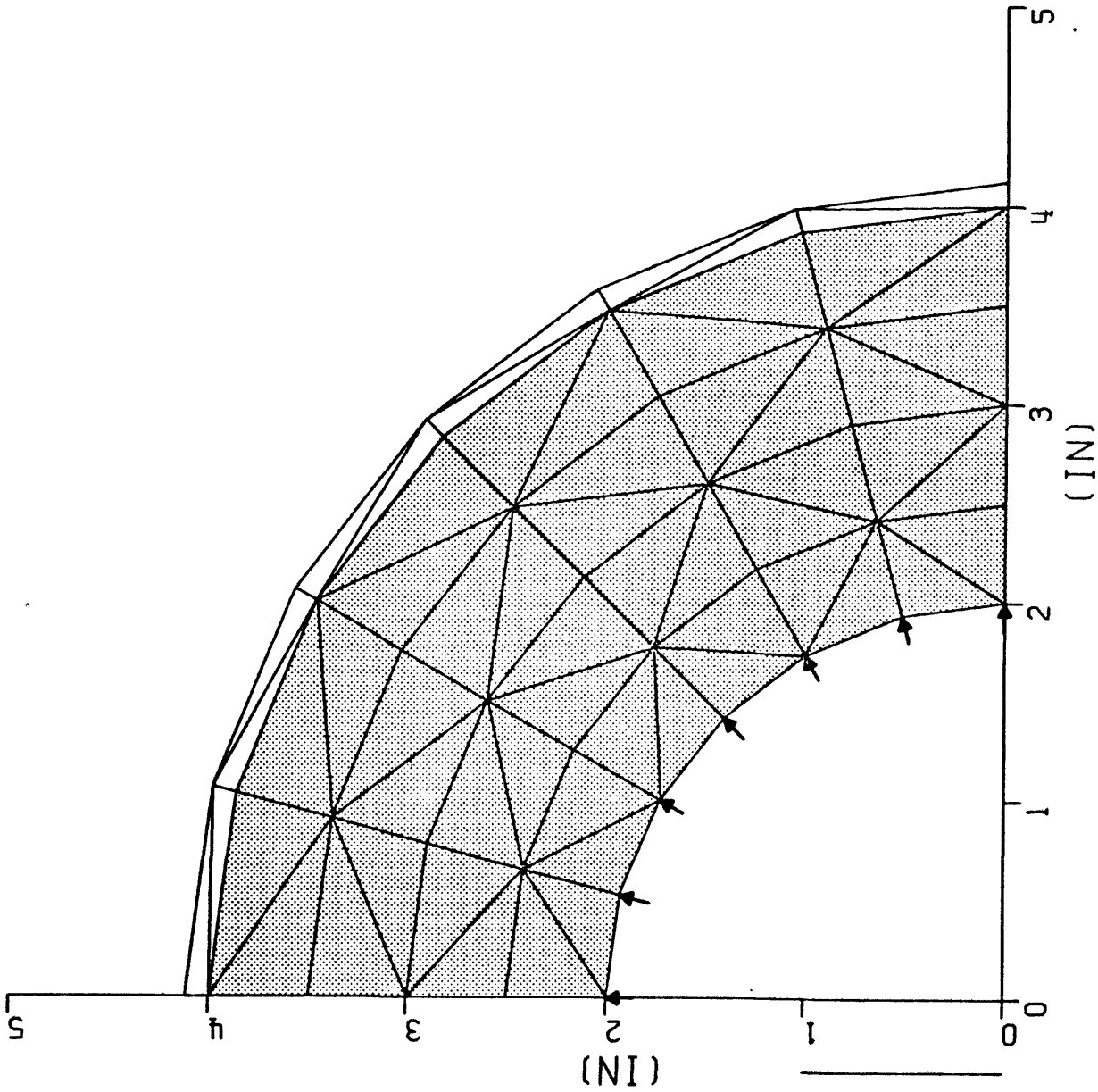


Fig. B11

Figure B.2

Principle stresses plotted on the element array. Two dimensionless times, 0. and 10., are shown in the diagram. The length of the line segment denotes the magnitude normalized by the pressure; the orientations correspond to the minimum and maximum stress. The scale is in the upper right. Asterisks mark the ends of the tensional stress. Note that time 10. generally obscures time 0. except when the asterisks are visible. By time 10. the stress within the viscoelastic cylinder is virtually isotropic; the elastic case absorbs the tensional hoop stress.

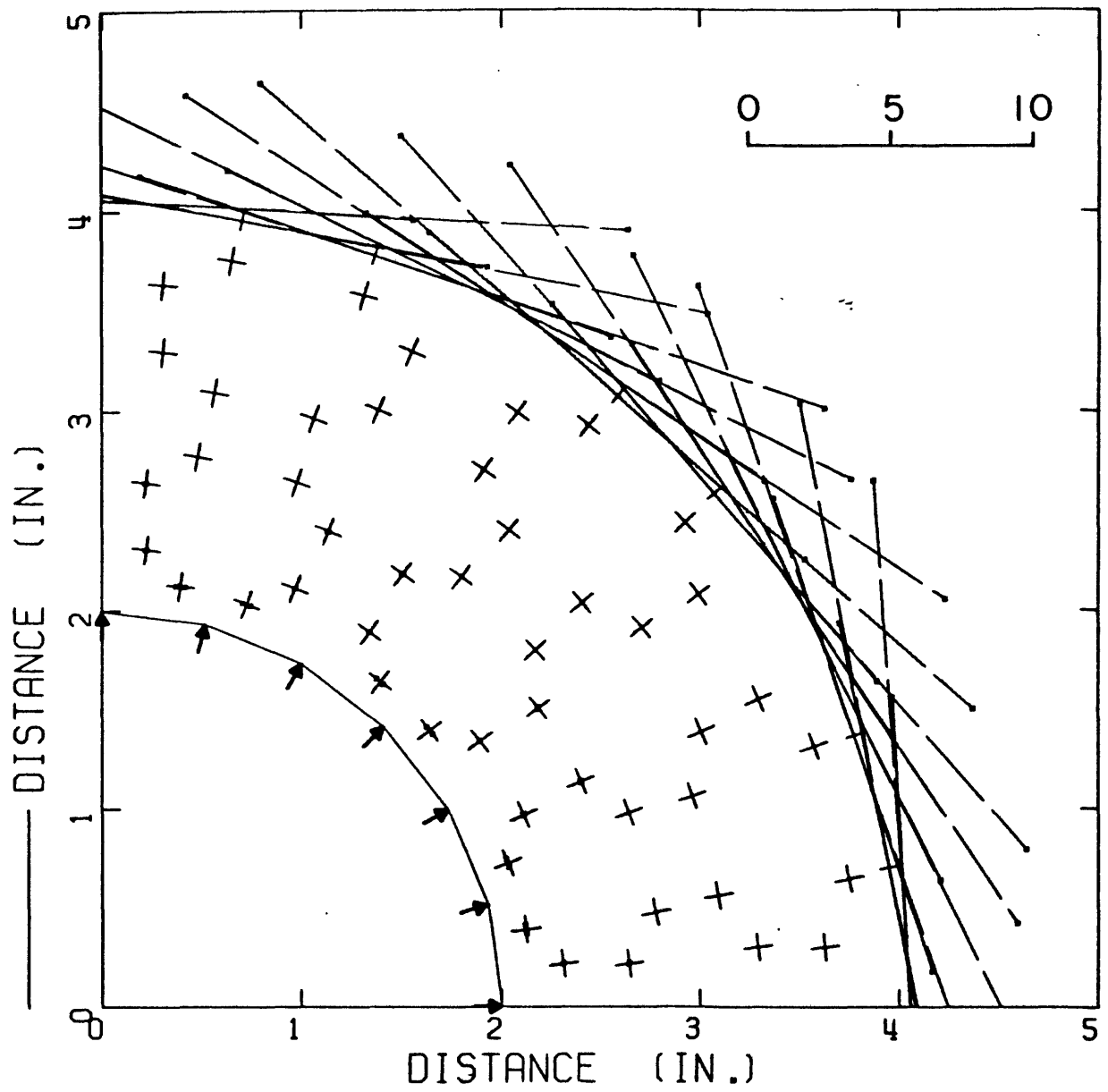


Fig. B.2

Figure B.3

Normalized radial stress from the finite element solution compared to an analytic solution by Lee, et al. (1959) for pressurization of a viscoelastic cylinder with an elastic case. Three times are shown after pressurization: 0., 1., 10. The symbols indicate the finite element solution at specific elements; the lines are the analytic solution. Averaging two adjacent elements yields the best approximation since CST (constant strain triangles) give constant stress. The comparison is then quite good.

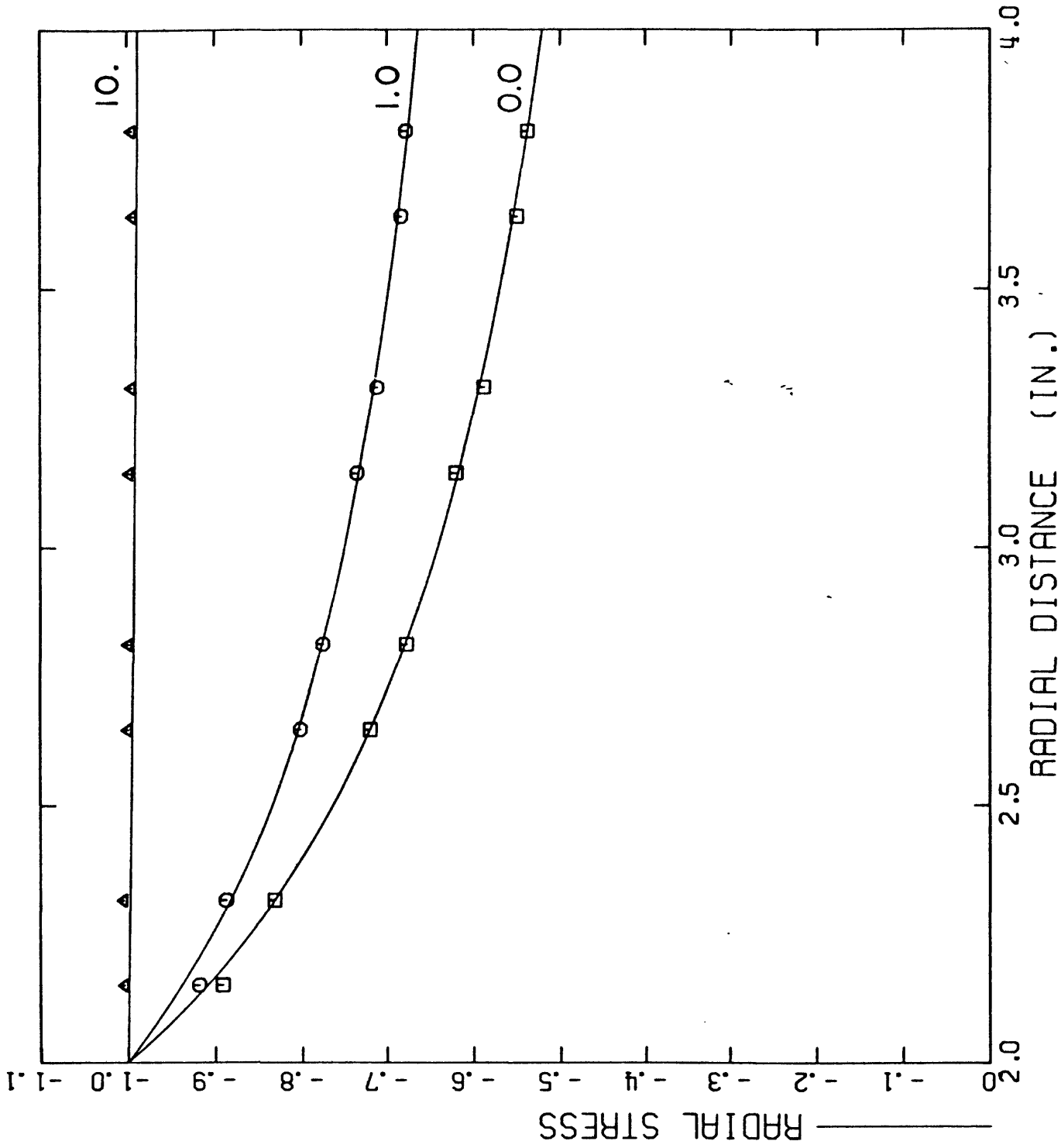

RADIAL STRESS
0
-.1
-.2
-.3
-.4
-.5
-.6
-.7
-.8
-.9
-1.0
-1.1
2.0
2.5
3.0
3.5
4.0
RADIAL DISTANCE (IN.)
0.0
1.0
10.

Fig. B.3

midway between the two (Desai and Abel, 1972). One effectively obtains then a second order approximation similar to a six node triangle.

The solution given by Zienkiewicz, et al. (1968) requires many more individual solutions than the operational strategy. His solutions use time increments of .1; thus, 100 solution steps span the time domain. This is 20 times more than the transformed solution. An iterative solution technique can successfully be applied to save time for both methods. It was not used for the transformed principle since many unique solutions are required for the fault plane inversion at each reduced time. The inversion solutions are also impossible with the time-stepping strategy. The operational strategy requires then only one twentieth the solutions as an efficient stepping method, not to mention the versatility gained for inversion problems and the avoidance of error propagation.

APPENDIX C.1


```
      C  STAGE1? PROBLEM SETUP
           PROGRAM STAGE1(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
          1 TAPE8,TAPE11)
      C
      C  NOTE? LENGTH OF REALK SUFFICENT FOR UP TO LQ IF LK=60000
      C
           DIMENSION REALK(2)
           DIMENSION DUM(100),IDUM(100),NODE(18),UNITS(6)
           DIMENSION XC(6),BOUND(3)
           DIMENSION TITLE(20),COORD(3,8),FBCON(4,2000),IFBCON(4,2000),CC(10)
           LOGICAL OUT,SKIP
           LOGICAL OLDWRK,GMALT,GRAV,GRAVS,GRAVB,OLDSOL,DISP,LOAD,RNODE,
          1 RCOORD
           EQUIVALENCE (REALK(1),INTGRK(1),FBCON(1,1),IFBCON(1,1))
           EQUIVALENCE (DUM(1),IDUM(1))
           EQUIVALENCE (HASH1,IHASH1)
      C
           LEVEL 2,REALK,INTGRK,FBCON,IFBCON
      C
           COMMON/IO/KR,KW,KP,KT1,KT2,KT3,OUT
           COMMON/SIZE/NET,NDT,NETNEW,NDTNEW,LNODNW
           COMMON/BEGIN/ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
           COMMON/END/LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
           COMMON/BLD/KPTR,LREC,IREC
           COMMON/BLDIO/XIOWR(2048)
           COMMON/INDEX/INDEX1(1800)
      C
           COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,NLAP,
          1 NDIM,DISP
           COMMON/GRV/NGRAV,RHOF,DRHO(4),XF(8),NODG(4)
           COMMON/ROT/IROW,JROW,KROW,ZANGLE,YANGLE,XANGLE
           COMMON/BOUND/NBON,TIME,DP
      C
           COMMON/K   /INTGRK(50000)
      C
           EQUIVALENCE (TIME,ITIME)
      C
           DATA KPTR/1/,LREC/2048/,IREC/1/
           DATA LOAD/.FALSE./,RNODE/.FALSE./,RCOORD/.FALSE./,OLDWRK/.FALSE./
           DATA NDIM/2/,OUT/.TRUE./,GMALT/.FALSE./,GRAV/.FALSE./
           DATA GRAVB/.FALSE./,GRAVS/.FALSE./,OLDSOL/.FALSE./
           DATA DISP/.FALSE./,NETNEW/0/,NDTNEW/0/,LNODNW/0/
      C
           NAMELIST/IN/NET,NDT,OLDWRK,NDIM,GMALT,GRAV,GRAVB,GRAVS,OLDSOL,THK,
          1 DISP,RNODE,RCOORD,NDIM,LOAD,NETNEW,NDTNEW,LNODNW
           NAMELIST/IOK/KR,KW,KP,KT1,KT2,KT3,OUT
           NAMELIST/SET/NCON,MASTRL
      C
      C**********************************************************************
      C
      C  SET INITIAL VALUES----------------------------------------------
      C
           LENGTH=3000000
           KR=5
           KW=6
```

C1-2

```
            KP=7
      C  WORKFILE - THESE ARE FIXED FOR READ
            KT1=11
      C  SOLUTION FILE OLD - SRLAD
   60       KT2=12
      C  NEW SOLUTION FILE - SBLD
            KT3=13
            READ(5,IOK)
            CALL OPENMS(11,INDEX1,1000,0)
   65 C
      C  READ MODEL SETUP-----------------------------------------------------
      C
            READ(KR,IN)
            READ(KR,SET)
   70       NET=NET-NETNEW
            NDT=NDT-NDTNEW
            CALL SETUP(LENGTH,NCON,MASTRL,REALK,INTGRK)
            IHASH1=NET**2+NDT
            READ(KR,10)(TITLE(I),I=1,20)
   75    10 FORMAT(20A4)
      C
      C  MASTER ASSEMBLY LIST -------------------------------------------------
      C--   ZERO MASTER ASSEMBLY LIST
            DO 1090 I=IMASTR,LMASTR
   80  1090 INTGRK(I)=0
      C
      C  IF GENERATE, GMALT=T
            IF(GMALT) GOTO 1200
      C
   85       JJ=0
            IC=NET+IMASTR
       1100 CONTINUE
            READ(KR,20)IET,(NODE(I),I=1,18)
            IF(IET.EQ.0)GOTO 1300
   90       IF(IET.LE.NETNEW)GOTO 1100
            IET=IET-NETNEW
            JJ=JJ+1
      20    FORMAT(I4,18I4)
            INTGRK(IMASTR+IET-1)=IC
   95       IF(RNODE)GOTO 1115
      C-----READ DEGREES OF FREEDOM FOR ELEMENT
            II=0
       1110 II=II+1
            IF(NODE(II).EQ.0.OR.II.GT.18)GOTO 1120
  100       IF(NODE(II).LE.NDTNEW)GOTO 9100
            INTGRK(IC)=NODE(II)-NDTNEW
            IC=IC+1
            GOTO 1110
      C-----READ NODES - CONVERT TO DOF
  105  1115 II=0
      C----------TEMPORARY CORRECTION FOR HEXB,STRIKE1
      C1115 II=4
      C-----------
       1116 II=II+1
  110 C--TEMPORARY CORRECTION TO PETERS PROGRAM, NEXT 3 STATEMENTS
```

C1-3

```
      C1115 II=5
      C1116 II=II-1
      C     IF(II.EQ.0)GOTO 1120
            IF(NODE(II).EQ.0.OR.II.GT.18)GOTO 1120
115         NODE(II)=NODE(II)*NDIM-NDIM
            IF(NODE(II).LT.NDTNEW)GOTO 9100
            DO 1117 I=1,NDIM
       1117 INTGRK(IC+I-1)=NODE(II)+I-NDTNEW
            IC=IC+NDIM
120   C-------TEMPORARY CORRECTION-HEX8,STRIKE1
      C     IF(II.EQ.8)II=0
      C     IF(II.EQ.4)II=8
      C-------
            GOTO 1116
125    1120 CONTINUE
       1150 CONTINUE
            GOTO 1100
       1200 CONTINUE
            CALL HALT
130    1300 CONTINUE
            INTGRK(IC)=0
            IC=IC+1
      C CHECK FOR ZERO ELEMENTS
            IF(JJ.NE.NET)WRITE(KW,28)JJ,NET
135      28 FORMAT(* ------ERROR IN ASSEMBLY LIST-------=ELEMENTS,NET*,2I10)
            IF(IC.GT.LMASTR)WRITE(KW,29)IC,LMASTR
         29 FORMAT(* -----ERROR IN ASSEMBLY LIST-------IC,LMASTR*,2I6)
            DO 1320 I=IMASTR,IC
            IF(INTGRK(I).EQ.0)WRITE(KW,30)I,IMASTR,LMASTR,IC,IHASH1,NDTNEW,NDIM,
140        1 NET,IET
         30 FORMAT(* ------ERROR IN ASSEMBLY LIST-----LOCATION=*,I6,/,7I10)
       1320 IHASH1=IHASH1+INTGRK(I)*I
      C
      C  OUTPUT PROBLEM FOR CHECK-----------------------------------------
145   C
            WRITE(KW,110)(TITLE(I),I=1,20)
        110 FORMAT(1H1,20A4)
            WRITE(KW,IN)
            WRITE(KW,120)NET,NDT
150     120 FORMAT(1X,///,*   NET=*,I6,* TOTAL NUMBER OF ELEMENTS*/,
           1 3X,*   NDT=*,I6,* TOTAL NUMBER OF DEGREES OF FREEDOM (CONSTRAINED
           2 AND UNCONSTRAINED)*//)
            WRITE(KW,130)KR,KW,KP,KT1,KT2,KT3
        130 FORMAT(* INPUT-OUTPUT UNITS?  KR=*,I2,* KW=*,I2,* KP=*,I2,/,
155        1 *  KT1=*,I2,* WORKFILE*,/,*  KT2=*,I2,* OLD SOLUTION FILE*,/,
           2 *  KT3=*,I2,* NEW SOLUTION FILE*//)
            WRITE(KW,140)ICON,LCON,IKOUNT,LKOUNT,ILNZ,LLNZ,IMASTR,LMASTR,
           1 IQ,LQ,IK,LK
        140 FORMAT(* ADDRESS INDEX PARAMETERS?  BEGIN/  END*/,
160        1 20X,* ICON/LCON=*,2I6,* GLOBAL NUMBER CONSTRAINT VECTOR*,/,
           2 16X,* IKOUNT/LKOUNT=*,2I6,* DIAG. ADDRESS OF ROW*,/,
           3 20X,* ILNZ/LLNZ=*,2I6,* LEAD NON-ZERO COLUMN, DOF*,/,
           4 16X,* IMASTR/LMASTR=*,2I6,* MASTER ASSEMBLY LIST*,/,
           5 24X,* IQ/LQ=*,2I6,* FORCE/DISPLACEMENT DOF*,/,
165        6 24X,* IK/LK=*,2I6,* MASTER STIFFNESS MATRIX*,///)
```

C1-4

```
      WRITE(KW,150)
  150 FORMAT(1H1,*MASTER ASSEMBLY LIST*)
      IF(OUT)WRITE(KW,155)(INTGRK(I),I=IMASTR,LMASTR)
  155 FORMAT(1H ,2016)
C
C  ALLOCATE SPACE FOR MASTER STIFFNESS MATRIX----------------------------
C
      CALL ORK(LENGTH,REALK,INTGRK)
C
C  WRITE WORKFILE FOR PROBLEM SETUP--------------------------------------
C
      CALL BLD(20,TITLE,KT1),RETURNS(9000)
      CALL BLD(6,KR,KT1),RETURNS(9000)
      CALL BLD(5,NET,KT1),RETURNS(9000)
      CALL BLD(6,ICON,KT1),RETURNS(9000)
      CALL BLD(6,LCON,KT1),RETURNS(9000)
      CALL BLD(11,HASH1,KT1),RETURNS(9000)
      CALL BLD2(LMASTR,REALK(1),KT1),RETURNS(9000)
      LEN=LQ-IQ+1
      CALL BLD2(LEN,REALK(IQ),KT1),RETURNS(9000)

C-----READ COORDINATES
C     INPUT UNITS FOR SMALLEST COORD. FIRST, BOUNDARY IN ORIGINAL
C     UNITS FOR BEND, THEN UNITS FOR LARGER COORD
      READ(KR,*)(UNITS(2*I-1),BOUND(I),UNITS(2*I),I=1,NDIM)
      DO 1390 I=1,NDIM
      XC(2*I)=0.
      XC(2*I-1)=BOUND(I)*(UNITS(2*I)-UNITS(2*I-1))
 1390 CONTINUE
      WRITE(KW,158)(UNITS(2*I-1),UNITS(2*I),BOUND(I),I=1,NDIM)
  158 FORMAT(*0COORDINATE UNITS*,10X,*BOUNDARY IN ORIGINAL UNITS*,/,1X,
     1 1P3E10.2)
      IF(.NOT.RCOORD)GOTO 1500
      II=0
      J=1
 1400 READ(KR,*)LNOD,(COORD(I,J),I=1,NDIM)
      IF(LNOD.EQ.0)GOTO 1495
      IF(LNOD.LE.LNODNW)GOTO 1400
      LNOD=LNOD-LNODNW
      II=II+1
      JJ=IK+(LNOD-1)*NDIM
      DO 1410 I=1,NDIM
 1410 REALK(JJ+I)=COORD(I,J)
      GOTO 1400
 1450 IC=INTGRK(IMASTR-1+LNUM)
      DO 1460 J=1,LNOD
      NOD=INTGRK(IC+NDIM-1)/NDIM
      II=IK+(NOD-1)*NDIM
      DO 1465 I=1,NDIM
 1465 COORD(I,J)=REALK(II+I)
      IC=IC+NDIM
 1460 CONTINUE
      GOTO 1515
 1490 WRITE(KW,160)II,(REALK(IK+I),I=1,NDT)
  160 FORMAT(* ------ERROR IN COORD.------*,I5,/,(1X,1P10E11.3))
```

C 1-5

```
      STOP
 1495 IF(II.NE.NDT/NDIM)GOTO 1490
C  READ ELEMENT INFORMATION AND OUTPUT TO BLD-----------------------------
C
 1500 CONTINUE
      SKIP=.FALSE.
      READ(KR,*)LNUM,LNOD
      IF(LNUM.EQ.0)GOTO 1600
      IF(LNUM.LE.NETNEW)SKIP=.TRUE.
      LNUM=LNUM-NETNEW
      IF(.NOT.RCOORD)READ(KR,*)((COORD(I,J),I=1,NDIM),J=1,LNOD)
      READ(KR,*)KC,NGRAVS,RHO,YGRV
      READ(KR,*)(CC(I),I=1,KC)
      IF(NGRAVS.EQ.0)GOTO 1510
      READ(KR,*)(NODG(I),DRHO(I),I=1,NGRAVS)
  210 FORMAT(2I5)
  215 FORMAT(8E10.4)
  220 FORMAT(I10,7E10.4)
  225 FORMAT(2E10.4,I10)
  230 FORMAT(8E10.4)
 1510 CONTINUE
C READ LOADS AT LOCAL NODE NUMBERS
      IF(.NOT.LOAD)GOTO 1513
      READ(KR,*)NBON
      IF(NBON.EQ.0)GOTO 1513
      READ(KR,*)(IFBCON(1,I),(IFBCON(J+1,I),J=1,NDIM),I=1,NBON)
 1513 CONTINUE
      IF(SKIP)GOTO 1500
      IDUM(1)=LNUM
      IDUM(2)=LNOD
      JJ=2
      IF(RCOORD)GOTO 1450
 1515 CONTINUE
      DO 1520 J=1,LNOD
      DO 1520 I=1,NDIM
      JJ=JJ+1
      II=1
      IF(COORD(I,J).GE.BOUND(I))II=0
 1520 DUM(JJ)=COORD(I,J)*UNITS(2*I-II)+XC(2*I-II)
      JJ=JJ+1
      IDUM(JJ)=KC
      DO 1525 I=1,KC
      JJ=JJ+1
 1525 DUM(JJ)=CC(I)
      JJ=JJ+1
      DUM(JJ)=RHO
      JJ=JJ+1
      DUM(JJ)=YGRV
      JJ=JJ+1
      IDUM(JJ)=NGRAVS
      IF(NGRAVS.EQ.0)GOTO 1535
      DO 1530 I=1,NGRAVS
      JJ=JJ+2
      IDUM(JJ-1)=NODG(I)
 1530 DUM(JJ)=DRHO(I)
```

C1-6

```
      1535 CONTINUE
           IF(.NOT.LOAD)GOTO 1540
           JJ=JJ+1
           IDUM(JJ)=NBON
280        IF(NBON.EQ.0)GOTO 1540
           DO 1539 I=1,NBON
           JJ=JJ+1
           IDUM(JJ)=IFBCON(1,I)
           DO 1539 J=1,NDIM
285        JJ=JJ+1
           DUM(JJ)=FBCON(1+J,I)
      1539 CONTINUE
      1540 CONTINUE
           CALL BLD(JJ,DUM,KT1),RETURNS(9000)
290        IF(OUT)WRITE(KW,241)IDUM(1),IDUM(2),JJ,KC,(DUM(I),I=3,JJ)
       241 FORMAT(1X,4I5,/,(1X,1P10E10.2))
           J=LNOD*NDIM
           IF(OUT)WRITE(KW,242)(INTGRK(INTGRK(IMASTR-1+IDUM(1))-1+I),I=1,J)
       242 FORMAT(1X,10I10)
295   1590 CONTINUE
           GOTO 1500
      1595 WRITE(KW,245)II
       245 FORMAT(* -------ERROR IN ELEMENT INFORMATION----*,I6)
           STOP
300   1600 CONTINUE
      C
      C  READ NODE ROTATION----------------------------------------
      C
           READ(KR,*)NROT,IDROT
305    250 FORMAT(2I10)
           IDUM(1)=NROT
           IDUM(2)=IDROT
           WRITE(KW,252)NROT,IDROT
       252 FORMAT(1H0,*NODE ROTATIONS NROT/IDROT=*,2I6)
310        CALL BLD(2,IDUM,KT1),RETURNS(9000)
           IF(NROT.EQ.0)GOTO 1690
           IF(IDROT.EQ.1)GOTO 1650
           DO 1620 I=1,NROT
           READ(KR,*)N,IROW,JROW,KROW,ZANGLE,YANGLE,XANGLE
315        IROW=IROW-NDTNEW
           JROW=JROW-NDTNEW
           KROW=KROW-NDTNEW
           IF(KROW.LT.0)KROW=0
           WRITE(KW,254)N,IROW,JROW,KROW,ZANGLE,YANGLE,XANGLE
320    254 FORMAT(1X,4I6,1P3E10.2)
       260 FORMAT(4I5,3E10.4)
           CALL BLD(6,IROW,KT1),RETURNS(9000)
      1620 CONTINUE
           GOTO 1690
325   1650 CONTINUE
           DO 1660 I=1,NROT
           READ(KR,*)N,IROW,JROW,KROW,X1,Y1,Z1,X2,Y2,Z2
       270 FORMAT(4I5,6E10.4)
           IROW=IROW-NDTNEW
330        JROW=JROW-NDTNEW
```

C1-7

```
            KROW=KROW-NDTNEW
            IF(KROW.LT.0)KROW=0
      C  ASSUME 2-D
            ZANGLE=ACOS((X2-X1)/SQRT((X2-X1)**2+(Y2-Y1)**2))
335         YANGLE=0.
            XANGLE=0.
            CALL BLD(6,IROW,KT1),RETURN5(9000)
       1660 CONTINUE
       1690 CONTINUE
340   C
      C  BOUNDARY CONDITIONS----------------------------------------
      C  EXTERNAL
      C
            WRITE(KW,304)
345    304 FORMAT(1H0,*BOUNDARY CONDITIONS? EXTERNAL AND INTERNAL, FORCES*)
            I=0
            LEN=0
       2100 CONTINUE
            READ(KR,*)IDOF,ITIME,DP,DPGRAV
350    310 FORMAT(I5,5X,I5,E10.4)
            IF(IDOF.EQ.0)GOTO 2150
            LEN=LEN+4
            I=I+1
            IDOF=IDOF-NDTNEW
355         IFBCON(1,I)=IDOF
            FBCON(2,I)=DPGRAV
            FBCON(3,I)=DP
            IFBCON(4,I)=ITIME
            IF(IDOF.GT.NDT)WRITE(KW,315)IDOF
360    315 FORMAT(* ------ERROR IN BC-----IDOF,IDOF2=*,2I10)
            GOTO 2100
       2150 CONTINUE
            IF(I.NE.0)CALL IORDCO2(I,IFBCON,1,4)
            LEN=4*I
365         IF(LEN.EQ.0)LEN=1
            NECON=I
            WRITE(KW,318)((FBCON(II,J),II=1,4),J=1,NECON)
       318 FORMAT(1X,4(I5,1P2E10.2,I5,2X))
            CALL BLD2(LEN,FBCON,KT1),RETURN5(9000)
370   C
      C  INTERNAL
            I=0
            LEN=0
       2200 CONTINUE
375         READ(KR,*)IDOF,IDOF2,ITIME,DP
       320 FORMAT(3I5,E10.4)
            IF(IDOF.EQ.0)GOTO 2250
            LEN=LEN+4
            I=I+1
380         IDOF=IDOF-NDTNEW
            IDOF2=IDOF2-NDTNEW
            IFBCON(1,I)=IDOF
            IFBCON(2,I)=IDOF2
            FBCON(3,I)=DP
385         IFBCON(4,I)=ITIME
```

C 1-8

```
                IF(IDOF.GT.NDT.OR.IDOF2.GT.NDT)WRITE(KW,315)IDOF,IDOF2
                GOTO 2200
             2250 CONTINUE
                IF(I.NE.0)CALL IORDCO2(1,IFBCON,2,4)
390             LEN=4*I
            C
                IF(LEN.EQ.0)LEN=1
                NICON=I
                WRITE(KW,319)((FBCON(II,J),II=1,4),J=1,NICON)
395          319 FORMAT(1X,4(I5,I10,1PE10.2,I5,2X))
                NBON=NICON+NECON
                WRITE(KW,324)NCON,NBON,NECON,NICON
             324 FORMAT(1X,*DISPLACEMENT CONSTRAINTS REQUESTED=*,I10/
               1 * DISPLACEMENT CONSTRAINTS NEEDED=*,I10/
400            2 * EXTERNAL CONSTRAINTS=*,I10/
               3 * INTERNAL CONSTRAINTS=*,I10)
                IF(NCON.LT.NICON+NECON)WRITE(KW,325)NCON,NECON,NICON
             325 FORMAT(* -----ERROR------NCON/NECON/NICON*,3I6)
                CALL BLD2(LEN,FBCON,KT1),RETURNS(9000)
405         C
            C FORCES Q
             2300 CONTINUE
                READ(KR,*)NBON,ITIME,DP
              330 FORMAT(I5,5X,I5,E10.4)
410             IF(NBON.EQ.0)GOTO 2350
                NBON=NBON-NDTNEW
                WRITE(KW,319)NBON,ITIME,DP
                CALL BLD(3,NBON,KT1),RETURNS(9000)
                GOTO 2300
415          2350 CALL BLD(1,NBON,KT1),RETURNS(9000)
            C
            C  END OF WORKFILE----------  -- --- ----- ---- ----------
            C
                CALL FRC(0,0,KT1),RETURNS(9000)
420         C
                WRITE(KW,340)
              340 FORMAT(1X,///,* ----------STAGE1 COMPLETED----------*)
                STOP
             9000 WRITE(6,9001)LEN
425          9001 FORMAT(* -----ERROR IN BLD-----*,I6)
                STOP
             9100 WRITE(6,9101)NODE(II),IET,IC
             9101 FORMAT(* ----IMPROPER NODE NUMBER----NODE/IET/IC=*,3I10)
                STOP
430             END
```

STATEMENTS BEGINNING AT BELOW LINE NUMBERS ARE UNREACHABLE (DEAD CODE), AND WILL NOT BE PROCESSED
297

SYMBOLIC REFERENCE MAP (R=1)

C1-9

ENTRY POINTS
2237 STAGE1

VARIABLES	SN	TYPE	RELOCATION						
5070	BOUND	REAL	ARRAY		5147	CC	REAL	ARRAY	
5117	COORD	REAL	ARRAY		12	DISP	LOGICAL		PROB
2	DP	REAL		BOUND	4662	DPGRAV	REAL		
2	DRHO	REAL	ARRAY	GRV	4666	DUM	REAL	ARRAY	
0	FICON	REAL	ARRAY	K	6	GMALT	LOGICAL		PROB
3	GRAV	LOGICAL		PROB	5	GRAVB	LOGICAL		PROB
4	GRAVS	LOGICAL		PROB	0	HASH1	REAL		PROB
4631	I	INTEGER			4633	IC	INTEGER		
0	ICON	INTEGER		BEGIN	4661	IDOF	INTEGER		
4664	IDOF2	INTEGER			4651	IDROT	INTEGER		
4666	IDUM	INTEGER	ARRAY		4634	IET	INTEGER		
0	IFBCON	INTEGER	ARRAY	K	0	IHASH1	INTEGER		PROB
4635	II	INTEGER			5	IK	INTEGER		BEGIN
1	IKOUNT	INTEGER		BEGIN	2	ILNZ	INTEGER		BEGIN
3	IMASTR	INTEGER		BEGIN	0	INDEX1	INTEGER	ARRAY	INDEX
0	INTGRK	INTEGER	ARRAY	K	4	IQ	INTEGER		BEGIN
2	IREC	INTEGER		BLD	0	IROW	INTEGER		ROT
1	ITIME	INTEGER		BOUND	4640	J	INTEGER		
4632	JJ	INTEGER			1	JROW	INTEGER		ROT
4644	KC	INTEGER			2	KP	INTEGER		IO
0	KPTR	INTEGER		BLD	0	KR	INTEGER		IO
2	KROW	INTEGER		ROT	3	KT1	INTEGER		IO
4	KT2	INTEGER		IO	5	KT3	INTEGER		IO
1	KW	INTEGER		IO	0	LCON	INTEGER		END
4637	LEN	INTEGER			2	LENGTH	INTEGER		PROB
5	LK	INTEGER		END	1	LKOUNT	INTEGER		END
2	LLNZ	INTEGER		END	3	LMASTR	INTEGER		END
4641	LNOD	INTEGER			4	LNODNW	INTEGER		SIZE
4642	LNUM	INTEGER			3505	LOAD	LOGICAL		
4	LQ	INTEGER		END	1	LREC	INTEGER		BLD
4630	MASTRL	INTEGER			4652	N	INTEGER		
0	NBON	INTEGER		BOUND	4627	NCON	INTEGER		
11	NDIM	INTEGER		PROB	4636	ND1NET	INTEGER	*UNDEF	
1	NDT	INTEGER		SIZE	3	NDTNEW	INTEGER		SIZE
4663	NECON	INTEGER			0	NET	INTEGER		SIZE
2	NETNEW	INTEGER		SIZE	7	NFLT	INTEGER		PROB
0	NGRAV	INTEGER		GRV	4645	NGRAVS	INTEGER		
4665	NICON	INTEGER			10	NLAP	INTEGER		PROB
4643	NOD	INTEGER			5032	NODE	INTEGER	ARRAY	
16	NODG	INTEGER	ARRAY	GRV	4650	NROT	INTEGER		
3510	OLDSOL	LOGICAL			1	OLDWRK	LOGICAL		PROB
6	OUT	LOGICAL		IO	3507	RCOORD	LOGICAL		
0	REALK	REAL	ARRAY	K	4646	RHO	REAL		
1	RHOF	REAL		GRV	3506	RNODE	LOGICAL		
4625	SKIP	LOGICAL			4626	THK	REAL		
1	TIME	REAL		BOUND	5073	TITLE	REAL	ARRAY	
5054	UNITS	REAL	ARRAY		5	XANGLE	REAL		ROT
5062	XC	REAL	ARRAY		6	XF	REAL	ARRAY	GRV
0	XIOWR	REAL	ARRAY	BLDIO	4653	X1	REAL		
4656	X2	REAL			4	YANGLE	REAL		ROT
4647	YGRV	REAL			4654	Y1	REAL		

C1-10

VARIABLES SN TYPE RELOCATION

4657	Y2	REAL	3	ZANGLE	REAL	ROT	
4655	Z1	REAL	4660	Z2	REAL		

FILE NAMES MODE

0	INPUT	445	OUTPUT	1557	TAPE11	0	TAPE5 NAME
445	TAPE6 FMT	1112	TAPE8				

EXTERNALS TYPE ARGS

ACOS	REAL	1 LIBRARY	BLD		3
BLD2		3	FRC		3
IOKDCO2		4	MALT		0
OPENMS		4	ORK		3
SETUP		5	SQRT	REAL	1 LIBRARY

NAMELISTS

IN IOK SET

STATEMENT LABELS

3626	10	FMT		3636	20	FMT		3645	28	FMT	
3661	29	FMT		3703	30	FMT		3716	110	FMT	
3730	120	FMT		3760	130	FMT		4020	140	FMT	
4070	150	FMT		4100	155	FMT		4122	158	FMT	
4146	160	FMT		4212	210	FMT	NO REFS	4214	215	FMT	NO REFS
4216	220	FMT	NO REFS	4221	225	FMT	NO REFS	4224	230	FMT	NO REFS
4252	241	FMT		4264	242	FMT		4272	245	FMT	
4305	250	FMT	NO REFS	4314	252	FMT		4345	254	FMT	
4350	260	FMT	NO REFS	4370	270	FMT	NO REFS	4376	304	FMT	
4414	310	FMT	NO REFS	4423	315	FMT		4435	318	FMT	
4464	319	FMT		4450	320	FMT	NO REFS	4477	324	FMT	
4525	325	FMT		4541	330	FMT	NO REFS	4555	340	FMT	
0	1090			2300	1100			2316	1110		
2326	1115			2333	1116			0	1117		
2346	1120			0	1150		INACTIVE	2347	1200		
2351	1300			0	1320			0	1390		
2525	1400			0	1410			2545	1450		
0	1460			0	1465			2573	1490		
2613	1495			2617	1500			2701	1510		
2735	1513			2743	1515			0	1520		
0	1525			0	1530			3007	1535		
0	1539			3026	1540			0	1590		INACTIVE
0	1595	INACTIVE		3062	1600			0	1620		
3114	1650			0	1660			3143	1690		
3147	2100			3171	2150			3212	2200		
3234	2250			3264	2300			3275	2350		
3304	9000			4567	9001	FMT		3307	9100		
4602	9101	FMT									

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
2272	1090	I	79 80	2B	INSTACK
2342	1117	I	117 118	3B	INSTACK
2365	1320	I	138 142	11B	EXT REFS
2454		I	190 190	11B	EXT REFS
2473	1390	I	191 194	4B	INSTACK
2503		I	195 195	11B	EXT REFS
2543	1410	I	207 208	2B	INSTACK

C) 11

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES		
2555	1460	J	211 217	16B		NOT INNER	
2565	1465	I	214 215	2B	INSTACK		
2601		I	219 219	10B		EXT REFS	
2644		J	231 231	10B		EXT REFS	
2670		I	235 235	10B		EXT REFS	
2711		I	246 246	23B		EXT REFS	NOT INNER
2720		J	246 246	10B		EXT REFS	
2750	1520	J	254 259	13B		NOT INNER	
2752	1520	I	255 259	10B	INSTACK		
2767	1525	I	262 264	2B	INSTACK		
3003	1530	I	272 275	3B	INSTACK		
3017	1539	I	281 287	7B		NOT INNER	
3022	1539	J	284 287	2B	INSTACK		
3050		I	293 293	10B		EXT REFS	
3076	1620	I	313 323	16B		EXT REFS	EXITS
3115	1660	I	326 338	26B		EXT REFS	EXITS

COMMON BLOCKS	LENGTH
IO	7
SIZE	5
BEGIN	6
END	6
BLD	3
BLDIO	2048
INDEX	1000
PROB	11
GRV	18
ROT	6
BOUND	3
K	50000 LCM

STATISTICS

PROGRAM LENGTH	3273B	1723
BUFFER LENGTH	1666B	950
SCM LABELED COMMON LENGTH	6051B	3113
LCM LABELED COMMON LENGTH	141520B	50000

100000B SCM USED

C1-12

```
1               SUBROUTINE BLD(LEN,X,NF),RETURNS(R1)
                DIMENSION X(1),XIO(2),NIO(2)
                EQUIVALENCE (NIO,XIO)
          C
5               COMMON/BLDIO/XIO
                COMMON/BLD  /KPTR,LREC,IREC
          C
                LREC1=LREC-1
                IF(KPTR.EQ.0.OR.NF.EQ.0)RETURN R1
10              IF(LEN.LT.1)RETURN R1
                IF(LEN.GT.LREC1)GOTO 5
                JPTR=KPTR+LEN
                IF(JPTR.LE.LREC)GOTO 30
              5 IE=LREC-KPTR
15           10 IS=1
                NIO(KPTR)=LEN
                IF(IE.LT.1)GOTO 21
             15 DO 20 I=IS,IE
             20 XIO(KPTR+I)=X(I)
20           21 IF(LEN-IE.LE.0)GOTO 25
                CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                IS=IE
                IE=LEN
25              IF(LEN-IS.GT.LREC)IE=LREC+IS
                IS=IS+1
                KPTR=1-IS
                GOTO 15
             25 KPTR=IE+1+KPTR
30              IF(KPTR.GT.LREC)GOTO 26
                RETURN
             26 CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                KPTR=1
35              RETURN
             30 NIO(KPTR)=LEN
                DO 40 I=1,LEN
             40 XIO(KPTR+I)=X(I)
                KPTR=JPTR+1
40              IF(KPTR.GT.LREC)GOTO 26
                RETURN
                ENTRY FRC
                NIO(KPTR)=-2
                CALL WRITMS(NF,XIO,LREC,IREC,0)
45              IREC=IREC+1
                KPTR=0
                RETURN
                END
```

SYMBOLIC REFERENCE MAP (R=1)

C1-13

ENTRY POINTS
3 BLD 113 FRC

VARIABLES	SN	TYPE		RELOCATION					
146	I	INTEGER			144	IE	INTEGER		
2	IREC	INTEGER		BLD	145	IS	INTEGER		
143	JPTR	INTEGER			0	KPTR	INTEGER		BLD
0	LEN	INTEGER		F.P.	1	LREC	INTEGER		BLD
142	LREC1	INTEGER			0	NF	INTEGER		F.P.
0	NIO	INTEGER	ARRAY	BLDIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BLDIO

EXTERNALS	TYPE	ARGS
WRITMS		5

STATEMENT LABELS

32	5	0	10	INACTIVE	37	15	
0	20	46	21		65	25	
72	26	100	30		0	40	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
44	20	I	18 19	2B	INSTACK
105	40	I	37 38	2B	INSTACK

COMMON BLOCKS	LENGTH
BLDIO	2
BLD	3

STATISTICS

PROGRAM LENGTH	147B	103
SCM LABELED COMMON LENGTH	5B	5

100000B SCM USED

C 1-14

```
      SUBROUTINE BLD2(LEN,X,NF),RETURNS(R1)
      DIMENSION X(1),XIO(2),NIO(2)
      LEVEL 2,X
      EQUIVALENCE (NIO,XIO)
      COMMON/BLDIO/XIO
      COMMON/BLD/KPTR,LREC,IREC
C
      LREC1=LREC-1
      IF(KPTR.EQ.0.OR.NF.EQ.0)RETURN R1
      IF(LEN.LT.1)RETURN R1
      IF(LEN.GT.LREC1)GOTO 5
      JPTR=KPTR+LEN
      IF(JPTR.LE.LREC)GOTO 30
    5 IE=LREC-KPTR
   10 IS=1
      NIO(KPTR)=LEN
      IF(IE.LT.1)GOTO 21
   15 DO 20 I=IS,IE
   20 XIO(KPTR+I)=X(I)
   21 IF(LEN-IE.LE.0)GOTO 25
      CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      KPTR=1-IS
      GOTO 15
   25 KPTR=IE+1+KPTR
      IF(KPTR.GT.LREC)GOTO 26
      RETURN
   26 CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      KPTR=1
      RETURN
   30 NIO(KPTR)=LEN
      DO 40 I=1,LEN
   40 XIO(KPTR+I)=X(I)
      KPTR=JPTR+1
      IF(KPTR.GT.LREC)GOTO 26
      RETURN
      ENTRY FRC2
      NIO(KPTR)=-2
      CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      KPTR=0
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

C1-15

ENTRY POINTS
3 BLD2 113 FRC2

VARIABLES	SN	TYPE		RELOCATION					
146	I	INTEGER			144	IE	INTEGER		
2	IREC	INTEGER		BLD	145	IS	INTEGER		
143	JPTR	INTEGER			0	KPTR	INTEGER		BLD
0	LEN	INTEGER		F.P.	1	LREC	INTEGER		BLD
142	LREC1	INTEGER			0	NF	INTEGER		F.P.
0	NIO	INTEGER	ARRAY	BLDIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BLDIO

EXTERNALS	TYPE	ARGS
WRITMS		5

STATEMENT LABELS

32	5	0	10	INACTIVE	37	15	
0	20	46	21		65	25	
72	26	100	30		0	40	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
44	20	I	18 19	2B	INSTACK
105	40	I	37 38	2B	INSTACK

COMMON BLOCKS	LENGTH
BLDIO	2
BLD	3

STATISTICS

PROGRAM LENGTH	147B	103
SCM LABELLED COMMON LENGTH	5B	5

100000B SCM USED

C1-16

```
      C-----------------------------------------------------------------
            SUBROUTINE ORK(LENGTH,REALK,INTGRK)
      C-----3/25/74-----DOES NOT ZERO IK-LK
      C FINITE ELEMENT ANALYSIS BASIC LIBRARY SUBROUTINE-VERSION 2
      C*****************************************************************
      C     THIS SUBROUTINE CREATES
      C             1) THE LNZ VECTOR WHICH HOLDS THE COLUMN NUMBER
      C                OF THE LEADING NON-ZERO ENTRY IN EACH ROW OF
      C                THE ASSEMBLED [K[ MATRIX
      C             2) THE ADDRESS COUNT VECTOR WHICH HOLDS THE ABSOLUTE
      C                ADDRESS OF THE DIAGONAL ENTRY FOR EACH ROW OF THE
      C                ASSEMBLED K MATRIX, MINUS THE ROW NUMBER, AND
      C                CHECKS THAT K FITS IN THE /DATA/ VECTOR
            DIMENSION REALK(2),INTGRK(2)
      C
            LEVEL 2,REALK,INTGRK
      C
            LOGICAL OUT
            COMMON /IO/ KR, KW, KP, KT1, KT2, KT3, OUT
            COMMON /SIZE/ NET, NDT
            COMMON /BEGIN/ ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
            COMMON /END/ LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
      C
      C VERSION 2 RELEASE 1..CORRECTIONS TO SEG. NOS. 68 THRU 74 (MARCH 1973)
      C
      C
      C PRINT CONTROL
        100 FORMAT(49H0THE LENGTH OF THE [DATA[ VECTOR FOR THIS CASE IS,I10,14
           1H WHICH EXCEEDS,I10,49H =THE MAXIMUM ALLOWED IN THE DIMENSION STAT
           2EMENT./39H EXECUTION TERMINATED IN SUBROUTINE ORK)
        200  FORMAT(5(4X,$ROW$,2X,$ILNZ$,2X,$ADR. DIAG.$,1X))
        300  FORMAT(5(1X,2I6,I10,3X))
        400 FORMAT(10H0THERE ARE,I10,68H NON-ZERO ENTRIES IN [K[. IF [K[ WERE
           1FULLY POPULATED THERE WOULD BE,I10, 9H ENTRIES.,/,20X,15HTHE DENSI
           2TY IS ,E15.6)
            ILNZM1=ILNZ-1
            IMSTM1=IMASTR-1
            IKOUM1=IKOUNT-1
            NETM1=NET-1
            IMSTP1=IMASTR+1
      C SET EACH LNZ COLUMN NO = ROW NO (DIAGONAL MATRIX)
            DO 30 IROW=1,NDT
            MSUB=ILNZM1+IROW
        30  INTGRK(MSUB)=IROW
      C EXAMINE MASTER ASSEMBLY LIST, ONE ELEMENT AT A TIME, TO CREATE
      C THE LNZ VECTOR
            DO 20 LNUM=1,NET
            MADDR = IMSTM1+LNUM
            MADDR = INTGRK(MADDR)-1
      C CALCULATE NO. DOF IN THE ELEMENT BY DIFFERENCING POINTERS, OR ...
            I = IMASTR+LNUM
            IF(LNUM.EQ.NET) GO TO 3
            NDE = INTGRK(I)-INTGRK(I-1)
            GO TO 4
      C ... BEGIN BY ASSUMING THE LIST IS FILLED, FOR LAST ELEMENT
```

C1-17

