TWO-PARAMETER CHARACTERIZATION OF CRACK-TIP FIELDS DURING THERMAL TRANSIENTS

by

DAGMAR E. HAUF

S.B., Mechanical Engineering
Massachusetts Institute of Technology, 1992

Submitted to the Department of
Mechanical Engineering
in Partial Fulfillment of
the Requirements for the Degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

April, 1994

Signature of Author_____
Department of Mechanical Engineering
March, 1994

Certified by_____
David M. Parks
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by_____
Ain A. Sonin, Chairman
Departmental Committee on Graduate Studies

Two-Parameter Characterization of Crack-Tip Fields during Thermal Transients

by

Dagmar E. Hauf

Submitted to the Department of Mechanical Engineering
on April 5, 1994, in partial fulfillment of the requirements for the
Degree of Master of Science in Mechanical Engineering

Abstract

Ample evidence exists that the elastic T-stress, which is the first non-singular term of the WILLIAMS eigen-expansion (1957), is *the* rigorous "second" crack-tip parameter in well-contained yielding. However, we have no knowledge that the two-parameter (J-integral and T) characterization has been examined for the case of transient thermal loading. In view of the marked sensitivity of both ductile (void growth) and brittle (cleavage) fracture mechanisms to crack-tip stress triaxiality, along with the observed strong dependence of stress triaxiality on T, we investigate the effect of the T-stress on the plane strain crack-tip fields during a thermal transient. Numerical techniques are developed to follow the evolution of the T-stress.

To verify the two-parameter characterization under transient thermal loading, we make use of the plane strain elastic-plastic Modified Boundary Layer (MBL) solutions of WANG (1991). The MBL solutions provide a family of stress states whose members can be identified by the value of the T-stress. If the two-parameter characterization holds, the elastically calculated T-stress value at any instant in time during the thermal transient should allow us to uniquely identify a member of the MBL family of crack-tip fields, and that particular MBL field should predict the behaviour of the corresponding elastic-plastic full-field plane-strain solution.

The ability of the MBL solutions in predicting the stress state of the elastic-plastic solutions is exceptional considering that the MBL loading is based on the first two terms of the WILLIAMS eigen-expansion, which neglects the presence of thermal strains in its derivation. Simple extraction of the T-stress variation from an elastic analysis of the problem allows us to predict the triaxiality of the stress state in the elastic-plastic full-field solution.

Thesis Advisor: David M. Parks
Title: Professor of Mechanical Engineering

Acknowledgements

I would like to thank Professor David M. Parks for his incredible support, particularly during the past six months of this project. I truly enjoyed working for you. Thank you for being a friend.

I am very grateful to the very dear friends I have made during my time at MIT: Simona, Richard, Aimee, Stefano, Michele. Thank you for always being there for me. I would like to thank Attilio for all his help and support during the initial stages of this project. I wish he could have been here for the rest.

Another, this time professional, thank you to Simona for letting me use her post-processing program; thanks to Omar, Apostolos and Jian for their system support; and thanks to Christine for battling the scanner problems with me. Of course thanks also to the rest of the Mechanics of Materials group. It's been fun working with you!

Final thanks go to my family in Germany for lending me their support from abroad.

This work was supported by the Office of Basic Energy Sciences, Department of Energy, under Grant $\#DE-FG02-85ER13331$. Computations were performed on SunSparc workstations obtained under ONR Grant $\#0014-89-J-3040$. The ABAQUS finite element program was made available under academic license from Hibbitt, Karlsson, and Sorensen, Inc., Pawtucket, RI.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Near Crack-Tip Stress Fields

Near crack-tip conditions of both linear elastic fracture mechanics (LEFM) and non-linear elastic fracture mechanics (NLEFM) conventionally are characterized by single parameters, the stress intensity factor K_I in the case of LEFM and the J-integral in the case of NLEFM.

One of the basic assumptions behind the application of LEFM to elastic-plastic materials is small-scale yielding (SSY). SSY requires the zone of plastic deformation at the crack-tip to be *much* smaller than any relevant specimen dimension (cf. ASTM E-399). Then, the stress state outside the plastic zone, but away from the specimen boundary, can be characterized by the first singular term of the WILLIAMS eigen-expansion (1957)

$$\sigma_{ij} = \frac{K_I}{\sqrt{2\pi r}} f_{ij}(\theta), \quad \text{with} \quad K_I = c\sigma\sqrt{\pi a}, \tag{1.1}$$

where r and θ are polar coordinates centered at the crack tip as shown in Fig. 1.1, $f_{ij}(\theta)$ are universal angular variations of the respective stress components, σ is the nominal stress, a is the crack length, and c is a dimensionless function which depends on the relevant geometrical dimensions. The entire stress field at the crack tip is

known when the stress intensity factor in Eq. (1.1) is known. That is, K_I completely defines the crack-tip conditions. K_I is said to determine the strength of the dominant elastic singularity.

HUTCHINSON (1968) and RICE and ROSENGREN (1968) independently showed that the J-integral characterizes crack tip conditions in a nonlinear elastic material. The J-integral is defined as the energy release rate in a nonlinear elastic body containing a crack and essentially measures the scale of crack-tip deformation. The line integral expression of J for any contour Γ encircling the crack tip in a counterclockwise direction (see Fig. 1.2) is given by

$$J = \int_{\Gamma} W dy - T_i \frac{\partial u_i}{\partial x} ds, \tag{1.2}$$

where W is the strain energy density, T_i are components of the traction vector acting outward on the contour Γ, u_i are the displacement vector components, and ds is a length increment along the contour Γ. The J-integral is independent of the path of integration around the crack provided that there is no crack face traction or body forces and the near crack-tip region undergoes proportional loading. Under SSY conditions the contour Γ can be chosen to fall within the region in which the K_I-characterized fields hold, thus allowing a relationship between J and K_I to be established as

$$J = \frac{K_I^2}{E'}, \tag{1.3}$$

where $E' = E$ for plane stress and $E' = E/(1-\nu^2)$ for plane strain, E is the Young's modulus, and ν is Poisson's ratio.

HUTCHINSON, and RICE and ROSENGREN showed that in order for J to remain path independent, the product of stress and strain must vary as $1/r$ near the crack tip. In a near-crack tip region, where the plastic strains are much larger than the elastic strains, the equivalent stress and strain are related by a power law in the form

$$\frac{\varepsilon^p}{\varepsilon_y} = \alpha \left(\frac{\sigma_e}{\sigma_y}\right)^n, \tag{1.4}$$

where σ_y is the effective tensile yield strength, $\varepsilon_y = \sigma_y/E$ is the associated tensile yield strain, n is the strain hardening component, and α is a constant. Based on J_2-deformation theory of plasticity and small strain asymptotic analysis, the near crack-tip fields within the plastic zone are then given as

$$\sigma_{ij}(r,\theta) \rightarrow \sigma_y\left(\frac{J}{\alpha\varepsilon_y\sigma_y I_n r}\right)^{\frac{1}{n+1}}\tilde{\sigma}_{ij}(\theta,n) \equiv \sigma_{ij}^{HRR}, \tag{1.5}$$

$$\varepsilon_{ij}(r,\theta) \rightarrow \alpha\varepsilon_y\left(\frac{J}{\alpha\varepsilon_y\sigma_y I_n r}\right)^{\frac{n}{n+1}}\tilde{\varepsilon}_{ij}(\theta,n) \equiv \varepsilon_{ij}^{HRR}, \tag{1.6}$$

where I_n is an integration constant that is a function of n, and $\tilde{\sigma}_{ij}$ and $\tilde{\varepsilon}_{ij}$ are dimensionless functions that depend on θ and n, and on whether plane strain or plane stress prevails in the vicinity of the crack tip. Eqs. (1.5) and (1.6) are called the HRR singularity.

The HRR stress singularity does not apply to points too close to the crack tip; that is, within a region approximately 2 - 3 crack-tip opening displacements (McMeeking, 1977). This restriction applies because the asymptotic HRR fields neglect the finite geometry change at the crack tip. Since the analysis leading to the HRR stress singularity is based on a nonlinear elastic material model and small geometry change, the HRR fields also do not apply where significant elastic unloading or nonproportional loading due to the interaction of thermal and mechanical loads, for instance, exists.

1.2 Two-Parameter Characterization of Near Crack-Tip Fields

Whether a near-crack-tip field is HRR-dominated, that is, whether the asymptotic HRR fields constitute a sufficiently accurate description of the crack-tip field to a radius which includes the fracture process zone, depends strongly upon geometry, loading condition, and strain hardening. The geometry dependence is especially strong

for low-hardening materials in plane strain. The varied ability of attaining HRR dominance at crack tips of different specimens is attributed to the difference in crack-tip constraint (WANG, 1991). A widely used constraint parameter is the stress triaxiality, which is defined as the ratio of hydrostatic stress, $\sigma_m = \frac{1}{3}\sigma_{kk}$, to the Mises equivalent stress, σ_e. Under plane strain conditions, high levels of crack-tip triaxiality are associated with: (a) essentially all states of well-contained yielding; and (b) virtually all load levels in specimens with sufficiently deep cracks under predominately bending load. Conversely, low levels of triaxiality occur in large-scale yielding and fully-plastic flow of single edge-cracked and center-cracked specimens under predominant tension loading, as well as in shallow edge-cracked specimens under bending (PARKS, 1992). A low level of triaxiality generally manifests itself in high crack-tip ductility and high macroscopic toughness. MCMEEKING and PARKS (1979) proposed that crack-tip stress triaxiality remained sufficiently "high" providing

$$J \leq \frac{\sigma_y l}{\mu_{cr}}, \tag{1.7}$$

where μ_{cr} is a "critical" lower limit, and l is a characteristic length parameter such as the uncracked ligament in a deeply-cracked specimen. Since J is directly related to the applied load magnitude, Eq. (1.7) can be interpreted as a limit for applied load to ensure HRR dominance.

Although the effect of specimen geometry and strain hardening on the attainment of HRR dominance is a relatively well known subject, there is no established criterion to define a crack-tip field as "HRR-dominated". Hancock and co-workers (BETEGÓN & HANCOCK, 1991; AL-ANI & HANCOCK, 1991; DU & HANCOCK, 1991) found that the elastic T-stress, which is the second term in the WILLIAMS eigen-expansion of near-crack-tip fields (1957), may be valuable in quantifying the deviation from HRR singularity fields. BETEGÓN & HANCOCK (1991) showed that the variation of stress triaxiality in various plane-strain specimens, as measured locally by the crack-opening stress profiles, can be adequately predicted by introducing T as the constraint pa-

rameter, even up to large scale yielding. AL-ANI & HANCOCK (1991) demonstrated that the deviation from SSY of near-tip crack-opening stress-profiles in plane-strain edge-cracked geometries can be accurately predicted by the two-parameter based solution (J and T) of the respective problems up through (and beyond) moderate-scale yielding. HANCOCK et al. (1993) showed how $\tau(\equiv T/\sigma_y)$ and associated changes in crack-tip stress triaxiality change the initial slopes of resistance curves in A710 steel specimens of varying geometry. WANG (1993) verified the two-parameter characterization of elastic-plastic crack-tip fields (J and T) with a 3-D study of stress fields along the crack fronts of surface-cracked plates (*SCP*s). Recently, O'DOWD & SHIH (1991, 1992) proposed a similar approach with Q as the second parameter, measuring the deviation in crack-tip stress triaxiality from a particular reference value of triaxiality. In small- to moderate-scale yielding Q is isomorphic to the T-stress (PARKS, 1992). However, Q can strictly be obtained only from detailed near-crack-tip fields based upon elastic-plastic finite-element solutions. In 3-D applications the required computational resources and data preparation and reduction make direct implementation of this approach to routine applications all but prohibitive (PARKS, 1992). On the contrary, the elastic T-stress can be evaluated easily from an elastic solution prior to a specific elastic-plastic solution (WANG, 1991).

Although extensive work has been done on the evaluation and correlation of the elastic T-stress, we have no knowledge that the two-parameter characterization has been examined for the case of transient thermal loading. In view of the marked sensitivity of both ductile (void growth) and brittle (cleavage) fracture mechanisms to crack-tip stress triaxiality, along with the observed dependence of stress triaxiality on τ, we investigate the effect of the T-stress on the plane strain crack-tip fields during a thermal transient. The study has potential applications in the power generation industry where the design and operation of conventional and nuclear power plants are founded on rigorous safety requirements. For example, the rules of ASME Section 11 (1974) require the consideration of both mechanical and thermal loads.

1.3 Scope of the Work

Chapter 2 introduces the concept of the elastic T-stress and discusses its use as a second parameter to characterize the near-crack fields. To verify the two-parameter characterization under transient thermal loading, we make use of the work of WANG (1991). He performed plane strain elastic-plastic finite element analyses using a Modified Boundary Layer (MBL) formulation under the assumption of small geometry change at various values of T and constant K_I or J. In this way he generated a family of stress states whose members can be identified by the value of the T-stress. In other words, the same T-stress value, regardless of the specimen geometry and loading type which produced such value, corresponds to a definite crack-tip stress field which is a member of the MBL family of fields (WANG, 1991). That is, if the two-parameter characterization holds for the case of transient thermal loading, the elastically-calculated T-stress value at any instant in time during the thermal transient should allow us to uniquely identify a member of the MBL family of crack-tip stress states, and that stress state should predict the behaviour of the corresponding elastic-plastic full-field plane-strain solution.

Before we can compare the near-crack-tip stress fields with predictions based upon J and T, the variation of the elastic T-stress during the thermal transient needs to be evaluated. In Chapter 3 we give a brief overview of numerical methods used to accurately and reliably evaluate the T-stress as a function of specimen geometry and loading conditions. Specifically we will use the interaction integral of NAKAMURA & PARKS (1992) to track the evolution of the T-stress during a thermal transient. To evaluate the interaction integral, the so-called domain-integral formulation (LI et al., 1985; SHIH et al., 1986) is adopted. Its computational implementation in a post-processing program for the commercial finite-element-code ABAQUS is discussed. Given the numerical tools described in Ch. 3, we obtain the variation of the T-stress in *SEN* specimens of various crack lengths (Chapter 4). The severity of the thermal

shock is varied by varying the Biot number. The Biot number is given by $\beta = h/wk$, where h is the heat transfer coefficient, w is the width of the specimen, and k is the material thermal conductivity, and essentially measures the resistance of the body to heat transfer. A Biot number of infinity corresponds to a step change in temperature on the surface of the specimen under consideration. We assume that the thermal stress problem is quasi-static and that inertia effects are negligible. We finally test the validity of the two-parameter characterization for transient thermal loading using the steps discussed above.

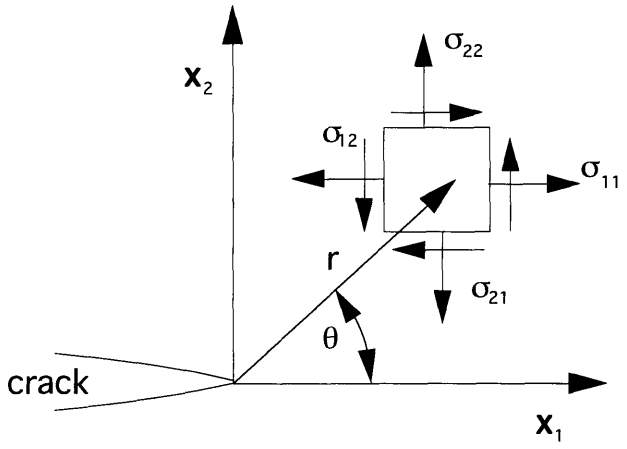


Figure 1.1: Geometry of near crack-tip region.

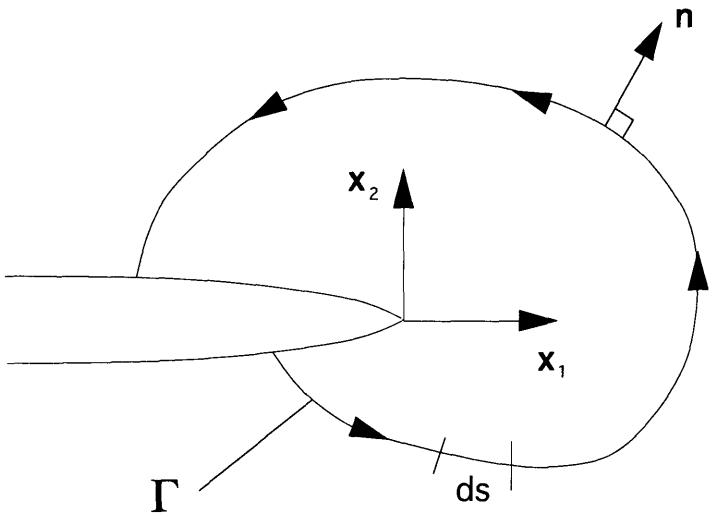


Figure 1.2: Contour definition of the J-integral.

Chapter 2

The Elastic T-Stress

2.1 Introduction

Under the conditions of SSY, the near-crack-tip stress and deformation fields are characterized by the stress intensity factor K_I, and the crack problem can be solved by using a boundary layer (BL) approach. This approach considers a semi-infinite crack in an infinite body and replaces the actual conditions of boundary loading by the asymptotic boundary conditions that

$$\sigma_{ij} = \frac{K_I}{\sqrt{2\pi r}} f_{ij}(\theta) \quad \text{as} \quad r \to \infty . \tag{2.1}$$

The magnitude of K_I is taken from the solution of the elastic boundary value problem modeling the elastic-plastic specimen. The extent to which the near-crack-tip fields of the BL solution and those of an actual specimen agree with each other is an indication of the validity of the K_I-based one-parameter characterization of crack-tip fields under SSY conditions.

Larsson & Carlsson (1973) performed plane-strain elastic-plastic finite-element analyses on four commonly-employed test specimens exhibiting a variety of crack-tip constraint under SSY conditions and compared their computed plastic zones with the

appropriately-scaled plane-strain BL solution. They found significant discrepancies with the BL formulation, even within the range of loads allowed by the ASTM Standard Test Method for Plane-Strain Fracture Toughness of Metallic Materials (E-399). At the maximum permitted load levels, the computed maximum plastic zone sizes for the center-cracked specimen and the double edge-cracked specimen, for example, were greater than that of the BL solution by $\sim 50\%$ and $\sim 25\%$, respectively (see Fig. 2.1). The plastic zones for the different cases would have coincided had the elastic-plastic crack-tip state been determined by K_I alone. LARSSON & CARLSSON showed that the observed differences in plastic zone sizes of the specimens, loaded to identical "small" K_I-levels, are due to specimen-to-specimen differences in the T-stress, the second term of the WILLIAMS (1957) eigen-expansion of near-crack-tip elastic stress fields. The T-stress is not singular as $r \to 0$, but it can alter the elastic-plastic crack-tip stress state, thus *modifying* the crack-tip plastic zone. Like K_I, the T-stress is a function of geometry and loading conditions, and is proportional to the nominal applied stress (LARSSON & CARLSSON (1973); LEEVERS & RADON (1982); etc.). For instance, in shallow-cracked specimens under predominately tensile loading, the proportionality constant is negative, while deeper-cracked specimens under bending often have a less negative or even positive T-stress.

2.2 Modified Boundary Layer Solutions

LARSSON & CARLSSON applied boundary tractions corresponding to the stress fields of the first two terms in the WILLIAMS eigen-expansion,

$$\begin{bmatrix} \sigma_{11}(r,\theta) & \sigma_{12}(r,\theta) \\ \sigma_{21}(r,\theta) & \sigma_{22}(r,\theta) \end{bmatrix} = \frac{K_I}{\sqrt{2\pi r}} \begin{bmatrix} f_{11}(\theta) & f_{12}(\theta) \\ f_{21}(\theta) & f_{22}(\theta) \end{bmatrix} + \begin{bmatrix} T_{11} & 0 \\ 0 & 0 \end{bmatrix}, \tag{2.2}$$

on the same plane-strain domain as that in the previous BL solution. The constant term "T_{11}" in Eq. (2.2) represents the T-stress. The T-stress was obtained from two elastic finite-element solutions; the first was a full-field solution with actual specimen

and loading, the second a BL solution with applied boundary tractions corresponding to Eq. (1.1). The T-stress was then calculated by averaging the difference of the respective x-direction near-tip stresses; that is,

$$T \doteq \sigma_{11}^{spec}(r,\pi) - \sigma_{11}^{BL}(r,\pi)\,, \tag{2.3}$$

where $\sigma_{11}^{spec}(r,\pi)$ is the "11"-component stress of the full-field solution, and $\sigma_{11}^{BL}(r,\pi)$ the stress of the BL solution, respectively. The *Modified Boundary Layer (MBL) Solutions* using this two-parameter description of the loading transmitted to the crack-tip region were in essentially exact agreement with those of each of the corresponding specimens for all loads up to those giving $K_I = 0.6\sigma_y\sqrt{\pi a}$ (see Fig. 2.2).

Recently, extensive work has been done on two-parameter characterizations of near-crack-tip fields. Betegón & Hancock (1991) analyzed near-crack-tip fields of plane-strain specimens having positive, zero, and negative T-stress. Deep within the plastic zone, the crack-opening stress profiles (tensile stress distribution on the plane $\theta = 0$ ahead of the crack) of the specimens closely followed those of the corresponding MBL prediction up to large-scale yielding. The MBL family of solutions are strongly affected by the sign and magnitude of the T-stress. A substantial reduction of crack-opening stress (relative to SSY) is seen for $\tau < 0$; moderate stress elevation above SSY is observed for $\tau > 0$. The effect of the T-stress on the large geometry change deformation field within two crack-tip opening displacements has been discussed by Bilby et al. (1986). Negative T-stresses were shown to reduce the level of maximum hydrostatic stress ahead of the crack. The MBL solutions predicted this decrease inside the plastic zone quite accurately. Al-Ani & Hancock (1991) analyzed plane-strain crack-opening stress in edge-cracked specimens of various crack depths. Remote tension or bending loads, ranging from SSY to large scale yielding, were applied to simulate different levels of crack-tip constraint. The crack-opening stresses were in excellent agreement with the MBL prediction using the calculated elastic-plastic J of the specimen and the elastically-scaled T-stress. Du & Hancock (1991) correlated

the near-crack-tip hydrostatic stress with the T-stress in a non-hardening material (see Fig. 2.3). They showed that the limiting Prandtl field, consisting of constant state regions on the crack flanks connected by centered fans of angular extent $\pi/2$ to a constant state region ahead of the tip, was obtained only for sufficiently positive values of τ. In contrast, when $\tau \leq 0$, an elastic zone of angular extent $\geq \pi/4$ emerges from the crack flanks, cutting into the extent of the centered fan within which hydrostatic stress builds up. The more negative τ becomes, the greater is the reduction in fan extent and crack-tip stress triaxiality (PARKS, 1992). Computational results for the circumferential variation of near-tip stress triaxiality with τ are shown in Fig. 2.5 for the case of strain hardening exponent $n = 10$ (WANG, 1991).

WANG (1993) investigated the influence of the T-stress on the crack-tip opening stress in a variety of *SCP*s. He used a plane-strain MBL formulation by applying displacement boundary conditions dictated by K_I and T on a semi-circular domain. His results are based on a deformation theory plasticity power-law material model exhibiting the tensile stress/strain relation

$$\varepsilon = \begin{cases} \frac{\sigma}{E} & \text{for } \sigma \leq \sigma_y \\ \varepsilon_y \left(\frac{\sigma}{\sigma_y}\right)^n & \text{for } \sigma > \sigma_y ; \quad 1 < n < \infty , \end{cases} \tag{2.4}$$

with $\varepsilon_y \equiv \sigma_y/E$, and a set of material constants representing a moderately hardening material, namely, $\varepsilon_y = 0.0025$, $n = 10$, and $\nu = 0.3$. This relation roughly fits the material data used in the present work (see Fig. 2.4). Fig. 2.6 shows the variation of normalized crack-opening stress (σ_{22} at $\theta = 0$) *vs.* normalized distance at various values of τ. The case $\tau = 0$ (thick solid line) is the opening stress profile at SSY, while the open circles indicate the HRR field. The crack-opening stress deviates considerably from the SSY stress with decreasing τ, while the deviation at high positive τ is less pronounced. At any point outside the crack-tip blunting zone, the stress profiles for different values of τ are roughly parallel to each other, which is consistent with the observation of BETEGÓN & HANCOCK (1991). This suggests that the deviation from SSY is essentially independent of normalized distance from the

crack tip.

Thus, the variation of the plane-strain crack-opening stress with respect to τ, at any normalized distance in the range $1 < r/(J/\sigma_y) < 6$, can be fitted in the following three-parameter form

$$\frac{\sigma_{22}^{MBL}(r/(J/\sigma_y);\tau)}{\sigma_y} = \frac{\sigma_{22}^{SSY}(r/(J/\sigma_y))}{\sigma_y} + A_n\tau + B_n\tau^2 + C_n\tau^3, \qquad (2.5)$$

where A_n, B_n, and C_n are constants dependent upon the strain hardening exponent n. This three-term polynomial fit was suggested by WANG (1991). Fig. 2.7 compares WANG's results with the finite-element solution at $r = 2J/\sigma_y$. The fitting parameters are $A_n = 0.6168$, $B_n = -0.5646$, and $C_n = 0.1231$ for $\varepsilon_y = 0.0025$, $n = 10$, and $\nu = 0.3$.

RICE (1974), SHIH et al. (1993), and WANG (1991) noted a strong variation of plastic zone size with the T-stress. Fig. 2.8, from the work by WANG, shows the plastic zone size at various values of τ, normalized by the SSY plastic zone size at ($\tau = 0$). The results of the simple shear band yielding model (band shear traction = τ_y, a constant) of RICE (1974) are shown for the purpose of comparison. His results are given in terms of T and the equivalent tensile strength $\sigma_y = \sqrt{3}\tau_y$, with the plastic zone size radius, r_p, as

$$r_p = \frac{\pi}{64}\frac{\sin^2\phi(1+\cos\phi)}{(1/\sqrt{3}+\tau\sin\phi\cos\phi)^2}\left(\frac{K_I}{\sigma_y}\right)^2. \qquad (2.6)$$

The angle ϕ in Eq. (2.6) is measured from the crack plane. In practice, ϕ in Eq. (2.6) is an implicit function of τ, $\phi = \hat{\phi}(\tau)$, which is obtained by maximizing r_p with respect to ϕ at fixed τ; at $\tau = 0$, this procedure results in $\phi = 70.6^o$ (RICE, 1974). Also shown are the results of SHIH et al. (1993) for a moderately hardening material ($n = 10$). At $\tau = 0$, $r_p^{max} \sim 0.15(K_I/\sigma_y)^2 \equiv r_p^{SSY}$, which differs only slightly from the generally used nominal plastic zone size, $(1/2\pi)(K_I/\sigma_y)^2$. The maximum plastic zone size grows monotonically with decreasing τ, reaching $\sim 50r_p^{SSY}$ when $\tau = -1.0$.

Positive values of τ cause the plastic zone size to first decrease, then increase, reaching $\sim 10 r_p^{SSY}$ when $\tau = 1.0$.

The effect of the T-stress on the equivalent plastic strain ε^p at a distance from the tip equal to $r = 1.22J/\sigma_y$ is shown in Fig. 2.9 (WANG, 1993). The thick solid line is the SSY solution ($\tau = 0$). Negative τ is associated with a large increase in ε^p (at $\tau = -1.0$, the peak ε^p has increased by $\sim 80\%$ compared to the corresponding peak ε^p at $\tau = 0$) and a shift of the peak to the forward section ($\theta < 90^o$). A slight decrease of peak ε^p is observed at a τ-value between 0.2 and 0.4. For $\tau > 0.4$, the peak ε^p increases (for $\tau = -1.0$ by $\sim 25\%$ compared to the peak value at $\tau = 0$), while the location of the peak at $\tau = 1.0$ shifts back toward the cracked flank.

Based on this noted sensitivity of plastic zone size and orientation to the sign and magnitude of the T-stress, HAUF et al. (1994) recently formulated a Modified Effective Crack Length for plane strain by including effects of T into the definition of the standard effective crack length. They demonstrated that their formulation consistently extends the load range for which accurate predictions of compliance, J-integral, and crack-tip constraint are obtained in several plane-strain specimen geometries.

Based on these observations, then, we establish the following criterion to determine the suitability of the two-parameter characterization of near-crack tip fields during thermal transients. That is, we consider the two-parameter (J and T) characterization to hold if the magnitude and sign of the T-stress given at a particular instant in time during the thermal transient

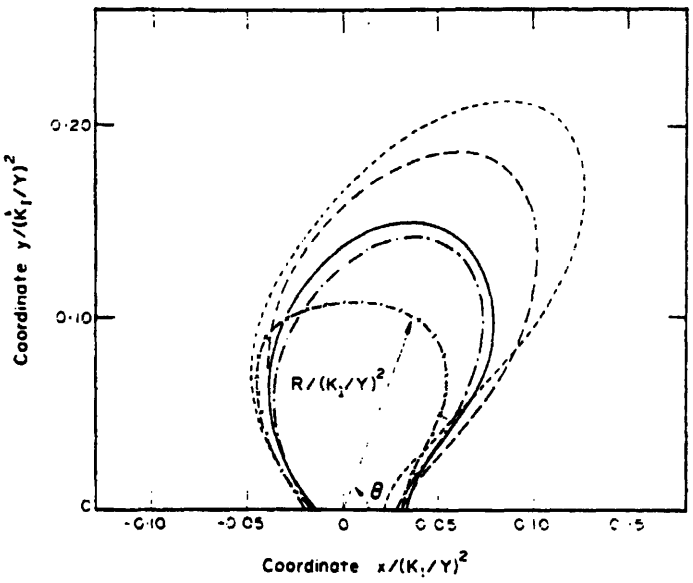
> allows us to identify a member of the MBL family of crack-tip stress states and the crack-tip fields of that particular MBL solution suitably describe the behaviour of the corresponding full-field plane-strain solution in the range $1 < r/(J/\sigma_y) < 6$.

We do not expect the MBL fields to precisely match those of the corresponding full-field solution, since the WILLIAMS eigen-expansion, on which the MBL loading is based, requires the absence of body forces and thermal strains in its derivation. Nevertheless, we expect that major features of the respective fields will correspond. Our approach is schematically depicted in Fig. 2.10.

It should be noted that under plane-strain conditions ($\varepsilon_{33} = 0$) the presence of thermal strains results in finite mechanical tension/compression out-of-plane strains tangential to the crack front which have an effect on the near-crack-tip stress fields. PARKS (1991) suggested a generalized form of Eq. (2.2) to express the linear-elastic stress distribution in the vicinity of the crack front; that is,

$$\begin{bmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{bmatrix} = \frac{K_I}{\sqrt{2\pi r}} \begin{bmatrix} f_{11}(\theta) & f_{12}(\theta) & f_{13}(\theta) \\ f_{21}(\theta) & f_{22}(\theta) & f_{23}(\theta) \\ f_{31}(\theta) & f_{32}(\theta) & f_{33}(\theta) \end{bmatrix} + \begin{bmatrix} T_{11} & 0 & T_{13} \\ 0 & 0 & 0 \\ T_{31} & 0 & T_{33} \end{bmatrix} . \quad (2.7)$$

Based on Eq. (2.7), WANG (1993) investigated the effects of out-of-plane strains on the near-crack-tip fields in the context of three-dimensionality of crack fronts for the special case $T_{13} = T_{31} = 0$ and finite T_{33} by varying the out-of-plane strain ε_{33} at the same value of τ. Figure 2.11 from his work shows the effect of out-of-plane strain on the crack-opening stresses. Clearly, the out-of-plane strain has a much smaller effect than the T-stress. At $r = 2J/\sigma_y$, the stresses decrease by $\sim 3\%$ at $\varepsilon_{33}/\varepsilon_y = -0.9$ compared to the value at $\varepsilon_{33} = 0$. The stress profile for $\varepsilon_{33}/\varepsilon_y = -0.9$ seems slightly rotated compared to that at $\varepsilon_{33} = 0$. That is, for a distance $r > 2J/\sigma_y$ the normalized crack-opening stress at $\varepsilon_{33}/\varepsilon_y = 0$ decreases more gradually compared to results at $\varepsilon_{33}/\varepsilon_y = -0.9$.



------ Center–cracked specimen

– – – – Double–edge–cracked specimen

—·—·— Bend specimen

– – – Compact tension specimen

——— Boundary layer solution .

Figure 2.1: Plastic Zones, with axes normalized by the characteristic length scale $(K_I/Y)^2 (Y = \sigma_y$: yield strength), of various specimens and BL solution at $K_I = 0.6\,Y\,a^{1/2}$ (Larsson and Carlsson, 1973).

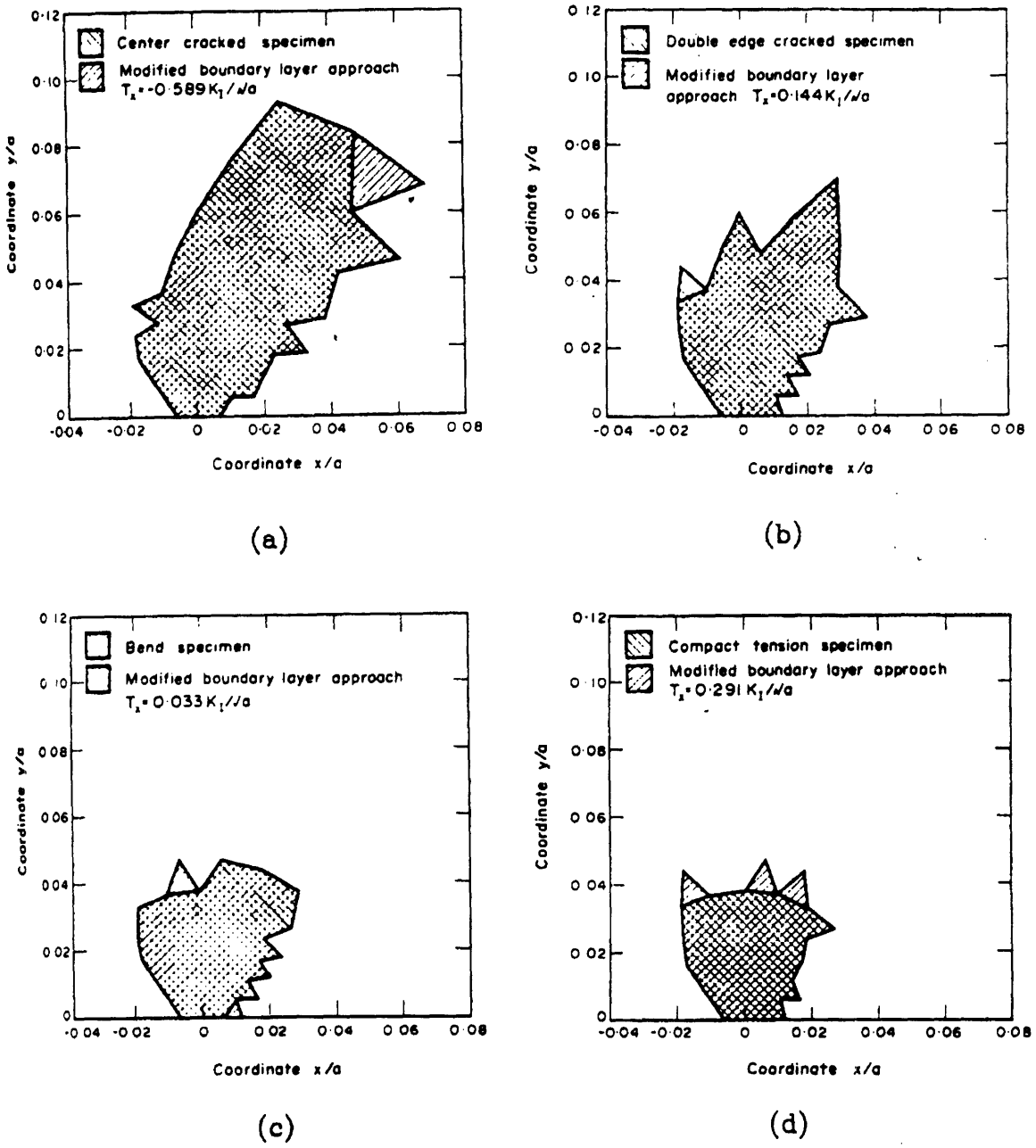


Figure 2.2: Plastic Zones at a load level $K_I = 0.6\,Y\,a^{1/2}$ for actual geometries and the MBL solution (Larsson and Carlsson, 1973).

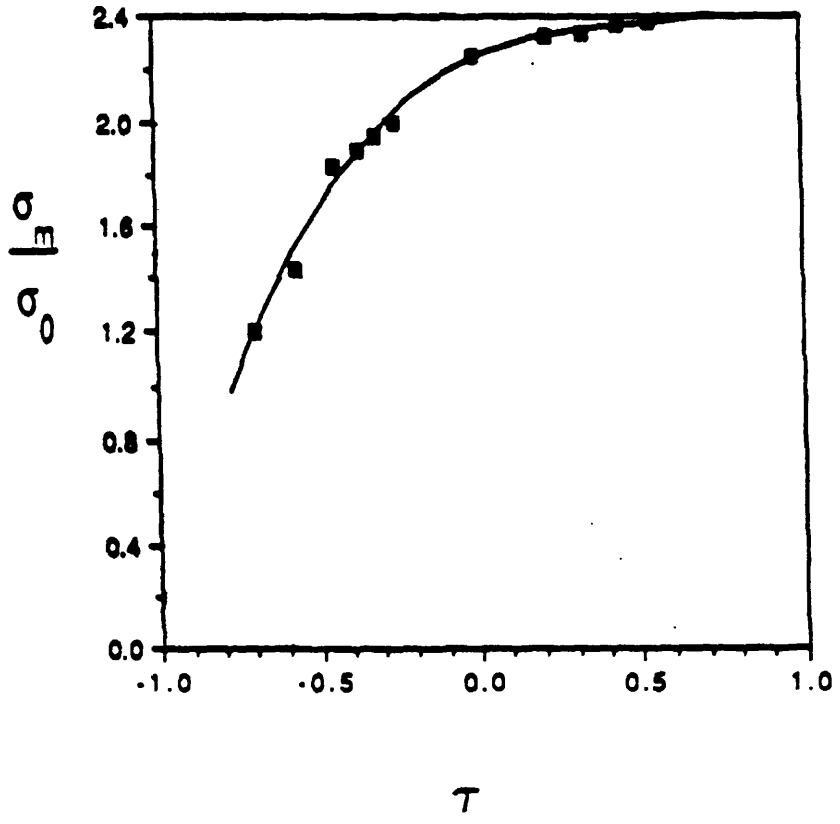


Figure 2.3: The variation of stress triaxiality near a crack tip with respect to $\tau = T/\sigma_y$ in a nonhardening material; $\sigma_o = \sigma_y$ (Du and Hancock, 1991).

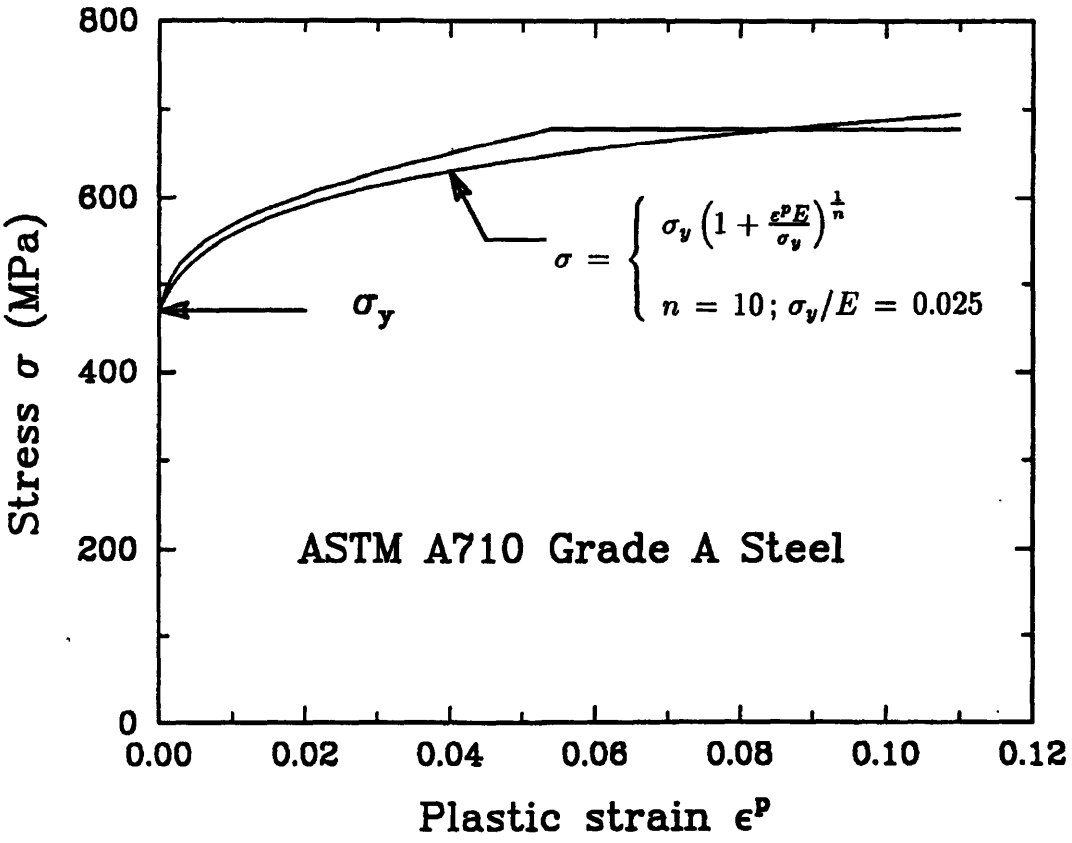


Figure 2.4: Engineering stress *vs.* plastic strain curve in uniaxial tension, multi-linearly modeled from experimental data of ASTM A710 Grade A steel and used in flow theory continuum finite-element solutions. Also shown is the power law fit used in Wang's work (1991) for a moderately hardening material with $n = 10$, and $\nu = 0.3$.

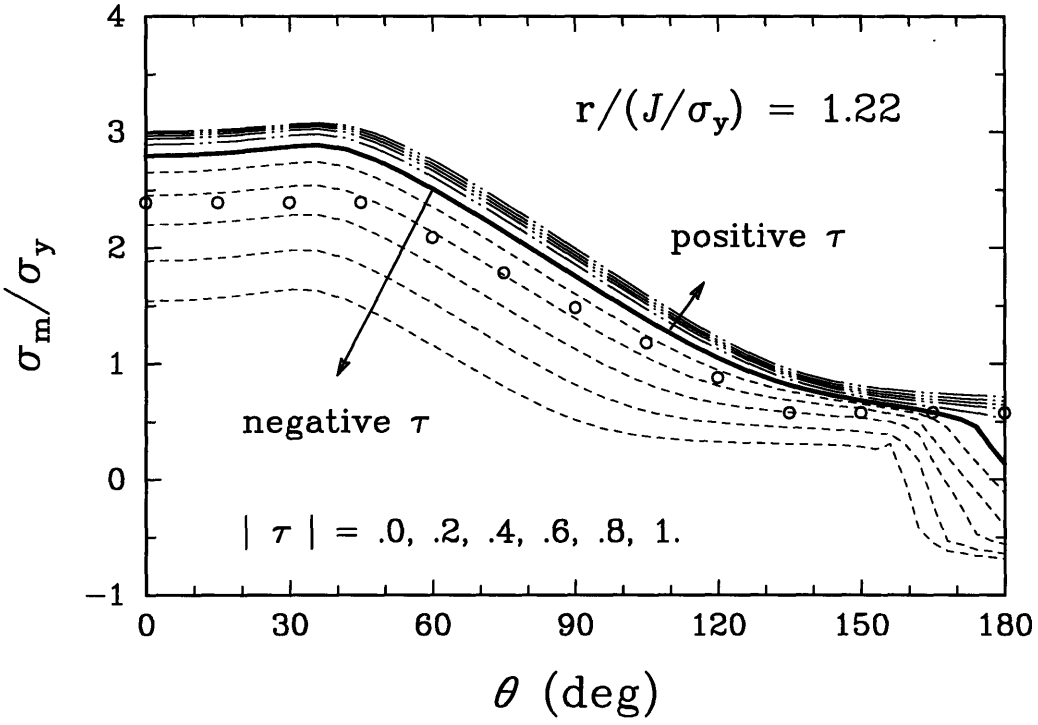


Figure 2.5: Angular variation of near-tip normalized hydrostatic stress for various values of τ in plane-strain MBL solutions with $n = 10$ (Wang, 1991).

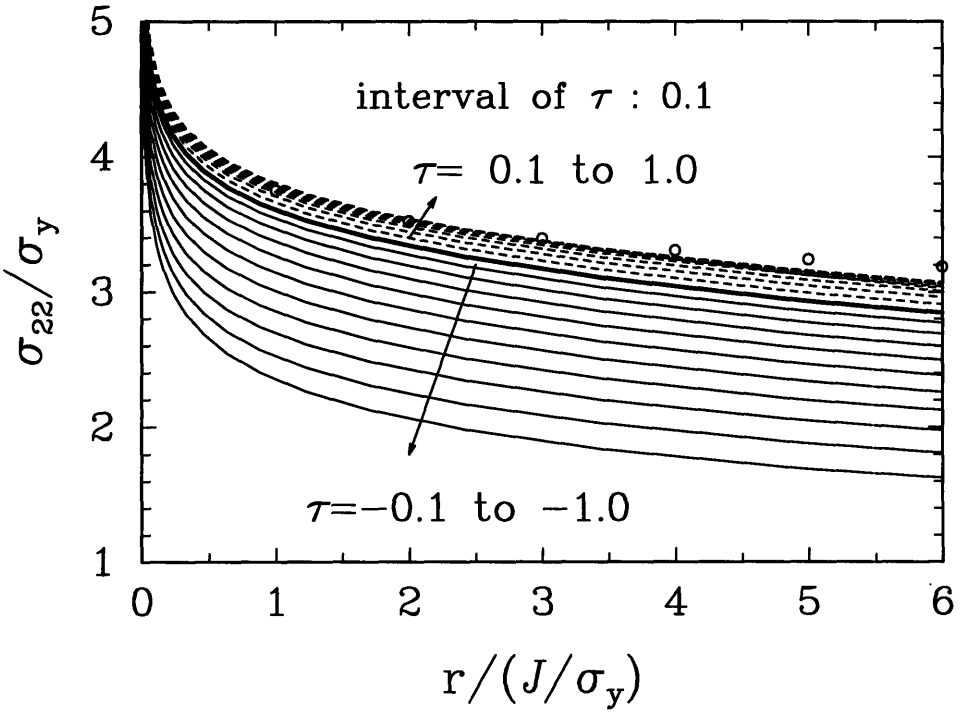


Figure 2.6: Normalized crack-opening stress profiles in plane-strain MBL solutions for hardening exponent $n = 10$, for various values of τ. The stresses marked with circles are HRR-singularity fields (Wang, 1993).

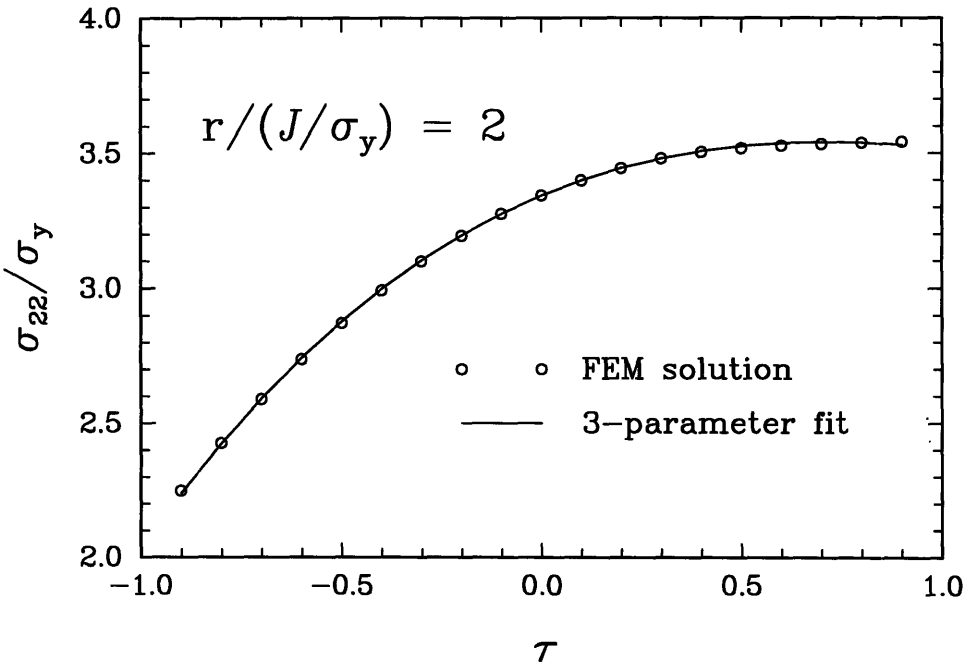


Figure 2.7: Comparison of relative accuracy of three-parameter fit to the normalized crack-opening stress a distance $2J/\sigma_y$ ahead of the crack tip in plane-strain MBL solutions for $n = 10$ with respect to the finite-element solution (Wang, 1991).

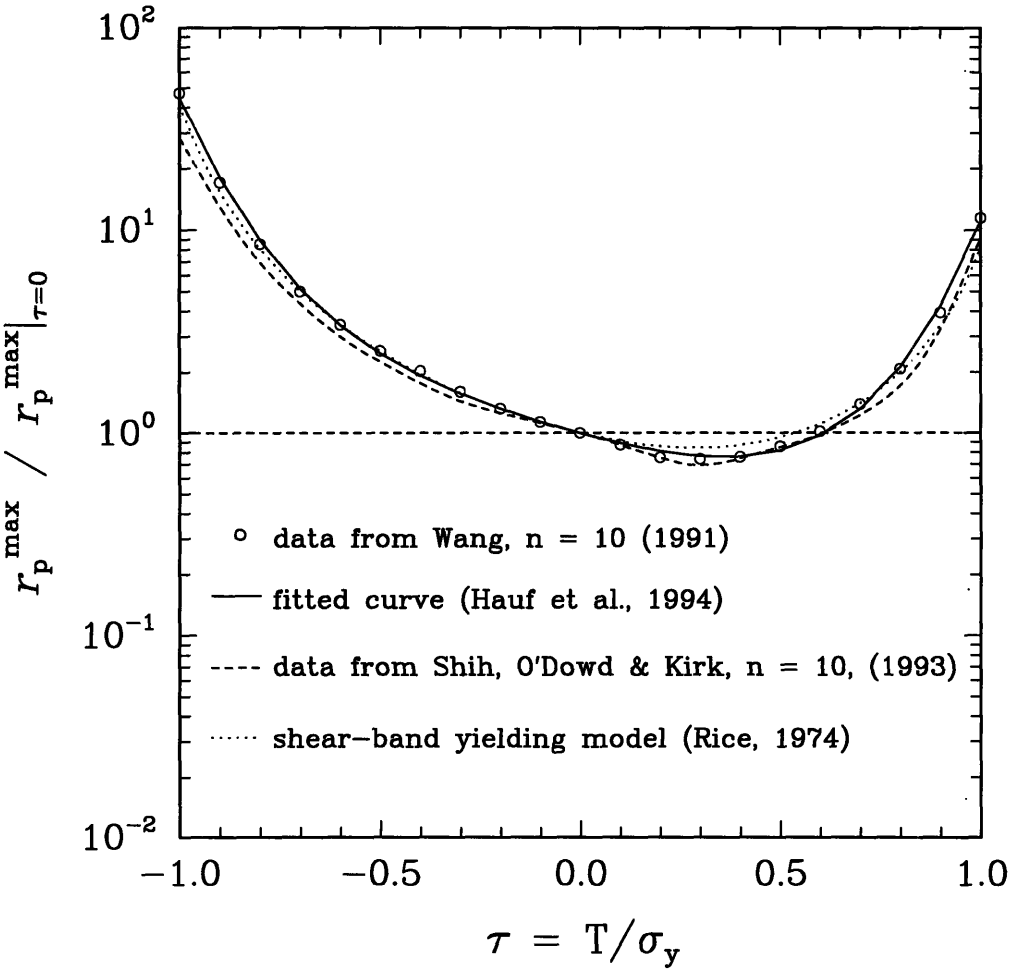


Figure 2.8: Variation of maximum plane-strain plastic zone size, normalized by the plastic zone size at $\tau = 0$, at various values of τ (Wang, 1991) in a modified boundary layer formulation.

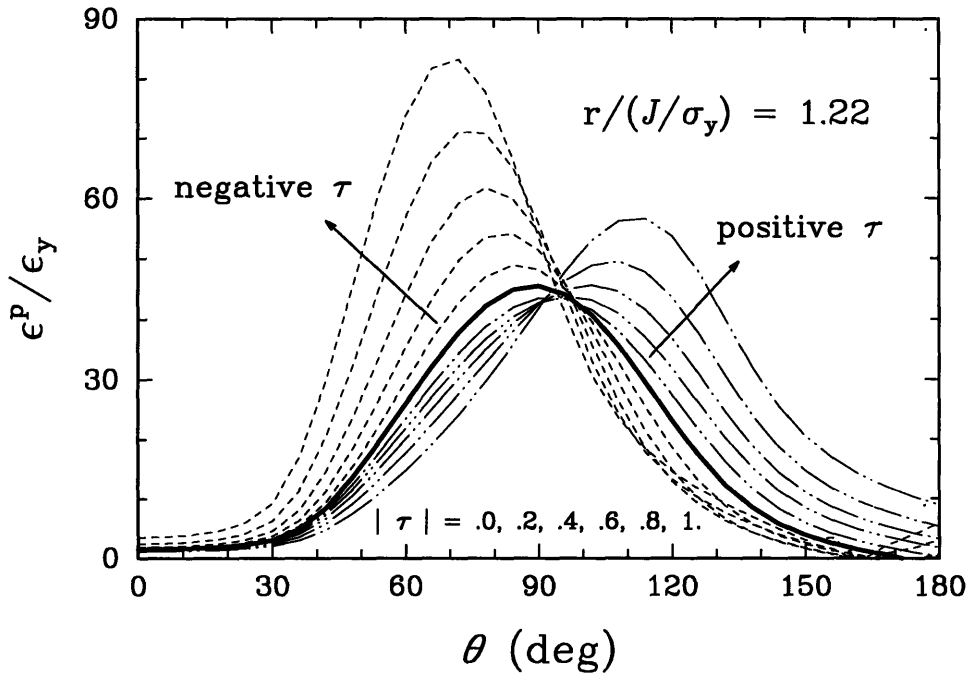


Figure 2.9: Angular variation of near-tip equivalent plastic strain for various values of τ in plane-strain MBL solutions with $n = 10$ (Wang, 1993).

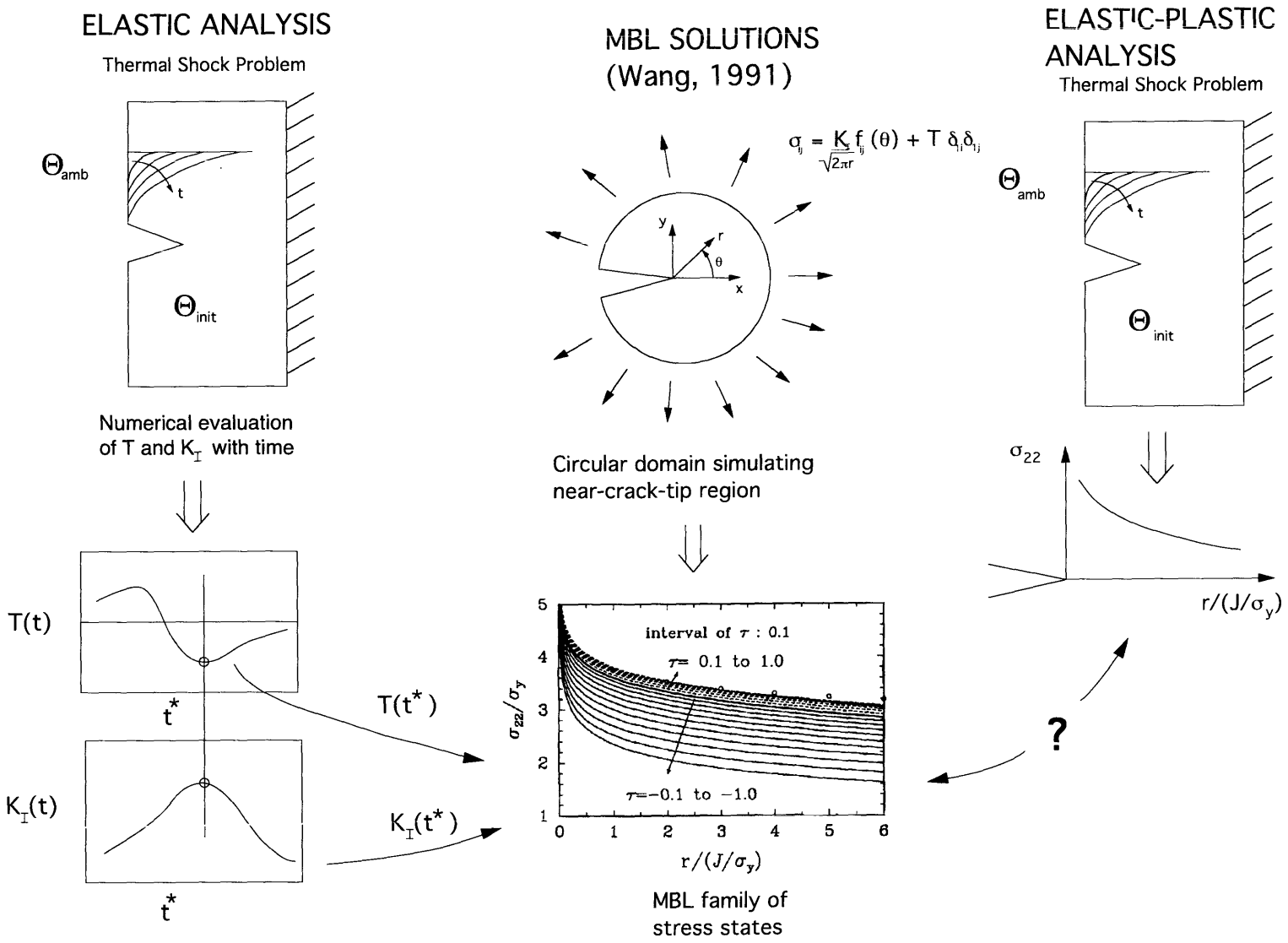


Figure 2.10: Verification of the two-parameter characterization.

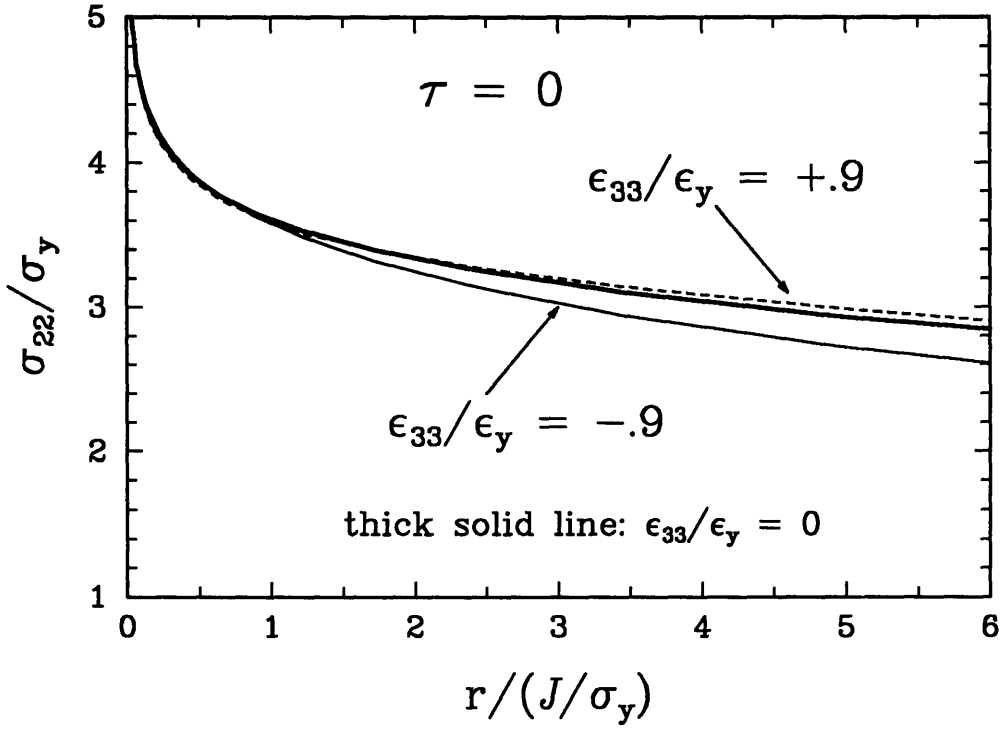


Figure 2.11: Normalized crack-opening stress profiles in plane-strain MBL solutions for hardening exponent $n = 10$ at various values of ϵ_{33} (K_I = constant, $\tau = 0$) (Wang, 1993).

Chapter 3

Numerical Methods

3.1 Introduction

Several methods are available in the literature for evaluating the elastic T-stress in 2-D specimens under various loading conditions. The most obvious method can be derived directly from the definition of the T-stress; that is, by using Eq. (2.2). Assuming the near-crack-tip stress fields can be adequately represented by the first two terms in the WILLIAMS (1957) eigen-expansion, the T-stress can be obtained as

$$T = \sigma_{11}^{spec}(r,\theta) - \frac{K_I}{\sqrt{2\pi r}} f_{11}(r,\theta)\,, \tag{3.1}$$

where $\sigma_{11}^{spec}(r,\theta)$ is the x_1-direction normal stress in the near-crack-tip region of an actual specimen and loading (WANG, 1991). LARSSON & CARLSSON (1973) determined T in this way using two elastic finite-element solutions. The first solution was obtained from an actual specimen and loading analysis. The second was a boundary layer formulation of a circular domain with a semi-infinite crack, in which the traction boundary conditions corresponding to the second term on the RHS of Eq. (3.1) were imposed. The magnitude of K_I applied in the elastic BL solution was determined from the solution of the elastic boundary value problem modeling the elastic-plastic specimen. LEEVERS & RADON (1982) calculated the coefficients of the WILLIAMS eigen-

expansion using a variational formulation. WANG & PARKS (1992) obtained approximate estimates of the T-stress distribution in a wide range of surface-cracked plates under tension and bending using the line-spring method. SHAM (1991) computed the 2-D elastic T-stress using second-order weight functions. Based on a theorem due to ESHELBY (CARDEW et al., 1984), KFOURI (1986) evaluated T in terms of the difference in J-integral of two finite-element solutions. The first elastic finite-element solution was generated from an actual specimen and loading analysis. A second solution was generated by superposing a point load solution to the first solution. The elastic T was then obtained from a relation involving the J-integrals of the two solutions. Since J can be accurately evaluated from a moderately refined elastic finite-element analysis, T can be obtained with a mesh much less refined than that of the LARSSON & CARLSSON method. Recently, NAKAMURA & PARKS (1992) extended this method to 3-D crack fronts using a domain interaction integral. We use the interaction integral of NAKAMURA & PARKS to evaluate the T-stress in 2-D plane-strain specimens under transient thermal loading. In the following paragraphs, which closely follow the derivation presented by NAKAMURA & PARKS, we describe the near-tip fields and line-load solutions needed in the evaluation of the interaction integral, present the resulting expression for the T-stress, and finally use the domain-integral method (LI et al., 1985; SHIH et al., 1986; MORAN & SHIH, 1987, for example) to represent the interaction integral in a form suited to numerical implementation.

3.2 Evaluation of the T-stress

3.2.1 Near-Tip Fields and Line-Load Solutions

In an isotropic linear-elastic body containing a crack subject to symmetric (mode I) loading, the leading terms [up to $O(1)$] in a series expansion of the stress field very

near the crack front are

$$\begin{aligned}
\sigma_{11} &= \frac{K_I}{\sqrt{2\pi r}}\cos\frac{\theta}{2}\left(1-\sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)+T\,,\\
\sigma_{22} &= \frac{K_I}{\sqrt{2\pi r}}\cos\frac{\theta}{2}\left(1+\sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)\\
\sigma_{33} &= \frac{K_I}{\sqrt{2\pi r}}2\nu\cos\frac{\theta}{2}+T_{33}\,,\\
\sigma_{12} &= \frac{K_I}{\sqrt{2\pi r}}\sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\frac{3\theta}{2}\,,\\
\sigma_{13} &= \sigma_{23}=0\,,
\end{aligned} \tag{3.2}$$

where r and θ are the in-plane coordinates of the plane normal to the crack front, K_I is the local stress intensity factor, and ν is the Poisson's ratio. Here x_1 is the direction formed by the intersection of the plane normal to the crack front and the plane tangential to the crack plane. The associated strain field is given by

$$\begin{aligned}
\varepsilon_{11} &= \frac{(1+\nu)}{E}\frac{K_I}{\sqrt{2\pi r}}\cos\frac{\theta}{2}\left(1-2\nu-\sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)+\frac{1}{E}(T-\nu T_{33})+\alpha(\Theta-\Theta_{init})\\
\varepsilon_{22} &= \frac{(1+\nu)}{E}\frac{K_I}{\sqrt{2\pi r}}\cos\frac{\theta}{2}\left(1-2\nu+\sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)-\frac{\nu}{E}(T+T_{33})+\alpha(\Theta-\Theta_{init})\\
\varepsilon_{33} &= \varepsilon_{33}^{th}+\varepsilon_{33}^{m}=\alpha(\Theta-\Theta_{init})+\varepsilon_{33}^{m}\\
\varepsilon_{12} &= \frac{(1+\nu)}{E}\frac{K_I}{\sqrt{2\pi r}}\sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\frac{3\theta}{2}\\
\varepsilon_{23} &= \varepsilon_{13}=0\,,
\end{aligned} \tag{3.3}$$

where α is the constant linear thermal expansion coefficient, Θ_{init} the reference temperature value at the undeformed state, and ε_{33}^{m} and ε_{33}^{th} are the mechanical and the thermal strains in the x_3-direction, respectively. The terms $T(=T_{11})$ and T_{33} are the amplitudes of the second order terms in the three-dimensional series expansion of the crack-front stress-field in the x_1- and x_3-directions. We can decompose T_{33} into $T_{33}=\nu T+\sigma^*$, where $\sigma^*=\varepsilon_{33}^{m}E=[\varepsilon_{33}-\alpha(\Theta-\Theta_{init})]E$. This decomposition of T_{33} is different from the one presented by NAKAMURA & PARKS (1992) as they did not consider thermal strains in their derivation.

To extract the T-stress, an auxiliary solution corresponding to a plane-strain line-load applied along the crack front in the direction of crack extension is used. The solution is a special case of a line-load symmetrically applied at the apex of a wedge of included angle 2π (TIMOSHENKO & GOODIER, 1970). Suppose that a line load with magnitude f (force per unit length) in the x_1-direction is locally applied along the same crack front segment. Then the stress field in the crack-tip region is given by

$$\begin{aligned}
\sigma_{11}^L &= -\frac{f}{\pi r}\cos^3\theta \\
\sigma_{22}^L &= -\frac{f}{\pi r}\cos\theta\sin^2\theta \\
\sigma_{33}^L &= -\frac{f}{\pi r}\nu\cos\theta \\
\sigma_{12}^L &= -\frac{f}{\pi r}\cos^2\theta\sin\theta \\
\sigma_{13}^L &= \sigma_{23}^L = 0\ .
\end{aligned} \tag{3.4}$$

Here, the superscript "L" denotes the line-load solution. The strain (ε_{ij}^L) and displacement (u_i^L) fields corresponding to the stress field in Eq. (3.4) are readily calculated (see Appendix A for a detailed derivation). The above solutions are valid at any material point as long as r, its radial distance to the crack front, is sufficiently small compared to other relevant physical dimensions.

3.2.2 The Interaction Integral

Consider the line-load, $f_i = f\mu_i(s)$, to be applied along the crack front as shown in Fig. 3.1(a). In the figure, s is an arc-length-measuring parameter representing the location of the crack-tip on the crack front, and $\mu_i(s)$ is a unit vector giving the direction formed by the intersection of the plane normal to the crack front and the plane tangential to the crack front at s. By superimposing the actual field, Eq. (3.2), with the field due to the line-load application, Eq. (3.4), a local conservation integral

is introduced as

$$I(s)\mu_k(s) = \lim_{\Gamma\to 0}\int_{\Gamma(s)}\left[\sigma_{ij}\varepsilon^L_{ij}n_k - \sigma_{ij}\frac{\partial u^L_i}{\partial x_k}n_j - \sigma^L_{ij}\frac{\partial u_i}{\partial x_k}n_j\right]d\Gamma. \tag{3.5}$$

A path $\Gamma(s)$ surrounds the crack front at s and lies in the plane perpendicular to the crackfront at s. The components n_i are those of a unit vector lying in this plane and normal to the tangent to Γ, as shown in Fig. 3.1(b). The limit ($\Gamma \to 0$) must be preserved in three-dimensional problems. However, the shape of the path may be arbitrary as Γ shrinks onto the tip.

Now suppose that $\mu_i(s)$ is given in the local x_1-direction and that the path Γ is circular with radius r. Then as the limit is taken ($r \to 0$), the stress fields in Eqs. (3.2) and (3.4) (and their associated kinematic fields) become applicable in the integrand. After substituting the fields into the integral, the interaction integral was evaluated with the help of the program $Mathematica^{TM}$ as

$$\begin{aligned} I(s) &= \frac{f}{E}\left[T(s)\,(1-\nu^2) + E\alpha\Delta\Theta(s) - \nu\sigma^*\right] \\ &= \frac{f}{E}\left[T(s)\,(1-\nu^2) + E\alpha\Delta\Theta(s)(1+\nu) - \nu E\varepsilon_{33}(s)\right], \end{aligned} \tag{3.6}$$

where $\Delta\Theta(s) = \Theta(s) - \Theta_{init}$ is the temperature difference between the crack-tip temperature and the reference temperature of the specimen. In the integration, the terms in the crack-tip fields, Eq. (3.2), containing K_I cancel out exactly, and only the non-singular terms in Eq. (3.2) contribute to $I(s)$. Solving for $T(s)$, we obtain

$$T(s) = \frac{E}{(1-\nu^2)}\left[\frac{I(s)}{f} - \alpha\Delta\Theta(s)(1+\nu) + \nu\varepsilon_{33}(s)\right]. \tag{3.7}$$

Under isothermal conditions , $\Delta\Theta = 0$, and Eq. (3.7) reduces to the expression given by Nakamura & Parks.

From a computational point of view, Eq. (3.5) is not suitable for evaluating $I(s)$ since accurate numerical evaluation of limiting fields along the crack front is difficult.

Here the so-called "domain-integral formulation" (LI et al., 1985; SHIH et al., 1986) is adopted. An approximate expression for $I(s)$ may be obtained as follows. The total interaction energy $\bar{I}(s)$ released when a finite segment L of the crack front advances an amount $\Delta a\, l_k(s)$ (see Fig. 3.2) at the point s in the direction normal to the crack front is given by

$$\Delta a \bar{I}(s) = \Delta a \int_L I(s) l_k(s) \mu_k(s) ds \,. \tag{3.8}$$

On employing Eq. (3.5) in Eq. (3.8) we obtain

$$\begin{aligned} \bar{I}(s) &= \int_L l_k(s) \left[\lim_{\Gamma \to 0} \int_{\Gamma(s)} \left[\sigma_{ij} \varepsilon^L_{ij} n_k - \sigma_{ij} \frac{\partial u^L_i}{\partial x_k} n_j - \sigma^L_{ij} \frac{\partial u_i}{\partial x_k} n_j \right] d\Gamma \right] ds \\ &= \int_{S_t} \left[\sigma_{ij} \varepsilon^L_{ij} n_k - \sigma_{ij} \frac{\partial u^L_i}{\partial x_k} n_j - \sigma^L_{ij} \frac{\partial u_i}{\partial x_k} n_j \right] l_k(s) dS \,, \end{aligned} \tag{3.9}$$

where for the case of a sharp crack S_t is the tubular surface enclosing the crack front segment as shown in Fig. 3.1(c), and the limiting process consists of shrinking the "tube" radius to zero. For simplicity, we will model the crack as a notch with notch thickness h in the following (see Fig. 3.4) and require $h \to 0$ in the sharp crack configuration of interest. The surface of the notch consists of faces S_A and S_B, with normals along $\pm x_2$-directions, respectively, and a face with a normal in the $x_1 - x_3$ plane.

Next we identify the arbitrary closed surface S with the surface $S_1 + S_+ + S_- - S_t$ (see Fig. 3.4) and introduce the continuous functions q_k defined by

$$q_k = \begin{cases} l_k \text{ on } S_t \\ 0 \text{ on } S_1 \\ \text{otherwise arbitrary} \end{cases} . \tag{3.10}$$

Requiring q_k to be sufficiently smooth in the volume V and invoking the divergence theorem, the expression for the interaction integral over a domain/volume is,

$$\bar{I}(s) = \int_{V(s)} \left[\left\{ \left(\sigma_{ij} \frac{\partial u^L_i}{\partial x_k} + \sigma^L_{ij} \frac{\partial u_i}{\partial x_k} \right) q_k \right\}_{,j} - \left(\sigma_{ij} \varepsilon^L_{ij} q_k \right)_{,k} \right] dV \,. \tag{3.11}$$

The stress strain relations for a linear material are $\sigma_{pq} = C_{pqrs}\,\varepsilon^m_{rs}$, where $C_{pqrs} = C_{rspq}$ are the elastic moduli. Hence,

$$\sigma_{ij}\,\varepsilon^L_{ij} \;=\; C_{ijkl}\varepsilon^m_{kl}\varepsilon^L_{ij} \;=\; \varepsilon^m_{kl}C_{klij}\varepsilon^L_{ij} \;=\; \sigma^L_{ij}\,\varepsilon^m_{ij}\,. \tag{3.12}$$

Using Eq. (3.12), we can express the second term on the RHS of Eq. (3.11) in terms of the mechanical strains; that is,

$$\sigma_{ij}\varepsilon^L_{ij} \;=\; \sigma^L_{ij}[\varepsilon_{ij} - \alpha(\Theta - \Theta_{init})\delta_{ij}]\,. \tag{3.13}$$

Using this result and invoking equilibrium

$$\sigma_{ij,j} + b_i \;=\; 0_i \tag{3.14}$$

$$\sigma^L_{ij,j} \;=\; 0_i\,, \tag{3.15}$$

where b_i is the body force per unit volume, we obtain the desired expression for the interaction integral

$$\bar{I}(s) \;=\; \int_{V(s)} \left[\left(\sigma_{ij}\frac{\partial u^L_i}{\partial x_k} + \sigma^L_{ij}\frac{\partial u_i}{\partial x_k} \right) \frac{\partial q_k}{\partial x_j} - \sigma_{ij}\varepsilon^L_{ij}\frac{\partial q_k}{\partial x_k} + \left(\alpha\frac{\partial \Theta}{\partial x_k}\sigma^L_{ii} - b_i\frac{\partial u^L_i}{\partial x_k} \right) q_k \right] dV\,. \tag{3.16}$$

We now let $h \to 0$ to obtain the desired expression for the interaction-energy decrease when a local segment of the crack front advances by $\Delta a\, l_k$ in its plane. In deriving Eq. (3.16) we have assumed the crack faces to be traction free. It should be emphasized, that with the presence of thermal strain, the domain of integration for Eq. (3.16) must include the near-tip region ($r \to 0^+$).

The domain expression, Eq. (3.16) gives the interaction energy per unit of crack advance over a finite segment of the crack front. In order to calculate the T-stress with the help of Eq. (3.7), however, we need a local value of the interaction energy. To a first approximation this value is obtained by assuming that $I(s)$ is constant over

some region of the crack front L. This allows us to bring $I(s)$ outside the integral sign in Eq. (3.8) to yield

$$\bar{I}(s) \doteq I(s) \int_L l_k(s)\mu_k(s)ds\,. \tag{3.17}$$

or

$$I(s) \doteq \bar{I}(s) / \int_L l_k(s)\mu_k(s)ds\,. \tag{3.18}$$

Thus, once $\bar{I}(s)$ has been calculated from Eq. (3.16) using the computed stress (σ_{ij}) and deformation field (u_i) of a boundary value problem, and the exact auxiliary solution of the line load, Eq. (3.4), with unit magnitude $(f = 1)$, the local value of T-stress at the crack front point s can be determined with Eq. (3.7).

In the case of a plane strain line-crack oriented along the x_1-axis (that is, a straight crack front of length L), $\Delta\Theta(s) \rightarrow \Delta\Theta$, $T(s) \rightarrow T$, and the interaction energy in Eq. (3.18) is given by $I = \bar{I}/L$.

3.3 Finite-Element Formulation for the Domain Integral Method: Two-Dimensional Implementation

The finite-element formulation of the area/volume integral method has been discussed by Li et al. (1985) in the context of the two-dimensional biquadratic (9-node) Lagrangian element and the three-dimensional triquadratic (27-node) Lagrangian element. We outline their implementation in the context of the two-dimensional isoparametric 8-node element, for which the nodal point numbers are shown in Fig. 3.3.

The 2-D expression of the interaction integral in Eq. (3.16) is given by

$$\bar{I}(s) = \int_A \underbrace{\left[\left(\sigma_{ij}\frac{\partial u_i^L}{\partial x_1} + \sigma_{ij}^L\frac{\partial u_i}{\partial x_1}\right)\frac{\partial q_1}{\partial x_j} - \sigma_{ij}\varepsilon_{ij}^L\frac{\partial q_1}{\partial x_1}\right.}_{\bar{I}_1(s)} + \underbrace{\left.\left(\alpha\frac{\partial \Theta}{\partial x_1}\sigma_{ii}^L - b_i\frac{\partial u_i^L}{\partial x_1}\right)q_1\right]}_{\bar{I}_2(s)} dA\,, \tag{3.19}$$

where $\bar{I}_2(s)$ is the contribution to $\bar{I}(s)$ due to thermal strains and body forces.

For isoparametric elements, the coordinates (x_1, x_2) in the physical space and the displacements (u_1, u_2) are written as

$$x_i = \sum_{K=1}^{8} N_K X_{iK}\,, \qquad u_i = \sum_{K=1}^{8} N_K U_{iK}\,, \qquad i = 1,2\ , \tag{3.20}$$

where N_K are the biquadratic shape functions (see Table 3.3), X_{iK} are the nodal coordinates and U_{iK} are the nodal displacements.

Table 3.1: 2-D Shape Functions

$N_1(\eta,\zeta)$	$= (-1/4)(1-\eta)(1-\zeta)(1+\eta+\zeta)$
$N_2(\eta,\zeta)$	$= (-1/4)(1+\eta)(1-\zeta)(1+\eta+\zeta)$
$N_3(\eta,\zeta)$	$= (-1/4)(1+\eta)(1+\zeta)(1+\eta-\zeta)$
$N_4(\eta,\zeta)$	$= (-1/4)(1-\eta)(1+\zeta)(1+\eta-\zeta)$
$N_5(\eta,\zeta)$	$= (1/2)(1-\eta)(1+\eta)(1-\zeta)$
$N_6(\eta,\zeta)$	$= (1/2)(1-\zeta)(1+\eta)(1+\zeta)$
$N_7(\eta,\zeta)$	$= (1/2)(1-\eta)(1+\eta)(1+\zeta)$
$N_8(\eta,\zeta)$	$= (1/2)(1-\zeta)(1+\zeta)(1-\eta)$

In 2-D, a suitable choice for the vector $q_i, i = 1,2$ is $(q_1, q_2) = (q_1(x_1, x_2), 0)$, where

$$q_1(x_1,x_2) = \begin{cases} 1 \ \text{ on } \ S_t \\ 0 \ \text{ on } \ S_1 \\ \text{otherwise arbitrary} \end{cases}\,. \tag{3.21}$$

Consistent with the isoparametric formulation, we take q_1 within an element as

$$q_1 = \sum_{I=1}^{8} N_I Q_{1I}\ , \tag{3.22}$$

where Q_{1I} are the nodal values for the I^{th} node. From the definition of q_1, Eq. (3.21), if the I^{th} node is on S_t, $Q_{1I} = 1$, whereas if the I^{th} node is on S_1, $Q_{1I} = 0$. In the area between S_t and S_1, Q_{1I} will be taken to vary between 1 and 0. It may be noted that a particular choice of interpolation scheme for Q_{1I} is equivalent to selecting a particular weighting scheme for the field quantities between Γ and C_1. Two possible choices for q_1 are a "pyramid" function and a "plateau" function (see Fig. 3.5). For the pyramid function, $q_1 = 1$ at the crack tip, $q_1 = 0$ on the edge of the domain and q_1 varies linearly between the peak and the rectangular edges. In this sense an equal weighting has been applied to $\bar{I}_1(s)$ in Eq. (3.19) ($\partial q_1/\partial x_k$ = piecewise constant), while the thermal contributions $\bar{I}_2(s)$ have been linearly weighted. The plateau q_1 function has a value (or height) of unity everywhere in the domain except in the outermost ring of elements. Here the value (or height) decreases linearly from unity to zero within one element width. The "pyramid" and "plateau" q_1-functions, together with the virtual crack extension interpretation of q_1 as a translation in the x_1-direction, is depicted in Fig. 3.5. It may be noted that in the subdomain where q_1 is constant (corresponding to a rigid translation of the subdomain in the context of the virtual crack extension technique) there is no contribution from $\bar{I}_1(s)$ to the domain integral.

Using Eqs. (3.20) and (3.22) and the chain rule, the spatial gradient of q_1 within an element is given by

$$\frac{\partial q_1}{\partial x_j} = \sum_{I=1}^{8} \sum_{k=1}^{2} \frac{\partial N_I}{\partial \eta_k} \frac{\partial \eta_k}{\partial x_j} Q_{1I} \,, \tag{3.23}$$

where $\partial \eta_k/\partial x_j$ is the inverse Jacobian matrix of the transformation, Eq. (3.20).

With 3×3 Gaussian integration, the discretized form of the domain expression for the interaction energy for plane-strain problems is

$$\begin{aligned} \bar{I} \;=\; \sum_{\text{all elements in A}} \sum_{p=1}^{9} \Bigg\{ \Bigg[\left(\sigma_{ij} \frac{\partial u_i^L}{\partial x_1} + \sigma_{ij}^L \frac{\partial u_i}{\partial x_1} \right) \frac{\partial q_1}{\partial x_j} - \\ \sigma_{ij} \varepsilon_{ij}^L \frac{\partial q_1}{\partial x_1} + \left(\alpha \frac{\partial \Theta}{\partial x_1} \sigma_{ii}^L - b_i \frac{\partial u_i^L}{\partial x_1} \right) q_1 \Bigg] \det \left(\frac{\partial x_k}{\partial \eta_k} \right) \Bigg\}_p w_p \,. \end{aligned} \tag{3.24}$$

Here the quantities within $\{\}_p$ are evaluated at the 9 Gauss points, and w_p are the respective weights.

Eq. (3.24) has been implemented in a postprocessing program for the commercial finite-element code ABAQUS. Some of the main features of the program are discussed in the next section.

3.4 The Computer Program T-STRESS

The program T-STRESS was developed using the framework of the computer program DOMAIN (SOCRATE, 1990). Particularly the mesh topology features of DOMAIN were used.

To evaluate the interaction integral in Eq. (3.24) a domain has to be defined. It should be noted that the program T-STRESS is developed only for rectilinear meshes and is thus limited to rectangular domains. A domain is defined by a set of nodes along the symmetry line of the specimen fixing the base and thus the width of the domain, and a number of element layers fixing its height (see Fig. 3.7). Once the domain is defined, the program assigns the chosen perturbation field, plateau or pyramid, and calculates the interaction energy according to Eq. (3.24) over all elements in the domain. The T-stress is subsequently calculated using Eq. (3.7). The domain variables used in the evaluation of Eq. (3.24) are read from the ABAQUS results file. A listing of the program is given in Appendix C.

In addition to calculating the T-stress for the cases of mechanical and thermal loading, the program is equipped to calculate the J-integral using an expression similar to Eq. (3.24). Furthermore, the temperature and stress distributions can be obtained along the symmetry line of the specimen. The flow chart in Fig. 3.6 shows a rough outline of the program.

We tested our code by calculating calibration factors for T in a *SEN* specimen subjected to remote tension and bending for various crack depths. Analogously to the K_I-calibration functions $\hat{\mathbf{k}}$, i.e., $K_I = \hat{\mathbf{k}}(a,w) \cdot \mathbf{Q}$, the normalized T-stress can be expressed as $\tau = (\hat{\mathbf{t}}(a,w) \cdot \mathbf{Q})/\sigma_y$, where $\hat{\mathbf{t}}(a,w) = [\hat{t}^N, \hat{t}^M]$[1] are T-stress calibration functions of the specimen under consideration (WANG & PARKS, 1992), and w is the width of the specimen. $\mathbf{Q}$ (components = $[N, M]$) is the vector of generalized load amplitudes with work conjugate displacements $\mathbf{q}$ (RICE, 1972). Using second-order weight functions, SHAM (1991) has tabulated values for the T-stress calibration functions for various specimens over essentially the entire range of relative crack length a/w ($0.1 \leq a/w \leq 0.9$).

Fig. 3.8 shows the results obtained with T-STRESS compared to SHAM's data. The agreement is exceptional for all relative crack depths. The two curves practically coincide. Domain independence obtained with T-STRESS was checked by comparing our J-integral values to the J-integral values provided by ABAQUS. The results for six different domains (six different contours in the case of ABAQUS), normalized by $\sigma_\Theta^2 a/E$, where $\sigma_\Theta = \alpha E(\Theta_{init} - \Theta_{amb})/(1-\nu^2)$, for a *SEN* specimen of relative crack depth $a/w = 0.1$ are given in Table 3.2. The smallest domain contains two elements adjoining the crack tip; the second domain, which includes the first domain and the adjoining layer of elements, contains eight elements. Domains three through six are also assembled in this fashion. The variation of J over the six domains is less than 2%, an indication of the overall accuracy of the calculation.

[1]The superscripts N and M denote tension and bending, respectively.

Table 3.2: Domain independence of J-integral.

Domain	1	2	3	4	5	6
J-integral ABAQUS	1.274018E-03	1.291190E-03	1.292163E-03	1.292200E-03	1.292207E-03	1.292207E-03
J-integral T-STRESS	1.274018E-03	1.291190E-03	1.292163E-03	1.292200E-03	1.292207E-03	1.292207E-03

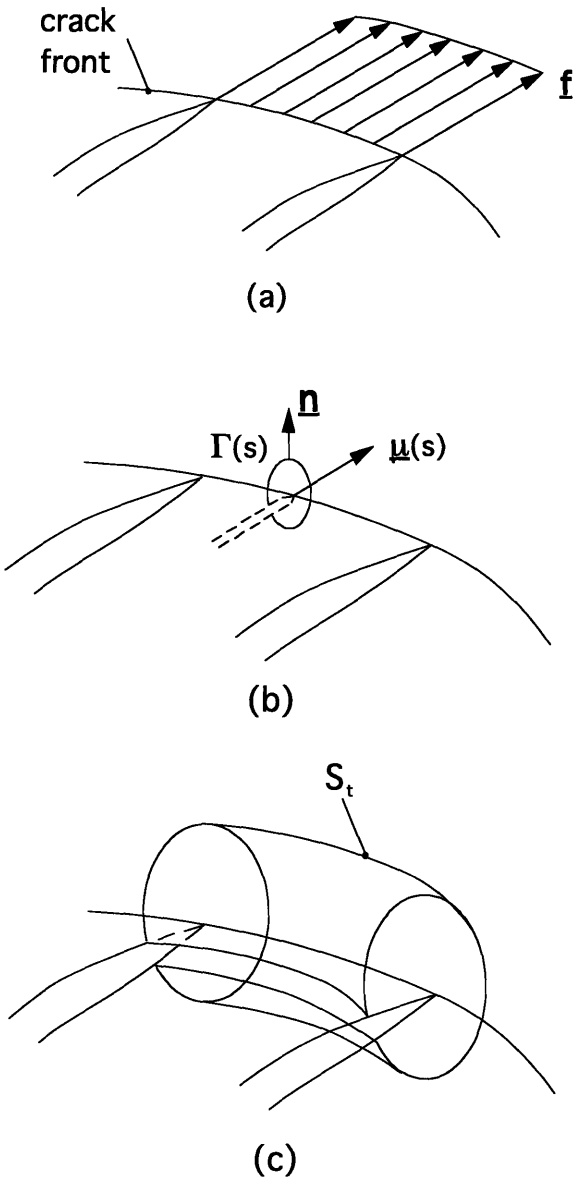


Figure 3.1: (a) Line-load applied in the direction of crack advance along the crack front. (b) Crack tip contour Γ on the plane locally perpendicular to the crack front where s represents the location of the crack tip. (c) Tubular surface S_t enclosing the crack-front segment.

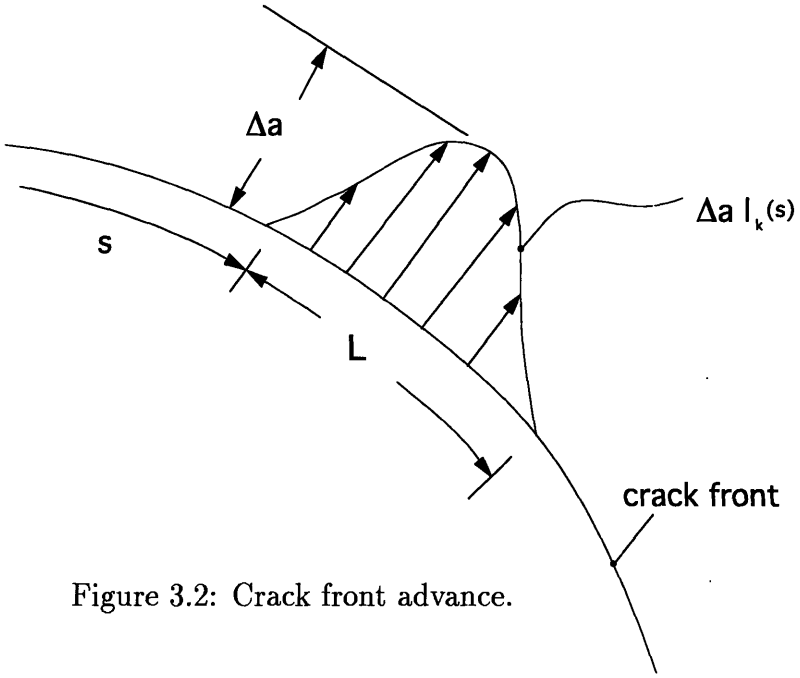


Figure 3.2: Crack front advance.

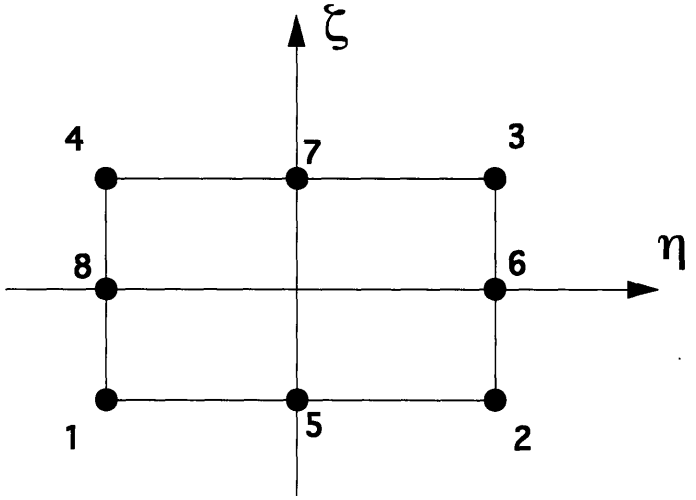
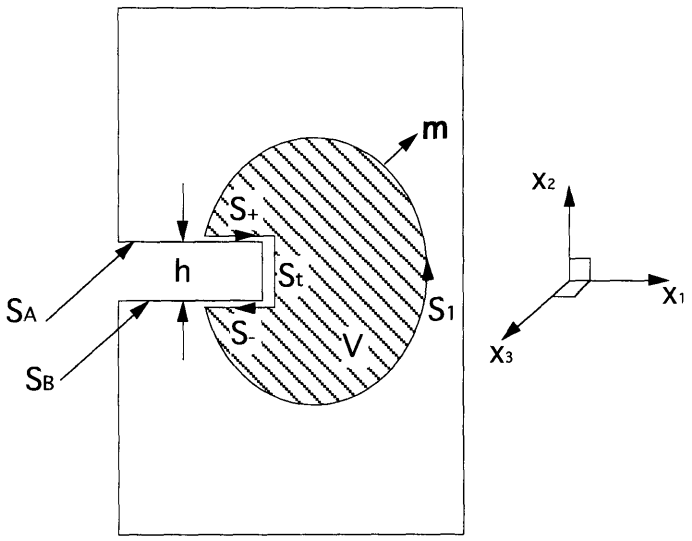
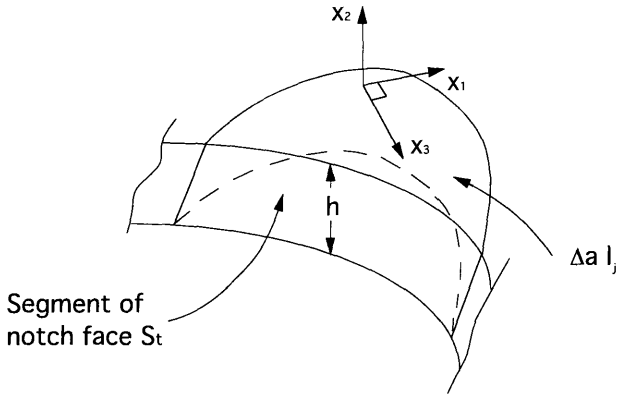


Figure 3.3: Nodal point numbers of 2-D isoparametric 8-node element.