```
      3    NDL = LMASTR-INTGRK(I-1)+1
     C INITIALIZE SMALLEST DOF NO. AT LARGEST POSSIBLE VALUE
      4    ISMALL=NDT
     C     FIND SMALLEST MASTR NUMBER FOR THIS ELEMENT
60           DO 5 JDOF=1,NDE
             INDEX=MADDR+JDOF
             IF(INTGRK(INDEX).GT.ISMALL) GO TO 5
     C DISCONTINUE SEARCH IF A ZERO IS FOUND, INDICATING EXCESS STORAGE
     C AND PREMATURE END OF LIST FOR LAST ELEMENT
65           IF(INTGRK(INDEX).EQ.0) GO TO 6
             ISMALL=INTGRK(INDEX)
      5    CONTINUE
     C     FIND COLUMN NUMBER OF LEADING NON-ZERO ENTRY IN ROW
      6    DO 10 JDOF=1,NDE
70           INDEX=MADDR+JDOF
     C DISCONTINUE OPERATION IF EXCESS STORAGE IS DISCOVERED
             IF(INTGRK(INDEX).EQ.0) GO TO 20
             INDEX=ILNZM1+INTGRK(INDEX)
     C CHANGE LNZ COL NO ONLY IF NEW ONE IS LESS THAN OLD ONE
75           IF(INDEX.GT.LENGTH)WRITE(KW,100)INDEX
             IF(INTGRK(INDEX).LT.ISMALL) GO TO 10
             INTGRK(INDEX)=ISMALL
     C
      10   CONTINUE
80    20   CONTINUE
     C CREATE ADDRESS COUNT VECTOR
             INTGRK(IKOUNT) = IK
             INDEX=IKOUNT
             DO 40 IROW=2,NDT
85           I = ILNZM1+IROW
             INTGRK(INDEX+1) = INTGRK(INDEX)+IROW+1-INTGRK(I)
      40   INDEX=INDEX+1
     C ADDRESS COUNT VECTOR NOW CONTAINS ABSOLUTE ADDRESS ONLY FOR THE
     C DIAGONAL ENTRIES, AND THUS INTGRK(LKOUNT) = LK EXACTLY
90           IF (INTGRK(LKOUNT) .LE. LENGTH) GO TO 50
             WRITE (KW,100) INTGRK(LKOUNT), LENGTH
             STOP
      50   LK = INTGRK(LKOUNT)
             IF(OUT)WRITE(KW,200)
95           IF(OUT)WRITE(KW,300)(IROW,INTGRK(ILNZM1+IROW),
            1 INTGRK(IKOUM1+IROW),IROW=1,NDT)
             DO 60 IROW=1,NDT
             J = IKOUM1+IROW
     C       REPLACE THE ABS. ADDRESS OF DIAG. BY (ABS. ADDRESS - ROW NO.)
100          INTGRK(J) = INTGRK(J)-IROW
      60   CONTINUE
             NENTRY = INTGRK(LKOUNT)+NDT-IK+1
             INDEX = (NDT*(NDT+1))/2
             DENS = FLOAT(NENTRY)/FLOAT(INDEX)
105          WRITE (KW,400) NENTRY, INDEX, DENS
             WRITE(KW,410)LK
      410 FORMAT(* END OF K MATRIX---LK=*,I10)
             RETURN
             END
```

C1 - 18

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 ORK

VARIABLES	SN	TYPE		RELOCATION					
347	DENS	REAL			340	I	INTEGER		
0	ICON	INTEGER		BEGIN	5	IK	INTEGER	BEGIN	
331	IKOUM1	INTEGER			1	IKOUNT	INTEGER	BEGIN	
2	ILNZ	INTEGER		BEGIN	327	ILNZM1	INTEGER		
3	IMASTR	INTEGER		BEGIN	330	IMSTM1	INTEGER		
333	IMSTP1	INTEGER			344	INDEX	INTEGER		
0	INTGRK	INTEGER	ARRAY	F.P.	4	IQ	INTEGER	BEGIN	
334	IROW	INTEGER			342	ISMALL	INTEGER		
345	J	INTEGER			343	JDOF	INTEGER		
2	KP	INTEGER		IO	0	KR	INTEGER	IO	
3	KT1	INTEGER		IO	4	KT2	INTEGER	IO	
5	KT3	INTEGER		IO	1	KW	INTEGER	IO	
0	LCON	INTEGER		END	0	LENGTH	INTEGER	F.P.	
5	LK	INTEGER		END	1	LKOUNT	INTEGER	END	
2	LLNZ	INTEGER		END	3	LMASTR	INTEGER	END	
336	LNUM	INTEGER			4	LQ	INTEGER	END	
337	MADDR	INTEGER			335	MSUB	INTEGER		
341	NDE	INTEGER			1	NDT	INTEGER	SIZE	
346	NENTRY	INTEGER			0	NET	INTEGER	SIZE	
332	NETM1	INTEGER			6	OUT	LOGICAL	IO	
0	REALK	REAL	ARRAY	F.P.					

INLINE FUNCTIONS	TYPE	ARGS	
FLOAT	REAL	1	INTRIN

STATEMENT LABELS

37	3		43	4		53	5		
54	6		72	10		76	20		
0	30		0	40		127	50		
0	60		210	100	FMT	233	200	FMT	
241	300	FMT	244	400	FMT	321	410	FMT	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES		
16	30	IROW	42 44	3B	INSTACK		
25	20	LNUM	47 80	57B		EXT REFS	NOT INNER
50	5	JDOF	60 67	4B	INSTACK	EXITS	
57	10	JDOF	69 79	17B		EXT REFS	EXITS
112	40	IROW	84 87	4B	INSTACK		
145		IROW	95 95	14B		EXT REFS	
166	60	IROW	97 101	3B	INSTACK		

COMMON BLOCKS	LENGTH
IO	7
SIZE	2
BEGIN	6
END	6

C1-19

STATISTICS

PROGRAM LENGTH	356B	238
SCM LABELLED COMMON LENGTH	25B	21

100000B SCM USED

C1-20

```
      SUBROUTINE SETUP(LENGTH,NCON,MASTRL,REALK,INTGRK)
C*********************************************************************
C FINITE ELEMENT ANALYSIS BASIC LIBRARY SUBROUTINE-VERSION 2
C*********************************************************************
      DIMENSION REALK(2),INTGRK(2)
      DIMENSION II(6), LI(6), KD(6)
C
      LEVEL 2,REALK,INTGRK
C
      COMMON /IO/ KR, KW, KP, KT1, KT2, KT3
      COMMON /SIZE/ NET, NDT
      COMMON /BEGIN/ ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON /END/ LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
C
C VERSION 2 RELEASE 1 AUGUST 1972
      EQUIVALENCE (ICON,II(1)), (LCON,LI(1)), (KR,KD(1))
C
C PRINT CONTROL
  901 FORMAT (1H0,53X,11HSETUP ENTRY,/,42X,35HUSER SPECIFICATIONS TO FEA
     1BL SYSTEM,/,1H0,51X,16HI/O DEVICE CODES,/,27X,6HREADER,5X,7HPRINTE
     2R,2X,10HCARD PUNCH,7X,5HTAPE1,7X,5HTAPE2,7X,5HTAPE3,/,21X,6(2X,I10
     3),/,1H0,53X,12HPROBLEM SIZE,/,42X,25HTOTAL NUMBER OF ELEMENTS=,I10
     4,/,42X,25HTOTAL DEGREES OF FREEDOM=,I10,/,1H0,51X,16HENTRY PARAMET
     5ERS,/,39X,32HASSUMED LENGTH OF /DATA/ VECTOR=,I10,/,32X,45HNUMBER
     6OF DISPLACEMENT CONSTRAINTS REQUESTED=,I10,/,33X,44HNUMBER OF WORD
     7S REQUESTED FOR ASSEMBLY LIST=,I10)
  902 FORMAT (1H0,43X,31H/DATA/ VECTOR ADDRESS INDEX MAP,/,22X,11HCONSTR
     1AINTS,2X,10HDG ABS ADR,1X,11HLNZE COL NO,1X,11HASMBLY LIST,2X,10HQ
     2/U VECTOR,4X,8HK MATRIX,/,16X,5HBEGIN,6(2X,I10),/,18X,3HEND,5(2X,I
     310),11X,1H+,/,1H0,36H+ LK IS CALCULATED BY SUBROUTINE DRK)
  903 FORMAT (40H0STORAGE EXCEEDS LENGTH OF /DATA/ VECTOR,/,1X,34HLENGTH
     1 SUGGESTED FOR THIS PROBLEM=,I12,8H WORDS,/,1X,40HEXECUTION TERMIN
     2ATED IN SUBROUTINE SETUP)
C **********************************************************************
C REMOVE THIS FORMAT AND WRITE STATEMENT INDICATED BELOW IF FEABL
C HEADING IS NOT DESIRED
C **********************************************************************
C PRINT ENTRY MESSAGE
C **********************************************************************
C REMOVE THIS WRITE STATEMENT IF FEABL HEADING IS NOT DESIRED
C **********************************************************************
      WRITE (KW,901) (KD(I), I = 1,6), NET, NDT, LENGTH, NCON, MASTRL
C CALCULATE ADDRESS INDEX VALUES
      ICON = 1
      LCON = NCON
      IKOUNT = LCON+1
      LKOUNT = LCON+NDT
      ILNZ = LKOUNT+1
      LLNZ = LKOUNT+NDT
      IMASTR = LLNZ+1
      LMASTR = LLNZ+MASTRL
      IQ = LMASTR+1
      LQ = LMASTR+NDT
      IK = LQ+1
C PRINT INDEX MAP
```

C1 = 21

```
                WRITE (KW,902) (II(I), I = 1,6), (LI(I), I = 1,5)
           C STORAGE BOUNDS TEST
                IF (LMASTR .LE. LENGTH) GO TO 1
           C STORAGE EXCEEDED - ESTIMATE REQUIRED LENGTH BASED ON LOWER TRIANGLE
 60        C ESTIMATED POPULATION FACTOR
                TR = (NDT*(NDT+1))/2
                DENS = 0.5
                IF (NDT .GT. 200) DENS=0.3
                IF (NDT .GT. 500) DENS=0.2
 65             IF (NDT .GT. 1500) DENS = 0.15
                IF (NDT .GT. 2000) DENS = 0.10
                TR = TR*DENS
                LENGTH = LQ+TR
                WRITE (KW,903) LENGTH
 70             STOP
           C ENOUGH STORAGE EXISTS TO GO THRU ORK
              1 DO 2 I = ICON,LCON
              2 INTGRK(I) = 0
                RETURN
 75             END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3 SETUP

VARIABLES	SN	TYPE		RELOCATION					
246	DENS	REAL			244	I	INTEGER		
0	ICON	INTEGER		BEGIN	0	II	INTEGER	ARRAY	BEGIN
5	IK	INTEGER		BEGIN	1	IKOUNT	INTEGER		BEGIN
2	ILNZ	INTEGER		BEGIN	3	IMASTR	INTEGER		BEGIN
0	INTGRK	INTEGER	ARRAY	F.P.	4	IQ	INTEGER		BEGIN
0	KD	INTEGER	ARRAY	IO	2	KP	INTEGER		IO
0	KR	INTEGER		IO	3	KT1	INTEGER		IO
4	KT2	INTEGER		IO	5	KT3	INTEGER		IO
1	KW	INTEGER		IO	0	LCON	INTEGER		END
0	LENGTH	INTEGER		F.P.	0	LI	INTEGER	ARRAY	END
5	LK	INTEGER		END	1	LKOUNT	INTEGER		END
2	LLNZ	INTEGER		END	3	LMASTR	INTEGER		END
4	LQ	INTEGER		END	0	MASTRL	INTEGER		F.P.
0	NCON	INTEGER		F.P.	1	NDT	INTEGER		SIZE
0	NET	INTEGER		SIZE	0	REALK	REAL	ARRAY	F.P.
245	TR	REAL							

STATEMENT LABELS

61	1		0	2			70	901	FMT
145	902	FMT	174	903	FMT				

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
65	2	I	72 73	2B	INSTACK

C1-22

COMMON BLOCKS LENGTH

IO	6
SIZE	2
BEGIN	6
END	6

STATISTICS

PROGRAM LENGTH	247B	167
SCM LABELLED COMMON LENGTH	24B	20

100000B SCM USED

C1 23

```
      SUBROUTINE IORDCO2(N,IARRAY,IROW,NROW)
C ORDERS ARRAY ACCORDING TO INDEX IROW IN AN ARRAY
C  IARRAY(IROW,N)
C
      DIMENSION IARRAY(NROW,N)
C
      LEVEL 2,IARRAY
C
      LOGICAL AGAIN
C
    1 LAST=N-1
  100 AGAIN=.FALSE.
      I=1
    2 IF(I.GT.LAST)GOTO 21
      IF(IARRAY(IROW,I+1)-IARRAY(IROW,I))4,16,20
C
    4 DO 6 J=1,NROW
      IX1=IARRAY(J,I)
      IARRAY(J,I)=IARRAY(J,I+1)
    6 IARRAY(J,I+1)=IX1
      GOTO 19
C
   16 IX1=I+1
      IF(IX1.GT.LAST)GOTO 110
      DO 18 II=IX1,LAST
      DO 18 J=1,NROW
   18 IARRAY(J,II)=IARRAY(J,II+1)
  110 N=N-1
      LAST=LAST-1
C
   19 AGAIN=.TRUE.
   20 CONTINUE
      I=I+1
      GOTO 2
C
   21 CONTINUE
      IF(AGAIN)GOTO 1
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 IORDCO2

VARIABLES	SN	TYPE		RELOCATION					
121	AGAIN	LOGICAL			123	I	INTEGER		
0	IARRAY	INTEGER	ARRAY	F.P.	126	II	INTEGER		
0	IROW	INTEGER		F.P.	125	IX1	INTEGER		
124	J	INTEGER			122	LAST	INTEGER		
0	N	INTEGER		F.P.	0	NROW	INTEGER		F.P.

C1 - 24

STATEMENT LABELS

15	1		26	2	0	4	INACTIVE
0	6		45	16	0	18	
73	19		74	20	104	21	
0	100	INACTIVE	70	110			

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
42	6	J	17 20	3B	INSTACK
56	18	II	25 27	12B	NOT INNER
62	18	J	26 27	2B	INSTACK

STATISTICS

PROGRAM LENGTH 135B 93

100000B SCM USED

(BOTTOM OF FILE)

APPENDIX C.2


```
      C  STAGE2 ASSEMBLE FOR EACH TIME AND SOLVE
           PROGRAM STAGE2(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE11=100,
          1 TAPE12=100,TAPE13=100,TAPE20,TAPE15=3000)
      C
      C     UNIT 10 -      DATA
      C     UNIT 11 - KT1 - WORKFILE
      C     UNIT 12 - KT2 - OLD SOLUTION FILE
      C     UNIT 13 - KT3   NEW SOLUTION FILE
      C     UNIT 20 -     SCRATCH
      C
      C  NOTE: LENTH2 SET TO FOOL LENGTH IN STAGE3
      C*****DIMENSION OF REALK MUST BE GREATER THAN LK FROM STAGE1********
      C
      C  DIMENSION DUM .GT. NDT
      C  DIMENSION IADR .GT. NDT
      C  NOTE: FOR INTERNAL BOUNDARIES, EXTRA DOF OF THE ADJACENT NODE
      C     MUST BE ADDED
      C
           DIMENSION COORD(3,3),FBCON(4,2000),TLAPO(20),TLAPN(20),TLAP(20)
           DIMENSION IFBCON(4,100)
           DIMENSION BMAT(200),SM(300),FV(100),B(3,8),C(36),CC(10)
           DIMENSION TITLE(20),INTGRK(2)
           EQUIVALENCE (REALK(1),INTGRK(1))
      C IF PROBLEM GREATER THAN 131K TOTAL, ------ -- --------- - --
      C  REMOVE FOLLOWING EQUIVALENCE STATEMENT BY ADDING COMMENT C,
      C  SET TWOK=.TRUE., AND REMOVE COMMENT C FROM DIMENSION STATEMENT
           EQUIVALENCE (REALK,REALKK)
           DIMENSION REALK(2)
      C--------- ------------------------------  ----------------
           INTEGER RBEGIN,REND,RMAX,RMIN,ENTRY
           LOGICAL GRAV,GRAVS,GRAVB,OLDSOL,DISP
           LOGICAL LOAD,OUT,DEBUG1,DEBUG2,OLDWRK,GMALT,SING
           LOGICAL AGAIN,CREEP,AXISYM,DISK,SPACE,FILL,TWOK
           DIMENSION DUM(1),IDUM(5000)
           EQUIVALENCE (DUM(1),IDUM(1))
           EQUIVALENCE (FBCON(1,1),IFBCON(1,1))
           DIMENSION TEMP(4)
      C
           LEVEL 2,REALKK
           LEVEL 2,REALK,INTGRK
      C
           COMMON/BREAD/NFX,LRECBR,KEYX(20),LENX(20),K1X(20)
           COMMON/BREADIO/XIOBR(2048)
           COMMON/BLD/KPTR,LREC,IREC
           COMMON/BLDIO/XIOWR(2048)
           COMMON/INDEX/INDEX1(1000),INDEX2(1000),INDEX3(1000),INDEXK(1000)
           COMMON/BADR/IADR(5000)
      C
           COMMON/KK  /REALKK(131000)
      C IF TWOK=.FALSE., REMOVE COMMON/K/ BY ADDING COMMENT C-  -----
      C     COMMON/K/REALK(15000)
      C---------------------- ---------------------------------- --
      C
           COMMON/IO/KR,KW,KP,KT1,KT2,KT3,OUT,DEBUG1,DEBUG2
           COMMON/SIZE/NET,NDT,NETNEW,NDTNEW,LNODNW
```

C2-2

```
      COMMON/BEGIN/ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON/END/LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
      COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,NLAP,
     1 NDIM,DISP,NECON,NICON
      COMMON/GRV/NGRAV,RHOF,DRHO(4),XF(8),NODG(4)
      COMMON/ROT/IROW,JROW,KROW,ZANGLE,YANGLE,XANGLE
      COMMON/BOUND/NBON,TIME,DP
      EQUIVALENCE (TIME,ITIME)
      COMMON/KEY/LEN,KEYTMP,KEYRL1,KEYLLQ,KEYPRB,KEYTL,KEYEL,KEYROT,
     1 KEYECO,KEYICO
      COMMON /SHIFT/ KUNIT,KLEN,KBEGIN,KEND,KMIN,KMAX,RBEGIN,REND,
     1 KOFF,DISK,SPACE,FILL,NUMROW,ENTRY
C
C
      DATA LOAD/.FALSE./,SING/.FALSE./,DEBUG1/.FALSE./,DEBUG2/.FALSE./
      DATA NUMROW/20/,KUNIT/10/,KOFF/0/,SPACE/.TRUE./,DISK/.FALSE./
      DATA KEND/0/,KBEGIN/1/
      DATA TEMP/0.,0.,0.,0./,THK/1.0/,DPST/1.E-6/,OLDSOL/.FALSE./
      DATA ENTRY/1/,KOFF/0/,AXISYM/.FALSE./,FILL/.FALSE./
C
      NAMELIST/IN/NET,NDT,OLDWRK,NDIM,GMALT,GRAV,GRAVB,GRAVS,OLDSOL,THK,
     1 DISP,SING,LOAD,DPST,RLAX,AXISYM,LENTHK,NUMROW,ENTRY
      NAMELIST/IOK/KR,KW,KP,KT1,KT2,KT3,OUT,DEBUG1,DEBUG2
C
      LENTH2=300000
C SET TO LENGTH OF REALKK AND SET TWOK=.TRUE. IF BOTH REALK
C   AND REALKK USED
      LENTHK=131000
      TWOK=.FALSE.
C ------------------------------------------------------------------
C
C*********************************************************************
C
C  READ TITLE FROM WORKFILE
C
      READ(5,IOK)
      WRITE(6,IOK)
      KEYTMP=0
      CALL OPENMS(KT1,INDEX1,1000,0)
      CALL OPENMS(KT2,INDEX2,1000,0)
      CALL OPENMS(KT3,INDEX3,1000,0)
      CALL BREAD(KEYTMP,LEN,TITLE,KT1),RETURNS(9000)
C
C
      KEYTMP=0
      CALL BREAD(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
  205 CONTINUE
C
C  READ WORKFILE
C
      KEYTMP=0
  210 CALL BREAD(KEYTMP,LEN,NET,KT1),RETURNS(9000)
      KEYTMP=0
      CALL BREAD(KEYTMP,LEN,ICON,KT1),RETURNS(9000)
      KEYTMP=0
```

C2-3

```
            CALL BREAD(KEYTMP,LEN,LCON,KT1),RETURNS(9000)
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN.HASH1,KT1),RETURNS(9000)
            KEYTMP=0
115         CALL BREAD2(KEYTMP,LEN.REALK(1),KT1),RETURNS(9000)
            KEYRL1=KEYTMP
            KEYTMP=0
            CALL BREAD2(KEYTMP,LEN.REALK(IQ),KT1),RETURNS(9000)
            KEYLLQ=KEYTMP
120         C
            C  WRITE TITLE AND COMPUTATION
            C
            READ(KR,IN)
            READ(KR,*)NNLAP
125         WRITE(6,10)(TITLE(I),I=1,20),NNLAP
          10 FORMAT(1H1,20A4,//,1X,I4,* LAPLACE REDUCED TIMES*)
            READ(KR,*)(TLAP(I),I=1,NNLAP)
            WRITE(6,15)(TLAP(I),I=1,NNLAP)
          15 FORMAT(10X,1PE10.3)
130         NLAP=NNLAP
            WRITE(KW,IN)
            C IF NOT SUFFICIENT SPACE IN REALK FOR K, SET UP FOR SHIFT
            C  REALKK FOR K MATRIX, IKOUNT CHANGED FOR NEW IK
            IK1=IK-1
135         C USE IF TWOK=.TRUE. ----------
            IF(.NOT.TWOK)GOTO 282
            DO 280 I=IKOUNT,LKOUNT
            INTGRK(I)=INTGRK(I)-IK1
          280 CONTINUE
140         LK=LK-IK1
            IK1=0
            IK=1
          282 CONTINUE
            C----------
145         KLEN=LENTHK-IK
            C  ENTRY=0 USES FACTSD,SIMULQ
            C  ENTRY.GT.0  USES SIMULQ2
            IF(ENTRY.GT.0)KLEN=KLEN-NDT
            IF(KLEN.GE.(LK-IK+1))GOTO 330
150         SPACE=.FALSE.
            CALL OPENMS(KUNIT,INDEXR,1000,0)
          330 CONTINUE
            C
            C  IF OLDSOL, SCAN BEGINNING
155         IF(.NOT.OLDSOL) GO TO 1100
            C
           1000 CONTINUE
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
160         KEYTMP=0
            CALL BREAD(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
            IF(DUM(1).EQ.HASH1.AND.LENGTH.LE.LENTH2) GO TO 1010
            WRITE(6,20) DUM(1),HASH1,LENGTH,LENTH2
           20 FORMAT(1H1,*------ -ERROR--------HASH1=*,2(2X,Z8),/,
165          1 * LENGTH/LENTH2=*,2I10)
```

C2-4

```
            STOP
         1010 CONTINUE
            KEYPRB=KEYTMP
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,TLAPO,KT2),RETURNS(9000)
            NLAPO=LEN
            KEYTL=KEYTMP
            WRITE(KW,22)(TLAPO(I),I=1,NLAPO)
            I=1
            J=1
            L=0
         1015 CONTINUE
            L=L+1
            IF(I.GT.NLAP)GO TO 1017
            IF(J.GT.NLAPO)GOTO 1016
            IF(TLAP(I).GT.TLAPO(J))GOTO 1018
         1016 CONTINUE
            TLAPN(L)=TLAP(I)
            IF(TLAP(I).EQ.TLAPO(J))J=J+1
            I=I+1
            GOTO 1015
         1017 IF(J.GT.NLAPO)GOTO 1020
         1018 CONTINUE
            TLAPN(L)=TLAPO(J)
            J=J+1
            GOTO 1015
         1020 CONTINUE
            LEN=L-1
            WRITE(KW,21)LEN
           21 FORMAT(' LAPLACE REDUCED TIMES IN OLD+NEW FILE - LEN=',I6)
            WRITE(KW,22)(TLAPN(I),I=1,LEN)
           22 FORMAT(1X,1P10E11.3)
            GOTO 1110

         C  WRITE SOLUTION FILE
          1100 CONTINUE
            DO 1105 I=1,NLAP
          1105 TLAPN(I)=TLAP(I)
            LEN=NLAP
          1110 CONTINUE
            CALL BLD(20,TITLE,KT3),RETURNS(9000)
            CALL BLD(11,HASH1,KT3),RETURNS(9000)
            CALL BLD(5,NET,KT3),RETURNS(9000)
            CALL BLD(LEN,TLAPN,KT3),RETURNS(9000)
         C
         C  LOOP ON LAPLACE REDUCED TIMES ----------
         C
            L1=1
            DO 3000 L=1,NLAP
         C  ZERO
            DO 900 I=ICON,LCON
            INTGRK(I)=0
          900 CONTINUE
```

C2-5

```
                IF(.NOT.SPACE)GOTO 911
                DO 910 I=IK,LK
                REALKK(I)=0.
              910 CONTINUE
      225       GOTO 919
              911 DO 912 I=IK,LENTHK
              912 REALKK(I)=0.
                DISK=.FALSE.
                REND=0
      230       RBEGIN=1
                KOFF=0
              919 CONTINUE
                DO 920 I=IQ,LQ
                REALK(I)=0.
      235     920 CONTINUE
            C
            C  IF OLDSOL, SCAN FOR STARTING POINT
                IF(.NOT.OLDSOL) GO TO 1200
            C
      240    1120 CONTINUE
                IF(LJ.GT.NLAPO)GOTO 1200
                IF(TLAP(L).LE.TLAPO(LJ))GOTO 1200
             1125 CONTINUE
                IF(LJ.GT.NLAPO) GO TO 1200
      245       DO 1130 N=1,NET
                KEYTMP=0
                CALL BREAD(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
                CALL BLD(LEN,DUM,KT3),RETURNS(9000)
             1130 CONTINUE
      250    1135 CONTINUE
                KEYTMP=0
                CALL BREAD(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
                CALL BLD(LEN,DUM,KT3),RETURNS(9000)
                IF(LEN.EQ.1)GOTO 1140
      255       GOTO 1135
??????????        1140 CONTINUE
                NFLT=IDUM(1)
                LJ=LJ+1
                IF(L.GT.NLAP)GOTO 1125
      260       GO TO 1120
            C
            C  GENERATE AND ASSEMBLE STIFFNESS MATRIX
            C
             1200 CONTINUE
      265       IF(L.GT.NLAP)GOTO 8010
                DO 1290 LL=1,NET
                KEYTMP=0
                IF(L.GT.1.AND.LL.EQ.1)KEYTMP=KEYEL
                CALL BREAD(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
      270       IF(L.EQ.1.AND.LL.EQ.1)KEYEL=KEYTMP
                LNUM=IDUM(1)
                LNOD=IDUM(2)
                JJ=2
                DO 1210 J=1,LNOD
      275       DO 1210 I=1,NDIM
```

C2-6

```
            JJ=JJ+1
            COORD(I,J)=DUM(JJ)
         1210 CONTINUE
            JJ=JJ+1
280         KC=IDUM(JJ)
            DO 1215 I=1,KC
            JJ=JJ+1
            CC(I)=DUM(JJ)
         1215 CONTINUE
285         JJ=JJ+1
            RHO=DUM(JJ)
            JJ=JJ+1
            YGRV=DUM(JJ)
            JJ=JJ+1
290         NGRAVS=IDUM(JJ)
            IF(NGRAVS.EQ.0)GOTO 1225
            DO 1220 I=1,NGRAVS
            JJ=JJ+2
            NODG(I)=IDUM(JJ-1)
295         DRHO(I)=DUM(JJ)
         1220 CONTINUE
         1225 CONTINUE
            IF(.NOT.LOAD)GOTO 1227
            JJ=JJ+1
300         NBON=IDUM(JJ)
            IF(NBON.EQ.0)GOTO 1227
            DO 1226 I=1,NBON
            JJ=JJ+1
            IFBCON(1,I)=IDUM(JJ)
305         DO 1226 J=1,NDIM
            JJ=JJ+1
            FBCON(1+J,I)=DUM(JJ)
         1226 CONTINUE
         1227 CONTINUE
310         NROT=0
            CALL CMAT(KC,CC,C,TLAP,L)
         C
            IF(DEBUG1)WRITE(KW,25)LNUM,(CC(J),J=1,KC),(C(J),J=1,7)
           25 FORMAT(1X,I5,1P10E11.3)
315      C
            IDOF=NDIM*LNOD
            DO 1228 J=1,IDOF
         1228 FV(J)=0.
            IF(LOAD.AND.NBON.NE.0)CALL LOADN(LNUM,NBON,COORD,THK,FBCON,FV,KW,
320        1 OUT)
            IF(AXISYM.AND.LNOD.EQ.4)GOTO 1234
            GOTO (1295,1295,1230,1235,1295,1295,1295,1231),LNOD
         1235 CONTINUE
            CALL QUAD4(COORD,THK,TEMP,C,NROT,NOD,ANGLE,FV,SM,BMAT,LNUM,KW,
325        1 OUT,GRAVB,RHO,YGRV)
            LEN=24
            GOTO 1280
         1234 CALL QUAX4(COORD,1.,1.,1.,0.,C,TEMP,NROT,NOD,ANGLE,FV,SM,
           1 BMAT,LNUM,KW,OUT)
330         LEN=32
```

C2-7

```
            GOTO 1280
      1231  CALL HEX8(COORD,C,NROT,NOD,ZANGLE,YANGLE,XANGLE,
           i SM,BMAT,DUM)
            LEN=150
335         GOTO 1280
      1230  CALL TRIM3(COORD,THK,C,NROT,NOD,ANGLE,FV,SM,BMAT,LNUM,KW,
           1 OUT,GRAVB,RHO,YGRV)
            LEN=10
      1280  CONTINUE
340         IF(GRAV)CALL GRF
            IF(GRAVS.AND.NGRAVS.NE.0)CALL GRS(LNUM,NGRAVS,COORD,THK,SM,YGRV,
           i KW)
            CALL BLD(LEN,BMAT,KT3),RETURNS(9000)
            CALL ASMLTV(LNUM,IDOF,SM,FV,REALK,INTGRK)
345   1290  CONTINUE
            GOTO 1300
      1295  WRITE(KW,23)LNUM,LNOD
        23  FORMAT(* ------ABORT IN EL GEN.----LNUM/LNOD=*,I6)
            STOP
350   C
      C  ROTATE NODES - SPECIFY IN DEGREES
      C
       1300 CONTINUE
            KEYTMP=0
355         CALL BREAD(KEYTMP,LEN,IDUM,KT1),RETURNS(9000)
            KEYROT=KEYTMP
            NROT=IDUM(1)
            IDROT=IDUM(2)
            IF(NROT.EQ.0)GOTO 1400
360         DO 1390 N=1,NROT
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,IROW,KT1),RETURNS(9000)
            CALL ROTATE(N,IROW,JROW,KROW,ZANGLE,YANGLE,XANGLE,REALK,INTGRK)
       1390 CONTINUE
365    1400 CONTINUE
      C
      C  BOUNDARY CONDITIONS - SPECIFY IN ICON,LCON----------
      C  EXTERNAL
      C
370         KEYTMP=0
            CALL BREAD(KEYTMP,LEN,FBCON,KT1),RETURNS(9000)
            KEYECO=KEYTMP
            NECON=LEN/4
            NUM=0
375         DO 1450 I=1,NECON
            NBON=IFBCON(1,I)
            ITIME=IFBCON(4,I)
            DP=FBCON(3,I)
            DPGRAV=FBCON(2,I)
380         IF(ITIME.EQ.4.OR.ITIME.EQ.5)GOTO 1430
            NUM=NUM+1
            INTGRK(ICON-1+NUM)=NBON
            IF(ITIME.EQ.1.OR.ITIME.EQ.3)CALL BTIME(I,FBCON,IFBCON,
           i REALK,+1450)
385         REALK(IQ-1+NBON)=DP
```

C2-8

```
      GOTO 1450
 1430 CONTINUE
      IF(.NOT.SPACE.AND.(NBON.LT.RBEGIN.OR.NBON.GT.REND))
     1 CALL SHIFT(REALKK,INTGRK,NBON,NBON)
      N=IKOUNT-1+NBON
      N=INTGRK(N)+NBON-KOFF
      IF(ITIME.EQ.4)REALKK(N)=REALKK(N)+DPGRAV
      IF(ITIME.EQ.5)REALKK(N)=REALKK(N)*DP+DPGRAV
 1450 CONTINUE
      NECON=NUM
C
C
C  INTERNAL
C
      KEYTMP=0
      CALL BREAD(KEYTMP,LEN,FBCON,KT1),RETURNS(9000)
      KEYICO=KEYTMP
      NICON=LEN/4
      IF(NICON.LE.0) GO TO 1560
C
C WRITE IADR VECTOR FOR CONSTRAINT DOF
      IF(L.NE.1)GOTO 1550
      DO 1520 I=1,NDT
 1520 IADR(I)=0
      DO 1522 I=1,NICON
      N=IFBCON(2,I)
      IF(IFBCON(4,I).GE.10)GOTO 1522
      IADR(N)=I
 1522 CONTINUE
 1550 CONTINUE
C
      DP=TLAP(L)
      TIME=PLAX
      CALL BCONIN(IFBCON,DUM,REALK,INTGRK,NROW,MROW,1.,1.)
 1560 CONTINUE
C
C  APPLY BOUNDARY CONDITIONS
C
      CALL BCON(REALK,INTGRK)
C
C
C  FORCES
C
 1570 CONTINUE
      KEYTMP=0
      CALL BREAD(KEYTMP,LEN,NBON,KT1),RETURNS(9000)
      IF(LEN.EQ.1)GOTO 1580
      IF(ITIME.EQ.2) CALL QTIME
      REALK(IQ-1+NBON)=REALK(IQ-1+NBON)+DP
      GOTO 1570
 1580 CONTINUE
C
C  SAVE Q/U FOR SOLVE AND WRITE
C
      CALL BREAD(KEYLLQ,LEN,DUM,KT1),RETURNS(9000)
```

C2-9

```
            CALL BWRITE2(KEYLLQ,LEN,REALK(IQ),KT1),RETURNS(9000)
            WRITE(6,30)
         30 FORMAT(1 DISPLACEMENT/FORCE VECTOR ----------------------------1)
            IF(OUT)WRITE(6,31)(REALK(I),I=IQ,LQ)
445      31 FORMAT(1H ,1P10E11.3)
      C
      C  FACTOR MATRIX ----------------------------------------
            IF(ENTRY.GT.0)GOTO 1608
      C
450         IF(SING)GOTO 1600
            CALL FACTPD(REALK,INTGRK)
            GOTO 1610
       1600 CALL FACTSD(REALK,INTGRK)
            GOTO 1610
455    1608 ENTRY=1
       1610 CONTINUE
      C
      C  LOOP ON SOLUTIONS ------------------------------------
      C     IFBCON(4,I) = 0 -SET BY INVERSION TO UNIT VALUE
460   C                   1 - SET BY FTIME WITH INVERSION
      C                   2 - NOT INCLUDED IN INVERSION -SET BY INPUT VALUE
      C                       USE FOR DISPLACEMENT SOLUTION
      C                   3 - NOT INCLUDED IN INVERSION - SET BY FTIME
      C                   4 - INVERT DISP., THEN FTIME FOR CREEP
465   C                  10 - STRESS INVERSION
      C
            NSOL=0
            IF(DISP)GOTO 2000
            IF(NICON.LE.0)GOTO 2000
470         CREEP=.FALSE.
            AGAIN=.FALSE.
       1620 DO 1690 I=1,NICON
            ITIME=IFBCON(4,I)
            IF(ITIME.EQ.2.OR.ITIME.EQ.3) GO TO 1600
475         IF(ITIME.EQ.4)CREEP=.TRUE.
            JJ=IFBCON(2,I)
            IF(AGAIN)GOTO 1660
      0
      C----UNIT STRESS INVERSION
480         IF(ITIME.EQ.10)CALL SOLVE2(I,FBCON,IFBCON,REALK,INTGRK,DPST,
           1 .TRUE.,DUM,IDUM),RETURNS(1670,9000)
      C----UNIT CREEP
            IF(ITIME.EQ.1)CALL SOLVE2(I,FBCON,IFBCON,REALK,INTGRK,
           1 FTIME(TLAP(L),FBCON(3,I),IROW),.FALSE.,DUM,IDUM),
485        2 RETURNS(1670,9000)
      C----UNIT DISP. INVERSION
            IF(ITIME.EQ.0)CALL SOLVE2(I,FBCON,IFBCON,REALK,INTGRK,1.,.FALSE.,
           1 DUM,IDUM),RETURNS(1670,9000)
            IF(ITIME.EQ.4)CALL SOLVE2(I,FBCON,IFBCON,REALK,INTGRK,1.,.FALSE.,
490        1 DUM,IDUM),RETURNS(1670,9000)
      C----CREEP PLUS DISP.
       1660 IF(ITIME.EQ.4)CALL SOLVE2(I,FBCON,IFBCON,REALK,INTGRK,
           1 FTIME(TLAP(L),FBCON(3,I),IROW),.FALSE.,DUM,IDUM),
           2 RETURNS(1670,9000)
495         READ(20)
```

C 2 - 10

```
                GOTO 1680
             1670 CONTINUE
                NSOL=NSOL+1
             1680 CONTINUE
500          1690 CONTINUE
                IF(AGAIN)GOTO 3000
                IF(CREEP)AGAIN=.TRUE.
                REWIND 20
                IF(AGAIN)GOTO 1620
505             GO TO 3000
             C
             C  NORMAL DISPLACEMENT SOLUTION
             2000 CONTINUE
                NSOL=1
510             CALL SOLVE2(I,FBCON,IFBCON,REALK,INTGRK,1.,.FALSE.,
               1 DUM,IDUM),RETURNS(3000,9000)
             3000 CONTINUE
                WRITE(6,40)NSOL,L,TLAP(L)
               40 FORMAT(* =)=)=)*,I4,* SOLUTIONS FOR REDUCED TIME=*,14,* COMPLETED
515             1USING TLAP=*,1PE11.4)
                NFLT=NSOL
                CALL BLD(1,NFLT,KT3),RETURNS(9000)
             C
             8000 CONTINUE
520             IF(OLDSOL.AND.LJ.LE.NLAPO)GOTO 1125
             8010 CONTINUE
                LEN=NICON*4
                IF(LEN.EQ.0)LEN=1
                CALL BLD(LEN,IFBCON,KT3),RETURNS(9000)
525             CALL FRC(0,0,KT3)
                WRITE(6,45)NLAP
               45 FORMAT(1X,///,* --------STAGE2 COMPLETED---------*,I4,* REDUCED TI
                1MES*)
                STOP
530          9000 CONTINUE
                WRITE(6,50)LEN,KEYTMP
               50 FORMAT(* ---------ERROR--------IN DREAD--*,218)
                STOP
                END
```

CARD NR. SEVERITY DETAILS DIAGNOSIS OF PROBLEM

534 I 216 NON-INNER LOOP BEGINNING AT THIS CARD IS ENTERED FROM OUTSIDE ITS RANGE.

SYMBOLIC REFERENCE MAP (R=1)

C 2-11

ENTRY POINTS
12335 STAGE2

VARIABLES	SN	TYPE	RELOCATION						
14552	AGAIN	LOGICAL			14603	ANGLE	REAL		
14202	AXISYM	LOGICAL			47376	B	REAL	*UNDEF	
46246	BMAT	REAL	ARRAY		47426	C	REAL	ARRAY	
47472	CC	REAL	ARRAY		46122	COORD	REAL	ARRAY	
14553	CREEP	LOGICAL			7	DEBUG1	LOGICAL		IO
10	DEBUG2	LOGICAL		IO	11	DISK	LOGICAL		SHIFT
12	DISP	LOGICAL		PROB	2	DP	REAL		BOUND
14606	DPGRAV	REAL			14200	DPST	REAL		
2	DRHO	REAL	ARRAY	GRV	14612	DUM	REAL	ARRAY	
15	ENTRY	INTEGER		SHIFT	26422	FBCON	REAL	ARRAY	
13	FILL	LOGICAL		SHIFT	47232	FV	REAL	ARRAY	
6	GMALT	LOGICAL		PROB	3	GRAV	LOGICAL		PROB
5	GRAVB	LOGICAL		PROB	4	GRAVS	LOGICAL		PROB
0	HASH1	REAL		PROB	14561	I	INTEGER		
0	IADR	INTEGER	ARRAY	BADR	0	ICON	INTEGER		BEGIN
14601	IDOF	INTEGER			14604	IDROT	INTEGER		
14612	IDUM	INTEGER	ARRAY		26422	IFBCON	INTEGER	ARRAY	
5	IK	INTEGER		BEGIN	1	IKOUNT	INTEGER		BEGIN
14562	IK1	INTEGER			2	ILNZ	INTEGER		BEGIN
3	IMASTR	INTEGER		BEGIN	5670	INDEXK	INTEGER	ARRAY	INDEX
0	INDEX1	INTEGER	ARRAY	INDEX	1750	INDEX2	INTEGER	ARRAY	INDEX
3720	INDEX3	INTEGER	ARRAY	INDEX	0	INTGRK	INTEGER	ARRAY	KK
4	IQ	INTEGER		BEGIN	2	IREC	INTEGER		BLD
0	IROW	INTEGER		ROT	1	ITIME	INTEGER		BOUND
14564	J	INTEGER			14573	JJ	INTEGER		
1	JROW	INTEGER		ROT	2	KBEGIN	INTEGER		SHIFT
14574	KC	INTEGER			3	KEND	INTEGER		SHIFT
10	KEYECO	INTEGER		KEY	6	KEYEL	INTEGER		KEY
11	KEYICO	INTEGER		KEY	3	KEYLLO	INTEGER		KEY
4	KEYPRB	INTEGER		KEY	2	KEYRL1	INTEGER		KEY
7	KEYROT	INTEGER		KEY	5	KEYTL	INTEGER		KEY
1	KEYTMP	INTEGER		KEY	2	KEYX	INTEGER	ARRAY	BREAD
1	KLEN	INTEGER		SHIFT	5	KMAX	INTEGER		SHIFT
4	KMIN	INTEGER		SHIFT	10	KOFF	INTEGER		SHIFT
2	KP	INTEGER		IO	0	KPTR	INTEGER		BLD
0	KR	INTEGER		IO	2	KROW	INTEGER		ROT
3	KT1	INTEGER		IO	4	KT2	INTEGER		IO
5	KT3	INTEGER		IO	0	KUNIT	INTEGER		SHIFT
1	KW	INTEGER		IO	52	K1X	INTEGER	ARRAY	BREAD
14565	L	INTEGER			0	LCON	INTEGER		END
0	LEN	INTEGER		KEY	2	LENGTH	INTEGER		PROB
14556	LENTHK	INTEGER			14557	LENTH2	INTEGER		
26	LENX	INTEGER	ARRAY	BREAD	14566	LJ	INTEGER		
5	LK	INTEGER		END	1	LKOUNT	INTEGER		END
14570	LL	INTEGER			2	LLNZ	INTEGER		END
3	LMASTR	INTEGER		END	14572	LNOD	INTEGER		
4	LNODNW	INTEGER		SIZE	14571	LNUM	INTEGER		
14175	LOAD	LOGICAL			4	LQ	INTEGER		END
1	LREC	INTEGER		BLD	1	LRECBR	INTEGER		BREAD
14610	MROW	INTEGER			14567	N	INTEGER		
0	NBON	INTEGER		BOUND	11	NDIM	INTEGER		PROB

C 2-12

VARIABLES	SN	TYPE		RELOCATION					
1	NDT	INTEGER		SIZE	3	NDTNEW	INTEGER		SIZE
13	NECON	INTEGER		PROB	0	NET	INTEGER		SIZE
2	NETNEW	INTEGER		SIZE	7	NFLT	INTEGER		PROB
0	NFX	INTEGER		BREAD	0	NGRAV	INTEGER		GRV
14577	NGRAVS	INTEGER			14	NICON	INTEGER		PROB
10	NLAP	INTEGER		PROB	14563	NLAPO	INTEGER		
14560	NNLAP	INTEGER			14602	NOD	INTEGER		
16	NODG	INTEGER	ARRAY	GRV	14600	NROT	INTEGER		
14607	NROW	INTEGER			14611	NSOL	INTEGER		
14605	NUM	INTEGER			14	NUMROW	INTEGER		SHIFT
14201	OLDSOL	LOGICAL			1	OLDWRK	LOGICAL		PROB
6	OUT	LOGICAL		IO	6	RBEGIN	INTEGER		SHIFT
0	REALK	REAL	ARRAY	KK	0	REALKK	REAL	ARRAY	KK
7	REND	INTEGER		SHIFT	14575	RHO	REAL		
1	RHOF	REAL		GRV	14555	RLAX	REAL		
14550	RMAX	INTEGER	*UNDEF		14551	RMIN	INTEGER	*UNDEF	
14176	SING	LOGICAL			46556	SM	REAL	ARRAY	
12	SPACE	LOGICAL		SHIFT	47530	TEMP	REAL	ARRAY	
14177	THK	REAL			1	TIME	REAL		BOUND
47504	TITLE	REAL	ARRAY		46222	TLAP	REAL	ARRAY	
46176	TLAPN	REAL	ARRAY		46152	TLAPO	REAL	ARRAY	
14554	TWOK	LOGICAL			5	XANGLE	REAL		ROT
6	XF	REAL	ARRAY	GRV	0	XIOBR	REAL	ARRAY	BREADIO
0	XIOWR	REAL	ARRAY	BLDIO	4	YANGLE	REAL		ROT
14576	YGRV	REAL			3	ZANGLE	REAL		ROT

FILE NAMES		MODE									
0	INPUT		445	OUTPUT	1112	TAPE11		1323	TAPE12		
1534	TAPE13		2412	TAPE18	1745	TAPE20	UNFMT	0	TAPE5	NAME	
445	TAPE6	MIXED									

EXTERNALS	TYPE	ARGS		
ASMLTV		6	BCON	2
BCONIN		8	BLD	3
BREAD		4	BREAD2	4
BTIME		5	BWRITE2	4
CMAT		5	FACTPD	2
FACTSD		2	FRC	3
FTIME	REAL	3	GRF	0
GRS		7	HEX8	10
LOADN		8	OPENMS	4
OTIME		0	QUAD4	16
QUAX4		16	ROTATE	9
SHIFT		4	SOLVE2	9
TRIM3		15		

NAMELISTS

IN	10K

STATEMENT LABELS

14323	10	FMT		14341	15	FMT		14356	20	FMT	
14376	21	FMT		14411	22	FMT		14432	23	FMT	
14422	25	FMT		14443	30	FMT		14457	31	FMT	
14474	40	FMT		14512	45	FMT		14527	50	FMT	
0	205	INACTIVE		0	210	INACTIVE		0	280		

C2-13

STATEMENT LABELS

12447	282	12462	330	0	900		
0	910	12607	911	0	912		
12616	919	0	920	0	1000	INACTIVE	
12501	1010	12525	1015	12532	1016		
12536	1017	12540	1018	12542	1020		
12555	1100	0	1105	12563	1110		
12623	1120	12631	1125	0	1130		
12645	1135	12655	1140	12663	1200		
0	1210	0	1215	0	1220		
12746	1225	0	1226	12765	1227		
0	1228	13042	1230	13036	1231		
13032	1234	13026	1235	13045	1280		
0	1290	13064	1295	13067	1300		
0	1390	13106	1400	13144	1430		
13170	1450	0	1520	13217	1522		
13222	1550	13227	1560	13231	1570		
13246	1580	13273	1600	13276	1608		
13277	1610	13305	1620	13343	1660		
13357	1670	13361	1680	0	1690		
13373	2000	13376	3000	0	8000		
13414	8010	13426	9000				

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES				
12442	280	I	137 139	2B	INSTACK				
12557	1105	I	204 205	3B	INSTACK				
12575	8000	L	216 519	614B		EXT REFS	ENTRIES	EXITS	NOT INNER
12577	900	I	218 220	2B	INSTACK				
12604	910	I	222 224	2B	INSTACK				
12611	912	I	226 227	2B	INSTACK				
12620	920	I	233 235	2B	INSTACK				
12635	1130	N	245 249	10B		EXT REFS	EXITS		
12670	1290	LL	266 345	174B		EXT REFS	EXITS	NOT INNER	
12715	1210	J	274 278	4B		NOT INNER			
12716	1210	I	275 278	2B	INSTACK				
12726	1215	I	281 284	2B	INSTACK				
12741	1220	I	292 296	4B	INSTACK				
12756	1226	I	302 308	7B		NOT INNER			
12761	1226	J	305 308	2B	INSTACK				
13001	1228	J	317 318	2B	INSTACK				
13076	1390	N	360 364	10B		EXT REFS	EXITS		
13116	1450	I	375 394	55B		EXT REFS			
13207	1520	I	408 409	2B	INSTACK				
13214	1522	I	410 414	5B	INSTACK				
13306	1690	I	472 500	56B		EXT REFS	EXITS		

COMMON BLOCKS	LENGTH
BREAD	62
BREADIO	2048
BLD	3
BLDIO	2048
INDEX	4000
BADR	5000
KK	131000 LCM
IO	9
SIZE	5

C2-14

COMMON BLOCKS	LENGTH
BEGIN	6
END	6
PROB	13
GRV	18
ROT	6
BOUND	3
KEY	10
SHIFT	14

STATISTICS

PROGRAM LENGTH	35732B	15322
BUFFER LENGTH	11602B	4994
SCM LABELED COMMON LENGTH	31703B	13251
LCM LABELED COMMON LENGTH	377670B	131000

130000B SCM USED

C2-15

```
      1              BLOCK DATA IO
                     COMMON/IO/KR,KW,KP,KT1,KT2,KT3,OUT
                     COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,
                    1 NLAP,NDIM,DISP,NECON,NICON
      5              COMMON/BREAD/NFX,LRECBR,KEYX(20),LENX(20),K1X(20)
                     COMMON/BLD/KPTR,LREC,IREC
                C
                     DATA KPTR/1/,LREC/2048/,IREC/1/
                     DATA LRECBR/2048/
     10              DATA NFX/0/,KEYX/20*0/,LENX/20*0/,K1X/20*0/
                     DATA KR/5/,KW/6/,KP/7/,KT1/11/,KT2/12/,KT3/13/,OUT/.FALSE./
                     DATA OLDWRK/.FALSE./
                     END
```

SYMBOLIC REFERENCE MAP (R=1)

VARIABLES	SN	TYPE	RELOCATION					
12	DISP	REAL	PROB	6	GMALT	REAL		PROB
3	GRAV	REAL	PROB	5	GRAVB	REAL		PROB
4	GRAVS	REAL	PROB	0	HASH1	REAL		PROB
2	IREC	INTEGER	BLD	2	KEYX	INTEGER	ARRAY	BREAD
2	KP	INTEGER	IO	0	KPTR	INTEGER		BLD
0	KR	INTEGER	IO	3	KT1	INTEGER		IO
4	KT2	INTEGER	IO	5	KT3	INTEGER		IO
1	KW	INTEGER	IO	52	K1X	INTEGER	ARRAY	BREAD
2	LENGTH	INTEGER	PROB	26	LENX	INTEGER	ARRAY	BREAD
1	LREC	INTEGER	BLD	1	LRECBR	INTEGER		BREAD
11	NDIM	INTEGER	PROB	13	NECON	INTEGER		PROB
7	NFLT	INTEGER	PROB	0	NFX	INTEGER		BREAD
14	NICON	INTEGER	PROB	10	NLAP	INTEGER		PROB
1	OLDWRK	REAL	PROB	6	OUT	REAL		IO

COMMON BLOCKS	LENGTH
IO	7
PROB	13
BREAD	62
BLD	3

STATISTICS

PROGRAM LENGTH	0B	0
SCM LABELED COMMON LENGTH	125B	85

130000B SCM USED

C2-16

```
      SUBROUTINE ASMLTV(LNUM,NDE,ELK,ELQ,REALK,INTGRK)
C
C*********************************************************************
C FINITE ELEMENT ANALYSIS BASIC LIBRARY
C SUBROUTINE FOR ASSEMBLY FOR ELEMENT WITH K IN LTV FORM
C
C .
C VERSION 2 RELEASE 2 (REPLACES ASEMBL IN VERSION 2 RELEASE 1)
C*********************************************************************
      DIMENSION REALK(2), INTGRK(2)
      LEVEL 2,REALK,INTGRK
      LEVEL 2,REALKK
      DIMENSION ELK(2), ELQ(2)
      INTEGER RMAX,RMIN
      LOGICAL SPACE,FILL
C
      COMMON/KK/REALKK(2)
C
      COMMON /BEGIN/ ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON/SHIFT/ KUNIT,KLEN,KBEGIN,KEND,KMIN,KMAX,RBEGIN,REND,
     1 KOFF,DISK,SPACE,FILL
      IKOUM1 = IKOUNT-1
      IQM1 = IQ-1
C SET ADDRESS POINTERS
      IDOF = IMASTR+LNUM-1
      JDOF = INTGRK(IDOF)
      IJ = 1
C
C  TEST IF SHIFT AND SET RMIN AND RMAX FOR ELEMENT
      IF(SPACE)GOTO 101
      RMAX=INTGRK(JDOF)
      RMIN=RMAX
      K=JDOF-1
      DO 100 LROW=1,NDE
      MCOL=INTGRK(K+LROW)
      RMAX=MAX0(MCOL,RMAX)
      RMIN=MIN0(MCOL,RMIN)
  100 CONTINUE
      FILL=.FALSE.
      CALL SHIFT(REALKK,INTGRK,RMIN,RMAX)
  101 CONTINUE
C----
C LOOP OVER LOWER TRIANGLE OF ELEMENT STIFFNESS MATRIX ELK (STORED IN
C LOWER TRIANGLE VECTOR FORM)
      DO 2 LROW = 1,NDE
C ROW MASTER NUMBER
      MROW = INTGRK(JDOF)
C ASSEMBLE FORCE COMPONENT
      K = IQM1+MROW
      REALK(K) = REALK(K)+ELQ(LROW)
C UPDATE ROW MASTER NUMBER LOCATION' SET COLUMN MASTER NUMBER LOCATION
      JDOF = JDOF+1
      KDOF = INTGRK(IDOF)
      DO 2 LCOL = 1,LROW
C COLUMN MASTER NUMBER AND LOCATION UPDATE
```

C2-17