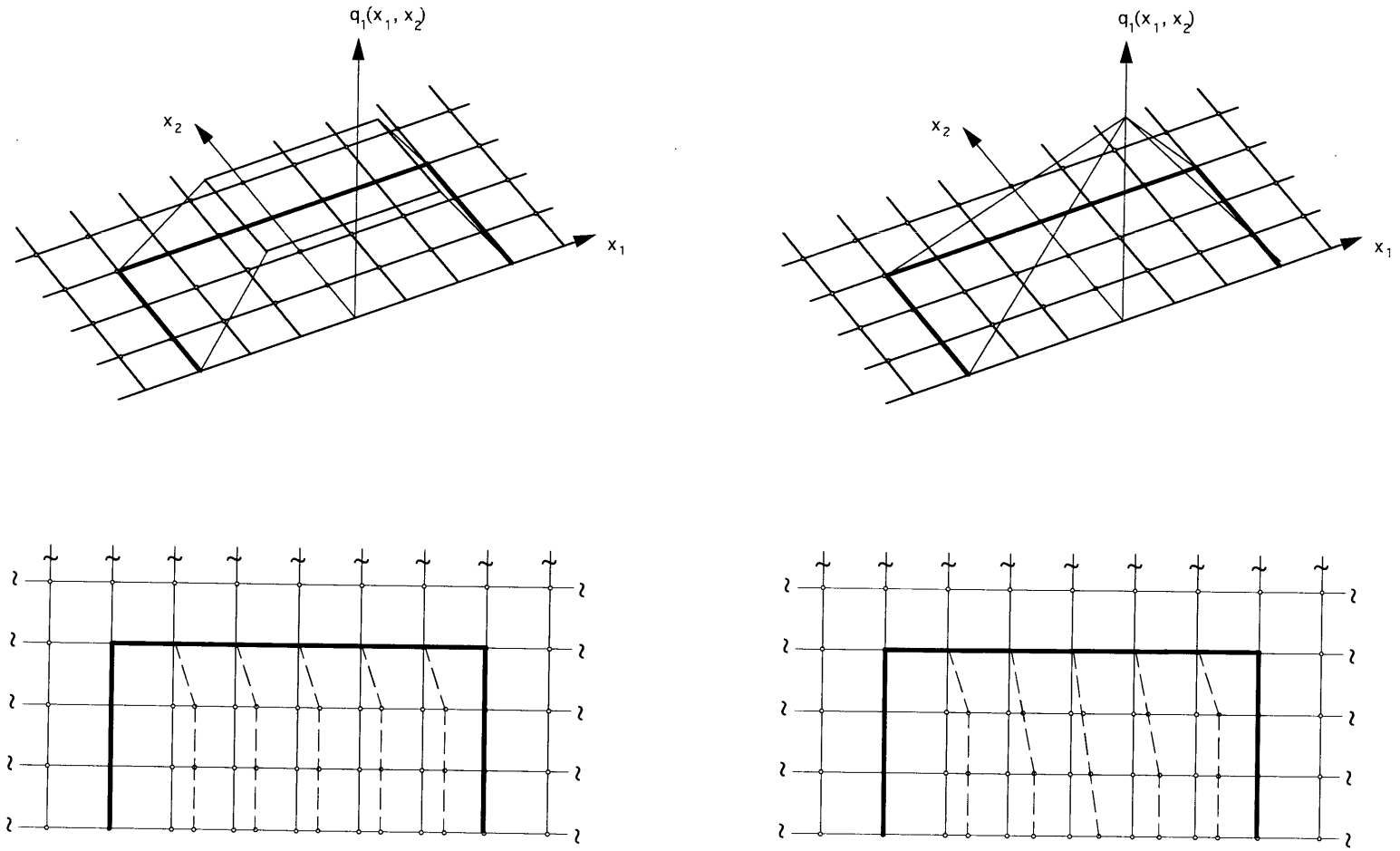


(a)



(b)

Figure 3.4: (a) Schematic of section of body and volume V in the $x_1 - x_2$ plane containing a notch of thickness h. (b) Schematic of notch face when the function $\Delta a l_j$ is interpreted as a virtual advance of a notch face segment in the direction normal to x_2.



(a) Plateau function.

(b) Pyramid function.

Figure 3.5: Pyramid and plateau functions.

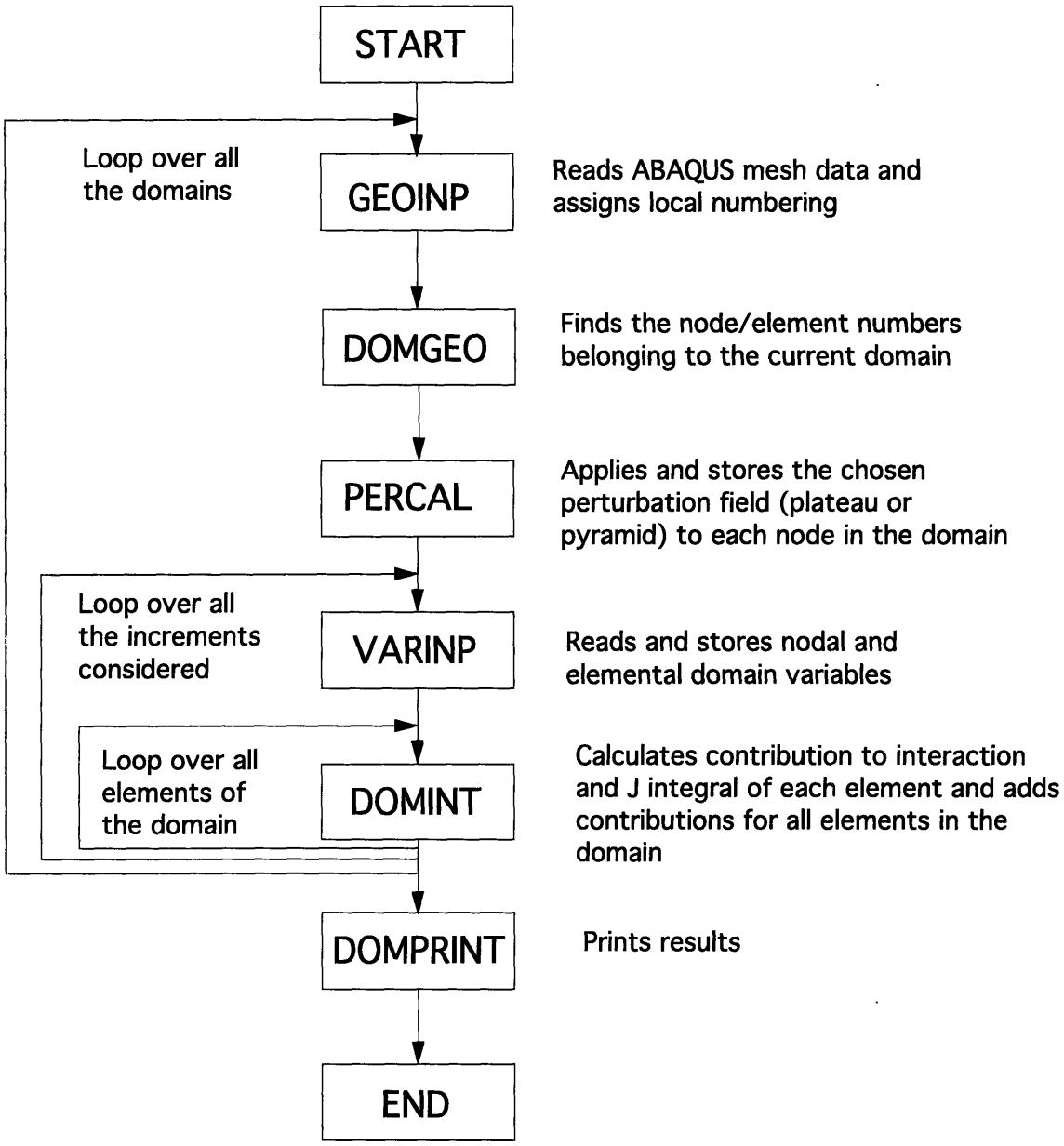


Figure 3.6: Flow chart of the program T-STRESS.

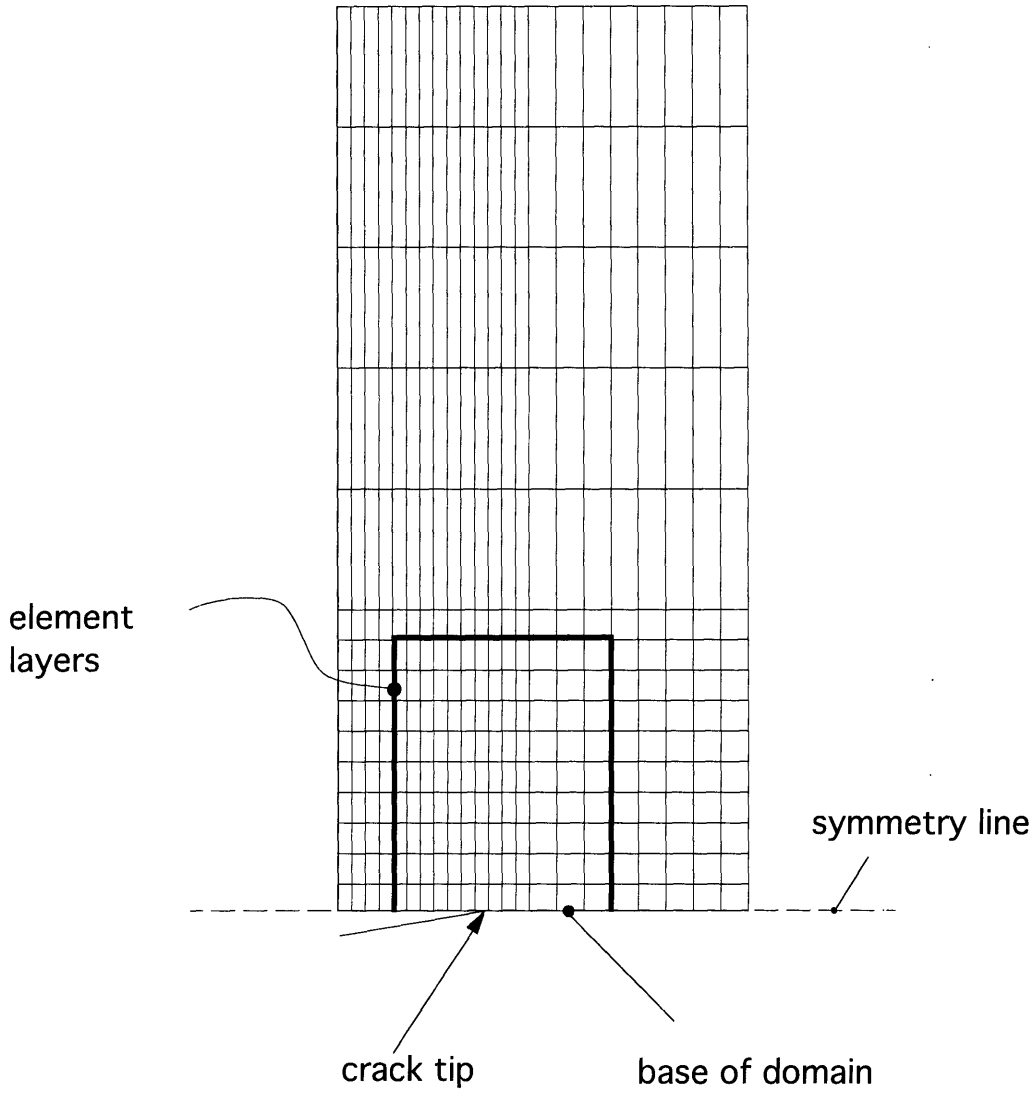


Figure 3.7: Domain definition.

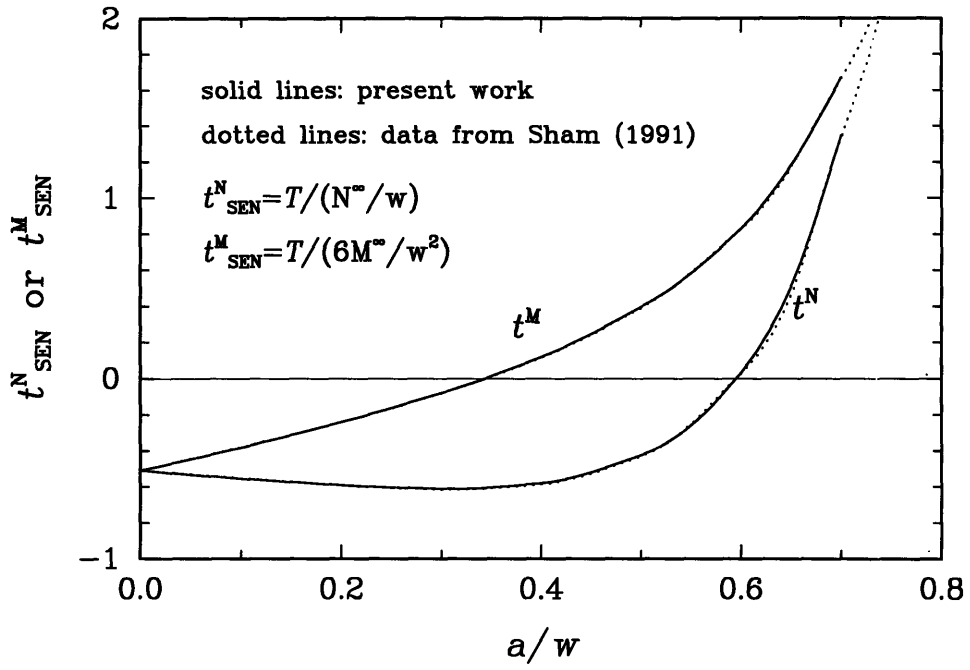


Figure 3.8: Normalized T-variation with respect to the crack depth in *SEN* specimen under remote tension or bending.

Chapter 4

T-Stress due to Thermal Transients

4.1 Introduction

Chapter 2 showed that the T-stress (especially: negative T-stress) has a strong effect on the near crack-tip fields. Negative T-stresses ($\tau < 0$) are associated with a substantial reduction in crack-tip stress triaxiality (compared to SSY), while positive T-stresses result in only modest elevation of triaxiality above SSY. HANCOCK & co-workers (1991) have shown that the variation of stress triaxiality in various plane-strain specimens can be adequately predicted by introducing T as the constraint parameter, even up to large scale yielding. WANG (1991) verified the two-parameter characterization of elastic-plastic crack-tip fields (J and T) with a 3D study of stress fields along the crack fronts of *SCP*s.

Given the numerical tools described in Chapter 3, we will examine/extend the two-parameter characterization for the case of transient thermal loading. Our approach is as follows: First we evaluate the variation of the T-stress during a thermal transient. Plane-strain elastic finite-element analyses for single edge-cracked specimens of varying crack depths are carried out and post-processed with the program T-STRESS for

this purpose. The severity of the thermal shock is varied by varying the Biot number. Next, we repeat the analysis for one of the cases assuming elastic-plastic material behavior. Specifically, we are interested in the crack-opening stress profiles at various instances of time during the thermal shock. Having obtained values for T and K_I as a function of time from the elastic analysis of the problem, we are now in the position to make predictions for the stress state using WANG's MBL solutions. If the two-parameter characterization holds, the MBL solutions will qualitatively predict the corresponding elastic-plastic results. We finally test the validity of the two-parameter characterization by analyzing an edge-cracked strip subjected to both thermal and mechanical loads.

4.2 Problem Statement

The problem of interest is depicted in Fig. 4.1. Consider the edge-cracked strip of width w and crack length a. Unit thickness in the plane is assumed. The entire strip is initially at temperature Θ_{init} and is perfectly insulated along the plane $x = w$. At time $t = 0$ the surface is suddenly subjected to Newtonian convective cooling while the surrounding temperature is kept at (ambient) temperature Θ_{amb}. The thermal conductivity of the material is k, and the heat transfer coefficient of the fluid/solid interface is h. The strip is assumed to be infinite, thus resulting in one-dimensional temperature distributions at any instant of time.

We will assume that the resulting transient thermal stress problem is quasi-static; that is, the inertia effects are negligible. A number of studies on dynamic thermoelasticity have validated this assumption (see, for example STERNBERG & CHAKRAVORTY, 1959a,b). Thermoelastic coupling effects and temperature-dependence of thermoelastic constants are also neglected.

The material considered is ASTM A710 steel having a Young's modulus of

$E = 207\,GPa$ (see Fig. 2.4 for tensile stress/strain curve), density $\rho = 7832\,kg/m^3$, specific heat $c = 0.6\,kJ/kg^oC$, thermal conductivity $k = 58.8\,W/m^oC$, and Poisson's ratio $\nu = 0.3$. For the elastic-plastic finite-element analyses, the material was modeled as isotropic, obeying J_2 flow theory plasticity. Small geometry changes were assumed. The flow strength was given as a function of the equivalent plastic strain, with an initial value of $\sigma_y = 470\,MPa$ and a saturation value of $677\,MPa$ at plastic strain $\varepsilon^p = 0.0538$, which essentially corresponds to a strain hardening exponent of $n = 10$.

4.3 Elastic Analysis

4.3.1 Methods

Taking advantage of the assumptions discussed in Section 4.2, the problem can be analyzed in two parts. First, the temperature distribution in the material is determined as a function of time. This temperature distribution is then used as input for the subsequent stress analysis of the problem to obtain the transient fields at the crack tip.

The mesh used in the finite-element analysis for $a/w = 0.1$ is shown in Fig. 4.2. The problem is symmetric about the $y = 0$ line; therefore, only half of the strip needs to be modeled. The crack-tip region is modeled by a rectangular domain for post-processing with T-STRESS. The inset portion of the mesh contains 32 elements across the width and 16 elements along the height. 8-node heat transfer elements were used for the temperature analysis, 8-node plane-strain full integration elements for the subsequent stress analysis of the problem (ABAQUS element types DC2D8 and CPE8, respectively). Finite-element analyses were performed on the *SEC* strip at five different crack depths ($a/w = 0.0375$, 0.05, 0.1, 0.3, and 0.4) and Biot numbers ($\beta = 1$, 5, 10, 20, and 100).

For the temperature analysis of the problem, the minimum usable time step was selected according to the equation

$$\Delta t \geq \frac{\rho c}{6k}\Delta l^2 , \tag{4.1}$$

where Δt is the time increment, ρ is the density, c is the specific heat, k is the thermal conductivity, and Δl is a typical element dimension (such as the length of a side of an element). If time increments smaller than this value are used, spurious oscillations may appear in the solution (ABAQUS, 1992).

4.3.2 Results

The temperature distributions for the cases $\beta = 5$ and $\beta = 100$ for various values of nondimensional time Fo, also known as the Fourier number, are shown in Figs. 4.3 and 4.4, respectively. The Fourier number is given as $Fo = Dt/w^2$, where $D = k/\rho c$ is the thermal diffusivity. Also shown in the Figure are the corresponding analytical solutions (see Appendix B for derivation). The finite-element results match the analytical predictions very well, an indication of the adequacy of the mesh design. Clearly, as the Biot number decreases, the thermal gradient through the strip becomes less severe. Accordingly, the thermal stresses and thus the stress intensity factors decrease, as can be seen in Figs. 4.5 through 4.9. The analytical predictions of NIED (1983) for K_I are shown in Fig. 4.7 for the purpose of comparison. The stress intensity factors were obtained from the elastic identity, Eq. (1.3), using K_I from the J-integral values calculated with T-STRESS ($K_I = \sqrt{E'J}$) and normalized as

$$K_I^* \equiv \frac{K_I}{E\alpha(\Theta_{init} - \Theta_{amb})\sqrt{\pi a}/(1-\nu)} . \tag{4.2}$$

For any given crack length ratio a/w, the nondimensional stress intensity factor increases, passes through a maximum, and then decreases as a function of Fourier number. The Biot number strongly controls the maximum stress intensity factor during

the thermal shock. Also greatly affected by the Biot number is the nondimensional time at which this maximum is reached. The lower the Biot number, the later the peak in K_I^*.

Nondimensional stress intensity factors for various crack lengths at fixed Biot number ($\beta = 100$) are shown in Fig. 4.10. As would be expected in a self-equilibrating stress field, the maximum normalized stress intensity factor is of greatest magnitude for a very short crack, decreasing with increasing crack length. Thus, decreasing the Biot number has an effect on K_I^* which is similar to increasing the crack length.

The variation of the nondimensional T-stress values during the thermal shock for the five different crack lengths is shown in Figs. 4.11 through 4.15. The T-stress values were calculated with T-STRESS according to Eq. (3.7) and normalized as

$$T^* \equiv \frac{T}{E\alpha(\Theta_{init} - \Theta_{amb})}. \tag{4.3}$$

Again, the strong influence of the Biot number in controlling the maximum values can be observed. More interesting, however, is how differently the T-stress values evolve for the various crack depths. This difference can be observed in Fig. 4.16, which shows the nondimensional T-stress values for the five crack depths at fixed Biot number ($\beta = 100$). For the short crack depths, the T-stress passes through a positive maximum value early during the transient, before rapidly becoming negative. Following this rapid "dip", the T-stress reaches a maximum negative value, and finally returns to zero at the end of the transient. As the crack length increases, the time at which the transition into the negative regime occurs is more and more delayed. Along with this trend, the initial maximum positive T-stress value increases, and the maximum negative values become less negative. For the case of relative crack depth $a/w = 0.1$, for example, the T-stress becomes only slightly negative towards the end of the transient. For even longer cracks, the maximum positive T^*-value finally occurs so late in the transient and is so "positive", that a transition into the negative regime does not take place at all. The relative decrease in maximum positive T-stress value

between the cases of relative crack depth $a/w = 0.3$ and $a/w = 0.4$ should be noted. This decrease seems to be a feature of the self-equilibrating nature of the stress fields under consideration.

In order to explore the differences in the evolution of the T-stress for the various crack depths, we examined the position of the temperature front relative to the crack tip at the occurrence of maximum T^*s for the various crack depths. We define a nondimensional penetration depth, δ^*, which relates the position of the temperature front to the relative crack depth of the specimen as $\delta^* \equiv (\delta/w)/(a/w)$. The location of the temperature front itself (δ) is determined by $(\Theta_{init} - \Theta(x = \delta))/(\Theta_{init} - \Theta_{amb}) = 0.01$. Figs. 4.17, 4.18, and 4.19 show the results for $a/w = 0.0375$, 0.05, and 0.1 for three different values of the Fourier number corresponding to the occurence of the positive maximum value of T, the zero point, and the maximum negative value, respectively. Fig. 4.20 shows the position of the temperature front relative to the crack tip for $a/w = 0.3$ and $a/w = 0.4$ at the respective maximum values. For the cases $a/w = 0.0375$, 0.05, and 0.1, the maximum negative value occurs when the temperature front has passed the crack tip by a distance approximately ten times its relative crack depth ($\delta^* \sim 10$). The maximum negative value of T^* for $a/w = 0.1$ is reached when the temperature front has practically arrived at the backface of the specimen. The maximum positive value for all cases occurs when the temperature front has passed the crack tip by a distance approximately half the relative crack depth of the specimen. Given the fact that the maximum negative T-stress value for the shallow crack cases occurs when the temperature front has passed the crack tip by a distance several times its depth, together with the observation that the Biot number controls the magnitude but not the sign of the T-stress, we identify the relative crack depth as the controlling parameter of the problem. That is, the position of the crack tip relative to the surface of the specimen determines whether the specimen sees negative T-stress values during the thermal transient. In terms of the stress fields under consideration, the relative proximity of the crack-tip stress fields to the surface

of the specimen, and thus the interaction of the crack-tip stress fields with the surface stresses, defines the evolution of the T-stress during the thermal transient.

Loci of K_I^* *vs.* T^* are shown in Figs. 4.21 through 4.25. Arrows indicate the direction of traverse during the transient. For any given relative crack depth, the loci are self-similar for the various Biot numbers. The K_I^* - T^* loci for the five crack depths at fixed Biot number ($\beta = 100$) are shown in Fig. 4.26. The effect of decreasing crack depth is a rotation of the loci about the origin towards the second quadrant. For the cases $a/w = 0.0375$ and $a/w = 0.05$, the maximum value of K_I^* occurs at a negative value of T^*. This result is important in view of the influence of negative T-stress values on the near-crack-tip stress fields. If the two-parameter characterization holds for transient thermal loading, the crack-opening stress profiles of the corresponding elastic-plastic solutions will be lower than predicted by the HRR fields for these cases. We will further investigate this aspect in the following section.

From these results it is interesting to finally plot the variation of T^* corresponding to the maximum value of K_I^* *vs.* relative crack depth (see Fig. 4.27). Again, the strong influence of the Biot number is notable. The stongest relative variation of T^* occurs for $\beta = 100$. For this case, the T^*-value at maximum K_I^* starts from -0.168 for $a/w = 0.0375$ and reaches 0.282 for $a/w = 0.4$. The variation in T^* for $\beta = 1$ is notably less pronounced, as the T^*-value for $a/w = 0.0375$ starts at -0.036 and reaches 0.057 for $a/w = 0.4$. Clearly, the strong effect of both high Biot numbers and low relative crack depths, resulting in negative T^*-values, has not saturated at $a/w = 0.0375$. Based on the results of HARLIN & WILLIS (1988), we expect the asymptotic T^*-value ($a/w \to 0$) to occur at $T^* \sim -0.5$.

4.4 Elastic-Plastic Analysis

4.4.1 Methods

Given the results from the elastic analysis of the problem, we select the strip of relative crack depth $a/w = 0.0375$ at $\beta = 100$ to examine the validity of the two-parameter characterization. This case was chosen because it exhibited the most negative of the T-stress values. Specifically, we are interested in the crack-opening stress profiles at four instances in time during the thermal transient. If the two-parameter characterization holds, the MBL solutions will qualitatively predict the corresponding elastic-plastic results. The four instances in time and their corresponding T-stress values are indicated in Fig. 4.28. The T-stress values were normalized as $\tau = T/\sigma_y$ for comparison with the corresponding MBL results. These values were chosen in order to examine four distinct variations in crack-opening stress profiles. The value $\tau = -0.221$ corresponds to the maximum value of K_I^* (see Fig. 4.29). We did not consider any times beyond that of K_{Imax}^*, as unloading will occur after this point.

The mesh used in the analysis is shown in Fig. 4.31. In comparison to the mesh used in the elastic analysis of the problem (Fig. 4.2), the near-tip region is modeled by the inset shown in Fig. 4.32. There were 32 fans of elements circumferentially, and 24 rings radially. The ratio of the radius of the outer boundary to the radius of the first ring of elements was on the order of 10^3. The first ring of elements were degenerated, so one side collapsed into a single point at the crack tip. Again, 8-node heat transfer elements were used for the temperature analysis (DC2D8) of the problem. Hybrid 8-node plane strain reduced integration elements were used (ABAQUS element type CPE8HR) in the subsequent stress analysis. Hybrid elements were used, as we observed oscillations in the crack-opening stress profiles typical of mesh locking with full integration elements (CPE8) (NAGTEGAAL et al., 1974).

4.4.2 Results

Fig. 4.33 shows the variation of normalized crack-opening stress (σ_{22} at $\theta = 0$) *vs.* normalized distance from the tip at the four values of normalized T-stress. An enlarged view of Fig. 4.33 is shown in Fig. 4.34. Also shown are WANG's MBL solutions. WANG's results are given in the range $|\tau| \leq 1.0$ in intervals of 0.1. The stresses were taken from extrapolated stresses at the nodes on the line $\theta = 0$, and the radial distance r from the crack tip was calculated from the nodal coordinate input files. The thick solid line at $\tau = 0$ is the stress profile at SSY. The stresses marked by the big circles are the HRR singularity fields calculated from Eq. (1.5) using the field constants given by SHIH (1983). Stresses inside the blunted zone $r <\sim J/\sigma_y$ should be ignored because the present small-strain analysis does not account for the finite strain inside the blunting zone. Clearly, the stress profiles for the four τ-values behave in the same way as the corresponding MBL solutions. Substantial stress reduction is seen for the two negative values of τ, and moderate stress elevation occurs at $\tau = 0.145$. However, the stress profiles of the four τ values seem to be slightly rotated compared to the MBL results. That is, for a distance $r > 2J/\sigma_y$ the normalized crack-opening stress decreases more gradually compared to the MBL results. In this respect they agree with the HRR fields, which also exhibit this gradual decrease beyond $r > 2J/\sigma_y$. Neglecting this relative rotation of the four stress profiles, which may be an effect of the out-of-plane mechanical strain, good agreement with the corresponding MBL solutions can be noted. The MBL solutions predict the corresponding elastic-plastic results well. The agreement is remarkable considering that the MBL loading is based on the first two terms of the WILLIAMS eigen-expansion, which neglects the presence of thermal strains in its derivation. It should also be noted that the stress/strain relationship used in the present work is only an approximation to the one WANG (1991) used in his analysis.

The variation of the four crack-opening stress profiles with respect to τ at a fixed dis-

tance $r/(J/\sigma_y) = 2$ from the crack tip is shown in Fig. 4.35. Also shown are WANG's MBL solutions. Again, our results are slighty rotated compared to WANG's results. Early during the transient, the crack-opening stresses are highest, corresponding to a value of $\tau = 0.145$. By the time the maximum value of the stress intensity factor is reached, τ has decreased to -0.221, and the crack-opening stresses have dropped considerably below those predicted by the HRR singularity.

Fig. 4.36 shows the circumferential variation of the hydrostatic stress at a fixed radial distance, $r = 1.22J/\sigma_y$, for the four τ-values and the corresponding MBL results. The thick solid line is the SSY solution ($\tau = 0$). Again, the elastic-plastic results show good agreement with the corresponding MBL results. The hydrostatic stresses decrease with respect to the SSY solution for the two negative values of τ and increase slightly for $\tau = 0.145$.

Fig. 4.37 shows the circumferential variation of normalized equivalent strain (ε^p) at $r = 1.22J/\sigma_y$ in comparison to the MBL solutions. The thick solid line indicates the SSY solution. Qualitatively, the strain profiles of the elastic-plastic analysis agree with the corresponding MBL results. For the two negative τ-values, a large increase in ε^p and a shift of the peak to the forward section ($\theta < 90^o$) can be observed. For $\tau = 0.145$, the peak ε^p-value slightly shifts backwards ($\theta > 90^o$). However, the maximum values of the thermal shock problem are approximately 10% higher than those predicted by the corresponding MBL results. This difference is acceptable considering the difference in nature of the strain fields under consideration.

We examine the strain components of the transient strain field as a function of normalized distance ahead of the crack in Fig. 4.38 ($\theta = 0^o$). Fig. 4.38 (a) shows the variation of equivalent plastic strain, ε^p, Fig. 4.38 (b) the "equivalent elastic" strain, σ_e/E, where σ_e is the equivalent Mises stress, Fig. 4.38 (c) the hydrostatic strain, ε_{kk}, and Fig. 4.38 (d) the variation of thermal strain, $\alpha(\Theta - \Theta_{init})$, respectively. The strain components are each normalized by the strain at yield, ε_y. As expected,

the variation of equivalent plastic strain is strongly singular at the crack tip, but the remaining strain components are bounded. Both, the "equivalent elastic" strains and the hydrostatic strains are finite at the crack tip due to the imposed nonhardening response for large strains (see Fig. 2.4). The hydrostatic strains decrease with decreasing τ, consistent with the behaviour of the hydrostatic stresses. The thermal strains are practically constant ahead of the crack tip, indicating only a slight temperature variation in the range $0 < r/(J/\sigma_y) < 6$.

4.5 Combined Thermal and Mechanical Loading

4.5.1 Methods

To further investigate the validity of the two-parameter approach we consider the case of combined thermal and mechanical loading. A pressure vessel in a nuclear reactor could see such loading conditions during a small-scale loss of coolant accident (LOCA), for example (also termed a "pressurized thermal shock"). We apply a fixed tensile traction equal to half the tensile yield strength to the ends of the edge-cracked strip during the thermal shock for this purpose. In the analysis we proceed in the same way as for the case of purely thermal loading. First, we evaluate the variation of the T-stress during the thermal transient from the elastic finite-element solution of the problem. Next, we repeat the analysis assuming elastic-plastic material behavior to obtain the variation of the crack-opening stress profiles during the transient and compare our results to the corresponing MBL solutions. For the same reasons as before, we focus our attention on the case of relative crack depth $a/w = 0.0375$. For a pressure vessel of wall thickness 20 cm this corresponds to a crack depth of 0.75 cm, which is both significant and reasonable.

4.5.2 Results

The variation of the stress intensity factor and the T-stress during the thermal transient for the combined loading case are shown in Figs. 4.39 and 4.40. The presence of the mechanical loads cause the two curves to shift upwards and downwards, respectively. These results are not surprising, as we already observed negative T-stress values for the shallow cracks of the edge crack specimen under pure tension (Fig. 3.8). In terms of the $T - K_I$ locus, the additional mechanical load results in a translation of the entire locus into the second quadrant. That is, in the presence of the superposed mechanical loads, negative T-stress values are seen by the strip for essentially the full duration of the transient. Thus, assuming the two-parameter approach is valid also for the case of combined thermal and mechanical loading, we expect the crack-opening stress profiles of the corresponding elastic-plastic solutions to be considerably lower than those predicted by the HRR singularity.

Again, we chose four instances in time at which we obtain the crack-opening stress profiles from the elastic-plastic analysis of the problem. The four instances in time and their corresponding T-stress values are indicated in Fig. 4.40. The value $\tau = -0.49$ corresponds to the maximum value of K_I (see Fig. 4.41).

The variation of normalized crack-opening stress (σ_{22} at $\theta = 0$) *vs.* normalized distance at the four τ values is shown in Fig. 4.42. An enlarged view of Fig. 4.42 is shown in Fig. 4.43. The stress profiles exhibit the expected drop predicted by the MBL solutions. That is, the two-parameter characterization also holds for the case of combined thermal and mechanical loading. The slight rotation of the stress profiles compared to the MBL results was already discussed before.

For completeness, we present the variation of hydrostatic stress, plastic equivalent strain, and the various strain components also for the combined loading case. Fig. 4.45 shows the circumferential variation of the hydrostatic stress at a fixed radial distance,

$r = 1.22J/\sigma_y$, for the four τ values and the corresponding MBL solutions. The elastic-plastic values agree well with the MBL results. During the decreasing-τ portion of the transient (after $\tau = \tau_{max}$) the hydrostatic stresses decrease with respect to the SSY solution as expected.

Fig. 4.37 shows the circumferential variation of normalized equivalent strain (ε^p) at $r = 1.22J/\sigma_y$ in comparison to the MBL solution. As before, qualitative agreement of the elastic-plastic strain profiles with the corresponding MBL solutions and a slight overestimation ($\sim 10\%$) can be noted.

The normalized radial variation of strain components of the transient strain field ($\theta = 0$) subjected to combined loading are shown in Fig. 4.47. The strain measures at the four τ-values are similar in magnitude, compared to the results for purely thermal loading.

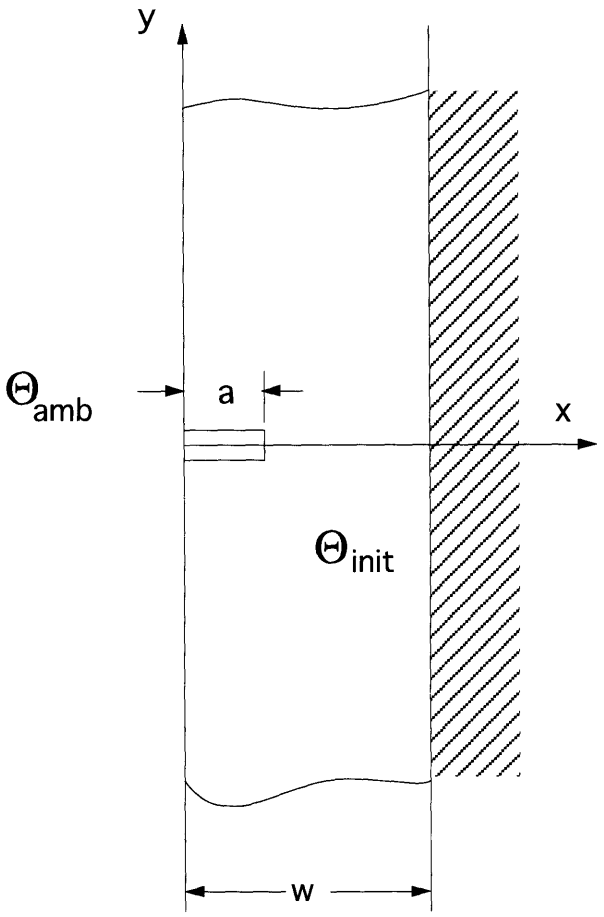


Figure 4.1: Strip with edge crack, subjected to thermal shock along $x = 0$.

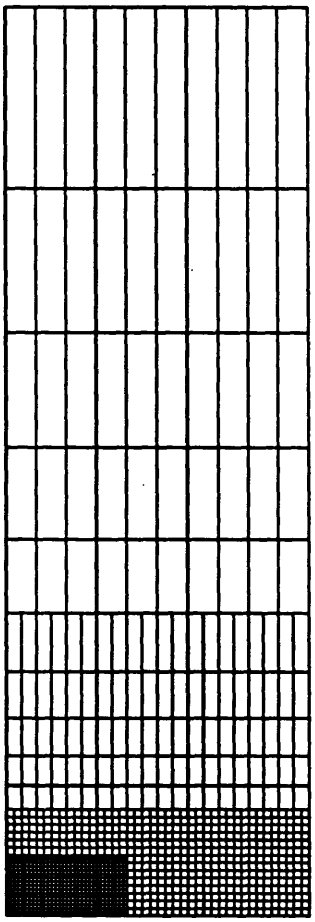


Figure 4.2: Mesh used in elastic analysis of the problem.

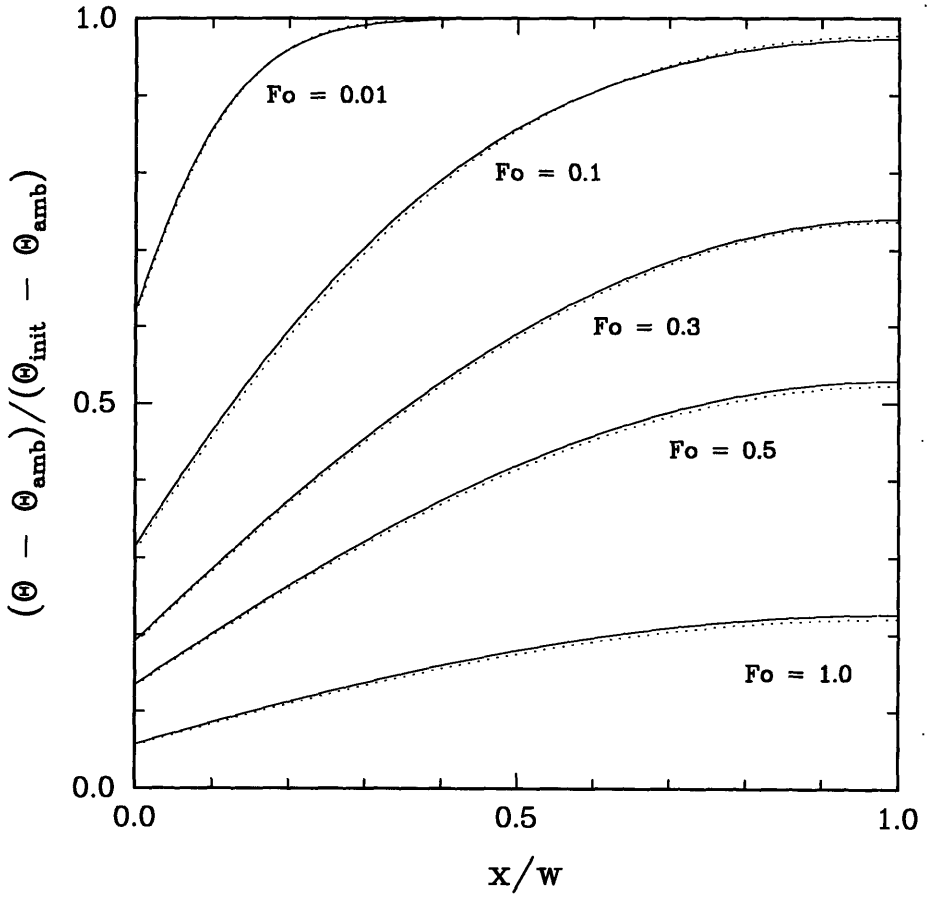


Figure 4.3: Transient temperature distribution in the strip for $\beta = 5$ ($Fo = Dt/w^2$).

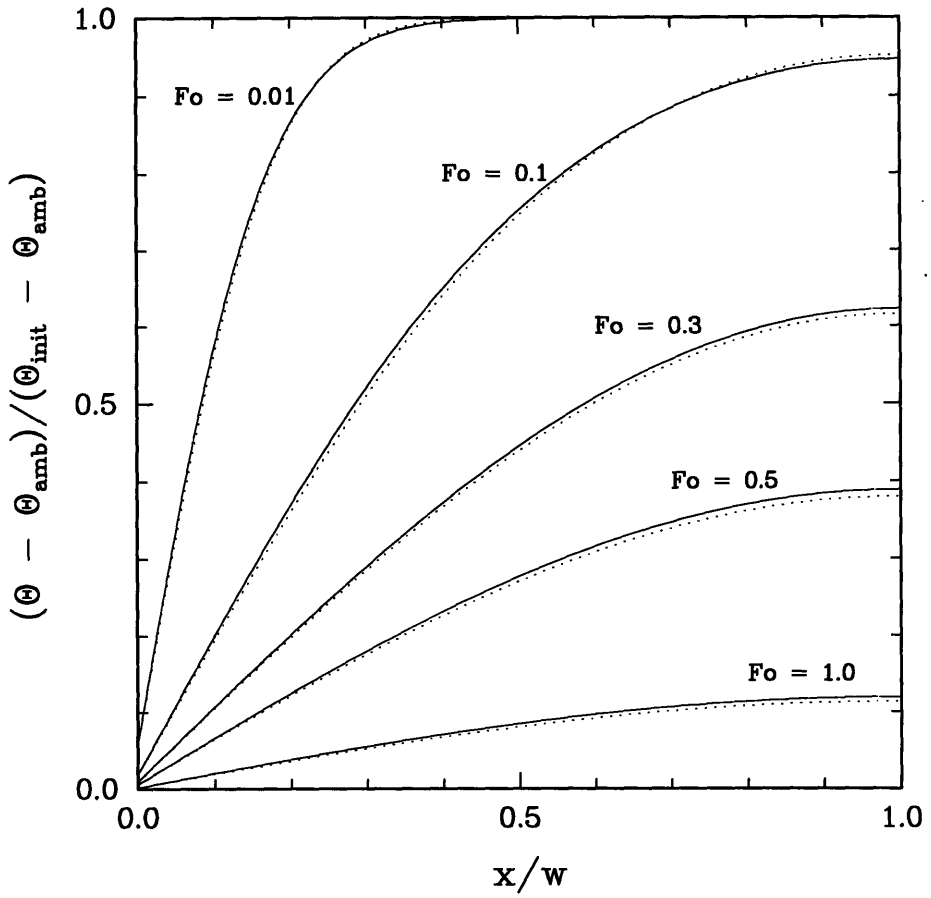


Figure 4.4: Transient temperature distribution in the strip for $\beta = 100$ ($Fo = Dt/w^2$).

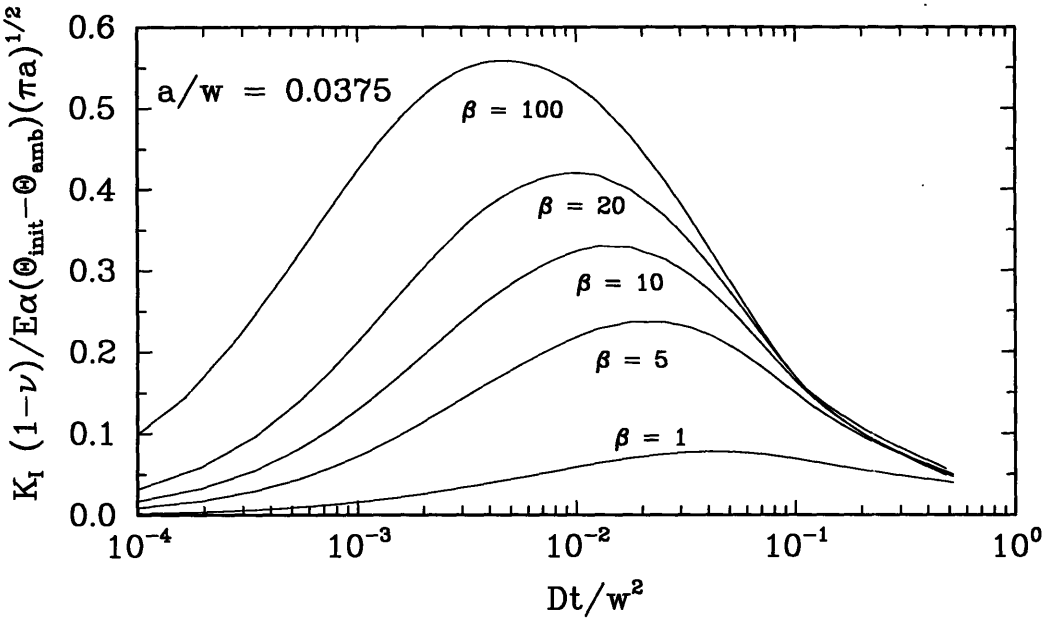


Figure 4.5: Stress intensity factors K_I^* for $a/w = 0.0375$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

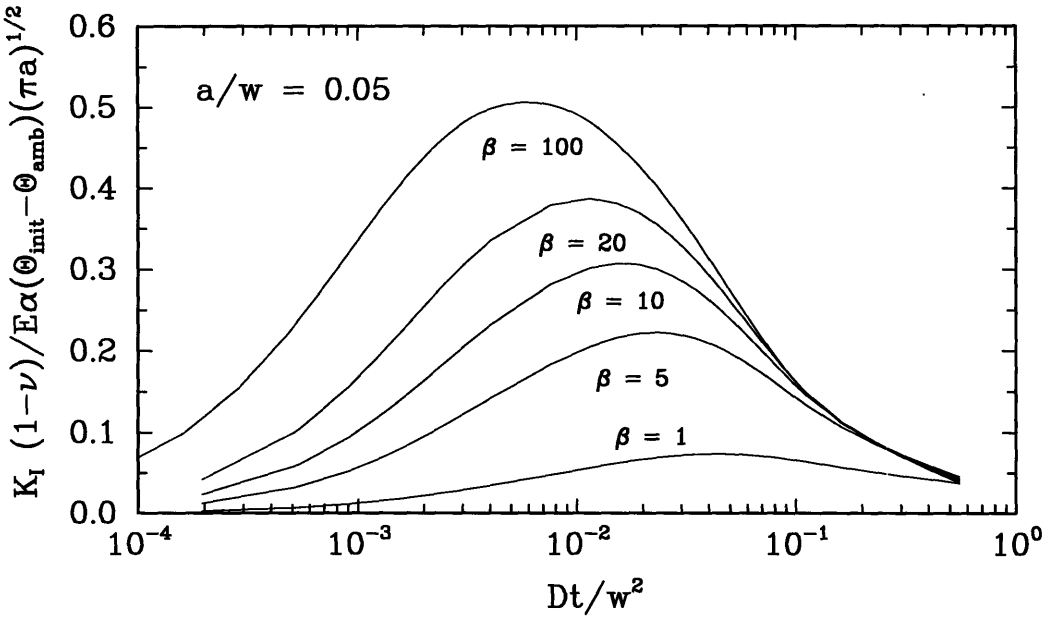


Figure 4.6: Stress intensity factors K_I^* for $a/w = 0.05$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

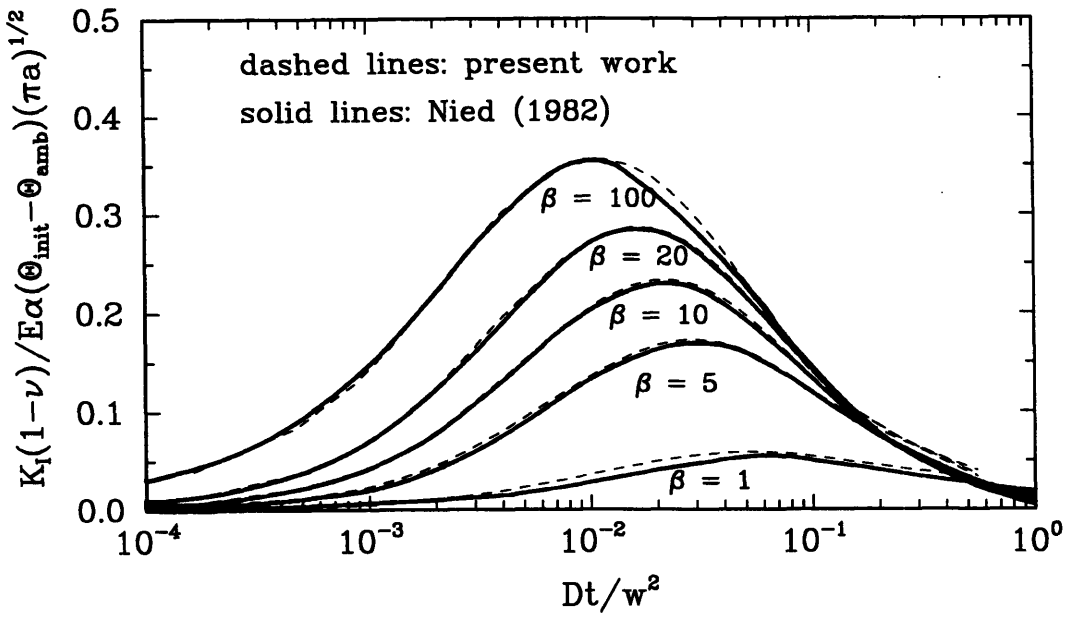


Figure 4.7: Stress intensity factors K_I^* for $a/w = 0.1$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

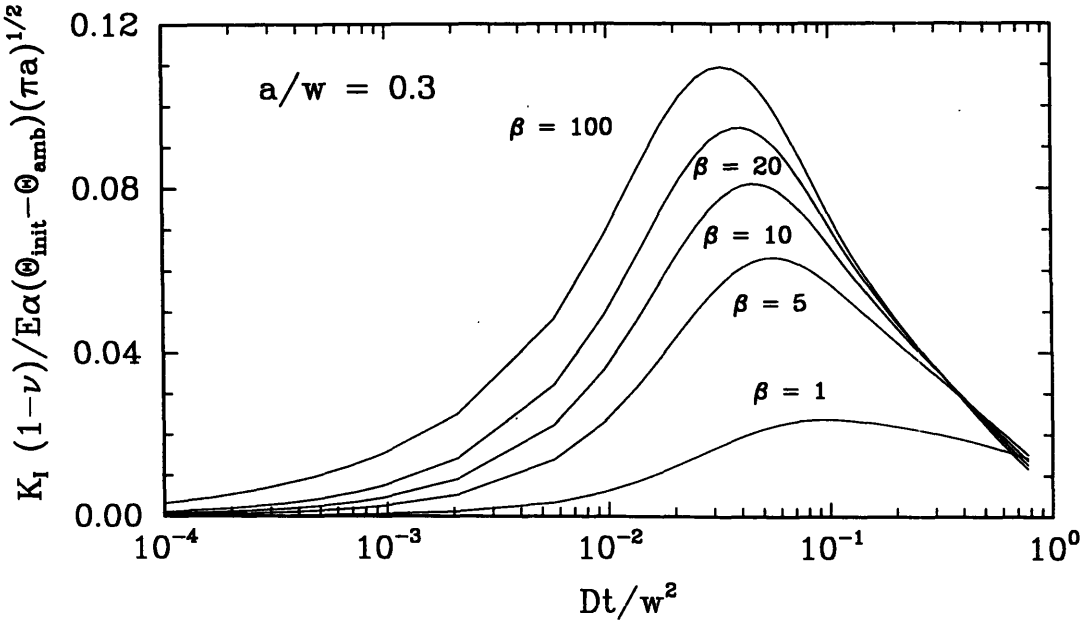


Figure 4.8: Stress intensity factors K_I^* for $a/w = 0.3$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

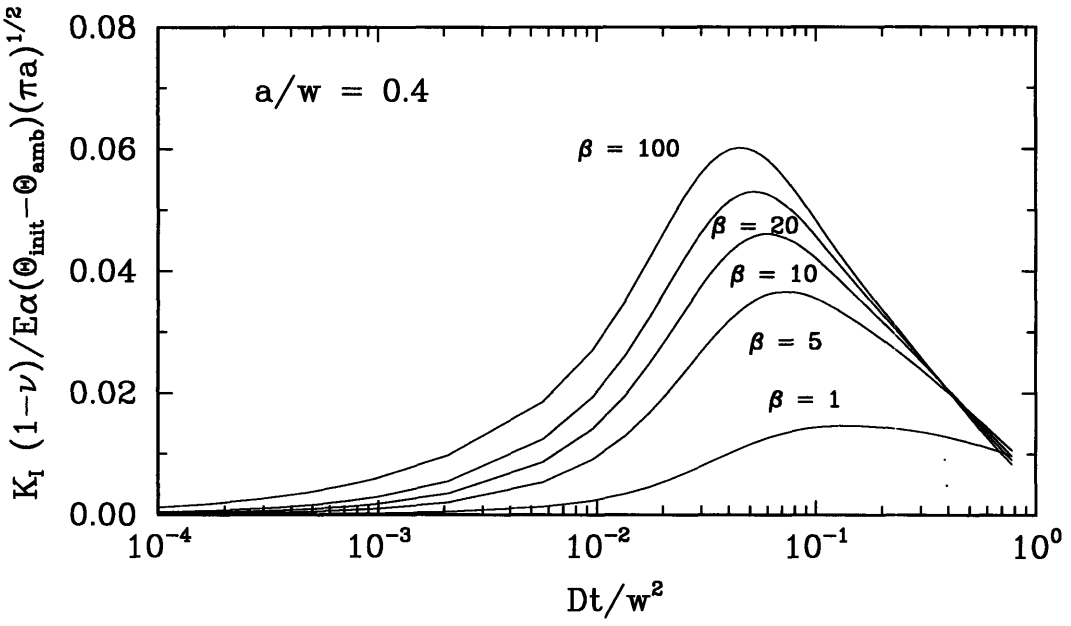


Figure 4.9: Stress intensity factors K_I^* for $a/w = 0.4$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

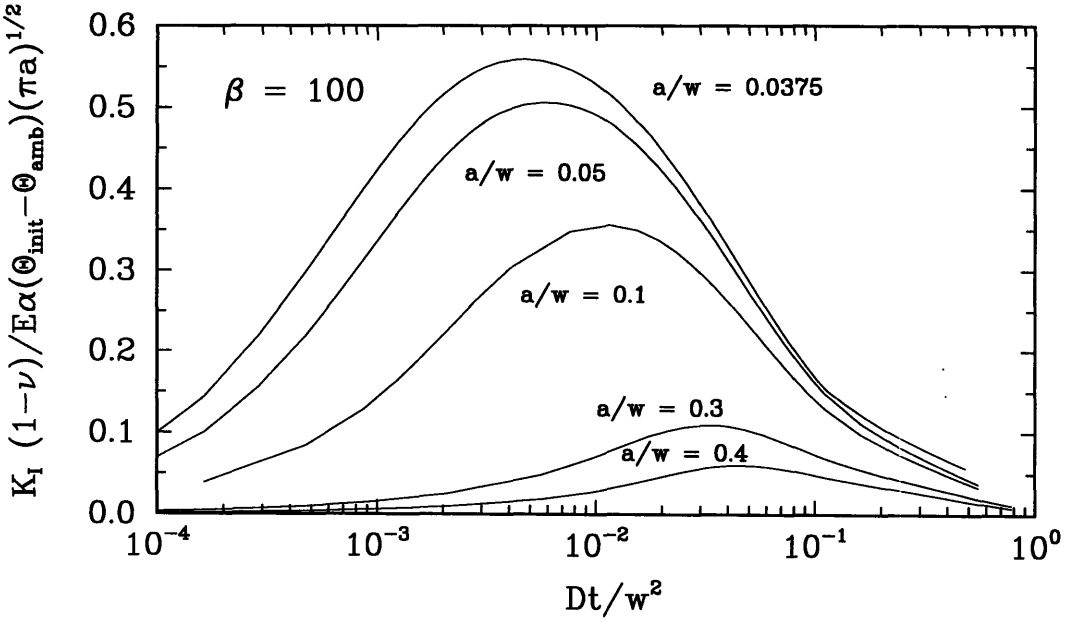


Figure 4.10: Stress intensity factors K_I^* as a function of nondimensional time Dt/w^2 for various crack lengths ($\beta = 100$).

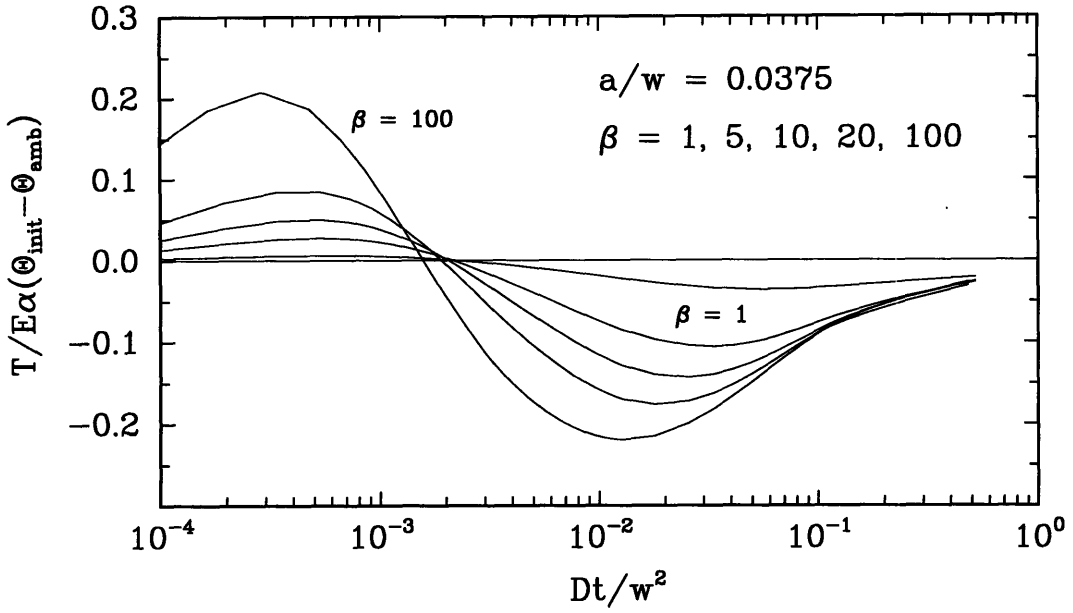


Figure 4.11: Non-dimensionalized T-stress values for $a/w = 0.0375$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

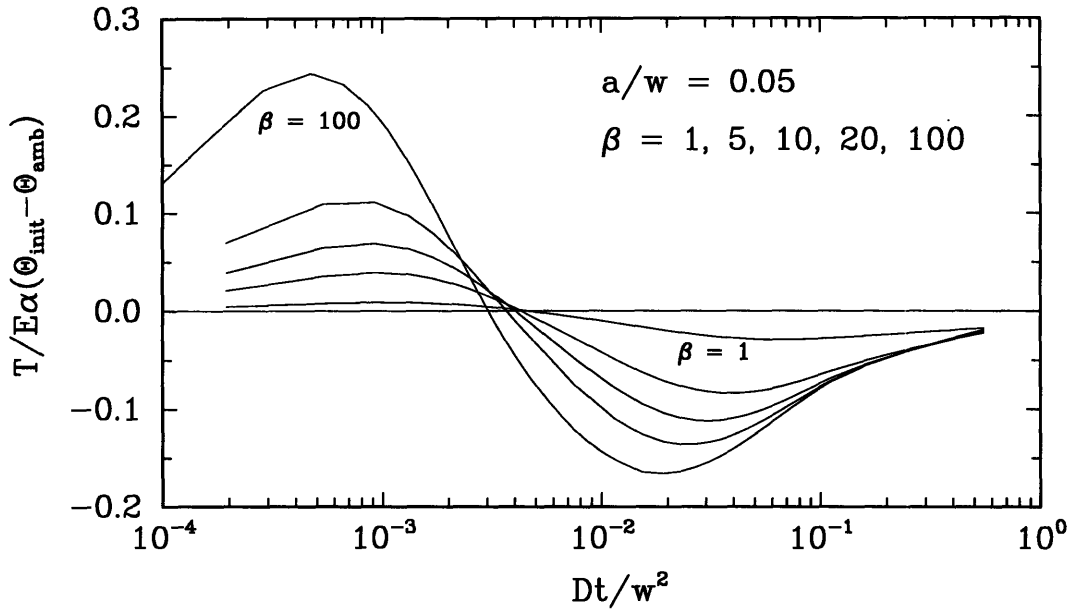


Figure 4.12: Non-dimensionalized T-stress values for $a/w = 0.05$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

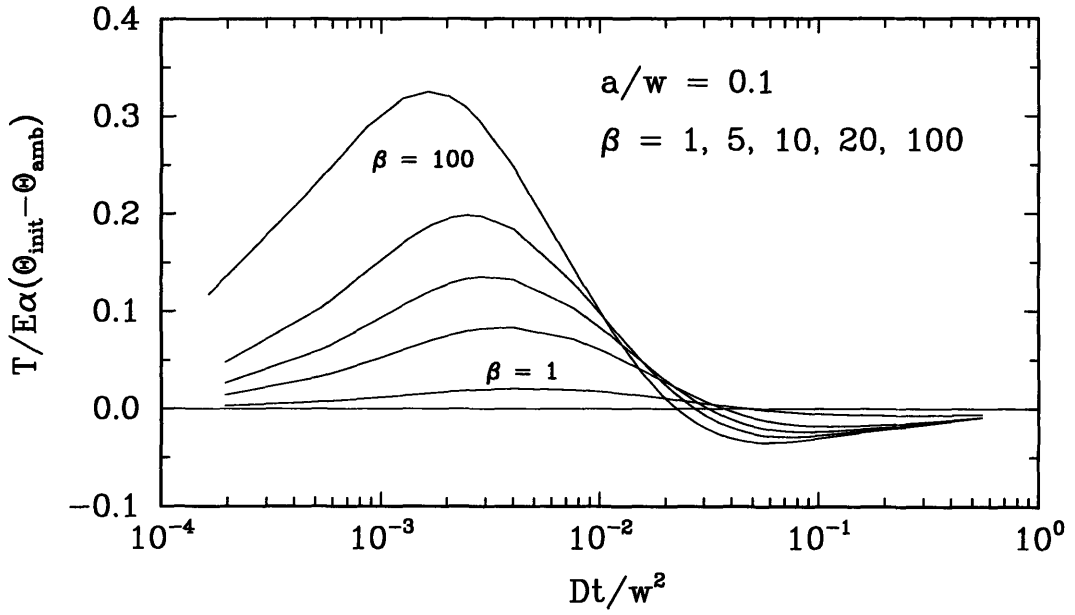


Figure 4.13: Non-dimensionalized T-stress values for $a/w = 0.1$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

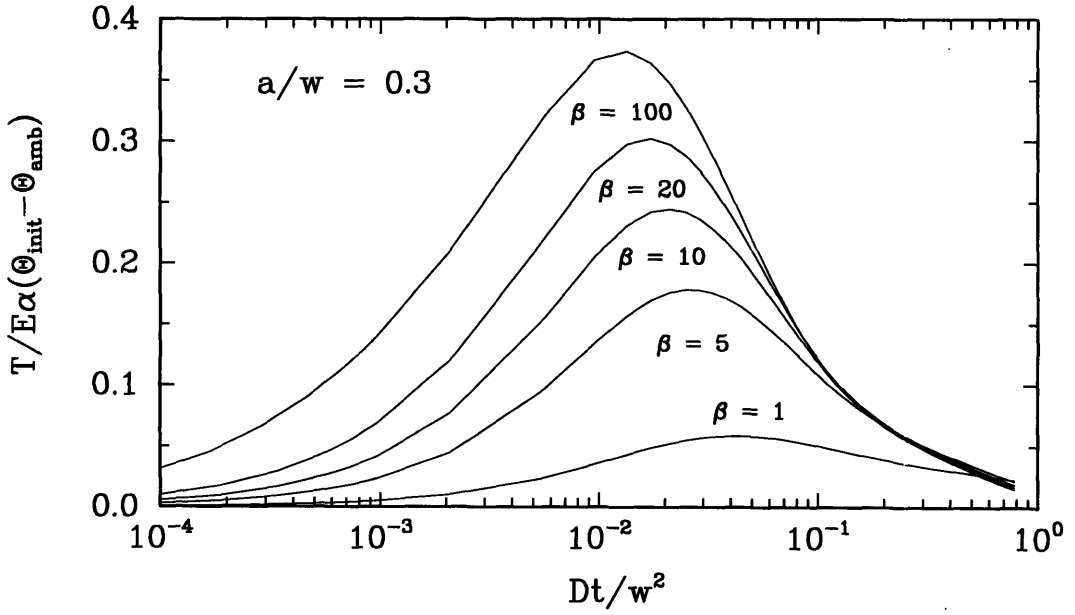


Figure 4.14: Non-dimensionalized T-stress values for $a/w = 0.3$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

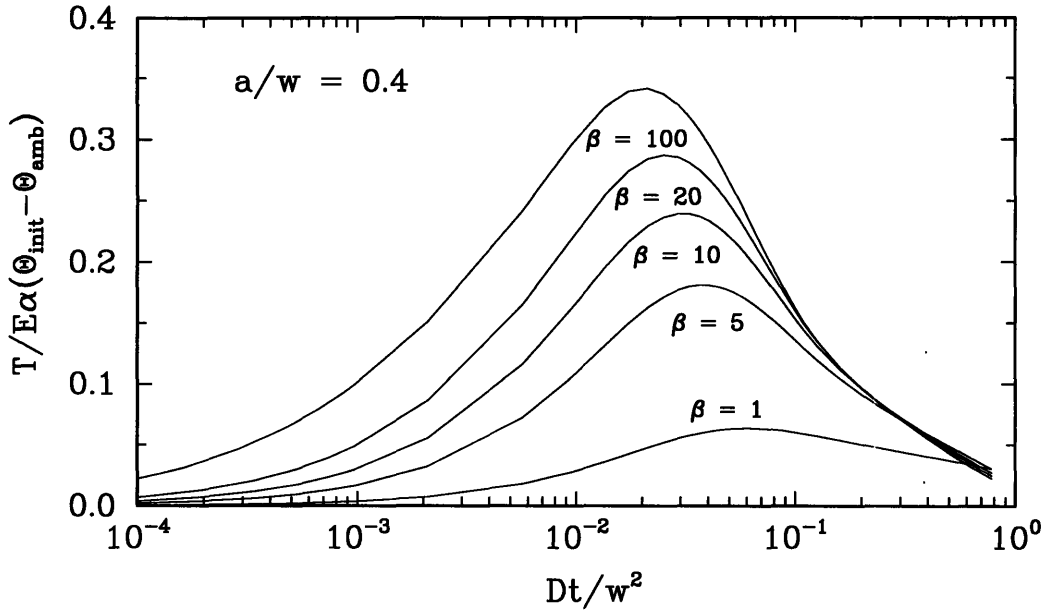


Figure 4.15: Non-dimensionalized T-stress values for $a/w = 0.4$ as a function of nondimensional time Dt/w^2 for various values of the Biot number.

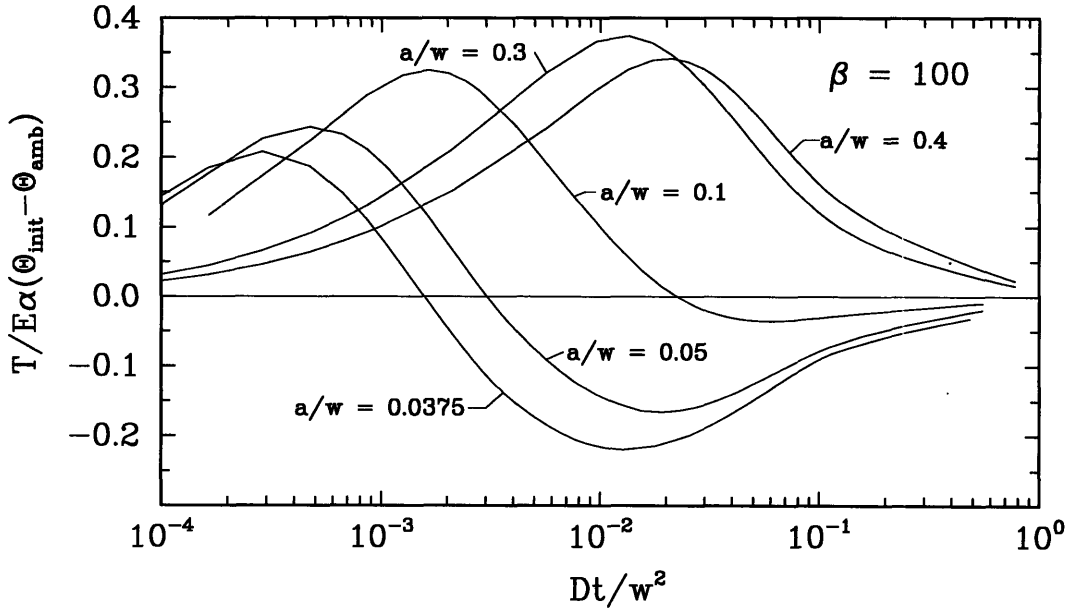
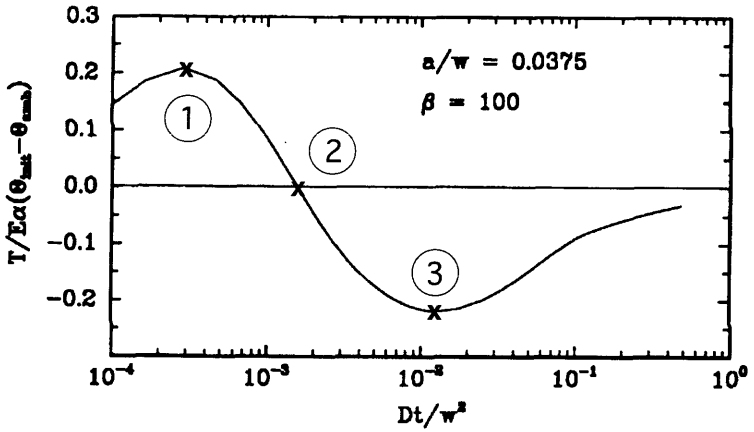
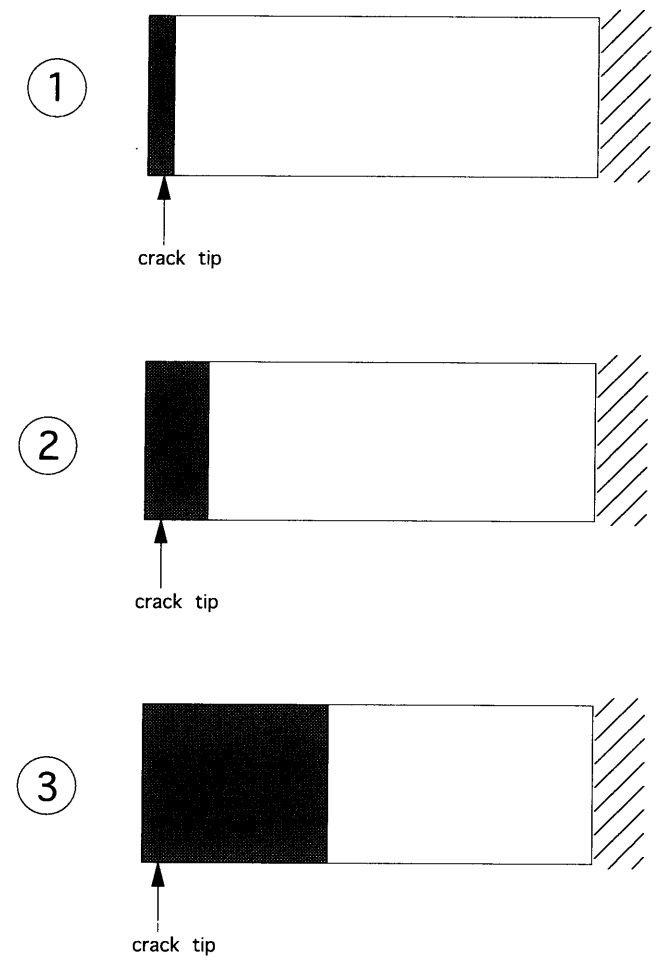


Figure 4.16: Non-dimensionalized T-stress values as a function of nondimensional time Dt/w^2 for various crack lengths ($\beta = 100$).



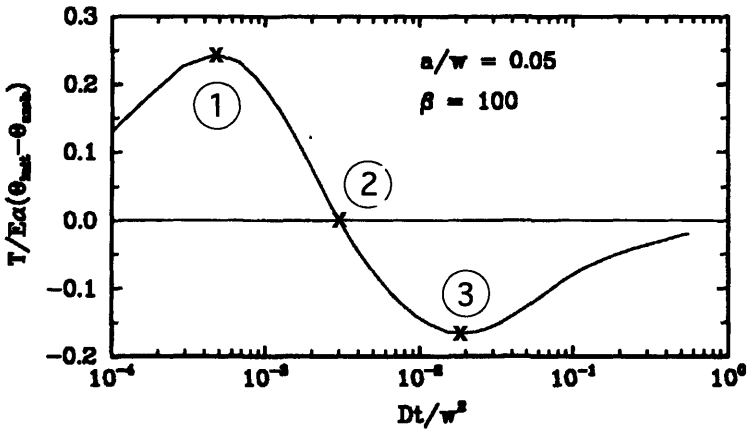


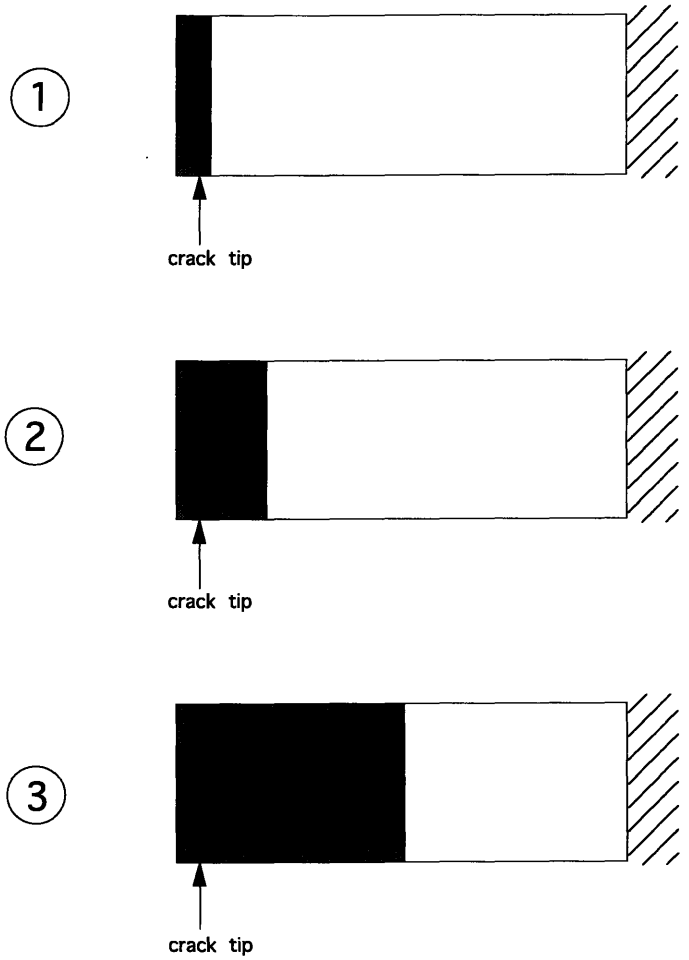
1 $T^* = 0.208$ $\delta^* = 1.537$ $Fo = 2.868 \times 10^{-4}$

2 $T^* = -0.00845$ $\delta^* = 3.733$ $Fo = 1.638 \times 10^{-3}$

3 $T^* = -0.2196$ $\delta^* = 10.918$ $Fo = 1.279 \times 10^{-2}$

Figure 4.17: Position of temperature front relative to crack-tip for $a/w = 0.0375$ at three values of normalized T ($\beta = 100$). Temperature front locations correspond to $(\Theta_{init} - \Theta)/(\Theta_{init} - \Theta_{amb}) = 0.01$.



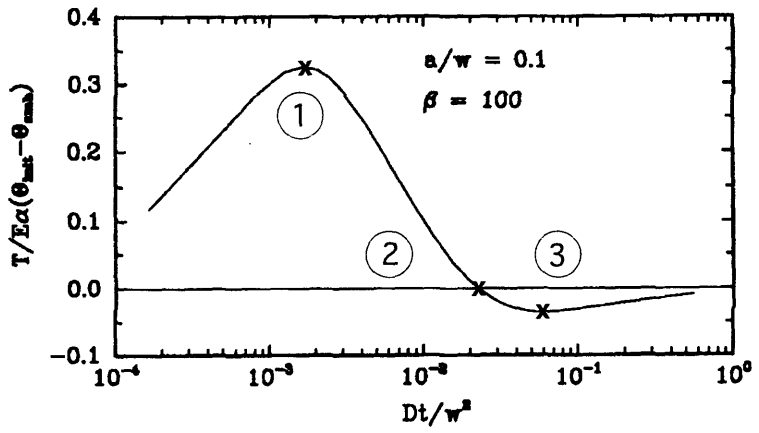


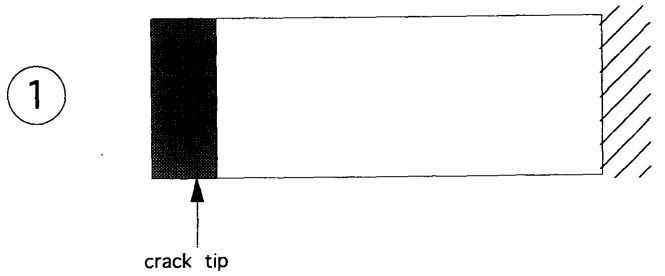
1 $T^* = 0.244$ $\delta^* = 1.482$ $Fo = 4.711 \times 10^{-4}$

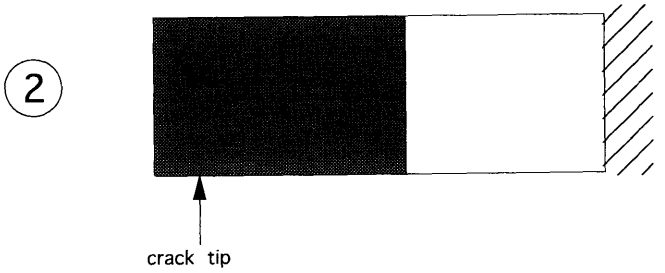
2 $T^* = -0.00458$ $\delta^* = 3.929$ $Fo = 3.111 \times 10^{-3}$

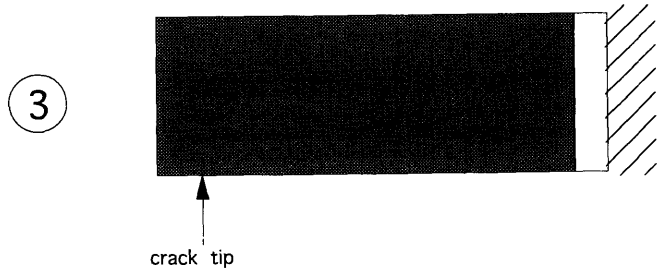
3 $T^* = -0.1659$ $\delta^* = 10.0$ $Fo = 1.928 \times 10^{-2}$

Figure 4.18: Position of temperature front relative to crack-tip for $a/w = 0.05$ at three values of normalized T ($\beta = 100$). Temperature front locations correspond to $(\Theta_{init} - \Theta)/(\Theta_{init} - \Theta_{amb}) = 0.01$.







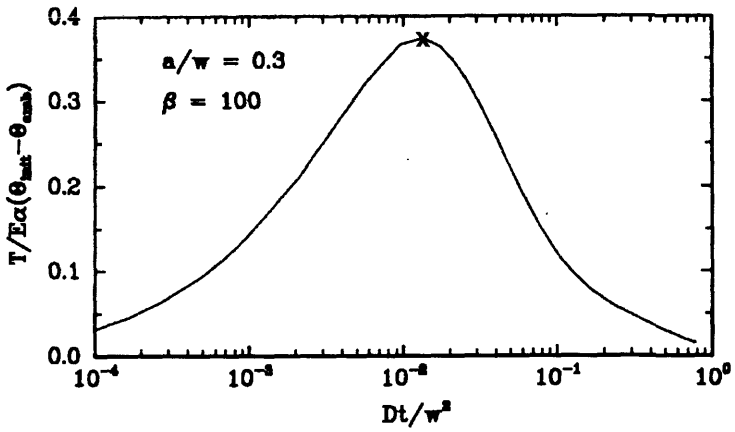


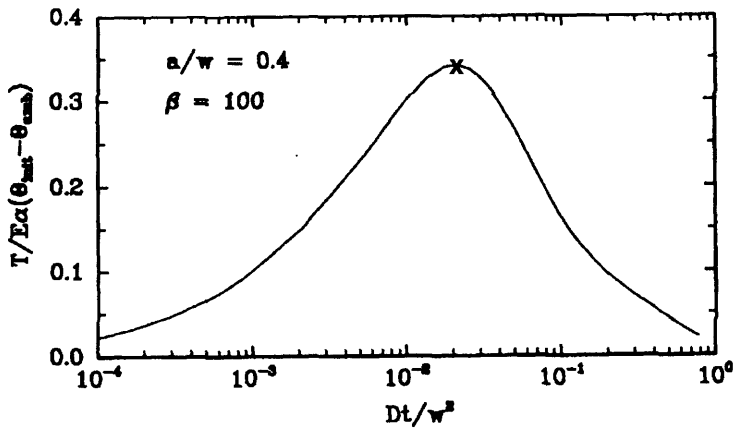
(1) $T^* = 0.325$ $\quad \delta^* = 1.424$
$Fo = 1.634 \times 10^{-3}$

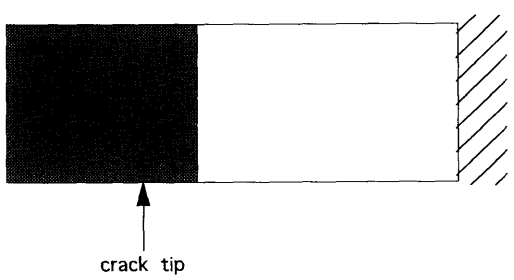
(2) $T^* = -0.00265$ $\quad \delta^* = 5.553$
$Fo = 2.316 \times 10^{-2}$

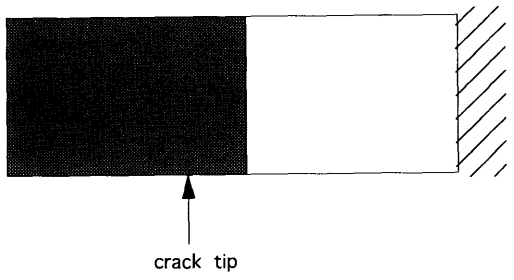
(3) $T^* = -0.0352$ $\quad \delta^* = 9.224$
$Fo = 5.817 \times 10^{-2}$

Figure 4.19: Position of temperature front relative to crack-tip for $a/w = 0.1$ at three values of normalized T ($\beta = 100$). Temperature front locations correspond to $(\Theta_{init} - \Theta)/(\Theta_{init} - \Theta_{amb}) = 0.01$.









$T^* = 0.374$ $\delta^* = 1.396$

$Fo = 1.344 \times 10^{-2}$

$T^* = 0.341$ $\delta^* = 1.318$

$Fo = 2.122 \times 10^{-2}$

Figure 4.20: Position of temperature front relative to crack-tip for $a/w = 0.3$ and $a/w = 0.4$ at maximum value of normalized T ($\beta = 100$). Temperature front locations correspond to $(\Theta_{init} - \Theta)/(\Theta_{init} - \Theta_{amb}) = 0.01$.

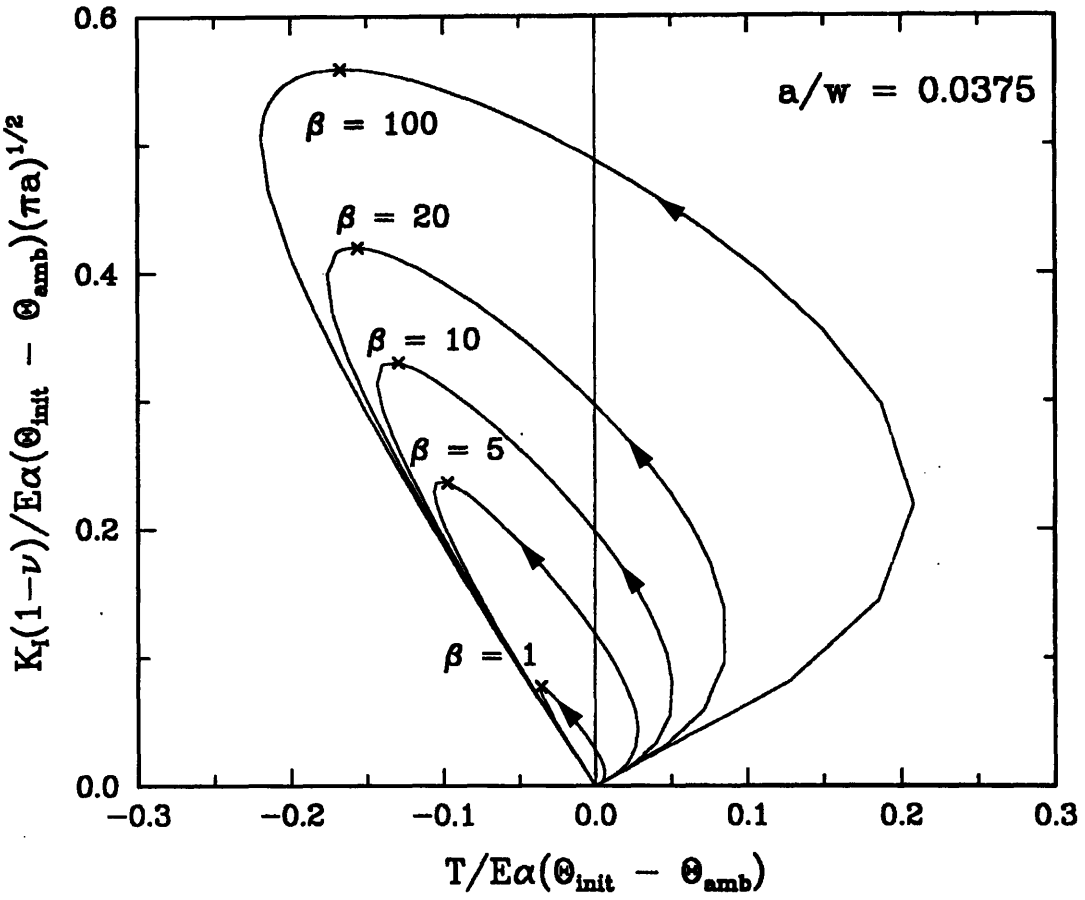


Figure 4.21: Non-dimensionalized K_I *vs.* T for $a/w = 0.0375$, for various values of the Biot number. Arrows indicate the direction of traverse.

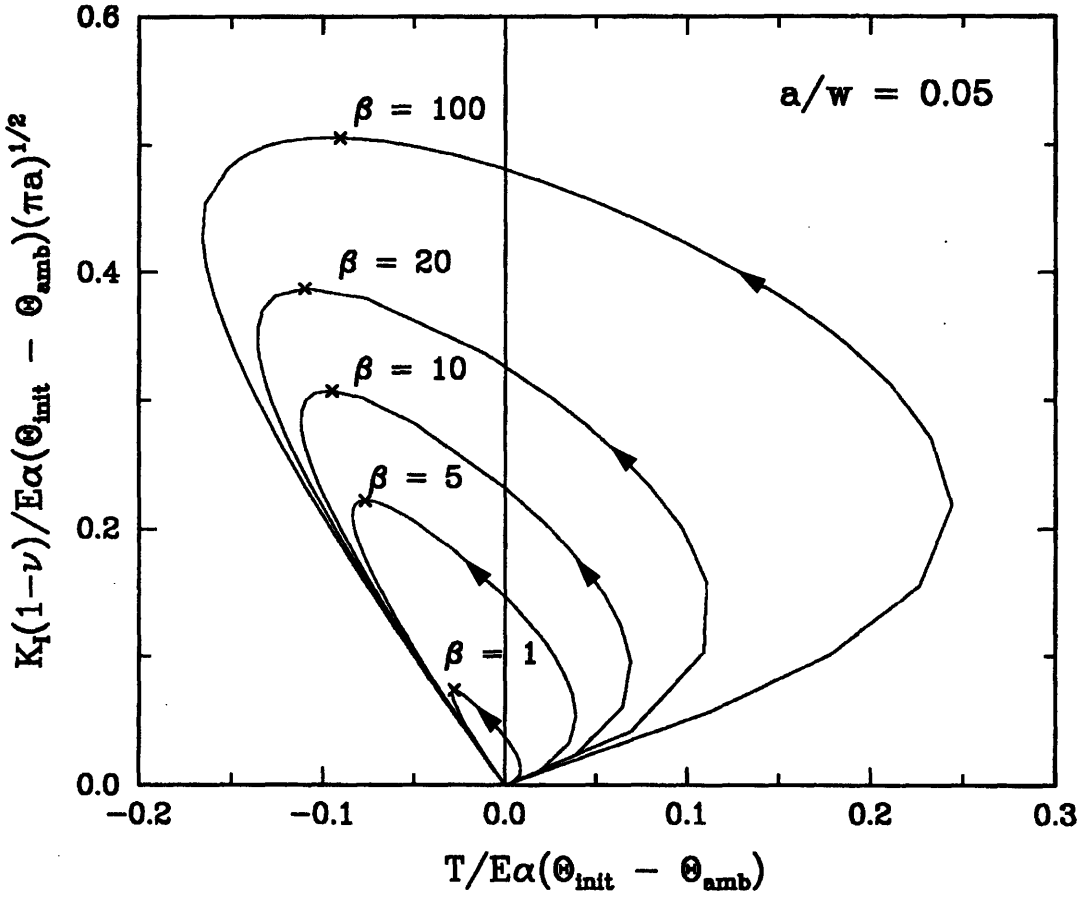


Figure 4.22: Non-dimensionalized K_I *vs.* T for $a/w = 0.05$, for various values of the Biot number. Arrows indicate the direction of traverse.

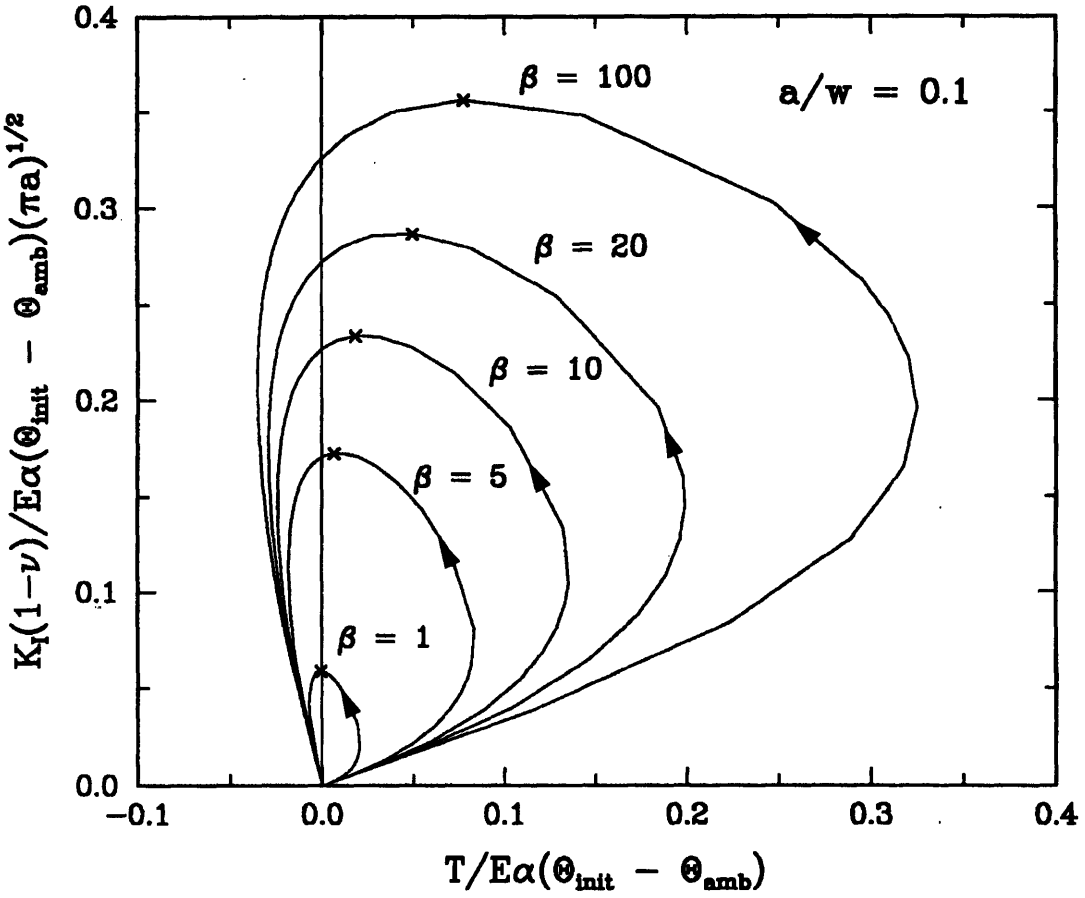


Figure 4.23: Non-dimensionalized K_I *vs.* T for $a/w = 0.1$, for various values of the Biot number. Arrows indicate the direction of traverse.