```
                    MCOL = INTGRK(KDOF)
                    KDOF = KDOF+1
                C SORT FOR LARGER ROW/COLUMN NUMBER
                    IROW = MROW
     60             JCOL = MCOL
                    IF (MROW .GE. MCOL) GO TO 1
                    IROW = MCOL
                    JCOL = MROW
                C CALCULATE ABSOLUTE ADDRESS OF K(IROW,JCOL)
     65           1 K = IKOUM1+IROW
                    K = INTGRK(K)+JCOL-KOFF
                C ASSEMBLE STIFFNESS COEFFICIENT
                    REALKK(K) = REALKK(K)+ELK(IJ)
                C UPDATE ADDRESS IN ELEMENT STIFFNESS MATRIX
     70           2 IJ = IJ+1
                    RETURN
                    END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 ASMLTV

VARIABLES	SN	TYPE		RELOCATION					
11	DISK	REAL		SHIFT	0	ELK	REAL	ARRAY	F.P.
0	ELQ	REAL	ARRAY	F.P.	13	FILL	LOGICAL		SHIFT
0	ICON	INTEGER		BEGIN	103	IDOF	INTEGER		
105	IJ	INTEGER			5	IK	INTEGER		BEGIN
101	IKOUM1	INTEGER			1	IKOUNT	INTEGER		BEGIN
2	ILNZ	INTEGER		BEGIN	3	IMASTR	INTEGER		BEGIN
0	INTGRK	INTEGER	ARRAY	F.P.	4	IQ	INTEGER		BEGIN
102	IQM1	INTEGER			114	IROW	INTEGER		
115	JCOL	INTEGER			104	JDOF	INTEGER		
106	K	INTEGER			2	KBEGIN	INTEGER		SHIFT
112	KDOF	INTEGER			3	KEND	INTEGER		SHIFT
1	KLEN	INTEGER		SHIFT	5	KMAX	INTEGER		SHIFT
4	KMIN	INTEGER		SHIFT	10	KOFF	INTEGER		SHIFT
0	KUNIT	INTEGER		SHIFT	113	LCOL	INTEGER		
0	LNUM	INTEGER		F.P.	107	LROW	INTEGER		
110	MCOL	INTEGER			111	MROW	INTEGER		
0	NDE	INTEGER		F.P.	6	RBEGIN	REAL		SHIFT
0	REALK	REAL	ARRAY	F.P.	0	REALKK	REAL	ARRAY	KK
7	REND	REAL		SHIFT	77	RMAX	INTEGER		
100	RMIN	INTEGER			12	SPACE	LOGICAL		SHIFT

EXTERNALS	TYPE	ARGS
SHIFT		4

INLINE FUNCTIONS	TYPE	ARGS							
MAX0	INTEGER	0	INTRIN			MIN0	INTEGER	0	INTRIN

C 2-18

```
STATEMENT LABELS
   62  1                                0  2                                0  100
   35  101

LOOPS  LABEL    INDEX     FROM-TO    LENGTH    PROPERTIES
   24  100      LROW        34 38       4B     INSTACK
   45  2        LROW        45 70      24B              NOT INNER
   56  2        LCOL        54 70      12B     INSTACK

COMMON BLOCKS   LENGTH
       KK            2 LCM
       BEGIN         6
       SHIFT        12

STATISTICS
  PROGRAM LENGTH                121B      81
  SCM LABELED COMMON LENGTH      22B      18
  LCM LABELED COMMON LENGTH       2B       2
        130000B SCM USED
```

C 2 - 19

```
      SUBROUTINE BCON(REALK,INTGRK)
C
C*****************************************************************
C FINITE ELEMENT ANALYSIS BASIC LIBRARY SUBROUTINE-VERSION 2
C
C*****************************************************************
      DIMENSION REALK(2),INTGRK(2),IIROW(10),DIROW(10)
      LEVEL 2,REALK,INTGRK
      LEVEL 2,REALKK
      LOGICAL OUT,SPACE,FILL,DEBUG1,DEBUG2
      INTEGER RBEGIN,REND
      COMMON /IO/ KR, KW, KP, KT1, KT2, KT3, OUT,DEBUG1,DEBUG2
      COMMON /SIZE/ NET, NDT
      COMMON /BEGIN/ ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON /END/ LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
      COMMON /SHIFT/ KUNIT,KLEN,KBEGIN,KEND,KMIN,KMAX,RBEGIN,REND,
     1 KOFF,DISK,SPACE,FILL
      COMMON/KK/REALKK(2)
C
C VERSION 2 RELEASE 1 AUGUST 1972
C
C
C PRINT CONTROL
  901 FORMAT(79H0DISPLACEMENT CONSTRAINTS HAVE BEEN APPLIED TO THE FOLLO
     1WING DEGREES OF FREEDOM,/,5X,7HDOF NO.,2X,12HDISPLACEMENT)
  902 FORMAT(1X,10I10/1X,1P10E10.2)
  903 FORMAT(1X,72H----------------------ABOVE DOF NUMBER APPEARS TWICE
     1 IN CONSTRAINT LIST,/,1X,85HEXECUTION TERMINATED IN SUBROUTINE BCO
     2N DUE TO POSSIBLITY OF BOUNDARY CONDITION ERROR)
  904 FORMAT(62H0YOUR STRUCTURE IS FLYING FREE. PLEASE CONSTRAIN IT NEXT
     1 TIME.,/,1X,28HEXECUTION TERMINATED IN BCON)
      IKOUM1 = IKOUNT-1
      ILNZM1 = ILNZ-1
      IQM1 = IQ-1
C PRINT ENTRY MESSAGE
      WRITE (KW,901)
C ORDER THE CONSTRAINT ROW NUMBERS IN ASCENDING SEQUENCE
      LAST = LCON-1
       IF (LCON .GT. 0) GO TO 1
       WRITE (KW,904)
        STOP
    1 IFLAG = 0
      DO 2 I = ICON,LAST
      IF (INTGRK(I+1) .GE. INTGRK(I)) GO TO 2
      J = INTGRK(I)
      INTGRK(I) = INTGRK(I+1)
      INTGRK(I+1) = J
      IFLAG = 1
    2 CONTINUE
      IF (IFLAG .EQ. 1) GO TO 1
      IE=0
      J=ICON
   50 IF(INTGRK(J).EQ.0.AND.IE.LE.LCON)GOTO 60
      GOTO 70
```

C2-20

```
      60 DO 65 I=J,LAST
      65 INTGRK(I)=INTGRK(I+1)
         IE=IE+1
         INTGRK(LCON)=0
         GOTO 50
      70 CONTINUE
C CHECK TO SEE IF ANY ROW NUMBERS HAVE BEEN ENTERED IN CONSTRAINT
C VECTOR -- ABORT THE RUN IF NONE HAVE BEEN
         J = 0
         DO 100 I = ICON,LCON
     100 J = J+INTGRK(I)
         IF (J .GT. 0) GO TO 200
         WRITE (KW,904)
         STOP
C OUTPUT CONSTRAINT LIST
     200 NROW=1
         DO 4 I = ICON,LCON
         IF (IINTGRK(I) .EQ. 0) GO TO 4
C CHECK FOR REPEATED DOF AFTER 1ST ONE
         IF (I .EQ. ICON) GO TO 3
         IF (INTGRK(I) .NE. INTGRK(I-1)) GO TO 3
         WRITE (KW,903)
         STOP
       3 J = IQM1+INTGRK(I)
         IF(.NOT.OUT)GOTO 4
         IIROW(NROW)=INTGRK(I)
         DIROW(NROW)=REALK(J)
         NROW=NROW+1
         IF(NROW.LT.11)GOTO 4
         WRITE(KW,902)IIROW,DIROW
         NROW=1
       4 CONTINUE
         IF(OUT)WRITE(KW,902)IIROW,DIROW
C LOOP OVER CONSTRAINED DOF FOR MODIFICATION OF COMPLETELY ASSEMBLED
C FORCE VECTOR
         FILL=.FALSE.
         DO 71 I = ICON,LCON
         IF (INTGRK(I) .EQ. 0) GO TO 71
C CHECK IF PRESCRIBED DISPLACEMENT = 0 -- IF IT DOES, SKIP FORCE VECTOR
         NROW = INTGRK(I)
         M = IQM1+NROW
         IF (REALK(M) .EQ. 0.) GO TO 71
C DISPL .NE. 0 -- LOOP OVER ALL ROWS TO MODIFY FORCE VECTOR -- SKIP
C CONSTRAINED ROWS (CONTROLLED BY VALUE OF NEXT)
         NEXT = ICON
         DO 7 NROW = 1,NDT
         IF (NROW .NE. INTGRK(NEXT)) GO TO 5
         NEXT = NEXT+1
         GO TO 7
       5 IF (NROW .GT. NROW) GO TO 51
C CHECK FOR COUPLING OF ROW NROW WITH COL NROW
         J = ILNZM1+NROW
         IF (IINTGRK(J) .GT. NROW) GO TO 7
         IROW = NROW
         ICOL = NROW
```

C 2-21

```
            GO TO 6
          C CHECK FOR COUPLING OF ROW NROW WITH COL NROW
           51 J = ILNZM1+NROW
              IF (INTGRK(J) .GT. NROW) GO TO 7
115           IROW = NROW
              ICOL = NROW
          C SUBTRACT K*(PRESCR DISPL) FROM FORCE VECTOR
            6 KADR = IKOUM1+IROW
              IF(.NOT.SPACE.AND.(IROW.LT.RBEGIN.OR.IROW.GT.REND))
120           1 CALL SHIFT(REALKK,INTGRK,IROW,IROW)
              KADR = INTGRK(KADR)+ICOL-KOFF
              N = IQM1+NROW
              REALK(N) = REALK(N)-REALKK(KADR)*REALK(M)
            7 CONTINUE
125        71 CONTINUE
          C LOOP OVER CONSTRAINED ROWS TO DECOUPLE THEM FROM REST OF K MATRIX
              DO 11 I = ICON,LCON
              IF (INTGRK(I) .EQ. 0) GO TO 11
              NROW = INTGRK(I)
130           INIT = ILNZM1+NROW
              INIT = INTGRK(INIT)
          C SHIFT ROWS IN REALKK IF NECESSARY
              IF(.NOT.SPACE.AND.(NROW.LT.RBEGIN.OR.NROW.GT.REND))
             1 CALL SHIFT(REALKK,INTGRK,NROW,NROW)
135       C SET ROW = 0
              M = IKOUM1+NROW
              M = INTGRK(M)-KOFF
              DO 8 NCOL = INIT,NROW
              KADR = M+NCOL
140           8 REALKK(KADR) = 0.
          C SET COLUMN = 0 IN ROWS WHOSE LNZE COL NO IS .LE. NROW -- SKIP THIS
          C SECTION IF NROW IS THE LAST ROW
              IF (NROW .EQ. NDT) GO TO 10
              INIT = NROW+1
145           DO 9 NROW = INIT,NDT
              N = ILNZM1+NROW
              IF (INTGRK(N) .GT. NROW) GO TO 9
              IF(.NOT.SPACE.AND.(NROW.LT.RBEGIN.OR.NROW.GT.REND))
             1 CALL SHIFT(REALKK,INTGRK,NROW,NROW)
150           KADR = IKOUM1+NROW
              KADR = INTGRK(KADR)+NROW-KOFF
              REALKK(KADR) = 0.
            9 CONTINUE
          C SET DIAGONAL ENTRY =
155        10 CONTINUE
              IF(SPACE)GOTO 101
              IF(NROW.LT.RBEGIN.OR.NROW.GT.REND)
             1 CALL SHIFT(REALKK,INTGRK,NROW,NROW)
              M=IKOUM1+NROW
160           M=INTGRK(M)-KOFF
           101 CONTINUE
              KADR = M+NROW
              REALKK(KADR) = 1.
            11 CONTINUE
165           RETURN
```

C2-22.

```
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 BCON

VARIABLES	SN	TYPE		RELOCATION					
7	DEBUG1	LOGICAL		IO	10	DEBUG2	LOGICAL		IO
515	DIROW	REAL	ARRAY		11	DISK	REAL		SHIFT
13	FILL	LOGICAL		SHIFT	465	I	INTEGER		
475	ICOL	INTEGER			0	ICON	INTEGER		BEGIN
467	IE	INTEGER			464	IFLAG	INTEGER		
503	IIROW	INTEGER	ARRAY		5	IK	INTEGER		BEGIN
460	IKOUM1	INTEGER			1	IKOUNT	INTEGER		BEGIN
2	ILNZ	INTEGER		BEGIN	461	ILNZM1	INTEGER		
3	IMASTR	INTEGER		BEGIN	500	INIT	INTEGER		
0	INTGRK	INTEGER	ARRAY	F.P.	4	IQ	INTEGER		BEGIN
462	IQM1	INTEGER			474	IROW	INTEGER		
466	J	INTEGER			476	KADR	INTEGER		
2	KBEGIN	INTEGER		SHIFT	3	KEND	INTEGER		SHIFT
1	KLEN	INTEGER		SHIFT	5	KMAX	INTEGER		SHIFT
4	KMIN	INTEGER		SHIFT	10	KOFF	INTEGER		SHIFT
2	KP	INTEGER		IO	0	KR	INTEGER		IO
3	KT1	INTEGER		IO	4	KT2	INTEGER		IO
5	KT3	INTEGER		IO	0	KUNIT	INTEGER		SHIFT
1	KW	INTEGER		IO	463	LAST	INTEGER		
0	LCON	INTEGER		END	5	LK	INTEGER		END
1	LKOUNT	INTEGER		END	2	LLNZ	INTEGER		END
3	LMASTR	INTEGER		END	4	LQ	INTEGER		END
472	M	INTEGER			501	MCOL	INTEGER		
471	MROW	INTEGER			477	N	INTEGER		
1	NDT	INTEGER		SIZE	0	NET	INTEGER		SIZE
473	NEXT	INTEGER			470	NROW	INTEGER		
6	OUT	LOGICAL		IO	6	RBEGIN	INTEGER		SHIFT
0	REALK	REAL	ARRAY	F.P.	0	REALKK	REAL	ARRAY	KK
7	REND	INTEGER		SHIFT	12	SPACE	LOGICAL		SHIFT

EXTERNALS	TYPE	ARGS
SHIFT		4

STATEMENT LABELS

21	1		31	2		101	3	
116	4		151	5		170	6	
213	7		0	8		277	9	
302	10		322	11		37	50	
161	51		44	60		0	65	
54	70		221	71		0	100	
316	101		65	200		351	901	FMT
366	902	FMT	372	903	FMT	414	904	FMT

C2-23

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES		
25	2	I	44 50	5B	INSTACK		
45	65	I	56 57	3B	INSTACK		
57	100	I	65 66	2B	INSTACK		
70	4	I	72 87	31B		EXT REFS	
130	71	I	92 125	74B		EXT REFS	NOT INNER
144	7	NROW	101 124	55B		EXT REFS	
225	11	I	127 164	100B		EXT REFS	NOT INNER
250	8	NCOL	138 140	2B	INSTACK		
256	9	NROW	145 153	24B		EXT REFS	

COMMON BLOCKS	LENGTH
IO	9
SIZE	2
BEGIN	6
END	6
SHIFT	12
KK	2 LCM

STATISTICS

PROGRAM LENGTH	527B	343
SCM LABELED COMMON LENGTH	43B	35
LCM LABELED COMMON LENGTH	2B	2

130000B SCM USED

C2-24

```
      SUBROUTINE BCONIN(IFBCON,DUM,REALK,INTGRK,NROW,MROW,DIRV,DIST),
     1 RETURNS(R1)
C----COMPUTES INTERNAL BOUNDARIES
C----ITYPE = IFBCON(4,I)
C     0-INVERSION FOR DISP. VALUE
C     1-SET BY FTIME
C     2-DISP. SET BY INPUT VALUE DP - NO INVERSION
C     3-DISP. SET BY FTIME- NO INVERSION
C     4- SAME AS 2 BUT FOR BC ON ANTI-SYM PROBLEMS
C     10-STRESS INVERSION
C
      DIMENSION REALK(1),INTGRK(1),IFBCON(4,1),DUM(1)
      LEVEL 2,REALK,INTGRK
      LEVEL 2,REALKK
      LOGICAL DISP, ENTRY2,SPACE,FILL
      INTEGER RBEGIN,REND
      EQUIVALENCE (IDP,DP)
      COMMON/SIZE/NET,NDT
      COMMON/BEGIN/ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON/END/LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
      COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,NLAP,
     1 NDIM,DISP,NECON,NICON
      COMMON/BOUND/NBON,RLAX,TLAP
      COMMON /SHIFT/ KUNIT,KLEN,KBEGIN,KEND,KMIN,KMAX,RBEGIN,REND,
     1 KOFF,DISK,SPACE,FILL
      COMMON/BADR  / IADR(1)
      COMMON/KK    / REALKK(2)
C
C*************************************************************************
C
      IKOUN1=IKOUNT-1
      ILNZ1=ILNZ-1
      IQ1=IQ-1
      ENTRY2=.FALSE.
      IF(.NOT.DISP)REWIND 20
C  LOOP OVER INTERVAL BOUNDARIES
      DO 1550 I=1,NICON
      ITYPE=IFBCON(4,I)
      IF(ITYPE.GE.10)GOTO 1550
      IX=0
      JJ=IFBCON(2,I)
      IJ=IFBCON(1,I)
      IF(IJ.GT.JJ)GOTO 1540
      FILL=.FALSE.
      IDP=IFBCON(3,I)
      IF(ITYPE.EQ.3)DP=DP*FTIME(TLAP,RLAX)
 1505 CONTINUE
C
      J1=ILNZ1+JJ
      J1=INTGRK(J1)
      DO 1530 II=J1,NDT
 1510 CONTINUE
C NEW VERSION 9/1/79
C
      IF(JJ.GT.II)GOTO 1515
```

C 2-25

```
      C WORK DOWN A COLUMN----------------
            J=ILN21+II
            J=INTGRK(J)
            IF(J.GT.JJ)GOTO 1530
60          IROW=II
            ICOL=JJ
            IF(.NOT.SPACE.AND.(IJ.LT.RBEGIN.OR.II.GT.REND))
           1 CALL SHIFT(REALKK,INTGRK,IJ,II)
            KADR1=IKOUN1+IROW
65          KADR=INTGRK(KADR1)+ICOL-KOFF
            RK=REALKK(KADR)
            IF(RK.EQ.0.)GOTO 1530
      C OFF DIAGONAL TERMS -- SHIFT ENTRY AND SUM TO DOF
      C--CHECK IF BOTH OFF DIAGONAL AND INTERNAL DOF--
70    C  IF SO, SKIP AND PICK IT UP ACROSS ROW LATER ON
            IF(IADR(IROW).NE.0.AND.IROW.NE.ICOL)GOTO 1530
      C
            IF(ITYPE.EQ.4.AND.IROW.NE.ICOL)RK=-RK
      C
75          IF(J.GT.IJ)GOTO 1540
            KADR=INTGRK(KADR1)+IJ-KOFF
            REALKK(KADR)=REALKK(KADR)+RK
            IF(IROW.NE.ICOL)GOTO 1530
      C DIAGONAL TERM--SPECIAL CASE
80          KADR=IKOUN1+IJ
            KADR=INTGRK(KADR)+IJ-KOFF
            REALKK(KADR)=REALKK(KADR)+RK
            GOTO 1530
      C
85    C WORK ACROSS A ROW----------------------
      C
       1515 CONTINUE
            J=ILN21+JJ
            J=INTGRK(J)
90          IF(J.GT.II)GOTO 1530
            IROW=JJ
            ICOL=II
            IF(.NOT.SPACE.AND.(IJ.LT.RBEGIN.OR.JJ.GT.REND))
           1 CALL SHIFT(REALKK,INTGRK,IJ,JJ)
95          KADR1=IKOUN1+IROW
            KADR=INTGRK(KADR1)+ICOL-KOFF
            RK=REALKK(KADR)
            IF(RK.EQ.0.)GOTO 1530
            INCOL=IADR(ICOL)
100   .     C
      C CHECK TO SEE IF ICOL IS INTERNAL BOUNDARY
            IF(INCOL.EQ.0)GOTO 1520
      C
      C FOR ICOL AN INTERNAL BOUNDARY DOF
105         JCOL=IFBCON(1,INCOL)
            JTYPE=IFBCON(4,INCOL)
      C--PICK UP OFF DIAGONAL TERM PREVIOUSLY SKIPPED
            IF(J.GT.JCOL)GOTO 1540
            IF(JTYPE.EQ.4)RK=-RK
110         KADR=INTGRK(KADR1)+JCOL-KOFF
```

C2-26

```
            REALKK(KADR)=REALKK(KADR)+RK
          C
          C FOLLOWING DOES NOT INCLUDE INTERSECTING FAULTS
            J=ILNZ1+IJ
115         J=INTGRK(J)
            IF(J.GT.JCOL)GOTO 1540
            IF(ITYPE.EQ.4)RK=-RK
          C
            KADR1=1KOUN1+IJ
120         KADR=INTGRK(KADR1)+JCOL-KOFF
            REALKK(KADR)=REALKK(KADR)+RK
            IF(JTYPE.EQ.4)RK=-RK
            IF(ICOL.GT.1J)GOTO 1518
            KADR=INTGRK(KADR1)+ICOL-KOFF
125         REALKK(KADR)=REALKK(KADR)+RK
            GOTO 1530
          C--DO DOF BETWEEN COLUMNS JJ AND IJ--
           1518 CONTINUE
            KADR=IKOUN1+ICOL
130         KADR=INTGRK(KADR)+IJ-KOFF
            REALKK(KADR)=REALKK(KADR)+RK
            GOTO 1530
          C
           1520 CONTINUE
135       C ICOL NOT INTERNAL DOF, BUT COULD BE ON BOUNDARY
          C  TEST TO SEE IF ON BOUNDARY AS COORD DOF
            IF(ITYPE.EQ.4)RK=-RK
            IF(ICOL.GT.IJ)GOTO 1525
          C SPECIAL CASE WHEN ICOL ADJACENT COORD DOF
140         KADR=IKOUN1+IJ
            KADR=INTGRK(KADR)+ICOL-KOFF
            REALKK(KADR)=REALKK(KADR)+RK
            IF(ICOL.EQ.IJ)REALKK(KADR)=REALKK(KADR)+RK
            GOTO 1530
145       C--ADD TERM TO ROW ICOL
           1525 CONTINUE
            KADR=IKOUN1+ICOL
            KADR=INTGRK(KADR)+IJ-KOFF
            REALKK(KADR)=REALKK(KADR)+RK
150         GOTO 1530
          C
           1530 CONTINUE
            REALK(IQ1+IJ)=REALK(IQ1+IJ)+REALK(IQ1+JJ)
           1535 CONTINUE
155        1536 CONTINUE
            IF((ITYPE.GE.2.AND.ITYPE.LE.4).OR.DISP)REALK(IQ1+JJ)=DP
           1550 CONTINUE
            GOTO 1555
           1540 WRITE(6,25)I,II,JJ,IX,J,IJ,INTGRK(J)
160          25 FORMAT(* ---ERRROR---NOT ENOUGH SPACE FOR INTERNAL BOUND.*,/,
              i (1X,15I6))
           9001 CALL XEQCCS(*/DUMP,0./GRUMP,MUV=10,SBX./*)
            STOP
           1555 CONTINUE
165         J=NECON+ICON-1
```

C2-27

```
                II=0
                DO 1580 I=1,NICON
                IF(IFBCON(4,I).GE.10)GOTO 1580
                II=II+1
170             JJ=II+J
                IF(JJ.GT.LCON)GOTO 1540
                INTGRK(JJ)=IFBCON(2,I)
             1580 CONTINUE
                IF(DISP)GOTO 2210
175             CALL IORDER2(JJ,INTGRK(ICON),DUM,0,1,DUM)
                GOTO 1600
              C
              C----ENTRY FOR APPLYING EACH INVERSION DISPLACEMENT
                ENTRY BCONAP
180             DP=DIRV
                ENTRY2=.TRUE.
                READ(20)IX
                READ(20)(DUM(J),J=1,IX)
                GOTO 2040
185           C
              C
             1600 CONTINUE
              C---WRITE DUM FOR CONSTRAINTS WITH INVERSION OR LOOP ON FORCE VECTOR
                DO 2200 I=1,NICON
190             ITYPE=IFBCON(4,I)
                IF((ITYPE.GE.2.AND.ITYPE.LE.4).OR.ITYPE.GE.10)GOTO 2200
                MROW=IFBCON(2,I)
              C
              C
195          2040 CONTINUE
                NEXT=ICON
                IX=0
                DO 2070 INROW=1,NDT
                IF(INROW.NE.INTGRK(NEXT))GOTO 2050
200             NEXT=NEXT+1
                GOTO 2070
             2050 IF(MROW.GT.INROW)GOTO 2055
                J=ILNZ1+INROW
                IF(INTGRK(J).GT.MROW)GOTO 2070
205             IROW=INROW
                ICOL=MROW
                GOTO 2060
             2055 J=ILNZ1+MROW
                IF(INTGRK(J).GT.INROW)GOTO 2070
210             IROW=MROW
                ICOL=INROW
             2060 CONTINUE
                IX=IX+1
                IF(ENTRY2)GOTO 2065
215             IF(.NOT.SPACE.AND.(IROW.LT.RBEGIN.OR.IROW.GT.REND))
               1 CALL SHIFT(REALKK,INTGRK,IROW,IROW)
                KADR=IKOUN1+IROW
                KADR=INTGRK(KADR)+ICOL-KOFF
                DUM(IX)=REALKK(KADR)
220             GOTO 2070
```

C2-28

```
                2065 CONTINUE
                     N=IQ1+INROW
                     REALK(N)=REALK(N)-DUM(IX)*DP
                2070 CONTINUE
  225                IF(ENTRY2)GOTO 2500
                     WRITE(20)IX
                     WRITE(20)(DUM(J),J=1,IX)
                2200 CONTINUE
                     REWIND 20
  230           2210 CONTINUE
                     RETURN
                2500 CONTINUE
                     N=IQ1+NROW
                     REALK(N)=DP
  235                IF(ENTRY2)RETURN R1
                     RETURN
                C----STRESS INVERSION
                     ENTRY BCONST
                     DP=DIRV*DIST
  240                N=IQ1+NROW
                     REALK(N)=REALK(N)-DP
                     N=IQ1+NROW
                     REALK(N)=REALK(N)+DP
                     RETURN R1
  245                END
```

```
CARD NR. SEVERITY  DETAILS    DIAGNOSIS OF PROBLEM

    245    I     189       NON-INNER LOOP BEGINNING AT THIS CARD IS ENTERED FROM OUTSIDE ITS RANGE.
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

326	BCONAP	3	BCONIN	470	BCONST

VARIABLES

SN		TYPE		RELOCATION	SN		TYPE		RELOCATION
0	DIRV	REAL		F.P.	11	DISK	REAL		SHIFT
12	DISP	LOGICAL		PROB	0	DIST	REAL		F.P.
644	DP	REAL			0	DUM	REAL	ARRAY	F.P.
615	ENTRY2	LOGICAL			13	FILL	LOGICAL		SHIFT
6	GMALT	REAL		PROB	3	GRAV	REAL		PROB
5	GRAVB	REAL		PROB	4	GRAVS	REAL		PROB
0	HASH1	REAL		PROB	621	I	INTEGER		
0	IADR	INTEGER	ARRAY	BADR	632	ICOL	INTEGER		
0	ICON	INTEGER		BEGIN	644	IDP	INTEGER		
0	IFBCON	INTEGER	ARRAY	F.P.	627	II	INTEGER		
625	IJ	INTEGER			5	IK	INTEGER		BEGIN
1	IKOUNT	INTEGER		BEGIN	616	IKOUN1	INTEGER		

C2-29

VARIABLES SN TYPE RELOCATION

2	ILNZ	INTEGER		BEGIN	617	ILNZ1	INTEGER		
3	IMASTR	INTEGER		BEGIN	636	INCOL	INTEGER		
642	INROW	INTEGER			0	INTGRK	INTEGER	ARRAY	F.P.
4	IQ	INTEGER		BEGIN	620	IQ1	INTEGER		
631	IROW	INTEGER			622	ITYPE	INTEGER		
623	IX	INTEGER			630	J	INTEGER		
637	JCOL	INTEGER			624	JJ	INTEGER		
640	JTYPE	INTEGER			626	J1	INTEGER		
634	KADR	INTEGER			633	KADR1	INTEGER		
2	KBEGIN	INTEGER		SHIFT	3	KEND	INTEGER		SHIFT
1	KLEN	INTEGER		SHIFT	5	KMAX	INTEGER		SHIFT
4	KMIN	INTEGER		SHIFT	10	KOFF	INTEGER		SHIFT
0	KUNIT	INTEGER		SHIFT	0	LCON	INTEGER		END
2	LENGTH	INTEGER		PROB	5	LK	INTEGER		END
1	LKOUNT	INTEGER		END	2	LLNZ	INTEGER		END
3	LMASTR	INTEGER		END	4	LQ	INTEGER		END
0	MROW	INTEGER		F.P.	643	N	INTEGER		
0	NBON	INTEGER		BOUND	11	NDIM	INTEGER		PROB
1	NDT	INTEGER		SIZE	13	NECON	INTEGER		PROB
0	NET	INTEGER		SIZE	641	NEXT	INTEGER		
7	NFLT	INTEGER		PROB	14	NICON	INTEGER		PROB
10	NLAP	INTEGER		PROB	0	NROW	INTEGER		F.P.
1	OLDWRK	REAL		PROB	6	RBEGIN	INTEGER		SHIFT
0	REALK	REAL	ARRAY	F.P.	0	REALKK	REAL	ARRAY	KK
7	REND	INTEGER		SHIFT	635	RK	REAL		
1	RLAX	REAL		BOUND	0	R1	RETURNS		
12	SPACE	LOGICAL		SHIFT	2	TLAP	REAL		BOUND

FILE NAMES MODE

TAPE20	UNFMT	TAPE6	FMT

EXTERNALS TYPE ARGS

FTIME	REAL	2	IORDER2	6	
SHIFT		4	XEQCCS	1	

STATEMENT LABELS

563	25	FMT	0	1505	INACTIVE	0	1510	INACTIVE	
114	1515		205	1518		214	1520		
231	1525		237	1530		0	1535	INACTIVE	
0	1536	INACTIVE	261	1540		256	1550		
272	1555		313	1580		350	1600		
361	2040		373	2050		403	2055		
411	2060		431	2065		435	2070		
453	2200		457	2210		460	2500		
0	9001	INACTIVE							

LOOPS LABEL INDEX FROM-TO LENGTH PROPERTIES

20	1550	I	37 157	241B	EXT REFS	EXITS	NOT INNER	
44	1530	II	51 152	176B	EXT REFS	EXITS		
304	1500	I	167 173	11B	INSTACK	EXITS		
351	2200	I	189 228	105B	EXT REFS	ENTRIES	EXITS	NOT INNER
367	2070	INROW	198 224	54B	EXT REFS			

C2-3C

COMMON BLOCKS LENGTH

SIZE	2
BEGIN	6
END	6
PROB	13
BOUND	3
SHIFT	12
BADK	1
KK	2 LCM

STATISTICS

PROGRAM LENGTH	705B	453
SCM LABELED COMMON LENGTH	53B	43
LCM LABELED COMMON LENGTH	2B	2

(BOTTOM OF FILE)

C2-31

```
      SUBROUTINE BLD(LEN,X,NF),RETURNS(R1)
      DIMENSION X(1),XIO(2),NIO(2)
      EQUIVALENCE (NIO,XIO)
C
      COMMON/BLDIO/XIO
      COMMON/BLD  /KPTR,LREC,IREC
C
      LREC1=LREC-1
      IF(KPTR.EQ.0.OR.NF.EQ.0)RETURN R1
      IF(LEN.LT.1)RETURN R1
      IF(LEN.GT.LREC1)GOTO 5
      JPTR=KPTR+LEN
      IF(JPTR.LE.LREC)GOTO 30
    5 IE=LREC-KPTR
   10 IS=1
      NIO(KPTR)=LEN
      IF(IE.LT.1)GOTO 21
   15 DO 20 I=IS,IE
   20 XIO(KPTR+I)=X(I)
   21 IF(LEN-IE.LE.0)GOTO 25
      CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      KPTR=1-IS
      GOTO 15
   25 KPTR=IE+1+KPTR
      IF(KPTR.GT.LREC)GOTO 26
      RETURN
   26 CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      KPTR=1
      RETURN
   30 NIO(KPTR)=LEN
      DO 40 I=1,LEN
   40 XIO(KPTR+I)=X(I)
      KPTR=JPTR+1
      IF(KPTR.GT.LREC)GOTO 26
      RETURN
      ENTRY FRC
      NIO(KPTR)=-2
      CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      KPTR=0
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

C2-32

ENTRY POINTS
3 BLD 113 FRC

VARIABLES	SN	TYPE		RELOCATION					
146	I	INTEGER			144	IE	INTEGER		
2	IREC	INTEGER		BLD	145	IS	INTEGER		
143	JPTR	INTEGER			0	KPTR	INTEGER		BLD
0	LEN	INTEGER		F.P.	1	LREC	INTEGER		BLD
142	LREC1	INTEGER			0	NF	INTEGER		F.P.
0	NIO	INTEGER	ARRAY	BLDIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BLDIO

EXTERNALS	TYPE	ARGS
WRITMS		5

STATEMENT LABELS

32	5	0	10	INACTIVE	37	15	
0	20	46	21		65	25	
72	26	100	30		0	40	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
44	20	I	18 19	2B	INSTACK
105	40	I	37 3B	2B	INSTACK

COMMON BLOCKS	LENGTH
BLDIO	2
BLD	3

STATISTICS

PROGRAM LENGTH	147B	103
SCM LABELED COMMON LENGTH	5B	5

130000B SCM USED

C2-33

```
1               SUBROUTINE BLD2(LEN,X,NF),RETURNS(R1)
                DIMENSION X(1),XIO(2),NIO(2)
                LEVEL 2,X
                EQUIVALENCE (NIO,XIO)
5               COMMON/BLDIO/XIO
                COMMON/BLD/KPTR,LREC,IREC
          C
                LREC1=LREC-1
                IF(KPTR.EQ.0.OR.NF.EQ.0)RETURN R1
10              IF(LEN.LT.1)RETURN R1
                IF(LEN.GT.LREC1)GOTO 5
                JPTR=KPTR+LEN
                IF(JPTR.LE.LREC)GOTO 30
              5 IE=LREC-KPTR
15           10 IS=1
                NIO(KPTR)=LEN
                IF(IE.LT.1)GOTO 21
             15 DO 20 I=IS,IE
             20 XIO(KPTR+I)=X(I)
20           21 IF(LEN-IE.LE.0)GOTO 25
                CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                IS=IE
                IE=LEN
25              IF(LEN-IS.GT.LREC)IE=LREC+IS
                IS=IS+1
                KPTR=1-IS
                GOTO 15
             25 KPTR=IE+1+KPTR
30              IF(KPTR.GT.LREC)GOTO 26
                RETURN
             26 CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                KPTR=1
35              RETURN
             30 NIO(KPTR)=LEN
                DO 40 I=1,LEN
             40 XIO(KPTR+I)=X(I)
                KPTR=JPTR+1
40              IF(KPTR.GT.LREC)GOTO 26
                RETURN
                ENTRY FRC2
                NIO(KPTR)=-2
                CALL WRITMS(NF,XIO,LREC,IREC,0)
45              IREC=IREC+1
                KPTR=0
                RETURN
                END
```

SYMBOLIC REFERENCE MAP (R=1)

C2-34

ENTRY POINTS
3 BLD2 112 FRC2

VARIABLES	SN	TYPE		RELOCATION					
145	I	INTEGER			143	IE	INTEGER		
2	IREC	INTEGER		BLD	144	IS	INTEGER		
142	JPTR	INTEGER			0	KPTR	INTEGER		BLD
0	LEN	INTEGER		F.P.	1	LREC	INTEGER		BLD
141	LREC1	INTEGER			0	NF	INTEGER		F.P.
0	NIO	INTEGER	ARRAY	BLDIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BLDIO

EXTERNALS	TYPE	ARGS
WRITMS		5

STATEMENT LABELS

32	5	0	10	INACTIVE	37	15	
0	20	46	21		65	25	
72	26	100	30		0	40	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
44	20	I	18 19	2B	INSTACK
104	40	I	37 38	2B	INSTACK

COMMON BLOCKS	LENGTH
BLDIO	2
BLD	3

STATISTICS

PROGRAM LENGTH	146B	102
SCM LABELLED COMMON LENGTH	5B	5

130000B SCM USED

C 2 - 35

```
      1            SUBROUTINE BREAD(KEY,LEN,X,NF),RETURNS(R1)
                 C-----SEQUENTIAL (KEY=0)/UPDATING TO IO FILE HANDLER
                 C-----MODIFIED AND REINPUT 4/18/74
                 C-----OPENMS(NF,INDEX,LENGTH OF INDEX,0) MUST BE INSERTED IN MAIN
      5            DIMENSION X(1),XIO(2)
                   DIMENSION NIO(2)
                   EQUIVALENCE (XIO,NIO)
                 C
                   COMMON/BREADIO/XIO
     10            COMMON/BREAD  /NFX,LREC,KEYX(20),LENX(20),K1X(20)
                 C
                   IF(NF.GT.20)RETURN R1
                   IF(KEY.EQ.0)KEY=KEYX(NF)+LENX(NF)+1
                   K1=(KEY-1)/LREC
     15            K2=KEY-LREC*K1
                   K1=K1+1
                   KEY=LREC*(K1-1)+K2
                   IF(K1.EQ.K1X(NF).AND.NF.EQ.NFX)GOTO 130
                 120 CALL READMS(NF,XIO,LREC,K1)
     20            NFX=NF
                   K1X(NF)=K1
                 130 LENX(NF)=NIO(K2)
                   IF(LENX(NF).EQ.-1)GOTO 160
                   IF(LENX(NF).LT.0)RETURN R1
     25            IF(K2+LENX(NF).GT.LREC)GOTO 170
                   LAST=LENX(NF)
                   DO 140 I=1,LAST
                 140 X(I)=XIO(I+K2)
. 25 NOV  22.27  RWPSS IS NOW RUNNING IN THE C MACHINE**BKY
                 150 CONTINUE
     30            LEN=LENX(NF)
                   KEYX(NF)=KEY
                   RETURN
                 160 K1=K1+1
                   K2=1
     35            GOTO 120
                 170 LEN=LENX(NF)
                   IS=1
                   IE=LREC-K2
                 175 DO 180 I=IS,IE
     40          180 X(I)=XIO(I+K2)
                   IF(LEN-IE.LE.0)GOTO 150
                   K1=K1+1
                   CALL READMS(NF,XIO,LREC,K1)
                   K1X(NF)=K1
     45            IS=IE
                   IE=LEN
                   IF(LEN-IS.GT.LREC)IE=LREC+IS
                   IS=IS+1
                   K2=1-IS
     50            GOTO 175
                   ENTRY BWRITE
                 C-----CAN ONLY REWRITE RECORD JUST READ - NEEDS LEN
                   IF(KEY.EQ.0.OR.NFX.NE.NF)RETURN R1
                   K1=(KEY-1)/LREC
     55            K2=KEY-LREC*K1
```

C2-36

```
                    K1=K1+1
                    IF(K2+LENX(NF).GT.LREC)GOTO 300
                    LAST=LENX(NF)
                    DO 220 I=1,LAST
 60           220 XIO(K2+I)=X(I)
                    CALL WRITMS(NF,XIO,LREC,K1,1,0)
                230 RETURN
                300 CONTINUE
                    LEN=LENX(NF)
 65                 IS=1
                    IE=LREC-K2
                310 CONTINUE
                    CALL READMS(NF,XIO,LREC,K1)
                    DO 320 I=IS,IE
 70           320 XIO(I+K2)=X(I)
                    CALL WRITMS(NF,XIO,LREC,K1,1,0)
                    IF(LEN-IE.LE.0)RETURN
                    IS=IE
                    IE=LEN
 75                 IF(LEN-IS.GT.LREC)IE=LREC+IS
                    IS=IS+1
                    K1=K1+1
                    K2=1-IS
                    GOTO 310
 80                 END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	BREAD	127	BWRITE

VARIABLES	SN	TYPE		RELOCATION					
244	I	INTEGER			246	IE	INTEGER		
245	IS	INTEGER			0	KEY	INTEGER		F.P.
2	KEYX	INTEGER	ARRAY	BREAD	241	K1	INTEGER		
52	K1X	INTEGER	ARRAY	BREAD	242	K2	INTEGER		
243	LAST	INTEGER			0	LEN	INTEGER		F.P.
26	LENX	INTEGER	ARRAY	BREAD	1	LREC	INTEGER		BREAD
0	NF	INTEGER		F.P.	0	NFX	INTEGER		BREAD
0	NIO	INTEGER	ARRAY	BREADIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BREADIO

EXTERNALS	TYPE	ARGS			
READMS		4	WRITMS		6

STATEMENT LABELS

34	120		42	130	0	140	
62	150		67	160	72	170	
77	175		0	180	0	220	
0	230	INACTIVE	167	300	174	310	
0	320						

C2-37

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
60	140	I	27 28	2B	INSTACK
104	180	I	39 40	2B	INSTACK
161	220	I	59 60	2B	INSTACK
203	320	I	69 70	2B	INSTACK

COMMON BLOCKS	LENGTH
BREADIO	2
BREAD	62

STATISTICS

PROGRAM LENGTH	255B	173
SCM LABELED COMMON LENGTH	100B	64

130000B SCM USED

C2 - 38

```
      SUBROUTINE BREAD2(KEY,LEN,X,NF),RETURNS(R1)
C-----SEQUENTIAL (KEY=0)/UPDATING TO IO FILE HANDLER
C-----MODIFIED AND REINPUT 4/18/74
C-----OPENMS(NF,INDEX,LENGTH OF INDEX,0) MUST BE INSERTED IN MAIN
      DIMENSION X(1),XIO(2)
      LEVEL 2,X
      DIMENSION NIO(2)
      EQUIVALENCE (XIO,NIO)
C
C
      COMMON/BREADIO/XIO
C
      COMMON/BREAD/NFX,LREC,KEYX(20),LENX(20),K1X(20)
C
      IF(NF.GT.20)RETURN R1
      IF(KEY.EQ.0)KEY=KEYX(NF)+LENX(NF)+1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
      K1=K1+1
      KEY=LREC*(K1-1)+K2
      IF(K1.EQ.K1X(NF).AND.NF.EQ.NFX)GOTO 130
  120 CALL READMS(NF,XIO,LREC,K1)
      NFX=NF
      K1X(NF)=K1
  130 LENX(NF)=NIO(K2)
      IF(LENX(NF).EQ.-1)GOTO 160
      IF(LENX(NF).LT.0)RETURN R1
      IF(K2+LENX(NF).GT.LREC)GOTO 170
      LAST=LENX(NF)
      DO 140 I=1,LAST
  140 X(I)=XIO(I+K2)
  150 CONTINUE
      LEN=LENX(NF)
      KEYX(NF)=KEY
      RETURN
  160 K1=K1+1
      K2=1
      GOTO 120
  170 LEN=LENX(NF)
      IS=1
      IE=LREC-K2
  175 DO 180 I=IS,IE
  180 X(I)=XIO(I+K2)
      IF(LEN-IE.LE.0)GOTO 150
      K1=K1+1
      CALL READMS(NF,XIO,LREC,K1)
      K1X(NF)=K1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      K2=1-IS
      GOTO 175
      ENTRY BWRITE2
C-----CAN ONLY REWRITE RECORD JUST READ - NEEDS LEN
```

C2 - 39

```
                      IF(KEY.EQ.0.OR.NFX.NE.NF)RETURN R1
                      K1=(KEY-1)/LREC
                      K2=KEY-LREC*K1
                      K1=K1+1
   60                 IF(K2+LENX(NF).GT.LREC)GOTO 300
                      LAST=LENX(NF)
                      DO 220 I=1,LAST
                  220 XIO(K2+I)=X(I)
                      CALL WRITMS(NF,XIO,LREC,K1,1,0)
   65                 RETURN
                  300 CONTINUE
                      LEN=LENX(NF)
                      IS=1
                      IE=LREC-K2
   70                 CONTINUE
                      CALL READMS(NF,XIO,LREC,K1)
                      DO 320 I=IS,IE
                  320 XIO(I+K2)=X(I)
                      CALL WRITMS(NF,XIO,LREC,K1,1,0)
   75                 IF(LEN-IE.LE.0)RETURN
                      IS=IE
                      IE=LEN
                      IF(LEN-IS.GT.LREC)IE=LREC+IS
                      IS=IS+1
   80                 K1=K1+1
                      K2=1-IS
                      GOTO 310
                      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	BREAD2	127	BWRITE2

VARIABLES

SN	TYPE		RELOCATION		SN	TYPE		RELOCATION
244	I	INTEGER			246	IE	INTEGER	
245	IS	INTEGER			0	KEY	INTEGER	F.P.
2	KEYX	INTEGER	ARRAY	BREAD	241	K1	INTEGER	
52	K1X	INTEGER	ARRAY	BREAD	242	K2	INTEGER	
243	LAST	INTEGER			0	LEN	INTEGER	F.P.
26	LENX	INTEGER	ARRAY	BREAD	1	LREC	INTEGER	BREAD
0	NF	INTEGER		F.P.	0	NFX	INTEGER	BREAD
0	NIO	INTEGER	ARRAY	BREADIO	0	R1	RETURNS	
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY BREADIO

EXTERNALS

	TYPE	ARGS			
READMS		4	WRITMS		6

STATEMENT LABELS

34	120	42	130	0	140		
62	150	67	160	72	170		

C 2 -40

STATEMENT LABELS

77	175			0	180	0	220
0	230	INACTIVE		166	300	173	310
0	320						

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
60	140	I	30 31	2B	INSTACK
104	180	I	42 43	2B	INSTACK
160	220	I	62 63	2B	INSTACK
203	320	I	72 73	2B	INSTACK

COMMON BLOCKS	LENGTH
BREADIO	2
BREAD	62

STATISTICS

PROGRAM LENGTH	254B	172
SCM LABELED COMMON LENGTH	100B	64

130000B SCM USED

C2-41

```
      SUBROUTINE CMAT(KC,CC,C,TLAP,L)
C-11/29/77 VERSION----3D ISOTROPIC CONSTANTS
C 6X6 MATRIX C
C     CC - MAXWELLIAN CONSTANTS
C        CC(1) - RELAXATION TIME
C        CC(2) - SHEAR MODULUS IN UNITS OF E12
C        CC(3) - BULK MODULUS IN UNITS OF E12
C     TLAP(L) - LAPLACE REDUCED TIME
C
      DIMENSION C(6,6),CC(3),TLAP(3)
      REAL KS,C,CC,TLAP,SHS
C
C  MAXWELLIAN FLUID IN LAPLACE SPACE - S$SHEAR MODULUS
      SHS=(CC(1)*CC(2)*TLAP(L))/(CC(1)*TLAP(L)+1)
      KS=CC(3)
C  COMPUTE LAMDA
      C12=KS-2.*SHS/3.
      C1=KS+4.*SHS/3.
      C(2,2)=C1
      C(3,3)=C1
      C(1,1)=C1
      C(1,2)=C12
      C(2,1)=C12
      C(1,3)=C12
      C(3,1)=C12
      C(2,3)=C12
      C(3,2)=C12
      C(4,4)=SHS
      C(5,5)=SHS
      C(6,6)=SHS
      C(1,4)=0.
      C(1,5)=0.
      C(1,6)=0.
      C(2,4)=0.
      C(2,5)=0.
      C(2,6)=0.
      C(3,4)=0.
      C(3,5)=0.
      C(3,6)=0.
      C(4,1)=0.
      C(4,2)=0.
      C(4,3)=0.
      C(5,1)=0.
      C(5,2)=0.
      C(5,3)=0.
      C(6,1)=0.
      C(6,2)=0.
      C(6,3)=0.
      C(4,5)=0.
      C(4,6)=0.
      C(5,4)=0.
      C(5,6)=0.
      C(6,4)=0.
      C(6,5)=0.
      RETURN
```

C2-42

```
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 CMAT

VARIABLES	SN	TYPE		RELOCATION					
0	C	REAL	ARRAY	F.P.	0	CC	REAL	ARRAY	F.P.
51	C1	REAL			50	C12	REAL		
0	KC	INTEGER	*UNUSED	F.P.	46	K5	REAL		
0	L	INTEGER		F.P.	47	SH5	REAL		
0	TLAP	REAL	ARRAY	F.P.					

STATISTICS
PROGRAM LENGTH 528 42
130000B SCM USED

CMAT C2-43

```
      SUBROUTINE CMAT(KC,CC,C,TLAP,L)
C
C  C  - 3X3 PLANE STRAIN CONSTANTS
C  CC - MAXWELLIAN CONSTANTS
C     CC(1) - RELAXATION TIME
C     CC(2) - SHEAR MODULUS IN UNITS OF E12
C     CC(3) - BULK MODULUS IN UNITS OF E12
C  TLAP(L)  - LAPLACE REDUCED TIME
C
      DIMENSION C(3,3),CC(3),TLAP(3)
C
C  MAXWELLIAN FLUID IN LAPLACE PLANE - S*SHEAR MODULUS
C
      SHS=(CC(1)*CC(2)*TLAP(L))/(CC(1)*TLAP(L)+1.)
      KS=CC(3)
C
      C(1,1)=KS+4.*SHS/3.
      C(2,1)=KS-2.*SHS/3.
      C(1,2)=C(2,1)
      C(2,2)=C(1,1)
      C(3,1)=0.
      C(3,2)=0.
      C(1,3)=0.
      C(2,3)=0.
      C(3,3)=SHS
      RETURN
      END
```

REAL KS

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 CMAT

VARIABLES	SN	TYPE		RELOCATION					
0	C	REAL	ARRAY	F.P.	0	CC	REAL	ARRAY	F.P.
0	KC	INTEGER	*UNUSED	F.P.	32	KS	INTEGER		
0	L	INTEGER		F.P.	31	SHS	REAL		
0	TLAP	REAL	ARRAY	F.P.					

STATISTICS
PROGRAM LENGTH 33B 27
120000B SCM USED

C2-44
CMATLAW

```
      1            SUBROUTINE CMAT(KC,CC,C,TLAP,L)
                   C
                   C  C  - 3X3 PLANE STRAIN CONSTANTS
                   C  CC - MAXWELLIAN CONSTANTS
      5            C     CC(1) - RELAXATION TIME
                   C     CC(2) - SHEAR MODULUS IN UNITS OF E12
                   C     CC(3) - LAMBDA -- MODIFIED 9/17/79
                   C  TLAP(L)  - LAPLACE REDUCED TIME
                   C
     10            DIMENSION C(3,3),CC(3),TLAP(3)
                   REAL KS,C,CC,SHS
                   C
                   C  MAXWELLIAN FLUID IN LAPLACE PLANE - S*SHEAR MODULUS
                   C
     15            SHS=(CC(1)*CC(2)*TLAP(L))/(CC(1)*TLAP(L)+1.)
                   KS=CC(3)+2.*SHS/3.
                   C
                   C(1,1)=KS+4.*SHS/3.
                   C(2,1)=KS-2.*SHS/3.
     20            C(1,2)=C(2,1)
                   C(2,2)=C(1,1)
                   C(3,1)=0.
                   C(3,2)=0.
                   C(1,3)=0.
     25            C(2,3)=0.
                   C(3,3)=SHS
                   RETURN
                   END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 CMAT

VARIABLES	SN	TYPE		RELOCATION					
0 C		REAL	ARRAY	F.P.	0	CC	REAL	ARRAY	F.P.
0 KC		INTEGER	*UNUSED	F.P.	31	KS	REAL		
0 L		INTEGER		F.P.	32	SHS	REAL		
0 TLAP		REAL	ARRAY	F.P.					

STATISTICS
PROGRAM LENGTH 33B 27
120000B SCM USED
(BOTTOM OF FILE)

C2-45

```
1                FUNCTION DOT(A,B,K)
                 C******
                 C     DOT PRODUCT IN FORTRAN 9/18/78
                 C*******
5                     DIMENSION A(2),B(2)
                      LEVEL 2,A,B
                 C
                      PROD=0.
                 C
10                    DO 10 I=1,K
                   10 PROD=PROD+A(I)*B(I)
                      DOT=PROD
                      RETURN
                      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
4 DOT

VARIABLES	SN	TYPE		RELOCATION					
0	A	REAL	ARRAY	F.P.	0	B	REAL	ARRAY	F.P.
15	DOT	REAL			17	I	INTEGER		
0	K	INTEGER		F.P.	16	PROD	REAL		

STATEMENT LABELS
0 10

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
11	10	I	10 11	3B	INSTACK

STATISTICS
PROGRAM LENGTH 20B 16
130000B SCM USED

C2-46

```
                                IDENT  DOT
                          *FUNCTION DOT(A,B,N) FOR 7600, A,B IN LCM
                                ENTRY  DOT
 0                        1     DOT    BSSZ   1
 1  5021000001                        SA2    A1+1        ADDRESS OF A IN X2
          5032000001              SA3    A2+1        ADDRESS OF N IN X3
 2  53330                          SA3    X3          FETCH N INTO X3
       63230                     SB2    X3          TRANSFER N FROM X3 TO B2
            43400                MX4    0           ZERO A
                 43500           MX5    0           ZERO B
 3  6140000000                     SB4    0           ZERO INDEX
          6150000001              SB5    1           USE FOR 1
 4  43600                          MX6    0           ZERO ACCUMULATION REGISTER
 5  40745                     LOOP FX7    X4*X5       A*B FOR OPT
       01441                     RX4    X1          FETCH A AT ADDRESS X1
            66445                SB4    B4+B5       INDEX COUNTER
                 01452           RX5    X2          FETCH B AT ADDRESS X2
 6  73115                          SX1    X1+B5        NEW ADDRESS FOR A
       73225                     SX2    X2+B5        INCREMENT ADDRESS FOR B
            30667                FX6    X6+X7       SUM NOW
 7  0742000005 +                   LT     B4,B2,LOOP  LOOP IF INDEX.LT.N
            40745                FX7    X4*X5       A*B FOR LAST ELEMENTS
                 30667           FX6    X6+X7       SUM LAST PRODUCT
10  24606                          NX6    X6          NORMALIZE
       0400000000 +               EQ     DOT         RETURN
11                                END

                 44100    STORAGE USED              26 STATEMENTS       2 SYMBOLS
                             7600 ASSEMBLY       0.016 SECONDS          5 REFERENCES
```

SYMBOLIC REFERENCE TABLE.