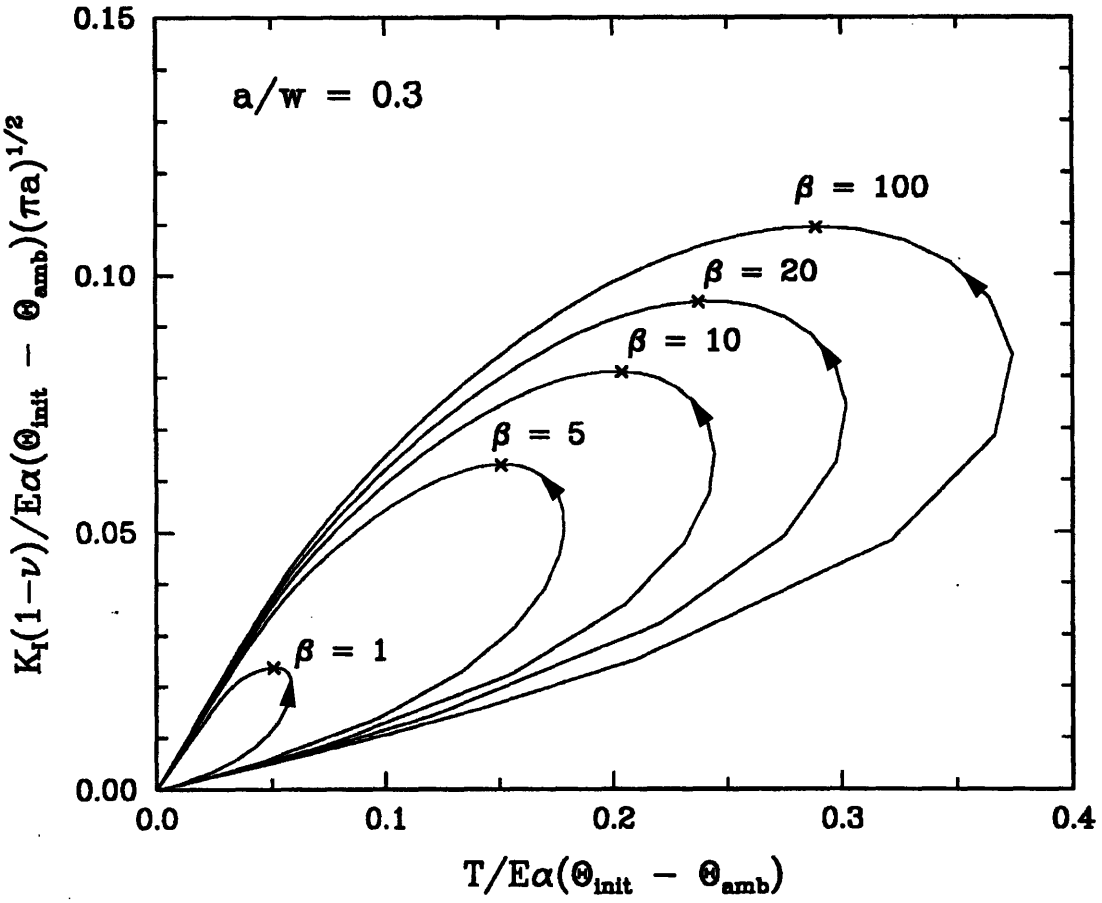


Figure 4.24: Non-dimensionalized K_I *vs.* T for $a/w = 0.3$, for various values of the Biot number. Arrows indicate the direction of traverse.

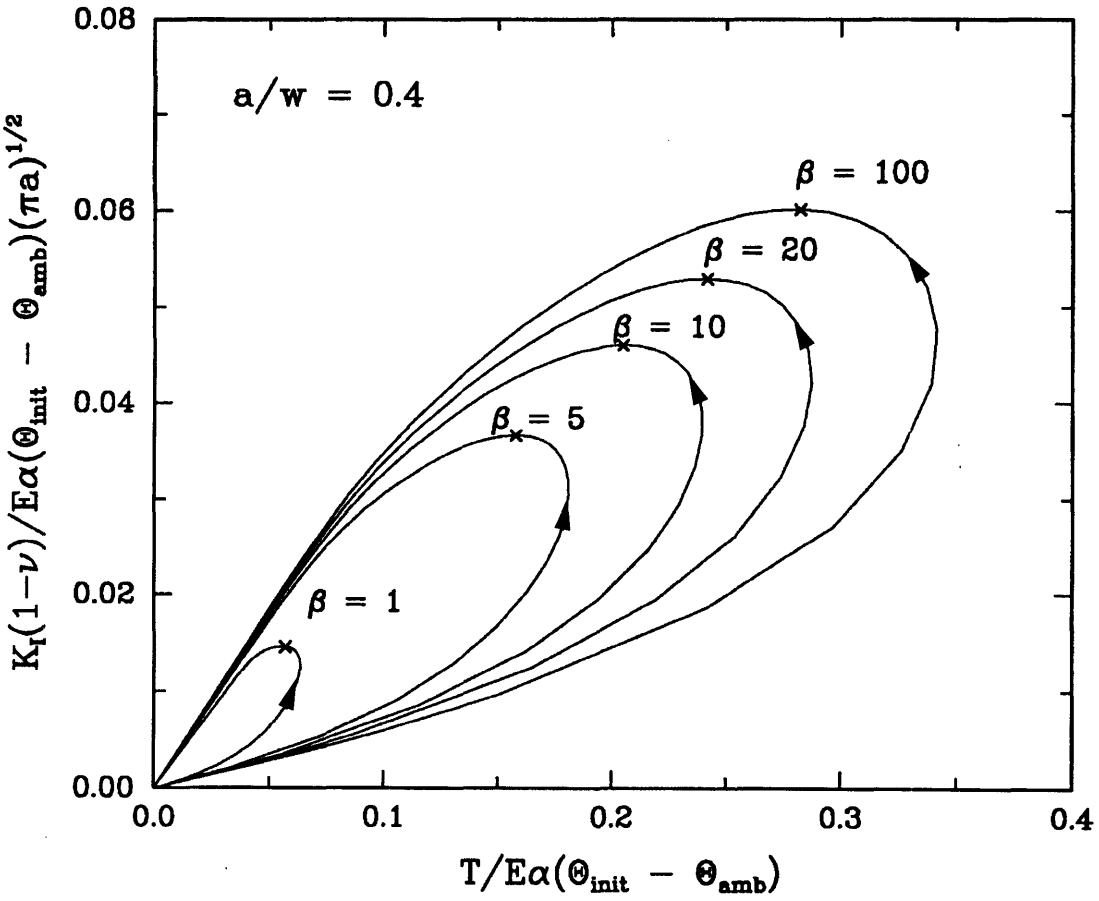


Figure 4.25: Non-dimensionalized K_I *vs.* T for $a/w = 0.4$, for various values of the Biot number. Arrows indicate the direction of traverse.

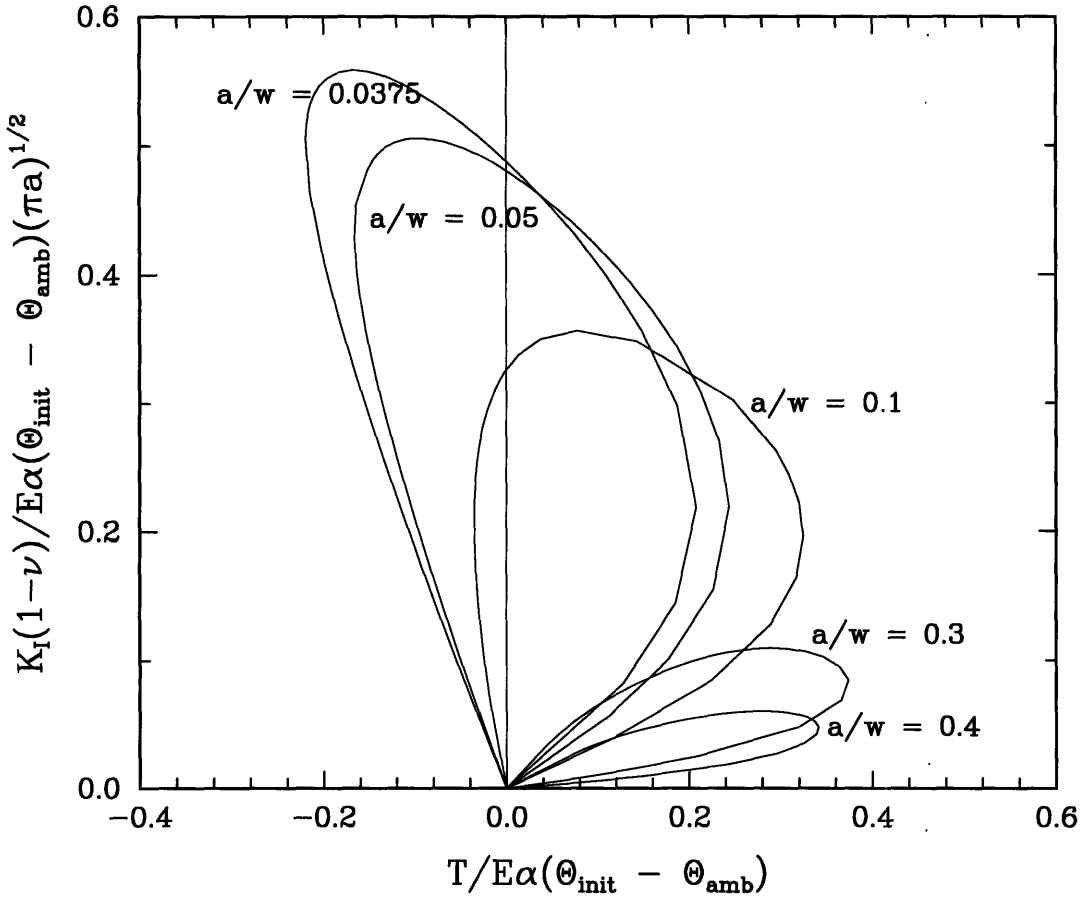


Figure 4.26: Non-dimensionalized K_I *vs.* T for various relative crack depths (β = 100).

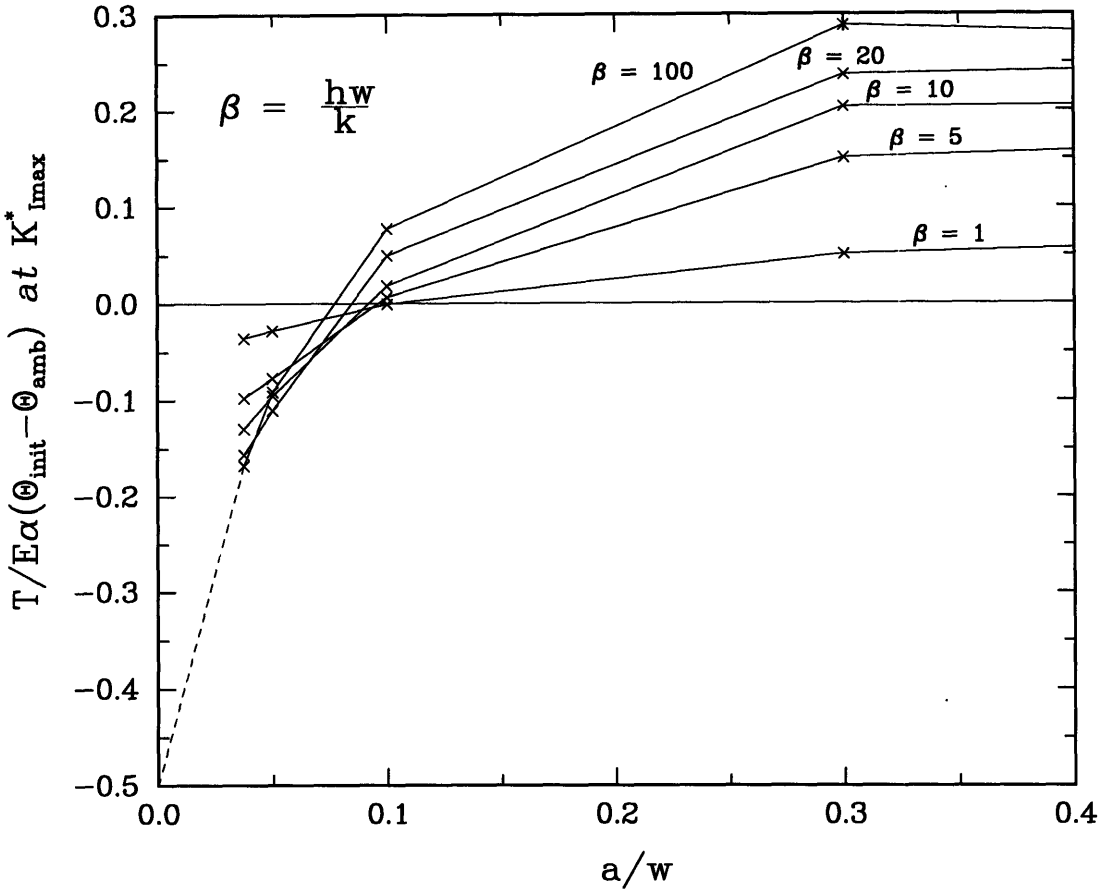


Figure 4.27: Non-dimensionalized T-stress values at maximum K_I^*, for various Biot numbers and relative crack depths. The dashed line indicates the expected asymptotic T^*-values ($a/w \rightarrow 0$).

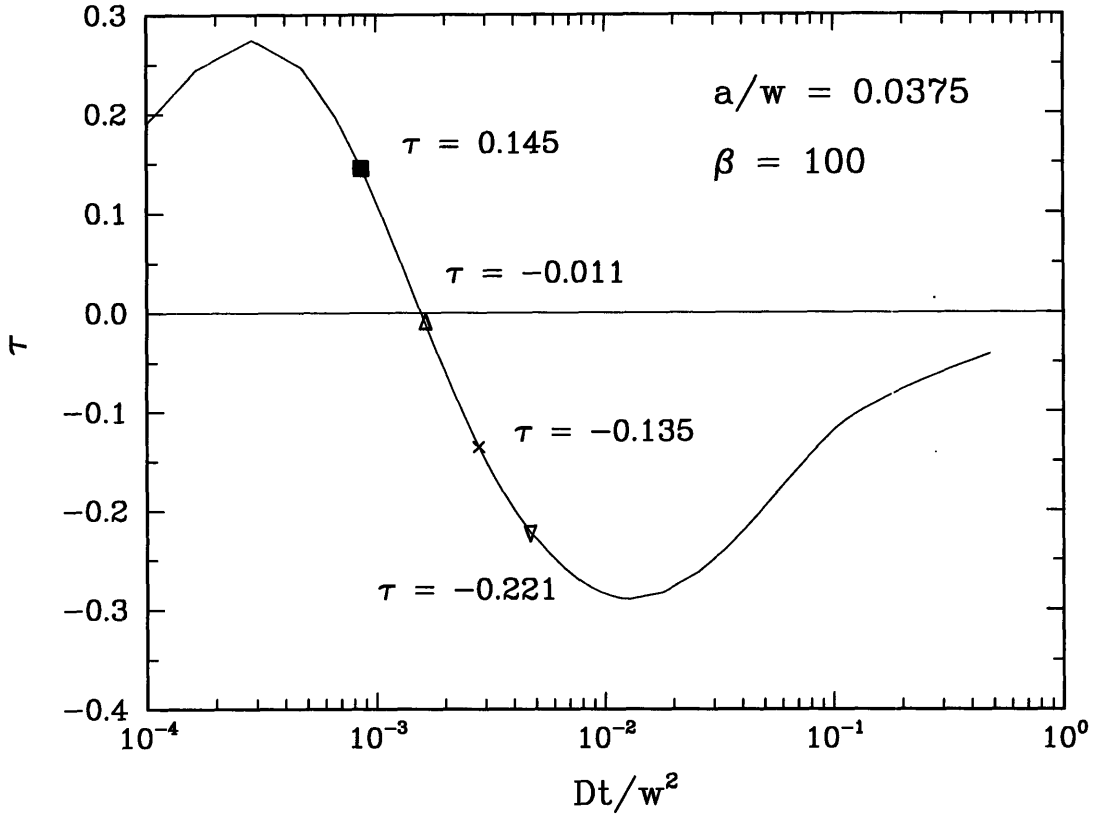


Figure 4.28: Non-dimensionalized T-stress values as a function of nondimensional time Dt/w^2 for $a/w = 0.0375$ ($\beta = 100$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). Symbols indicate instances in time for which crack-opening stress profiles are obtained.

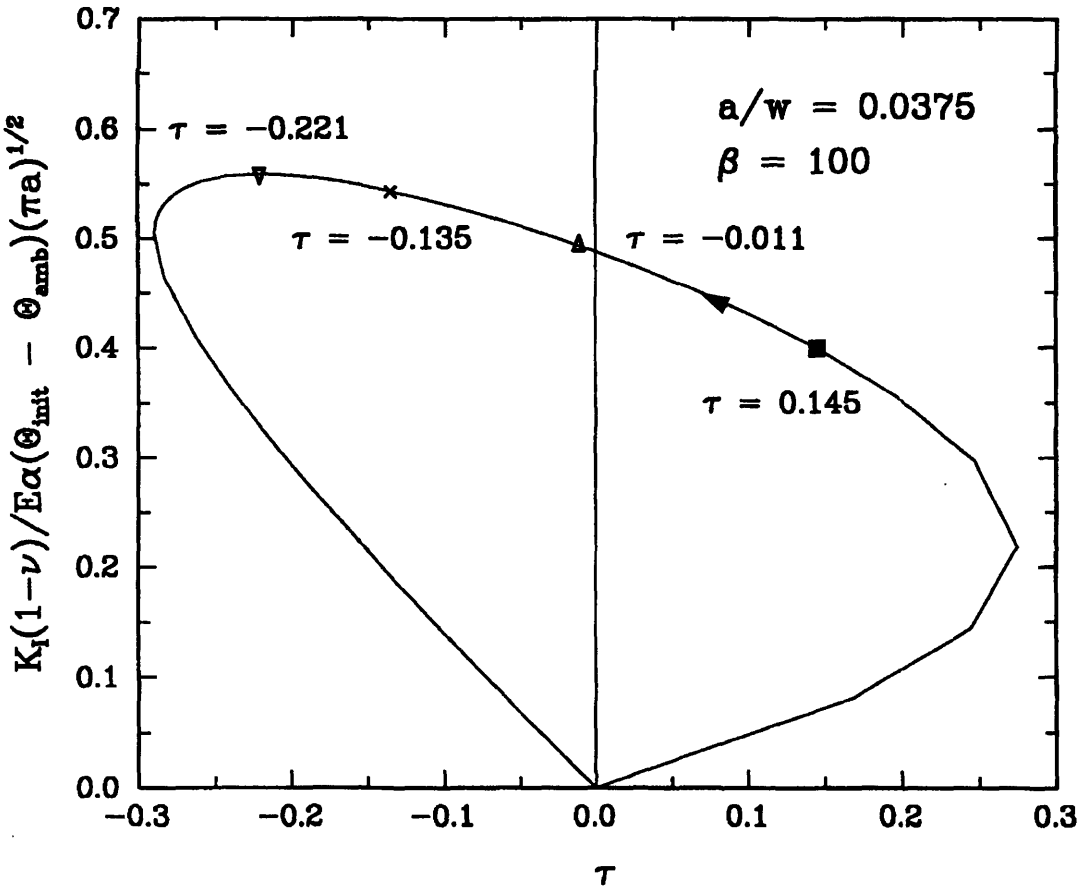


Figure 4.29: Non-dimensionalized K_I *vs.* T for $a/w = 0.0375$ ($\beta = 100$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). Symbols indicate instances in time for which crack-opening stress profiles are obtained.

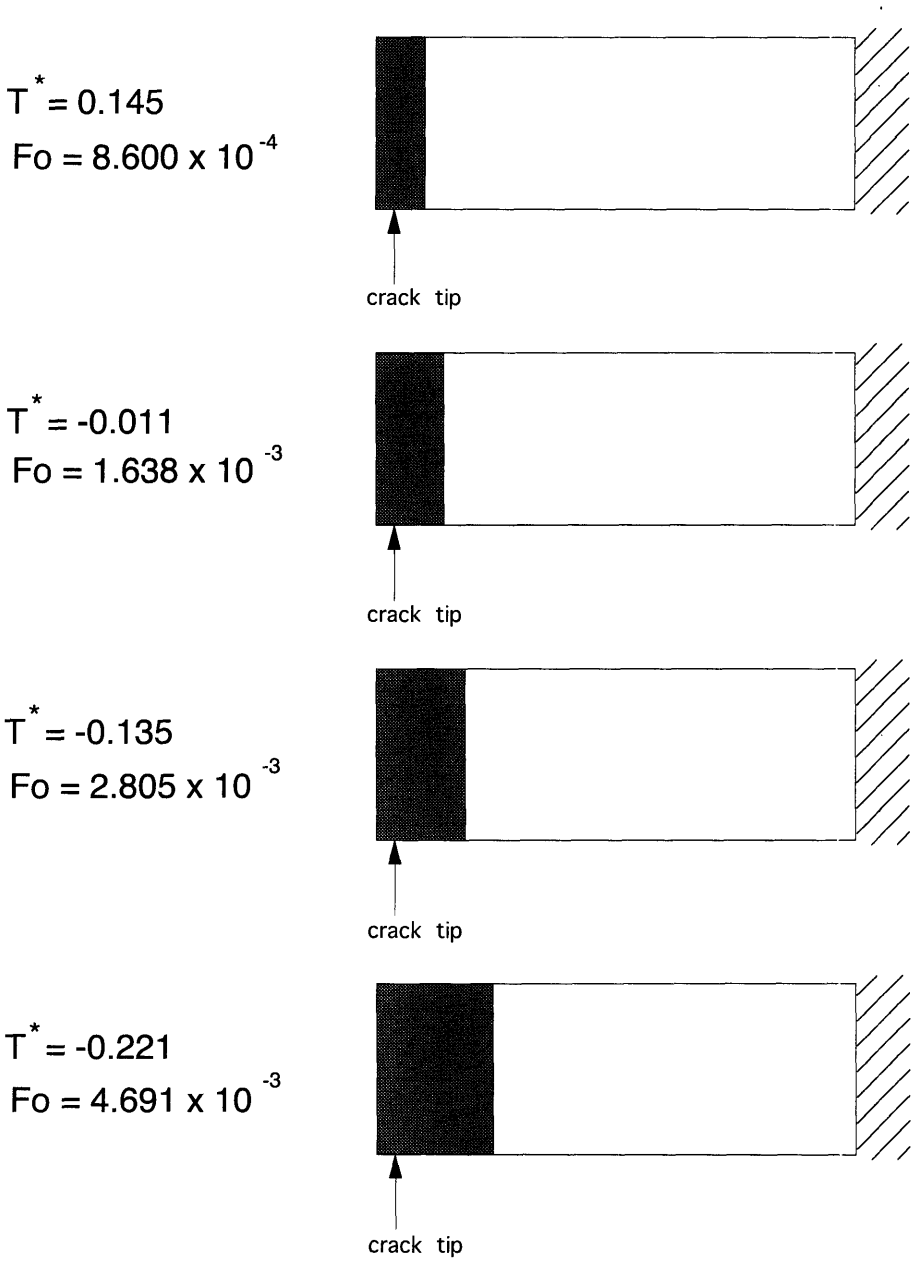


Figure 4.30: Position of temperature front relative to crack-tip for $a/w = 0.0375$ at four values of normalized T ($\beta = 100$). Temperature front locations correspond to $(\Theta_{init} - \Theta)/(\Theta_{init} - \Theta_{amb}) = 0.01$.

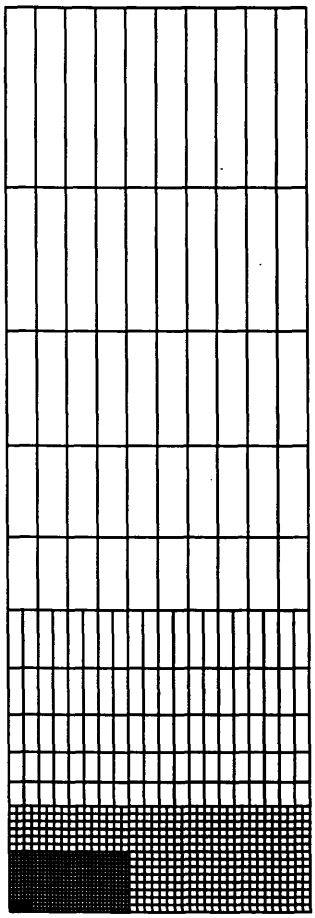


Figure 4.31: Mesh used in elastic-plastic analysis of the problem.

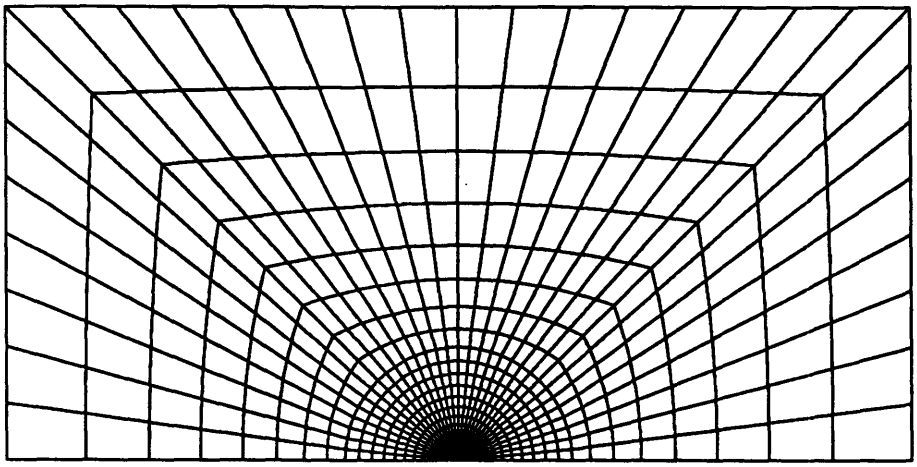
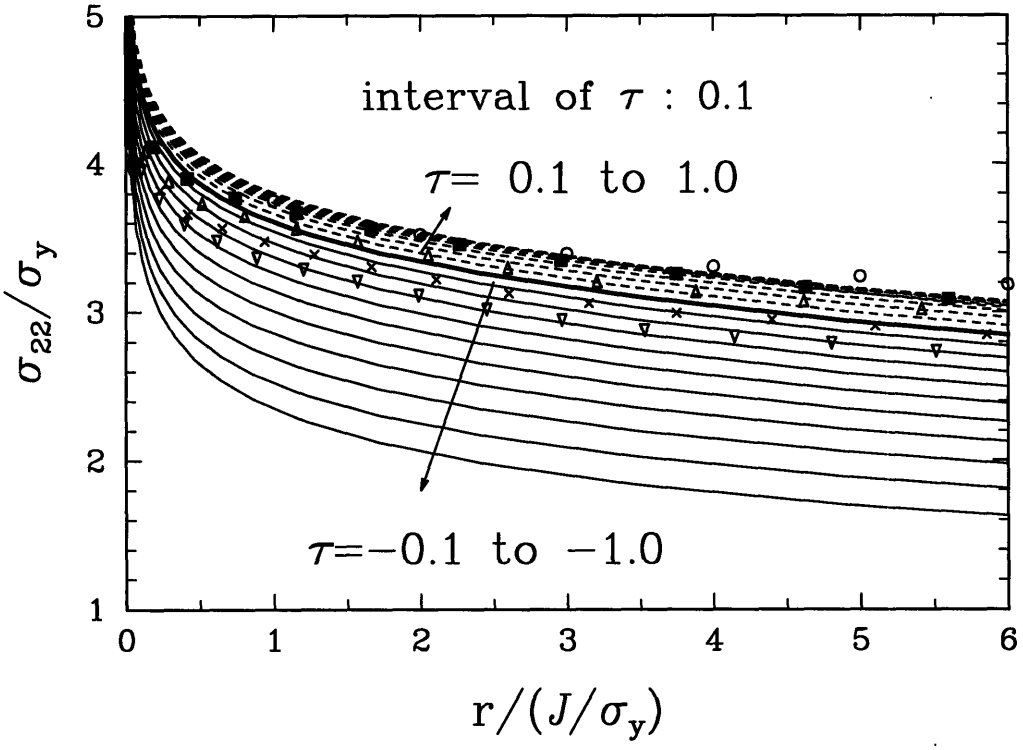


Figure 4.32: Near-tip region of mesh shown in Fig. 4.31 .



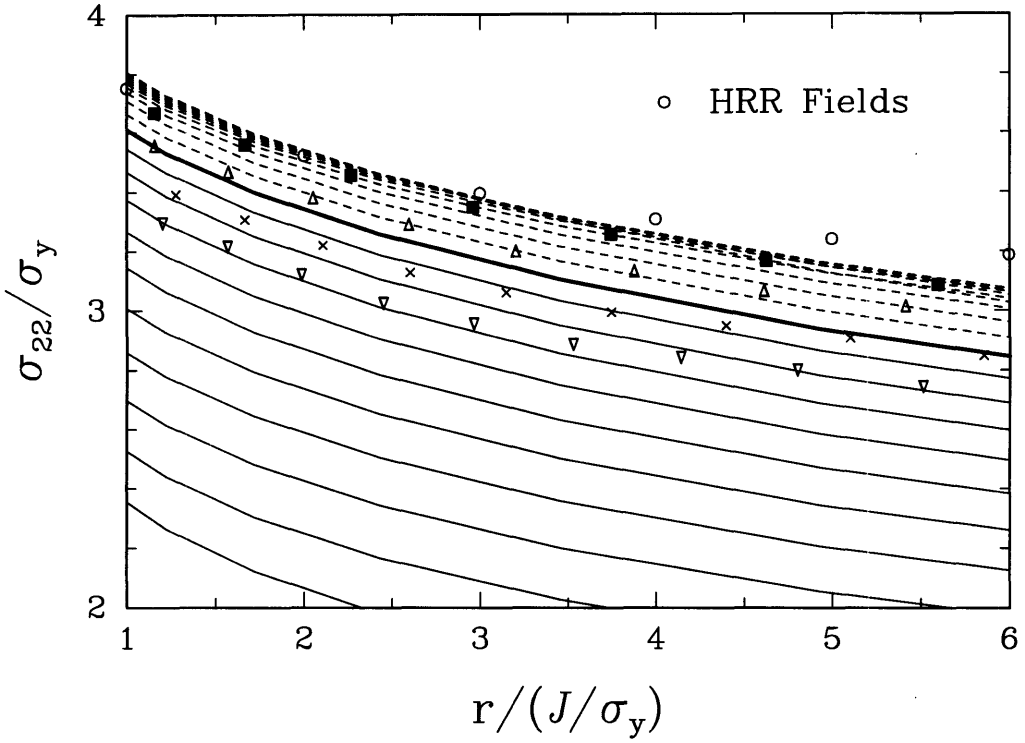
■ ■ ■ τ = 0.145 Fo = 8.600 x 10^{-4}
K_I^* = 0.399

▲ ▲ ▲ τ = −0.011 Fo = 1.638 x 10^{-3}
K_I^* = 0.493

× × × τ = −0.135 Fo = 2.805 x 10^{-3}
K_I^* = 0.543

▼ ▼ ▼ τ = −0.221 Fo = 4.691 x 10^{-3}
K_I^* = 0.559

Figure 4.33: Normalized crack-opening stress profiles at four instances of time and corresponding MBL solutions ($a/w = 0.0375$, $\beta = 100$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). K_I^* are stress intensity factors normalized according to Eq. (4.2).



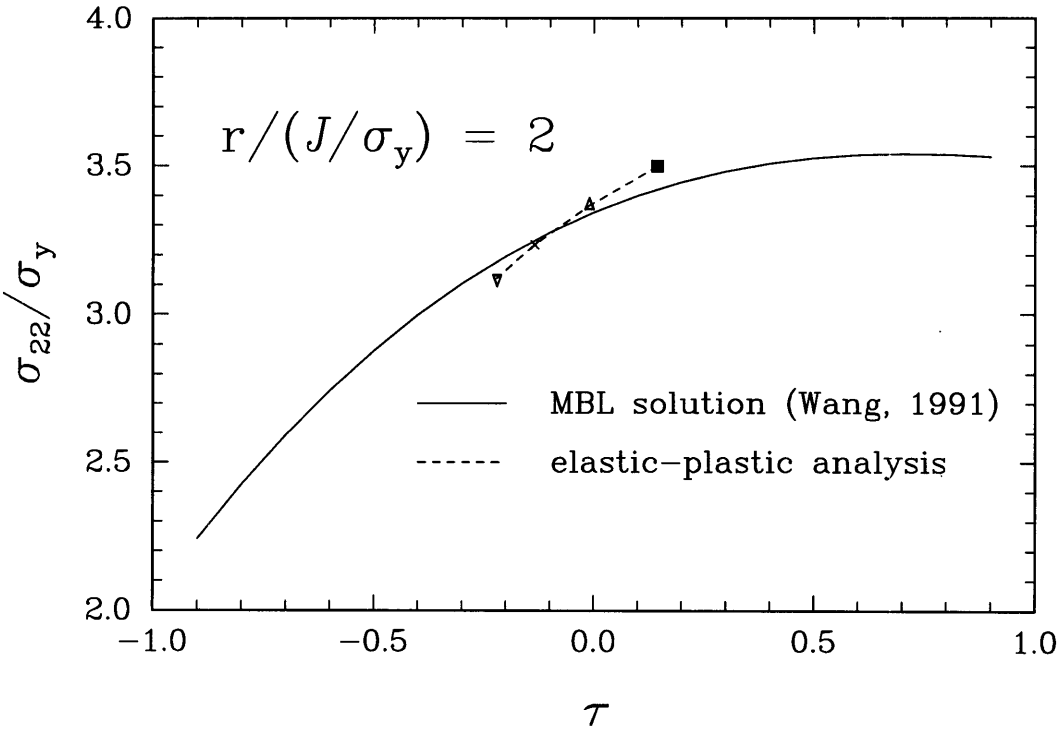
■ ■ ■ τ = 0.145 Fo = 8.600 x 10^{-4}
K_I^* = 0.399

▲ ▲ ▲ τ = −0.011 Fo = 1.638 x 10^{-3}
K_I^* = 0.493

× × × τ = −0.135 Fo = 2.805 x 10^{-3}
K_I^* = 0.543

▽ ▽ ▽ τ = −0.221 Fo = 4.691 x 10^{-3}
K_I^* = 0.559

Figure 4.34: Enlarged view of Fig. 4.33. K_I^* are stress intensity factors normalized according to Eq. (4.2).



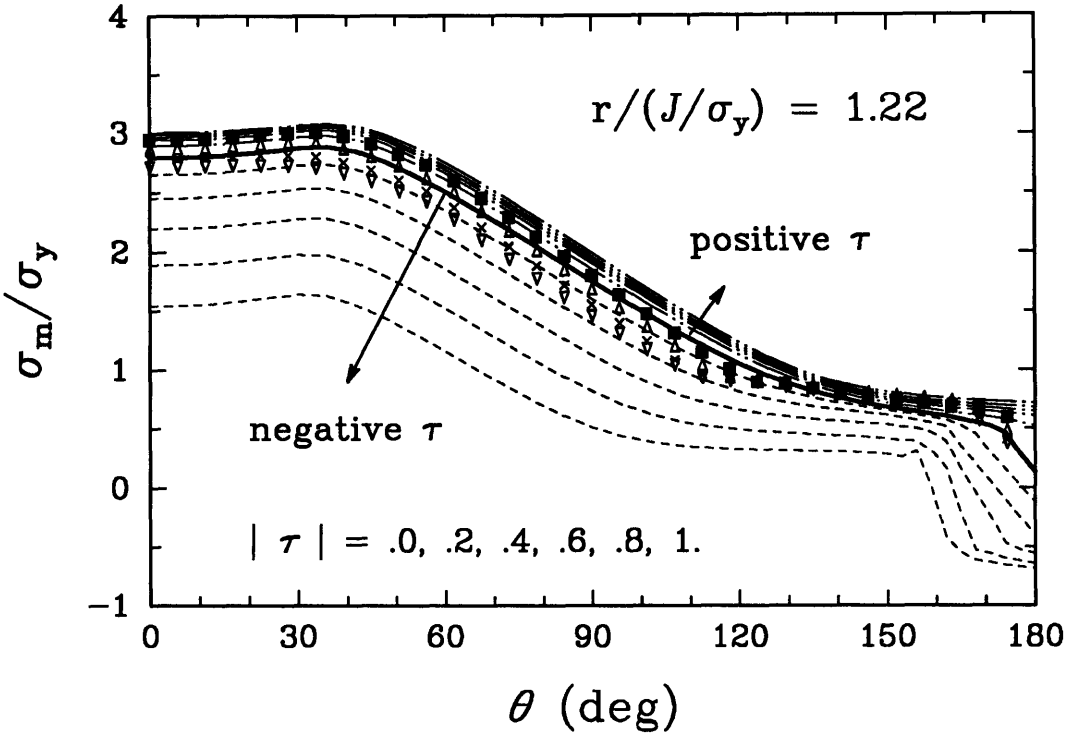
■ ■ ■ τ = 0.145 Fo = 8.600 x 10^{-4} K_I^* = 0.399

▵ ▵ ▵ τ = −0.011 Fo = 1.638 x 10^{-3} K_I^* = 0.493

× × × τ = −0.135 Fo = 2.805 x 10^{-3} K_I^* = 0.543

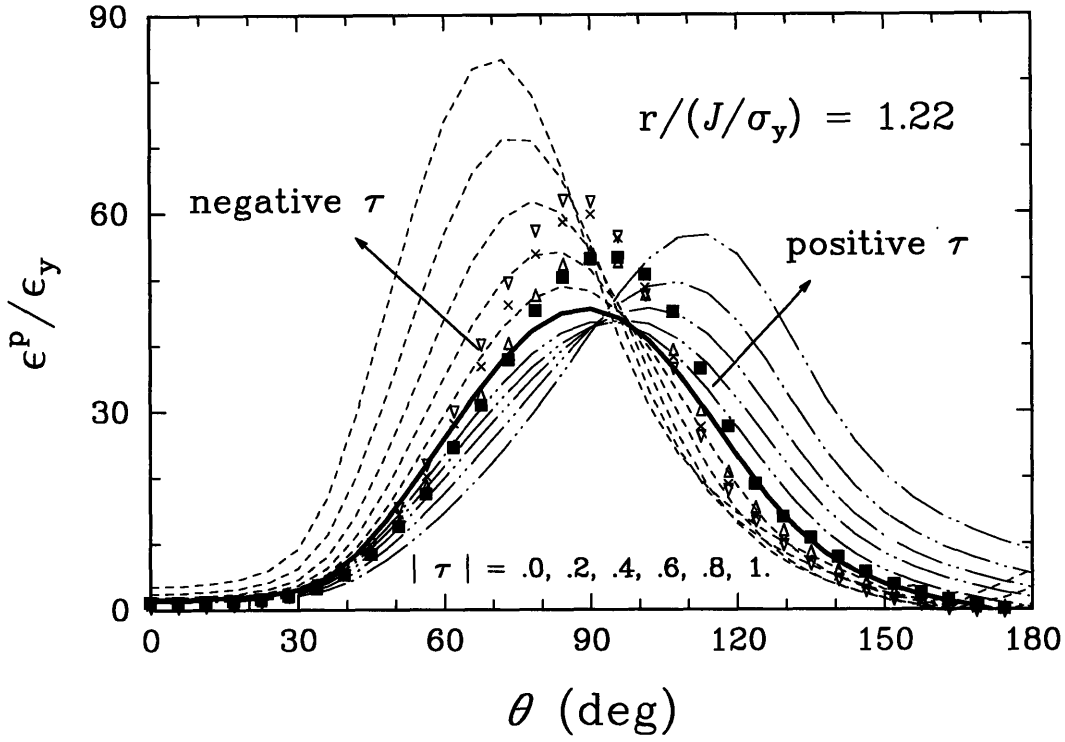
▿ ▿ ▿ τ = −0.221 Fo = 4.691 x 10^{-3} K_I^* = 0.559

Figure 4.35: Normalized crack-opening stresses at $r/(J/\sigma_y) = 2$ *vs.* τ at four instances of time and corresponding MBL results ($a/w = 0.0375$, $\beta = 100$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$).



■ ■ ■	τ = 0.145	Fo = 8.600 x 10^{-4} K_I^* = 0.399
▲ ▲ ▲	τ = −0.011	Fo = 1.638 x 10^{-3} K_I^* = 0.493
× × ×	τ = −0.135	Fo = 2.805 x 10^{-3} K_I^* = 0.543
▼ ▼ ▼	τ = −0.221	Fo = 4.691 x 10^{-3} K_I^* = 0.559

Figure 4.36: Variation of hydrostatic stress at four instances of time and corresponding MBL solutions ($a/w = 0.0375$, $\beta = 100$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). K_I^* are stress intensity factors normalized according to Eq. (4.2).



■ ■ ■	τ = 0.145	Fo = 8.600 x 10^{-4} K_I^* = 0.399
▲ ▲ ▲	τ = −0.011	Fo = 1.638 x 10^{-3} K_I^* = 0.493
× × ×	τ = −0.135	Fo = 2.805 x 10^{-3} K_I^* = 0.543
▼ ▼ ▼	τ = −0.221	Fo = 4.691 x 10^{-3} K_I^* = 0.559

Figure 4.37: Variation of equivalent plastic strain at four instances of time and corresponding MBL solutions ($a/w = 0.0375$, $\beta = 100$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). K_I^* are stress intensity factors normalized according to Eq. (4.2).

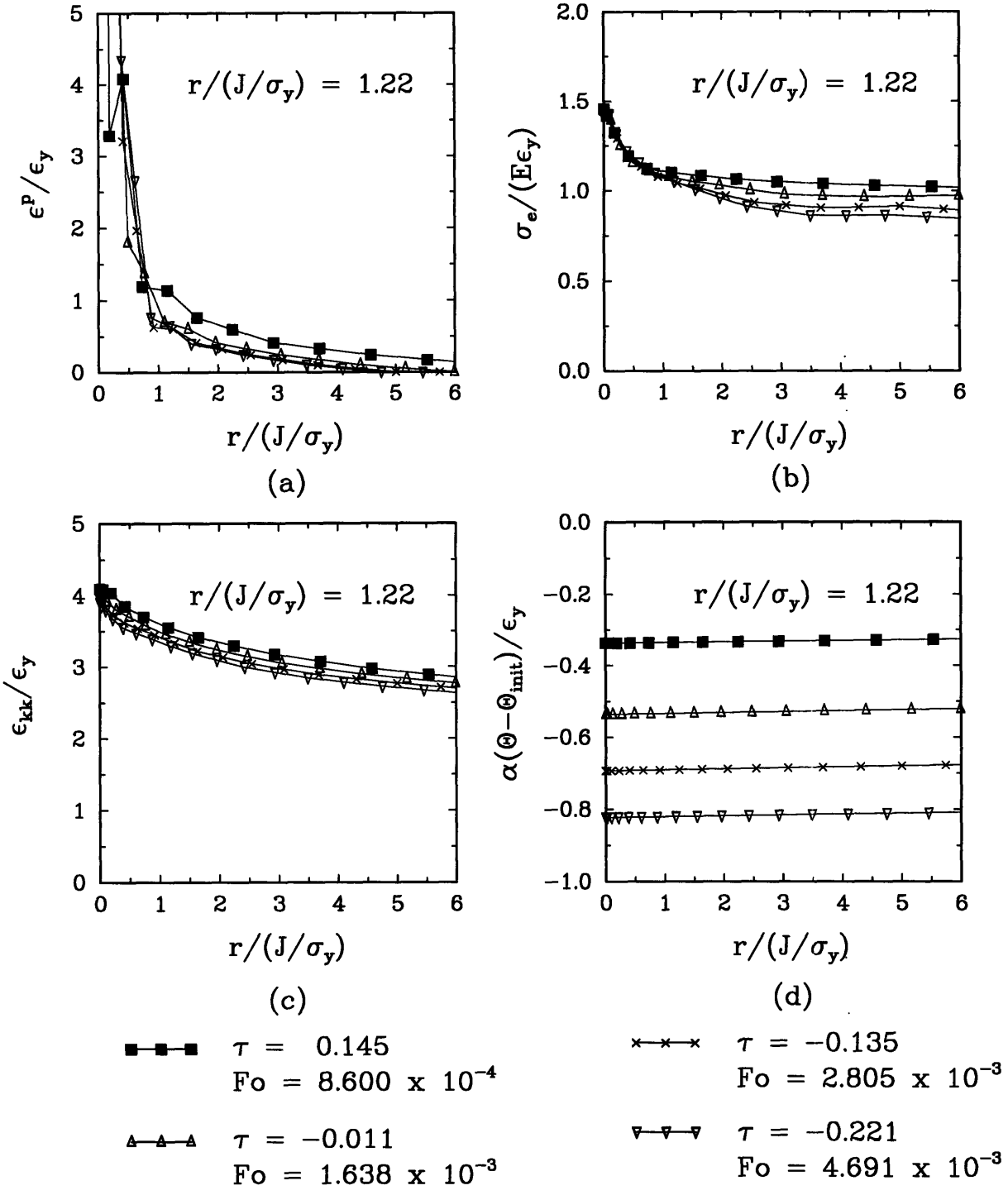


Figure 4.38: Strain components ahead of the crack at four instances of time during the thermal transient ($a/w = 0.0375$, $\beta = 100$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$).

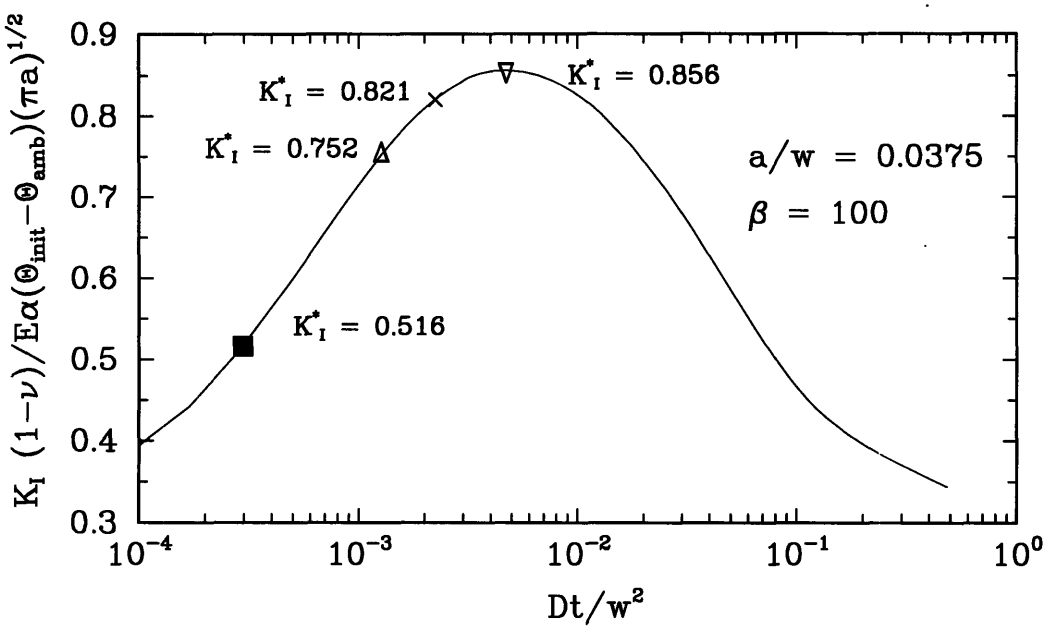


Figure 4.39: Stress intensity factors K_I^* for $a/w = 0.0375$ under combined thermal and mechanical loading as a function of nondimensional time Dt/w^2 ($\beta = 100$, $\sigma^\infty = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$).

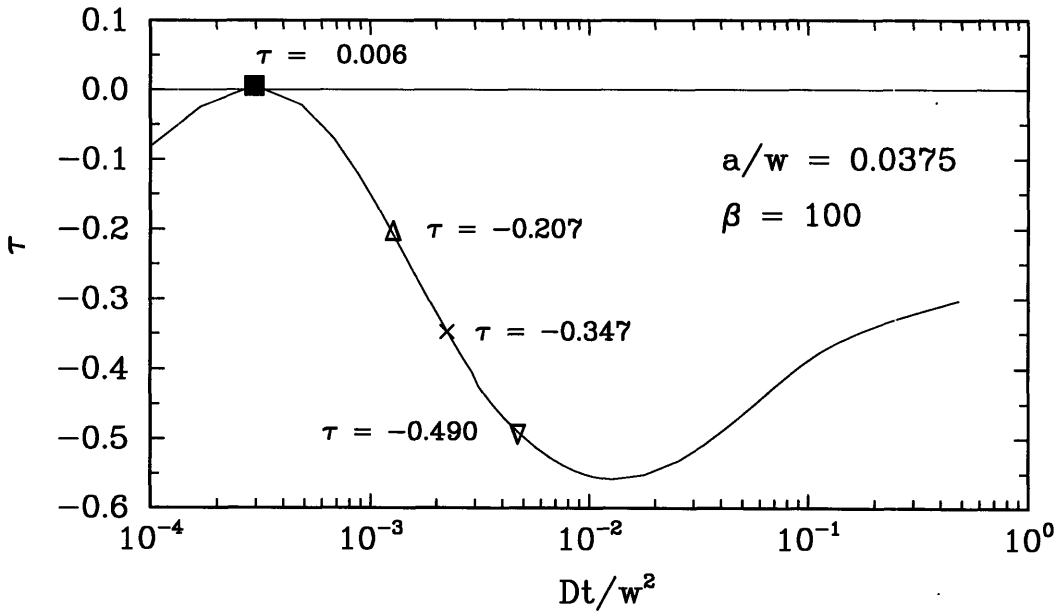


Figure 4.40: Non-dimensionalized T-stress values for $a/w = 0.0375$ under combined thermal and mechanical loading as a function of nondimensional time Dt/w^2 ($\beta = 100$, $\sigma^\infty = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$).

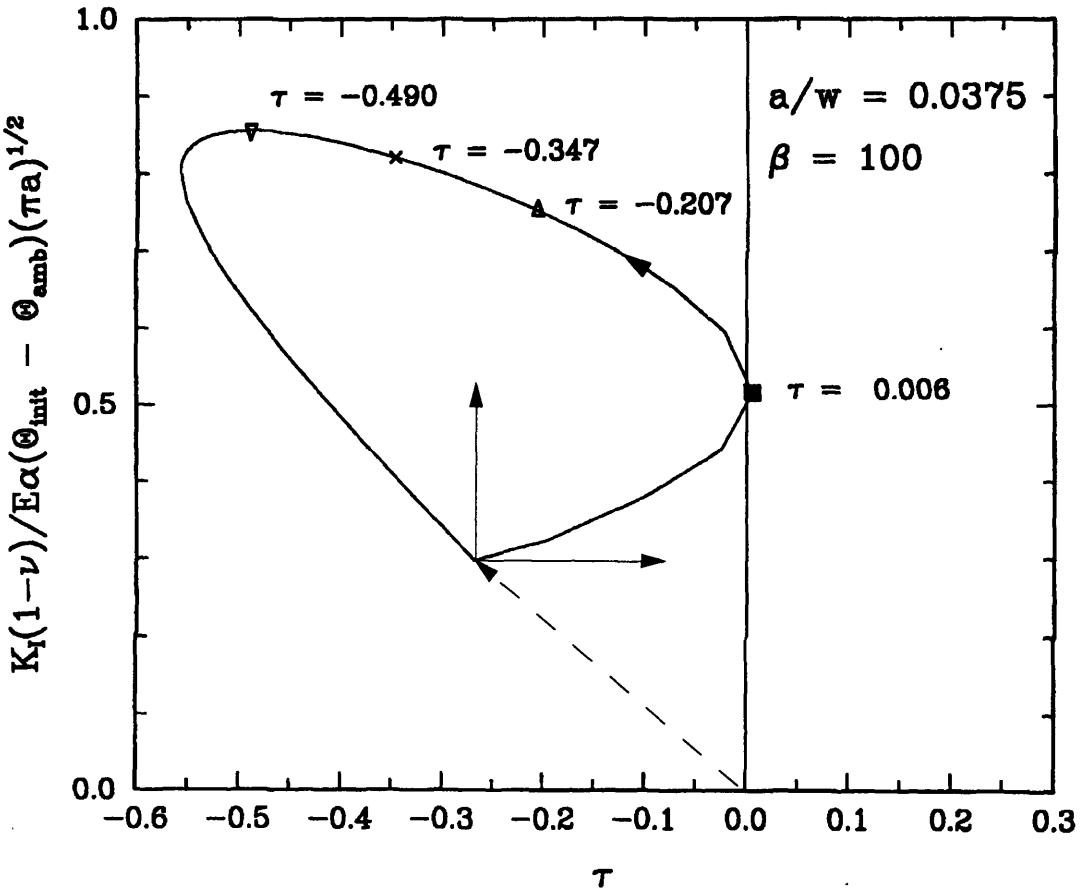
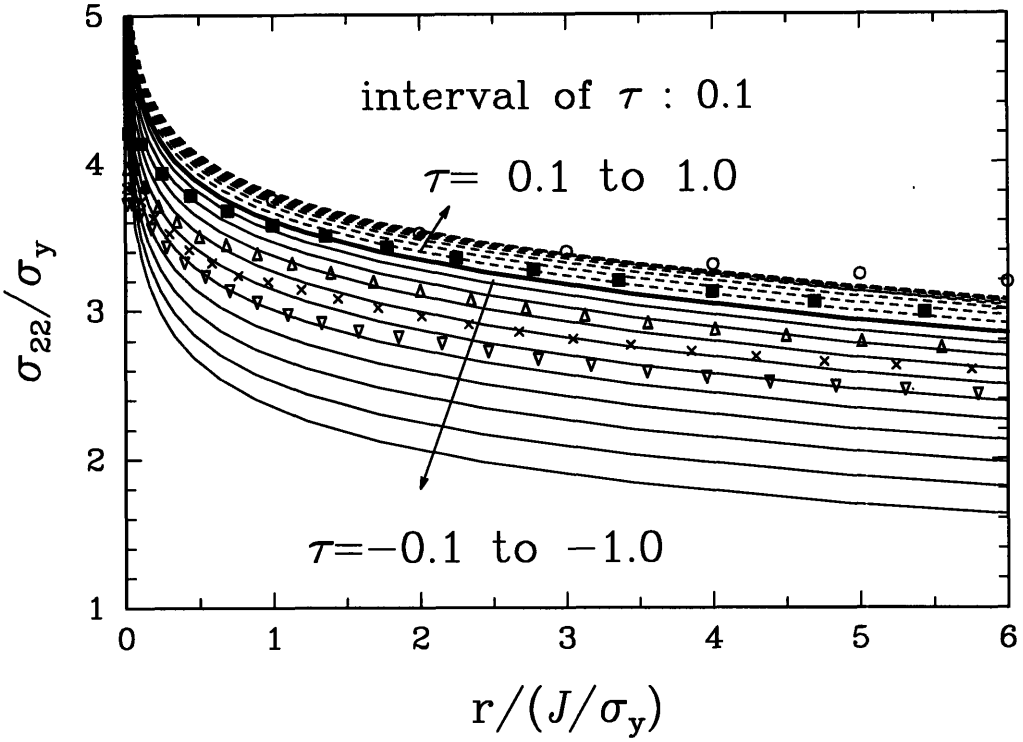
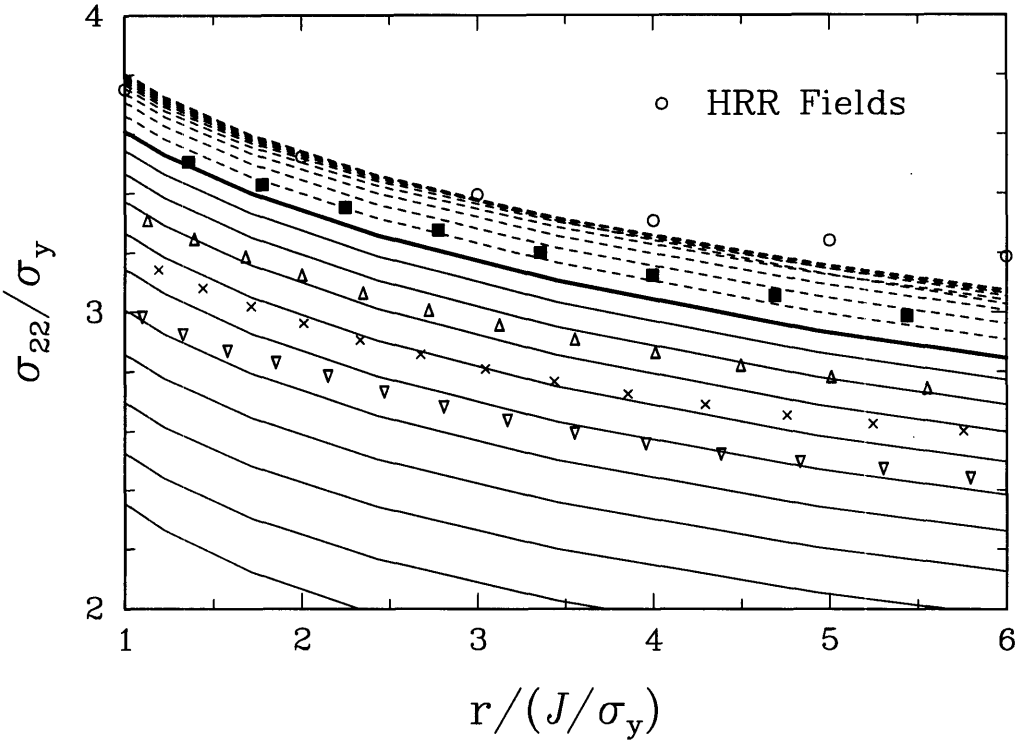


Figure 4.41: Non-dimensionalized K_I *vs.* T for $a/w = 0.0375$ under combined thermal and mechanical loading as a function of nondimensional time Dt/w^2 ($\beta = 100$, $\sigma^\infty = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$).



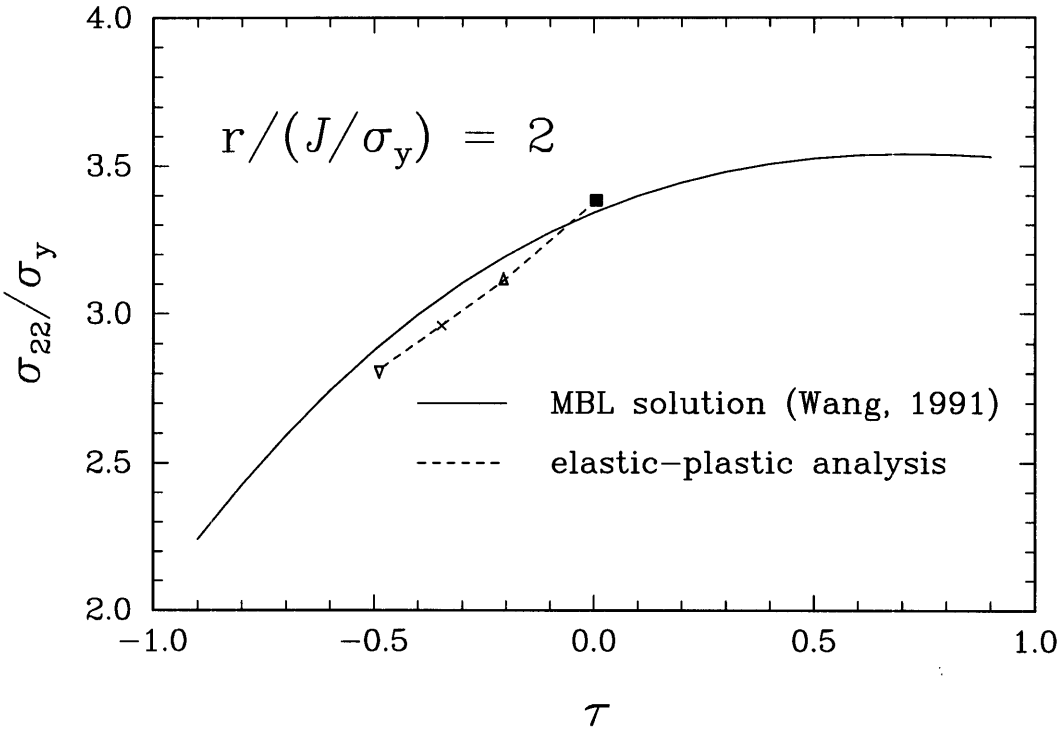
■ ■ ■	τ = 0.006	Fo = 2.961 x 10^{-4} K_I^* = 0.516
▲ ▲ ▲	τ = −0.207	Fo = 1.264 x 10^{-3} K_I^* = 0.752
× × ×	τ = −0.347	Fo = 2.236 x 10^{-3} K_I^* = 0.821
▼ ▼ ▼	τ = −0.490	Fo = 4.691 x 10^{-3} K_I^* = 0.856

Figure 4.42: Normalized crack-opening stress profiles at four instances of time and corresponding MBL solutions (a/w = 0.0375, β = 100, $\sigma^\infty = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). K_I^* are stress intensity factors normalized according to Eq. (4.2).



■ ■ ■	τ = 0.006	Fo = 2.961 x 10^{-4} K_I^* = 0.516
△ △ △	τ = −0.207	Fo = 1.264 x 10^{-3} K_I^* = 0.752
× × ×	τ = −0.347	Fo = 2.236 x 10^{-3} K_I^* = 0.821
▽ ▽ ▽	τ = −0.490	Fo = 4.691 x 10^{-3} K_I^* = 0.856

Figure 4.43: Enlarged view of Fig. 4.42. K_I^* are stress intensity factors normalized according to Eq. (4.2).



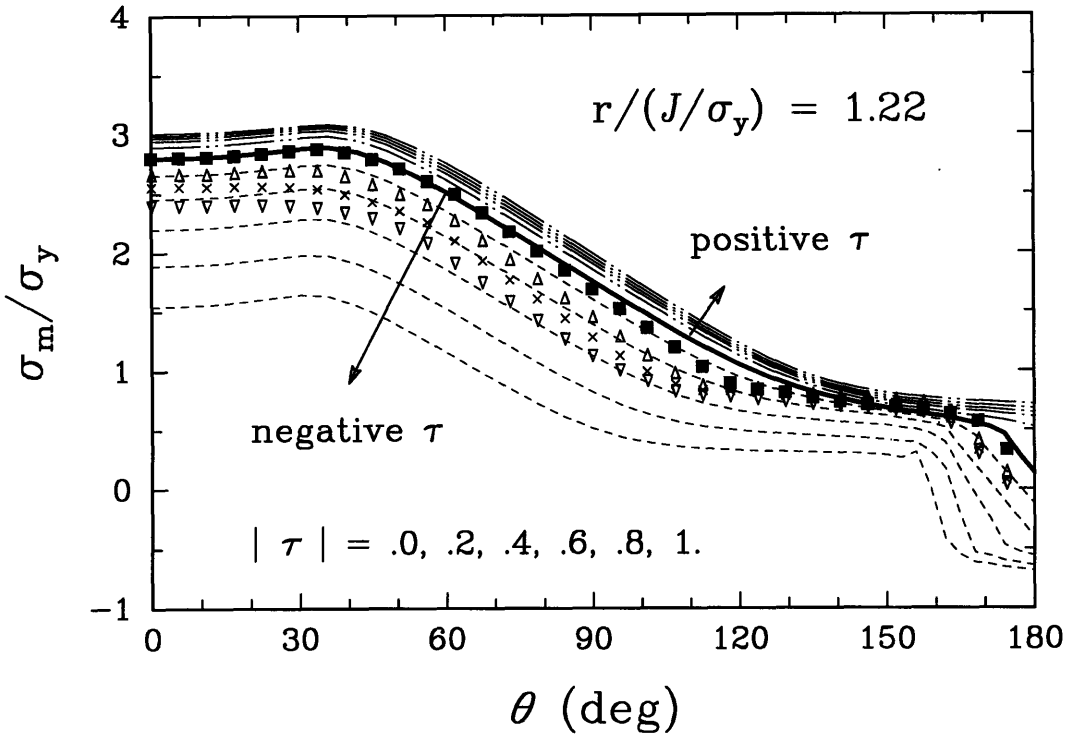
■ ■ ■ τ = 0.006 Fo = 2.961 x 10^{-4} K_I^* = 0.516

▲ ▲ ▲ τ = −0.207 Fo = 1.264 x 10^{-3} K_I^* = 0.752

× × × τ = −0.347 Fo = 2.236 x 10^{-3} K_I^* = 0.821

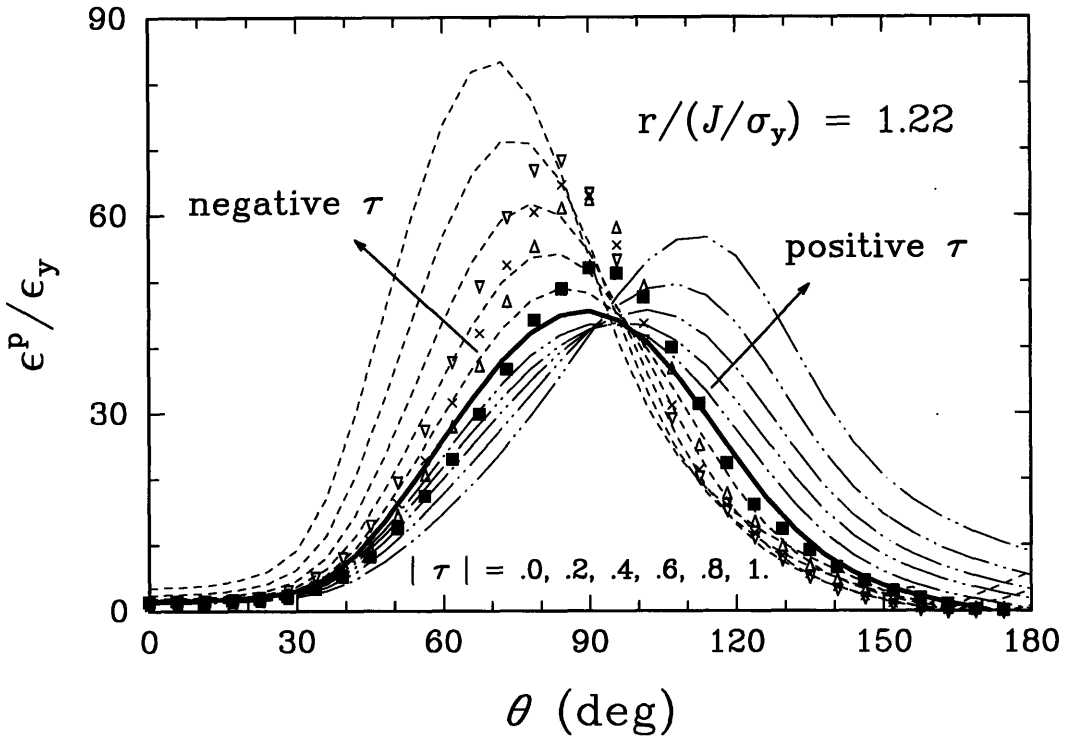
▽ ▽ ▽ τ = −0.490 Fo = 4.691 x 10^{-3} K_I^* = 0.856

Figure 4.44: Normalized crack-opening stresses at $r/(J/\sigma_y) = 2$ *vs.* τ at four instances of time and corresponding MBL results ($a/w = 0.0375$, $\beta = 100$, $\sigma^\infty = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$).



■ ■ ■	τ = 0.006	Fo = 2.961 x 10^{-4} K_I^* = 0.516
▲ ▲ ▲	τ = −0.207	Fo = 1.264 x 10^{-3} K_I^* = 0.752
× × ×	τ = −0.347	Fo = 2.236 x 10^{-3} K_I^* = 0.821
▽ ▽ ▽	τ = −0.490	Fo = 4.691 x 10^{-3} K_I^* = 0.856

Figure 4.45: Variation of hydrostatic stress at four instances of time and corresponding MBL solutions ($a/w = 0.0375$, $\beta = 100$, $\sigma^\infty = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). K_I^* are stress intensity factors normalized according to Eq. (4.2).



■ ■ ■	τ = 0.006	Fo = 2.961 x 10^{-4}
		K_I^* = 0.516
▲ ▲ ▲	τ = −0.207	Fo = 1.264 x 10^{-3}
		K_I^* = 0.752
× × ×	τ = −0.347	Fo = 2.236 x 10^{-3}
		K_I^* = 0.821
▼ ▼ ▼	τ = −0.490	Fo = 4.691 x 10^{-3}
		K_I^* = 0.856

Figure 4.46: Variation of equivalent plastic strain at four instances of time and corresponding MBL solutions ($a/w = 0.0375$, $\beta = 100$, $\sigma^\infty = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$). K_I^* are stress intensity factors normalized according to Eq. (4.2).

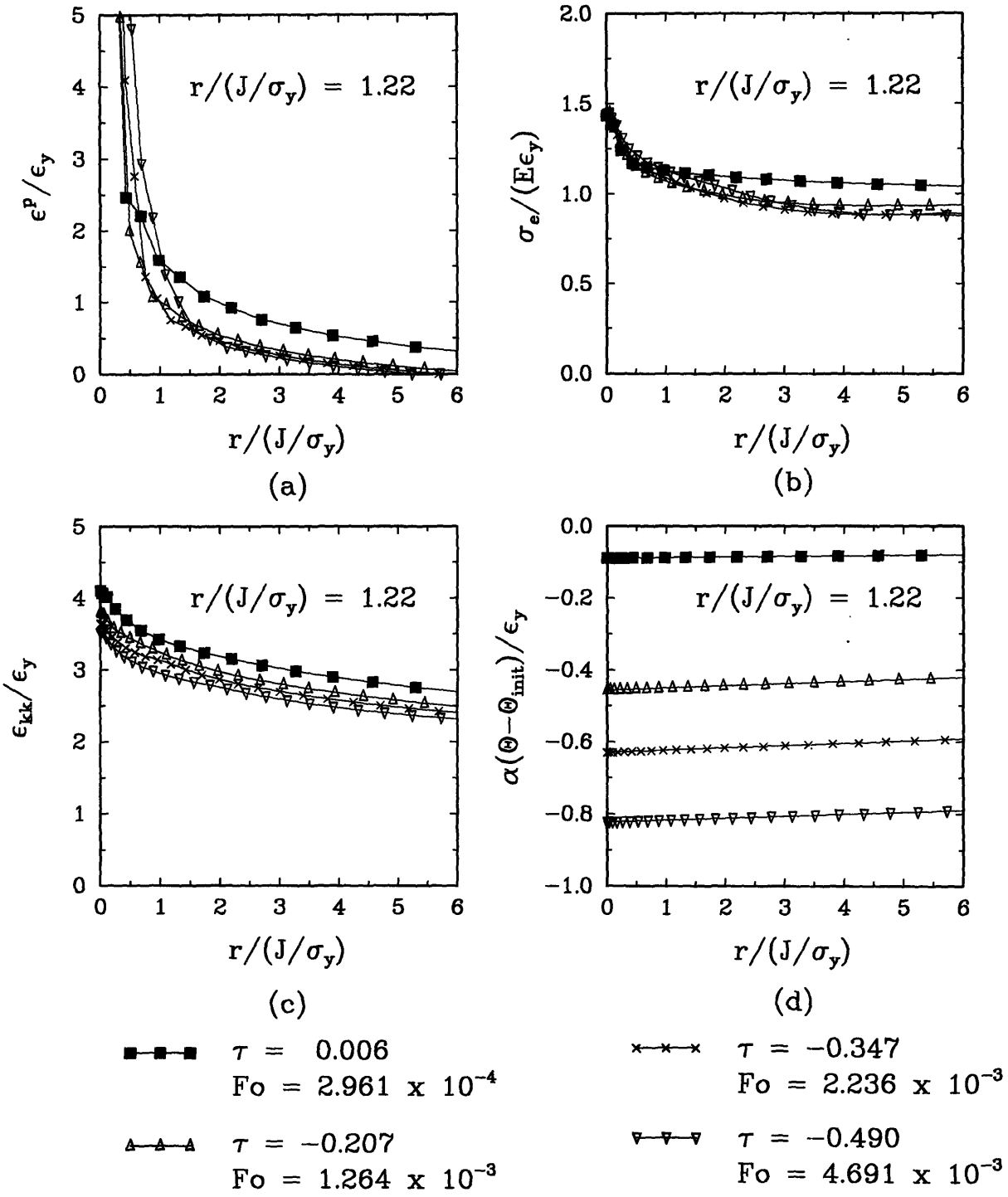


Figure 4.47: Strain components ahead of the crack at four instances of time during the thermal transient ($a/w = 0.0375$, $\beta = 100$, $\sigma^{\infty} = \sigma_y/2$, $E\alpha(\Theta_{init} - \Theta_{amb})/\sigma_y = 1.32$).

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In summary, we have examined and validated the two-parameter characterization for the case of transient thermal loading. The ability of the MBL solutions in predicting the stress state of the elastic-plastic solutions is exceptional considering that the MBL loading is based on the first two terms of the WILLIAMS eigen-expansion, which neglects the presence of thermal strains in its derivation. That is, simple extraction of the T-stress variation from an elastic analysis of the problem allows us to predict the triaxiality of the stress state in the elastic-plastic full-field solution.

The importance of these results becomes even clearer when the T-stress effect on fracture toughness is taken into consideration. BETEGÓN & HANCOCK, (1990), for example, examined the dependence of cleavage fracture toughness on τ in three-point-bend specimens of various crack depths. The varying crack depth provided large variation of crack-tip constraint (thus a large range of τ values). Their results in terms of (J_c, τ) at final failure are shown in Fig. 5.1. The experimental data has a large scatter, as do most cleavage toughness tests. Clearly, though, the cleavage fracture toughness shows a significant increase at large negative values of τ.

Thus, for negative τ-values, the observed drop in crack-opening stress profiles is associated with a simultaneous increase in fracture toughness. This means that failure, otherwise predicted using a single-parameter approach, will not occur for these cases. The temperature dependence of the fracture toughness should not be neglected here. Namely, toughness generally decreases with decreasing temperature (see Fig. 5.2). The temperature variation at the crack tip of relative crack depth $a/w = 0.375$ for an overcooling transient, for example, is shown in Fig. 5.3. That is, during a pressurized overcooling transient three interacting effects have to be taken into consideration: (a) the toughness decreases due to the drop in temperature; (b) the acting dead-loads shift the K_I^*-τ locus into the negative quadrant and thus into a "safer" area with respect to the fracture toughness which is increased due to the influence of the T-stress; and finally, (c), the K_I^*- τ - Θ spiral traversed in this 3-D space, reaches its maximum K_I^*-value at an even more negative value of τ, thus resulting in a further "gain" with respect to the fracture toughness, counteracting the adverse temperature effect. This result is schematically depicted in Figs. 5.4 and 5.5.

Fig. 5.6 from a review paper by STAHLKOPF (1982) puts these results into contact with traditional one-parameter-based approaches for the assessment of vessel integrity. Shown in the figure are the cracking response of two hypothetical reactor vessels exposed to a severe overcooling transient. Vessel (a) represents a case of severe embrittlement; vessel (b) a case of light embrittlement. A one-parameter based approach predicts grack growth for the highly embrittled reactor vessel, and no crack initiation for the case of low embrittlement. On the contrary, using a two-parameter approach, the prediction for the case of severe embrittlement could look like the prediction for the case of light embrittlement if the relative crack depth were sufficiently small.

In the ongoing efforts aimed at developing two-parameter descriptions of crack-tip fields, the results of this work confirm the T-stress as *the* rigorous "second" crack-tip parameter in well-contained yielding (PARKS, 1992). Besides this work, ample evidence exists (BETEGÓN & HANCOCK (1991), AL-ANI & HANCOCK (1991)) that the

T-stress is valuable in characterizing the stress triaxiality of plane strain and 3-D elastic-plastic crack-tip fields. This strong dependence of crack-tip stress triaxiality on τ, along with the marked sensitivity of both ductile (void growth) and brittle (cleavage) fracture mechanisms to stress triaxiality, has profound influences on crack toughness and growth ductility. Thus, the parameter τ, as elastically calculated based on the applied loading, plus the parameter J, as calculated based on the actual elastic-plastic deformation field, rigorously and accurately describe the local crack-tip stress and deformation (PARKS, 1992).

5.2 Future Work

The most obvious extension of this work is the investigation of the T-effect on the near-tip stress fields using temperature-dependent material properties. KOKINI (1986), for example, showed that for the problem of a strip containing an edge crack using constant material properties over large temperature ranges can lead to considerable underestimation of the maximum stress intensity factors.

To realize the full potential the two-parameter characterization of elastic-plastic fields offers, systematic experimental testing is needed to establish parametric limits. This needs to be done in conjunction with further numerical investigation of the limits of the two-parameter approach in predicting the near-crack-tip fields of various specimens (WANG, 1991).

RICE (1972) showed that the so-called line-spring could be used to calculate K_I for thermal or residual stresses that vary through the thickness of a plate. By applying the "no-crack" tractions as reverse pressures on the crack faces, the extra elastic compliance and stress intensity factor can be readily computed. PATIL (1993) implemented these methods into a line-spring program. WANG & PARKS (1992) demonstrated how this line spring-model can be used to calculate T for mechanical loading of the single-

face-cracked specimen subject to combined tension and bending. A natural extension of their work could explore the case of combined mechanical/transient thermal loading.

The performance of the modified effective crack-length-formulation of HAUF et al. (1994) for contained yielding in the presence of thermal stresses could also be examined. The nonlinear compliance of mechanical work-conjugate displacements, $\mathbf{q}$, should be affected by the presence of thermal stresses.

Examination of the T-effect on three-dimensional stress fields and extension to other specimen geometries constitute further possible areas of future work. Finally, any future work on this topic aimed at the nulear power industry should examine the attenuation of fast neutrons through the reactor wall and attendant radiation embrittlement gradients.

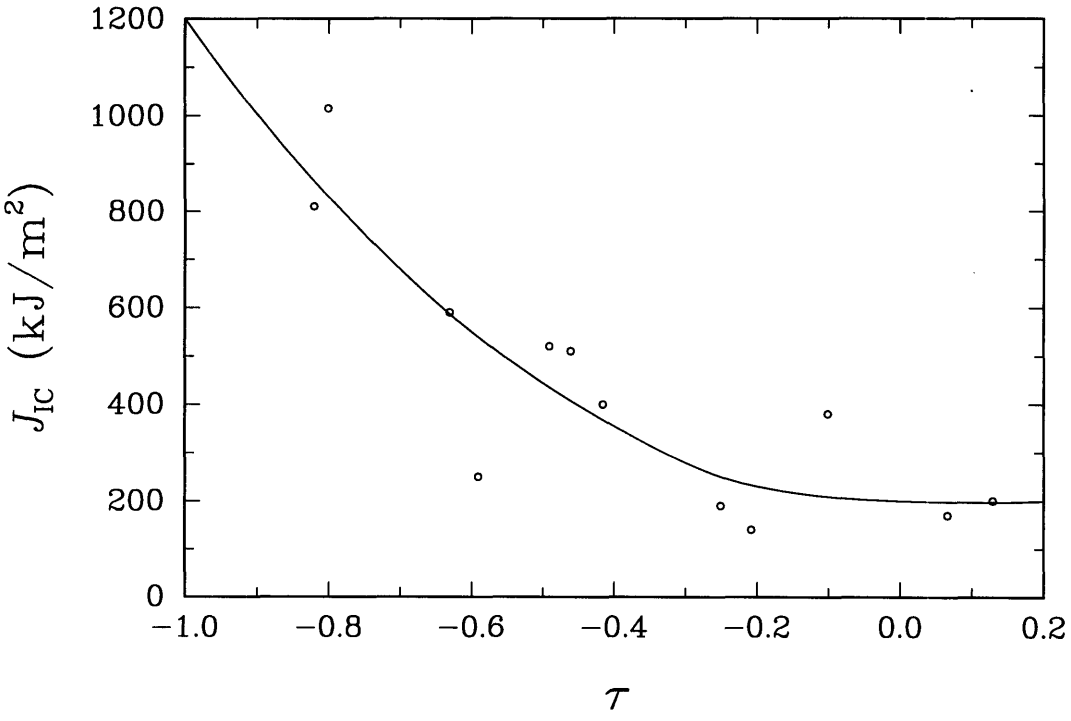


Figure 5.1: Variation of cleavage fracture toughness with τ (Betegón and Hancock, 1990).

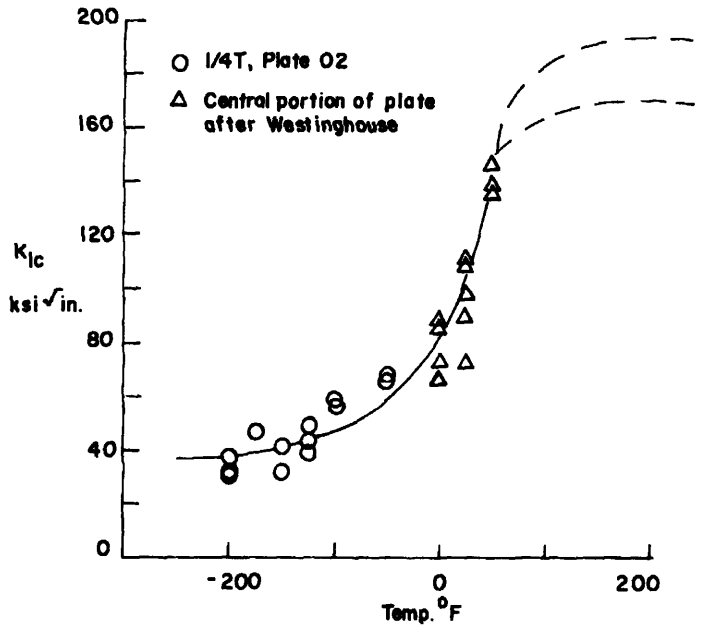


Figure 5.2: Variation of fracture toughness K_{IC} with temperature for A533 B Steel (Sailors and Corten, 1972).

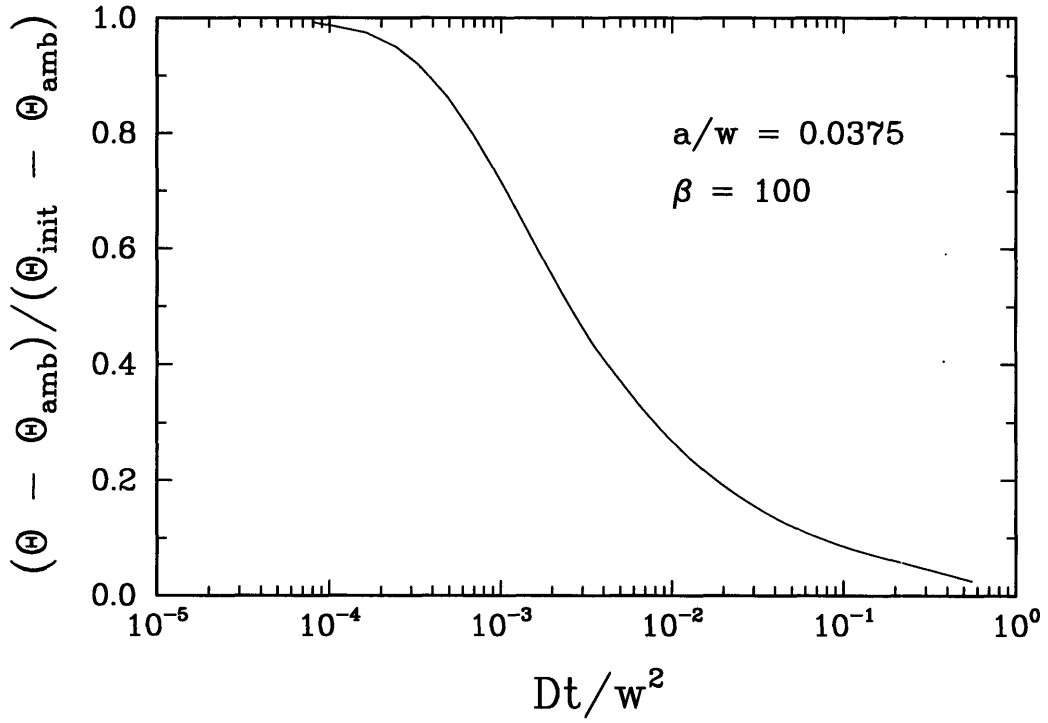


Figure 5.3: Variation of crack-tip temperature during a thermal transient ($a/w = 0.0375, \beta = 100$).

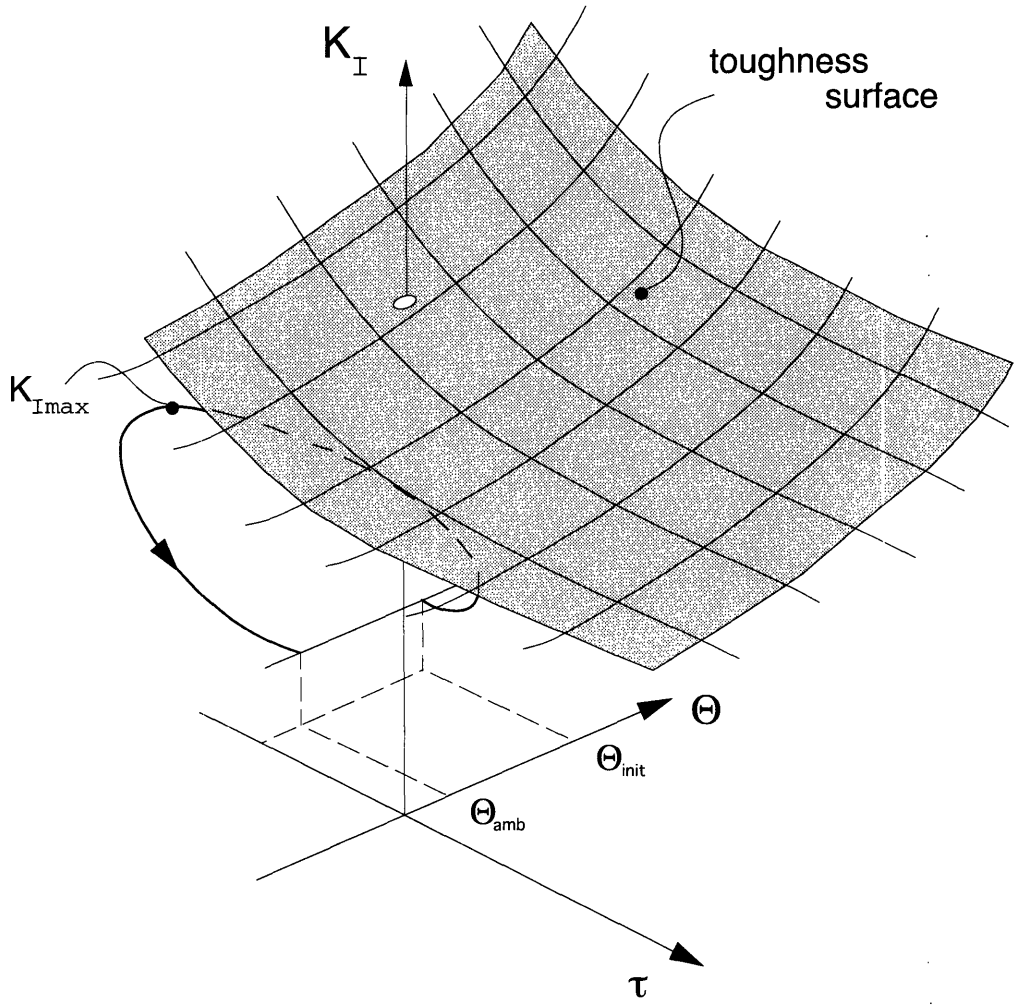


Figure 5.4: Schematic variation of fracture toughness as a function of τ and Θ and $\tau - K_I^* - \Theta$ spiral.

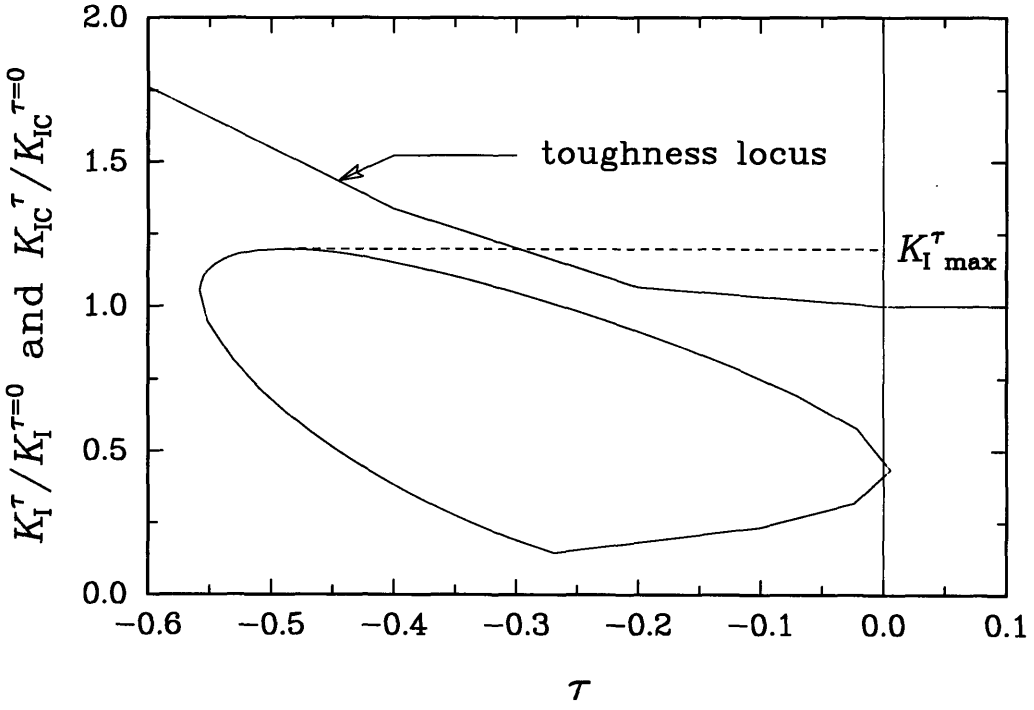


Figure 5.5: Two-dimensional slice of Fig. 5.4.

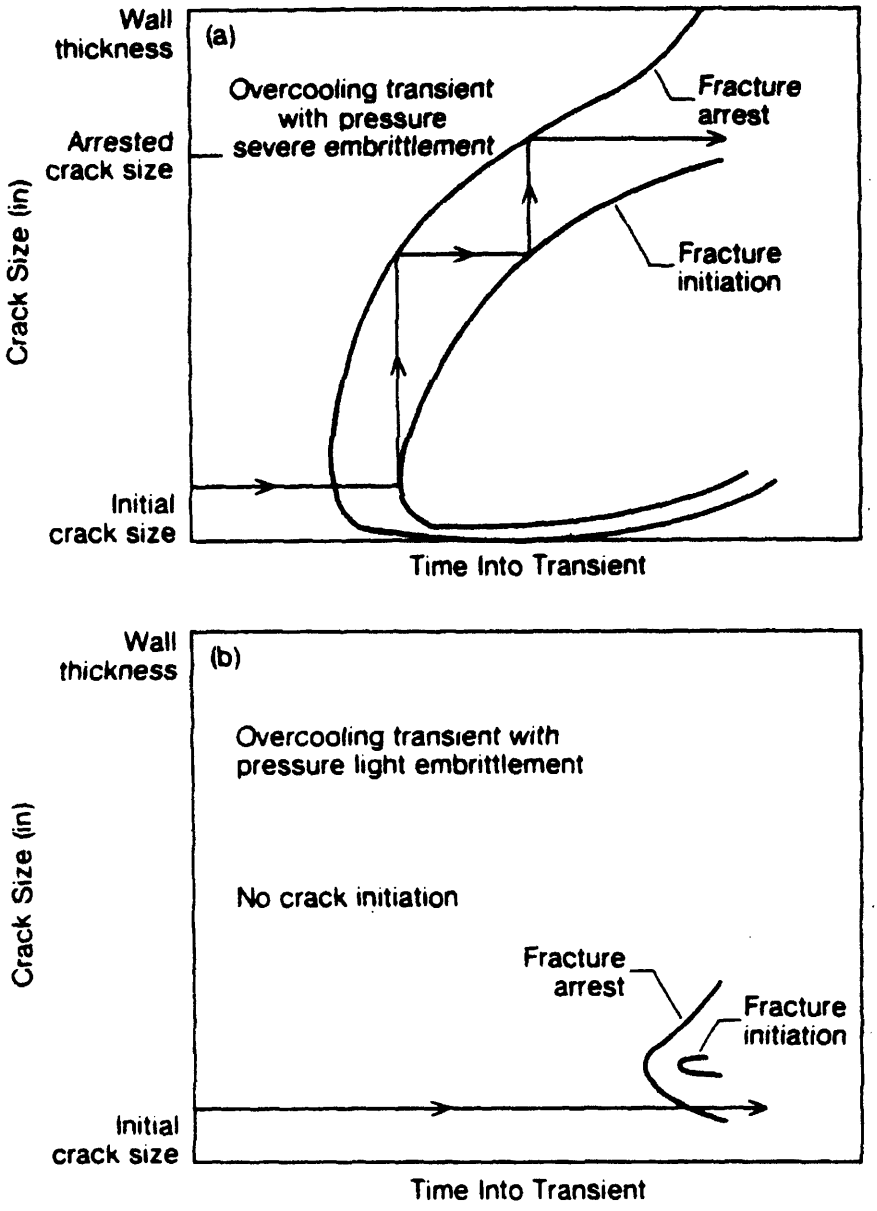


Figure 5.6: Prediction of the behaviour of a 1-in. deep crack during an overcooling transient for a reactor vessel with (a) severe and (b) light embrittlement (Stahlkopf, 1982).