C2-47

DOT	0	PROGRAM#	2/03 E	2/04 L	2/25
LOOP	5	PROGRAM#	2/14 L	2/21	

C2-48

```
      SUBROUTINE GRS(LNUM,NGRAVS,COORD,THK,SM,YGRV,KW)
C
      DIMENSION COORD(3,1),SM(1),XD(4)
      LOGICAL OUT
      COMMON/GRV/NGRAV,RHOF,DRHO(4),XF(8),NODG(4)
      COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,NLAP,
     1 NDIM,DISP
      COMMON/IO/KR,KW1,KP,KT1,KT2,KT3,OUT
C
C  ASSUME VERTICAL GRAV FORCE YGRV
C  (SHOULD BE NEGATIVE IF OUTWARD IS POSITIVE)
C     NGRAVS - NUMBER OF SURFACE NODES
C     NODG   - LOCAL NODE NUMBERS AT INTERFACE
C     DRHO   - LOCAL DENSITY DISCONTINUITY FOR NODE NODG(I)
C                   (NOTE SIGN)
C     YGRV   - VERTICAL GRAVITY
C
C  NOTE?  DO NOT INCLUDE ELEMENTS WHICH ONLY *PEAK* AT SURFACE
C
C**********************************************************************
C
      ID=0
      IF(NDIM.EQ.3)ID=1
      XD1=ABS(COORD(1,NODG(1))-COORD(1,NODG(2)))*.5*THK
      XD(1)=XD1
      XD(2)=XD1
      IF(OUT)WRITE(KW,10)LNUM,XD1,YGRV
   10 FORMAT(1H0,20X,*ENTRY GRS - ELEMENT NUMBER*,I6,/,5X,
     1 *GRAVITATIONAL BODY FORCE -*,1PE11.3,2X,*DISCONTINUITIES - GRAVIT
     2Y=*,1PE11.3,/,1X,*LOCAL NODE NUMBER*,5X,*LOCAL DENSITY JUMP*,
     3 5X,*SMIJ*,6X,*SMGR*,6X,*SM(IJ)*)
      IF(NGRAVS.LE.2)GOTO 900
      XD1=ABS(COORD(1,NODG(3))-COORD(1,NODG(4)))*.5*THK
      XD(3)=XD1
      XD(4)=XD1
  900 CONTINUE
      DO 1000 I=1,NGRAVS
      J=NODG(I)*NDIM-ID
      IJ=(J+1)*J/2
      SMIJ=SM(IJ)
      SMGR=-YGRV*DRHO(I)*XD(I)
      SM(IJ)=SMGR+SMIJ
      IF(OUT)WRITE(KW,20)NODG(I),DRHO(I),SMIJ,SMGR,SM(IJ)
   20 FORMAT(5X,I11,10X,1PE11.3,5X,1P3E11.3)
 1000 CONTINUE
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

C 2-49

ENTRY POINTS
3 GRS

VARIABLES	SN	TYPE		RELOCATION					
0	COORD	REAL	ARRAY	F.P.	12	DISP	REAL		PROB
2	DRHO	REAL	ARRAY	GRV	6	GMALT	REAL		PROB
3	GRAV	REAL		PROB	5	GRAVB	REAL		PROB
4	GRAVS	REAL		PROB	0	HASH1	REAL		PROB
146	I	INTEGER			144	ID	INTEGER		
150	IJ	INTEGER			147	J	INTEGER		
2	KP	INTEGER		IO	0	KR	INTEGER		IO
3	KT1	INTEGER		IO	4	KT2	INTEGER		IO
5	KTC	INTEGER		IO	0	KW	INTEGER		F.P.
1	KW1	INTEGER		IO	2	LENGTH	INTEGER		PROB
0	LNUM	INTEGER		F.P.	11	NDIM	INTEGER		PROB
7	NFLT	INTEGER		PROB	0	NGRAV	INTEGER		GRV
0	NGRAVS	INTEGER		F.P.	10	NLAP	INTEGER		PROB
16	NODG	INTEGER	ARRAY	GRV	1	OLDWRK	REAL		PROB
6	OUT	LOGICAL		IO	1	RHOF	REAL		GRV
0	SM	REAL	ARRAY	F.P.	152	SMGR	REAL		
151	SMIJ	REAL			0	THK	REAL		F.P.
153	XD	REAL	ARRAY		145	XD1	REAL		
6	XF	REAL	ARRAY	GRV	0	YGRV	REAL		F.P.

INLINE FUNCTIONS	TYPE	ARGS	
ABS	REAL	1	INTRIN

STATEMENT LABELS

101	10	FMT		137	20	FMT		41	900
0	1000								

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
45	1000	I	37 45	24B	EXT REFS

COMMON BLOCKS	LENGTH
GRV	18
PROB	11
IO	7

STATISTICS

PROGRAM LENGTH	164B	116
SCM LABELED COMMON LENGTH	44B	36

130000B SCM USED

C 2-45

```
      SUBROUTINE HEX8(X,C,MROT,NOD,THETZ,THETY,THETX,XK,B,REFPT)
C*********************************************************************
C HEXAHEDRON
C*********************************************************************
C ARGUMENT LIST LEGEND
C   1. X      GLOBAL COORDINATES OF HEXAHEDRON
C   2. C      6X6 MATRIX OF ELASTIC CONSTANTS FOR ISOTROPIC MATERIAL
C   3. MROT   NUMBER OF ROTATED NODES
C   4. NOD    LOCAL NODE NUMBERS OF ROTATED NODES
C   5. THETZ  EULER ANGLES FOR NODAL ROTATION ABOUT Z-AXIS
C   6. THETY  EULER ANGLES FOR NODAL ROTATION ABOUT Y-AXIS
C   7. THETX  EULER ANGLES FOR NODAL ROTATION ABOUT X-AXIS
C   8. XK     MEMBER STIFFNESS MATRIX CALCULATED BY THIS SUBROUTINE
C   9. B      MEMBER STRESS DISPLACEMENT MATRIX FOR REFERENCE POINT
C             CALCULATED BY THIS SUBROUTINE
C  10. REFPT  COORDINATES OF REFERENCE POINT CALCULATED BY THIS
C             SUBROUTINE
C*********************************************************************
C
C.....APRIL 3, 1973 VERSION.....
C
      DIMENSION X(24), C(6,6), NOD(8), THETZ(8), THETY(8), THETX(8), XK(
     1300), B(6,25), REFPT(3)
C INTERNAL - TEMPORARY STORAGE
      DIMENSION IQX(27),IQY(27),IQZ(27),XX(3,27),W(27),YJ(3,3),Q27(7)
      DIMENSION IX(2,3),IF1(4,3),IF2(4,3),F(3,8),XJ(3,3)
      DIMENSION T(3,3,8),EK(24,24)
      DIMENSION EE(9),XE(9),IE1(3,3),IE2(2,3),YF(8),XF(8)
      DIMENSION ZF(8),ID(24),BB(6,24)
      DATA IQX/1,2,1,1,3,1,1,4,5,4,5,5,5,4,4,4*1,6,6,7,7,6,6,7,7/
      DATA IQY/1,1,2,1,1,3,1,4,5,5,4,5,4,5,4,6,6,7,7,4*1,6,7,6,7/
      DATA IQZ/3*1,2,1,1,3,4,3*5,3*4,5,6,7,6,7,6,7,6,7,4*1/
      DATA Q27/0.0,0.848418011,-0.848418011,0.652816472,-0.652816472,1.1
     *06412899,-1.106412899/
      DATA IX/2,3,1,3,2,1/
      DATA IF1/1,3,5,7,1,3,5,7,1,8,2,7/
      DATA IF2/2,4,6,8,4,2,8,6,5,4,6,3/
      DATA IE1/1,9,8,9,3,7,8,7,6/
      DATA IE2/2,9,4,8,5,7/
      DATA ID/1,4,7,10,13,16,19,22,2,5,8,11,14,17,20,23,3,6,9,12,15,18,2
     *1,24/
      DATA W/0.788073483,6*0.499369002,8*0.478508449,12*0.032303742/
      DO 1 I=1,27
      XX(1,I)=Q27(IQX(I))
      XX(2,I)=Q27(IQY(I))
    1 XX(3,I)=Q27(IQZ(I))
      K=0
      DO 20 I=1,3
      EE(I+6)=C(3+I,3+I)
      DO 20 J=1,I
      K=K+1
   20 EE(K)=C(I,J)
      DO 11 I=1,300
   11 XK(I)=0.0
      DO 15 K=1,27
```

C 2-51

```
      DO 5 I=1,3
      XA=XX(IX(1,I),K)
      XB=XX(IX(2,I),K)
      F(1,IF1(1,I))=-0.125*(1.-XA)*(1.-XB)
60    F(I,IF1(2,I))=0.125*(1.+XA)*(1.-XB)
      F(I,IF1(3,I))=-0.125*(1.-XA)*(1.+XB)
      F(I,IF1(4,I))=0.125*(1.+XA)*(1.+XB)
      DO 5 J=1,4
    5 F(I,IF2(J,I))=-F(I,IF1(J,I))
65    DO 6 I=1,3
      DO 6 J=1,3
      XJ(I,J)=0.0
      DO 6 L=1,8
    6 XJ(I,J)=XJ(I,J)+F(I,L)*X(J+(L-1)*3)
70    YJ(1,1)=XJ(2,2)*XJ(3,3)-XJ(2,3)*XJ(3,2)
      YJ(1,2)=XJ(1,3)*XJ(3,2)-XJ(1,2)*XJ(3,3)
      YJ(1,3)=XJ(1,2)*XJ(2,3)-XJ(1,3)*XJ(2,2)
      YJ(2,1)=XJ(2,3)*XJ(3,1)-XJ(2,1)*XJ(3,3)
      YJ(2,2)=XJ(1,1)*XJ(3,3)-XJ(1,3)*XJ(3,1)
75    YJ(2,3)=XJ(1,3)*XJ(2,1)-XJ(1,1)*XJ(2,3)
      YJ(3,1)=XJ(2,1)*XJ(3,2)-XJ(2,2)*XJ(3,1)
      YJ(3,2)=XJ(1,2)*XJ(3,1)-XJ(1,1)*XJ(3,2)
      YJ(3,3)=XJ(1,1)*XJ(2,2)-XJ(1,2)*XJ(2,1)
      DET=XJ(1,1)*YJ(1,1)+XJ(2,1)*YJ(1,2)+XJ(3,1)*YJ(1,3)
80    DO 7 I=1,3
      DO 7 J=1,3
    7 YJ(I,J)=YJ(I,J)/DET
      DO 8 I=1,8
      DO 9 J=1,3
85    XF(J)=0.0
      DO 9 L=1,3
    9 XF(J)=XF(J)+YJ(J,L)*F(L,I)
      DO 8 L=1,3
    8 F(L,I)=XF(L)
90    IF(K .GT. 1) GO TO 114
      DO 111 I=1,3
      DO 111 J=1,3
      JJ=(J-1)*8
      DO 111 L=1,8
95  111 B(I,JJ+L)=C(I,J)*F(J,L)
      DO 112 I=1,3
      II=I+3
      DO 112 J=1,2
      JJ=(IX(J,I)-1)*8
100   IJ=IX(3-J,I)
      DO 112 L=1,8
  112 B(II,JJ+L)=C(II,II)*F(IJ,L)
      DO 113 I=1,3
      II=I+3
105   IJ=(I-1)*8
      DO 113 J=1,8
  113 B(II,IJ+J)=0.
  114 CONTINUE
      XB=W(K)*DET
110   DO 10 I=1,9
```

C2-52

```
      10 XE(I)=EE(I)*XB
         DO 13 I=1,3
         II=(I-1)*8
         XA=XE(IE1(1,1))
115      XB=XE(IE1(I,2))
         XC=XE(IE1(I,3))
         DO 12 L=1,8
         XF(L)=XA*F(1,L)
         YF(L)=XB*F(2,L)
120      12 ZF(L)=XC*F(3,L)
         DO 13 L=1,8
         IL=II+L
         DO 13 LL=1,L
         IST=IL*(IL-1)/2+II+LL
125      13 XK(IST)=XK(IST)+XF(L)*F(1,LL)+YF(L)*F(2,LL)+ZF(L)*F(3,LL)
         IK=0
         DO 15 I=1,2
         II=I*8
         DO 15 J=1,I
130      IK=IK+1
         JJ=(J-1)*8
         IJ=IE2(1,IK)
         JI=IE2(2,IK)
         DO 14 L=1,8
135      XF(L)=XE(IJ)*F(I+1,L)
      14 YF(L)=XE(JI)*F(J,L)
         DO 15 L=1,8
         DO 15 IJ=1,8
         IST=(II+L)*(II+L-1)/2+JJ+IJ
140      15 XK(IST)=XK(IST)+XF(L)*F(J,IJ)+YF(L)*F(I+1,IJ)
         L=0
         DO 17 I=1,24
         IS=ID(I)
         DO 35 K=1,6
145      35 BB(K,IS)=B(K,I)
         DO 17 J=1,I
         L=L+1
         DET=XK(L)
         IT=ID(J)
150      EK(IS,IT)=DET
      17 EK(IT,IS)=DET
         IF(MROT .EQ. 0) GO TO 32
         DO 26 K=1,8
         DO 26 I=1,3
155      DO 27 J=1,3
      27 T(I,J,K)=0.0
      26 T(I,I,K)=1.0
         DO 28 K=1,MROT
         KK=NOD(K)
160      ZX=THETX(K)
         ZY=THETY(K)
         ZZ=THETZ(K)
         CX=COS(ZX)
         CY=COS(ZY)
165      CZ=COS(ZZ)
```

C2-53

```
      SX=SIN(ZX)
      SY=SIN(ZY)
      SZ=SIN(ZZ)
      T(1,1,KK)=CY*CZ
170   T(1,2,KK)=SY*SX*CZ-SZ*CX
      T(1,3,KK)=SY*CX*CZ+SX*SZ
      T(2,1,KK)=CY*SZ
      T(2,2,KK)=SX*SY*SZ+CX*CZ
      T(2,3,KK)=SY*CX*SZ-SX*CZ
175   T(3,1,KK)=-SY
      T(3,2,KK)=CY*SX
   28 T(3,3,KK)=CY*CX
      DO 18 I=1,8
      II=(I-1)*3
180   DO 23 J=1,I
      JJ=(J-1)*3
      DO 19 K=1,3
      IK=II+K
      DO 19 L=1,3
185   JL=JJ+L
   19 XJ(K,L)=EK(IK,JL)
      DO 22 K=1,3
      DO 22 L=1,3
      YJ(K,L)=0.
190   DO 22 IT=1,3
   22 YJ(K,L)=YJ(K,L)+XJ(K,IT)*T(IT,L,J)
      DO 23 K=1,3
      IK=II+K
      LL=3
195   IF(I .EQ. J) LL=K
      DO 23 L=1,LL
      IS=IK*(IK-1)/2+JJ+L
      XK(IS)=0.
      DO 23 IT=1,3
200   23 XK(IS)=XK(IS)+T(IT,K,1)*YJ(IT,L)
      DO 18 J=1,6
      DO 24 K=1,3
      XF(K)=0.0
      DO 24 L=1,3
205   IL=II+L
   24 XF(K)=XF(K)+BB(J,IL)*T(L,K,I)
      DO 18 K=1,3
      IL=II+K
   18 B(J,IL)=XF(K)
210   GO TO 36
   32 L=0
      DO 33 I=1,24
      DO 34 J=1,6
   34 B(J,I)=BB(J,I)
215   DO 33 K=1,I
      L=L+1
   33 XK(L)=EK(I,K)
   36 CONTINUE
      DO 30 I=1,3
220   REFPT(I)=0.0
```

C2-57

```
                      DO 31 J=1,8
                      IS=(I-1)*8+J
                      IJ=ID(IS)
                   31 REFPT(I)=REFPT(I)+X(IJ)
      225          30 REFPT(I)=REFPT(I)*0.125
                      DO 40 J = 1,6
                   40 B(J,25) = 0.
                      RETURN
                      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	HEX8

VARIABLES	SN	TYPE		RELOCATION					
0	B	REAL	ARRAY	F.P.	2770	BB	REAL	ARRAY	
0	C	REAL	ARRAY	F.P.	1014	CX	REAL		
1015	CY	REAL			1016	CZ	REAL		
774	DET	REAL			2647	EE	REAL	ARRAY	
1547	EK	REAL	ARRAY		1376	F	REAL	ARRAY	
766	I	INTEGER			2740	ID	INTEGER	ARRAY	
2671	IE1	INTEGER	ARRAY		2702	IE2	INTEGER	ARRAY	
1346	IF1	INTEGER	ARRAY		1362	IF2	INTEGER	ARRAY	
776	II	INTEGER			777	IJ	INTEGER		
1004	IK	INTEGER			1001	IL	INTEGER		
1023	IQX	INTEGER	ARRAY		1056	IQY	INTEGER	ARRAY	
1111	IQZ	INTEGER	ARRAY		1006	IS	INTEGER		
1003	IST	INTEGER			1007	IT	INTEGER		
1340	IX	INTEGER	ARRAY		770	J	INTEGER		
1005	JI	INTEGER			775	JJ	INTEGER		
1022	JL	INTEGER			767	K	INTEGER		
1010	KK	INTEGER			773	L	INTEGER		
1002	LL	INTEGER			0	MROT	INTEGER		F.P.
0	NOD	INTEGER	ARRAY	F.P.	1331	Q27	REAL	ARRAY	
0	REFPT	REAL	ARRAY	F.P.	1017	SX	REAL		
1020	SY	REAL			1021	SZ	REAL		
1437	T	REAL	ARRAY		0	THETX	REAL	ARRAY	F.P.
0	THETY	REAL	ARRAY	F.P.	0	THETZ	REAL	ARRAY	F.P.
1265	W	REAL	ARRAY		0	X	REAL	ARRAY	F.P.
771	XA	REAL			772	XB	REAL		
1000	XC	REAL			2660	XE	REAL	ARRAY	
2720	XF	REAL	ARRAY		1426	XJ	REAL	ARRAY	
0	XK	REAL	ARRAY	F.P.	1144	XX	REAL	ARRAY	
2710	YF	REAL	ARRAY		1320	YJ	REAL	ARRAY	
2730	ZF	REAL	ARRAY		1011	ZX	REAL		
1012	ZY	REAL			1013	ZZ	REAL		

EXTERNALS	TYPE	ARGS			
COS	REAL	1 LIBRARY	SIN	REAL	1 LIBRARY

C 2-55

STATEMENT LABELS

0	1	0	5	0	6
0	7	0	8	0	9
0	10	0	11	0	12
0	13	0	14	0	15
0	17	0	18	0	19
0	20	0	22	0	23
0	24	0	26	0	27
0	28	0	30	0	31
706	32	0	33	0	34
0	35	732	36	0	40
0	111	0	112	0	113
261	114				

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	
11	1	I	43 46	5B	INSTACK	
22	20	I	48 52	12B		NOT INNER
26	20	J	50 52	3B	INSTACK	
35	11	I	53 54	2B	INSTACK	
40	15	K	55 140	346B		NOT INNER
47	5	I	56 64	32B		NOT INNER
72	5	J	63 64	4B	INSTACK	
102	6	I	65 69	12B		NOT INNER
103	6	J	66 69	7B		NOT INNER
106	6	L	68 69	3B	INSTACK	
147	7	I	80 82	4B		NOT INNER
150	7	J	81 82	2B	INSTACK	
155	8	I	83 89	21B		NOT INNER
160	9	J	84 87	6B		NOT INNER
162	9	L	86 87	3B	INSTACK	
171	8	L	88 89	2B	INSTACK	
201	111	I	91 95	17B		NOT INNER
205	111	J	92 95	11B		NOT INNER
211	111	L	94 95	2B	INSTACK	
224	112	I	96 102	25B		NOT INNER
231	112	J	98 102	13B		NOT INNER
241	112	L	101 102	2B	INSTACK	
253	113	I	103 107	6B		NOT INNER
256	113	J	106 107	2B	INSTACK	
265	10	I	110 111	2B	INSTACK	
270	13	I	112 125	43B		NOT INNER
301	12	L	117 120	5B	INSTACK	
311	13	L	121 125	17B		NOT INNER
321	13	LL	123 125	6B	INSTACK	
335	15	I	127 140	47B		NOT INNER
340	15	J	129 140	41B		NOT INNER
352	14	L	134 136	3B	INSTACK	
361	15	L	137 140	16B		NOT INNER
371	15	IJ	138 140	5B	INSTACK	
414	17	I	142 151	30B		NOT INNER
424	35	K	144 145	2B	INSTACK	
433	17	J	146 151	4B	INSTACK	
452	26	K	153 157	17B		NOT INNER
456	26	I	154 157	5B		NOT INNER
457	27	J	155 156	2B	INSTACK	

C2-56

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
472	20	K	158 177	45B	EXT REFS
544	18	I	178 209	141B	NOT INNER
552	23	J	180 200	71B	NOT INNER
556	19	K	182 186	5B	NOT INNER
557	19	L	184 186	3B	INSTACK
567	22	K	187 191	15B	NOT INNER
572	22	L	188 191	7B	NOT INNER
575	22	IT	190 191	3B	INSTACK
611	23	K	192 200	25B	NOT INNER
622	23	L	196 200	10B	NOT INNER
626	23	IT	199 200	3B	INSTACK
650	18	J	201 209	26B	NOT INNER
655	24	K	202 206	7B	NOT INNER
660	24	L	204 206	3B	INSTACK
670	18	K	207 209	2B	INSTACK
713	33	I	212 217	17B	NOT INNER
717	34	J	213 214	2B	INSTACK
725	33	K	215 217	2B	INSTACK
736	30	I	219 225	10B	NOT INNER
740	31	J	221 224	3B	INSTACK
747	40	J	226 227	2B	INSTACK

STATISTICS

PROGRAM LENGTH 3221B 1681

130000B SCM USED

C2-57

```
      SUBROUTINE IORDER2(NUM,IX,Y,NY,LYDIM,AUX)
C ORDERS IN ASCENDING VALUE - X(LYDIM) AND Y(LYDIM,NY) ACCORDING TO
C     IX(NUM)
      DIMENSION IX(1),Y(LYDIM,1),AUX(1)
      LOGICAL AGAIN
      LEVEL 2,IX
C
C
    1 LAST=NUM-1
  100 AGAIN=.FALSE.
      DO 20 I=1,LAST
      IF(IX(I+1).GE.IX(I))GOTO 20
      IX1=IX(I)
      IX(I)=IX(I+1)
      IX(I+1)=IX1
    5 IF(NY.EQ.0)GOTO 19
      DO 6 J=1,NY
    6 AUX(J)=Y(I,J)
      DO 10 J=1,NY
   10 Y(I,J)=Y(I+1,J)
      DO 15 J=1,NY
   15 Y(I+1,J)=AUX(J)
   19 AGAIN=.TRUE.
   20 CONTINUE
      IF(AGAIN)GOTO 100
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 IORDER2

VARIABLES	SN	TYPE		RELOCATION					
54	AGAIN	LOGICAL			0	AUX	REAL	ARRAY	F.P.
56	I	INTEGER			0	IX	INTEGER	ARRAY	F.P.
57	IX1	INTEGER			60	J	INTEGER		
55	LAST	INTEGER			0	LYDIM	INTEGER		F.P.
0	NUM	INTEGER		F.P.	0	NY	INTEGER		F.P.
0	Y	REAL	ARRAY	F.P.					

STATEMENT LABELS							
0	1	INACTIVE	0	5	INACTIVE	0	6
0	10		0	15		46	19
47	20		14	100			

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
16	20	I	11 24	32B	NOT INNER
27	6	J	17 18	2B	INSTACK
35	10	J	19 20	2B	INSTACK
44	15	J	21 22	2B	INSTACK

C2-58

STATISTICS
  PROGRAM LENGTH 65B 53
      130000B SCM USED

```
      SUBROUTINE QUAD4(COORD,THK,TEMP,C,NROT,NODE,ANGLE,F,SM,B,LNUM,KW,
     1 OUT,GRAVB,RHO,YGRV)
C*********************************************************************
C ELEMENT GENERATOR LIBRARY
C FOUR-NODE PLANE STRESS/PLANE STRAIN QUADRILATERAL
C
C
C*********************************************************************
C ARGUMENT LIST LEGEND
C   1.   COORD - CORNER COORDINATES IN CW OR CCW ORDER? X1,Y1,Z1,X2,...
C                ...,Z4' Z COORDINATES ARE NOT USED BY THIS SUBR
C   2.   THK -   ELEMENT THICKNESS IN Z DIRECTION
C   3.   TEMP -  CHANGE FROM REFERENCE TEMPERATURE AT ELEMENT CORNERS
C   4.   C -     3X3 ELASTIC CONSTANTS MATRIX (PLANE STRESS OR PLANE
C                STRAIN' ISOTROPIC OR ANISOTROPIC)
C   5.   NROT -  NUMBER OF NODES (CORNERS) AT WHICH ROTATION TO SKEWED
C                COORDINATE SYSTEM FOR MIXED BOUNDARY CONDITION IS TO
C                BE PERFORMED
C   6.   NODE -  VECTOR CONTAINING LOCAL NODE NUMBERS OF NODES AT WHICH
C                ROTATIONS ARE TO BE PERFORMED
C   7.   ANGLE - ROTATION ANGLES (IN RADIAN MEASURE) CORRESPONDING TO
C                NODE VECTOR
C   8.   F -     ELEMENT CONSISTENT NODAL FORCES ARE PLACED HERE
C  10.   SM -    ELEMENT STIFFNESSES IN LTV FORM ARE PLACED HERE
C  11.   B -     ELEMENT STRESS-DISPLACEMENT TRANSFER MATRIX IS PLACED
C                HERE
C  12.   LNUM    USERS ELEMENT NUMBER
C  13.   KW      FORTRAN UNIT NUMBER FOR ON-LINE PRINTER AT USERS
C                COMPUTING FACILITY
C*********************************************************************
C.....APRIL 3, 1973 VERSION.....................................
      DIMENSION COORD(12), TEMP(4), C(3,3), NODE(4), ANGLE(4), F(8),
     2 SM(36), B(3,9)
      LOGICAL OUT,GRAVB
C INTERNAL - ARRAYS FOR QUADRATURE DATA
      DIMENSION A1(3,3), A2(3,3), A3(3,3), A4(3,3), B1(3,3), B2(3,3),
     2 B3(3,3), B4(3,3), B5(3,3), B6(3,3), B7(3,3), B8(3,3), G(3,3),
     3 XI(3), ETA(3), W(3)
C INTERNAL - TEMPORARY STORAGE
      DIMENSION BB(3,8), E(8,8), EE(8,8), FF(8), KT(8,6)
C SET UP DATA FOR 3X3-POINT GAUSSIAN QUADRATURES ON UNIT SQUARE
C COORDINATES
      DATA XI/-0.7745967,0.,0.7745967/
      DATA ETA/-0.7745967,0.,0.7745967/
C WEIGHTS
      DATA W/0.5555556,0.8888889,0.5555556/
C PRODUCTS
      DATA A1/3.149193,1.774597,0.4,1.774597,1.,0.2254033,0.4,0.2254033,
     2 0.05080665/
      DATA A2/0.4,1.774597,3.149193,0.2254033,1.,1.774597,0.05080665,
     2 0.2254033,0.4/
      DATA A3/0.05080665,0.2254033,0.4,0.2254033,1.,1.774597,0.4,
     2 1.774597,3.149193/
      DATA A4/0.4,0.2254033,0.05080665,1.774597,1.,0.2254033,3.149193,
     2 1.774597,0.4/
```

```
C PRINT CONTROL
  901 FORMAT(1H0,42X,31HFOUR NODE QUADRILATERAL ELEMENT,I5,/,52X,18HCORN
     2ER COORDINATES,/,14X,24H----X1----  ----Y1----  ----X2----  ----Y2
     3----  ----X3----  ----Y3----  ----X4----  ----Y4----,/,14X,
     4 8(E10.3,2X))
  902 FORMAT(1H0,40HDEGENERATE ELEMENT' EXECUTION TERMINATES)
  903 FORMAT(1H0,38HBOOMERANG ELEMENT' EXECUTION CONTINUES)
  904 FORMAT(1H0,* ZERO THICKNESS OR ELASTIC CONSTANTS STIFFNESS=ZERO*)
C ENTRY? DEGENERACY AND OTHER CHECKS
      B(3,9) = 0.
      B(2,9) = 0.
      B(3,9) = 0.
      IFLAG = 0
C COORDINATE TRANSFER
      X1 = COORD(1)
      Y1 = COORD(2)
      X2 = COORD(4)
      Y2 = COORD(5)
      X3 = COORD(7)
      Y3 = COORD(8)
      X4 = COORD(10)
      Y4 = COORD(11)
C AREAS
      A123 = ABS(X2*Y3+X3*Y1+X1*Y2-X3*Y2-X1*Y3-X2*Y1)
      A134 = ABS(X3*Y4+X4*Y1+X1*Y3-X4*Y3-X1*Y4-X3*Y1)
      A124 = ABS(X2*Y4+X4*Y1+X1*Y2-X4*Y2-X1*Y4-X2*Y1)
      A234 = ABS(X3*Y4+X4*Y2+X2*Y3-X4*Y3-X2*Y4-X3*Y2)
C CHECK ELASTIC CONSTANTS AND THICKNESS
      DO 1 I = 1,3
      DO 1 J = 1,3
      IF (E(I,J)  NE. 0.) GO TO 2
    1 CONTINUE
      IFLAG = 1
    2 IF (THK .EQ. 0.) IFLAG = 1
C BOOMERANG
      IF (A123+A134 .NE. 0.) DIFF = ABS((A123+A134-A124-A234)/(A123
     2 +A134))
      IF (DIFF .GT. 0.3) IFLAG = IFLAG+2
C DEGENERACY
      IF (A123+A234 .EQ. 0. .OR. A124+A234 .EQ. 0.) IFLAG = 4
C DUMP RESULTS OF TESTS
      IF (IFLAG .EQ. 0) GO TO 3
      WRITE (KW,901) LNUM,X1,Y1,X2,Y2,X3,Y3,X4,Y4
      IF (IFLAG .EQ. 1 .OR. IFLAG .EQ. 3) WRITE (KW,904)
      IF (IFLAG .EQ. 2 .OR. IFLAG .EQ. 3) WRITE (KW,903)
      IF (IFLAG .NE. 4) GO TO 3
      WRITE (KW,902)
      STOP
 1000 CONTINUE
      F2=0.
      F4=0.
      F6=0.
      F8=0.
      FG=RHO*THK*YGRV/64.
      GOTO 1010
```

C2-61

```
 1100 CONTINUE
      F6=F6+ABS((-1.-XI(I))*G(I,J)*(-1.-ETA(J)))*W(I)*W(J)
      F8=F8+ABS((1.-XI(I))*G(I,J)*(-1.-ETA(J)))*W(I)*W(J)
      F2=F2+ABS((1.-XI(I))*G(I,J)*(1.-ETA(J)))*W(I)*W(J)
      F4=F4+ABS((-1.-XI(I))*G(I,J)*(1.-ETA(J)))*W(I)*W(J)
      GOTO 1110
 1200 CONTINUE
      F2=FG*F2
      F4=FG*F4
      F6=FG*F6
      F8=FG*F8
      F(2)=F2+F(2)
      F(4)=F4+F(4)
      F(6)=F6+F(6)
      F(8)=F8+F(8)
      IF(OUT)WRITE(KW,1201)LNUM,F2,F4,F6,F8
 1201 FORMAT(* ELEMENT=*,I5,* NODAL GRAV FORCE=*,1P4E10.2)
      GOTO 1210
C PERPARE FOR CALCULATION OF STIFFNESS MATRIX
    3 DO 4 K = 1,36
    4 SM(K) = 0.
      IF(GRAVB.AND.RHO.NE.0.)GOTO 1000
 1010 CONTINUE
C PRE-GAUSS COMPUTATIONS
      DO 5 I = 1,3
      DO 5 J = 1,3
C JACOBIAN
      G(I,J) = (A1(I,J)+A2(I,J))*(X1*Y2-X2*Y1)+(A2(I,J)+A3(I,J))*(X2*Y3-
     2X3*Y2)+(A3(I,J)+A4(I,J))*(X3*Y4-X4*Y3)+(A4(I,J)+A1(I,J))*(X4*Y1-
     3X1*Y4)+(A2(I,J)-A4(I,J))*(X3*Y1-X1*Y3)+(A1(I,J)-A3(I,J))*(X2*Y4-
     4X4*Y2)
      IF(GRAVB.AND.RHO.NE.0.)GOTO 1100
 1110 CONTINUE
C CENTRAL JACOBIAN TERM FOR STRESS-DISPLACEMENT MATRIX
      IF (I .EQ. 2 .AND. J .EQ. 2) GC = 1./G(2,2)
C WEIGHTING FACTORS X 1/JACOBIAN
      G(I,J) = ABS(2.*W(I)*W(J)/G(I,J))
C STRAIN-DISPLACEMENT MATRIX ENTRIES
      B1(I,J) = A1(I,J)*(Y2-Y4)+A4(I,J)*(Y3-Y4)+A2(I,J)*(Y2-Y3)
      B3(I,J) = A2(I,J)*(Y3-Y1)+A3(I,J)*(Y3-Y4)+A1(I,J)*(Y4-Y1)
      B5(I,J) = A3(I,J)*(Y4-Y2)+A2(I,J)*(Y1-Y2)+A4(I,J)*(Y4-Y1)
      B7(I,J) = A4(I,J)*(Y1-Y3)+A1(I,J)*(Y1-Y2)+A3(I,J)*(Y2-Y3)
      B2(I,J) = A1(I,J)*(X4-X2)+A4(I,J)*(X4-X3)+A2(I,J)*(X3-X2)
      B4(I,J) = A2(I,J)*(X1-X3)+A3(I,J)*(X4-X3)+A1(I,J)*(X1-X4)
      B6(I,J) = A3(I,J)*(X2-X4)+A2(I,J)*(X2-X1)+A4(I,J)*(X1-X4)
    5 B8(I,J) = A4(I,J)*(X3-X1)+A1(I,J)*(X2-X1)+A3(I,J)*(X3-X2)
      IF(GRAVB.AND.RHO.NE.0.)GOTO 1200
 1210 CONTINUE
C GAUSSIAN QUADRATURES LOOP
      DO 6 I = 1,3
      DO 6 J = 1,3
      SM(1) = SM(1)+G(I,J)*(C(1,1)*B1(I,J)**2+2.*C(1,3)*B1(I,J)*B2(I,J)
     2 +C(3,3)*B2(I,J)**2)
      SM(2) = SM(2)+G(I,J)*(C(1,3)*B1(I,J)**2+C(2,3)*B2(I,J)**2
     2 +(C(1,2)+C(3,3))*B1(I,J)*B2(I,J))
```

C 2-62

```
      SM(3) = SM(3)+G(I,J)*(C(2,2)*B2(I,J)**2+2.*C(2,3)*B1(I,J)*B2(I,J)
     2 +C(3,3)*B1(I,J)**2)
      SM(4) = SM(4)+G(I,J)*(C(1,1)*B1(I,J)*B3(I,J)+C(1,3)*(B2(I,J)*
     2 B3(I,J)+B1(I,J)*B4(I,J))+C(3,3)*B2(I,J)*B4(I,J))
      SM(5) = SM(5)+G(I,J)*(C(1,2)*B2(I,J)*B3(1,J)+C(1,3)*B1(I,J)*
     2 B3(I,J)+C(2,3)*B2(1,J)*B4(I,J)+C(3,3)*B1(I,J)*B4(I,J))
      SM(6) = SM(6)+G(I,J)*(C(1,1)*B3(I,J)**2+2.*C(1,3)*B3(I,J)*B4(I,J)
     2 +C(3,3)*B4(I,J)**2)
      SM(7) = SM(7)+G(I,J)*(C(1,2)*B1(I,J)*B4(I,J)+C(2,3)*B2(I,J)*
     2 B4(I,J)+C(1,3)*B1(I,J)*B3(I,J)+C(3,3)*B2(I,J)*B3(I,J))
      SM(8) = SM(8)+G(I,J)*(C(2,2)*B2(I,J)*B4(I,J)+C(2,3)*(B1(I,J)*
     2 B4(I,J)+B2(I,J)*B3(I,J))+C(3,3)*B1(I,J)*B3(I,J))
      SM(9) = SM(9)+G(I,J)*(C(1,3)*B3(I,J)**2+C(2,3)*B4(I,J)**2
     2 +(C(1,2)+C(3,3))*B3(1,J)*B4(I,J))
      SM(10) = SM(10)+G(I,J)*(C(2,2)*B4(I,J)**2+C(3,3)*B3(I,J)**2
     2 +2.*C(2,3)*B3(I,J)*B4(I,J))
      SM(11) = SM(11)+G(I,J)*(C(1,1)*B1(I,J)*B5(I,J)+C(1,3)*(B2(I,J)*
     2 B5(I,J)+B1(I,J)*B6(I,J))+C(3,3)*B2(I,J)*B6(I,J))
      SM(12) = SM(12)+G(I,J)*(C(1,2)*B2(I,J)*B5(I,J)+C(1,3)*B1(1,J)*
     2 B5(I,J)+C(2,3)*B2(I,J)*B6(I,J)+C(3,3)*B1(I,J)*B6(1,J))
      SM(13) = SM(13)+G(I,J)*(C(1,1)*B3(I,J)*B5(I,J)+C(1,3)*(B4(I,J)*
     2 B5(I,J)+B3(I,J)*B6(I,J))+C(3,3)*B4(I,J)*B6(I,J))
      SM(14) = SM(14)+G(1,J)*(C(1,2)*B4(I,J)*B5(I,J)+C(1,3)*B3(I,J)*
     2 B5(I,J)+C(2,3)*B4(I,J)*B6(I,J)+C(3,3)*B3(1,J)*B6(I,J))
      SM(15) = SM(15)+G(I,J)*(C(1,1)*B5(I,J)**2+C(3,3)*B6(I,J)**2
     2 +2.*C(1,3)*B5(I,J)*B6(I,J))
      SM(16) = SM(16)+G(1,J)*(C(1,2)*B1(1,J)*B6(I,J)+C(2,3)*B2(I,J)*
     2 B6(I,J)+C(1,3)*B1(I,J)*B5(I,J)+C(3,3)*B2(I,J)*B5(I,J))
      SM(17) = SM(17)+G(I,J)*(C(2,2)*B2(I,J)*B6(I,J)+C(2,3)*(B1(I,J)*
     2 B6(I,J)+B2(I,J)*B5(I,J))+C(3,3)*B1(I,J)*B5(I,J))
      SM(18) = SM(18)+G(I,J)*(C(1,2)*B3(I,J)*B6(I,J)+C(2,3)*B4(I,J)*
     2 B6(I,J)+C(1,3)*B3(1,J)*B5(I,J)+C(3,3)*B4(I,J)*B5(I,J))
      SM(19) = SM(19)+G(I,J)*(C(2,2)*B4(I,J)*B6(I,J)+C(2,3)*(B3(I,J)*
     2 B6(I,J)+B4(I,J)*B5(I,J))+C(3,3)*B3(I,J)*B5(1,J))
      SM(20) = SM(20)+G(I,J)*(C(1,3)*B5(I,J)**2+C(2,3)*B6(I,J)**2
     2 +(C(1,2)+C(3,3))*B5(I,J)*B6(I,J))
      SM(21) = SM(21)+G(1,J)*(C(2,2)*B6(I,J)**2+C(3,3)*B5(I,J)**2
     2 +2.*C(2,3)*B5(I,J)*B6(1,J))
      SM(22) = SM(22)+G(I,J)*(C(1,1)*B1(I,J)*B7(I,J)+C(1,3)*(B2(I,J)*
     2 B7(I,J)+B1(I,J)*B8(I,J))+C(3,3)*B2(I,J)*B8(I,J))
      SM(23) = SM(23)+G(I,J)*(C(1,2)*B2(I,J)*B7(I,J)+C(1,3)*B1(I,J)*
     2 B7(I,J)+C(2,3)*B2(1,J)*B8(I,J)+C(3,3)*B1(1,J)*B8(I,J))
      SM(24) = SM(24)+G(I,J)*(C(1,1)*B3(I,J)*B7(I,J)+C(1,3)*(B4(I,J)*
     2 B7(I,J)+B3(I,J)*B8(I,J))+C(3,3)*B4(I,J)*B8(I,J))
      SM(25) = SM(25)+G(I,J)*(C(1,2)*B4(I,J)*B7(1,J)+C(1,3)*B3(I,J)*
     2 B7(I,J)+C(2,3)*B4(I,J)*B8(I,J)+C(3,3)*B3(I,J)*B8(I,J))
      SM(26) = SM(26)+G(I,J)*(C(1,1)*B5(I,J)*B7(I,J)+C(1,3)*(B6(I,J)*
     2 B7(I,J)+B5(I,J)*B8(I,J))+C(3,3)*B6(I,J)*B8(I,J))
      SM(27) = SM(27)+G(1,J)*(C(1,2)*B6(I,J)*B7(I,J)+C(1,3)*B5(I,J)*
     2 B7(I,J)+C(2,3)*B6(I,J)*B8(I,J)+C(3,3)*B5(I,J)*B8(I,J))
      SM(28) = SM(28)+G(I,J)*(C(1,1)*B7(I,J)**2+C(3,3)*B8(I,J)**2
     2 +2.*C(1,3)*B7(I,J)*B8(I,J))
      SM(29) = SM(29)+G(I,J)*(C(1,2)*B1(I,J)*B8(I,J)+C(2,3)*B2(I,J)*
     2 B8(I,J)+C(1,3)*B1(1,J)*B7(I,J)+C(3,3)*B2(I,J)*B7(I,J))
      SM(30) = SM(30)+G(I,J)*(C(2,2)*B2(I,J)*B8(I,J)+C(2,3)*(B1(I,J)*
```

C2-63

```
     2 B8(I,J)+B2(I,J)*B7(I,J))+C(3,3)*B1(I,J)*B7(I,J))
      SM(31) = SM(31)+G(I,J)*(C(1,2)*B3(I,J)*B8(I,J)+C(2,3)*B4(I,J)*
     2 B8(I,J)+C(1,3)*B3(1,J)*B7(I,J)+C(3,3)*B4(I,J)*B7(I,J))
      SM(32) = SM(32)+G(I,J)*(C(2,2)*B4(I,J)*B8(I,J)+C(2,3)*(B3(1,J)*
     2 B8(I,J)+B4(I,J)*B7(I,J))+C(3,3)*B3(I,J)*B7(I,J))
      SM(33) = SM(33)+G(I,J)*(C(1,2)*B5(I,J)*B8(I,J)+C(2,3)*B6(I,J)*
     2 B8(I,J)+C(1,3)*B5(I,J)*B7(I,J)+C(3,3)*B6(I,J)*B7(I,J))
      SM(34) = SM(34)+G(I,J)*(C(2,2)*B6(I,J)*B8(I,J)+C(2,3)*(B5(I,J)*
     2 B8(I,J)+B6(I,J)*B7(I,J))+C(3,3)*B5(I,J)*B7(I,J))
      SM(35) = SM(35)+G(I,J)*(C(1,3)*B7(I,J)**2+C(2,3)*B8(I,J)**2
     2 +(C(1,2)+C(3,3))*B7(I,J)*B8(I,J))
     6 SM(36) = SM(36)+G(I,J)*(C(2,2)*B8(I,J)**2+C(3,3)*B7(I,J)**2
      2 +2.*C(2,3)*B7(I,J)*B8(I,J))
C COMMON FACTOR
      DO 7 K = 1,36
    7 SM(K) = THK*SM(K)/32.
C FORCES
    8 DO 9 I = 1,8
    9 F(I) = 0.
C STRESS-DISPLACEMENT MATRIX
      DO 10 I = 1,2
      DO 10 J = 1,8
   10 BB(I,J) = 0.
      BB(1,1) = 2.*(Y2-Y4)*GC
      BB(1,3) = 2.*(Y3-Y1)*GC
      BB(1,5) = -BB(1,1)
      BB(1,7) = -BB(1,3)
      BB(2,2) = 2.*(X4-X2)*GC
      BB(2,4) = 2.*(X1-X3)*GC
      BB(2,6) = -BB(2,2)
      BB(2,8) = -BB(2,4)
      DO 11 J = 1,7,2
   11 BB(3,J) = BB(2,J+1)
      DO 12 J = 2,8,2
   12 BB(3,J) = BB(1,J-1)
      DO 13 I = 1,3
      DO 13 J = 1,8
      B(I,J) = 0.
      DO 13 K = 1,3
   13 B(I,J) = B(I,J)+C(I,K)*BB(K,J)
C ROTATIONS, IF ANY
      IF (NROT .EQ. 0) GO TO 21
C PREPARE ROTATION MATRIX
      DO 15 I = 1,8
      DO 14 J = 1,8
   14 RT(I,J) = 0.
   15 RT(I,I) = 1.
      DO 16 N = 1,NROT
      J = 2*NODE(N)
      I = J-1
      RT(I,1) = COS(ANGLE(N))
      RT(J,J) = RT(I,I)
      RT(I,J) = SIN(ANGLE(N))
   16 RT(J,I) = -RT(1,J)
C PREPARE TEMPORARY STORAGE
```

C 2-64

```
                      K = 0
                      DO 17 I = 1,8
                      FF(I) = F(I)
                      DO 17 J = 1,I
  280                 K = K+1
                      E(I,J) = SM(K)
                   17 E(J,I) = SM(K)
                      DO 18 I = 1,3
                      DO 18 J = 1,8
  285                 BB(I,J) = B(I,J)
                C TRANSFORMATION
                      DO 19 I = 1,8
                      F(I) = 0.
                      DO 19 J = 1,8
  290                 F(I) = F(I)+RT(I,J)*FF(J)
                      IF (I .LE. 3) B(I,J) = 0.
                      EE(I,J) = 0.
                      DO 19 K = 1,8
                      IF (I .LE. 3) B(I,J) = B(I,J)+BB(I,K)*RT(J,K)
  295                 DO 19 N = 1,8
                   19 EE(I,J) = EE(I,J)+RT(I,K)*E(K,N)*RT(J,N)
                C RECONVERT STIFFNESS MATRIX TO LTV FORM
                      K = 0
                      DO 20 I = 1,8
  300                 DO 20 J = 1,I
                      K = K+1
                   20 SM(K) = EE(I,J)
                   21 RETURN
                      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3 QUAD4

VARIABLES	SN	TYPE							
0	ANGLE	REAL	ARRAY	F.P.	1610	A1	REAL	ARRAY	
1571	A123	REAL			1573	A124	REAL		
1572	A134	REAL			1621	A2	REAL	ARRAY	
1574	A234	REAL			1632	A3	REAL	ARRAY	
1643	A4	REAL	ARRAY		0	B	REAL	ARRAY	F.P.
2006	BB	REAL	ARRAY		1654	B1	REAL	ARRAY	
1665	B2	REAL	ARRAY		1676	B3	REAL	ARRAY	
1707	B4	REAL	ARRAY		1720	B5	REAL	ARRAY	
1731	B6	REAL	ARRAY		1742	B7	REAL	ARRAY	
1753	B8	REAL	ARRAY		0	C	REAL	ARRAY	F.P.
0	COORD	REAL	ARRAY	F.P.	1577	DIFF	REAL		
2036	E	REAL	ARRAY		2136	EE	REAL	ARRAY	
2000	ETA	REAL	ARRAY		0	F	REAL	ARRAY	F.P.
2236	FF	REAL	ARRAY		1604	FC	REAL		
1600	F2	REAL			1601	F4	REAL		

C2-65

VARIABLES	SN	TYPE	RELOCATION						
1602	F6	REAL			1603	F8	REAL		
1764	G	REAL	ARRAY		1606	GC	REAL		
0	GRAVB	LOGICAL		F.P.	1575	I	INTEGER		
1560	IFLAG	INTEGER			1576	J	INTEGER		
1605	K	INTEGER			0	KW	INTEGER		F.P.
0	LNUM	INTEGER		F.P.	1607	N	INTEGER		
0	NODE	INTEGER	ARRAY	F.P.	0	NROT	INTEGER		F.P.
0	OUT	LOGICAL		F.P.	0	RHO	REAL		F.P.
2246	RT	REAL	ARRAY		0	SM	REAL	ARRAY	F.P.
0	TEMP	REAL	ARRAY	F.P.	0	THK	REAL		F.P.
2003	W	REAL	ARRAY		1775	XI	REAL	ARRAY	
1561	X1	REAL			1563	X2	REAL		
1565	X3	REAL			1567	X4	REAL		
0	YGRV	REAL		F.P.	1562	Y1	REAL		
1564	Y2	REAL			1566	Y3	REAL		
1570	Y4	REAL							

EXTERNALS	TYPE	ARGS				
COS	REAL	1	LIBRARY	SIN	REAL	1 LIBRARY

INLINE FUNCTIONS	TYPE	ARGS	
ABS	REAL	1	INTRIN

STATEMENT LABELS

0	1		71	2		211	3		
0	4		0	5		0	6		
0	7		0	8	INACTIVE	0	9		
0	10		0	11		0	12		
0	13		0	14		0	15		
0	16		0	17		0	18		
0	19		0	20		1330	21		
1435	701	FMT	1461	902	FMT	1467	903	FMT	
1475	904	FMT	140	1000		220	1010		
147	1100		317	1110		171	1200		
1541	1201	FMT	375	1210					

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES		
63	1	* I	84 87	5B		EXITS	NOT INNER
65	1	* J	85 87	2B	INSTACK	EXITS	
213	4	K	130 131	2B	INSTACK		
272	5	* I	135 156	77B		EXITS	NOT INNER
301	5	* J	136 156	66B		ENTRIES	EXITS
406	6	* I	160 232	474B		NOT INNER	
410	6	J	161 232	470B	OPT		
1105	7	K	235 236	2B	INSTACK		
1111	9	I	238 239	2B	INSTACK		
1114	10	* I	241 243	4B		NOT INNER	
1115	10	J	242 243	2B	INSTACK		
1137	11	J	252 253	2B	INSTACK		
1142	12	J	254 255	2B	INSTACK		
1146	13	* I	256 260	16B		NOT INNER	
1150	13	* J	257 260	12B		NOT INNER	
1155	13	K	259 260	3B	INSTACK		
1167	15	* I	264 267	5B		NOT INNER	
1170	14	J	265 266	2B	INSTACK		

C2-66

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
1175	16	*	N	268 274	20B	EXT REFS
1221	17	*	I	277 282	13B	NOT INNER
1226	17		J	279 282	3B	INSTACK
1237	18	*	I	283 285	5B	NOT INNER
1241	18		J	284 285	2B	INSTACK
1246	19	*	I	287 296	52B	NOT INNER
1255	19	*	J	289 296	40B	NOT INNER
1274	19	*	K	293 296	15B	NOT INNER
1303	19		N	295 296	4B	INSTACK
1323	20	*	I	299 302	5B	NOT INNER
1325	20		J	300 302	2B	INSTACK