References

AL-ANI, A. M., and HANCOCK, J. W., 1991, "J-Dominance of Short Cracks in Tension and Bending," *Journal of the Mechanics and Physics of Solids*, Vol. 39, No. 1, pp. 23-43.

ASME, 1974, Rules of in service and inspection of nuclear power plant components, *Boiler and Pressure Vessel Code*, Section XI, New York.

ASTM, 1983, Standard Test Method for Plane-Strain Fracture Toughness of Metallic Materials, *Annual Book of ASTM Standards*, E399-83, American Society for Testing and Materials, Philadelphia, PA.

BETEGÓN, C., and HANCOCK, J. W., 1990, *Proceedings ECF8 Turin*, (Edited by D. Firrao), EMAS, Warley, UK.

BETEGÓN, C., and HANCOCK, J. W., 1991, "Two-Parameter Characterization of Elastic-Plastic Crack-Tip Fields," *Journal of Applied Mechanics*, Vol. 58, pp. 104-110.

BILBY, B. A., CARDEW, G. E., GOLDTHORPE, M. R., and HOWARD, I. C., 1986, "A Finite Element Investigation of the Effect of Specimen Geometry on the Fields of Stress and Strain at the Tips of Stationary Cracks," in *Size Effects in Fracture*, The Institution of Mechanical Engineers, London, 1986, pp. 37-46.

CARDEW, G. E., GOLDTHORPE, I. C., HOWARD, I. C., and KFOURI, A. P., 1984, in *Fundamentals of Deformation and Fracture*, Eshelby Memorial Symposium, Cambridge University Press, pp. 465-476.

CARSLAW, H. S., and JAEGER, J. C., 1950, *Conduction of Heat in Solids*, Oxford University Press.

DU, Z.-Z., and HANCOCK, J. W., 1991, "The Effect of Non-Singular Stresses on Crack-

Tip Constraint," *Journal of the Mechanics and Physics of Solids,* Vol. 39, pp. 555-567.

HANCOCK, J. W., REUTER, W. G., and PARKS, D. M., 1993, "Constraint and Toughness Parameterized by T," *Constraint Effects in Fracture, ASTM STP 1171* (Edited by E. M. Hackett, K.-H. Schwalbe, and R. H. Dodds), American Society for Testing and Materials, Philadelphia, pp. 21-40.

HARLIN, G., and WILLIS, J. R., 1988, "The Influence of Crack Size on the Ductile-Brittle Transition," *Proceedings of Royal Society of London A415,* pp. 197-226.

HAUF, D. E., PARKS, D. M., and LEE, H., 1994, "A Modified Effective Crack Length Formulation in Elastic-Plastic Fracture Mechanics," manuscript submitted to *Mechanics and Materials.*

HIBBITT, KARLSSON and SORENSEN, Inc., 1992, *ABAQUS User's Manual*, version 5.2, Hibbitt, Karlsson and Sorensen, Inc., Pawtucket, RI.

HUTCHINSON, J. W., 1968, "Singular Behavior at the End of a Tensile Crack in a Hardening Material," *Journal of the Mechanics and Physics of Solids*, Vol. 16, pp. 13-31.

KFOURI, A. P., 1986, "Some Evaluations of the Elastic T-term Using Eshelby's Method," *International Journal of Fracture*, Vol. 30, pp. 301-315.

KOKINI, K., 1986, "Thermal Shock of a Cracked Strip: Effect of Temperature-Dependent Material Properties", *Engineering Fracture Mechanics*, Vol. 25, pp. 167-176.

LI, F. Z., SHIH, C. F., and NEEDLEMAN, A., 1985, "A Comparison of Methods for Calculating Energy Release Rate," *Engineering Fracture Mechanics*, Vol. 21, pp. 405-421.

LARSSON, S. G., and CARLSSON, A. J., 1973, "Influence of Non-singular Stress Terms

and Specimen Geometry on Small-Scale Yielding at Crack Tips in Elastic-Plastic Material," *Journal of the Mechanics and Physics of Solids*, Vol. 21, pp. 263-277.

LEEVERS, P. S., and RADON, J. C., 1982, "Inherent Stress Biaxiality in Various Fracture Specimen Geometries," *International Journal of Fracture*, Vol. 19, pp. 311-325.

MCMEEKING, R. M., 1977, "Finite Deformation Analysis of Crack-Tip Opening in Elastic-Plastic Materials and Implications for Fracture," *Journal of the Mechanics and Physics of Solids*, Vol. 25, pp. 357-381.

MCMEEKING, R. M., and PARKS, D. M., 1979, "On Criteria for J-Dominance of Crack-Tip Fields in Large-Scale Yielding," *Elastic-Plastic Fracture*, ASTM STP 668, American Society for Testing and Materials, Philadelphia, pp. 175-194.

MORAN, B., and SHIH, C. F., 1987, "Crack Tip and Associated Domain Integrals from Momentum and Energy Balance," *Engineering Fracture Mechanics*, Vol. 27, No. 6, pp. 615-642.

NAGTEGAAL, J. C. , PARKS, D. M., and RICE, J. R., 1974, "On Numerically Accurate Finite Element Solutions in the Fully Plastic Range", *Computer Methods in Applied Mechanics and Engineering,* Vol. 4, pp. 153-177.

NAKAMURA, T., and PARKS, D. M., 1992, "Determination of Elastic T-Stress along three-dimensional Crack Fronts Using an Interaction Integral," *International Journal of Solids and Structures*, Vol. 29, No. 13, pp. 1597-1611.

NIED, H. F., 1983, "Thermal Shock Fracture in an Edge-Cracked Plate," *Journal of Thermal Stresses*, Vol. 6, pp. 217-229.

O'DOWD, N. P., and SHIH, C. F., 1991, "Family of Crack-Tip Fields characterized by a Triaxiality Parameter-I. Structure of Fields," *Journal of the Mechanics and Physics of Solids*, Vol. 39, No.8, pp. 989-1015.

O'Dowd, N. P., and Shih, C. F., 1992, "Family of Crack-Tip Fields characterized by a Triaxiality Parameter-II. Fracture Applications," *Journal of the Mechanics and Physics of Solids*, Vol. 40, No.5, pp. 939-963.

Parks, D. M., 1992, "Advances in Characterization of Elastic-Plastic Crack-Tip Fields." In *Topics in Fracture and Fatigue*, McClintock Festschrift (Edited by A. S. Argon) pp. 58-98. Springer Verlag, New York.

Patil, R. R., 1993, "Computer Simulation of Thermal Stress Transients in Cracked Structures," SB Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, May, 1993.

Rice, J. R., 1968, "A Path Independent Integral and the Approximate Analysis of Strain Concentrations by Notches and Cracks," *Journal of Applied Mechanics*, Vol. 35, pp. 379-386.

Rice, J. R., 1972, "The Line-Spring Model for Surface Flaws," in *The Surface Crack: Physical Problems and Computational Solutions*, J. L. Swedlow, Eds., American Society of Mechanical Engineers, New York, pp. 171-185.

Rice, J. R., 1974, "Limitations to the Small-Scale Yielding Approximation for Crack-Tip Plasticity," *Journal of the Mechanics and Physics of Solids*, Vol. 22, pp. 17-26.

Rice, J. R., and Rosengren, G. F., 1968, "Plane Strain Deformation Near a Crack Tip in a Power Law Hardening Material," *Journal of the Mechanics and Physics of Solids,* Vol. 16, pp. 1-12.

Sailors, R. H., and Corten, H. T., 1972, "Relationship between Material Fracture Toughness using Fracture Mechanics and Transition Temperature Tests," *Fracture Toughness, Preceedings of the 1971 National Symposium on Fracture Mechanics, Part II, ASTM STP 514*, American Society for Testing and Materials, pp. 164-191.

SHAM, T.–L., 1991, "The Determination of the Elastic T-term Using Higher Order Weight Functions," *International Journal of Fracture*, Vol. 48, pp. 81-102.

SHIH, C. F., 1983, "Tables of Hutchinson-Rice-Rosengren Singular Field Quantities," Division of Engineering, Brown University, Providence, RI, June 1983.

SHIH, C. F., MORAN, B., and NAKAMURA, T., 1986, "Energy Release-Rate along a Three-Dimensional Crack Front in a Thermally Stressed Body," *International Journal of Fracture*, Vol. 30, pp. 79-102.

SHIH, C. F., O'DOWD, N. P., and KIRK, M. T., 1993, "A Framework for Quantifying Crack Tip Constraint," *Constraint Effects in Fracture, ASTM STP 1171* (Edited by E. M. Hackett, K.-H. Schwalbe, and R. H. Dodds), American Society for Testing and Materials, Philadelphia, pp. 2-20.

SOCRATE, S., 1990, "Numerical Determination of Forces Acting on Material Interfaces: An Application to Rafting in Ni-Superalloys," MS Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, August, 1990.

STAHLKOPF, K. E., 1982, "Light Water Reactor Pressure Boundary Components: A Critical Review of Problems," in *Structural Integrity of Light Water Reactor Components* (Edited by L. E. Steele, K. E. Stahlkopf, and L. H. Larsson), Applied Science Publishers, London, pp. 29-54.

STERNBERG, E., and CHAKRAVORTY, J. G., 1959, "On Inertia Effects in a Transient Thermoelastic Problem," *Journal of Applied Mechanics*, Vol. 26, p. 503.

STERNBERG, E., and CHAKRAVORTY, J. G., 1959, "Thermal Shock in an Elastic Body with a Spherical Cavity," *Q. of Applied Mathematics*, Vol. 17, p. 205.

TIMOSHENKO, S. P., and GOODIER, 1970, *Theory of Elasticity*, 3rd Edition, McGraw-Hill, Inc.

WANG, Y.–Y., 1991, "A Two-Parameter Characterization of Elastic-Plastic Crack-Tip Fields and Applications to Cleavage Fracture," PhD. Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, Sept., 1991.

WANG, Y.–Y., 1993, "On the Two-Parameter Characterization of Elastic-Plastic Crack-Tip Fields in Surface-Cracked Plates," *Constraint Effects in Fracture, ASTM STP 1171* (Edited by E. M. Hackett, K.-H. Schwalbe, and R. H. Dodds), American Soci ety for Testing and Materials, Philadelphia, pp. 120-138.

WANG, Y.–Y., and PARKS, D. M., 1992, "Evaluation of the Elastic T-stress in Surface-Cracked Plates Using the Line-Spring Method," *International Journal of Fracture*, Vol. 56, pp. 25-40.

WILLIAMS, M. L., 1957, "On the Stress Distribution at the Base of a Stationary Crack," *Journal of Applied Mechanics*, Vol. 24, pp. 111-114.

Appendix A

Derivation of Line-Load Strain and Displacement Fields

Given the stress fields, Eq. (3.2), we derive the corresponding strain and displacement fields needed in the evaluation of the Interaction Integral of Chapter 2. Using the strain-displacement relations in polar coordinates and the elastic constitutive equations for plane strain, we have

$$\begin{aligned} \varepsilon_{rr} &= \frac{\partial u_r}{\partial r} = -\frac{f}{\pi E}\frac{\cos\theta}{r}(1-\nu^2), \\ \varepsilon_{\theta\theta} &= \frac{u_r}{r} + \frac{1}{r}\frac{\partial u_\theta}{\partial \theta} = \nu\frac{f}{\pi E}\frac{\cos\theta}{r}(1+\nu) \\ \varepsilon_{r\theta} &= \frac{1}{2}\left(\frac{1}{r}\frac{\partial u_r}{\partial \theta} + \frac{\partial u_\theta}{\partial r} - \frac{u_\theta}{r}\right) = 0. \end{aligned} \tag{A.1}$$

Integrating the first of these equations, we find

$$u_r = \int -\frac{f}{\pi E}\frac{\cos\theta}{r}(1-\nu^2)\,dr = -\frac{f}{\pi E}\cos\theta(1-\nu^2)\log r + F(\theta), \tag{A.2}$$

where $F(\theta)$ is a function of θ only.

Substituting in the second of Eqs. (A.1) and integrating it, we obtain

$$u_\theta = \frac{\nu f}{\pi E}\sin\theta\,(1+\nu) + \frac{f}{\pi E}\sin\theta\log r\,(1-\nu^2) - \int F(\theta)\,d\theta + G(r), \tag{A.3}$$

in which $G(r)$ is a function of r only.

Partially differentiating Eq. (A.2) by θ and Eq. (A.3) by r, we have

$$\frac{\partial u_r}{\partial \theta} = \frac{f}{\pi E}\sin\theta\,(1-\nu^2) + \frac{\partial F(\theta)}{\partial \theta} \tag{A.4}$$

$$\frac{\partial u_\theta}{\partial r} = \frac{f}{\pi E}\sin\theta\,\frac{1}{r}(1-\nu^2) + \frac{\partial G(r)}{\partial r}. \tag{A.5}$$

Substituting (A.4) and (A.5) in the third of Eqs. (A.1), we conclude that

$$F(\theta) = -\frac{(1-\nu-\nu^2)}{2\pi E} f\,\theta\,\sin\theta + A\sin\theta + B\cos\theta \tag{A.6}$$

$$G(r) = Cr, \tag{A.7}$$

where A, B, and C are constants of integration to be determined from the conditions of constraints. Using Eqs. (A.6) and (A.7) in Eqs. (A.2) and (A.3), the expressions for the displacements are

$$u_r = -\frac{f}{\pi E}\cos\theta\,(1-\nu^2)\log r - \frac{(1-\nu-2\nu^2)}{2\pi E} f\,\theta\,\sin\theta + A\sin\theta + B\cos\theta, \tag{A.8}$$

$$u_\theta = \frac{\nu f}{\pi E}\sin\theta\,(1+\nu) + \frac{f}{\pi E}\sin\theta\,\log r\,(1-\nu^2) + \frac{(1-\nu-2\nu^2)}{2\pi E} f\,[\sin\theta - \theta\cos\theta] + A\cos\theta - B\sin\theta + Cr. \tag{A.9}$$

The constraint is such that the points on the x axis have no lateral displacement. Then $u_\theta = 0$, for $\theta = 0$, and we find from Eq. (A.9) that $A = 0$, $C = 0$. The constant B is determined considering a material point lying on the x axis at distance d, say, from the origin which does not move out radially. From Eq. (A.8) we then obtain that $B = \frac{f}{\pi E}(1-\nu^2)\log d$.

By geometric considerations it is possible to derive the displacement field in cartesian coordinates from the polar components. Namely

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} u_r \\ u_\theta \end{bmatrix}. \tag{A.10}$$

Taking the derivative of u_1 and u_2 with respect to x_2 and x_1, respectively, we have

$$\begin{aligned}
\frac{\partial u_1}{\partial x_2} &= \frac{\partial r}{\partial x_2}\left\{\frac{\partial u_r}{\partial r}\cos\theta - \frac{\partial u_\theta}{\partial r}\sin\theta\right\} + \\
&\quad \frac{\partial \theta}{\partial x_2}\left\{\frac{\partial u_r}{\partial \theta}\cos\theta - u_\theta\cos\theta - \frac{\partial u_\theta}{\partial \theta}\sin\theta - u_r\sin\theta\right\}. \qquad \text{(A.11)} \\
\frac{\partial u_2}{\partial x_1} &= \frac{\partial r}{\partial x_1}\left\{\frac{\partial u_r}{\partial r}\sin\theta + \frac{\partial u_\theta}{\partial r}\cos\theta\right\} + \\
&\quad \frac{\partial \theta}{\partial x_1}\left\{\frac{\partial u_r}{\partial \theta}\sin\theta + u_r\cos\theta + \frac{\partial u_\theta}{\partial \theta}\cos\theta - u_\theta\sin\theta\right\}. \qquad \text{(A.12)}
\end{aligned}$$

Then, after taking the derivatives of u_r and u_θ that appear in Eqs. (A.11) and (A.12), considering that

$$\frac{\partial r}{\partial x_1} = \cos\theta \qquad \text{and} \qquad \frac{\partial \theta}{\partial x_1} = -\frac{\sin\theta}{r},$$

and substituting everything in Eq. (A.12), we obtain

$$\frac{\partial u_1}{\partial x_2} = \frac{\sin\theta}{r}\left\{\frac{f}{\pi E}(1+\nu)(\nu - \cos^2\theta - 1)\right\} \qquad \text{(A.13)}$$

and

$$\frac{\partial u_2}{\partial x_1} = -\frac{\sin\theta}{r}\left\{\frac{f}{\pi E}(1+\nu)(\nu - \sin^2\theta)\right\}. \qquad \text{(A.14)}$$

The strain components in cartesian coordinates can be obtained directly starting from the stress field (Eqs. (2)) and using the elastic constitutive equations for plane strain. Then

$$\begin{aligned}
\varepsilon_{11} &= \frac{\partial u_1}{\partial x_1} = \frac{1}{E}[(1-\nu^2)\sigma_{11} - \nu(1+\nu)\sigma_{22}] \\
&= \frac{1}{E}\left[(1-\nu^2)\left(-\frac{f}{\pi r}\cos^3\theta\right) + \nu(1+\nu)\left(\frac{f}{\pi r}\cos\theta\sin^2\theta\right)\right] \\
\varepsilon_{22} &= \frac{\partial u_2}{\partial x_2} = \frac{1}{E}[(1-\nu^2)\sigma_{22} - \nu(1+\nu)\sigma_{11}] \\
&= \frac{1}{E}\left[(1-\nu^2)\left(-\frac{f}{\pi r}\cos\theta\sin^2\theta\right) + \nu(1+\nu)\left(\frac{f}{\pi r}\cos^3\theta\right)\right] \qquad \text{(A.15)} \\
\varepsilon_{33} &= \frac{\partial u_3}{\partial x_3} = 0
\end{aligned}$$

$$
\begin{aligned}
\varepsilon_{12} &= \frac{1}{2}\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right) = \frac{(1+\nu)}{E}\sigma_{12} \\
&= -\frac{(1+\nu)}{E}\left(\frac{f}{\pi r}\cos^2\theta\sin\theta\right) \\
\varepsilon_{13} &= \varepsilon_{23} = 0\,.
\end{aligned}
$$

Appendix B

Temperature Distribution under Convective Cooling

The uncoupled transient temperature distribution for the strip in Fig. 4.1 may be determined from the solution of the one-dimensional diffusion equation

$$\frac{\partial^2 \Theta^*(x,t)}{\partial x^2} = \frac{1}{D}\frac{\partial \Theta^*(x,t)}{\partial t}, \tag{B.1}$$

where

$$\Theta^*(x,t) = \Theta(x,t) - \Theta_{amb}. \tag{B.2}$$

$\Theta(x,t)$ is the temperature in the strip at location x and time t, Θ_{amb} is the ambient temperature. D in Eq. (B.1) is the thermal diffusivity; that is $D = k/\rho c$, where ρ is the mass density and c is the specific heat per unit mass of the material.

The initial condition is given by

$$\Theta^*(x,0) = \Theta_{init} - \Theta_{amb}, \tag{B.3}$$

where Θ_{init} is the uniform initial temperature throughout the strip. The boundary condition on the plane $x = w$ is expressed as

$$\left.\frac{\partial \Theta^*(x,t)}{\partial x}\right|_{x=w} = 0. \tag{B.4}$$

The mixed boundary condition

$$k\frac{\partial\Theta(0,t)}{\partial x} = h\left[\Theta_{amb} - \Theta(0,t)\right]. \tag{B.5}$$

assures continuity of the heat flux on the plane $x = 0$. That is, the heat flux is removed from the surface $x = 0$ by convection to the environment.

Applying the method of separation of variables and making use of the conditions of Eqs. (B.3), (B.4), and (B.5), Eq. (B.1) can be solved straightforwardly (CARSLAW & JAEGER, 1950) to give the following non-dimensional temperature distribution along the strip

$$\frac{\Theta(X^*,Fo) - \Theta_{amb}}{\Theta_{init} - \Theta_{amb}} = 2\sum_{n=1}^{\infty}\left[\frac{\sin(\lambda_n)\cos[\lambda_n(1-X^*)]}{\lambda_n + \frac{1}{2}\sin(2\lambda_n)}\right]\exp(-\lambda_n^2 Fo), \tag{B.6}$$

where X^* is the dimensionless coordinate, x/w, along the crack, and the non-dimensional time, $Fo = Dt/w^2$, is the Fourier number. The boundary conditions generate eigenvalues, λ_n, that are the roots of the following transcendental equation,

$$\lambda_n \tan(\lambda_n) = \beta, \tag{B.7}$$

where β is the Biot number defined as $\beta = hw/k$.