STATISTICS

PROGRAM LENGTH 2420B 1296

120000B SCM USED

C2-67

```
      SUBROUTINE ROTATE(NODE,IROW,JROW,KROW,ZANGLE,YANGLE,XANGLE,REALK,
     1INTGRK)
C*********************************************************************
C FINITE ELEMENT ANALYSIS BASIC LIBRARY SUBROUTINE-VERSION 2
C
C
      DIMENSION REALK(2),INTGRK(2)
      LEVEL 2,REALK,INTGRK
      LEVEL 2,REALKK
      DIMENSION A(3,3), B(3,3), IR(3), INFO(10), KDR(3), PERM(3,3),
     2 PROTOR(3,3), ROTOR(3,3), TEMPC(3), TEMPR(3), TEMPCC(3), TEMPRR(3)
      LOGICAL OUT,DEBUG1,DEBUG2,SPACE
      COMMON /IO/ KR, KW, KP, KT1, KT2, KT3, OUT,DEBUG1,DEBUG2
      COMMON /SIZE/ NET, NDT
      COMMON /BEGIN/ ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON /SHIFT/ KUNIT,KLEN,KBEGIN,KEND,KMIN,KMAX,RBEGIN,REND,
     1 KOFF,DISK,SPACE
      COMMON/KK  /REALKK(2)
C
C VERSION 2 RELEASE 1..CORRECTIONS TO SEQ. NOS. 70 THRU 76 (MARCH 1973)
C
C
C
C PRINT CONTROL
  901 FORMAT(1H0,41X,26HROTATION REQUESTED AT NODE,I10,/,24X,29HASSOCIAT
     1ED MASTER NUMBERS FOR,8X,5HFIRST,6X,6HSECOND,7X,5HTHIRD,3X,3HDOF,/
     2,54X,3(2X,I10),/,27X,27HEULER ANGLES IN DEGREES ARE,8X,7HTHETA Z,5
     3X,7HTHETA Y,5X,7HTHETA X,/,57X,3(2X,E10.3))
  902 FORMAT (1X,79HHALF TRANSFORM (K)*(THETA TRANSPOSE) WAS APPLIED TO
     1THE FOLLOWING DOF BELOW ROW,I10,16H IN THE K MATRIX)
  903 FORMAT (10(2X,I10))
  904 FORMAT(103H0ABOVE REQUEST INCONSISTENT DUE TO INEQUALITY OF LEADIN
     1G NON-ZERO ENTRY COLUMN NUMBERS OF ARGUMENT ROWS,/,9X,3HROW,1X,11H
     2LNZE COL NO,/,3(2X,I10,2X,I10,/))
  905 FORMAT (1X,41HEXECUTION TERMINATED IN SUBROUTINE ROTATE)
  906 FORMAT (54H0ABOVE REQUEST INCONSISTENT DUE TO REPEATED ROW NUMBER)
  907 FORMAT(73H0ABOVE REQUEST INVALID DUE TO INCORRECT LOCATION OF UNDE
     1FINED ROW NUMBERS)
      IR(1)=IROW
      IR(2)=JROW
      IR(3)=KROW
C PRINT ENTRY MESSAGE
      NNODE=IABS(NODE)
      IF(DEBUG2)WRITE(KW,901)NNODE,(IR(I),I=1,3),ZANGLE,YANGLE,XANGLE
C ESTABLISH DOF ROTATION LIMIT
      LIMIT = 3
      IF (KROW .LE. 0) LIMIT = 2
      IF(IROW .GT. 0 .AND. JROW .GT. 0) GO TO 100
      WRITE(KW,907)
      WRITE(KW,905)
      STOP
C GET LNZE COL NOS AND TEST FOR EQUALITY
  100 DO 1 I = 1,LIMIT
      J = ILNZ+IR(I)-1
    1 INFO(I) = INTGRK(J)
```

C2-68

```
      IF (LIMIT .EQ. 3 .AND. INFO(1) .EQ. INFO(2) .AND. INFO(2) .EQ.
     2 INFO(3)) GO TO 2
      IF (LIMIT .EQ. 2 .AND. INFO(1) .EQ. INFO(2)) GO TO 2
C SOMETHING WRONG - PRINT ERROR MESSAGE
   60 WRITE (KW,904) (IR(I), INFO(I), I = 1,LIMIT)
      WRITE (KW,905)
      STOP
C CHECK FOR KEYPUNCH ERROR IN ARGUMENT LIST
    2 IF (IROW .NE. JROW .AND. JROW .NE. KROW .AND. KROW .NE. IROW)
   65  2 GO TO 3
      WRITE (KW,906)
      WRITE (KW,905)
      STOP
C OK TO PROCEED - FORM ROTOR MATRIX AFTER CONVERTING
   70 C ANGLES TO RADIAN MEASURE
    3 Z = 3.14159*ZANGLE/180.
      Y = 3.14159*YANGLE/180.
      X = 3.14159*XANGLE/180.
      ROTOR(1,1) = COS(Y)*COS(Z)
   75 ROTOR(1,2) = COS(Y)*SIN(Z)
      ROTOR(1,3) = -SIN(Y)
      ROTOR(2,2) = COS(X)*COS(Z)+SIN(X)*SIN(Y)*SIN(Z)
      ROTOR(2,3) = SIN(X)*COS(Y)
      ROTOR(3,3) = COS(X)*COS(Y)
   80 ROTOR(2,1) = SIN(X)*SIN(Y)*COS(Z)-COS(X)*SIN(Z)
      ROTOR(3,1) = SIN(X)*SIN(Z)+COS(X)*SIN(Y)*COS(Z)
      ROTOR(3,2) = COS(X)*SIN(Y)*SIN(Z)-SIN(X)*COS(Z)
C     SKIP K MATRIX SECTION IF A RE-ROTATION
      IF(NODE .LT. 0) GO TO 38
   85 C ESTABLISH LEAST AND LARGEST DOF NUMBERS FROM ARGUMENT LIST
      LEAST = IROW
      LARGST = JROW
      DO 4 I = 1,LIMIT
      IF (IR(I) .LT. LEAST) LEAST = IR(I)
   90 IF (IR(I) .GT. LARGST) LARGST = IR(I)
    4 CONTINUE
      IF(.NOT.SPACE)CALL SHIFT(REALKK,INTGRK,LEAST,LARGST)
C APPLY (ROTOR)*(K) TO ROWS IROW, JROW, KROW - ONE COLUMN AT A TIME
C FROM LNZE TO LEAST-1 -- SKIP SECTION IF LEAST ROW = 1ST ROW
   95 INIT = INFO(1)
      IF (LEAST .EQ. 1 .OR. INIT .EQ. LEAST) GO TO 8
      LAST = LEAST-1
      DO 7 MCOL = INIT,LAST
C GET ENTRIES OF K MATRIX INTO TEMPR, ZERO TEMPC
  100 DO 5 I = 1,LIMIT
      J = IKOUNT+IR(I)-1
      KADR = INTGRK(J)+MCOL-KOFF
      TEMPR(I) = REALKK(KADR)
    5 TEMPC(I) = 0.
C APPLY TRANSFORM
  105 DO 6 I = 1,LIMIT
      DO 6 J = 1,LIMIT
    6 TEMPC(I) = TEMPC(I)+ROTOR(I,J)*TEMPR(J)
C RESTORE TRANSFORMED COLUMN TO K MATRIX
  110 DO 7 I = 1,LIMIT
```

C2-69

```
            J = IKOUNT+IR(I)-1
            KADR = INTGRK(J)+NCOL-KOFF
          7 REALKK(KADR) = TEMPC(I)
      C TEST FOR ROWS INTERVENING BETWEEN IROW, JROW AND KROW
        8 IF (LARGST .EQ. LEAST+LIMIT-1) GO TO 26
      C**********************************************************************
      C SPECIAL ALGORITHMS FOR INTERVENING ROWS
      C**********************************************************************
      C PRINT HEADING
            IF(DEBUG2)WRITE (KW,902) LEAST
      C FIND VALUE OF MIDDLE ARGUMENT ROW IF 3 DOF ARE BEING ROTATED
            IF (LIMIT .EQ. 2) GO TO 10
            DO 9 I = 1,3
            IF (IR(I) .NE. LEAST .AND. IR(I) .NE. LARGST) MIDDLE = IR(I)
        9 CONTINUE
      C TEST TO SEE IF ARGUMENT ROWS WERE GIVEN IN ASCENDING ORDER.  IF
      C NOT, A PERMUTATION SIMILARITY TRANSFORM MUST BE APPLIED TO THE
      C ROTOR MATRIX TO TRANSFORM THE INTERVENING ROWS PROPERLY
            IF (IROW .LT. JROW .AND. JROW .LT. KROW) GO TO 15
       10 IF (KROW .EQ. 0 .AND. IROW .LT. JROW) GO TO 15
      C PREPARE STORAGE
            DO 11 I = 1,LIMIT
            DO 11 J = 1,LIMIT
            A(I,J) = 0.
            PROTOR(I,J) = 0.
       11 PERM(I,J) = 0.
            DO 12 I = 1,LIMIT
            IF (IR(I) .EQ. LEAST) PERM(1,I) = 1.
            IF (IR(I) .EQ. LARGST) PERM(LIMIT,I) = 1.
            IF (LIMIT .EQ. 2) GO TO 12
            IF (IR(I) .EQ. MIDDLE) PERM(2,I) = 1.
       12 CONTINUE
            DO 13 I = 1,LIMIT
      C APPLY SIMILARITY TRANSFORM
            DO 13 J = 1,LIMIT
            DO 13 K = 1,LIMIT
       13 A(I,J) = A(I,J)+PERM(I,K)$ROTOR(K,J)
            DO 14 I = 1,LIMIT
            DO 14 J = 1,LIMIT
            DO 14 K = 1,LIMIT
       14 PROTOR(I,J) = PROTOR(I,J)+A(I,K)$PERM(J,K)
            GO TO 17
      C PERMUTATION NOT REQUIRED
       15 DO 16 I = 1,3
            DO 16 J = 1,3
       16 PROTOR(I,J) = ROTOR(I,J)
      C INITIALIZE INFO VECTOR FILL INDEX AND ESTABLISH LOOP LIMITS FOR
      C INTERVENING ROWS
       17 INFOX = 0
            INIT = LEAST+1
      C LAST ROW TO BE DONE DEPENDS ON WHETHER 2 OR 3 DOF ARE BEING ROTATED
            IF (LIMIT .EQ. 2) LAST = LARGST-1
            IF (LIMIT .EQ. 3) LAST = MIDDLE-1
      C LOOP OVER REMAINING COLUMNS, (PROTOR)$(K), AND FIRST ROWS,
      C (K)$(PROTOR-T), FROM INIT TO LAST -- SKIP IF MIDDLE DOF = LEAST+1
```

C2-70

```
      C WHEN 3 DOF ARE BEING ROTATED
           IF (LIMIT .EQ. 3 .AND. LAST .EQ. LEAST) GO TO 25
      C THIS IS THE RE-ENTRY POINT IF 3 DOF ARE BEING ROTATED AND ROWS
      C INTERVENE BETWEEN MIDDLE AND LARGEST
170   C
        18 CONTINUE
           IF(.NOT.SPACE.AND.(INIT.LT.RBEGIN.OR.LAST.GT.REND))
          1 CALL SHIFT(REALKK,INTGRK,INIT,LAST)
      C
175        DO 24 MROW = INIT,LAST
      C CHECK LNZE COL NO OF CURRENT ROW -- IF .GT. LOWER ROTATED DOF,
      C WHICH = INIT-1, K MATRIX CONTAINS STORED OR NON-STORED ZEROS AND
      C BOTH ROW AND COLUMN TRANSFORM CAN BE SKIPPED
           J = ILNZ+MROW-1
180        IF (INTGRK(J) .GT. INIT-1) GO TO 24
      C INCREMENT INFO FILL INDEX AND STORE CURRENT ROW NUMBER
           INFOX = INFOX+1
           INFO(INFOX) = MROW
      C IF TEN ROW NUMBERS HAVE BEEN COLLECTED, PRINT THEM AND RESET INDEX
185        IF (INFOX .LT. 10) GO TO 19
           IF(DEBUG2)WRITE (KW,903) (INFO(J), J = 1,10)
           INFOX = 0
      C GET CORRECT ABSOLUTE ADDRESSES FOR K MATRIX ENTRIES, REPLACING COLUMN
      C SECTORS ABOVE LOWER TRIANGLE BY ROW SECTORS BELOW, AND VICE-VERSA
190      19 J = IKOUNT+MROW-1
           KDR(1) = INTGRK(J)
           J = IKOUNT+LAST
           KDR(2) = INTGRK(J)+MROW
           IF (INIT .EQ. LEAST+1) GO TO 20
195        KDR(2) = KDR(1)+INIT-1
         20 KDR(1) = KDR(1)+LEAST
           IF (LIMIT .EQ. 2) GO TO 21
           J = IKOUNT+LARGST-1
           KDR(3) = INTGRK(J)+MROW
200   C PICK UP K MATRIX ENTRIES IN TEMPC, TEMPR AND ZERO TEMPCC, TEMPRR
         21 DO 22 I = 1,LIMIT
           KADR = KDR(I)-KOFF
           TEMPC(I) = REALKK(KADR)
           TEMPR(I) = REALKK(KADR)
205        TEMPCC(I) = 0.
         22 TEMPRR(I) = 0.
      C APPLY TRANSFORMS (PROTOR)*(TEMPC) AND (TEMPR)*(PROTOR-TRANSPOSE)
           DO 23 I = 1,LIMIT
           DO 23 J = 1,LIMIT
210        TEMPCC(I) = TEMPCC(I)+PROTOR(I,J)*TEMPC(J)
         23 TEMPRR(I) = TEMPRR(I)+TEMPR(J)*PROTOR(I,J)
      C RESTORE TRANSFORMED VALUES TO K MATRIX
           KADR = KDR(1)-KOFF
           REALKK(KADR) = TEMPCC(1)
215        KADR = KDR(2)-KOFF
           IF (INIT .EQ. LEAST+1) REALKK(KADR) = TEMPRR(2)
           IF (INIT .NE. LEAST+1) REALKK(KADR) = TEMPCC(2)
           IF (LIMIT .EQ. 2) GO TO 24
           KADR = KDR(3)-KOFF
220        REALKK(KADR) = TEMPRR(3)
```

C2-71

```
   24 CONTINUE
C PRINT PARTIALLY FILLED INFO VECTOR IF THERE IS AT LEAST ONE ROW,
C RESET INDEX
      IF (INFOX .EQ. 0) GO TO 25
225   IF(DEBUG2)WRITE (KW,903) (INFO(I), I = 1,INFOX)
      INFOX = 0
C COMPLETION TEST
   25 IF (LIMIT .EQ. 2 .OR. INIT .NE. LEAST+1) GO TO 26
C CHECK FOR INTERVENING ROWS BETWEEN MIDDLE AND LARGEST
230   IF (MIDDLE+1 .EQ. LARGST) GO TO 26
C RESET LOOP LIMITS, PRINT NEW HEADING
      INIT = MIDDLE+1
      LAST = LARGST-1
      IF(DEBUG2)WRITE (KW,902) MIDDLE
235   GO TO 18
C*****************************************************************************
C END OF SPECIAL ALGORITHM SECTION
C*****************************************************************************
C FOR 2X2 OR 3X3 K MATRIX ENTRIES DIRECTLY ASSOCIATED WITH DOF BEING
240   C ROTATED, APPLY FULL SIMILARITY TRANSFORM
   26 DO 28 I = 1,LIMIT
      DO 28 J = 1,I
      K = IKOUNT+IR(I)-1
      KADR = INTGRK(K)+IR(J)-KOFF
245   IF (IR(I) .GE. IR(J)) GO TO 27
      K = IKOUNT+IR(J)-1
      KADR = INTGRK(K)+IR(1)-KOFF
C STORE K MATRIX ENTRY IN A, ZERO B AND SYMMETRIZE
   27 A(I,J) = REALKK(KADR)
250   B(I,J) = 0.
      A(J,I) = A(I,J)
   28 B(J,I) = 0.
C APPLY HALF OF TRANSFORM, (ROTOR)$(A)
      DO 29 I = 1,LIMIT
255   DO 29 J = 1,LIMIT
      DO 29 K = 1,LIMIT
   29 B(I,J) = B(I,J)+ROTOR(I,K)$A(K,J)
C PREPARE STORAGE
      DO 30 I = 1,LIMIT
260   DO 30 J = 1,LIMIT
   30 A(I,J) = 0.
C APPLY 2ND HALF OF TRANSFORM, (B)$(ROTOR-TRANSPOSE)
      DO 31 I = 1,LIMIT
      DO 31 J = 1,LIMIT
265   DO 31 K = 1,LIMIT
   31 A(I,J) = A(I,J)+B(I,K)$ROTOR(J,K)
C RESTORE TRANSFORMED VALUES TO K MATRIX
      DO 32 I = 1,LIMIT
      DO 32 J = 1,I
270   K = IKOUNT+IR(I)-1
      KADR = INTGRK(K)+IR(J)-KOFF
      IF (IR(I) .GE. IR(J)) GO TO 32
      K = IKOUNT+IR(J)-1
      KADR = INTGRK(K)+IR(I)-KOFF
275   32 REALKK(KADR) = A(I,J)
```

C2-72

```
      C APPLY (K)*(ROTOR-TRANSPOSE) TO ROWS BELOW LARGEST IF LNZE COL NOS
      C ARE .LE. LEAST. (IF LNZE COL NO IS .GT. LEAST, THE ROW CONTAINS
      C STORED OR NON-STORED ZEROS IN COLUMNS IROW, JROW, KROW AND CAN
      C BE SKIPPED.) -- SKIP THIS SECTION IF LARGEST DOF = NDT
280         IF (LARGST .EQ. NDT) GO TO 38
      C PRINT NEW HEADING, RESET LOOP LOWER LIMIT AND INFO INDEX
            IF(DEBUG2)WRITE (KW,902) LARGST
            INIT = LARGST+1
            INFOX = 0
285         DO 37 MROW = INIT,NDT
      C CHECK LNZE COL NO
            J = ILNZ+MROW-1
            IF (INTGRK(J) .GT. LEAST) GO TO 37
      C INCREMENT INFO INDEX, STORE ROW NUMBER
290         INFOX = INFOX+1
            INFO(INFOX) = MROW
      C IF INFO VECTOR IS FILLED, PRINT AND RESET INDEX
            IF (INFOX .LT. 10) GO TO 33
            IF(DEBUG2)WRITE (KW,903) (INFO(I), I = 1,10)
295         INFOX = 0
      C GET ENTRIES OF K MATRIX INTO TEMPC, ZERO TEMPR
         33 K = IKOUNT+MROW-1
      C
            IF(.NOT.SPACE.AND.(MROW.LT.RBEGIN.OR.MROW.GT.REND))
300        1 CALL SHIFT(REALKK,INTGRK,MROW,MROW)
      C
            K = INTGRK(K)-KOFF
            DO 34 I = 1,LIMIT
            KADR = K+IR(I)
305         TEMPC(I) = REALKK(KADR)
         34 TEMPR(I) = 0.
      C APPLY TRANSFORM
            DO 35 I = 1,LIMIT
            DO 35 J = 1,LIMIT
310      35 TEMPR(I) = TEMPR(I)+TEMPC(J)*ROTOR(I,J)
      C RESTORE TO K
            DO 36 I = 1,LIMIT
            KADR = K+IR(I)
         36 REALKK(KADR) = TEMPR(I)
315      37 CONTINUE
      C CHECK FOR PARTIALLY FILLED INFO VECTOR
            IF (INFOX .EQ. 0) GO TO 38
            IF(DEBUG2)WRITE (KW,903) (INFO(I), I = 1,INFOX)
      C APPLY TRANSFORM (ROTOR)*(Q) TO ASSEMBLED ELEMENT EQUIVALENT
320   C NODAL FORCES
         38 DO 39 I = 1,LIMIT
            J = IQ+IR(I)-1
            TEMPR(I) = REALK(J)
         39 TEMPC(I) = 0.
325         DO 40 I = 1,LIMIT
            DO 40 J = 1,LIMIT
         40 TEMPC(I) = TEMPC(I)+ROTOR(I,J)*TEMPR(J)
            DO 41 I = 1,LIMIT
            J = IQ+IR(I)-1
330      41 REALK(J) = TEMPC(I)
```

C2-73

```
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
   3 ROTATE

VARIABLES	SN	TYPE		RELOCATION					
1376	A	REAL	ARRAY		1407	B	REAL	ARRAY	
7	DEBUG1	LOGICAL		IO	10	DEBUG2	LOGICAL		IO
11	DISK	REAL		SHIFT	1355	I	INTEGER		
0	ICON	INTEGER		BEGIN	5	IK	INTEGER		BEGIN
1	IKOUNT	INTEGER		BEGIN	2	ILNZ	INTEGER		BEGIN
3	IMASTR	INTEGER		BEGIN	1423	INFO	INTEGER	ARRAY	
1373	INFOX	INTEGER			1365	INIT	INTEGER		
0	INTGRK	INTEGER	ARRAY	F.P.	4	IQ	INTEGER		BEGIN
1420	IR	INTEGER	ARRAY		0	IROW	INTEGER		F.P.
1357	J	INTEGER			0	JROW	INTEGER		F.P.
1372	K	INTEGER			1370	KADR	INTEGER		
2	KBEGIN	INTEGER		SHIFT	1435	KDR	INTEGER	ARRAY	
3	KEND	INTEGER		SHIFT	1	KLCN	INTEGER		SHIFT
5	KMAX	INTEGER		SHIFT	4	KMIN	INTEGER		SHIFT
10	KOFF	INTEGER		SHIFT	2	KP	INTEGER		IO
0	KR	INTEGER		IO	0	KROW	INTEGER		F.P.
3	KT1	INTEGER		IO	4	KT2	INTEGER		IO
5	KT3	INTEGER		IO	0	KUNIT	INTEGER		SHIFT
1	KW	INTEGER		IO	1364	LARGST	INTEGER		
1366	LAST	INTEGER			1363	LEAST	INTEGER		
1356	LIMIT	INTEGER			1367	MCOL	INTEGER		
1371	MIDDLE	INTEGER			1374	MROW	INTEGER		
1	NDT	INTEGER		SIZE	0	NET	INTEGER		SIZE
1354	NNODE	INTEGER			0	NODE	INTEGER		F.P.
6	OUT	LOGICAL		IO	1440	PERM	REAL	ARRAY	
1451	PROTOR	REAL	ARRAY		6	RBEGIN	REAL		SHIFT
0	REALK	REAL	ARRAY	F.P.	0	REALKK	REAL	ARRAY	KK
7	REND	REAL		SHIFT	1462	ROTOR	REAL	ARRAY	
12	SPACE	LOGICAL		SHIFT	1473	TEMPC	REAL	ARRAY	
1501	TEMPCC	REAL	ARRAY		1476	TEMPR	REAL	ARRAY	
1504	TEMPRR	REAL	ARRAY		1362	X	REAL		
0	XANGLE	REAL		F.P.	1361	Y	REAL		
0	YANGLE	REAL		F.P.	1360	Z	REAL		
0	ZANGLE	REAL		F.P.					

EXTERNALS	TYPE	ARGS			
COS	REAL	1	LIBRARY	SHIFT	4
SIN	REAL	1	LIBRARY		

INLINE FUNCTIONS	TYPE	ARGS	
IABS	INTEGER	1	INTRIN

C2-74

STATEMENT LABELS

0	1		72	2		106	3				
0	4		0	5		0	6				
0	7		314	8		0	9				
342	10		0	11		374	12				
0	13		0	14		434	15				
0	16		442	17		461	18				
525	19		535	20		542	21				
0	22		0	23		606	24				
624	25		641	26		660	27				
0	28		0	29		0	30				
0	31		752	32		1002	33				
0	34		0	35		0	36				
1047	37		1061	38		0	39				
0	40		0	41		32	100				
1143	901	FMT	1171	902	FMT	1205	903	FMT			
1210	904	FMT	1230	905	FMT	1236	906	FMT			
1245	907	FMT									

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	
36	1	I	53 55	3B	INSTACK	
56		I	60 60	10B	EXT REFS	
234	4	I	88 91	5B	INSTACK	
254	7	NCOL	98 113	40B	NOT INNER	
261	5	I	100 104	6B	INSTACK	
272	6	I	106 108	6B	NOT INNER	
274	6	J	107 108	3B	INSTACK	
305	7	I	110 113	5B	INSTACK	
330	9	I	123 125	5B	INSTACK	
351	11	I	132 136	6B	NOT INNER	
353	11	J	133 136	3B	INSTACK	
365	12	I	137 142	10B	INSTACK	
402	13	I	143 147	13B	NOT INNER	
404	13	J	145 147	7B	NOT INNER	
407	13	K	146 147	3B	INSTACK	
421	14	I	148 151	13B	NOT INNER	
423	14	J	149 151	6B	NOT INNER	
425	14	K	150 151	3B	INSTACK	
435	16	I	154 156	5B	NOT INNER	
436	16	J	155 156	3B	INSTACK	
511	24	NROW	175 221	103B	EXT REFS	NOT INNER
545	22	I	201 206	6B	INSTACK	
556	23	I	208 211	10B	NOT INNER	
560	23	J	209 211	4B	INSTACK	
644	28	I	241 252	23B	NOT INNER	
652	28	J	242 252	14B	OPT	
673	29	I	254 257	13B	NOT INNER	
675	29	J	255 257	7B	NOT INNER	
700	29	K	256 257	3B	INSTACK	
712	30	I	259 261	4B	NOT INNER	
713	30	J	260 261	2B	INSTACK	
721	31	I	263 266	13B	NOT INNER	
723	31	J	264 266	6B	NOT INNER	
725	31	K	265 266	3B	INSTACK	
737	32	I	268 275	20B	NOT INNER	

C2-75

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
744	32	J	269 275	12B	INSTACK
766	37	NROW	285 315	64B	EXT REFS NOT INNER
1023	34	I	303 306	4B	INSTACK
1032	35	I	308 310	6B	NOT INNER
1034	35	J	309 310	3B	INSTACK
1044	36	I	312 314	3B	INSTACK
1065	39	I	321 324	4B	INSTACK
1074	40	I	325 327	6B	NOT INNER
1076	40	J	326 327	3B	INSTACK
1106	41	I	328 330	3B	INSTACK

COMMON BLOCKS	LENGTH
IO	9
SIZE	2
BEGIN	6
SHIFT	11
KK	2 LCM

STATISTICS

PROGRAM LENGTH	1507B	839
SCM LABELED COMMON LENGTH	34B	28
LCM LABELED COMMON LENGTH	2B	2

130000B SCM USED

C2-76

```
      SUBROUTINE SHIFT(REALK,INTGRK,RMIN,RMAX)
C---PROGRAM ACCESSES VECTOR K IN REALK STORED ON DISK
C
C VARIABLES IN COMMON IN ROUTINES
C
C     RBEGIN - BEGINNING ROW STORED IN K
C     REND   - LAST ROW STORED IN K WITHIN VECTOR REALK
C     KBEGIN - FIRST ADDRESS STORED AT LOCATION IK OF REALK
C     KEND   - LAST ADDRESS IN REALK AT LOCATION IK+(KEND-KBEGIN)
C     KLEN   - LENGTH OF USABLE SPACE IN REALK FOR K MATRIX
C     RMIN   - FIRST ROW NEEDED FOR OPERATION, ADDRESS KMIN
C     RMAX   - LAST ROW NEEDED FOR OPERATION, ENDS AT ADRESS KMAX
C     FILL   -.TRUE.- ADDS ROWS TO END UNTIL KLEN IS FILLED
C            -.FALSE. ONLY ADDS ENOUGH ROWS FOR CASE
C
C  FEABL ROUTINES MUST BE MODIFIED TO SUBRACT KOFF FROM
C     ADDRESS KADR ON NORMAL LOCATION OF K IN REALK
C
C  FOLLOWING VARIABLES MUST BE SET IN MAIN PROGRAM
C     SPACE - .TRUE. - LK-IK SUFFICIENT FOR SOLUTION WITHOUT SHIFT
C             .FALSE.- SHIFT REQUIRED FOR SOLUTION
C           IN EACH CASE LK-IK READ ONTO DISK FOR SOLUTION ROUTINE
C     KLEN  = LK-IK+1
C     KOFF  = MUST BE SET TO ZERO BEFORE EACH SOLUTION
C     REND  = MUST BE SET TO ZERO BEFORE EACH TIME STEP IN MAIN
C     NROW  - NUMBER OF ROWS IN DATA TRANSFER
C     REND  - MUST BE SET TO ZERO BEFORE EACH SOLUTION GENERATION
C
      INTEGER RBEGIN,REND,RMAX,RMIN
      LOGICAL DISK,SPACE,FILL
      DIMENSION REALK(1),INTGRK(1)
      LEVEL 2,REALK,INTGRK
      COMMON/SHIFT/KUNIT,KLEN,KBEGIN,KEND,KMIN,KMAX,RBEGIN,REND,
     1 KOFF,DISK,SPACE,FILL,NROW
      COMMON/BEGIN/ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON/SIZE/ NET,NDT
      DATA KEND/0/,KMIN/0/,KMAX/0/
      DATA RBEGIN/1/,REND/0/,DISK/.FALSE./
C
      IF(RMAX.LT.RMIN)GOTO 9020
      IF(RMIN.GE.RBEGIN.AND.RMAX.LE.REND)RETURN
      IKOUN1=IKOUNT-1
      ILNZ1=ILNZ-1
      IK1=IK-1
      KLEN1=KLEN-1
      NROW1=NROW-1
      LMAX=(((RMAX-1)/NROW)+1)*NROW
      IF(LMAX.GT.NDT)LMAX=NDT
      LMIN=(((RMIN-1)/NROW))*NROW+1
      J=INTGRK(ILNZ1+LMIN)
      KMIN=INTGRK(IKOUN1+LMIN)+J
      KMAX=INTGRK(IKOUN1+LMAX)+LMAX
      J=INTGRK(ILNZ1+RBEGIN)
      KBEGIN=INTGRK(IKOUN1+RBEGIN)+J
      IF(KMAX-KMIN.GT.KLEN1)GOTO 9010
```

C2-77

```
      C---
            IF(DISK)GOTO 800
            JM1N=NDT
            JMAX=0
            IMAX=0
        800 CONTINUE
      C---
            IF(LMAX.LE.REND)GOTO 2000
            IF(LMIN.GE.RBEGIN)GOTO 1000
      C  BOTH OUTSIDE
            GOTO 2000
      C
      C  READ IN DATA AT END OF WORK K UNTIL RMAX INSIDE, SHIFT AT BOTTOM IF
      C     NECESSARY
       1000 CONTINUE
            K=INTGRK(IKOUN1+LMAX)+LMAX
            IF(K-KBEGIN.LE.KLEN1)GOTO 1200
      C
      C     SHIFT LOWER ROWS DOWN IF SPACE INSUFFICIENT
            DISK=.TRUE.
            KSHIFT=(K-KBEGIN)-KLEN1
            K=INTGRK(ILNZ1+RBEGIN)
            K=INTGRK(IKOUN1+RBEGIN)+K
       1100 CONTINUE
            LAST=RBEGIN+NROW1
            IF(LAST.GT.NDT)LAST=NDT
            J=INTGRK(IKOUN1+LAST)+LAST
            J=J-K+1
            IF(LAST.GT.REND)GOTO 1130
            IRECORD=RBEGIN/NROW+1
            CALL WRITMS(KUNIT,REALK(K-KOFF),J,IRECORD,-1,0)
            IMAX=MAX0(IMAX,LAST)
       1130 CONTINUE
            RBEGIN=RBEGIN+NROW
            J=K
            K=INTGRK(ILNZ1+RBEGIN)
            K=INTGRK(IKOUN1+RBEGIN)+K
            II=K-J
            KSHIFT=KSHIFT-II
            IF(KSHIFT.GT.0)GOTO 1100
            IF(RBEGIN.GT.REND)GOTO 1160
            IS=K-KOFF
            IE=KEND-KOFF
            II=K-KBEGIN
            DO 1150 I=IS,IE
            REALK(I-II)=REALK(I)
       1150 CONTINUE
       1160 CONTINUE
            KBEGIN=K
            KOFF=KBEGIN-IK
      C
       1200 CONTINUE
            IF(RBEGIN.GT.REND)REND=RBEGIN-1
            LAST=REND+1
            REND=REND+NROW
```

(2-78

```
                IF(REND.GT.NDT)REND=NDT
                K=INTGRK(IKOUN1+REND)
                KEND=K+REND
              C---
  115           IF(.NOT.DISK)GOTO 1208
                J=INTGRK(ILNZ1+LAST)
                K=INTGRK(IKOUN1+LAST)
                K=K+J
                J=KEND-K+1
  120           K=K-KOFF
                IF(REND.LE.IMAX)GOTO 1205
                IF(JMIN.LE.REND.AND.REND.LE.JMAX)GOTO 1205
                K=K-1
                DO 1204 I=1,J
  125          1204 REALK(I+K)=0.
                GOTO 1208
               1205 CONTINUE
                IRECORD=LAST/NROW+1
                CALL READMS(KUNIT,REALK(K),J,IRECORD)
  130          1208 CONTINUE
              C---
                IF((KEND-KBEGIN).GT.KLEN1)GOTO 9030
                IF(REND.LT.RMAX) GOTO 1200
                IF(REND.GE.NDT.OR..NOT.FILL)GOTO 1210
  135           LAST=REND+NROW
                IF(LAST.GT.NDT)LAST=NDT
                J=INTGRK(IKOUN1+LAST)+LAST
                IF((J-KBEGIN).LE.KLEN1)GOTO 1200
               1210 CONTINUE
  140           RETURN
              C
              C SHIFT TOP ROWS UP AND ADD MORE ROWS AT BOTTOM
               2000 CONTINUE
                J=INTGRK(ILNZ1+LMIN)
  145           K=INTGRK(IKOUN1+LMIN)+J
                IF((KEND-K).LT.KLEN1)GOTO 2200
                DISK=.TRUE.
                KSHIFT=KBEGIN-K
               2100 CONTINUE
  150           LAST=REND-NROW1
                IF(REND.GE.NDT)LAST=((REND-1)/NROW)*NROW+1
                K=INTGRK(IKOUN1+LAST)
                J=INTGRK(ILNZ1+LAST)
                K=K+J
  155           J=KEND-K+1
                IF(REND.LT.RBEGIN)GOTO 2150
                IRECORD=LAST/NROW+1
                CALL WRITMS(KUNIT,REALK(K-KOFF),J,IRECORD,-1,0)
                JMIN=MIN0(LAST,JMIN)
  160           JMAX=MAX0(REND,JMAX)
               2150 CONTINUE
                REND=LAST-1
                KSHIFT=KSHIFT-J
                K=INTGRK(IKOUN1+REND)
  165           KEND=K+REND
```

C2-79

```
                IF(REND-NROW.GE.LMAX)GOTO 2100
                IF(KSHIFT.GT.0)GOTO 2100
             2200 CONTINUE
                J=INTGRK(ILNZ1+LMIN)
170             K=INTGRK(IKOUN1+LMIN)+J
                J=KBEGIN-K
                IS=KBEGIN-KOFF
                IE=KEND-KOFF
                KBEGIN=K
175             KOFF=KBEGIN-IK
                IF(REND.LT.RBEGIN)GOTO 2221
                K=IE+IS
                DO 2220 I=IS,IE
                II=K-I
180          2220 REALK(II+J)=REALK(II)
             2221 CONTINUE
             C---
                IF(REND.LT.RBEGIN)RBEGIN=REND+1
                RBEGIN=RBEGIN-1
185             DO 2230 I=LMIN,RBEGIN,NROW
                J=INTGRK(ILNZ1+I)
                K=INTGRK(IKOUN1+I)
                K=K+J
                LAST=I+NROW1
190             IF(LAST.GT.NDT)LAST=NDT
                J=INTGRK(IKOUN1+LAST)+LAST
                J=J-K+1
                K=K-KOFF
                IF(I.LE.IMAX)GOTO 2225
195             IF(JMIN.LE.I.AND.I.LE.JMAX)GOTO 2225
                K=K-1
                DO 2223 II=1,J
             2223 REALK(II+K)=0.
                GOTO 2230
200          2225 CONTINUE
                IRECORD=I/NROW+1
                CALL READMS(KUNIT,REALK(K),J,IRECORD)
             2230 CONTINUE
             2240 CONTINUE
205             RBEGIN=LMIN
             C---
                IF((KEND-KBEGIN).GT.KLEN1)GOTO 9040
             C BOTH OUTSIDE
                IF(RMAX.GT.REND)GOTO 1000
210             RETURN
             C
                ENTRY WSHIFT
             C WRITES EVERYTHING TO DISK BUT LEAVES RMIN AND RMAX THE SAME
                DO 3000 I=RBEGIN,REND,NROW
215             J=INTGRK(ILNZ1+I)
                K=INTGRK(IKOUN1+I)+J
                LAST=I+NROW1
                IF(LAST.GT.NDT)LAST=NDT
                J=INTGRK(IKOUN1+LAST)+LAST
220             J=J-K+1
```

(2-30)

```
                    IRECORD=I/NROW+1
                    CALL WRITMS(KUNIT,REALK(K-KOFF),J,IRECORD,-1,0)
                3000 CONTINUE
                    DISK=.TRUE.
   225              JMIN=MIN0(RBEGIN,JMIN)
                    JMAX=MAX0(REND,JMAX)
                    RETURN
                C----
                9010 IER=9010
   230              WRITE(6,*)IER,KMAX,KMIN,KLEN1,RMAX,RMIN
                    GOTO 9500
                9020 IER=9020
                    WRITE(6,*)IER,RMAX,RMIN
                    GOTO 9500
   235          9030 IER=9030
                    WRITE(6,*)IER,KEND,KBEGIN,KLEN1
                    GOTO 9500
                9040 IER=9040
                    WRITE(6,*)IER,KEND,KBEGIN,KLEN1
   240          9500 CALL XEQCCS(*/DUMP,0./GRUMP,SBX,NUV=10.//*)
                    STOP
                    END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	SHIFT	444	WSHIFT

VARIABLES SN TYPE RELOCATION

11	DISK	LOGICAL		SHIFT	13	FILL	LOGICAL		SHIFT
633	I	INTEGER			0	ICON	INTEGER		BEGIN
632	IE	INTEGER			634	IER	INTEGER		
630	II	INTEGER			5	IK	INTEGER		BEGIN
1	IKOUNT	INTEGER		BEGIN	611	IKOUN1	INTEGER		
613	IK1	INTEGER			2	ILNZ	INTEGER		BEGIN
612	ILNZ1	INTEGER			3	IMASTR	INTEGER		BEGIN
623	IMAX	INTEGER			0	INTGRK	INTEGER	ARRAY	F.P.
4	IQ	INTEGER		BEGIN	627	IRECORD	INTEGER		
631	IS	INTEGER			620	J	INTEGER		
622	JMAX	INTEGER			621	JMIN	INTEGER		
624	K	INTEGER			2	KBEGIN	INTEGER		SHIFT
3	KEND	INTEGER		SHIFT	1	KLEN	INTEGER		SHIFT
614	KLEN1	INTEGER			5	KMAX	INTEGER		SHIFT
4	KMIN	INTEGER		SHIFT	10	KOFF	INTEGER		SHIFT
625	KSHIFT	INTEGER			0	KUNIT	INTEGER		SHIFT
626	LAST	INTEGER			616	LMAX	INTEGER		
617	LMIN	INTEGER			1	NDT	INTEGER		SIZE
0	NET	INTEGER		SIZE	14	NROW	INTEGER		SHIFT
615	NROW1	INTEGER			6	RBEGIN	INTEGER		SHIFT
0	REALK	REAL	ARRAY	F.P.	7	REND	INTEGER		SHIFT
0	RMAX	INTEGER		F.P.	0	RMIN	INTEGER		F.P.

C2-81

VARIABLES SN TYPE RELOCATION
12 SPACE LOGICAL SHIFT

FILE NAMES MODE
TAPE6 FREE

EXTERNALS	TYPE	ARGS			
READMS		4	WRITMS		6
XEQCCS		1			

INLINE FUNCTIONS	TYPE	ARGS				
MAX0	INTEGER	0	INTRIN	MIN0	INTEGER	0 INTRIN

STATEMENT LABELS

60	800	65	1000	101	1100		
130	1130	0	1150	155	1160		
160	1200	0	1204	217	1205		
227	1208	250	1210	251	2000		
264	2100	322	2150	334	2200		
0	2220	357	2221	0	2223		
421	2225	431	2230	0	2240	INACTIVE	
0	3000	515	9010	521	9020		
525	9030	531	9040	534	9500		

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES		
152	1150	I	100 102	3B	INSTACK		
214	1204	I	124 125	2B	INSTACK		
354	2220	I	178 180	3B	INSTACK		
366	2230	I	185 203	46B	EXT REFS	NOT INNER	
416	2223	II	197 198	2B	INSTACK		
455	3000	I	214 223	31B	EXT REFS		

COMMON BLOCKS	LENGTH
SHIFT	13
BEGIN	6
SIZE	2

STATISTICS

PROGRAM LENGTH	643B	419
SCM LABELED COMMON LENGTH	25B	21

130000B SCM USED

C 2-82

```
      SUBROUTINE SOLVE2(I,FBCON,IFBCON,REALK,INTGRK,DIRV,STRESS,DUM,
     1 IDUM),RETURNS(R1,R2)
C----READS WORK FILE, CALL SIMULQ AND BOUNDARY CONDITION FOR INVERSION,
C    PERFORMS ROTATION BACK, AND WRITES ON SOLUTION FILE
C
      DIMENSION REALK(1),INTGRK(1),IFBCON(4,1),FBCON(4,1),DUM(3),IDUM(3)
      LEVEL 2,REALK,INTGRK
      LEVEL 2,REALKK
      LOGICAL STRESS,DISP
      INTEGER ENTRY,RBEGIN,REND
      COMMON/IO/KR,KW,KP,KT1,KT2,KT3,OUT
      COMMON/BEGIN/ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
      COMMON/END/LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
      COMMON/SIZE/NET,NDT
      COMMON/PROB/MASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,NLAP,
     1 NDIM,DISP,NECON,NICON
      COMMON/ROT/IROW,JROW,KROW,ZANGLE,YANGLE,XANGLE
      COMMON/KEY/LEN,KEYTMP,KEYRL1,KEYLLQ,KEYPRB,KEYTL,KEYEL,KEYROT,
     1 KEYECO,KEYICO
      COMMON /SHIFT/ KUNIT,KLEN,KBEGIN,KEND,KMIN,KMAX,RBEGIN,REND,
     1 KOFF,DISK,SPACE,FILL,NUMROW,ENTRY
      COMMON/KK  /REALKK(2)
C
      CALL BREAD2(KEYLLQ,LEN,REALK(IQ),KT1),RETURNS(9000)
      IF(STRESS)CALL BCONST(IFBCON,DUM,REALK,INTGRK,
     1 IFBCON(1,I),IFBCON(2,I),DIRV,FBCON(3,I)),RETURNS(1650)
      IF(.NOT.DISP)CALL BCONAP(IFBCON,DUM,REALK,INTGRK,
     1 IFBCON(1,I),IFBCON(2,I),DIRV,FBCON(3,I)),RETURNS(1650)
C--------
 1650 CONTINUE
      IF(ENTRY.GT.0)GOTO 1652
      CALL SIMULQ(ENERGY,REALK,INTGRK)
      GOTO 1653
 1652 CALL SIMULQ2(ENERGY,REALK,INTGRK)
 1653 CONTINUE
C--------
      DUM(1)=ENERGY
      IF(.NOT.DISP)GOTO 1655
      IDUM(2)=0
      IDUM(3)=2
      GOTO 1656
C----IROW SET IN STAGE2 TO FAULT DOF SHIFT
 1655 IDUM(2)=IFBCON(2,I)
      IF(IFBCON(4,I).EQ.4.AND.DIRV.NE.1.0)IDUM(2)=IDUM(2)+IROW
      IDUM(3)=IFBCON(4,I)
 1656 CONTINUE
      CALL BLD(3,DUM,KT3),RETURNS(9000)
C----DISPLACEMENTS AT INTERELEMENT BOUNDARIES
      IF(NICON.LE.0)GOTO 1666
      DO 1665 II=1,NICON
      IF(IFBCON(4,II).GE.10)GOTO 1665
      INDEX=IFBCON(2,II)+IQ-1
      REALK(INDEX)=REALK(INDEX)+REALK(IQ-1+IFBCON(1,II))
 1665 CONTINUE
 1666 CONTINUE
```

C2-83

```
                C----ROTATE BACK
                     CALL BREAD(KEYROT,LEN,IDUM,KT1),RETURNS(9000)
                     NROT=IDUM(1)
                     IDROT=IDUM(2)
     60              IF(NROT.EQ.0)GOTO 1675
                     DO 1670 N=1,NROT
                     KEYTMP=0
                     CALL BREAD(KEYTMP,LEN,IROW,KT1),RETURNS(9000)
                     IF(YANGLE.NE.0..OR.XANGLE.NE.0.)GOTO 1668
     65              CALL ROTATE(-N,IROW,JROW,KROW,-ZANGLE,YANGLE,XANGLE,REALK,INTGRK)
                     GOTO 1670
                 1668 IF(XANGLE.NE.0.)GOTO 1669
                     CALL ROTATE(-N,IROW,JROW,KROW,-YANGLE,-ZANGLE,XANGLE,REALK,INTGRK)
                     GOTO 1670
     70          1669 CALL ROTATE(-N,IROW,JROW,KROW,-XANGLE,-YANGLE,-ZANGLE,REALK,
                    1 INTGRK)
                 1670 CONTINUE
                 1675 CONTINUE
                     LEN=LQ-IQ+1
     75              CALL BLD2(LEN,REALK(IQ),KT3),RETURNS(9000)
                     RETURN R1
                 9000 RETURN R2
                     END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	SOLVE2

VARIABLES

	SN	TYPE		RELOCATION		SN	TYPE		RELOCATION
0	DIRV	REAL		F.P.	11	DISK	REAL		SHIFT
12	DISP	LOGICAL		PROB	0	DUM	REAL	ARRAY	F.P.
342	ENERGY	REAL			15	ENTRY	INTEGER		SHIFT
0	FBCON	REAL	ARRAY	F.P.	13	FILL	REAL		SHIFT
6	GMALT	REAL		PROB	3	GRAV	REAL		PROB
5	GRAVB	REAL		PROB	4	GRAVS	REAL		PROB
0	HASH1	REAL		PROB	0	I	INTEGER		F.P.
0	ICON	INTEGER		BEGIN	346	IDROT	INTEGER		
0	IDUM	INTEGER	ARRAY	F.P.	0	IFBCON	INTEGER	ARRAY	F.P.
343	II	INTEGER			5	IK	INTEGER		BEGIN
1	IKOUNT	INTEGER		BEGIN	2	ILNZ	INTEGER		BEGIN
3	IMASTR	INTEGER		BEGIN	344	INDEX	INTEGER		
0	INTGRK	INTEGER	ARRAY	F.P.	4	IQ	INTEGER		BEGIN
0	IROW	INTEGER		ROT	1	JROW	INTEGER		ROT
2	KBEGIN	INTEGER		SHIFT	3	KEND	INTEGER		SHIFT
10	KEYECO	INTEGER		KEY	6	KEYEL	INTEGER		KEY
11	KEYICO	INTEGER		KEY	3	KEYLLQ	INTEGER		KEY
4	KEYPRB	INTEGER		KEY	2	KEYRL1	INTEGER		KEY
7	KEYROT	INTEGER		KEY	5	KEYTL	INTEGER		KEY
1	KEYTMP	INTEGER		KEY	1	KLEN	INTEGER		SHIFT
5	KMAX	INTEGER		SHIFT	4	KMIN	INTEGER		SHIFT

C2-84

VARIABLES	SN	TYPE		RELOCATION					
	10	KOFF	INTEGER		SHIFT	2	KP	INTEGER	IO
	0	KR	INTEGER		IO	2	KROW	INTEGER	ROT
	3	KT1	INTEGER		IO	4	KT2	INTEGER	IO
	5	KT3	INTEGER		IO	0	KUNIT	INTEGER	SHIFT
	1	KW	INTEGER		IO	0	LCON	INTEGER	END
	0	LEN	INTEGER		KEY	2	LENGTH	INTEGER	PROB
	5	LK	INTEGER		END	1	LKOUNT	INTEGER	END
	2	LLNZ	INTEGER		END	3	LMASTR	INTEGER	END
	4	LQ	INTEGER		END	347	N	INTEGER	
	11	NDIM	INTEGER		PROB	1	NDT	INTEGER	SIZE
	13	NECON	INTEGER		PROB	0	NET	INTEGER	SIZE
	7	NFLT	INTEGER		PROB	14	NICON	INTEGER	PROB
	10	NLAP	INTEGER		PROB	345	NROT	INTEGER	
	14	NUMROW	INTEGER		SHIFT	1	OLDWRK	REAL	PROB
	6	OUT	REAL		IO	6	RBEGIN	INTEGER	SHIFT
	0	REALK	REAL	ARRAY	F.P.	0	REALKK	REAL	ARRAY KK
	7	REND	INTEGER		SHIFT	0	R1	RETURNS	
	0	R2	RETURNS			12	SPACE	REAL	SHIFT
	0	STRESS	LOGICAL		F.P.	5	XANGLE	REAL	ROT
	4	YANGLE	REAL		ROT	3	ZANGLE	REAL	ROT

EXTERNALS	TYPE	ARGS			
BCONAP		8	BCONST	8	
BLD		3	BLD2	3	
BREAD		4	BREAD2	4	
ROTATE		9	SIMULQ	3	
SIMULQ2		3			

STATEMENT LABELS

47	1650	56	1652	62	1653		
70	1655	107	1656	126	1665		
127	1666	154	1668	166	1669		
177	1670	202	1675	213	9000		

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES		
120	1665	II	50 54	7B	INSTACK		
135	1670	N	61 72	45B		EXT REFS	EXITS

COMMON BLOCKS	LENGTH
IO	7
BEGIN	6
END	6
SIZE	2
PROB	13
ROT	6
KEY	10
SHIFT	14
KK	2 LCM

STATISTICS

PROGRAM LENGTH	351B	233
SCM LABELED COMMON LENGTH	100B	64
LCM LABELED COMMON LENGTH	2B	2

130000B SCM USED

APPENDIX C.3

c.3-a

```
      C  DSTAGE3? TIME INVERSION OF DISPLACEMENTS
           PROGRAM DSTAGE3(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE12=100,
          1 TAPE14=100,TAPE20,TAPE11=100)
      C
      C------INPUT
      C UNIT 5
      C     NAMELIST/IOK/ - ANY CHANGES IN INPUT/OUTPUT UNITS
      C
      C UNIT 12
      C     NAMELIST/IN/ - ANY CHANGES
      C     LT     - ORDER OF APPROXIMATION (LT.LESS THAN LAPN)
      C     RLAX(I) - LT RELAXATION TIMES FOR SERIES
      C     LAPN  - NUMBER OF TIMES FOR APPROX. TO SERIES
      C             (IF *ALL*, USES ALL TIMES IN SOLUTION FILE)
      C     TLAP(I)- LAPN TIMES FOR SOLUTION - WILL ABORT IF NOT ALL
      C             IN SOLUTION FILE (IF *ALL* SPECIFIED FOR LAPN, SKIP)
      C     DELT,TMAX,SCALET,SCALEY - FOR TIMES AND PLOTTING/LIST
      C        DELT   TIME INTERVAL FOR COMPUTING SERIES (IF 0.0 VALUE
      C             IS SET IN PLOTD - LIST IS THEN AT INTERVALS SCALET)
      C        TMAX - MAX. TIME FROM ZERO FOR PLOT OR LIST
      C        SCALET - TIME SCALE IN UNITS/INCH
      C        SCALEY - FUNCTION SCALE IN UNITS/INCH (IF PLOT DOES NOT
      C                 FIT, RESCALE)
      C     NDOF  - NUMBER OF DEGREES OF FREEDOM FOR INVERSION (IF *ALL*,
      C           COMPUTES FOR ALL)
      C     IDOF(I) - NDOF DEGREES OF FREEDOM NUMBERS IN ASCENDING ORDER
      C              TO BE INVERTED (SKIP IF *ALL* SPECIFIED)
      C
           DIMENSION IUP(20,1),UP(20,1)
           EQUIVALENCE (UT,UP,IUP)
           DIMENSION TITLE(20),TLAP(20),TLAPO(20),IDUM(10)
           EQUIVALENCE (DUM,IDUM)
           DIMENSION IS(2,20),KEY(20)
           DIMENSION TYPE(2),XTITLE(10),YTITLE(10)
           INTEGER ALL
           LOGICAL STRESS,DISP,FLAG,FAULT,OLDFIL,PLOT,LIST
           LOGICAL FLOW,ERROR,JUMP,CREEP,AGAIN,OUT
      C
           LEVEL 2,INDEX2,INDEX4,INDEX1
           LEVEL 2,IFBCON,IDOF,IELEM
           LEVEL 2,DUM,IDUM
           LEVEL 2,UT,UP,IUP
           LEVEL 2,INTGRK
      C
      C
           COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,NLAP,
          1 NDIM,DISP
           COMMON/PLOTS/LT,DELT,PLOT,TMAX,SCALET,SCALEY,NPLT,ERROR
           COMMON/SIZE/NET,NDT,NETNEW,NDTNEW,LNODNW
           COMMON/IO/KR,KW,KP,KT1,KT2,KT3,KT4,KT5
           COMMON/BEGIN/ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
           COMMON/END  /LCON,LKOUNT,LLNZ,LMASTR,LQ,LK
           COMMON/RTIME/FLOW,RLAX(20)
      C
           COMMON/INDEX/INDEX2(1000),INDEX4(1000),INDEX1(1000)
```

C.3-2

```
      COMMON/BLDIO/XIOWR(3256)
      COMMON/BLD/KPTR,LREC,IREC
      COMMON/BREAD1O/XIOBR(3256)
      COMMON/BREAD/NFX,LRECBR,KEYX(20),LENX(20),K1X(20)
60    COMMON/DOF/IFBCON(1000),IDOF(2000),IELEM(500)
      COMMON/DUM/DUM(10000)
      COMMON/TIME/UT(20,1000)
      COMMON/K   /INTGRK(40000)
C
65    C
      DATA KPTR/1/,LREC/2048/,IREC/1/
      DATA LRECBR/2048/
      DATA NFX/0/,KEYX/20*0/,LENX/20*0/,K1X/20*0/
C
70    DATA ALL/3HALL/,IMAX/1000/,ITMAX/20/,CRLAX/1.E07/
      DATA STRESS/.FALSE./,FLAG/.FALSE./,FAULT/.FALSE./,OLDFIL/.FALSE./
      DATA LIST/.FALSE./,JUMP/.FALSE./,CREEP/.FALSE./,OUT/.TRUE./
      DATA ERROR/.FALSE./,DISP/.TRUE./,PLOT/.FALSE./,FLOW/.FALSE./
      DATA TYPE/#    #,#DOF #/
75    DATA XTITLE/# TIM#,#E - #,#CGS #,#UNIT#,#S   #/
      DATA YTITLE/# DIS#,#PLAC#,#EMEN#,#TS  #,# CGS#,# UNI#,#TS  #/
C
      NAMELIST/IN/STRESS,DISP,FAULT,PLOT,OLDFIL,NET,NDT,LIST,FLOW,NPLT,
     1 ERROR,CREEP,CRLAX
80    NAMELIST/IOK/KR,KW,KP,KT1,KT2,KT3,KT4,KT5,OUT
C
C*********************************************************************
C
C  SET UP
85    C
      DISP=.TRUE.
      FLOW=.TRUE.
      ERROR=.FALSE.
      PLOT=.FALSE.
90    NPLT=1
C-----DATA FILE
      KR=5
      KW=6
      KP=7
95    C------SOLUTION FILE FROM STAGE2
      KT1=11
      KT2=12
C-------TIME FILE
      KT3=13
100   KT4=14
      KT5=15
      READ(5,IOK)
      WRITE(6,IOK)
      CALL OPENMS(KT1,INDEX1,1000,0)
105   CALL OPENMS(KT2,INDEX2,1000,0)
      CALL OPENMS(KT4,INDEX4,1000,0)
C
C  INPUT FROM SOLUTION FILE
C
110   KEYTMP=0
```

C3-3

```
            CALL BREAD(KEYTMP,LEN,TITLE,KT2),RETURNS(9000)
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,HASH1,KT2),RETURNS(9000)
            KEYTMP=0
115         CALL BREAD(KEYTMP,LEN,NET,KT2),RETURNS(9000)
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,TLAPO,KT2),RETURNS(9000)
            LAPO=LEN
            READ(KR,IN)
120      C
         C  INPUT ORDER OF APPROXIMATION AND RLAX TIMES
         C
            WRITE(KW,10)
           10 FORMAT(10X,*INPUT DATA*)
125         WRITE(KW,IN)
            READ(KR,*)LT
            READ(KR,*)(RLAX(I),I=1,LT)
         C
         C  INPUT TIMES TO BE USED
130      C
            READ(KR,*)LAPN
            IF(LAPN.EQ.ALL)GOTO 1100
            READ(KR,*)(TLAP(I),I=1,LAPN)
         C  CHECK FOR MISSING TIME
135         J=0
            DO 1020 I=1,LAPN
         1005 J=J+1
            IF(J.GT.LAPO)GOTO 9002
            IF(TLAP(I).EQ.TLAPO(J))GOTO 1020
140         GOTO 1005
         1020 CONTINUE
            LAP=LAPN
            GOTO 1150
         1100 CONTINUE
145         DO 1120 I=1,LAPO
         1120 TLAP(I)=TLAPO(I)
            LAP=LAPO
         1150 CONTINUE
            WRITE(KW,*)LAP
150         WRITE(KW,22)(TLAP(I),I=1,LAP)
           22 FORMAT(1X,1P10E11.3)
            IF(LAP.GT.ITMAX)GOTO 1170
            GOTO 1200
         1170 WRITE(6,25)LAP
155        25 FORMAT(* ------ABORT--------LAP=*,I4)
            GOTO 9500
         C
         1200 CONTINUE
         C
160      C INPUT WORK FILE FOR DOF------ -------
         C
            KEYTMP=0
            CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
            KEYTMP=0
165         CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
```

C3-4

```
      KEYTMP=0
      CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
      KEYTMP=0
      CALL BREAD(KEYTMP,LEN,ICON,KT1),RETURNS(9000)
      KEYTMP=0
      CALL BREAD(KEYTMP,LEN,LCON,KT1),RETURNS(9000)
      KEYTMP=0
      CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
      KEYTMP=0
      CALL BREAD2(KEYTMP,LEN,INTGRK,KT1),RETURNS(9000)
      KEYRL1=KEYTMP
      IF(LEN.NE.LMASTR)WRITE(KW,*)LEN,LMASTR
      KEYTMP=0
      CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
C-----------------------------------
C
C INPUT PLOT TIMES, SCALES
C
      READ(KR,*)DELT,TMAX,SCALET,SCALEY
      WRITE(KW,22)DELT,TMAX,SCALET,SCALEY
 1300 CONTINUE
C
C  DISPLACEMENT DOF TIME INVERSION----------------------------------
C  INPUT ELEMENT NUMBERS AND FIND DOF NUMBERS
C
      IF(.NOT.DISP) GOTO 1170
C
C  INPUT ELEMENT NUMBERS
C
      READ(KR,*)NLINES
      IF(NELEM.EQ.ALL)GOTO 1330
C
      NELEM=0
      DO 1310 I=1,NLINES
      READ(KR,*)MIELM,MAELM
      DO 1305 J=MIELM,MAELM
      NELEM=NELEM+1
      IELEM(NELEM)=J
 1305 CONTINUE
 1310 CONTINUE
C
      GOTO 1340
 1330 NELEM=NET
      DO 1335 I=1,NELEM
 1335 IELEM(I)=I
 1340 CONTINUE
   27 FORMAT(1X,20I6)
      CALL IORDEL3(NELEM,IELEM,DUM,0,NELEM,DUM)
      WRITE(KW,*)NELEM
      WRITE(KW,27)(IELEM(I),I=1,NELEM)
C
C FIND DOF FOR ELEMENTS
C-----DETERMINE NODE NUMBERS USED IN FILE AND ORDER THESE TO
C     SAVE SPACE
C
```

C3-5

```
      JJ=0
      IMASTR1=IMASTR-1
      DO 1347 I=1,NELEM
      LNUM=IELEM(I)
225   IADDR=INTGRK(IMASTR1+LNUM)
      IF(LNUM.EQ.NLT)GOTO 1342
      IADDR2=INTGRK(IMASTR+LNUM)-1
      GOTO 1344
C
230    1342 CONTINUE
      DO 1343 J=IADDR,LMASTR
      IF(INTGRK(J).NE.0)GOTO 1343
      IADDR2=J-1
      GOTO 1344
235   C
       1343 CONTINUE
      C
       1344 CONTINUE
      IADDR=IADDR+NDIM-1
240   DO 1346 J=IADDR,IADDR2,NDIM
      JJ=JJ+1
      IDUM(JJ)=INTGRK(J)/NDIM
       1346 CONTINUE
       1347 CONTINUE
245   NOD=JJ
      CALL IORDEL3(NOD,IDUM,DUM,0,NOD,DUM)
      IF(OUT)WRITE(KW,36)(IDUM(I),I=1,NOD)
      C
      C  CONVERT NODE NUMBERS TO DOF
250   C
      JJ=0
      DO 1360 I=1,NOD
      II=IDUM(I)*NDIM-NDIM
      C
255   DO 1358 J=1,NDIM
      JJ=JJ+1
      II=II+1
      IDOF(JJ)=II
       1358 CONTINUE
260   IF(JJ.GT.2000)GOTO 9500
       1360 CONTINUE
      NDOF=JJ
      C
      C
265   C BLD TIML FILE
      C
      CALL BLD(20,TITLE,KT4),RETURNS(9000)
      CALL BLD(11,HASH1,KT4),RETURNS(9000)
      CALL BLD(5,NET,KT4),RETURNS(9000)
270   CALL BLD(LAP,TLAP,KT4),RETURNS(9000)
      LEN=LT+1
      CALL BLD(LEN,FLOW,KT4),RETURNS(9000)
      CALL BLD2(NDOF,IDOF,KT4),RETURNS(9000)
      C
275   C  OUTPUT HEADINGS
```

C3-6

```
      C
            WRITE(KW,30)(TITLE(I),I=1,20),HASH1
         30 FORMAT(1H1,20A4,//,* INVERSION OF LAPLACE TRANSFORM SPACE*,/,10X,
           1*MODEL HASH CODE=*,Z9,///,* LAPLACE REDUCED TIMES FOR INVERSION*)
280         WRITE(KW,32)(TLAP(I),I=1,LAP)
         32 FORMAT(1X,1PE11.3)
            WRITE(KW,33)(RLAX(I),I=1,LT)
         33 FORMAT(* RELAXATION TIMES*,/,(1X,1P10E10.2))
            WRITE(KW,34)NDOF,FAULT
285      34 FORMAT(1X,///,* SOLUTION FOR*,I6,* DOF WITH FAULT=*,L2)
            WRITE(KW,36)(IDOF(I),I=1,NDOF)
         36 FORMAT(1X,20I5)
      C
      C  SCAN SOLUTION FILE FOR SUCCESSIVE SOLUTIONS
290   C
            NFAULT=0
            AGAIN=.FALSE.
       1380 CONTINUE
            NFAULT=NFAULT+1
295         DO 1390 I=1,20
            IS(1,I)=1
            IS(2,I)=1
       1390 KEY(I)=0
            I=0
300         J=0
            IC=0
       1400 IE=IC
            IC=0
            I=I+1
305         IF(I.GT.LAP)GOTO 1620
       1410 J=J+1
            JUMP=.FALSE.
            IF(J.GT.LAPO)GOTO 9002
            DO 1420 N=1,NET
310         KEYTMP=0
            IF(AGAIN.AND.N.EQ.1.AND.J.EQ.1)KEYTMP=KEYST
            CALL BREAD2(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
            IF(CREEP.AND.N.EQ.1.AND.J.EQ.1)KEYST=KEYTMP
       1420 CONTINUE
315    1430 CONTINUE
            KEYTMP=0
            CALL BREAD2(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
            IF(LEN.EQ.1)GOTO 1435
            IF(JUMP)GOTO 1433
320         IF(TLAP(I).NE.TLAPO(J))GOTO 1433
            FT=1.
            IF(AGAIN)FT=FTIME(TLAP(I),CRLAX,IFT)
            IC=IC+1
            IF(I.EQ.1)IUP(1,IC)=0
325         UP(I+1,IC)=DUM(1)*FT
            IF(IDUM(2).NE.0)FAULT=.TRUE.
            IF(I.EQ.1.AND.FAULT)IFBCON(NFAULT)=IDUM(2)
            IF(I.EQ.1.AND.FAULT.AND.AGAIN)IFBCON(NFAULT)=IFBCON(NFAULT)+IFT
            IF(I.EQ.1.AND.FAULT)NFAULT=NFAULT+1
330         IF(LEN.NE.3)GOTO 1170
```

C3-7

```
      1433 KEYTMP=0
           CALL BREAD2(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
      1435 IF(TLAP(I).EQ.TLAPO(J))GOTO 1440
           IF(LEN.EQ.1)GOTO 1410
335        GOTO 1430
      1440 CONTINUE
           IF(LEN.EQ.1)GOTO 1500
           IF(JUMP)GOTO 1430
      1450 CONTINUE
340        DO 1460 II=1,NDOF
      1455 IC=IC+1
           IF(IC.GT.IMAX-3)GOTO 1600
           IF(I.EQ.1)IUP(1,IC)=IDOF(II)
           UP(I+1,IC)=DUM(IDOF(II))*FT
345   1460 CONTINUE
        40 FORMAT(1X,10I11)
        41 FORMAT(1X,1P10E11.3)
           GOTO 1430
      1500 CONTINUE
350        IF(I.EQ.1.AND.OUT)WRITE(KW,40)(IUP(1,IP),IP=1,IC)
           IF(OUT)WRITE(KW,41)(UP(I+1,IP),IP=1,IC)
           GOTO 1400
      1600 CONTINUE
           FLAG=.TRUE.
355        JUMP=.TRUE.
           IF(I.EQ.1.AND.FAULT)NFAULT=NFAULT-1
           IC=IC-1
           IS(1,I)=II-1
           IS(2,I)=J
360        KEY(I)=KEYTMP
           GOTO 1430
      1620 IC=IE
           IF(ERROR)REWIND 20
           DO 1700 JJ=1,IC
365        CALL LAPTIM2(UP(2,JJ),TLAP,LAP,UT(3,JJ),UT(2,JJ),LT,DUM,ERROR)
      1700 CONTINUE
           LEN=LT+3
           IF(FLOW)LEN=LT+4
           DO 1800 JJ=1,IC
370        CALL BLD2(LEN,UT(1,JJ),KT4),RETURNS(9000)
     C
           IF(LIST)WRITE(KW,43)(UT(II,JJ),II=1,LEN)
      1800 CONTINUE
        43 FORMAT(1X,I10,1P10E11.3)
375  C     IF(PLOT.OR.LIST)GOTO 1900
           IF(FLAG)GOTO 1810
           GOTO 2000
      1810 FLAG=.FALSE.
           I=0
380   1815 IE=IC
           IC=0
           I=I+1
           JUMP=.FALSE.
           IF(I.GT.LAP)GOTO 1620
385        J=IS(2,I)
```

C3-8

```
                II=IS(1,I)
                KEYTMP=KEY(I)
                GOTO 1825
           1818 KEYTMP=0
390             CALL BREAD2(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
                IF(LEN.EQ.1)GOTO 1827
                IF(JUMP)GOTO 1820
                IF(TLAP(I).NE.TLAPO(J))GOTO 1820
                FT=1.
395             IF(AGAIN)FT=FTIME(TLAP(1),CRLAX,IFT)
                IF(I.EQ.1.AND.FAULT)NFAULT=NFAULT+1
                IC=IC+1
                IF(I.EQ.1)IUP(1,IC)=0
                UP(I+1,IC)=DUM(1)*FT
400             IF(I.EQ.1.AND.FAULT)IFBCON(NFAULT)=IDUM(2)
                IF(I.EQ.1.AND.FAULT.AND.AGAIN)IFBCON(NFAULT)=IFBCON(NFAULT)+IFT
                IF(LEN.NE.3)GOTO 1170
           1820 KEYTMP=0
                II=0
405        1825 CALL BREAD2(KEYTMP,LEN,DUM,KT2),RETURNS(9000)
           1827 IF(LEN.EQ.1)GOTO 1850
                IF(JUMP)GOTO 1818
                FT=1.
                IF(AGAIN)FT=FTIME(TLAP(J),CRLAX,IFT)
410        1830 II=II+1
                IF(II.GT.NDOF)GOTO 1818
                IC=IC+1
                IF(IC.GT.IMAX-3)GOTO 1840
                IF(I.EQ.1)IUP(1,IC)=IDOF(II)
415             UP(I+1,IC)=DUM(IDOF(I1))*FT
                GOTO 1830
           1840 FLAG=.TRUE.
                JUMP=.TRUE.
                IC=IC-1
420             IS(2,I)=J
                IS(1,I)=II-1
                KEY(I)=KEYTMP
                GOTO 1818
           1850 CONTINUE
425             IF(I.EQ.1.AND.OUT)WRITE(KW,40)(IUP(1,IP),IP=1,IC)
                IF(OUT)WRITE(KW,41)(UP(I+1,IP),IP=1,IC)
                GOTO 1815
           C
           C1900 CONTINUE
430        C     IF(ERROR)REWIND 20
           C     DO 1950 JJ=1,IC,NPLT
           C     CALL PLOTD2(UT(1,JJ),TYPE,TITLE,XTITLE,YTITLE,ITMAX,DUM,KW)
           C1950 CONTINUE
           C     IF(FLAG)GOTO 1810
435        C     GOTO 2000
           C
            2000 CONTINUE
                IF(AGAIN)GOTO 2010
                IF(CREEP)AGAIN=.TRUE.
440             IF(AGAIN)GOTO 1380
```

C3-9

```
             2010 CONTINUE
                  CALL BLD(1,NDOF,KT4),RETURNS(9000)
                  IF(NFAULT.LE.0)NFAULT=1
                  CALL BLD2(NFAULT,IFBCON,KT4),RETURNS(9000)
  445             CALL FRC(0,0,KT4),RETURNS(9000)
                  WRITE(6,50)
               50 FORMAT(1X,///,* --------DSTAGE3 COMPLETED-------*)
                  STOP
             9000 WRITE(6,9001)KEYTMP,LEN
  450        9001 FORMAT(* -------ERROR IN BREAD/BLD-------*,2I8)
                  GOTO 9500
             9002 WRITE(6,20)LAPN,LAPO,TLAP(I)
               20 FORMAT(* -------ABORT--NEW TIMES=*,I4,*  OLD TIMES=*,I4,/,
                 1 * TIME=*,1PE10.3,*  NOT IN SOLUTION SET*)
  455        9500 CALL XEQCCS(*/DUMP,0./GRUMP,MUV=10,SBX.//*)
                  STOP
                  END
```

CARD NR. SEVERITY DETAILS DIAGNOSIS OF PROBLEM

75	I	XTITLE	DATA VARIABLE LIST EXCEEDS ITEM LIST, EXCESS VARIABLES NOT INITIALIZED.
76	I	YTITLE	DATA VARIABLE LIST EXCEEDS ITEM LIST, EXCESS VARIABLES NOT INITIALIZED.

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
2427 DSTAGE3

VARIABLES	SN	TYPE	RELOCATION						
4515	AGAIN	LOGICAL			4055	ALL	INTEGER		
4067	CREEP	LOGICAL			4060	CRLAX	REAL		
1	DELT	REAL		PLOTS	12	DISP	LOGICAL		PROB
0	DUM	REAL	ARRAY	DUM	7	ERROR	LOGICAL		PLOTS
4063	FAULT	LOGICAL			4062	FLAG	LOGICAL		
0	FLOW	LOGICAL		RTIME	4547	FT	REAL		
6	GMALT	REAL		PROB	3	GRAV	REAL		PROB
5	GRAVB	REAL		PROB	4	GRAV5	REAL		PROB
0	HASH1	REAL		PROB	4521	I	INTEGER		
4535	IADDR	INTEGER			4536	IADDR2	INTEGER		
4543	IC	INTEGER			0	ICON	INTEGER		BEGIN
1750	IDOF	INTEGER	ARRAY	DOF	0	IDUM	INTEGER	ARRAY	DUM
4544	IE	INTEGER			5670	IELEM	INTEGER	ARRAY	DOF
0	IFBCON	INTEGER	ARRAY	DOF	4550	IFT	INTEGER		
4540	II	INTEGER			5	IK	INTEGER		BEGIN
1	IKOUNT	INTEGER		BEGIN	2	ILNZ	INTEGER		BEGIN
3	IMASTR	INTEGER		BEGIN	4533	IMASTR1	INTEGER		
4056	IMAX	INTEGER			3720	INDEX1	INTEGER	ARRAY	INDEX
0	INDEX2	INTEGER	ARRAY	INDEX	1750	INDEX4	INTEGER	ARRAY	INDEX

C3-10

VARIABLES SN TYPE RELOCATION

0	INTGRK	INTEGER	ARRAY	K	4551	IP	INTEGER		
4	IQ	INTEGER		BEGIN	2	IREC	INTEGER		BLD
4647	IS	INTEGER	ARRAY		4057	ITMAX	INTEGER		
0	IUP	INTEGER	ARRAY	TIME	4552	I1	INTEGER		
4523	J	INTEGER			4532	JJ	INTEGER		
4066	JUMP	LOGICAL			4717	KEY	INTEGER	ARRAY	
4525	KEYRL1	INTEGER			4546	KEYST	INTEGER		
4516	KEYTM0	INTEGER			2	KEYX	INTEGER	ARRAY	BREAD
2	KP	INTEGER		IO	0	KPTR	INTEGER		BLD
0	KR	INTEGER		IO	3	KT1	INTEGER		IO
4	KT2	INTEGER		IO	5	KT3	INTEGER		IO
6	KT4	INTEGER		IO	7	KT5	INTEGER		IO
1	KW	INTEGER		IO	52	K1X	INTEGER	ARRAY	BREAD
4524	LAP	INTEGER			4522	LAPN	INTEGER		
4520	LAPO	INTEGER			0	LCON	INTEGER		END
4517	LEN	INTEGER			2	LENGTH	INTEGER		PROB
26	LENX	INTEGER	ARRAY	BREAD	4065	LIST	LOGICAL		
5	LK	INTEGER		END	1	LKOUNT	INTEGER		END
3	LLNZ	INTEGER		END	3	LMASTR	INTEGER		END
4	LNODNM	INTEGER		SIZE	4534	LNUM	INTEGER		
4	LQ	INTEGER		END	1	LREC	INTEGER		BLD
1	LRECDR	INTEGER		BREAD	0	LT	INTEGER		PLOTS
4531	MAELM	INTEGER			4530	MIELM	INTEGER		
4545	N	INTEGER			11	NDIM	INTEGER		PROB
4541	NDOF	INTEGER			1	NDT	INTEGER		SIZE
3	NDTNEW	INTEGER		SIZE	4527	NELEM	INTEGER		
0	NET	INTEGER		SIZE	2	NETNEW	INTEGER		SIZE
4542	NFAULT	INTEGER			7	NFLT	INTEGER		PROB
0	NFX	INTEGER		BREAD	10	NLAP	INTEGER		PROB
4526	NLINES	INTEGER			4537	NOD	INTEGER		
6	NPLT	INTEGER		PLOTS	4064	OLDFIL	LOGICAL		
1	OLDWRK	REAL		PROB	4070	OUT	LOGICAL		
2	PLOT	LOGICAL		PLOTS	1	RLAX	REAL	ARRAY	RTIME
4	SCALET	REAL		PLOTS	5	SCALEY	REAL		PLOTS
4061	STRESS	LOGICAL			4553	TITLE	REAL	ARRAY	
4577	TLAP	REAL	ARRAY		4623	TLAPO	REAL	ARRAY	
3	TMAX	REAL		PLOTS	4743	TYPE	REAL	ARRAY	
0	UP	REAL	ARRAY	TIME	0	UT	REAL	ARRAY	TIME
0	XIOBR	REAL	ARRAY	BREADIO	0	XIOWR	REAL	ARRAY	BLDIO
4745	XTITLE	REAL	ARRAY		4757	YTITLE	REAL	ARRAY	

FILE NAMES MODE

0	INPUT		445	OUTPUT		2201	TAPE11		1112	TAPE12	
1323	TAPE14		1534	TAPE20		0	TAPE5	NAME	445	TAPE6	MIXED

EXTERNALS TYPE ARGS

BLD		3	BLD2		3
BREAD		4	BREAD2		4
FRC		3	FTIME	REAL	3
IORDEL3		6	LAPTIM2		8
OPENMS		4	XEQCCS		1

NAMELISTS

IN IOK

C3-11

STATEMENT LABELS

4167	10	FMT		4473	20	FMT		4224	22	FMT	
4233	25	FMT		4273	27	FMT		4315	30	FMT	
4337	32	FMT		4346	33	FMT		4360	34	FMT	
4372	36	FMT		4374	40	FMT		4376	41	FMT	
4422	43	FMT		4444	50	FMT		2526	1005		
2532	1020			2536	1100			0	1120		
2544	1150			2557	1170			2562	1200		
0	1300	INACTIVE		0	1305			0	1310		
2646	1330			0	1335			2655	1340		
2704	1342			2712	1343			2713	1344		
0	1346			0	1347			0	1358		
0	1360			3031	1380			0	1390		
3042	1400			3053	1410			0	1420		
3104	1430			3162	1433			3165	1435		
3172	1440			0	1450	INACTIVE		0	1455	INACTIVE	
0	1460			3214	1500			3254	1600		
3271	1620			0	1700			0	1800		
3340	1810			3342	1815			3365	1818		
3432	1320			0	1825			3436	1827		
3457	1830			3470	1840			3501	1850		
3541	2000			3545	2010			3561	9000		
4457	7001	FMT		3564	9002			3571	9500		

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
2525	1020	I	136 141	7B	INSTACK EXITS
2540	1120	I	145 146	3B	INSTACK
2631	1310	I	199 205	15B	EXT REFS NOT INNER
2637	1305	J	201 204	3B	INSTACK
2651	1335	I	209 210	4B	INSTACK
2674	1347	I	223 244	34B	NOT INNER
2706	1343	J	231 236	5B	INSTACK EXITS
2721	1346	J	240 243	4B	INSTACK
2746	1360	I	252 261	10B	EXITS NOT INNER
2750	1358	J	255 259	3B	INSTACK
3035	1390	I	295 298	3B	INSTACK
3061	1420	N	309 314	21B	EXT REFS EXITS
3205	1460	II	340 345	6B	INSTACK EXITS
3223		IP	350 350	10B	EXT REFS
3242		IP	351 351	10B	EXT REFS
3277	1700	JJ	364 366	12B	EXT REFS
3317	1800	JJ	369 373	17B	EXT REFS EXITS
3510		IP	425 425	10B	EXT REFS
3527		IP	426 426	10B	EXT REFS

COMMON BLOCKS	LENGTH
PROB	11
PLOTS	8
SIZE	5
ID	8
BEGIN	6
END	6
RTIME	21
INDEX	3000 LCM
BLDIO	3256

C3-12

```
  COMMON BLOCKS   LENGTH
        BLD          3
        BREADIO   3256
        BREAD       62
        DOF       3500 LCM
        DUM      10000 LCM
        TIME     20000 LCM
        K        40000 LCM

  STATISTICS
    PROGRAM LENGTH                3033B    1563
    BUFFER LENGTH                 1742B     994
    SCM LABELED COMMON LENGTH    14762B    6642
    LCM LABELED COMMON LENGTH   225324B   76500
          60000B SCM USED
(BOTTOM OF FILE)
```

C3-13

```
      C     *** AGF ********* VERSION 1, MODIFICATION LEVEL 0 *** DK020912 ***
      C     *                                                               *
      C     *   SET UP THE NORMAL EQUATIONS FOR A LEAST SQUARES             *
      C     *   APPROXIMATION OF A GIVEN FUNCTION TABLE                     *
      C     *   IN TERMS OF USER SPECIFIED FUNDAMENTAL FUNCTIONS            *
      C     *                                                               *
      C     *                                                               *
      C     *****************************************************************
      C
            SUBROUTINE AGF2(FCT,X,Y,W,N,IP,WORK,AUX,IER)
            DIMENSION X(1),Y(1),W(1),WORK(1),AUX(1)
      C
            LEVEL 2,Y,WORK
      C
            JER  =IER
            JMP  =1
            IER  =1000
            IF (IP) 9,1,1
          1 IF (N-IP) 9,9,2
          2 IER  =0
            LAST =1+(N-1)*JMP
            IPA  =IP+1
            MP   =IPA*(IPA+1)/2
            MPA  =MP+IPA+1
            Wi   =W(1)
      C
      C     INITIALIZE WORKING STORAGE AND RIGHT HAND SIDE
      C
            DO 3 I=1,MPA
          3   WORK(I)=0.
            SUM  =0.
            WI   =1.
            DO 8 I=1,LAST,JMP
               YI   =Y(I)
               IF (Wi) 6,4,4
          4    WI   =W(I)
               IF (WI) 5,5,6
          5    IER  =100
          6    CALL FCT(X(I),IP,AUX)
               J    =0
               JJ   =MP
      C
      C     CALCULATE COEFFICIENT MATRIX OF THE NORMAL EQUATIONS
      C
               DO 7 K=1,IPA
                  A    =AUX(K)*WI
                  JJ   =JJ+1
                  WORK(JJ)=WORK(JJ)+A*YI
                  DO 7 L=1,K
                     J    =J+1
          7          WORK(J)=WORK(J)+AUX(L)*A
          8    SUM  =SUM+YI*YI*WI
            WORK(MPA)=SUM
            IF (IER) 11,11,9
          9 CONTINUE
```

C3-14

```
   10 CALL WIER(IER,20912)
   11 RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 AGF2

VARIABLES	SN	TYPE	RELOCATION						
140	A	REAL			0	AUX	REAL	ARRAY	F.P.
131	I	INTEGER			0	IER	INTEGER		F.P.
0	IP	INTEGER	F.P.		125	IPA	INTEGER		
135	J	INTEGER			122	JER	INTEGER		
136	JJ	INTEGER			123	JMP	INTEGER		
137	K	INTEGER			141	L	INTEGER		
124	LAST	INTEGER			126	MP	INTEGER		
127	MPA	INTEGER			0	N	INTEGER		F.P.
132	SUM	REAL			0	W	REAL	ARRAY	F.P.
133	WI	REAL			0	WORK	REAL	ARRAY	F.P.
130	W1	REAL			0	X	REAL	ARRAY	F.P.
0	Y	REAL	ARRAY	F.P.	134	YI	REAL		

EXTERNALS	TYPE	ARGS			
FCT		3	F.P.	WIER	2

STATEMENT LABELS

0	1	INACTIVE	0	2	INACTIVE	0	3
0	4	INACTIVE	0	5	INACTIVE	46	6
0	7		0	8		104	9
0	10	INACTIVE	107	11			

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES	
25	3	I	29 30	2B	INSTACK	
36	8	* I	33 52	41B	EXT REFS	NOT INNER
57	7	* K	45 51	13B	NOT INNER	
66	7	L	49 51	3B	INSTACK	

STATISTICS
PROGRAM LENGTH 150B 104
60000B SCM USED

C3-15

```
      C     *** ASNE ******** VERSION 1, MODIFICATION LEVEL 0 *** DK020913 ***
      C     *                                                               *
      C     *  SOLUTION OF THE NORMAL EQUATIONS FOR THE                     *
      C     *  LEAST SQUARES APPROXIMATION OF A TABULATED FUNCTION          *
      C     *  UP TO SPECIFIED ORDER OR PRECISION                           *
      C     *                                                               *
      C     *                                                               *
      C     *****************************************************************
      C
            SUBROUTINE ASNE2(WORK,IP,IOPT,ETA,IRES,IER,ERROR)
            DIMENSION WORK(1)
            DOUBLE PRECISION DAUX,S,WE,TEST
            LOGICAL ERROR
      C
            LEVEL 2,WORK
      C
            JER  =IER
            IRES =0
            IER  =1000
            IF (IP) 27,1,1
          1 IPA  =IP+1
            IPC  =IPA+1
            MP   =IPA*IPC/2
            MPA  =MP+IPC
            WE   =WORK(MPA)
            IER  =0
            TEST =-1.D75
            IF (ETA) 5,2,2
          2 IER  =1
            TEST =ETA*WE
            IF (ETA-1.) 5,4,4
          4 IER  =IER+100
          5 LL   =0
            L    =1
            LL1  =1
      C
      C     FACTORIZE GIVEN MATRIX
      C
            DO 17 I=1,IPA
               LL   =LL+I
               K    =0
          6    S    =0.D0
               J    =LL1
          7    IF (LL-J) 9,9,8
          8    DAUX =WORK(L)
               S    =S+WORK(J)*DAUX
               L    =L+1
               J    =J+1
               GOTO 7
          9    R    =WORK(L)
               S    =R-S
               IF (L-LL) 13,10,13
      C
      C     TEST ON LOSS OF SIGNIFICANCE IN PIVOTAL DIVISOR
      C
```

C3-16

```
   10    IF (S-ABS(1.E-6*R)) 11,11,12
   11    IER  =IER+10
         GOTO 18
   12    S    =DSQRT(S)
         Q    =S
         GOTO 14
   13    S    =S/Q
   14    WORK(L)=S
         K    =K+1
         L    =L+K
         IF (K+1-IPC) 6,6,15
   15    L    =LL+1
         LL1  =L
         WE   =WE-S*S
         IRES =IRES+1
C
C     TEST ON SPECIFIED PRECISION
C
         IF (WE-TEST) 16,16,17
   16    IER  =IER-1
         LL   =LL+1+I
         GOTO 19
   17    CONTINUE
      LL   =MPA
   18 IF (IOPT) 19,27,19
C
C     CALCULATE LEAST SQUARES APPROXIMATION(S)
C
   19 CONTINUE
      IF(.NOT.ERROR)GOTO 200
      IS=(IP+2)*(IP+3)/2
      IE=IPA*(IPA+1)/2
      DO 110 I=1,IE
  110 WORK(IS+I)=WORK(I)
C
  200 IF (IRES) 27,27,20
   20 IS   =MP+IRES
      IG   =LL-1-IRES
      DO 26 I=1,IRES
         J    =IRES+1-I
         Q    =WORK(IG)
         R    =WORK(IS)
         DAUX =R
         WORK(IS)=WE
         WE   =WE+DAUX*DAUX
         IS   =IS-1
         IG   =IG-J
         LL   =LL-1
         L    =LL
         K    =IRES-J
         IL   =IRES
C
C     CALCULATE THE I-TH COEFFICIENT OF THE HIGHEST FIT
C     AND OPTIONALLY OF ALL LOWER FITS
C
```

C3-17

```
                21    L    =L-IL
                      IL   =IL-1
                      LLL  =L
                      ILK  =IL
    115               S    =0.D0
                      J    =L+K
                22    IF (J-L) 24,24,23
                23    DAUX =WORK(J)
                      S    =S+WORK(LLL)*DAUX
    120               LLL  =LLL-ILK
                      ILK  =ILK-1
                      J    =J-1
                      GOTO 22
                24    WORK(L)=(R-S)/Q
    125               K    =K-1
                      IF (IOPT-2) 26,25,26
                25    IF (K) 26,21,21
                26    CONTINUE
                27 IRES =IRES-1
    130            IF (IER) 30,30,28
                28 CONTINUE
                29 CALL WIER(IER,20913)
                30 RETURN
                   END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 ASNE2

VARIABLES	SN	TYPE	RELOCATION					
267	DAUX	DOUBLE		0	ERROR	LOGICAL		F.P.
0	ETA	REAL	F.P.	307	I	INTEGER		
315	IE	INTEGER		0	IER	INTEGER		F.P.
316	IG	INTEGER		317	IL	INTEGER		
321	ILK	INTEGER		0	IOPT	INTEGER		F.P.
0	IP	INTEGER	F.P.	300	IPA	INTEGER		
301	IPC	INTEGER		0	IRES	INTEGER		F.P.
314	IS	INTEGER		311	J	INTEGER		
277	JER	INTEGER		310	K	INTEGER		
305	L	INTEGER		304	LL	INTEGER		
320	LLL	INTEGER		306	LL1	INTEGER		
302	MP	INTEGER		303	MPA	INTEGER		
313	Q	REAL		312	R	REAL		
271	S	DOUBLE		275	TEST	DOUBLE		
273	WE	DOUBLE		0	WORK	REAL	ARRAY	F.P.

EXTERNALS	TYPE	ARGS			
DSQRT	DOUBLE	1	LIBRARY	WIER	2

C3-18

INLINE FUNCTIONS TYPE ARGS
ABS REAL 1 INTRIN

STATEMENT LABELS

0	1	INACTIVE	0	2	INACTIVE	0	4	INACTIVE
34	5		42	6		46	7	
0	8	INACTIVE	55	9		0	10	INACTIVE
0	11	INACTIVE	74	12		100	13	
106	14		0	15	INACTIVE	0	16	INACTIVE
141	17		145	18		0	19	INACTIVE
0	20	INACTIVE	213	21		220	22	
0	23	INACTIVE	231	24		0	25	INACTIVE
245	26		246	27		0	28	INACTIVE
0	29	INACTIVE	254	30		0	110	
162	200							

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES	EXT REFS	EXITS
37	17	*	I	39 78	105B			
160	110		I	88 89	2B	INSTACK		
174	26		I	94 128	52B	OPT		

STATISTICS
PROGRAM LENGTH 324B 212
60000B SCM USED

C3-19

```
      SUBROUTINE BREAD(KEY,LEN,X,NF),RETURNS(R1)
C-----SEQUENTIAL (KEY=0)/UPDATING TO IO FILE HANDLER
C-----MODIFIED AND REINPUT 4/18/74
C-----OPENMS(NF,INDEX,LENGTH OF INDEX,0) MUST BE INSERTED IN MAIN
      DIMENSION X(1),XIO(2)
      DIMENSION NIO(2)
      EQUIVALENCE (XIO,NIO)
C
      COMMON/BREADIO/XIO
      COMMON/BREAD  /NFX,LREC,KEYX(20),LENX(20),K1X(20)
C
      IF(NF.GT.20)RETURN 1
      IF(KEY.EQ.0)KEY=KEYX(NF)+LENX(NF)+1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
      K1=K1+1
      KEY=LREC*(K1-1)+K2
      IF(K1.EQ.K1X(NF).AND.NF.EQ.NFX)GOTO 130
  120 CALL READMS(NF,XIO,LREC,K1)
      NFX=NF
      K1X(NF)=K1
  130 LENX(NF)=NIO(K2)
      IF(LENX(NF).EQ. 1)GOTO 160
      IF(LENX(NF).LT.0)RETURN R1
      IF(K2+LENX(NF).GT.LREC)GOTO 170
      LAST=LENX(NF)
      DO 140 I=1,LAST
  140 X(I)=XIO(I+K2)
  150 CONTINUE
      LEN=LENX(NF)
      KEYX(NF)=KEY
      RETURN
  160 K1=K1+1
      K2=1
      GOTO 120
  170 LEN=LENX(NF)
      IS=1
      IE=LREC-K2
  175 DO 180 I=IS,IE
  180 X(I)=XIO(I+K2)
      IF(LEN-IE.LE.0)GOTO 150
      K1=K1+1
      CALL READMS(NF,XIO,LREC,K1)
      K1X(NF)=K1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      K2=1-IS
      GOTO 175
      ENTRY BWRITE
C-----CAN ONLY REWRITE RECORD JUST READ - NEEDS LEN
      IF(KEY.EQ.0.OR.NFX.NE.NF)RETURN R1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
```

C3-20

```
                      K1=K1+1
                      IF(K2+LENX(NF).GT.LREC)GOTO 300
                      LAST=LENX(NF)
                      DO 220 I=1,LAST
   60               220 XIO(K2+I)=X(I)
                      CALL WRITMS(NF,XIO,LREC,K1,1,0)
                    230 RETURN
                    300 CONTINUE
                      LEN=LENX(NF)
   65                 IS=1
                      IE=LREC-K2
                    310 CONTINUE
                      CALL READMS(NF,XIO,LREC,K1)
                      DO 320 I=IS,IE
   70               320 XIO(I+K2)=X(I)
                      CALL WRITMS(NF,XIO,LREC,K1,1,0)
                      IF(LEN-IE.LE.0)RETURN
                      IS=IE
                      IE=LEN
   75                 IF(LEN-IS.GT.LREC)IE=LREC+IS
                      IS=IS+1
                      K1=K1+1
                      K2=1-IS
                      GOTO 310
   80                 END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 BREAD 127 BWRITE

VARIABLES	SN	TYPE			RELOCATION				
244	I	INTEGER				246	IE	INTEGER	
245	IS	INTEGER				0	KEY	INTEGER	F.P.
2	KEYX	INTEGER	ARRAY	BREAD		241	K1	INTEGER	
52	K1X	INTEGER	ARRAY	BREAD		242	K2	INTEGER	
243	LAST	INTEGER				0	LEN	INTEGER	F.P.
26	LENX	INTEGER	ARRAY	BREAD		1	LREC	INTEGER	BREAD
0	NF	INTEGER		F.P.		0	NFX	INTEGER	BREAD
0	NIO	INTEGER	ARRAY	BREADIO		0	R1	RETURNS	
0	X	REAL	ARRAY	F.P.		0	XIO	REAL	ARRAY BREADIO

EXTERNALS	TYPE	ARGS			
READMS		4	WRITMS		6

STATEMENT LABELS

34	120		42	130	0	140	
62	150		67	160	72	170	
77	175		0	180	0	220	
0	230	INACTIVE	167	300	174	310	
0	320						

C3-21

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
60	140	I	27 28	2B	INSTACK
104	180	I	39 40	2B	INSTACK
161	220	I	59 60	2B	INSTACK
203	320	I	69 70	2B	INSTACK

COMMON BLOCKS	LENGTH
BREADIO	2
BREAD	62

STATISTICS

PROGRAM LENGTH	255B	173
SCM LABELED COMMON LENGTH	100B	64

60000B SCM USED

C3-22

```
      SUBROUTINE BREAD2(KEY,LEN,X,NF),RETURNS(R1)
C-----SEQUENTIAL (KEY=0)/UPDATING TO IO FILE HANDLER
C-----MODIFIED AND REINPUT 4/18/74
C-----OPENMS(NF,INDEX,LENGTH OF INDEX,0) MUST BE INSERTED IN MAIN
      DIMENSION X(1),XIO(2)
      LEVEL 2,X
      DIMENSION NIO(2)
      EQUIVALENCE (XIO,NIO)
C
C
      COMMON/BREADIO/XIO
C
      COMMON/BREAD/NFX,LREC,KEYX(20),LENX(20),K1X(20)
C
      IF(NF.GT.20)RETURN R1
      IF(KEY.EQ.0)KEY=KEYX(NF)+LENX(NF)+1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
      K1=K1+1
      KEY=LREC*(K1-1)+K2
      IF(K1.EQ.K1X(NF).AND.NF.EQ.NFX)GOTO 130
  120 CALL READMS(NF,XIO,LREC,K1)
      NFX=NF
      K1X(NF)=K1
  130 LENX(NF)=NIO(K2)
      IF(LENX(NF).EQ.-1)GOTO 160
      IF(LENX(NF).LT.0)RETURN R1
      IF(K2+LENX(NF).GT.LREC)GOTO 170
      LAST=LENX(NF)
      DO 140 I=1,LAST
  140 X(I)=XIO(I+K2)
  150 CONTINUE
      LEN=LENX(NF)
      KEYX(NF)=KEY
      RETURN
  160 K1=K1+1
      K2=1
      GOTO 120
  170 LEN=LENX(NF)
      IS=1
      IE=LREC-K2
  175 DO 180 I=IS,IE
  180 X(I)=XIO(I+K2)
      IF(LEN-IE.LE.0)GOTO 150
      K1=K1+1
      CALL READMS(NF,XIO,LREC,K1)
      K1X(NF)=K1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      K2=1-IS
      GOTO 175
      ENTRY BWRITE2
C-----CAN ONLY REWRITE RECORD JUST READ - NEEDS LEN
```

C3-23

```
                      IF(KEY.EQ.0.OR.NFX.NE.NF)RETURN R1
                      K1=(KEY-1)/LREC
                      K2=KEY-LREC*K1
                      K1=k1+1
     60               IF(K2+LENX(NF).GT.LREC)GOTO 300
                      LAST=LENX(NF)
                      DO 220 I=1,LAST
                  220 XIO(K2+I)=X(I)
                      CALL WRITMS(NF,XIO,LREC,K1,1,0)
     65           230 RETURN
                  300 CONTINUE
                      LEN=LENX(NF)
                      IS=1
                      IE=LREC-K2
     70           310 CONTINUE
                      CALL READMS(NF,XIO,LREC,K1)
                      DO 320 I=IS,IE
                  320 XIO(I+K2)=X(I)
                      CALL WRITMS(NF,XIO,LREC,K1,1,0)
     75               IF(LEN-IE.LE.0)RETURN
                      IS=IE
                      IE=LEN
                      IF(LEN-IS.GT.LREC)IE=LREC+IS
                      IS=IS+1
     80               K1=K1+1
                      K2=1-IS
                      GOTO 310
                      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	BREAD2	126	BWRITE2

VARIABLES	SN	TYPE			RELOCATION					
244	I	INTEGER				246	IE	INTEGER		
245	IS	INTEGER				0	KEY	INTEGER		F.P.
2	KEYX	INTEGER	ARRAY	BREAD		241	K1	INTEGER		
52	K1X	INTEGER	ARRAY	BREAD		242	K2	INTEGER		
243	LAST	INTEGER				0	LEN	INTEGER		F.P.
26	LENX	INTEGER	ARRAY	BREAD		1	LREC	INTEGER		BREAD
0	NF	INTEGER		F.P.		0	NFX	INTEGER		BREAD
0	NIO	INTEGER	ARRAY	BREADIO		0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.		0	XIO	REAL	ARRAY	BREADIO

EXTERNALS	TYPE	ARGS		
READMS		4	WRITMS	6

STATEMENT LABELS

34	120	42	130	0	140		
62	150	67	160	72	170		

C3-24

STATEMENT LABELS

77	175		0	180	0	220	
0	230	INACTIVE	166	300	173	310	
0	320						

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
60	140	I	30 31	2B	INSTACK
104	180	I	42 43	2B	INSTACK
160	220	I	62 63	2B	INSTACK
203	320	I	72 73	2B	INSTACK

COMMON BLOCKS	LENGTH
BREADIO	2
BREAD	62

STATISTICS

PROGRAM LENGTH	254B	172
SCM LABELED COMMON LENGTH	100B	64

60000B SCM USED

C3-25

```
1               SUBROUTINE BLD(LEN,X,NF),RETURNS(R1)
                DIMENSION X(1),XIO(2),NIO(2)
                EQUIVALENCE (NIO,XIO)
          C
5               COMMON/BLDIO/XIO
                COMMON/BLD  /KPTR,LREC,IREC
          C
                LREC1=LREC-1
                IF(KPTR.EQ.0.OR.NF.EQ.0)RETURN R1
10              IF(LEN.LT.1)RETURN R1
                IF(LEN.GT.LREC1)GOTO 5
                JPTR=KPTR+LEN
                IF(JPTR.LE.LREC)GOTO 30
              5 IE=LREC-KPTR
15           10 IS=1
                NIO(KPTR)=LEN
                IF(IE.LT.1)GOTO 21
             15 DO 20 I=IS,IE
             20 XIO(KPTR+I)=X(I)
20           21 IF(LEN-IE.LE.0)GOTO 25
                CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                IS=IE
                IE=LEN
25              IF(LEN-IS.GT.LREC)IE=LREC+IS
                IS=IS+1
                KPTR=1-IS
                GOTO 15
             25 KPTR=IE+1+KPTR
30              IF(KPTR.GT.LREC)GOTO 26
                RETURN
             26 CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                KPTR=1
35              RETURN
             30 NIO(KPTR)=LEN
                DO 40 I=1,LEN
             40 XIO(KPTR+I)=X(I)
                KPTR=JPTR+1
40              IF(KPTR.GT.LREC)GOTO 26
                RETURN
                ENTRY FRC
                NIO(KPTR)=-2
                CALL WRITMS(NF,XIO,LREC,IREC,0)
45              IREC=IREC+1
                KPTR=0
                RETURN
                END
```

SYMBOLIC REFERENCE MAP (R=1)

C3-26

ENTRY POINTS
3 BLD 113 FRC

VARIABLES	SN	TYPE		RELOCATION					
146	I	INTEGER			144	IE	INTEGER		
2	IREC	INTEGER		BLD	145	IS	INTEGER		
143	JPTR	INTEGER			0	KPTR	INTEGER		BLD
0	LEN	INTEGER		F.P.	1	LREC	INTEGER		BLD
142	LRECI	INTEGER			0	NF	INTEGER		F.P.
0	NIO	INTEGER	ARRAY	BLDIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BLDIO

EXTERNALS	TYPE	ARGS
WRITMS		5

STATEMENT LABELS

32	5	0	10	INACTIVE	37	15	
0	20	46	21		65	25	
72	26	100	30		0	40	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
44	20	I	18 19	2B	INSTACK
105	40	I	37 38	2B	INSTACK

COMMON BLOCKS	LENGTH
BLDIO	2
BLD	3

STATISTICS

PROGRAM LENGTH	147B	103
SCM LABELED COMMON LENGTH	5B	5

60000B SCM USED

C3-27

```
1               SUBROUTINE BLD2(LEN,X,NF),RETURNS(R1)
                DIMENSION X(1),XIO(2),NIO(2)
                LEVEL 2,X
                EQUIVALENCE (NIO,XIO)
5               COMMON/BLDIO/XIO
                COMMON/BLD/KPTR,LREC,IREC
          C
                LREC1=LREC-1
                IF(KPTR.EQ.0.OR.NF.EQ.0)RETURN R1
10              IF(LEN.LT.1)RETURN R1
                IF(LEN.GT.LREC1)GOTO 5
                JPTR=KPTR+LEN
                IF(JPTR.LE.LREC)GOTO 30
              5 IE=LREC-KPTR
15           10 IS=1
                NIO(KPTR)=LEN
                IF(IE.LT.1)GOTO 21
             15 DO 20 I=IS,IE
             20 XIO(KPTR+I)=X(I)
20           21 IF(LEN-IE.LE.0)GOTO 25
                CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                IS=IE
                IE=LEN
25              IF(LEN-IS.GT.LREC)IE=LREC+IS
                IS=IS+1
                KPTR=1-IS
                GOTO 15
             25 KPTR=IE+1+KPTR
30              IF(KPTR.GT.LREC)GOTO 26
                RETURN
             26 CALL WRITMS(NF,XIO,LREC,IREC,0)
                IREC=IREC+1
                KPTR=1
35              RETURN
             30 NIO(KPTR)=LEN
                DO 40 I=1,LEN
             40 XIO(KPTR+I)=X(I)
                KPTR=JPTR+1
40              IF(KPTR.GT.LREC)GOTO 26
                RETURN
                ENTRY FRC2
                NIO(KPTR)=-2
                CALL WRITMS(NF,XIO,LREC,IREC,0)
45              IREC=IREC+1
                KPTR=0
                RETURN
                END
```

SYMBOLIC REFERENCE MAP (R=1)

C3-28

ENTRY POINTS
3 BLD2 113 FRC2

VARIABLES	SN	TYPE		RELOCATION					
146	I	INTEGER			144	IE	INTEGER		
2	IREC	INTEGER		BLD	145	IS	INTEGER		
143	JPTR	INTEGER			0	KPTR	INTEGER		BLD
0	LEN	INTEGER		F.P.	1	LREC	INTEGER		BLD
142	LREC1	INTEGER			0	NF	INTEGER		F.P.
0	NIO	INTEGER	ARRAY	BLDIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BLDIO

EXTERNALS	TYPE	ARGS
WRITMS		5

STATEMENT LABELS

32	5	0	10	INACTIVE	37	15	
0	20	46	21		65	25	
72	26	100	30		0	40	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
44	20	I	18 19	2B	INSTACK
105	40	I	37 38	2B	INSTACK

COMMON BLOCKS	LENGTH
BLDIO	2
BLD	3

STATISTICS

PROGRAM LENGTH	147B	103
SCM LABELLED COMMON LENGTH	5B	5

60000B SCM USED

C3-29

```
1               SUBROUTINE IORDEL3(NUM,IX,Y,NY,LYDIM,AUX)
              C ORDERS IN ASCENDING VALUE - X(LYDIM) AND Y(LYDIM,NY) ACCORDING TO
              C     IX(NUM)
                    DIMENSION IX(1),Y(LYDIM,1),AUX(1)
5                   LOGICAL AGAIN
              C
                    LEVEL 2,IX,Y,AUX
              C
              C
10            C
                  1 LAST=NUM-1
                100 AGAIN=.FALSE.
              C
                    I=1
15                2 IF(I.GT.LAST)GOTO 21
              C
                    IF(IX(I+1)-IX(I))4,16,20
                  4 IX1=IX(I)
                    IX(I)=IX(I+1)
20                  IX(I+1)=IX1
                  5 IF(NY.EQ.0)GOTO 19
                    DO 6 J=1,NY
                  6 AUX(J)=Y(I,J)
                    DO 10 J=1,NY
25                10 Y(I,J)=Y(I+1,J)
                    DO 15 J=1,NY
                  15 Y(I+1,J)=AUX(J)
                    GO TO 19
              C
30                16 IX1=I+1
                    IF(IX1.GT.LAST)GOTO 110
                    DO 18 J=IX1,LAST
                    IX(J)=IX(J+1)
                    IF(NY.EQ.0)GO TO 18
35                  DO 17 JJ=1,NY
                  17 Y(J,JJ)=Y(J+1,JJ)
                  18 CONTINUE
                 110 NUM=NUM-1
                    LAST=LAST-1
40            C
                  19 AGAIN=.TRUE.
                  20 CONTINUE
                    I=I+1
                    GOTO 2
45            C
                  21 CONTINUE
                    IF(AGAIN)GOTO 1
                    RETURN
                    END
```

C3-30

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 IORDEL3

VARIABLES	SN	TYPE	RELOCATION					
102	AGAIN	LOGICAL		0	AUX	REAL	ARRAY	F.P.
104	I	INTEGER		0	IX	INTEGER	ARRAY	F.P.
105	IX1	INTEGER		106	J	INTEGER		
107	JJ	INTEGER		103	LAST	INTEGER		
0	LYDIM	INTEGER	F.P.	0	NUM	INTEGER		F.P.
0	NY	INTEGER	F.P.	0	Y	REAL	ARRAY	F.P.

STATEMENT LABELS

12	1		16	2		0	4	INACTIVE
0	5	INACTIVE	0	6		0	10	
0	15		52	16		0	17	
67	18		73	19		74	20	
76	21		0	100	INACTIVE	70	110	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
31	6	J	22 23	3B	INSTACK
40	10	J	24 25	2B	INSTACK
47	15	J	26 27	3B	INSTACK
60	18	* J	32 37	10B	NOT INNER
65	17	JJ	35 36	2B	INSTACK

STATISTICS
PROGRAM LENGTH 116B 78
60000B SCM USED

C3-31

```
      SUBROUTINE LAPTIM2(DLAP,TLAP,LAP,S,FIT,LT,WORK,ERROR)
C
C  LAPLACE TO TIME INVERSION FOR INDIVIDUAL POINT
C     DLAP - FUNCTION VALUES IN LAPLACE SPACE
C     TLAP - LAPLACE REDUCED TIMES FOR EACH DLAP (ARGUMENT)
C     LAP  - NUMBER OF LAPLACE REDUCED TIMES (VECTOR LENGTH)
C     S    - OUTPUT CONSTANTS S0,S1,S2,... FOR TIME SERIES
C            E(T)=S0+S1(1-EXP(-T/RLAX(1)))+S2(1-EXP(-T/RLAX(2)))...
C            (VECTOR LENGTH LT)
C     FIT  - VARIANCE OF FIT TO SERIES
C     LT   - ORDER OF APPROXIMATION
C     LT+1 - GIVEN NUMBER OF TERMS IN SERIES TO BE FITTED
C            (LESS THAN LAP)
C     FLOW - .TRUE.  IF CONSTANT FLOW TERM INCLUDED IN UT
C     RLAX - RELAXATION TIMES TO BE USED (LENGTH LT)
C
C
      DIMENSION DLAP(1),TLAP(1),S(1)
      DIMENSION WORK(1),AUX(20),W(20)
      LOGICAL FLOW,ERROR
C
      LEVEL 2,DLAP,S,FIT,WORK
C
      COMMON/RTIME/FLOW,RLAX(20)
      EXTERNAL FCT
C
C*********************************************************************
C
C  SET UP
C
      IORDER=LT
      IF(FLOW)IORDER=IORDER+1
      LEN=IORDER+1
      DO 1100 I=1,LAP
      W(I)=-1.
 1100 CONTINUE
C
      CALL AGF2(FCT,TLAP,DLAP,W,LAP,IORDER,WORK,AUX,IER)
C
C  SOLUTION OF NORMAL EQUATIONS
C
      ETA=-1.
      IOPT=1
      CALL ASNE2(WORK,IORDER,IOPT,ETA,IRES,IER,ERROR)
C
      IC=IORDER*(IORDER+1)/2
      IF=(IORDER+1)*(IORDER+2)/2+1+IORDER
      DO 1200 I=1,LEN
      S(I)=WORK(IC+I)
 1200 CONTINUE
      FIT=ABS(WORK(IF))/FLOAT(LAP-LEN)
      IF(.NOT.ERROR)GOTO 1500
      IS=(IORDER+2)*(IORDER+3)/2
      IC=LEN*(LEN+1)/2
      DO 1310 I=1,LEN
```

C3-32

```
                    DO 1310 J=1,I
                    IOLD=(I-1)*I/2+I-J+1
                    INEW=(I-1)*I/2+J
                    WORK(INEW)=WORK(IOLD+IS)
 60          1310 CONTINUE
                    CALL MIS2(WORK(1),LEN,IER)
                    DO 1320 I=1,IC
                    WORK(I)=WORK(I)*FIT
             1320 CONTINUE
 65                 WRITE(20)(WORK(I),I=1,IC)
             1500 CONTINUE
                    FIT=SQRT(FIT)
                    RETURN
                    END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 LAPTIM2

VARIABLES	SN	TYPE		RELOCATION					
165	AUX	REAL	ARRAY		0	DLAP	REAL	ARRAY	F.P.
0	ERROR	LOGICAL		F.P.	154	ETA	REAL		
0	FIT	REAL		F.P.	0	FLOW	LOGICAL		RTIME
152	I	INTEGER			157	IC	INTEGER		
153	IER	INTEGER			160	IF	INTEGER		
164	INEW	INTEGER			163	IOLD	INTEGER		
155	IOPT	INTEGER			150	IORDER	INTEGER		
156	IRES	INTEGER			161	IS	INTEGER		
162	J	INTEGER			0	LAP	INTEGER		F.P.
151	LEN	INTEGER			0	LT	INTEGER		F.P.
1	RLAX	REAL	ARRAY	RTIME	0	S	REAL	ARRAY	F.P.
0	TLAP	REAL	ARRAY	F.P.	211	W	REAL	ARRAY	
0	WORK	REAL	ARRAY	F.P.					

FILE NAMES	MODE
TAPE20	UNFMT

EXTERNALS	TYPE	ARGS			
AGF2		9	ASNE2		7
FCT		0	MIS2		3
SQRT	REAL	1 LIBRARY			

INLINE FUNCTIONS	TYPE	ARGS					
ABS	REAL	1	INTRIN	FLOAT	REAL	1	INTRIN

STATEMENT LABELS

0	1100	0	1200	0	1310
0	1320	111	1500		

C3-33

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
15	1100	I	34 36	2B	INSTACK
44	1200	I	48 50	2B	INSTACK
64	1310	* I	55 60	10B	NOT INNER
67	1310	J	56 60	4B	INSTACK
101	1320	I	62 64	3B	INSTACK

COMMON BLOCKS	LENGTH
RTIME	21

STATISTICS

PROGRAM LENGTH	235B	157
SCM LABELED COMMON LENGTH	25B	21

60000B SCM USED

C3-34

```
      C
            SUBROUTINE FCT(TLAP,IORDER,DLAP)
      C
      C  FUNDAMENTAL FUNCTIONS FOR LAPLACE INVERSION
      C
            DIMENSION DLAP(1)
            LOGICAL FLOW
            COMMON/RTIME/FLOW,RLAX(20)
            DLAP(1)=1.
            IF(FLOW)GOTO 2000
            IF(IORDER.LE.0)RETURN
            IX=1
            IE=IORDER
      1000  DO 1100 I=1,IE
            DLAP(I+IX)=1./(1.+RLAX(I)*TLAP)
      1100  CONTINUE
            RETURN
      2000  CONTINUE
            DLAP(2)=1./TLAP
            IE=IORDER-1
            IF(IORDER.LE.0)RETURN
            IX=2
            GOTO 1000
            END
```

CARD NR. SEVERITY DETAILS DIAGNOSIS OF PROBLEM

19 I DLAP ARRAY REFERENCE OUTSIDE DIMENSION BOUNDS.

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 FCT

VARIABLES	SN	TYPE		RELOCATION					
0	DLAP	REAL	ARRAY	F.P.	0	FLOW	LOGICAL	RTIME	
36	I	INTEGER			35	IE	INTEGER		
0	IORDER	INTEGER		F.P.	34	IX	INTEGER		
1	RLAX	REAL	ARRAY	RTIME	0	TLAP	REAL	F.P.	

STATEMENT LABELS
14 1000 0 1100 24 2000

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
21	1100	I	14 16	3B	INSTACK

C3-35

COMMON BLOCKS LENGTH
RTIME 21

TATISTICS

PROGRAM LENGTH	37B	31
SCM LABELED COMMON LENGTH	25B	21

60000B SCM USED

C3-36

```
      1              SUBROUTINE ORDER(NUM,X,Y,NY,LYDIM,AUX)
                   C ORDERS IN ASCENDING VALUE - X(LYDIM) AND Y(LYDIM,NY) ACCORDING TO
                   C     X(NUM)
                         DIMENSION X(1),Y(LYDIM,1),AUX(1)
      5                  LOGICAL AGAIN
                   C
                   C
                       1 LAST=NUM-1
                     100 AGAIN=.FALSE.
     10                  DO 20 I=1,LAST
                       2 IF(X(I+1).GE.X(I))GOTO 20
                         XI=X(I)
                         X(I)=X(I+1)
                         X(I+1)=XI
     15                  IF(NY.EQ.0)GOTO 19
                         DO 6 J=1,NY
                       6 AUX(J)=Y(I,J)
                         DO 10 J=1,NY
                      10 Y(I,J)=Y(I+1,J)
     20                  DO 15 J=1,NY
                      15 Y(I+1,J)=AUX(J)
                      19 AGAIN=.TRUE.
                      20 CONTINUE
                         IF(AGAIN)GOTO 100
     25                  RETURN
                         END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	ORDER

VARIABLES

SN		TYPE			RELOCATION	SN		TYPE		RELOCATION
54	AGAIN	LOGICAL				0	AUX	REAL	ARRAY	F.P.
56	I	INTEGER				60	J	INTEGER		
55	LAST	INTEGER				0	LYDIM	INTEGER		F.P.
0	NUM	INTEGER			F.P.	0	NY	INTEGER		F.P.
0	X	REAL	ARRAY		F.P.	57	XI	REAL		
0	Y	REAL	ARRAY		F.P.					

STATEMENT LABELS

0	1	INACTIVE		0	2	INACTIVE		0	5	INACTIVE	
0	6			0	10			0	15		
46	19			47	20			14	100		

LOOPS

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
16	20	*	I	10 23	32B	NOT INNER
27	6		J	16 17	2B	INSTACK
35	10		J	18 19	2B	INSTACK
44	15		J	20 21	2B	INSTACK

C3-37

STATISTICS
PROGRAM LENGTH 66B 54
60000B SCM USED

APPENDIX C.4


```
      C  PROGRAM SOLBLD7 BUILDS NEW TIME FILE FROM INVERSION TIME FILE
      C     GIVEN FAULT DISPLACEMENTS
            PROGRAM SOLBLD(INPUT,OUTPUT,TAPE8,TAPE11,TAPE13,
           1 TAPE5=INPUT,TAPE6=OUTPUT)
      C
      C  READS FROM SOLUTION FILE
      C  INPUTS TIMES FOR COMPUTATION AND DATA
      C  DISP, ERROR, ID ALL NOBS LONG
      C
            DIMENSION TITLE(20),PTITLE(20)
            DIMENSION IFBCON(200),IFAULT(200,2),IDOF(200)
            DIMENSION DFAULT(200,2)
            EQUIVALENCE (IFAULT,DFAULT)
            DIMENSION TIME(20),DUM(500),IDUM(500),TLAP(20)
            DIMENSION UT(20,200),IUT(20,200)
            EQUIVALENCE (DUM,IDUM),(UT,IUT)
            LOGICAL FLOW,GRAV,OLDWRK,GRAVS,GRAVB,GMALT,OUT,FAULT,AGAIN
      C
            COMMON/IO/KR,KW,KP,KT1,KT2,KT3
            COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,
           1 NLAP,NDIM,DISP2
            COMMON/SIZE/NET,NDT,NETNEW,NDTNEW,LNODNW
            COMMON/RTIME/FLOW,RLAX(20)
      C
            COMMON/BREAD/NFX,LRECBR,KEYX(20),LENX(20),K1X(20)
            COMMON/BREADIO/XIOBR(2048)
            COMMON/BLD  /KPTR,LREC,IREC
            COMMON/BLDIO/XIOWR(2048)
            COMMON/INDEX/INDEX1(1000),INDEX3(1000)
      C
            DATA KPTR/1/,LREC/2048/,IREC/1/
            DATA NFX/0/,KEYX/20*0/,LENX/20*0/,K1X/20*0/,LRECBR/2048/
      C
            NAMELIST/IOK/KR,KW,KP,KT1,KT2,KT3,OUT
            NAMELIST/IN/TUNIT,FLOW,OLDWRK,GRAV,GRAVS,GRAVB,GMALT,NDIM,DISP2
      C
      C*************************************************************************
      C
            OUT=.FALSE.
      C----SETUP
            KR=5
            KW=6
            KP=7
      C----NEW TIME FILE
            KT1=11
            KT2=12
      C----TIME FILE
            KT3=13
      C
            READ(5,IOK)
            CALL OPENMS(KT1,INDEX1,1000,0)
            CALL OPENMS(KT3,INDEX3,1000,0)
            READ(KR,IN)
      C
      C----INPUT SOLUTION FILE
```

C4-2

```
      C
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,TITLE,KT3),RETURNS(9000)
            KEYTMP=0
 60         CALL BREAD(KEYTMP,LEN,HASH1,KT3),RETURNS(9000)
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,NET,KT3),RETURNS(9000)
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,TLAP,KT3),RETURNS(9000)
 65         LAP=LEN
            KEYTMP=0
            CALL BREAD(KEYTMP,LEN,FLOW,KT3),RETURNS(9000)
            LT=LEN-1
            KEYTMP=0
 70         CALL BREAD(KEYTMP,NDOF,IDOF,KT3),RETURNS(9000)
      C
      C----READ TITLE,INPUT DATA
      C
            READ(KR,10)PTITLE
 75       10 FORMAT(20A4)
      C
       1100 CONTINUE
      C----READ FAULT PARAMETERS
            READ(KR,*)NFAULT
 80         READ(KR,*)(IFAULT(J,1),DFAULT(J,2),J=1,NFAULT)
       1150 CONTINUE
      C
      C----SCAN SOLUTION FILE FOR NUMBERS OF FAULT
      C
 85         KEYTMP=0
            CALL BREAD(KEYTMP,LEN,DUM,KT3),RETURNS(9000)
            KEYST=KEYTMP
            IF(LEN.EQ.1)GOTO 9100
       1200 CONTINUE
 90         KEYTMP=0
            CALL BREAD(KEYTMP,LEN,DUM,KT3),RETURNS(9000)
            IF(LEN.EQ.1)GOTO 1210
            GOTO 1200
       1210 KEYTMP=0
 95         CALL BREAD(KEYTMP,NFLT,IFBCON,KT3),RETURNS(9000)
            KEYFLT=KEYTMP
            IF(NFLT.EQ.1)GOTO 9200
       1230 CONTINUE
      C
100   C----OUTPUT PROBLEM HEADINGS
      C
            WRITE(KW,20)TITLE,PTITLE
         20 FORMAT(1H1,*PROGRAM SOLBLD? BUILDS TIME FILE FROM INVERSION*,
           1/,* SOLUTION FILE? *,20A4/* NEW TIME FILE? *,20A4//
105        2 * INPUT DATA?*)
            WRITE(KW,IN)
            WRITE(KW,22)HASH1,NET,NDT
         22 FORMAT(1H0,*MODEL HASH CODE=*,Z8,5X,*NET=*,I10,10X,*NDT=*,I10)
            WRITE(KW,24)LT,FLOW,(RLAX(I),I=1,LT)
110      24 FORMAT(1H0,*TIME SERIES CONSTANTS*,5X,*ORDER=*,I4,10X,*FLOW=*,L2,/
```

C4-3

```
     1 ,20(1X,1PE11.3/))
       WRITE(KW,28)NDOF,(IDOF(I),I=1,NDOF)
    28 FORMAT(1H0,*DOF IN TIME FILE - NDOF=*,I6/,20(1X,20I5/))
       WRITE(KW,32)NFLT,(IFBCON(I),I=1,NFLT)
115    32 FORMAT(1H0,*FAULT DOF IN TIME FILE - NFLT=*,I6/,20(1X,20I5,/))
       WRITE(KW,34)NFAULT
    34 FORMAT(1H0,*FAULT DOF REQUESTED FOR SOLUTION - NFAULT=*,I6)
  1255 WRITE(KW,36)(IFAULT(I,1),DFAULT(I,2),I=1,NFAULT)
    36 FORMAT(1X,I5,1PE11.3)
120  1260 CONTINUE
     C
     C----CHECK FOR ALL FAULT ELEMENTS REPRESENTED
     C
          NEXT=1
125  1320 AGAIN=.FALSE.
          DO 1330 I=1,NFLT
          IF(IFAULT(NEXT,1).NE.IFBCON(I))GOTO 1330
          IDUM(NEXT)=I
          NEXT=NEXT+1
130       IF(NEXT.GT.NFAULT)GOTO 1350
  1325 CONTINUE
          AGAIN=.TRUE.
  1330 CONTINUE
          IF(AGAIN)GOTO 1320
135       GOTO 9300
  1350 CONTINUE
          CALL IORDER(NFAULT,IDUM,IFAULT,2,100,DUM(101))
     C
     C
140  C----READ AND CONSTRUCT ARRAY
     C
          WRITE(KW,48)
     48 FORMAT(1H0,*CONSTRUCT TIME FILE*)
          CALL BLD(20,PTITLE,KT1),RETURNS(9000)
145       CALL BLD(11,HASH1,KT1),RETURNS(9000)
          CALL BLD(5,NET,KT1),RETURNS(9000)
          CALL BLD(LAP,TLAP,KT1),RETURNS(9000)
          LEN=LT+1
          CALL BLD(LEN,FLOW,KT1),RETURNS(9000)
150       CALL BLD(NDOF,IDOF,KT1),RETURNS(9000)
          NDOF=NDOF+1
          DO 2010 JJ=1,NDOF
          DO 2010 I=1,20
  2010 UT(I,JJ)=0.
155       NEXT=1
  2100 CONTINUE
          FAULT=.FALSE.
          DO 2550 II=1,NFLT
          IFB=IFBCON(II)
160       AGAIN=.FALSE.
          DP=DFAULT(NEXT,2)
          IFLT=IFAULT(NEXT,1)
          KEYTMP=0
          IF(II.EQ.1)KEYTMP=KEYST
165       CALL BREAD(KEYTMP,LEN,DUM,KT3),RETURNS(9000)
```

C4-4

```
      2150 CONTINUE
           IF(IFLT.NE.IFB)GOTO 2200
           FAULT=.TRUE.
           AGAIN=.TRUE.
170        IF(IDUM(1).NE.0)GOTO 9300
           IF(NEXT.NE.1)GOTO 2160
           UT(1,1)=DUM(1)
      2160 UT(2,1)=SQRT(UT(2,1)**2+DUM(2)**2)
           DO 2170 I=3,LEN
175        UT(I,1)=DUM(I)*DP+UT(I,1)
      2170 CONTINUE
      2200 CONTINUE
           DO 2500 JJ=2,NDOF
           KEYTMP=0
180        CALL BREAD(KEYTMP,LEN,DUM,KT3),RETURNS(9000)
           IF(.NOT.AGAIN)GOTO 2500
           IF(NEXT.GT.1)GOTO 2210
           UT(1,JJ)=DUM(1)
      2210 UT(2,JJ)=SQRT(UT(2,JJ)**2+DUM(2)**2)
           DO 2250 I=3,LEN
185        UT(I,JJ)=DUM(I)*DP+UT(I,JJ)
      2250 CONTINUE
      2500 CONTINUE
      2510 IF(FAULT)GOTO 2540
           GOTO 2550
190   2540 CONTINUE
           NEXT=NEXT+1
           FAULT=.FALSE.
           IF(NEXT.GT.NFAULT)GOTO 2580
195   2550 CONTINUE
           GOTO 9300
      2580 CONTINUE
      C
           DO 2600 I=1,NDOF
200        CALL BLD(LEN,UT(1,I),KT1),RETURNS(9000)
           IF(OUT)WRITE(KW,55)(UT(JJ,I),JJ=1,LEN)
      2600 CONTINUE
        55 FORMAT(1X,I10,1P10E11.3)
           NDOF=NDOF-1
205        CALL BLD(1,NDOF,KT1),RETURNS(9000)
           CALL BLD(1,NDOF,KT1),RETURNS(9000)
           CALL FRC(0,0,KT1),RETURNS(9000)
      C
           WRITE(KW,99)
210     99 FORMAT(* -------COMPLETION OF SOLBLD------*)
           STOP
      C
      C---------ERROR MESSAGES
      C
215   9000 WRITE(6,9001)KEYTMP,LEN
      9001 FORMAT(* -----ERROR IN BREAD----KEYTMP/LEN=*,2I10)
           STOP
      9100 WRITE(6,9101)
      9101 FORMAT(* -------NO SOLUTIONS IN TIME FILE------*)
220        STOP
```

C4-5

```
            9200 WRITE(6,9201)
            9201 FORMAT(* -----NO INTERNAL BOUNDARIES-----*)
                 STOP
            9300 WRITE(6,9301)IFAULT(NEXT,1),NEXT
  225       9301 FORMAT(* -------NOT INCLUDED FAULT DOF/DOF/NEXT=*,3I10)
                 STOP
                 END
```

9300 CALL XFER(3)

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
2705 SOLBLD

VARIABLES	SN	TYPE		RELOCATION					
4056	AGAIN	LOGICAL			4100	DFAULT	REAL	ARRAY	
12	DISP2	REAL		PROB	4076	DP	REAL		
4720	DUM	REAL	ARRAY		4055	FAULT	LOGICAL		
0	FLOW	LOGICAL		RTIME	6	GMALT	LOGICAL		PROB
3	GRAV	LOGICAL		PROB	5	GRAVB	LOGICAL		PROB
4	GRAVS	LOGICAL		PROB	0	HASH1	REAL		PROB
4071	I	INTEGER			16124	IDOF	INTEGER	ARRAY	
4720	IDUM	INTEGER	ARRAY		4100	IFAULT	INTEGER	ARRAY	
4075	IFB	INTEGER			15614	IFBCON	INTEGER	ARRAY	
4077	IFLT	INTEGER			4074	II	INTEGER		
0	INDEX1	INTEGER	ARRAY	INDEX	1750	INDEX3	INTEGER	ARRAY	INDEX
2	IREC	INTEGER		BLD	5704	IUT	INTEGER	ARRAY	
4066	J	INTEGER			4073	JJ	INTEGER		
4070	KEYFLT	INTEGER			4067	KEYST	INTEGER		
4060	KEYTMP	INTEGER			2	KEYX	INTEGER	ARRAY	BREAD
2	KP	INTEGER		IO	0	KPTR	INTEGER		BLD
0	KR	INTEGER		IO	3	KT1	INTEGER		IO
4	KT2	INTEGER		IO	5	KT3	INTEGER		IO
1	KW	INTEGER		IO	52	K1X	INTEGER	ARRAY	BREAD
4062	LAP	INTEGER			4061	LEN	INTEGER		
2	LENGTH	INTEGER		PROB	26	LENX	INTEGER	ARRAY	BREAD
4	LNODNW	INTEGER		SIZE	1	LREC	INTEGER		BLD
1	LRECBR	INTEGER		BREAD	4063	LT	INTEGER		
11	NDIM	INTEGER		PROB	4064	NDOF	INTEGER		
1	NDT	INTEGER		SIZE	3	NDTNEW	INTEGER		SIZE
0	NET	INTEGER		SIZE	2	NETNEW	INTEGER		SIZE
4072	NEXT	INTEGER			4065	NFAULT	INTEGER		
7	NFLT	INTEGER		PROB	0	NFX	INTEGER		BREAD
10	NLAP	INTEGER		PROB	1	OLDWRK	LOGICAL		PROB
4054	OUT	LOGICAL			15570	PTITLE	REAL	ARRAY	
1	RLAX	REAL	ARRAY	RTIME	16434	TIME	REAL	*UNDEF	
15544	TITLE	REAL	ARRAY		16460	TLAP	REAL	ARRAY	
4057	TUNIT	REAL			5704	UT	REAL	ARRAY	
0	XIOBR	REAL	ARRAY	BREADIO	0	XIOWR	REAL	ARRAY	BLDIO

C4-6

```
FILE NAMES      MODE
   0 INPUT                445 OUTPUT              1557 TAPE11            2224 TAPE13
   0 TAPE5    NAME        445 TAPE6     FMT       1112 TAPE8

EXTERNALS       TYPE  ARGS
      BLD               3                    BREAD            4
      FRC               3                    IORDER           6
      OPENMS            4                    SQRT     REAL    1 LIBRARY

NAMELISTS
      IN              IOK

STATEMENT LABELS
 3574 10     FMT                3616 20     FMT                3645 22     FMT
 3662 24     FMT                3700 28     FMT                3713 32     FMT
 3726 34     FMT                3744 36     FMT                3752 48     FMT
 3762 55     FMT                3770 99     FMT                   0 1100       INACTIVE
    0 1150       INACTIVE        3003 1200                      3011 1210
    0 1230       INACTIVE           0 1255       INACTIVE        0 1260       INACTIVE
 3072 1320                       0 1325       INACTIVE        3100 1330
 3103 1350                       0 2010                          0 2100       INACTIVE
    0 2150       INACTIVE        3161 2160                      0 2170
 3173 2200                       3207 2210                      0 2250
 3222 2500                          0 2510       INACTIVE     3230 2540
 3234 2550                       3237 2580                      0 2600
 3273 9000                       4002 9001   FMT                3276 9100
 4013 9101   FMT                3301 9200                      4024 9201   FMT
 3304 9300                       4036 9301   FMT

LOOPS  LABEL   INDEX    FROM-TO   LENGTH   PROPERTIES
 2764          J         80 80      10B          EXT REFS
 3056          I        118 118     10B          EXT REFS
 3074  1330    I        126 133      5B   INSTACK   EXITS
 3131  2010    JJ       152 154      4B          NOT INNER
 3132  2010    I        153 154      2B   INSTACK
 3137  2550    II       158 195    100B          EXT REFS   EXITS   NOT INNER
 3170  2170    I        174 176      3B   INSTACK
 3200  2500    JJ       178 188     26B          EXT REFS   EXITS   NOT INNER
 3217  2250    I        185 187      3B   INSTACK
 3243  2600    I        199 202     16B          EXT REFS   EXITS

COMMON BLOCKS   LENGTH
      IO             6
      PROB          11
      SIZE           5
      RTIME         21
      BREAD         62
      BREADIO     2048
      BLD            3
      BLDIO       2048
      INDEX       2000

STATISTICS
  PROGRAM LENGTH          14216B   6286
  BUFFER LENGTH            2266B   1206
```

C4-7

```
  STATISTICS
    SCM LABELED COMMON LENGTH   14074B    6204
          100000B SCM USED
(BOTTOM OF FILE)
```

C4-8

```
      SUBROUTINE BREAD(KEY,LEN,X,NF),RETURNS(R1)
C-----SEQUENTIAL (KEY=0)/UPDATING TO ID FILE HANDLER
C-----MODIFIED AND REINPUT 4/18/74
C-----OPENMS(NF,INDEX,LENGTH OF INDEX,0) MUST BE INSERTED IN MAIN
      DIMENSION X(1),XIO(2)
      DIMENSION NIO(2)
      EQUIVALENCE (XIO,NIO)
C
      COMMON/BREADIO/XIO
      COMMON/BREAD  /NFX,LREC,KEYX(20),LENX(20),K1X(20)
C
      IF(NF.GT.20)RETURN R1
      IF(KEY.EQ.0)KEY=KEYX(NF)+LENX(NF)+1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
      K1=K1+1
      KEY=LREC*(K1-1)+K2
      IF(K1.EQ.K1X(NF).AND.NF.EQ.NFX)GOTO 130
  120 CALL READMS(NF,XIO,LREC,K1)
      NFX=NF
      K1X(NF)=K1
  130 LENX(NF)=NIO(K2)
      IF(LENX(NF).EQ.-1)GOTO 160
      IF(LENX(NF).LT.0)RETURN R1
      IF(K2+LENX(NF).GT.LREC)GOTO 170
      LAST=LENX(NF)
      DO 140 I=1,LAST
  140 X(I)=XIO(I+K2)
  150 CONTINUE
      LEN=LENX(NF)
      KEYX(NF)=KEY
      RETURN
  160 K1=K1+1
      K2=1
      GOTO 120
  170 LEN=LENX(NF)
      IS=1
      IE=LREC-K2
  175 DO 180 I=IS,IE
  180 X(I)=XIO(I+K2)
      IF(LEN-IE.LE.0)GOTO 150
      K1=K1+1
      CALL READMS(NF,XIO,LREC,K1)
      K1X(NF)=K1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      K2=1-IS
      GOTO 175
      ENTRY BWRITE
C-----CAN ONLY REWRITE RECORD JUST READ - NEEDS LEN
      IF(KEY.EQ.0.OR.NFX.NE.NF)RETURN R1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
```

C4-9

```
                    K1=K1+1
                    IF(K2+LENX(NF).GT.LREC)GOTO 300
                    LAST=LENX(NF)
                    DO 220 I=1,LAST
60          220 XIO(K2+I)=X(I)
                    CALL WRITMS(NF,XIO,LREC,K1,1,0)
                230 RETURN
                300 CONTINUE
                    LEN=LENX(NF)
65                  IS=1
                    IE=LREC-K2
                310 CONTINUE
                    CALL READMS(NF,XIO,LREC,K1)
                    DO 320 I=IS,IE
70          320 XIO(I+K2)=X(I)
                    CALL WRITMS(NF,XIO,LREC,K1,1,0)
                    IF(LEN-IE.LE.0)RETURN
                    IS=IE
                    IE=LEN
75                  IF(LEN-IS.GT.LREC)IE=LREC+IS
                    IS=IS+1
                    K1=K1+1
                    K2=1-IS
                    GOTO 310
80                  END
```

SYMBOLIC REFERENCE MAP (R=1)

0 ENTRY3P0BREAD 127 BWRITE

VARIABLES	SN	TYPE			RELOCATION					
244	I	INTEGER				246	IE	INTEGER		
245	IS	INTEGER				0	KEY	INTEGER		F.P.
2	KEYX	INTEGER	ARRAY	BREAD		241	K1	INTEGER		
52	K1X	INTEGER	ARRAY	BREAD		242	K2	INTEGER		
243	LAST	INTEGER				0	LEN	INTEGER		F.P.
26	LENX	INTEGER	ARRAY	BREAD		1	LREC	INTEGER		BREAD
0	NF	INTEGER		F.P.		0	NFX	INTEGER		BREAD
0	NIO	INTEGER	ARRAY	BREADIO		0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.		0	XIO	REAL	ARRAY	BREADIO

EXTERNALS	TYPE	ARGS		TYPE	ARGS
READMS		4	WRITMS		6

STATEMENT LABELS

34	120		42	130	0	140	
62	150		67	160	72	170	
77	175		0	180	0	220	
0	230	INACTIVE	167	300	174	310	
0	320						

C4-10

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
60	140	I	27 28	2B	INSTACK
104	180	I	39 40	2B	INSTACK
161	220	I	59 60	2B	INSTACK
203	320	I	69 70	2B	INSTACK

COMMON BLOCKS	LENGTH
BREADIO	2
BREAD	62

STATISTICS

PROGRAM LENGTH	255B	173
SCM LABELLED COMMON LENGTH	100B	64

100000B SCM USED

C4-11

```
      SUBROUTINE BLD(LEN,X,NF),RETURNS(R1)
      DIMENSION X(1),XIO(2),NIO(2)
      EQUIVALENCE (NIO,XIO)
C
      COMMON/BLDIO/XIO
      COMMON/BLD  /KPTR,LREC,IREC
C
      LREC1=LREC-1
      IF(KPTR.EQ.0.OR.NF.EQ.0)RETURN R1
      IF(LEN.LT.1)RETURN R1
      IF(LEN.GT.LREC1)GOTO 5
      JPTR=KPTR+LEN
      IF(JPTR.LE.LREC)GOTO 30
    5 IE=LREC-KPTR
   10 IS=1
      NIO(KPTR)=LEN
      IF(IE.LT.1)GOTO 21
   15 DO 20 I=IS,IE
   20 XIO(KPTR+I)=X(I)
   21 IF(LEN-IE.LE.0)GOTO 25
      CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      KPTR=1-IS
      GOTO 15
   25 KPTR=IE+1+KPTR
      IF(KPTR.GT.LREC)GOTO 26
      RETURN
   26 CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      KPTR=1
      RETURN
   30 NIO(KPTR)=LEN
      DO 40 I=1,LEN
   40 XIO(KPTR+I)=X(I)
      KPTR=JPTR+1
      IF(KPTR.GT.LREC)GOTO 26
      RETURN
      ENTRY FRC
      NIO(KPTR)=-2
      CALL WRITMS(NF,XIO,LREC,IREC,0)
      IREC=IREC+1
      KPTR=0
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

C4-12

ENTRY POINTS
3 BLD 113 FRC

VARIABLES	SN	TYPE		RELOCATION					
146	I	INTEGER			144	IE	INTEGER		
2	IREC	INTEGER		BLD	145	IS	INTEGER		
143	JPTR	INTEGER			0	KPTR	INTEGER		BLD
0	LEN	INTEGER		F.P.	1	LREC	INTEGER		BLD
142	LREC1	INTEGER			0	NF	INTEGER		F.P.
0	NIO	INTEGER	ARRAY	BLDIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BLDIO

EXTERNALS	TYPE	ARGS
WRITMS		5

STATEMENT LABELS

32	5	0	10	INACTIVE	37	15	
0	20	46	21		65	25	
72	26	100	30		0	40	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
44	20	I	18 19	2B	INSTACK
105	40	I	37 38	2B	INSTACK

COMMON BLOCKS	LENGTH
BLDIO	2
BLD	3

STATISTICS

PROGRAM LENGTH	147B	103
SCM LABELED COMMON LENGTH	5B	5

100000B SCM USED

C4-13

```
      SUBROUTINE IORDER(NUM,IX,Y,NY,LYDIM,AUX)
C ORDERS IN ASCENDING VALUE - X(LYDIM) AND Y(LYDIM,NY) ACCORDING TO
C     IX(NUM)
      DIMENSION IX(1),Y(LYDIM,1),AUX(1)
      LOGICAL AGAIN
C
C
    1 LAST=NUM-1
  100 AGAIN=.FALSE.
      DO 20 I=1,LAST
      IF(IX(I+1).GE.IX(I))GOTO 20
      IX1=IX(I)
      IX(I)=IX(I+1)
      IX(I+1)=IX1
    5 IF(NY.EQ.0)GOTO 19
      DO 6 J=1,NY
    6 AUX(J)=Y(I,J)
      DO 10 J=1,NY
   10 Y(I,J)=Y(I+1,J)
      DO 15 J=1,NY
   15 Y(I+1,J)=AUX(J)
   19 AGAIN=.TRUE.
   20 CONTINUE
      IF(AGAIN)GOTO 100
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 IORDER

VARIABLES	SN	TYPE		RELOCATION					
53	AGAIN	LOGICAL			0	AUX	REAL	ARRAY	F.P.
55	I	INTEGER			0	IX	INTEGER	ARRAY	F.P.
56	IX1	INTEGER			57	J	INTEGER		
54	LAST	INTEGER			0	LYDIM	INTEGER		F.P.
0	NUM	INTEGER		F.P.	0	NY	INTEGER		F.P.
0	Y	REAL	ARRAY	F.P.					

STATEMENT LABELS

0	1	INACTIVE	0	5	INACTIVE	0	6
0	10		0	15		45	19
46	20		14	100			

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
16	20	* I	10 23	31B	NOT INNER
26	6	J	16 17	2B	INSTACK
34	10	J	18 19	2B	INSTACK
43	15	J	20 21	2B	INSTACK

C4-14

STATISTICS
PROGRAM LENGTH 65B 53
100000B SCM USED

APPENDIX C.5 · C.5-1

```
      PROGRAM FPLOT3D(INPUT,OUTPUT,FILM,TEKP,TAPE11=100,TAPE13=100,
     1 TAPE8,TAPE9,TAPE5=INPUT,TAPE6=OUTPUT)
C  PLOTTING PROGRAM FOR FINITE ELEMENT
C     INT=0 SETS PROGRAM FOR INTERACTIVE USE ON 6000
C     INT=1 FOR DD FILE ON 7600, NON-INTERACTIVE
C
C
C-----------------------------------------------------------------
      DIMENSION XTITLE(8),YTITLE(8),TIME(20),TITLE(20)
      DIMENSION REALK(2),IDUM(2),IDOF(2),IUT(2)
C
      LOGICAL ELEM,INTRAC,OUT,CONTRS,IBAD,SHOSHR,LBLSPC,I2PASS
      LOGICAL RANGE,ZERO1,COSES
C
      EQUIVALENCE (IDOF,DOF),(DUM,IDUM)
      EQUIVALENCE (INTGRK,REALK),(UT,IUT)
C
      LEVEL 2,INTGRK,REALK
      LEVEL 2,IDOF,DOF,INUM,XYZ
      LEVEL 2,UT,IUT
      LEVEL 2,DUM,IDUM
      LEVEL 2,INDEX1,INDEX3
      LEVEL 2,IADR
C
      COMMON/RTIME/ FLOW,RLAX(20)
      COMMON/SIZE/NET,NDT,NETNEW,NDTNEW,LNODNW
      COMMON/IO/KR,KW,KP,KT1,KT2,KT3,OUT
      COMMON/BEGIN/ICON,IKOUNT,ILNZ,INASTR,IQ,IK
      COMMON/END/LCON,LKOUNT,LLNZ,LNASTR,LQ,LK
      COMMON/PROB/HASH1,OLDWRK,LENGTH,GRAV,GRAVS,GRAVB,GMALT,NFLT,NLAP,
     1 NDIM,DISP2
      COMMON/AXIS/IAXIS1,IAXIS2,IAXIS3,C1UNIT,C2UNIT,C3UNIT,
     1 D1MEAN,D2MEAN,D3MEAN
      COMMON/RANGE/RANGE,ZINC,XM(2),YM(2),ZM(2)
C
      COMMON/IFORMP/IARG,C1,C2,C3
      COMMON/FILES/INFILE,IOFIL,INTRAC,IERRC,NOW,DEBUG
      COMMON/QFORIO/CONTRS,IDVICE,IBAD,SHOSHR,LBLSPC,I2PASS
      COMMON/CONLEV/CONLEV(33)
      COMMON/BUCKY/IB(1024)
      COMMON/DPOINT/IADR(6000)
      COMMON/FACE  /IFACE(4,20)
      COMMON/TRANS /NPIC,TRANS(10,10)
      COMMON/PLOT3D/MAXRES,KDISP,INT,ELEM,NUM
      COMMON/K     /INTGRK(40000)
      COMMON/UT    /UT(50)
      COMMON/NUM   /INUM(1500)
      COMMON/DOF   /DOF(5000)
      COMMON/XYZ   /XYZ(10000)
C
      COMMON/BREAD /NFX,LRECBR,KEYX(20),LENX(20),K1X(20)
      COMMON/BREADIO/XIOBR(3256)
      COMMON/INDEX /INDEX1(1000),INDEX3(1000)
      COMMON/DUM   /DUM(5000)
C
```

C5-2

```
      C
            DATA KR/5/,KW/6/,KP/7/,KT1/11/,KT2/12/,KT3/13/,ELEM/.FALSE./
            DATA INFILE/5/,IOFIL/6/,INTRAC/.FALSE./,IERRC/0/,NOW/0/,DEBUG/0./
            DATA MAXRES/512/,KDISP/1/,INT/1/,NPIC/1/,TUNIT/3.16E7/,ALL/$ALL$/
 60         DATA NFX/0/,KEYX/20$0/,LENX/20$0/,K1X/20$0/,LRECBR/2048/
            DATA IAXIS1/1/,IAXIS2/2/,IAXIS3/3/,C1UNIT/1.E5/,C2UNIT/1.E5/,
           1 C3UNIT/1./,D1MEAN/0./,D2MEAN/0./,D3MEAN/0./
            DATA TRANS/10$(0.,0.,0.,0.,0.,0.,1.,1.,1.,45.)/
            DATA ((IFACE(I,J),I=1,4),J=1,8)/28$0,5,6,7,8/
 65         DATA OUT/.FALSE./,ZERO1/.FALSE./,COSES/.FALSE./
      C
            DATA XNDCMN/.225/,XNDCMX/.9/,YNDCMN/.1/,YNDCMX/.9/
            DATA WXMIN/0./,WXMAX/512./,WYMIN/0./,WYMAX/512./
            DATA CONTRS/.FALSE./,IDVICE/1/,IBAD/.FALSE./,SMOSHR/.TRUE./
 70         DATA LBLSPC/5/,I2PASS/.FALSE./,ZINC/1./,RANGE/.TRUE./
      C
      C
            NAMELIST/IOK/KR,KW,KP,KT1,KT2,KT3,OUT,INFILE,IOFIL
            NAMELIST/PLOT3D/MAXRES,KDISP,INT,TRANS,NPIC,IFACE,TUNIT,ELEM,
 75        1 XNDCMN,XNDCMX,YNDCMN,YNDCMX,WXMIN,WXMAX,WYMIN,WYMAX,
           2 CONTRS,IDVICE,SMOSHR,LBLSPC,ZINC,RANGE
            NAMELIST/AXIS/IAXIS1,IAXIS2,IAXIS3,C1UNIT,C2UNIT,C3UNIT,
           1 D1MEAN,D2MEAN,D3MEAN,ZERO1,COSES
      C
 80   C*****************************************************************************
      C
      C  OPEN WORKFILE
            CALL OPENMS(KT1,INDEX1,1000,0)
      C
 85   C  OPEN TIME FILE
            CALL OPENMS(KT3,INDEX3,1000,0)
            READ(5,IOK)
            READ(KR,PLOT3D)
      C
 90         WRITE(KW,10)
           10 FORMAT(1H1,10X,*PROGRAM FPLOT3D ---3D PLOTTING*,/,* INPUT DATA*)
            WRITE(KW,PLOT3D)
      C
            READ(KR,AXIS)
 95         WRITE(KW,AXIS)
      C
      C  READ INPUT DATA FOR PLOT
      C
            IF(ELEM)GOTO 1100
100         READ(KR,*)NTIME
            READ(KR,*)(TIME(I),I=1,NTIME)
         1100 CONTINUE
            READ(KR,20)XTITLE
            READ(KR,20)YTITLE
105        20 FORMAT(8A10)
      C
            IF(ELEM)GOTO 1300
            DO 1110 I=1,NTIME
         1110 TIME(I)=TIME(I)*TUNIT
110         WRITE(KW,30)NTIME
```

C5-3

```
      30 FORMAT(1H0,* TIMES IN PROBLEM UNITS FOR OUTPUT@*,I6,* TIMES*)
         WRITE(KW,31)(TIME(I),I=1,NTIME)
      31 FORMAT(1X,1P10E11.3)
         WRITE(KW,35)
115   35 FORMAT(1H0,10X,* 3D DISPLACEMENT PLOT*)
      C
      C  INPUT TIME FILE
      C
           KEYTMP=0
120        CALL BREAD(KEYTMP,LEN,TITLE,KT3),RETURNS(9000)
           KEYTMP=0
           CALL BREAD2(KEYTMP,LEN,DUM,KT3),RETURNS(9000)
           NDIM=IDUM(10)
           HASH2=DUM(1)
125        KEYTMP=0
           CALL BREAD(KEYTMP,LEN,NET,KT3),RETURNS(9000)
           KEYTMP=0
           CALL BREAD2(KEYTMP,LEN,DUM,KT3),RETURNS(9000)
           KEYTMP=0
130        CALL BREAD(KEYTMP,LEN,FLOW,KT3),RETURNS(9000)
           LT=LEN-1
           KEYTMP=0
           CALL BREAD2(KEYTMP,LEN,IDUM,KT3),RETURNS(9000)
           NFIL=LEN
135   C
           WRITE(KW,50)TITLE,FLOW,LT
      50 FORMAT(1H1,* PROBLEM@*,20A4,//,* PARAMETERS FOR SERIES@ FLOW=*,
        1 L2,5X ,*ORDER=*,I6,/,* RELAXATION TIMES@*)
           WRITE(KW,51)(RLAX(I),I=1,LT)
140   51 FORMAT(1X,1P10E11.3)
           WRITE(KW,52)
      52 FORMAT(1H0,* DOF IN TIME FILE@*)
           IF(DEBUG.NE.0.AND.OUT)WRITE(KW,56)(IDUM(I),I=1,LEN)
      C
145   C
      C  INPUT ELEMENT NUMBERS FOR DISPLACEMENT SOLUTION
      C
      1300 CONTINUE
           READ(KR,*)NLINES
150        IF(NUM.EQ.ALL)GOTO 1302
      C INPUT ELEMENT NUMBERS BY GROUPS
           NUM=0
           DO 1210 I=1,NLINES
           READ(KR,*)NIELM,NAELM
155        DO 1205 J=NIELM,NAELM
           NUM=NUM+1
           INUM(NUM)=J
      1205 CONTINUE
      1210 CONTINUE
160   C
           CALL IORDER3(NUM,INUM,INUM,0,NUM,DUM)
           GOTO 1310
      C
      1302 CONTINUE
165        DO 1305 I=1,NET
```

C5-4

```
             1305 INUM(I)=I
                  NUM=NET
             1310 CONTINUE
                C
170                WRITE(KW,55)
                55 FORMAT(1H0,* ELEMENTS USED IN PLOT*)
                   WRITE(KW,56)(INUM(I),I=1,NUM)
                56 FORMAT(1X,20I6)
                C
175             C  INPUT WORK FILE
                C
             1350 CONTINUE
                   KEYTMP=0
                   CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
180                KEYTMP=0
                   CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
                   KEYTMP=0
                   CALL BREAD(KEYTMP,LEN,NET,KT1),RETURNS(9000)
                   KEYTMP=0
185                CALL BREAD(KEYTMP,LEN,ICON,KT1),RETURNS(9000)
                   KEYTMP=0
                   CALL BREAD(KEYTMP,LEN,LCON,KT1),RETURNS(9000)
                   KEYTMP=0
                   CALL BREAD(KEYTMP,LEN,HASH1,KT1),RETURNS(9000)
190                IF(ELEM)HASH2=HASH1
                   IF(HASH1.NE.HASH2)GOTO 9300
                   KEYTMP=0
                   CALL BREAD2(KEYTMP,LEN,INTGRK,KT1),RETURNS(9000)
                   KEYRL1=KEYTMP
195                KEYTMP=0
                   CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
                C
                C  READ ELEMENTS
                C
200             C  X AXIS FOR PLOT IS IAXIS1
                C  Y AXIS FOR PLOT IS IAXIS2
                C  INOD - CONSTRUCT POLYGON STARTING AT LOCAL NODE INOD FOR
                C         ELEMENT WITH LNOD NODES
                C
205             C  MNOD - MAXIMUM NUMBER OF NODES IN POLYGON
                C
             2000 CONTINUE
                   NEXT=1
                   NDOF=0
210                INASTR1=INASTR-1
                C
                   DO 2700 L=1,NET
                   KEYTMP=0
                   CALL BREAD2(KEYTMP,LEN,DUM,KT1),RETURNS(9000)
215                LNUM=IDUM(1)
                   IF(INUM(NEXT).NE.LNUM)GOTO 2700
"STG223) OUTPUT  QUEUED PR 45 HO
                   NODES=IDUM(2)
                C
                   CALL COORD3D(XYZ,DUM,IDUM,NEXT,NUM,NDIM,NLOCAL)
220             C
```

CS-6

```
      C  FETCH DEGREES OF FREEDOM TO BE PLOTTED FOR DISPLACEMENT SOLUTION
      C
            IADDR=INTGRK(IMASTR1+LNUM)-1
      C
225         DO 2610 J=1,NLOCAL
            INOD=IFACE(J,NODES)
      C
            NDOF=NDOF+1
            IDOF(NDOF)=INTGRK(IADDR+IAXIS3+NDIM*(INOD-1))
230      2610 CONTINUE
      C
       2650 CONTINUE
            INUM(NUM+NEXT)=NODES
            INUM(2*NUM+NEXT)=NLOCAL
235         NEXT=NEXT+1
       2700 CONTINUE
            L=3*NUM
            IF(DEBUG.NE.0.AND.OUT)WRITE(KW,56)(INUM(I),I=1,L)
            NEXT=NEXT-1
240         IF(NEXT.NE.NUM)GOTO 9400
      C
      C  ORDER DOF USED IN PLOT OF DISPLACEMENTS
      C
            CALL IORDEL3(NDOF,IDOF,DUM,0,NDOF,DUM)
245         WRITE(KW,60)NDOF
         60 FORMAT(' DOF USED IN PLOT=',I6)
            WRITE(KW,56)(IDOF(I),I=1,NDOF)
      C
      C  FETCH DOF
250         KEYTMP=0
            CALL BREAD2(KEYTMP,LEN,DUM,KT3),RETURNS(9000)
            NEXT=1
      C
      C  FETCH DOF FOR DISPLACEMENT SOLUTION
255   C
            DO 3200 J=1,NFIL
            KEYTMP=0
            CALL BREAD2(KEYTMP,LEN,UT,KT3),RETURNS(9000)
            IF(IUT(1).NE.IDOF(NEXT))GOTO 3200
260   C
            CALL CALT2(UT(3),LT,DUM,TIME,NTIME)
            ZS=0.
            DO 3150 I=1,NTIME
            II=NDOF*I+NEXT
265         DOF(II)=(DUM(I)-D3MEAN)/C3UNIT-ZS
            IF(COSES.AND.I.EQ.1)ZS=DOF(II)
            IF(ZERO1.AND.I.EQ.1.AND.COSES)DOF(II)=0.
       3150 CONTINUE
            NEXT=NEXT+1
270   C
       3200 CONTINUE
            NEXT=NEXT-1
            IF(NEXT.NE.NDOF)GO TO 9500
      C
275   C  ARRANGE ADDRESS VECTOR FOR DOF
```

C5-6

```
      C
            DO 3300 I=1,NDOF
            IA=IDOF(I)
            IADR(IA)=I
 280   3300 CONTINUE
      C
       3500 CONTINUE
      C
      C  SET UP PLOT
 285  C
            IF(INT.NE.0)GOTO 3550
            CALL CONNECT(4LFILM)
            CALL CONNECT(INFILE)
            CALL CONNECT(IOFIL)
 290        CALL TVOPEN(2HTK,4LFILM)
            CALL TVOPEN(2HDD,4LTEKP)
       3550 CONTINUE
      C
            CALL INFREE(MAXRES)
 295  C  IF NOT INTERACTIVE, INT=1
            IARG=INT
            CALL CINTER
            CALL TVINIT
      C
 300        CALL TVSET(5HWXMIN,WXMIN)
            CALL TVSET(5HWXMAX,WXMAX)
            CALL TVSET(5HWYMIN,WYMIN)
            CALL TVSET(5HWYMAX,WYMAX)
      C
 305        CALL TVSET(6HXNDCMN,XNDCMN)
            CALL TVSET(6HXNDCMX,XNDCMX)
            CALL TVSET(6HYNDCMN,YNDCMN)
            CALL TVSET(6HYNDCMX,YNDCMX)
      C
 310  C
            IF(.NOT.ELEM)GOTO 4520
            NTIME=1
      C
       4520 CONTINUE
 315  C
      C FIND RANGE OF VALUES FOR FPATCH IF NEEDED
      C
            IF(RANGE)CALL RANGXYZ(XYZ,INUM,DOF,NUM,NDOF,NTIME)
      C
 320  C  PLOT SUBROUTINE
      C
            DO 5000 I=1,NTIME
            IARG=INT
            CALL ZPLT3D(DOF(NDOF*I+1),NDIM,NDOF)
 325    5000 CONTINUE
      C
      C  END OF PLOT PROGRAM
      C
            WRITE(KW,99)
 330      99 FORMAT(' END OF PLOT PROGRAM')
```

C5-7

```
                IF(CONTRS)WRITE(KW,80)(CONLEV(I),I=1,33)
             80 FORMAT(1X,1P10E12.3)
                IF(OUT)WRITE(KW,PLOT3D)
                CALL TVEND
    335         STOP
              C
             9000 WRITE(KW,101)
              101 FORMAT(* ERROR IN READ/WRITE*)
                GOTO 9900
    340       9300 WRITE(KW,110)HASH1,HASH2
              110 FORMAT(* HASH1,HASH2=*,2A10)
                GOTO 9900
             9400 WRITE(KW,120)NEXT,NUM,LNUM,INUM(NEXT),INUM(NEXT+1)
              120 FORMAT(* ERROR IN ELEMENTS*,5I10)
    345         GOTO 9900
             9500 WRITE(KW,130)NEXT,NDOF,NFIL,IUT(1),IDOF(NEXT),IDOF(NEXT+1)
              130 FORMAT(* ERROR IN DOF*,6I10)
                GOTO 9900
             9900 CALL XEQCCS($/DUMP,0./GRUMP,MUV=10,SBX.*//*)
    350         STOP
                END
```

CARD NR. SEVERITY DETAILS DIAGNOSIS OF PROBLEM

123 I IDUM ARRAY REFERENCE OUTSIDE DIMENSION BOUNDS.

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3777 FPLOT3D

VARIABLES	SN	TYPE		RELOCATION					
4774	ALL	REAL			0	CONLEV	REAL	ARRAY	CONLEV
0	CONTRS	LOGICAL		QFORIO	4776	COSES	LOGICAL		
1	C1	REAL		IFORMP	3	C1UNIT	REAL		AXIS
2	C2	REAL		IFORMP	4	C2UNIT	REAL		AXIS
3	C3	REAL		IFORMP	5	C3UNIT	REAL		AXIS
5	DEBUG	REAL		FILES	12	DISP2	REAL		PROB
0	DOF	REAL	ARRAY	DOF	0	DUM	REAL	ARRAY	DUM
6	D1MEAN	REAL		AXIS	7	D2MEAN	REAL		AXIS
10	D3MEAN	REAL		AXIS	3	ELEM	LOGICAL		PLOT3D
0	FLOW	REAL		RTIME	6	GMALT	REAL		PROB
3	GRAV	REAL		PROB	5	GRAVB	REAL		PROB
4	GRAVS	REAL		PROB	0	HASH1	REAL		PROB
5470	HASH2	REAL			5465	I	INTEGER		
5513	IA	INTEGER			5507	IADDR	INTEGER		
0	IADR	INTEGER	ARRAY	DPOINT	0	IARG	INTEGER		IFORMP
0	IAXIS1	INTEGER		AXIS	1	IAXIS2	INTEGER		AXIS

CS-8

VARIABLES	SN	TYPE		RELOCATION					
2	IAXIS3	INTEGER		AXIS	0	IB	INTEGER	ARRAY	BUCKY
2	IBAD	LOGICAL		QFORIO	0	ICON	INTEGER		BEGIN
0	IDOF	INTEGER	ARRAY	DOF	0	IDUM	INTEGER	ARRAY	DUM
1	IDVICE	INTEGER		QFORIO	3	IERRC	INTEGER		FILES
0	IFACE	INTEGER	ARRAY	FACE	5512	II	INTEGER		
5	IK	INTEGER		BEGIN	1	IKOUNT	INTEGER		BEGIN
2	ILNZ	INTEGER		BEGIN	3	IMASTR	INTEGER		BEGIN
5502	IMASTR1	INTEGER			0	INDEX1	INTEGER	ARRAY	INDEX
1750	INDEX3	INTEGER	ARRAY	INDEX	0	INFILE	INTEGER		FILES
5510	INOD	INTEGER			2	INT	INTEGER		PLOT3D
0	INTGRK	INTEGER	ARRAY	K	2	INTRAC	LOGICAL		FILES
0	INUM	INTEGER	ARRAY	NUM	1	IOFIL	INTEGER		FILES
4	IQ	INTEGER		BEGIN	0	IUT	INTEGER	ARRAY	UT
5	I2PASS	LOGICAL		QFORIO	5476	J	INTEGER		
1	KDISP	INTEGER		PLOT3D	5477	KEYRL1	INTEGER		
5466	KEYTMP	INTEGER			2	KEYX	INTEGER	ARRAY	BREAD
2	KP	INTEGER		IO	0	KR	INTEGER		IO
3	KT1	INTEGER		IO	4	KT2	INTEGER		IO
5	KT3	INTEGER		IO	1	KW	INTEGER		IO
52	K1X	INTEGER	ARRAY	BREAD	5503	L	INTEGER		
4	LBLSPC	LOGICAL		QFORIO	0	LCON	INTEGER		END
5467	LEN	INTEGER			2	LENGTH	INTEGER		PROB
26	LENX	INTEGER	ARRAY	BREAD	5	LK	INTEGER		END
1	LKOUNT	INTEGER		END	2	LLNZ	INTEGER		END
3	LMASTR	INTEGER		END	4	LNODNW	INTEGER		SIZE
5504	LNUM	INTEGER			4	LQ	INTEGER		END
1	LRECBR	INTEGER		BREAD	5471	LT	INTEGER		
5475	MAELM	INTEGER			0	MAXRES	INTEGER		PLOT3D
5474	MIELM	INTEGER			11	NDIM	INTEGER		PROB
5501	NDOF	INTEGER			1	NDT	INTEGER		SIZE
3	NDTNEW	INTEGER		SIZE	0	NET	INTEGER		SIZE
2	NETNEW	INTEGER		SIZE	5500	NEXT	INTEGER		
5472	NFIL	INTEGER			7	NFLT	INTEGER		PROB
0	NFX	INTEGER		BREAD	10	NLAP	INTEGER		PROB
5473	NLINES	INTEGER			5506	NLOCAL	INTEGER		
5505	NODES	INTEGER			4	NOW	INTEGER		FILES
0	NPIC	INTEGER		TRANS	5464	NTIME	INTEGER		
4	NUM	INTEGER		PLOT3D	1	OLDWRK	REAL		PROB
6	OUT	LOGICAL		IO	0	RANGE	LOGICAL		RANGE
0	REALK	REAL	ARRAY	K	1	RLAX	REAL	ARRAY	RTIME
3	SHOSHR	LOGICAL		QFORIO	5534	TIME	REAL	ARRAY	
5560	TITLE	REAL	ARRAY		1	TRANS	REAL	ARRAY	TRANS
4773	TUNIT	REAL			0	UT	REAL	ARRAY	UT
5004	WXMAX	REAL			5003	WXMIN	REAL		
5006	WYMAX	REAL			5005	WYMIN	REAL		
0	XIOBR	REAL	ARRAY	BREADIO	2	XM	REAL	ARRAY	RANGE
4777	XNDCMN	REAL			5000	XNDCMX	REAL		
5514	XTITLE	REAL	ARRAY		0	XYZ	REAL	ARRAY	XYZ
4	YM	REAL	ARRAY	RANGE	5001	YNDCMN	REAL		
5002	YNDCMX	REAL			5524	YTITLE	REAL	ARRAY	
4775	ZERO1	LOGICAL			1	ZINC	REAL		RANGE
6	ZM	REAL	ARRAY	RANGE	5511	ZS	REAL		

C5-9

FILE NAMES MODE

1112	FILM		0	INPUT		445	OUTPUT		2224	TAPE11	
2435	TAPE13		0	TAPE5	NAME	445	TAPE6		2646	TAPE8	
3313	TAPE9		1557	TEKP							

EXTERNALS TYPE ARGS

BREAD	4	BREAD2	4
CALT2	5	CINTER	0
CONNECT	1	COORD3D	7
INFREE	1	IORDEL3	6
IORDER3	6	OPENMS	4
RANGXYZ	6	TVEND	0
TVINIT	0	TVOPEN	2
TVSET	2	XEQCCS	1
ZPLT3D	3		

NAMELISTS

AXIS IOK PLOT3D

STATEMENT LABELS

5156	10	FMT	5221	20	FMT		5227	30	FMT		
5242	31	FMT	5250	35	FMT		5263	50	FMT		
5302	51	FMT	5310	52	FMT		5334	55	FMT		
5345	56	FMT	5357	60	FMT		5402	80	FMT		
5372	99	FMT	5413	101	FMT		5424	110	FMT		
5440	120	FMT	5455	130	FMT		4031	1100			
0	1110		0	1205			0	1210			
4126	1300		4153	1302			0	1305			
4162	1310		0	1350	INACTIVE		0	2000	INACTIVE		
0	2610		0	2650	INACTIVE		4272	2700			
0	3150		4366	3200			0	3300			
0	3500	INACTIVE	4414	3550			4445	4520			
0	5000		4474	9000			4477	9300			
4502	9400		4512	9500			4521	9900			

LOOPS

LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
1110	I	108 109	2B	INSTACK
1210	I	153 159	15B	EXT REFS NOT INNER
1205	J	155 158	3B	INSTACK
1305	I	165 166	4B	INSTACK
2700	L	212 236	41B	EXT REFS EXITS NOT INNER
2610	J	225 230	4B	INSTACK
3200	J	256 271	37B	EXT REFS EXITS NOT INNER
3150	I	263 268	14B	OPT
3300	I	277 280	5B	INSTACK
5000	I	322 325	11B	EXT REFS

COMMON BLOCKS LENGTH

RTIME	21
SIZE	5
IO	7
BEGIN	6
END	6
PROB	11
AXIS	9

COMMON BLOCKS	LENGTH	
RANGE	8	
IFORMP	4	
FILES	6	
QFORIO	6	
CONLEV	33	
BUCKY	1024	
DPOINT	6000	LCM
FACE	80	
TRANS	101	
PLOT3D	5	
K	40000	LCM
UT	50	LCM
NUM	1500	LCM
DOF	5000	LCM
XYZ	10000	LCM
BREAD	62	
BREADIO	3256	
INDEX	2000	LCM
DUM	5000	LCM

STATISTICS

PROGRAM LENGTH	2444B	1316
BUFFER LENGTH	3176B	1662
SCM LABELED COMMON LENGTH	11052B	4650
LCM LABELED COMMON LENGTH	207656B	69550

70000B SCM USED

(BOTTOM OF FILE)

CS-11

```
      SUBROUTINE COORD3D(XYZ,DATA,IDATA,N,NUM,NDIM,NLOCAL)
C  FETCHS COORDINATES FOR ELEMENT NODES INCLUDED IN PLOT
C   SETS UP COORDINATE ARRAY XYZ
C
C  NUM = TOTAL NUMER OF ELEMENT IN PLOT
C  XYZ(N,I,J)  COORDINATE ARRAY WHERE
C     N = ELEMENT INDEX WITH DIMENSION EQUAL TO NUM
C     I = COORDINATE AXES WHERE 1 = X AXIS
C                               2 = Y AXIS
C                               3 = Z AXIS IN PLOT
C         DIMENSIONED NDIM (USUALLY 3)
C     J = NODE NUMBER IN ELEMENT FOR PLOTTED FACE
C         IN LOCAL NUMBER FRAME RATHER THAN GLOBAL
C
C  DATA(1)     DATA ARRAY FROM BREAD WITH COORDINATES FOR
C                ELEMENT IN SEQUENTIAL ORDER OF LOCAL NUMBERING
C
C  NLOCAL    NUMBER OF NODES IN ELEMENT USED FOR PLOT
C
C  COMMON DATA ----------------------------------------
C
C  FACE  TO PLOT ONLY ONE FACE OF ELEMENT MUST SPECIFY LOCAL NODES
C       IFACE(NOD,NODES)  ASSUMES NUMBER OF NODES IN ELEMENT
C          ENOUGH TO SPECIFY LOCAL NUMBERS NEED.  NOD SPECIFIES
C          NUMBER OF LOCAL NODES IN FACE TO BE PLOTTED.
C
C  AXIS  DETERMINES AXES IN DATA USED FOR PLOT AND SCALING
C        Z AXIS (IAXIS3) BECOMES SURFACE.
C     IAXIS1@  COORDINATE DIRECTION FOR X AXIS (1,2, OR 3)
C     IAXIS2#  DIRECTION FOR Y AXIS
C     IAXIS3@  DIRECTION FOR Z AXIS
C
C-----------------------------------------------------------
C
      DIMENSION XYZ(NUM,NDIM,1), DATA(1),IDATA(2)
C
C
      LEVEL 2, XYZ,DATA,IDATA
C
      COMMON/AXIS/IAXIS1,IAXIS2,IAXIS3,C1UNIT,C2UNIT,C3UNIT,
     1 D1MEAN,D2MEAN,D3MEAN
C
      COMMON/FACE/IFACE(4,1)
      COMMON/IO  /KR,KW,KP,KT1,KT2,KT3
C
      DATA NMAX/4/
C
C
C-----------------------------------------------------------
C
      NODES=IDATA(2)
      NOD=1
C
      DO 2100 J=1,NMAX
      INOD=IFACE(NOD,NODES)
```

C5-12

```
                IF(INOD.EQ.0)GOTO 2200
            C
                JJ=2+NDIM*(INOD-1)
                DO 2000 I=1,NDIM
 60             JJ=JJ+1
                IF(I.EQ.IAXIS1) XYZ(N,1,NOD)=(DATA(JJ)-D1MEAN)/C1UNIT
                IF(I.EQ.IAXIS2) XYZ(N,2,NOD)=(DATA(JJ)-D2MEAN)/C2UNIT
                IF(I.EQ.IAXIS3) XYZ(N,3,NOD)=(DATA(JJ)-D3MEAN)/C3UNIT
             2000 CONTINUE
 65             NOD=NOD+1
                GOTO 2100
             2100 CONTINUE
             2200 CONTINUE
                NLOCAL=NOD-1
 70         C
                RETURN
                END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 COORD3D

VARIABLES	SN	TYPE		RELOCATION					
3	C1UNIT	REAL		AXIS	4	C2UNIT	REAL		AXIS
5	C3UNIT	REAL		AXIS	0	DATA	REAL	ARRAY	F.P.
6	D1MEAN	REAL		AXIS	7	D2MEAN	REAL		AXIS
10	D3MEAN	REAL		AXIS	101	I	INTEGER		
0	IAXIS1	INTEGER		AXIS	1	IAXIS2	INTEGER		AXIS
2	IAXIS3	INTEGER		AXIS	0	IDATA	INTEGER	ARRAY	F.P.
0	IFACE	INTEGER	ARRAY	FACE	77	INOD	INTEGER		
76	J	INTEGER			100	JJ	INTEGER		
2	KP	INTEGER		IO	0	KR	INTEGER		IO
3	KT1	INTEGER		IO	4	KT2	INTEGER		IO
5	KT3	INTEGER		IO	1	KW	INTEGER		IO
0	N	INTEGER		F.P.	0	NDIM	INTEGER		F.P.
0	NLOCAL	INTEGER		F.P.	73	NMAX	INTEGER		
75	NOD	INTEGER			74	NODES	INTEGER		
0	NUM	INTEGER		F.P.	0	XYZ	REAL	ARRAY	F.P.

STATEMENT LABELS
0 2000 0 2100 63 2200

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
22	2100	* J	54 67	41B	EXITS NOT INNER
40	2000	I	59 64	16B	OPT

COMMON BLOCKS	LENGTH
AXIS	9
FACE	4
IO	6

C5-13

```
STATISTICS
  PROGRAM LENGTH                1048      68
  SCM LABELED COMMON LENGTH       23B      19
        70000B SCM USED
```

CS-14

```
      SUBROUTINE BREAD(KEY,LEN,X,NF),RETURNS(R1)
C-----SEQUENTIAL (KEY=0)/UPDATING TO IO FILE HANDLER
C-----MODIFIED AND REINPUT 4/18/74
C-----OPENMS(NF,INDEX,LENGTH OF INDEX,0) MUST BE INSERTED IN MAIN
      DIMENSION X(1),XIO(2)
      DIMENSION NIO(2)
      EQUIVALENCE (XIO,NIO)
C
      COMMON/BREADIO/XIO
      COMMON/BREAD  /NFX,LREC,KEYX(20),LENX(20),K1X(20)
C
      IF(NF.GT.20)RETURN R1
      IF(KEY.EQ.0)KEY=KEYX(NF)+LENX(NF)+1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
      K1=K1+1
      KEY=LREC*(K1-1)+K2
      IF(K1.EQ.K1X(NF).AND.NF.EQ.NFX)GOTO 130
  120 CALL READMS(NF,XIO,LREC,K1)
      NFX=NF
      K1X(NF)=K1
  130 LENX(NF)=NIO(K2)
      IF(LENX(NF).EQ.-1)GOTO 160
      IF(LENX(NF).LT.0)RETURN R1
      IF(K2+LENX(NF).GT.LREC)GOTO 170
      LAST=LENX(NF)
      DO 140 I=1,LAST
  140 X(I)=XIO(I+K2)
  150 CONTINUE
      LEN=LENX(NF)
      KEYX(NF)=KEY
      RETURN
  160 K1=K1+1
      K2=1
      GOTO 120
  170 LEN=LENX(NF)
      IS=1
      IE=LREC-K2
  175 DO 180 I=IS,IE
  180 X(I)=XIO(I+K2)
      IF(LEN-IE.LE.0)GOTO 150
      K1=K1+1
      CALL READMS(NF,XIO,LREC,K1)
      K1X(NF)=K1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      K2=1-IS
      GOTO 175
      ENTRY BWRITE
C-----CAN ONLY REWRITE RECORD JUST READ - NEEDS LEN
      IF(KEY.EQ.0.OR.NFX.NE.NF)RETURN R1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
```

CS-15

```
                    K1=K1+1
                    IF(K2+LENX(NF).GT.LREC)GOTO 300
                    LAST=LENX(NF)
                    DO 220 I=1,LAST
60                220 XIO(K2+I)=X(I)
                    CALL WRITMS(NF,XIO,LREC,K1,1,0)
                230 RETURN
                300 CONTINUE
                    LEN=LENX(NF)
65                  IS=1
                    IE=LREC-K2
                310 CONTINUE
                    CALL READMS(NF,XIO,LREC,K1)
                    DO 320 I=IS,IE
70                320 XIO(I+K2)=X(I)
                    CALL WRITMS(NF,XIO,LREC,K1,1,0)
                    IF(LEN-IE.LE.0)RETURN
                    IS=IE
                    IE=LEN
75                  IF(LEN-IS.GT.LREC)IE=LREC+IS
                    IS=IS+1
                    K1=K1+1
                    K2=1-IS
                    GOTO 310
80                  END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3	BREAD	127	BWRITE

VARIABLES	SN	TYPE								
244	I	INTEGER			246	IE	INTEGER		
245	IS	INTEGER			0	KEY	INTEGER		F.P.
2	KEYX	INTEGER	ARRAY	BREAD	241	K1	INTEGER		
52	K1X	INTEGER	ARRAY	BREAD	242	K2	INTEGER		
243	LAST	INTEGER			0	LEN	INTEGER		F.P.
26	LENX	INTEGER	ARRAY	BREAD	1	LREC	INTEGER		BREAD
0	NF	INTEGER		F.P.	0	NFX	INTEGER		BREAD
0	NIO	INTEGER	ARRAY	BREADIO	0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.	0	XIO	REAL	ARRAY	BREADIO

EXTERNALS	TYPE	ARGS		
READMS		4	WRITMS	6

STATEMENT LABELS

34	120		42	130		0	140		
62	150		67	160		72	170		
77	175		0	180		0	220		
0	230	INACTIVE	167	300		174	310		
0	320								

CS-16

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
60	140	I	27 28	2B	INSTACK
104	180	I	39 40	2B	INSTACK
161	220	I	59 60	2B	INSTACK
203	320	I	69 70	2B	INSTACK

COMMON BLOCKS	LENGTH
BREADIO	2
BREAD	62

STATISTICS

PROGRAM LENGTH	255B	173
SCM LABELED COMMON LENGTH	100B	64

70000B SCM USED

CS-17

```
      SUBROUTINE BREAD2(KEY,LEN,X,NF),RETURNS(R1)
C-----SEQUENTIAL (KEY=0)/UPDATING TO IO FILE HANDLER
C-----MODIFIED AND REINPUT 4/18/74
C-----OPENMS(NF,INDEX,LENGTH OF INDEX,0) MUST BE INSERTED IN MAIN
      DIMENSION X(1),XIO(2)
      LEVEL 2,X
      DIMENSION NIO(2)
      EQUIVALENCE (XIO,NIO)
C
C
      COMMON/BREADIO/XIO
C
      COMMON/BREAD/NFX,LREC,KEYX(20),LENX(20),K1X(20)
C
      IF(NF.GT.20)RETURN R1
      IF(KEY.EQ.0)KEY=KEYX(NF)+LENX(NF)+1
      K1=(KEY-1)/LREC
      K2=KEY-LREC*K1
      K1=K1+1
      KEY=LREC*(K1-1)+K2
      IF(K1.EQ.K1X(NF).AND.NF.EQ.NFX)GOTO 130
  120 CALL READMS(NF,XIO,LREC,K1)
      NFX=NF
      K1X(NF)=K1
  130 LENX(NF)=NIO(K2)
      IF(LENX(NF).EQ.-1)GOTO 160
      IF(LENX(NF).LT.0)RETURN R1
      IF(K2+LENX(NF).GT.LREC)GOTO 170
      LAST=LENX(NF)
      DO 140 I=1,LAST
  140 X(I)=XIO(I+K2)
  150 CONTINUE
      LEN=LENX(NF)
      KEYX(NF)=KEY
      RETURN
  160 K1=K1+1
      K2=1
      GOTO 120
  170 LEN=LENX(NF)
      IS=1
      IE=LREC-K2
  175 DO 180 I=IS,IE
  180 X(I)=XIO(I+K2)
      IF(LEN-IE.LE.0)GOTO 150
      K1=K1+1
      CALL READMS(NF,XIO,LREC,K1)
      K1X(NF)=K1
      IS=IE
      IE=LEN
      IF(LEN-IS.GT.LREC)IE=LREC+IS
      IS=IS+1
      K2=1-IS
      GOTO 175
      ENTRY BWRITE2
C-----CAN ONLY REWRITE RECORD JUST READ - NEEDS LEN
```

CS-18

```
                IF(KEY.EQ.0.OR.NFX.NE.NF)RETURN R1
                K1=(KEY-1)/LREC
                K2=KEY-LREC*K1
                K1=K1+1
   60           IF(K2+LENX(NF).GT.LREC)GOTO 300
                LAST=LENX(NF)
                DO 220 I=1,LAST
            220 XIO(K2+I)=X(I)
                CALL WRITMS(NF,XIO,LREC,K1,1,0)
   65       230 RETURN
            300 CONTINUE
                LEN=LENX(NF)
                IS=1
                IE=LREC-K2
   70       310 CONTINUE
                CALL READMS(NF,XIO,LREC,K1)
                DO 320 I=IS,IE
            320 XIO(I+K2)=X(I)
                CALL WRITMS(NF,XIO,LREC,K1,1,0)
   75           IF(LEN-IE.LE.0)RETURN
                IS=IE
                IE=LEN
                IF(LEN-IS.GT.LREC)IE=LREC+IS
                IS=IS+1
   80           K1=K1+1
                K2=1-IS
                GOTO 310
                END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 BREAD2 126 BWRITE2

VARIABLES	SN	TYPE			RELOCATION					
244	I	INTEGER				246	IE	INTEGER		
245	IS	INTEGER				0	KEY	INTEGER		F.P.
2	KEYX	INTEGER	ARRAY	BREAD		241	K1	INTEGER		
52	K1X	INTEGER	ARRAY	BREAD		242	K2	INTEGER		
243	LAST	INTEGER				0	LEN	INTEGER		F.P.
26	LENX	INTEGER	ARRAY	BREAD		1	LREC	INTEGER		BREAD
0	NF	INTEGER		F.P.		0	NFX	INTEGER		BREAD
0	NIO	INTEGER	ARRAY	BREADIO		0	R1	RETURNS		
0	X	REAL	ARRAY	F.P.		0	XIO	REAL	ARRAY	BREADIO

EXTERNALS	TYPE	ARGS			
READMS		4	WRITMS		6

STATEMENT LABELS

34	120	42	130	0	140		
62	150	67	160	72	170		

C5-19

STATEMENT LABELS

77	175			0	180	0	220
0	230	INACTIVE		166	300	173	310
0	320						

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
60	140	I	30 31	2B	INSTACK
104	180	I	42 43	2B	INSTACK
160	220	I	62 63	2B	INSTACK
203	320	I	72 73	2B	INSTACK

COMMON BLOCKS	LENGTH
BREADIO	2
BREAD	62

STATISTICS

PROGRAM LENGTH	254B	172
SCM LABELED COMMON LENGTH	100B	64

70000B SCM USED

C5-20

```
      SUBROUTINE CALT2(SC,LT,Y,TIME,NTIME)
C  CALCULATES Y FROM TIME SERIES SC
C     SC(1) - S0
C     SC(2) - S1 ....
C  FOR SERIES
C     Y(NTIME)=S0+S1*(1-EXP(-TIME(NTIME)/RLAX(1))+S2*(---)...
C  WHERE
C     RLAX - RELAXATION TIMES FOR FIT (LENGTH LT)
C     LT   - ORDER OF APPROXIMATION
C     FLOW - .TRUE. IF CONSTANT FLOW TERM INCLUDED IN SERIES
C     TIME - NTIME VALUES FOR COMPUTING
C
      DIMENSION SC(2),Y(2),TIME(2)
      LOGICAL FLOW
C
      LEVEL 2,Y,SC
C
      COMMON/RTIME/FLOW,RLAX(20)
C
C*******************************************************************************
C
      IX=1
      IF(FLOW)IX=2
      DO 2000 I=1,NTIME
      Y(I)=SC(1)
      IF(LT.LE.0)GOTO 1600
      DO 1500 J=1,LT
      X=-TIME(I)/RLAX(J)
      IF(X.GT.-170.)GOTO 1400
      X=0.
      GOTO 1450
 1400 X=EXP(X)
 1450 CONTINUE
 1500 Y(I)=SC(IX+J)*(1.-X)+Y(I)
 1600 CONTINUE
      IF(FLOW)Y(I)=Y(1)+SC(2)*TIME(I)
 2000 CONTINUE
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
   3 CALT2

VARIABLES	SN	TYPE		RELOCATION					
0	FLOW	LOGICAL		RTIME	70	I	INTEGER		
67	IX	INTEGER			71	J	INTEGER		
0	LT	INTEGER		F.P.	0	NTIME	INTEGER		F.P.
1	RLAX	REAL	ARRAY	RTIME	0	SC	REAL	ARRAY	F.P.
0	TIME	REAL	ARRAY	F.P.	72	X	REAL		

CS-21

```
VARIABLES     SN  TYPE          RELOCATION
    0  Y          REAL        ARRAY     F.P.

EXTERNALS           TYPE   ARGS
       EXP          REAL      1 LIBRARY

STATEMENT LABELS
   34  1400                              36  1450                              0  1500
   47  1600                               0  2000

LOOPS  LABEL    INDEX      FROM-TO    LENGTH    PROPERTIES
   16  2000   * I            24 37       42B             EXT REFS  NOT INNER
   26  1500   * J            27 34       21B             EXT REFS

COMMON BLOCKS   LENGTH
       RTIME        21                    .

STATISTICS
  PROGRAM LENGTH                  102B      66
  SCM LABELED COMMON LENGTH        25B      21
        70000B SCM USED
```

C5-22

```
1               SUBROUTINE FPATCH
              C 12SEP73                                                      3
              C THIS SUBROUTINE WILL PLOT A THREE DIMENSIONAL SURFACE        4
              C ACCORDING TO THE MESH WHICH IS PASSED OVER TO THIS           5
5             C ROUTINE. THIS PROGRAM IS INTERACTIVE IN THE SENSE THAT       6
              C THE USER CAN SCALE, ROTATE, AND TRANSLATE THE SURFACE. HIDDEN 7
              C SURFACE REMOVAL IS PERFORMED BY THE CODE AND THE PICTURE     8
              C WILL BE DISPLAYED ON THE TMDS, TELETYPE, OR WRITTEN TO       9
              C DISK AS A SHADED PICTURE. THIS PROGRAM WAS WRITTEN AND        10
10            C DOCUMENTED BY MICHAEL ARCHULETA, COMPUTER GRAPHICS SECTION,  11
              C LAWRENCE LIVERMORE LABORATORY.                               12
                   LOGICAL IBAD,IGOMOR,NTRNL,RANGE
              C
                   COMMON/XYZ/ELM(2)
15                 COMMON/NUM/IELM(2)
                   COMMON/PLOT3D/MAXR,KDISP,INT,ELEM,NPOLY
              C
                   COMMON/FILES/INFILE,IDTFIL,INTRAC,IERRC,NOW,DEBUG             14
                   COMMON/RANGE/RANGE,ZINC,XM(2),YM(2),ZM(2)
20                 COMMON/SEYES/OX,OY,OZ,ISURF                                   15
                   COMMON/INTENS/IHI,ILO,IBACK,IFX,IFY                           16
                   COMMON/POLDAT/IEDG,NTRNL,X(12),Y(12),Z(12),S(12),KOL(12),CON(12), 17
                  1              IS(12)                                         18
                   COMMON/QFORID/CONTRS,IDVICE,IBAD,SHOSHR,LBLSPC,I2PASS         PR16MAY.1
25            C      I2PASS=.FALSE. - ONLY HORIZONTAL SCANNING                  PR16MAY.2
                   COMMON/ZRANGE/ZMIN,ZMAX                                       20
              C MATRIX A CONTAINS THE CURRENT UPDATED TRANSFORMATION             21
              C MATRIX B CONTAINS THE REQUESTED TRANSFORMATION                   22
                   COMMON/QQQMAT/A(3,3),B(3,3),ALPHA,IGOMOR                      23
30                 COMMON/JUMPY/JUMP1,JUMP2,NTRACT                               24
              C FPATCH IS THE ENTRY POINT WHERE A USER DEFINED SURFACE            25
              C IS PROCESSED AND DISPLAYED. GET THE XYZ MINS AND MAXS.           26
              C
                   LEVEL 2,ELM,IELM
35            C
              C STATEMENT FUNCTION TO INDEX XYZ AND INUM
              C
                   INUM(L,1)=IELM(L+NPOLY*(1-1))
                   XYZ(L,1,J)=ELM(L+NPOLY*(I-1)+3*NPOLY*(J-1))
40            C
              C-----------------------------------------------
              C
                   IF(DEBUG .NE. 0) WRITE(IDTFIL,5)                              28
                 5 FORMAT(1X,*FPATCH CALLED*)                                    29
45                 IF(RANGE)GOTO 11
                   X(1)=Y(1)=Z(1)=-1E30                                          30
                   X(2)=Y(2)=Z(2)=+1E30                                          31
                   DO 10 I=1,NPOLY                                               32
                   NLOCAL=INUM(I,3)
50                 DO 9 J=1,NLOCAL
                     IF(XYZ(1,1,J).GT.X(1)) X(1)=XYZ(I,1,J)                        34
                     IF(XYZ(I,1,J).LT.X(2)) X(2)=XYZ(I,1,J)                        35
                     IF(XYZ(I,2,J).GT.Y(1)) Y(1)=XYZ(I,2,J)                        36
                     IF(XYZ(I,2,J).LT.Y(2)) Y(2)=XYZ(I,2,J)                        37
55                   IF(XYZ(I,3,J).GT.Z(1)) Z(1)=XYZ(I,3,J)                        38
```

CS-23

```
                IF(XYZ(I,3,J).LT.Z(2)) Z(2)=XYZ(I,3,J)                    39
              9 CONTINUE
             10 CONTINUE                                                  40
                GOTO 12
60           11 CONTINUE
                X(1)=XM(1)
                X(2)=XM(2)
                Y(1)=YM(1)
                Y(2)=YM(2)
65              Z(1)=ZM(1)
                Z(2)=ZM(2)
             12 CONTINUE
           C
                XC=(X(1)+X(2))/2.                                         41
70              YC=(Y(1)+Y(2))/2.                                         42
                ZC=(Z(1)+Z(2))/2.                                         43
                DX=ABS(X(2)-X(1))                                         44
                DY=ABS(Y(2)-Y(1))                                         45
                DZ=ABS(Z(2)-Z(1))                                         46
75              CALL SURF                                                 47
           C GO INITIALIZE                                                48
                IF(IBAD) GO TO 50                                         49
                ASSIGN 20 TO JUMP1                                        50
                IF(NTRACT.NE.0) GO TO 50                                  51
80           15 CALL SURTOP                                               52
                IF(.NOT.IGOMOR) GO TO 50                                  53
             20 CONTINUE                                                  54
                IP=1                                                      55
                NTRNL=.FALSE.                                             56
85              CALL INTCLP                                               57
                DO 40 K=1,NPOLY                                           58
           C GET THE NUMBER OF EDGES FOR THIS POLYGON                     59
                   IEDG=INUM(NPOLY,3)
           C GET THE COORDINATES                                          61
90                 DO 30 I=1,IEDG                                        62
                      AX=XYZ(K,1,I)-XC                                       64
                      AY=XYZ(K,2,I)-YC                                     65
                      AZ=XYZ(K,3,I)-ZC                                     66
           C PERFORM THE TRANSFORMATION OF THE COORDINATE                 67
95                       X(I)=AX*A(1,1)+AY*A(1,2)+AZ*A(1,3)+DX            68
                         Y(I)=AX*A(2,1)+AY*A(2,2)+AZ*A(2,3)+DY            69
                         Z(I)=(AX*A(3,1)+AY*A(3,2)+AZ*A(3,3)+DZ)*ALPHA    70
             30          CONTINUE                                         71
                         IP=IP+IEDG+1                                     72
100                      CALL FACMAK(0)
                         CALL POLCLP                                      73
           C JUMP IF POLCLP FAILED. INCREASE SIZE OF FREE, PROBABLY.      74
                         IF(IBAD) GO TO 50                                75
             40 CONTINUE                                                  76
105        C NOW TAKE A PICTURE                                           77
                CALL HIDDEN                                               78
           C JUMP IF MORE ACTION IS NEEDED                                79
           C     IF(NTRACT) JUMP2,15,JUMP2                                80
                IF(NTRACT .EQ. 0) GO TO 15                                81
110             IF(DEBUG .NE. 0) WRITE(IOTFIL,1) JUMP2                    82
```

C5-24

```
      1 FORMAT(1X,*IN FPATCH,JUMP2=*,020)                              83
        GO TO JUMP2 (15,20)                                            84
     50 RETURN                                                         85
        END                                                            86
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
1 FPATCH

VARIABLES	SN	TYPE		RELOCATION					
0	A	REAL	ARRAY	QQQMAT	22	ALPHA	REAL		QQQMAT
224	AX	REAL			225	AY	REAL		
226	AZ	REAL			11	B	REAL	ARRAY	QQQMAT
76	CON	REAL	ARRAY	POLDAT	0	CONTRS	REAL		QFORIO
5	DEBUG	REAL		FILES	3	ELEM	REAL		PLOT3D
0	ELM	REAL	ARRAY	XYZ	214	I	INTEGER		
2	IBACK	INTEGER		INTENS	2	IBAD	LOGICAL		QFORIO
1	IDVICE	INTEGER		QFORIO	0	IEDG	INTEGER		POLDAT
0	IELM	INTEGER	ARRAY	NUM	3	IERRC	INTEGER		FILES
3	IFX	INTEGER		INTENS	4	IFY	INTEGER		INTENS
23	IGOMOR	LOGICAL		QQQMAT	0	IHI	INTEGER		INTENS
1	ILO	INTEGER		INTENS	0	INFILE	INTEGER		FILES
2	INT	INTEGER		PLOT3D	2	INTRAC	INTEGER		FILES
1	IOTFIL	INTEGER		FILES	222	IP	INTEGER		
112	IS	INTEGER	ARRAY	POLDAT	3	ISURF	INTEGER		SEYES
5	I2PASS	INTEGER		QFORIO	216	J	INTEGER		
0	JUMP1	INTEGER		JUMPY	1	JUMP2	INTEGER		JUMPY
223	K	INTEGER			1	KDISP	INTEGER		PLOT3D
62	KOL	INTEGER	ARRAY	POLDAT	4	LBLSPC	INTEGER		QFORIO
0	MAXR	INTEGER		PLOT3D	215	NLOCAL	INTEGER		
4	NOW	INTEGER		FILES	4	NPOLY	INTEGER		PLOT3D
2	NTRACT	INTEGER		JUMPY	1	NTRNL	LOGICAL		POLDAT
0	OX	REAL		SEYES	1	OY	REAL		SEYES
2	OZ	REAL		SEYES	0	RANGE	LOGICAL		RANGE
46	S	REAL	ARRAY	POLDAT	3	SHOSHR	REAL		QFORIO
2	X	REAL	ARRAY	POLDAT	217	XC	REAL		
2	XM	REAL	ARRAY	RANGE	16	Y	REAL	ARRAY	POLDAT
220	YC	REAL			4	YM	REAL	ARRAY	RANGE
32	Z	REAL	ARRAY	POLDAT	221	ZC	REAL		
1	ZINC	REAL		RANGE	6	ZM	REAL	ARRAY	RANGE
1	ZMAX	REAL		ZRANGE	0	ZMIN	REAL		ZRANGE

EXTERNALS	TYPE	ARGS			
FACMAK		1	HIDDEN		0
INTCLP		0	POLCLP		0
SURF		0	SURTOP		0

INLINE FUNCTIONS	TYPE	ARGS					
ABS	REAL	1	INTRIN	INUM	INTEGER	2	SF
XYZ	REAL	3	SF				

CS - 25

STATEMENT LABELS

202	1	FMT	172	5	FMT	0	9
0	10		46	11		56	12
77	15		102	20		0	30
0	40		164	50			

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES			
16	10	* I	48 58	27B		NOT INNER		
20	9	J	50 57	24B	OPT			
107	40	* K	86 104	46B		EXT REFS	EXITS	NOT INNER
116	30	I	90 98	25B	OPT			

COMMON BLOCKS	LENGTH
XYZ	2 LCM
NUM	2 LCM
PLOT3D	5
FILES	6
RANGE	8
SEYES	4
INTENS	5
POLDAT	86
QFORIO	6
ZRANGE	2
QQQMAT	20
JUMPY	3

STATISTICS

PROGRAM LENGTH	2278	151
SCM LABELED COMMON LENGTH	221B	145
LCM LABELED COMMON LENGTH	4B	4

70000B SCM USED

C5-26

```
      SUBROUTINE IORDER3(NUM,IX,Y,NY,LYDIM,AUX)
C ORDERS IN ASCENDING VALUE - X(LYDIM) AND Y(LYDIM,NY) ACCORDING TO
C     IX(NUM)
      DIMENSION IX(1),Y(LYDIM,1),AUX(1)
      LOGICAL AGAIN
C
      LEVEL 2,IX,Y,AUX
C
C
C
    1 LAST=NUM-1
  100 AGAIN=.FALSE.
      DO 20 I=1,LAST
      IF(IX(I+1).GE.IX(I))GOTO 20
      IX1=IX(I)
      IX(I)=IX(I+1)
      IX(I+1)=IX1
    5 IF(NY.EQ.0)GOTO 19
      DO 6 J=1,NY
    6 AUX(J)=Y(I,J)
      DO 10 J=1,NY
   10 Y(I,J)=Y(I+1,J)
      DO 15 J=1,NY
   15 Y(I+1,J)=AUX(J)
   19 AGAIN=.TRUE.
   20 CONTINUE
      IF(AGAIN)GOTO 100
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 IORDER3

VARIABLES	SN	TYPE		RELOCATION					
53	AGAIN	LOGICAL			0	AUX	REAL	ARRAY	F.P.
55	I	INTEGER			0	IX	INTEGER	ARRAY	F.P.
56	IX1	INTEGER			57	J	INTEGER		
54	LAST	INTEGER			0	LYDIM	INTEGER		F.P.
0	NUM	INTEGER		F.P.	0	NY	INTEGER		F.P.
0	Y	REAL	ARRAY	F.P.					

STATEMENT LABELS

0	1	INACTIVE	0	5	INACTIVE	0	6
0	10		0	15		46	19
47	20		13	100			

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
15	20	* I	13 26	33B	NOT INNER
26	6	J	19 20	3B	INSTACK

C5-27

LOOPS	LABEL	INDEX	FROM	TO	LENGTH	PROPERTIES
34	10	J	21	22	2B	INSTACK
43	15	J	23	24	3B	INSTACK

STATISTICS
  PROGRAM LENGTH 63B 51
  70000B SCM USED

CS -28

```
      SUBROUTINE IORDEL3(NUM,IX,Y,NY,LYDIM,AUX)
C ORDERS IN ASCENDING VALUE  X(LYDIM) AND Y(LYDIM,NY) ACCORDING TO
C     IX(NUM)
      DIMENSION IX(1),Y(LYDIM,1),AUX(1)
      LOGICAL AGAIN
C
      LEVEL 2,IX,Y,AUX
C
C
C
    1 LAST=NUM-1
  100 AGAIN=.FALSE.
C
      I=1
    2 IF(I.GT.LAST)GOTO 21
C
      IF(IX(I+1)-IX(I))4,16,20
    4 IX1=IX(I)
      IX(I)=IX(I+1)
      IX(I+1)=IX1
    5 IF(NY.EQ.0)GOTO 19
      DO 6 J=1,NY
    6 AUX(J)=Y(I,J)
      DO 10 J=1,NY
   10 Y(I,J)=Y(I+1,J)
      DO 15 J=1,NY
   15 Y(I+1,J)=AUX(J)
      GO TO 19
C
   16 IX1=I+1
      IF(IX1.GT.LAST)GOTO 110
      DO 18 J=IX1,LAST
      IX(J)=IX(J+1)
      IF(NY.EQ.0)GO TO 18
      DO 17 JJ=1,NY
   17 Y(J,JJ)=Y(J+1,JJ)
   18 CONTINUE
  110 NUM=NUM-1
      LAST=LAST-1
C
   19 AGAIN=.TRUE.
   20 CONTINUE
      I=I+1
      GOTO 2
C
   21 CONTINUE
      IF(AGAIN)GOTO 1
      RETURN
      END
```

C5-35
29

```
      SUBROUTINE RANGXYZ(XYZ,INUM,DOF,NPOLY,NDOF,NTIME)
C FINDS MAX AND MIN VALUES FOR COORDINATES X AND Y IN ARRAY
C     XYZ.  ALSO DETERMINES RANGE OF Z VALUES STORED IN
C     ARRAY DOF FOR ALL NTIME USED.  RESULTS STORED IN COMMON/RANGE/
C     FOR USE IN FPATCH.  ALLOWS SAME SCALING FOR A SERIES OF
C     PLOTS.
C
      DIMENSION XYZ(NPOLY,3,1),DOF(NDOF,1),INUM(NPOLY,3)
      LEVEL 2,XYZ,DOF,INUM
      COMMON/RANGE/RANGE,ZINC,X(2),Y(2),Z(2)
C-----------------------------------------------------------------------
C
      X(1)=Y(1)=Z(1)=-1.E30
      X(2)=Y(2)=Z(2)= 1.E30
C
      DO 10 I=1,NPOLY
      NLOCAL=INUM(I,3)
      DO 9  J=1,NLOCAL
      IF(XYZ(I,1,J).GT.X(1))X(1)=XYZ(I,1,J)
      IF(XYZ(I,1,J).LT.X(2))X(2)=XYZ(I,1,J)
      IF(XYZ(I,2,J).GT.Y(1))Y(1)=XYZ(I,2,J)
      IF(XYZ(I,2,J).LT.Y(2))Y(2)=XYZ(I,2,J)
    9 CONTINUE
   10 CONTINUE
C SET UP RANGE FOR LARGEST VARIATION IN DOF
C
      DO 15 I=1,NDOF
      IA=I+1
      DO 14 J=1,NTIME
      IF(DOF(IA,J).GT.Z(1))Z(1)=DOF(IA,J)
      IF(DOF(IA,J).LT.Z(2))Z(2)=DOF(IA,J)
   14 CONTINUE
   15 CONTINUE
C
C ROUND OFF Z TO NEAREST UNIT
C
      IF(Z(1).GT.0.)Z(1)=ZINC*(AINT(Z(1)/ZINC)+1.)
      IF(Z(1).LT.0.)Z(1)=ZINC*AINT(Z(1)/ZINC)
      IF(Z(2).GT.0.)Z(2)=ZINC*AINT(Z(2)/ZINC)
      IF(Z(2).LT.0.)Z(2)=ZINC*(AINT(Z(2)/ZINC)-1.)
C
      RETURN
      END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 RANGXYZ

C5 - ~~29~~ 30

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 IORDEL3

VARIABLES	SN	TYPE	RELOCATION					
102	AGAIN	LOGICAL		0	AUX	REAL	ARRAY	F.P.
104	I	INTEGER		0	IX	INTEGER	ARRAY	F.P.
105	IX1	INTEGER		106	J	INTEGER		
107	JJ	INTEGER		103	LAST	INTEGER		
0	LYDIM	INTEGER	F.P.	0	NUM	INTEGER		F.P.
0	NY	INTEGER	F.P.	0	Y	REAL	ARRAY	F.P.

STATEMENT LABELS

12	1		16	2		0	4	INACTIVE
0	5	INACTIVE	0	6		0	10	
0	15		52	16		0	17	
67	18		73	19		74	20	
76	21		0	100	INACTIVE	70	110	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
31	6	J	22 23	3B	INSTACK
40	10	J	24 25	2B	INSTACK
47	15	J	26 27	3B	INSTACK
60	18	* J	32 37	10B	NOT INNER
65	17	JJ	35 36	2B	INSTACK

STATISTICS
PROGRAM LENGTH 116B 78
70000B SCM USED

C5-31

```
VARIABLES     SN  TYPE          RELOCATION
    0  DOF        REAL       ARRAY     F.P.          120  I          INTEGER
  123  IA         INTEGER                                0  INUM       INTEGER     ARRAY     F.P.
  122  J          INTEGER                                0  NDOF       INTEGER               F.P.
  121  NLOCAL     INTEGER                                0  NPOLY      INTEGER               F.P.
    0  NTIME      INTEGER                F.P.            0  RANGE      REAL                  RANGE
    2  X          REAL       ARRAY     RANGE           0  XYZ        REAL        ARRAY     F.P.
    4  Y          REAL       ARRAY     RANGE           6  Z          REAL        ARRAY     RANGE
    1  ZINC       REAL                 RANGE

INLINE FUNCTIONS   TYPE   ARGS
      AINT         REAL      1  INTRIN

STATEMENT LABELS
    0  9                                        0  10                                    0  14
    0  15

LOOPS  LABEL    INDEX      FROM-TO    LENGTH    PROPERTIES
   20  10      * I          16 24       24B              NOT INNER
   25  9         J          18 23       16B        OPT
   55  15      * I          27 33       13B              NOT INNER
   57  14        J          29 32       10B     INSTACK

COMMON BLOCKS   LENGTH
      RANGE          8

STATISTICS
  PROGRAM LENGTH                  124B      84
  SCM LABELLED COMMON LENGTH       10B       8
        70000B SCM USED
```

C5-32

```
      SUBROUTINE ZPLT3D(DOF,NDIM,NDOF)
C  PLOTS POLYGONS STORED IN XYZ USING FPATCH
C
C  XYZ(NUM,NDIM,NOD)
C     ARRAY OF ELEMENT NODE COORDINATES USED IN PLOT
C
C  INUM(NUM,1)  ARRAY OF ELEMENT NUMBERS OF NUM ELEMENTS TO BE PLOTTED
C  INUM(NUM,2)  NUMBER OF NODES IN CORRESPONDING ELEMENT
C  INUM(NUM,3)  NUMBER OF NODES TO BE PLOTTED IN ELEMENT, LAST CONNECTS
C     TO FIRST NODE
C  DOF(IA)      DISPLACEMENTS FOR DOF NUMBER JDOF WHERE
C           IA=IADR(JDOF)
C
C  COMMON ----------
C
C  IADR(JDOF)   POINTER INDICATING LOCATION OF DISPLACEMENT FOR
C            JDOF IN ARRAY DOF
C  IFACE(NOD,NODES)  LOCAL NODE NUMBER NOD USED IN PLOT FOR ELEMENT
C            WITH TOTAL NODES (OR FACES).  ASSUMES SAME LOCAL NODES
C            NEED FOR A SPECIFIC TYPE ELEMENT.
C  TRANS     NPIC INDICATES NUMBER OF DIFFERENT PICTURES OF SAME
C     SURFACE.
C      TRANS(i0,NPIC) HOLDS COORDINATE TRANSFORMS USED FOR
C         EACH PICTURE.
C
C-----------------------------------------------------------
C
      DIMENSION DOF(1)
C
      LOGICAL ELEM,OUT
C
      LEVEL 2, XYZ,IELM,DOF,IADR,INTGRK
C
      COMMON/IO/KR,KW,KP,KT1,KT2,KT3,OUT
      COMMON/DPOINT/IADR(1)
      COMMON/FACE/IFACE(4,1)
      COMMON/TRANS/NPIC,TRANS(10,1)
      COMMON/AXIS /IAXIS1,IAXIS2,IAXIS3,C1UNIT,C2UNIT,C3UNIT,
     1 D1MEAN,D2MEAN,D3MEAN
C
      COMMON/NUM/IELM(2)
      COMMON/XYZ/XYZ(2)
C
C
      COMMON/IFORMP/IARG,C1,C2,C3
      COMMON/PLOT3D/MAXRES,KDISP,INT,ELEM,NUM
      COMMON/K     /INTGRK(2)
      COMMON/BEGIN/ICON,IKOUNT,ILNZ,IMASTR,IQ,IK
C
C-----------------------------------------------------------
C
C STATEMENT FUNCTION INUM TO REPLACE ARGUMENTS
      INUM(L,I)=IELM(L+NUM*(I-1))
C-----------------------------------------------------------
      IF(ELEM)GOTO 1200
```

CS-33

```
      IMASTR1=IMASTR-1
C
C  PLACE DISPLACEMENTS FOR IAXIS3 INTO XYZ ARRAY
C
      DO 1100 L=1,NUM
      NOD=INUM(L,3)
      NODES=INUM(L,2)
      LNUM=INUM(L,1)
      IADDR=INTGRK(IMASTR1+LNUM)-1
C
      DO 1050 J=1,NOD
      INOD=IFACE(J,NODES)
      JDOF=INTGRK(IADDR+NDIM*(INOD-1)+IAXIS3)
      IA=IADR(JDOF)
C CALCULATE INDEX OF XYZ FOR I=3
      XYZ(L+3*NUM*J-NUM)=DOF(IA)
 1050 CONTINUE
      IF(.NOT.OUT)GOTO 1100
      WRITE(KW,10)((XYZ(L+NUM*(I-1)+3*NUM*(J-1)),I=1,NDIM),J=1,NOD)
   10 FORMAT(1X,1P12E10.2)
C
 1100 CONTINUE
C
 1200 CONTINUE
C
C  SET UP POLYGONS WITH FPATCH
C
      CALL FPATCH
C
C IF INTERACTIVE, IARG=0, RETURN AFTER FPATCH.
      IF(IARG.EQ.0)GOTO 5000
C
C  MOVIE LOOP WITH TRANSFORMATIONS
C VIEWER IS INITIALLY LOOKING DOWN Z AXIS AND LOCATED AT ORIGIN
C   DATA IS ALSO PLOTTED WITH CENTROID AT ORIGIN
C   TRANSFORMATIONS MOVE DATA AROUND ITS CENTER
C
C
      DO 2100 J=1,NPIC
C INITIALIZE TRANSFORMATION MATRIX
      CALL CINIT
C
C ROTATE FIRST AROUND Z AXIS BY C1 DEGREES
      C1=TRANS(3,J)
      CALL CZROT
C ROTATE AROUND X AXIS BY C1 DEGREES
      C1=TRANS(1,J)
      CALL CXROT
C  ROTATE AROUND Y AXIS BY C1 DEGREES
      C1=TRANS(2,J)
      CALL CYROT
C  TRANSLATE BY C1,C2,C3 UNITS
      C1=TRANS(4,J)
      C2=TRANS(5,J)
      C3=TRANS(6,J)
```

CS-34

```
                    CALL CTRAN
                C  SCALE BY C1,C2,C3 UNITS
                    C1=TRANS(7,J)
                    C2=TRANS(8,J)
      115           C3=TRANS(9,J)
                    CALL CSCAL
                C
                C DISPLAY ON UNIT KDISP WITH FIELD OF VIEW C1 DEGREES
                    C1=TRANS(10,J)
      120           IARG=KDISP
                    CALL CDISP
                C
                C  END OF MOVIE LOOP
                 2100 CONTINUE
      125       C
                 5000 RETURN
                    END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS

3 ZPLT3D

VARIABLES	SN	TYPE		RELOCATION					
1	C1	REAL		IFORMP	3	C1UNIT	REAL		AXIS
2	C2	REAL		IFORMP	4	C2UNIT	REAL		AXIS
3	C3	REAL		IFORMP	5	C3UNIT	REAL		AXIS
0	DOF	REAL	ARRAY	F.P.	6	D1MEAN	REAL		AXIS
7	D2MEAN	REAL		AXIS	10	D3MEAN	REAL		AXIS
3	ELEM	LOGICAL		PLOT3D	203	I	INTEGER		
202	IA	INTEGER			176	IADDR	INTEGER		
0	IADR	INTEGER	ARRAY	DPOINT	0	IARG	INTEGER		IFORMP
0	IAXIS1	INTEGER		AXIS	1	IAXIS2	INTEGER		AXIS
2	IAXIS3	INTEGER		AXIS	0	ICON	INTEGER		BEGIN
0	IELM	INTEGER	ARRAY	NUM	0	IFACE	INTEGER	ARRAY	FACE
5	IK	INTEGER		BEGIN	1	IKOUNT	INTEGER		BEGIN
2	ILNZ	INTEGER		BEGIN	3	IMASTR	INTEGER		BEGIN
171	IMASTR1	INTEGER			200	INOD	INTEGER		
2	INT	INTEGER		PLOT3D	0	INTGRK	INTEGER	ARRAY	K
4	IQ	INTEGER		BEGIN	177	J	INTEGER		
201	JDOF	INTEGER			1	KDISP	INTEGER		PLOT3D
2	KP	INTEGER		IO	0	KR	INTEGER		IO
3	KT1	INTEGER		IO	4	KT2	INTEGER		IO
5	KT3	INTEGER		IO	1	KW	INTEGER		IO
172	L	INTEGER			175	LNUM	INTEGER		
0	MAXRES	INTEGER		PLOT3D	0	NDIM	INTEGER		F.P.
0	NDOF	INTEGER	*UNUSED	F.P.	173	NOD	INTEGER		
174	NODES	INTEGER			0	NPIC	INTEGER		TRANS
4	NUM	INTEGER		PLOT3D	6	OUT	LOGICAL		IO
1	TRANS	REAL	ARRAY	TRANS	0	XYZ	REAL	ARRAY	XYZ

C5-35

```
EXTERNALS          TYPE   ARGS
      CDISP                 0                    CINIT                 0
      CSCAL                 0                    CTRAN                 0
      CXROT                 0                    CYROT                 0
      CZROT                 0                    FPATCH                0

INLINE FUNCTIONS   TYPE   ARGS
      INUM       INTEGER   2    SF

STATEMENT LABELS
  164  10      FMT                      0  1050                      67  1100
   76  1200                             0  2100                     146  5000

LOOPS  LABEL    INDEX     FROM-TO    LENGTH    PROPERTIES
   20  1100   * L          60 77       56B             EXT REFS  NOT INNER
   34  1050     J          66 72        5B     INSTACK
   47         * J          74 74       17B             EXT REFS  NOT INNER
   52         * I          74 74       10B             EXT REFS
  103  2100   * J          94 124      43B             EXT REFS

COMMON BLOCKS   LENGTH
      IO              7
      DPOINT          1 LCM
      FACE            4
      TRANS          11
      AXIS            9
      NUM             2 LCM
      XYZ             2 LCM
      IFORMP          4
      PLOT3D          5
      K               2 LCM
      BEGIN           6

STATISTICS
  PROGRAM LENGTH                207B     135
  SCM LABELED COMMON LENGTH      56B      46
  LCM LABELED COMMON LENGTH       7B       7
        70000B SCM USED
```