Appendix C

Listing of the Program T-STRESS

```
C
C
C  ****************************************************************
C  *
C  *
C  *     PROGRAM  T-STRESS
C  *
C  *
C  ****************************************************************
C
C
      PROGRAM TSTRESS
C
      INCLUDE 'domain_common'
C
C     Common for abaqus routines
C
      COMMON / NAME / FILE_NAME
C
      DIMENSION IDONE(20000)
C
      DOUBLE PRECISION ARRAY
      DIMENSION ARRAY(513), JRRAY(2,513),LRUNIT(2,1)
      EQUIVALENCE(ARRAY(1),JRRAY(1,1))
C
C     File Handling
C
      CHARACTER*20 FILN
      CHARACTER*25 FILINP
      CHARACTER*25 FILCHK
      CHARACTER*25 FILOUT
      CHARACTER*25 FILGEO
      CHARACTER*25 FILNOD
      CHARACTER*25 FILVAR
      CHARACTER*25 FILBCK
      CHARACTER*25 FILMIS
      CHARACTER*25 FILDOC
      CHARACTER*25 FILP01
      CHARACTER*25 FILP02
      CHARACTER*25 FILP03
      CHARACTER*25 FILP04
      CHARACTER*25 FILP05
      CHARACTER*25 FILP06
      CHARACTER*25 FILP07
      CHARACTER*25 FILP08
      CHARACTER*25 FILP09
      CHARACTER*25 FILP10
      CHARACTER*25 FILP11
      CHARACTER*25 FILP12
      CHARACTER*25 FILP13
      CHARACTER*25 FILP14
      CHARACTER*25 FILP15
      CHARACTER*25 FILP16
      CHARACTER*25 FILP17
      CHARACTER*25 FILP18
      CHARACTER*25 FILP19
      CHARACTER*25 FILE_NAME
      CHARACTER*80 FNAME
      CHARACTER*4  TINP
      CHARACTER*4  TABQ
      CHARACTER*4  TCHK
      CHARACTER*4  TOUT
      CHARACTER*4  TGEO
      CHARACTER*4  TNOD
      CHARACTER*4  TVAR
      CHARACTER*4  TBCK
      CHARACTER*4  TMIS
      CHARACTER*4  TDOC
      CHARACTER*4  TP01
      CHARACTER*4  TP02
      CHARACTER*4  TP03
      CHARACTER*4  TP04
      CHARACTER*4  TP05
      CHARACTER*4  TP06
      CHARACTER*4  TP07
      CHARACTER*4  TP08
      CHARACTER*4  TP09
      CHARACTER*4  TP10
      CHARACTER*4  TP11
      CHARACTER*4  TP12
      CHARACTER*4  TP13
      CHARACTER*4  TP14
      CHARACTER*4  TP15
      CHARACTER*4  TP16
      CHARACTER*4  TP17
      CHARACTER*4  TP18
      CHARACTER*4  TP19
      CHARACTER*1  RESP
C
      DATA TINP /'.inp'/
      DATA TABQ /'.abq'/
      DATA TCHK /'.chk'/
      DATA TOUT /'.out'/
      DATA TNOD /'.nod'/
```

```
      DATA TGEO /'.geo'/
      DATA TBCK /'.bck'/
      DATA TVAR /'.var'/
      DATA TMIS /'.mis'/
      DATA TDOC /'.doc'/
      DATA TP01 /'.p01'/
      DATA TP02 /'.p02'/
      DATA TP03 /'.p03'/
      DATA TP04 /'.p04'/
      DATA TP05 /'.p05'/
      DATA TP06 /'.p06'/
      DATA TP07 /'.p07'/
      DATA TP08 /'.p08'/
      DATA TP09 /'.p09'/
      DATA TP10 /'.p10'/
      DATA TP11 /'.p11'/
      DATA TP12 /'.p12'/
      DATA TP13 /'.p13'/
      DATA TP14 /'.p14'/
      DATA TP15 /'.p15'/
      DATA TP16 /'.p16'/
      DATA TP17 /'.p17'/
      DATA TP18 /'.p18'/
      DATA TP19 /'.p19'/
C
      OPEN(UNIT = 44,FILE = 'INPUTNAME',STATUS = 'UNKNOWN')
      READ(44,2000) INM,FILN
C
C     PRINT *,''
C     PRINT *,'* * *        PROGRAM T-STRESS              * * *'
C     PRINT *,' '
C     PRINT *,' ABAQUS OUTPUT (FILE 8)  MUST BE NAMED  Job_name.abq'
C     PRINT *,' DOMAIN INPUT  FILE      MUST BE NAMED  Job_name.inp'
C     PRINT *,' DOMAIN OUTPUT FILE      WILL BE NAMED  Job_name.out'
C     PRINT *,' DOMAIN CHECK  FILE      WILL BE NAMED  Job_name.chk'
C     PRINT *,' DOMAIN GEOM.  FILE      WILL BE NAMED  Job_name.geo'
C     PRINT *,' DOMAIN PLOT   FILES     WILL BE NAMED  Job_name.p**'
C     PRINT *,' '
C     PRINT *,' COORDINATES MUST BE CENTERED AT CRACK-TIP'
C     PRINT *,''
C
C     PRINT 1000,' Please enter the Job_name (MAX 20 Char.)'
C     READ(*,2000) INM,FILN
C
      FILINP(1:INM) = FILN(1:INM)
      FILINP(INM+1:INM+4) = TINP(1:4)
C
      FILE_NAME(1:INM) = FILN(1:INM)
      FILE_NAME(INM+1:INM+4) = TABQ(1:4)
C
      FILCHK(1:INM) = FILN(1:INM)
      FILCHK(INM+1:INM+4) = TCHK(1:4)
C
      FILOUT(1:INM) = FILN(1:INM)
      FILOUT(INM+1:INM+4) = TOUT(1:4)
C
      FILGEO(1:INM) = FILN(1:INM)
      FILGEO(INM+1:INM+4) = TGEO(1:4)
C
      FILNOD(1:INM) = FILN(1:INM)
      FILNOD(INM+1:INM+4) = TNOD(1:4)
C
      FILVAR(1:INM) = FILN(1:INM)
      FILVAR(INM+1:INM+4) = TVAR(1:4)
C
      FILBCK(1:INM) = FILN(1:INM)
      FILBCK(INM+1:INM+4) = TBCK(1:4)
C
      FILMIS(1:INM) = FILN(1:INM)
      FILMIS(INM+1:INM+4) = TMIS(1:4)
C
      FILDOC(1:INM) = FILN(1:INM)
      FILDOC(INM+1:INM+4) = TDOC(1:4)
C
      FILP01(1:INM) = FILN(1:INM)
      FILP01(INM+1:INM+4) = TP01(1:4)
      FILP02(1:INM) = FILN(1:INM)
      FILP02(INM+1:INM+4) = TP02(1:4)
      FILP03(1:INM) = FILN(1:INM)
      FILP03(INM+1:INM+4) = TP03(1:4)
      FILP04(1:INM) = FILN(1:INM)
      FILP04(INM+1:INM+4) = TP04(1:4)
      FILP05(1:INM) = FILN(1:INM)
      FILP05(INM+1:INM+4) = TP05(1:4)
      FILP06(1:INM) = FILN(1:INM)
      FILP06(INM+1:INM+4) = TP06(1:4)
      FILP07(1:INM) = FILN(1:INM)
      FILP07(INM+1:INM+4) = TP07(1:4)
      FILP08(1:INM) = FILN(1:INM)
      FILP08(INM+1:INM+4) = TP08(1:4)
      FILP09(1:INM) = FILN(1:INM)
      FILP09(INM+1:INM+4) = TP09(1:4)
      FILP10(1:INM) = FILN(1:INM)
      FILP10(INM+1:INM+4) = TP10(1:4)
```

```
      FILP11(1:INM) = FILN(1:INM)
      FILP11(INM+1:INM+4) = TP11(1:4)
      FILP12(1:INM) = FILN(1:INM)
      FILP12(INM+1:INM+4) = TP12(1:4)
      FILP13(1:INM) = FILN(1:INM)
      FILP13(INM+1:INM+4) = TP13(1:4)
      FILP14(1:INM) = FILN(1:INM)
      FILP14(INM+1:INM+4) = TP14(1:4)
      FILP15(1:INM) = FILN(1:INM)
      FILP15(INM+1:INM+4) = TP15(1:4)
      FILP16(1:INM) = FILN(1:INM)
      FILP16(INM+1:INM+4) = TP16(1:4)
      FILP17(1:INM) = FILN(1:INM)
      FILP17(INM+1:INM+4) = TP17(1:4)
      FILP18(1:INM) = FILN(1:INM)
      FILP18(INM+1:INM+4) = TP18(1:4)
      FILP19(1:INM) = FILN(1:INM)
      FILP19(INM+1:INM+4) = TP19(1:4)
C
      OPEN(UNIT = 10,FILE = FILINP,STATUS = 'OLD',ERR=10)
      GO TO 20
 10   PRINT*,''
      PRINT*,''
      PRINT*,'Unable to open file : ',FILINP
      PRINT*,''
      PRINT*,''
      PRINT 1000,' Do you want to try again? [Y]'
      READ (*,2000)INUTIL, RESP
      IF (RESP.EQ.'N'.OR.RESP.EQ.'n') STOP
      PRINT 1000,' Choose another Job_name '
      READ (*,2000) INM,FILN

      FILINP(1:INM) = FILN(1:INM)
      FILINP(INM+1:INM+4) = TINP(1:4)
      FILINP(INM+5:25) = ''
C
      FILE_NAME(1:INM) = FILN(1:INM)
      FILE_NAME(INM+1:INM+4) = TABQ(1:4)
      FILE_NAME(INM+5:25) = ''
C
      FILCHK(1:INM) = FILN(1:INM)
      FILCHK(INM+1:INM+4) = TCHK(1:4)
      FILCHK(INM+5:25) = ''
C
      FILOUT(1:INM) = FILN(1:INM)
      FILOUT(INM+1:INM+4) = TOUT(1:4)
      FILOUT(INM+5:25) = ''
C
      FILGEO(1:INM) = FILN(1:INM)
      FILGEO(INM+1:INM+4) = TGEO(1:4)
      FILGEO(INM+5:25) = ''
C
      FILNOD(1:INM) = FILN(1:INM)
      FILNOD(INM+1:INM+4) = TNOD(1:4)
      FILNOD(INM+5:25) = ''
C
      FILVAR(1:INM) = FILN(1:INM)
      FILVAR(INM+1:INM+4) = TVAR(1:4)
      FILVAR(INM+5:25) = ''
C
      FILBCK(1:INM) = FILN(1:INM)
      FILBCK(INM+1:INM+4) = TBCK(1:4)
      FILBCK(INM+5:25) = ''
C
      FILMIS(1:INM) = FILN(1:INM)
      FILMIS(INM+1:INM+4) = TMIS(1:4)
      FILMIS(INM+5:25) = ''
C
      FILDOC(1:INM) = FILN(1:INM)
      FILDOC(INM+1:INM+4) = TDOC(1:4)
      FILDOC(INM+5:25) = ''
C
      FILP01(1:INM) = FILN(1:INM)
      FILP01(INM+1:INM+4) = TP01(1:4)
      FILP01(INM+5:25) = ''
      FILP02(1:INM) = FILN(1:INM)
      FILP02(INM+1:INM+4) = TP02(1:4)
      FILP02(INM+5:25) = ''
      FILP03(1:INM) = FILN(1:INM)
      FILP03(INM+1:INM+4) = TP03(1:4)
      FILP03(INM+5:25) = ''
      FILP04(1:INM) = FILN(1:INM)
      FILP04(INM+1:INM+4) = TP04(1:4)
      FILP04(INM+5:25) = ''
      FILP05(1:INM) = FILN(1:INM)
      FILP05(INM+1:INM+4) = TP05(1:4)
      FILP05(INM+5:25) = ''
      FILP06(1:INM) = FILN(1:INM)
      FILP06(INM+1:INM+4) = TP06(1:4)
      FILP06(INM+5:25) = ''
      FILP07(1:INM) = FILN(1:INM)
      FILP07(INM+1:INM+4) = TP07(1:4)
      FILP07(INM+5:25) = ''
      FILP08(1:INM) = FILN(1:INM)
```

```
      FILP08(INM+1:INM+4) = TP08(1:4)
      FILP08(INM+5:25) = ''
      FILP09(1:INM) = FILN(1:INM)
      FILP09(INM+1:INM+4) = TP09(1:4)
      FILP09(INM+5:25) = ''
      FILP10(1:INM) = FILN(1:INM)
      FILP10(INM+1:INM+4) = TP10(1:4)
      FILP10(INM+5:25) = ''
      FILP11(1:INM) = FILN(1:INM)
      FILP11(INM+1:INM+4) = TP11(1:4)
      FILP11(INM+5:25) = ''
      FILP12(1:INM) = FILN(1:INM)
      FILP12(INM+1:INM+4) = TP12(1:4)
      FILP12(INM+5:25) = ''
      FILP13(1:INM) = FILN(1:INM)
      FILP13(INM+1:INM+4) = TP13(1:4)
      FILP13(INM+5:25) = ''
      FILP14(1:INM) = FILN(1:INM)
      FILP14(INM+1:INM+4) = TP14(1:4)
      FILP14(INM+5:25) = ''
      FILP15(1:INM) = FILN(1:INM)
      FILP15(INM+1:INM+4) = TP15(1:4)
      FILP15(INM+5:25) = ''
      FILP16(1:INM) = FILN(1:INM)
      FILP16(INM+1:INM+4) = TP16(1:4)
      FILP16(INM+5:25) = ''
      FILP17(1:INM) = FILN(1:INM)
      FILP17(INM+1:INM+4) = TP17(1:4)
      FILP17(INM+5:25) = ''
      FILP18(1:INM) = FILN(1:INM)
      FILP18(INM+1:INM+4) = TP18(1:4)
      FILP18(INM+5:25) = ''
      FILP19(1:INM) = FILN(1:INM)
      FILP19(INM+1:INM+4) = TP19(1:4)
      FILP19(INM+5:25) = ''
C
      OPEN(UNIT = 10,FILE = FILINP,STATUS = 'OLD',ERR=10)
C
 20   CONTINUE
C
      OPEN(UNIT = 13,FILE = FILCHK)
      OPEN(UNIT = 14,FILE = FILOUT)
      OPEN(UNIT = 15,FILE = FILGEO)
      OPEN(UNIT = 35,FILE = FILVAR)
      OPEN(UNIT = 36,FILE = FILNOD)
      OPEN(UNIT = 37,FILE = FILMIS)
      OPEN(UNIT = 38,FILE = FILDOC)
      OPEN(UNIT = 12,FILE = FILBCK)
      OPEN(UNIT = 16,FILE = FILP01)
      OPEN(UNIT = 17,FILE = FILP02)
      OPEN(UNIT = 18,FILE = FILP03)
      OPEN(UNIT = 19,FILE = FILP04)
      OPEN(UNIT = 20,FILE = FILP05)
      OPEN(UNIT = 21,FILE = FILP06)
      OPEN(UNIT = 22,FILE = FILP07)
      OPEN(UNIT = 23,FILE = FILP08)
      OPEN(UNIT = 24,FILE = FILP09)
      OPEN(UNIT = 25,FILE = FILP10)
      OPEN(UNIT = 26,FILE = FILP11)
C     OPEN(UNIT = 27,FILE = FILP12)
C     OPEN(UNIT = 28,FILE = FILP13)
C     OPEN(UNIT = 29,FILE = FILP14)
C     OPEN(UNIT = 30,FILE = FILP15)
C     OPEN(UNIT = 31,FILE = FILP16)
C     OPEN(UNIT = 32,FILE = FILP17)
C     OPEN(UNIT = 33,FILE = FILP18)
C     OPEN(UNIT = 34,FILE = FILP19)
C
C
      FNAME(1:25) = FILE_NAME(1:25)
C
      NORI = 10
      NOWC = 13
      NOWO = 14
      NOWG = 15
      NOWS = 35
      NOWN = 36
      NOWM = 37
      NOWB = 12
      NOWD = 38
      NOWP(1) = 16
      NOWP(2) = 17
      NOWP(3) = 18
      NOWP(4) = 19
      NOWP(5) = 20
      NOWP(6) = 21
      NOWP(7) = 22
      NOWP(8) = 23
      NOWP(9) = 24
      NOWP(10) =25
      NOWP(11) =26
      NOWP(12) =27
      NOWP(13) =28
      NOWP(14) =29
```

```
      NOWP(15) =30
      NOWP(16) =31
      NOWP(17) =32
      NOWP(18) =33
      NOWP(19) =34
C
      KERROR = 0
C
C     Accessing file 8
C
      NRU = 1
      LOUTF = 0
      LRUNIT(1,1) = 8
      LRUNIT(2,1) = 2
      CALL INITPF(FNAME,NRU,LRUNIT,LOUTF)
      JUNIT = 8
      CALL DBRNU(JUNIT)
C
C     Input flags and material data
C
      CALL FLAG
C
      IF(KERROR.GT.0) GO TO 200
C
      DO 121 NDOM=1,NDOMT
C
C     Reset to zero flags and variables
C
      CALL ZERO(NDOM)
C
C     Geometry input. Connectivity matrices
C
      CALL GEOINP(NDOM,IDONE)
C
      IF(KERROR.GT.0) GO TO 200
C
C     Shape function matrices
C
      IF(NDOM.EQ.1) CALL PRESFN
C
      IF(KERROR.GT.0) GO TO 200
C
C     Input the nodeset for the temperature distribution
C
      IF(NDOM.EQ.1) CALL TEMPINP(IDONE)
C
      IF(KERROR.GT.0) GO TO 200
C
C     Evaluating the perturbation field/domain topology
C
      CALL DOMGEO(NDOM)
C
      IF(KERROR.GT.0) GO TO 200
C
      IF(KPSTOP.EQ.1) GO TO 300
C
C     Evaluate the domain integrals for all the steps/increments
C     required by user
C
      DO 100 NOUT = 1,NTOUT
C
C        reset to zero flags and variables
C
         CALL RESETV(NOUT)
C
C        Obtaining the temperature distribution
C
         IF(ITEMP.EQ.1) CALL TEMPDIS(NDOM,NOUT,TIME)
C
         IF(ITDIS.EQ.1) GO TO 100
C
         IF(KERROR.GT.0) GO TO 200
C
C        Input the physical quantities of the procedure
C
         CALL VARINP(NOUT)
C
C        Obtaining the stress distribution along the
C        symmetry-line of the specimen
C
         IF(NDOM.EQ.1) CALL STRDIS(NDOM,NOUT)
C
         IF(KERROR.GT.0) GO TO 200
C
C        Output nodal and IP variables
C
         IF(IPRINT.EQ.1) CALL VAROUT(NOUT)
C
         IF(KERROR.GT.0) GO TO 200
C
C        Evaluating the domain integrals
C
         CALL DOMINT(NDOM,NOUT,TIME)
C
```

```
      IF(KERROR.GT.0) GO TO 200
C
 100  CONTINUE
C
 121  CONTINUE
C
      CALL INTPRIN
C
      IF(KERROR.EQ.0) GO TO 300
C
      STOP
C
 200  CONTINUE
C
      WRITE (NOWC,3000) KERROR
      WRITE (NOWO,3000) KERROR
      WRITE (NOWD,3000) KERROR
C
 300  CONTINUE
C
 1000 FORMAT($,A,' : ')
 2000 FORMAT(Q,A)
 3000 FORMAT(1H1,//,10X,'* * *  ERROR DETECTED. PROGRAM STOP : KERROR',
     &'= ',I5,' * * *')
C
      STOP
      END
C
C
C
C     NOTE : Routines follow in alphabetical order
C
C
```



```
C
C  ***************************************************************
C  *
C  *
C  *      SUBROUTINE ANALYT
C  *
C  *
C  ***************************************************************
C
      SUBROUTINE ANALYT(TIME,ATEMP,AXCO,NDOM,NOUT)
C
C     Obtaines the analytical through-thickness temperature-distribution
C     of the specimen
C
      INCLUDE 'domain_common'
      DIMENSION ATEMP(200), AXCO(200), EIGEN(100)
C
      IF(NDOM.EQ.1.AND.NOUT.EQ.1) CALL EIGENV(EIGEN)
C
      DO 100 I=1,NTOTTE
         N1 = NTEMP(1)
         NNLC = NTEMP(I)
         XD = COORDS(1,NNLC) + DABS(COORDS(1,N1))
         AXCO(NNLC) = XD/WIDTH
         TH = 0.d+00
         DO 200 N=1,100
           EIG = EIGEN(N)
           COSARG = EIG * (1.d+00 - AXCO(NNLC))
           SINARG = 2.d+00 * EIG
           EXPARG = - EIG**2 * FOU(NOUT)
           THNOM = DSIN(EIG) * DCOS(COSARG)
           THDEN = EIG + 0.5d+00 * DSIN(SINARG)
           TH = TH + THNOM/THDEN * DEXP(EXPARG)
 200  CONTINUE
        ATEMP(NNLC) = 2.d+00 * TH
 100  CONTINUE
C
      RETURN
      END
C
C
```

```
C
C  ***********************************************************************
C  *
C  *
C  *      SUBROUTINE CHKSET
C  *
C  *
C  ***********************************************************************
C
C
      SUBROUTINE CHKSET(ISET,NAB,NLC)
C
C     Find the local identity number
C
C     Parameters
C        I/   ISET : Set flag
C                    ISET = 0  : Element set
C                    ISET = 1  : Node input set
C                    ISET = 2  : Node set
C                    ISET = 3  : Node set for stress and
C                           temp. distribution
C                    ISET = 4  : Domain node set
C        I/   NAB  : Abaqus number
C        O/   NLC  : Local number: NLC =
C                                  IELC (ISET = 0)
C                                +/- NNIS (ISET = 1)
C                                  NNLC (ISET = 2)
C                           NLC = 0 if the node/elt doesn't
C                                 belong to the set
C
      INCLUDE 'domain_common'
C
      NLC = 0
C
      GO TO (100,200,300,400) ISET
C
C
C     Element set
C
      DO 50 IE = 1,NTELDO
         IEL = IDOEL(IE)
         IF (IELTOP(1,IEL).NE.NAB)GO TO 50
         NLC = IEL
         GO TO 999
   50 CONTINUE
      GO TO 999
C
C     Node input set
C
  100 CONTINUE
C
      DO 150 NNIS = 1,NTOTIS
      DO 150 ISIDE = 1,IDOUBL
         IF(NCONN(1,ISIDE,NNIS).NE.NAB) GO TO 150
         NLC = NNIS
         IF(ISIDE.EQ.1) NLC = -NNIS
         GO TO 999
  150 CONTINUE
      GO TO 999
C
C
C     Node set
C
  200 CONTINUE
C
      DO 250 IN = 1,NTNOD
         IF(NABAQ(IN).NE.NAB) GO TO 250
         NLC = IN
         GO TO 999
  250 CONTINUE
      GO TO 999
C
C
C     Node set for stress distributions
C
  300 CONTINUE
C
      DO 350 IN = 1,NTOTTE
         NNLC = NTEMP(IN)
         IF(NABAQ(NNLC).NE.NAB) GO TO 350
         NLC = NNLC
         GO TO 999
  350 CONTINUE
      GO TO 999
C
C
C     Domain node set
C
  400 CONTINUE
C
      DO 450 IN = 1,NTNDO
         INL = IDON(IN)
         IF(NABAQ(INL).NE.NAB) GO TO 450
```

```
      NLC = INL
      GO TO 999
 450  CONTINUE
C
 999  CONTINUE
C
      RETURN
      END
C
C
```



```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE CKPERM
C  *
C  *
C  ****************************************************************
C
C
      SUBROUTINE CKPERM(ID,IMOVE,INEW,B,X)
C
C  Check if in the matrix AM rows have been permuted. If yes(IMOVE=1)
C  it permutes the Right Hand Side B according to the order given by
C  vector INEW
C
      INCLUDE 'domain_common'
C
      DIMENSION INEW(ID),B(ID),X(ID)
C
      IF(IMOVE.NE.1) GO TO 500
C
      DO 100 NR = 1,ID
 100    X(NR) = B(INEW(NR))
      DO 200 NR = 1,ID
 200    B(NR) = X(NR)
C
 500  CONTINUE
C
      RETURN
      END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE CONNEC
C  *
C  *
C  ****************************************************************
C
C
      SUBROUTINE CONNEC(IFISNO,IDONE,NDOM)
C
C
C     Evaluates the connectivity matrices NCONN , IELTOP & NABAQ
C     Evaluates the total number of nodes NTNOD and element NTELT
C
C     Parameters
C       I/    IFISNO(20000) :local input-set-number for abaqus nodes
C       O/    IDONE (20000) :local number for abaqus nodes
C
      INCLUDE 'domain_common'
      DOUBLE PRECISION ARRAY
      DIMENSION IFISNO(20000), IDONE(20000)
      DIMENSION ARRAY(513), JRRAY(2,513)
      EQUIVALENCE (ARRAY(1), JRRAY(1,1))
C
C     Rewind file 8
C
      CALL DBFILE(2,ARRAY,JRCD)
C
C
C     Scanning file 8
C
      NTNOD = 0
      NTELT =1
C
      DO 500 K =1,99999
C
        CALL DBFILE(0,ARRAY,JRCD)
        IF(JRCD.NE.0) GO TO 600
C
        LR  = JRRAY(1,1)
        KEY  = JRRAY(1,2)
        IF(KEY.NE.1900) GO TO 500
        IEAB = JRRAY(1,3)
        IELTOP(1,NTELT) = IEAB
C
C       fill IELTOP
C
        DO 200 NN = 1,NNELT
          NNAB = JRRAY(1,NN+4)
C
C         Check if the node has been already numbered in the
C         local list. If not -> put it in the list.
C
          IF(IDONE(NNAB).NE.0) GO TO 50
            NTNOD = NTNOD+1
            IDONE(NNAB) = NTNOD
            NABAQ(NTNOD) = NNAB
C
 50       CONTINUE
C
C         Check if NNAB is a node of the input set
C
          IF(IFISNO(NNAB).EQ.0) GO TO 100
C
C           NNAB is a node of the input set -> fill NCONN
C
            NNIS = IFISNO(NNAB)
            IF(NNIS.LT.0) THEN
              ISIDE = 1
              NNIS = -NNIS
            ELSE
              ISIDE = 2
            ENDIF
            NCONN(2,ISIDE,NNIS) = IDONE(NNAB)
            KP = 4 + NCONN(3,ISIDE,NNIS)*3
            NCONN(3,ISIDE,NNIS) = NCONN(3,ISIDE,NNIS) + 1
C
C           check max number of element to which NNIS belongs
C
            IF(NCONN(3,ISIDE,NNIS).LE.6) GO TO 80
              KERROR = KERROR +1
              WRITE(NOWC,1000) NNAB
              GO TO 700
 80           CONTINUE
C
C           filling NCONN
C
            NCONN(KP,ISIDE,NNIS)   = IEAB
            NCONN(KP+1,ISIDE,NNIS) = NTELT
            NCONN(KP+2,ISIDE,NNIS) = NN
C
100       CONTINUE
```

```
C
        IA = 2+(NN-1)*2
        IELTOP(IA,NTELT)   = NNAB
        IELTOP(IA+1,NTELT) = IDONE(NNAB)
 200    CONTINUE
      NTELT = NTELT +1
C
C   Check max number of elt's (2000) and nodes (8000)
C

    IF(NTELT.LT.2000.AND.NTNOD.LT.8000) GO TO 500
    WRITE(NOWC,2000) NTELT,NTNOD
    GO TO 600
C
 500  CONTINUE

 600  CONTINUE
C
    NTELT = NTELT -1
C
 700  CONTINUE
C
    DO 790 ISIDE = 1,IDOUBL
     WRITE(NOWG,3000)ISIDE
C
     DO 750 NNIS = 1,NTOTIS
      IMAX = NCONN(3,ISIDE,NNIS)*3+3
 750   WRITE(NOWG,4000) NNIS,(NCONN(I,ISIDE,NNIS),I=1,IMAX)
 790  CONTINUE
C
    IF(NDOM.EQ.1) THEN
    WRITE(NOWN,5000)
C
    DO 800 IELC = 1,NTELT
    IMAX = NNELT*2 + 1
 800  WRITE(NOWN,6000) IELC,(IELTOP(I,IELC),I = 1,IMAX)
C
    WRITE(NOWN,7000)
C
    NAPPO = (NTNOD-1)/8+1
    DO 850 NROW = 1,NAPPO
    NF = (NROW-1)*8
 850  WRITE(NOWN,8000) (NF+K,NABAQ(NF+K),K = 1,8)
    END IF
C
    IF(NDOM.EQ.1) THEN
      WRITE(NOWO,9000) NTELT, NTNOD
    END IF
C
 900  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. CONNEC  * * *',//,
    &  10X,'MAX NUMBER OF ELEMENTS CONNECTED TO ONE NODE
EXCEEDED',//,
    &  10X,'MORE THAN 6 ELEMENTS ARE CONNECTED TO NODE ',I4)
 2000 FORMAT(1H1,///,20X,'* * *WARNING OF  SBR. CONNEC  * * *',//,
    &  10X,'MAX NUMBER OF ELEMENTS and/or NODES EXCEEDED',//,
    &  10X,'TOTAL NUMBER OF READ ELEMENTS = ',I4,' (MAX:2000)',//,
    &  10X,'TOTAL NUMBER OF READ NODES = ',I4,' (MAX: 8000)')
 3000 FORMAT(1H1,///,40X,'* * * N C O N N * * *',/,
    &  40X,'* * * I S I D E  =  ',I2,'  * * *',//,
    &  1X,'NNIS NNAB NNLC NBE IAB1 LC1 P1 IAB2 LC2 P2 IAB3 LC3 P3 ',
    &    'IAB4 LC4 P4 IAB5 LC5 P5 IAB6 LC6 P6',/)
 4000 FORMAT(1X,I3,2X,I4,1X,I4,1X,I4,6(1X,I4,1X,I4,1X,I3))
 5000 FORMAT(1H1,///,40X,'* * * I E L T O P * * *',//,
    &  1X,'IELC IEAB NAB1 NLC1 NAB2 NLC2 NAB3 NLC3 NAB4 NLC4 NAB5 ',
    &    'NLC5 NAB6 NLC6 NAB7 NLC7 NAB8 NLC8',/)
 6000 FORMAT(18(1X,I4))
 7000 FORMAT(1H1,///,40X,'* * * N A B A Q * * *',//,
    &  1X,8('NNLC NNAB  '),/)
 8000 FORMAT(8(2(1X,I5),1X))
 9000 FORMAT(1H1,////,20X,'* * * M E S H  I N F O * * *',//,
    &  'Total number of elements: ',I5,//,
    &  'Total number of nodes: ',I5,/)
C
    RETURN
    END
C
C
```

```
C
C  ******************************************************************
C  *
C  *
C  *     FUNCTION  DLISCO
C  *
C  *
C  ******************************************************************
C
C
      FUNCTION DLISCO(L,K,J,I,IR)
C
C     Evaluates the dimensionless isoparametric coordinates of the nodes
C     and of the integration points
C
C     Parameters
C           L : identity flag: 0-> node  1-> int.pt.
C           K : node/int.pt  number
C           J : required coordinate
C                        1 : G
C                        2 : H
C                        3 : R
C           I : Element type
C                     8 : 2D - 8 nodes  isop. element
C           IR: Reduced integration flag
C                     0 : full integration (9 i.p.)
C                     1 : reduced integration (4 i.p.)
C
C     NOTE!!!! ONLY FOR 2nd ORDER - 8 NODES - ISOPARAMETRIC 2D
ELEMENTS
C
=================================================================
C
      INCLUDE 'domain_common'
C
      DLISCO = 0.
      FIV = .577350269189626
      SEV = .774596669241483
C
      IF ( I.EQ.8) GO TO 100
        KERROR = KERROR + 1
        WRITE(NOWC,1000) I
        GO TO 900
  100 CONTINUE
C
      IF(L.EQ.1) GO TO 400
C
C     node coords
C
      GO TO (200,300) J
        KERROR = KERROR + 1
        WRITE(NOWC,2000) J
        GO TO 900
C
C     First coordinate : G
C
  200 CONTINUE
C
      GO TO (210,220,230,240,250,260,270,280) K
C
        KERROR = KERROR + 1
        WRITE(NOWC,3000) K
        GO TO 900
C
  210 CONTINUE
      DLISCO = -1.
      GO TO 900
C
  220 CONTINUE
      DLISCO =  1.
      GO TO 900
C
  230 CONTINUE
      DLISCO =  1.
      GO TO 900
C
  240 CONTINUE
      DLISCO = -1.
      GO TO 900
C
  250 CONTINUE
      DLISCO =  0.
      GO TO 900
C
  260 CONTINUE
      DLISCO =  1.
      GO TO 900
C
  270 CONTINUE
      DLISCO =  0.
      GO TO 900
C
```

```
 280  CONTINUE
      DLISCO = -1.
      GO TO 900
C
C     Second coordinate : H
C
 300  CONTINUE
C
      GO TO (310,320,330,340,350,360,370,380) K
C
        KERROR = KERROR + 1
        WRITE(NOWC,3000) K
        GO TO 900
C
 310  CONTINUE
      DLISCO = -1.
      GO TO 900
C
 320  CONTINUE
      DLISCO = -1.
      GO TO 900
C
 330  CONTINUE
      DLISCO =  1.
      GO TO 900
C
 340  CONTINUE
      DLISCO =  1.
      GO TO 900
C
 350  CONTINUE
      DLISCO = -1.
      GO TO 900
C
 360  CONTINUE
      DLISCO =  0.
      GO TO 900
C
 370  CONTINUE
      DLISCO =  1.
      GO TO 900
C
 380  CONTINUE
      DLISCO =  0.
      GO TO 900
C
 400  CONTINUE
C
C     integration points coords
C
      IF(IR.EQ.1) GO TO 700
C
C     Full integration
C
      GO TO (500,600) J
        KERROR = KERROR + 1
        WRITE(NOWC,2000) J
        GO TO 900
C
C     First coordinate : G
C
 500  CONTINUE
C
      GO TO (510,520,530,540,550,560,570,580,590) K
C
        KERROR = KERROR + 1
        WRITE(NOWC,4000) K
        GO TO 900
C
 510  CONTINUE
      DLISCO = -SEV
      GO TO 900
C
 520  CONTINUE
      DLISCO = 0.
      GO TO 900
C
 530  CONTINUE
      DLISCO = SEV
      GO TO 900
C
 540  CONTINUE
      DLISCO = -SEV
      GO TO 900
C
 550  CONTINUE
      DLISCO = 0.
      GO TO 900
C
 560  CONTINUE
      DLISCO = SEV
      GO TO 900
C
 570  CONTINUE
```

```
      DLISCO = -SEV
      GO TO 900
C
 580  CONTINUE
      DLISCO = 0.
      GO TO 900
C
 590  CONTINUE
      DLISCO = SEV
      GO TO 900
C
C     Second coordinate : H
C
 600  CONTINUE
C
      GO TO (610,620,630,640,650,660,670,680,690) K
C
        KERROR = KERROR + 1
        WRITE(NOWC,4000) K
        GO TO 900
C
 610  CONTINUE
      DLISCO = -SEV
      GO TO 900
C
 620  CONTINUE
      DLISCO = -SEV
      GO TO 900
C
 630  CONTINUE
      DLISCO = -SEV
      GO TO 900
C
 640  CONTINUE
      DLISCO = 0.
      GO TO 900
C
 650  CONTINUE
      DLISCO = 0.
      GO TO 900
C
 660  CONTINUE
      DLISCO = 0.
      GO TO 900
C
 670  CONTINUE
      DLISCO = SEV
      GO TO 900
C
 680  CONTINUE
      DLISCO = SEV
      GO TO 900
C
 690  CONTINUE
      DLISCO = SEV
      GO TO 900
C
 700  CONTINUE
C
C     Reduced integration
C
      GO TO (800,850) J
        KERROR = KERROR + 1
        WRITE(NOWC,2000) J
        GO TO 900
C
C     First coordinate : G
C
 800  CONTINUE
C
      GO TO (810,820,830,840) K
C
        KERROR = KERROR + 1
        WRITE(NOWC,5000) K
        GO TO 900
C
 810  CONTINUE
      DLISCO = -FIV
      GO TO 900
C
 820  CONTINUE
      DLISCO = FIV
      GO TO 900
C
 830  CONTINUE
      DLISCO = -FIV
      GO TO 900
C
 840  CONTINUE
      DLISCO = FIV
      GO TO 900
CC
C     Second coordinate : H
C
```

```
 850  CONTINUE
C
      GO TO (860,870,880,890) K
C
        KERROR = KERROR + 1
        WRITE(NOWC,5000) K
        GO TO 900
C
 860  CONTINUE
      DLISCO = -FIV
      GO TO 900
C
 870  CONTINUE
      DLISCO = -FIV
      GO TO 900
C
 880  CONTINUE
      DLISCO = FIV
      GO TO 900
C
 890  CONTINUE
      DLISCO = FIV
      GO TO 900
C
 900  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.DLISCO  * * *',//,
     & 10X,'ONLY ELEMENT-TYPE 8 IS IMPLEMENTED',//,
     & 10X,'ELEMENT-TYPE = ',I4)
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.DLISCO  * * *',//,
     & 10X,'THE REQUIRED-COORDINATE-CODE MUST BE 1 (G) OR 2(H)',//,
     & 10X,'REQUIRED COORDINATE CODE = ',I4)
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.DLISCO  * * *',//,
     & 10X,'FOR THIS EL-TYPE THE NODE NUMBER MUST BE BETW. 1 AND 8',//,
     & 10X,'NODE NUMBER = ',I4)
 4000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.DLISCO  * * *',//,
     & 10X,'FOR THIS EL-TYPE THE IT.P NUMBER MUST BE BETW. 1 AND 9',//,
     & 10X,'IT.P NUMBER = ',I4)
 5000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.DLISCO  * * *',//,
     & 10X,'FOR THIS EL-TYPE THE IT.P NUMBER MUST BE BETW. 1 AND 4',//,
     & 10X,'IT.P NUMBER = ',I4)
C
      RETURN
      END
C
C
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE DOMGEO
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE DOMGEO(NDOM)
C
C  Evaluates the perturbation field of the domain and the
C  domain topology
C
      INCLUDE 'domain_common'
C
      DIMENSION IDOTOP(9,15)
C
      DO 500 ISIDE=1,IDOUBL
C
        IF(IRING.EQ.1) THEN
          LAYMAX=NTLELT(NDOM)/2
          IF(LAYMAX.EQ.0) THEN
            KERROR = KERROR+1
            WRITE(NOWC,2000) NTLELT(NDOM),NDOM,IRING
          END IF
        ELSE
          LAYMAX=NLAYER(NDOM)
        END IF
C
        IF(IPRINT.EQ.1) WRITE(NOWG,4000)
        NTELDO = 0
        NTNDO = 0
        NTELD = 0
        NTND = 0
C
        DO 400 NNIS = 1,NTOTIS
C
          CALL IDTCAL(NNIS,ISIDE,LAYMAX,IDOTOP)
          IF (KERROR.GT.0) GO TO 900
C
C         Print the pertubation field topology
C
          IF(IPRINT.EQ.1) THEN
          DO 440 LAN=1,LAYMAX
 440        WRITE(NOWG,6000) LAN, NNIS, (IDOTOP(I,LAN),I=1,9)
```

```
      END IF
C
C
C       Determine the corner nodes of the domain
C
      IF(NNIS.EQ.1) THEN
       IFNN1 = NCONN(2,1,1)
       IFNNL = NCONN(2,1,NTOTIS)
       IFE1 = IDOTOP(2,1)
       NELLEF = IDOTOP(2,LAYMAX)
       NCORL = IELTOP(9,NELLEF)
       CALL IDTCAL(NTOTIS,ISIDE,LAYMAX,IDOTOP)
       IF (KERROR.GT.0) GO TO 900
       IFEL = IDOTOP(2,1)
       NELRI = IDOTOP(2,LAYMAX)
       NCORR = IELTOP(7,NELRI)
      ENDIF
C
      CALL PERCAL(NNIS,ISIDE,LAYMAX,IDOTOP,NTELD,NTND)
      IF (KERROR.GT.0) GO TO 900
C
 400   CONTINUE
 500  CONTINUE
    NTELDO = NTELD
    NELEM(NDOM) = NTELDO
    NTNDO = NTND
C
    IF(NTELDO.LE.800) GO TO 310
      KERROR = KERROR + 1
      WRITE(NOWC,7000) NTELDO
 310   CONTINUE
    IF(KERROR.NE.0) GO TO 900
C
    IF(NTNDO.LE.3200) GO TO 320
      KERROR = KERROR + 1
      WRITE(NOWC,8000) NTNDO
 320   CONTINUE
    IF(KERROR.NE.0) GO TO 900
C
C   Calculate the perturbation field gradient
C   of the domain
C
    DO 340 NEL = 1,NTELDO
      IELC = IDOEL(NEL)
      IF(IQGRAD(IELC).NE.1) CALL GRADIQ(IELC)
C       IF (IPFLAG.EQ.1) THEN
C         CALL GRQPYR(IELC,NEL)
C       ELSE IF (IPFLAG.EQ.0) THEN
C         CALL GRQPLA(IELC,NEL)
C       END IF
C
 340  CONTINUE
C
    IF(IPRINT.EQ.1) CALL DOMPRINT
C
    WRITE(NOWG,1000)
    DO 360 NEL = 1,NTELDO
      IELC = IDOEL(NEL)
 360    WRITE(NOWG,1100) NEL, IELC, IELTOP(1,IELC)
C
    WRITE(NOWG,1500)
    DO 370 NN = 1,NTNDO
      NNLC = IDON(NN)
 370    WRITE(NOWG,1100) NN, NNLC, NABAQ(NNLC)
C
 900  CONTINUE
C
 1000 FORMAT(1H1,///,3X,'NELD    IELC    IEAB',/)
 1100 FORMAT(2X,I4,3X,I4,3X,I5)
 1500 FORMAT(1H1,///,3X,'NND    NNLC    NNAB',/)
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. DOMGEO  * * *',//,
   &  10X,'NUMBER OF LINEAR ELEMENTS = ',I4,' IN DOMAIN ',I4,//,
   &  10X,'IRING = ',I4)
 4000 FORMAT(1H1,////,10X,'* * *  P E R T U R B A T I O N  '
   &  'T O P O L O G Y (I D O T O P) * * *',//
   &     ' LAYER  NNIS  ',/)
 6000 FORMAT(2(1X,I4),9(2X,I4,2X))
 7000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. DOMGEO  * * *',//,
   &  10X,'MAX NUMBER OF ELEMENTS IN DOMAIN EXCEEDED ',//,
   &  10X,'TOTAL NUMBER OF ELEMENTS = ',I4,' (MAX: 800)')
 8000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. DOMGEO  * * *',//,
   &  10X,'MAX NUMBER OF NODES IN DOMAIN EXCEEDED ',//,
   &  10X,'TOTAL NUMBER OF NODES = ',I4,' (MAX: 3200)')
C
    RETURN
    END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE DOMINT
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE DOMINT(NDOM,NOUT,TIME)
C
C  Evaluates the domain integrals
C
      INCLUDE 'domain_common'
C
C  Calculate the J-integral
C
      TJINT = 0.d+00
C
      IF(IPRINT.EQ.1) WRITE(NOWB,99)
      DO 100 NEL = 1,NTELDO
        IELC = IDOEL(NEL)
        CALL JINTGR(IELC,AINTG)
        IF(IPRINT.EQ.1)THEN
          WRITE(NOWB,112) AINTG, IELC, IELTOP(1,IELC)
        END IF
        TJINT = TJINT + AINTG
  100 CONTINUE
C
      TJ(NDOM,NOUT) = 2.d+00 * TJINT
C
C  Calculate the interaction integral
C
      IF(NOUT.EQ.1) WRITE(NOWN,6000)
C
      TINTA = 0.d+00
C
      IF(IPRINT.EQ.1) WRITE(NOWB,111)
      DO 200 NEL = 1,NTELDO
        IELC = IDOEL(NEL)
        CALL INTACT(IELC,BINTG,NOUT)
        IF(IPRINT.EQ.1)THEN
          WRITE(NOWB,112) BINTG, IELC, IELTOP(1,IELC)
        END IF
        TINTA = TINTA + BINTG
  200 CONTINUE
C
      VTINTA(NDOM,NOUT) = TINTA
C
C  Calculate and normalize the T-stress
C
      IF(ITEMP.EQ.1)THEN
C       NNLC = NABAQ(NCRACK)
        TCRACK = TEMP(NCRLC)
        DELT = TCRACK - TINIT
        DELTH = TINIT - TAMB
        TSTR1 = 2.d+00 * TINTA/FLINE * YOUNG/(1.d+00-POISS**2)
        TSTR2 = - COTHER * DELT * YOUNG/(1.d+00-POISS)
        WRITE(NOWO,*) NCRLC, NCRACK,
     &              TCRACK, DELT, DELTH, TSTR1, TSTR2
        TSTR(NDOM,NOUT) = TSTR1 + TSTR2
        TNORM(NDOM,NOUT) = TSTR(NDOM,NOUT)/(COTHER*YOUNG*DELTH)
      ELSE
        TPARE = 2.d+00 * TINTA/FLINE
        TSTR(NDOM,NOUT) = YOUNG/(1.d+00-POISS**2) * TPARE
        IF(ITEN.EQ.1) THEN
          SFAR(NOUT) = RFNOD(2)/(WIDTH*THI)
        ELSE
          SFAR(NOUT) = 6.d+00*DABS(RFNOD(1))/(WIDTH*THI**2)
        END IF
        TNORM(NDOM,NOUT) = TSTR(NDOM,NOUT)/SFAR(NOUT)
      END IF
C
C  Calculate and normalize KI
C
      TJSYM = 2.d+00 * TJINT
      ARGJ = (TJSYM*YOUNG)/(1.d+00-POISS**2)
      VKONE(NDOM,NOUT) = DSQRT(ARGJ)
      ONENE = -1.d+00
      PI = DACOS(ONENE)
      ARG = PI * CRACK
      IF(ITEMP.EQ.1) THEN
        DELTJ = TINIT - TAMB
        DENOM = YOUNG * COTHER * DELTJ * DSQRT(ARG)
        VKONOR(NDOM,NOUT) = VKONE(NDOM,NOUT)*(1.d+00 - POISS)/DENOM
      ELSE
        IF(ITEN.EQ.1) THEN
          SFAR(NOUT) = RFNOD(2)/(WIDTH*THI)
        ELSE
          SFAR(NOUT) = 6.d+00*DABS(RFNOD(1))/(WIDTH*THI**2)
        END IF
        DENOM = SFAR(NOUT) * DSQRT(ARG)
      END IF
```

```
C
      LAYMAX = NTELDO/NTLELT(NDOM)
      WRITE(NOWB,990) NDOM, NTELDO, NTLELT(NDOM),LAYMAX,
     &        NKSTEP(NOUT), NKINCR(NOUT),TIME
C
C     Print the J-integral values
C
      WRITE(NOWB,1000)
      WRITE(NOWB,3000) TJSYM,(VARJ(I,NOUT),I=1,NCONJ)
C
C     Print the T-stress
C
      IF(ITEMP.EQ.1) THEN
        WRITE(NOWB,4100) TINTA,TSTR(NDOM,NOUT),TNORM(NDOM,NOUT)
      ELSE
        WRITE(NOWB,4000) TINTA,TSTR(NDOM,NOUT),SFAR(NOUT),
     &              TNORM(NDOM,NOUT)
      END IF
C
      IF(SFAR(NOUT).LE.YSTR) GO TO 300
       WRITE(NOWB,5000) YSTR
  300 CONTINUE
C
  900 CONTINUE
C
   99 FORMAT(1H1,////,20X,'* * * J-INTEGRAL CONTRIBUTION '
     & 'OF ELEMENTS * * *',//,
     & 1X,'Value:      IELC   IEAB',/)
  111 FORMAT(1H1,////,20X,'* * * INTERACTION INTEGRAL CONTRIBUTION '
     & 'OF ELEMENTS * * *',//,
     & 1X,'Value:      IELC   IEAB',/)
  112 FORMAT(1X,E11.4,3X,I4,3X,I5)
  990 FORMAT(1H1,////,20X,'* * * D O M A I N  I N F O * * *',//,
     & 'Domain No. ',I4,//,
     & 'Number of elements in domain: ',I4,1X,'(',I3,1X,'x',I3,')',//,
     & 'Step No. ',I4,//,
     & 'Increment No. ',I4,//,
     & 'Time: ',E10.4,/)
 1000 FORMAT(1H1.////,20X,'* * * J - I N T E G R A L  '
     & 'E S T I M A T E S  * * *',//,
     & 20X,'     C O N T O U R S',//,
     & 1X,' CALCULATED     1        2        3        4'
     & '         5        6',/)
 3000 FORMAT(1X,E11.4,6(1X,E11.4))
 4100 FORMAT(1H1,////,20X,'* * * I N T E R A C T I O N - '
     & 'I N T E G R A L  * * *',//,
     & 1X,' Value: ',E14.7,//,
     & 1X,' T-stress: ',E14.7,' psi',//,
     & 1X,' Tau: ',E14.7)
 4000 FORMAT(1H1,////,20X,'* * * I N T E R A C T I O N - '
     & 'I N T E G R A L  * * *',//,
     & 1X,' Value: ',E14.7,//,
     & 1X,' T-stress: ',E14.7,' psi',//,
     & 1X,' Far-field stress: ',E14.7,' psi',//,
     & 1X,' Normalized T-stress: ',E14.7)
 5000 FORMAT(1H1,////,20X,'* * *  W A R N I N G ! ! ! * * * ',//,
     & 1X,'Far-field stress exceeding yield strength of',E11.4,
     & ' psi',/)
 6000 FORMAT(1H1,////,20X,'* * * I P - '
     & 'C O O R D I N A T E S  * * *',//,
     & ' IP IELC   X(1)       X(2)     THETA',/)
C
      RETURN
      END
C
C
```

```
C
C     ****************************************************************
C     *
C     *
C     *       SUBROUTINE DOMPRINT
C     *
C     *
C     ****************************************************************
C
      SUBROUTINE DOMPRINT
C
C     Prints the perturbation field and gradient fields of the domain
C     and the domain topology
C
      INCLUDE 'domain_common'
C
C      WRITE(NOWG,7000)IFNN1,NABAQ(IFNN1),COORDS(1,IFNN1),
C     &           COORDS(2,IFNN1),
C     &      IFNNL,NABAQ(IFNNL),COORDS(1,IFNNL),COORDS(2,IFNNL),
C     &      NCORL,NABAQ(NCORL),COORDS(1,NCORL),COORDS(2,NCORL),
C     &      NCORR,NABAQ(NCORR),COORDS(1,NCORR),COORDS(2,NCORR)
C
C     Print the nodal perturbation field values
C
      WRITE(NOWG,7100)
      DO 450 NEL = 1,NTELDO
        IELC = IDOEL(NEL)
        DO 460 NN=1,NNELT
          KP = (NN-1)*2 + 3
          NNLC = IELTOP(KP,IELC)
          WRITE(NOWG,8000) IELC, NN, NNLC, NABAQ(NNLC),
     &              RPERT(NN,IELC,1)
  460   CONTINUE
  450   CONTINUE
C
C     Print the perturbation field gradient
C
      DO 350 NEL = 1,NTELDO
        IELC = IDOEL(NEL)
        DO 350 J = 1,9
        IF(J.EQ.1.AND.NEL.EQ.1) WRITE(NOWG,9200)
  350     WRITE(NOWG,9300)IELC,J,QGRAD(1,J,IELC),QGRAD(4,J,IELC)
C 350     WRITE(NOWG,9300) IELC, J, (QGRAD(I,J,IELC),I=1,9)
C
C
C 7000 FORMAT(1H1,///,40X,'* * * R P E R T * * *',//,
C     & ,//,
C     & 1X,'              NN   NNAB  x-coor.  y-coor.',//,
C     & 1X,'1st node of interface: ',3X,I4,3X,I4,3X,F7.3,3X,F7.3,//,
C     & 1X,'Last node of interface:',3X,I4,3X,I4,3X,F7.3,3X,F7.3,//,
C     & 1X,'Left dom. corner node: ',3X,I4,3X,I4,3X,F7.3,3X,F7.3,//,
C     & 1X,'Right dom. corner node:',3X,I4,3X,I4,3X,F7.3,3X,F7.3,/)
 7100 FORMAT(1H1,///,1X,'IELC  NN NNLC NNAB  RPERT',/)
 8000 FORMAT(4(1X,I4),2X,F8.3)
 9200 FORMAT(1H1,///,40X,'* * * Q G R A D * * *',//,
     & 1X,'IELC  IP   Q11    Q12',/)
C 9200 FORMAT(1H1,///,40X,'* * * Q G R A D * * *',//,
C     & 1X,'IELC  IP   Q11    Q22    Q33    Q12    '
C     & 'Q13    Q23    '
C     & 'Q21    Q31    Q32',/)
 9300 FORMAT(2(1X,I4),2(E12.3))
C
      RETURN
      END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE DOMTOP
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE DOMTOP(IELC,NTELD,NTND)
C
C  Evaluates the domain topology
C
C  Parameters
C       I/   IELC : Local elt number
C       O/   NTELD: Total number of elements in the domain
C       O/   NTND : Total number of nodes in the domain
C       O/   IDOEL: Mapping of the local to domain element
C                   numbering
C       O/   IDON : Mapping of the local to domain node
C                   numbering
C
      INCLUDE 'domain_common'
C
      DO 200 NN = 1,NNELT
C
        NLCP = (NN-1)*2+3
        NNLC = IELTOP(NLCP,IELC)
        IF(INCNT(NNLC).EQ.0) THEN
          NTND = NTND + 1
          INCNT(NNLC) = 1
          IDON(NTND) = NNLC
        END IF
C
 200  CONTINUE
C
      IECNT(IELC) = 1
C
      NTELD = NTELD + 1
      IDOEL(NTELD) = IELC
C
      RETURN
      END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE EIGENV
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE EIGENV(EIGEN)
C
C  Evaluates the eigenvalues for the through-thickness tempertaure
C  distribution of the specimen
C
      INCLUDE 'domain_common'
C
      DIMENSION EIGEN(100)
C
C  Tolerance
C
      EPSIL = 1.0d-09
C
      ONENE = -1.d+00
      PI = DACOS(ONENE)
      EIGEN(1) = 1.55d+00
      WRITE(NOWM,1000) BIOT
      DO 10 N=1,100
        IF(N.EQ.1) GO TO 11
        EIGEN(N) = EIGEN(N-1) + PI
 11     CONTINUE
        EIG = EIGEN(N)
C
C     Newton-Raphson to solve the transcendental equation
C     l_n * tan(l_n) = Biot
C
        DO 20 J=1,20
          F = EIG * DTAN(EIG) - BIOT
          DF = DTAN(EIG) + EIG*(1.d+00+DTAN(EIG)**2)
          DEIG = F/DF
          EIG = EIG - DEIG
          IF(DABS(DEIG).LT.EPSIL) GO TO 40
 20     CONTINUE
        PAUSE 'EIGEN exceeding maximum iterations'
 40     WRITE(NOWM,2000) N, EIG
        EIGEN(N) = EIG
 10     CONTINUE
```

```
C
 1000 FORMAT(1H1,///,10X,'* * * E I G E N V A L U E S  * * *',//,
   & 10X,' N   EIGEN(N)    (BIOT = ',E11.4,')',/)
 2000 FORMAT(8X,I5,2X,F11.7)
C
    RETURN
    END
C
```



```
C  ****************************************************************
C  *
C  *
C  *      S U B R O U T I N E   F 8 I P I N
C  *
C  *
C  ****************************************************************
C
    SUBROUTINE F8IPIN(NOUT,KREQ,VARELT)
C
C   Read variable data from Abaqus-file 8 at element integration points
C
C   Parameters
C      I/ NOUT  : Serial number of the required step/increment
C      I/ KREQ  : Abaqus file 8-read-key for the required variable
C      O/ VARELT: Variable values at integration points
C
C              VARELT(I,J,K) = value of the Ith component
C                              at int. pt. J
C                              of element  K
C       the components of the variable are stored as :
C
C                     scalar    vector     tensor
C
C         ROW 1         a        v(1)      t(1,1)
C         ROW 2         /        v(2)      t(2,2)
C         ROW 3         /        v(3)      t(3,3)
C         ROW 4         /        /         t(1,2)
C         ROW 5         /        /         t(1,3)
C         ROW 6         /        /         t(2,3)
C         ROW 7         /        /         t(2,1)
C         ROW 8         /        /         t(3,1)
C         ROW 9         /        /         t(3,2)
C
    INCLUDE 'domain_common'
C
    DOUBLE PRECISION ARRAY
    DIMENSION ARRAY(513),JRRAY(2,513)
    DIMENSION JV(9,2000),VECT(9),VARELT(9,9,2000)
    EQUIVALENCE (ARRAY(1),JRRAY(1,1))
C
    DO 10 K=1,2000
    DO 10 J= 1,9
    VECT(J)=0.d+00
 10 JV(J,K) = 0
C
```

```
C   Rewind file 8
C
    CALL DBFILE(2,ARRAY,JRCD)
C
C   Scanning file 8
C
    DO 330 K = 1,999999
       CALL DBFILE(0,ARRAY,JRCD)
     IF(JRCD.NE.0) GO TO 350
       LR  = JRRAY(1,1)
       KEY  = JRRAY(1,2)
C
C        Finding the right step/increment
C
       IF (KEY.NE.2000) GO TO 330
       NST = JRRAY(1,8)
       NIN = JRRAY(1,9)
       IF(NKSTEP(NOUT).NE.NST.OR.NKINCR(NOUT).NE.NIN) GO TO 330
C
    JF = 1
    DO 300 KK = 1,999999
       CALL DBFILE(0,ARRAY,JRCD)
     IF(JRCD.NE.0) GO TO 350
       LR  = JRRAY(1,1)
       KEY  = JRRAY(1,2)
     IF(KEY.EQ.2001) GO TO 350
       IEAB = JRRAY(1,3)
       IP  = JRRAY(1,4)
       ILOC = JRRAY(1,6)
C
     IF(KEY.NE.1.OR.ILOC.NE.0) GO TO 300
C
C        The subsequent record of file 8 contains
C        values at integration points of elt IEAB
C
        CALL DBFILE(0,ARRAY,JRCD)
        LR = JRRAY(1,1)
        KEY = JRRAY(1,2)
C
     IF(KEY.NE.KREQ) GO TO 300
C
C        Fill VARELT
C
       DO 140 IE = 1,NTELDO
C
        IEL = IDOEL(IE)
        IF(IELTOP(1,IEL).NE.IEAB) GO TO 140
C
        CALL FILLIN(KREQ,LR,ARRAY,VECT)
        CALL CHKSET(0,IEAB,IELC)
        IF (IELC.GT.0) GO TO 150
          WRITE(NOWC,2000) IEAB
          GO TO 300
 150      CONTINUE
        DO 200 I = 1,9
 200      VARELT(I,IP,IELC) = VECT(I)
        JV(IP,IELC) = 1
C
 140    CONTINUE
C
 300 CONTINUE
 330 CONTINUE
 350 CONTINUE
C
C   Check if the required step/incr has been found
C
    IF(JF.EQ.1) GO TO 400
       KERROR = KERROR +1
       WRITE(NOWC,3000) NKINCR(NOUT),NKSTEP(NOUT)
 400 CONTINUE
C
C   Check if all int.pts. have been found
C
    DO 500 K = 1,NTELDO
      KL = IDOEL(K)
    DO 500 J = 1, NINTP
       IF(JV(J,KL).EQ.1) GO TO 500
       KERROR = KERROR +1
       WRITE(NOWC,1000) J,IELTOP(1,KL),KREQ
 500 CONTINUE
C
 600 CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. F8IPIN  * * *',//,
   & 10X,'INT.PT. ',I4,' OF ELT',I4,'  HAS NOT BEEN FOUND',
   & ' FOR KEY = ',I4)
 2000 FORMAT(1H1,///,20X,'* * *WARNING OF SBR. F8IPIN  * * *',//,
   & 10X,'LOCAL NUMBER OF THE ELEMENT',I4,' HAS NOT BEEN FOUND')
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. F8IPIN  * * *',//,
   & 10X,'INCREMENT ',I4,' OF STEP',I4,'  HAS NOT BEEN FOUND')
C
    RETURN
    END
C
```

```
C
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE F8JINT
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE F8JINT(NOUT,KREQ)
C
C  Read J-integral data from Abaqus-file 8
C
C  Parameters
C     I/ NOUT  : Serial number of the required step/increment
C     I/ KREQ  : Abaqus file 8-read-key for the required variable
C     O/ VARJ: Variable values of J-integral
C
C             VARJ(I) = value of the Ith contour
C
C      the components of the variable are stored as :
C
C                  scalar    vector     tensor
C
C        ROW 1        a       v(1)      t(1,1)
C        ROW 2        /       v(2)      t(2,2)
C        ROW 3        /       v(3)      t(3,3)
C        ROW 4        /       /         t(1,2)
C        ROW 5        /       /         t(1,3)
C        ROW 6        /       /         t(2,3)
C        ROW 7        /       /         t(2,1)
C        ROW 8        /       /         t(3,1)
C        ROW 9        /       /         t(3,2)
C
C
      INCLUDE 'domain_common'
C
      DOUBLE PRECISION ARRAY
      DIMENSION ARRAY(513),JRRAY(2,513)
      DIMENSION VECT(9)
      EQUIVALENCE (ARRAY(1),JRRAY(1,1))
C
C   Rewind file 8
C
      CALL DBFILE(2,ARRAY,JRCD)
C
C   Scanning file 8
C
      DO 330 K = 1,999999
         CALL DBFILE(0,ARRAY,JRCD)
       IF(JRCD.NE.0) GO TO 350
         LR  = JRRAY(1,1)
         KEY  = JRRAY(1,2)
C
C        Finding the right step/increment
C
         IF (KEY.NE.2000) GO TO 330
         NST = JRRAY(1,8)
         NIN = JRRAY(1,9)
         IF(NKSTEP(NOUT).NE.NST.OR.NKINCR(NOUT).NE.NIN) GO TO 330
C
      JF = 1
      DO 300 KK = 1,999999
         CALL DBFILE(0,ARRAY,JRCD)
       IF(JRCD.NE.0) GO TO 350
         LR  = JRRAY(1,1)
         KEY  = JRRAY(1,2)
       IF(KEY.EQ.2001) GO TO 350
         NCJ = JRRAY(1,5)
C
      IF(KEY.NE.KREQ) GO TO 300
C
C          Fill VARJ
C
          NCONJ = NCJ
          CALL FILLIN(KREQ,LR,ARRAY,VECT)
          DO 200 I = 1,NCONJ
  200       VARJ(I,NOUT) = VECT(I)
C
  300 CONTINUE
  330 CONTINUE
  350 CONTINUE

C   Check if the required step/incr has been found
C
     IF(JF.EQ.1) GO TO 400
         KERROR = KERROR +1
         WRITE(NOWC,3000) NKINCR(NOUT),NKSTEP(NOUT)
  400 CONTINUE
C
  600 CONTINUE
C
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. F8JINT  * * *',//,
```

```
     & 10X,'INCREMENT ',I4,' OF STEP',I4,'  HAS NOT BEEN FOUND')
C
      RETURN
      END
C
C
C
C
C     ****************************************************************
C     *
C     *
C     *       SUBROUTINE F8NOIN
C     *
C     *
C     ****************************************************************
C
      SUBROUTINE F8NOIN(NOUT,KREQ,VARNOD,TIME,TTIME)
C
C     Read variable data  at nodes from Abaqus-file 8
C
C     Parameters
C        I/ NOUT  : Serial number of the required step/increment
C        I/ KREQ  : Abaqus file 8-read-key for the required variable
C        O/ VARNOD: Variable values at all  nodes
C
C                VARNOD(I,J) = value of the Ith component
C                              at node J
C         the components of the variable are stored as :
C
C                       scalar    vector      tensor
C
C          ROW 1          a        v(1)       t(1,1)
C          ROW 2          /        v(2)       t(2,2)
C          ROW 3          /        v(3)       t(3,3)
C          ROW 4          /        /          t(1,2)
C          ROW 5          /        /          t(1,3)
C          ROW 6          /        /          t(2,3)
C          ROW 7          /        /          t(2,1)
C          ROW 8          /        /          t(3,1)
C          ROW 9          /        /          t(3,2)
C
C
      INCLUDE 'domain_common'
C
      DOUBLE PRECISION ARRAY
      DIMENSION ARRAY(513),JRRAY(2,513)
      DIMENSION JV(8000),VARNOD(9,8000),VECT(9)
      EQUIVALENCE (ARRAY(1),JRRAY(1,1))
C
      DO 10 K=1,8000
   10 JV(K) = 0
C
C     Rewind file 8
```

```
C
      CALL DBFILE(2,ARRAY,JRCD)
C
C     Scanning file 8
C
      DO 330 K = 1,999999
         CALL DBFILE(0,ARRAY,JRCD)
       IF(JRCD.NE.0) GO TO 350
         LR   = JRRAY(1,1)
         KEY  = JRRAY(1,2)
C
C          Finding the right step/increment
C
         IF (KEY.NE.2000) GO TO 330
         TIME = ARRAY(3)
         TTIME = ARRAY(4)
         NST = JRRAY(1,8)
         NIN = JRRAY(1,9)
         IF(NKSTEP(NOUT).NE.NST.OR.NKINCR(NOUT).NE.NIN) GO TO 330
C
      JF = 1
      DO 300 KK = 1,999999
         CALL DBFILE(0,ARRAY,JRCD)
       IF(JRCD.NE.0) GO TO 350
         LR   = JRRAY(1,1)
         KEY  = JRRAY(1,2)
       IF(KEY.EQ.2001) GO TO 350
         NNAB = JRRAY(1,3)
       IF(NNAB.EQ.10000.AND.KEY.NE.104) GO TO 300
C
C          Check if a node or an element variable is required
C
         IF(KREQ.GE.100) GO TO  50
C
C           Element variable is required
C
          ILOC = JRRAY(1,6)
       IF(KEY.NE.1.OR.ILOC.NE.4) GO TO 300
C
C           The subsequent record of file 8 contains nodal averaged
C           values at node NNAB
C
          CALL DBFILE(0,ARRAY,JRCD)
          LR = JRRAY(1,1)
          KEY = JRRAY(1,2)
C
C         start from here if nodal variable is required
C
 50     CONTINUE
C
      IF(KEY.NE.KREQ) GO TO 300
C
C          Fill VARNOD
C
C          Reaction force at node 10000 for mechanical loading
C
         IF(KREQ.EQ.104) THEN
           CALL FILLIN(KREQ,LR,ARRAY,VECT)
           NNLC = 10000
           RFNOD(1) = VECT(1)
           RFNOD(2) = VECT(2)
C
C          Temperature
C
         ELSE IF (KREQ.EQ.201) THEN
           DO 100 NN=1,NTNDO
             NNLC = IDON(NN)
             IF(NABAQ(NNLC).NE.NNAB) GO TO 100
               CALL FILLIN(KREQ,LR,ARRAY,VECT)
               CALL CHKSET(4,NNAB,NNLC)
               JV(NNLC) = 1
               IF (NNLC.GT.0) GO TO 103
                 WRITE(NOWC,1000) NNAB,KREQ
                GO TO 300
103              CONTINUE
               DO 104 I = 1,9
104              VARNOD(I,NNLC) = VECT(I)
               GO TO 102
100          CONTINUE
           DO 101 NN=1,NTOTTE
             NNLC = NTEMP(NN)
             IF(NABAQ(NNLC).NE.NNAB) GO TO 101
               CALL FILLIN(KREQ,LR,ARRAY,VECT)
               CALL CHKSET(3,NNAB,NNLC)
               JV(NNLC) = 1
               IF (NNLC.GT.0) GO TO 106
                 WRITE(NOWC,1000) NNAB,KREQ
                 GO TO 300
106              CONTINUE
               DO 107 I = 1,9
107              VARNOD(I,NNLC) = VECT(I)
101          CONTINUE
102          CONTINUE
C
```

```
C         Stresses at nodes along symmetry line of specimen
C
        ELSE IF (KREQ.EQ.11) THEN
          DO 105 NN = 1,NTOTTE
            NNLC = NTEMP(NN)
            IF(NABAQ(NNLC).NE.NNAB) GO TO 105
            CALL FILLIN(KREQ,LR,ARRAY,VECT)
            CALL CHKSET(3,NNAB,NNLC)
            IF (NNLC.GT.0) GO TO 110
              WRITE(NOWC,1000) NNAB,KREQ
              GO TO 300
 110        CONTINUE
            JV(NNLC) = 1
            DO 111 I=1,9
 111          VARNOD(I,NNLC) = VECT(I)
 105        CONTINUE
C
C         Other domain variables
C
        ELSE
          DO 120 NN=1,NTNDO
            NNL = IDON(NN)
            IF(NABAQ(NNL).NE.NNAB) GO TO 120
            CALL FILLIN(KREQ,LR,ARRAY,VECT)
            CALL CHKSET(4,NNAB,NNLC)
            IF (NNLC.GT.0) GO TO 130
              WRITE(NOWC,1000) NNAB,KREQ
              GO TO 300
 130        CONTINUE
            JV(NNLC) = 1
            DO 131 I = 1,9
 131          VARNOD(I,NNLC) = VECT(I)
 120        CONTINUE
        END IF
C
 300  CONTINUE
 330  CONTINUE
 350  CONTINUE
C
C    Check if the required step/incr has been found
C
    IF(JF.EQ.1) GO TO 400
        KERROR = KERROR +1
        WRITE(NOWC,3000) NKINCR(NOUT),NKSTEP(NOUT)
 400  CONTINUE
C
    IF(KREQ.EQ.104) GO TO 600
C
C    Check if all nodes have been found
C
    IF(KREQ.EQ.201.OR.KREQ.EQ.11) THEN
C
    IF(KREQ.EQ.11) GO TO 460
C
    DO 450 K = 1,NTNDO
        KL = IDON(K)
        IF(JV(KL).EQ.1) GO TO 450
        KERROR = KERROR + 1
        WRITE( NOWC,2000 ) KL,NABAQ(KL),KREQ
 450  CONTINUE
C
 460  CONTINUE
C
    DO 500 K = 1,NTOTTE
        KL = NTEMP(K)
        IF(JV(KL).EQ.1) GO TO 500
        KERROR = KERROR + 1
        WRITE( NOWC,2000 ) KL,NABAQ(KL),KREQ
 500  CONTINUE
C
    ELSE
C
    DO 550 K = 1,NTNDO
        KL = IDON(K)
        IF(JV(KL).EQ.1) GO TO 550
        KERROR = KERROR + 1
        WRITE( NOWC,2000 ) KL,NABAQ(KL),KREQ
 550  CONTINUE
C
    END IF
C
 600  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * WARNING OF SBR. F8NOIN  * * *',//,
   &  10X,'LOCAL NUMBER FOR NODE ',I5,'KEY ',I4,' HAS NOT'
   &  ' BEEN FOUND')
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. F8NOIN  * * *',//,
   &  10X,'NODE NUMBER ',I4,' (NNAB = ',I4,') OF THE NODE SET HAS'
   &  ' NOT BEEN FOUND FOR KEY = ',I5)
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. F8NOIN  * * *',//,
   &  10X,'INCREMENT ',I4,' OF STEP',I4,'  HAS NOT BEEN FOUND')
 4000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. F8NOIN  * * *',//,
   &  10X,'NODE NUMBER ',I4,' (NNAB = ',I4,') OF THE STRESS DIS. NODE'
   &  ' SET HAS NOT BEEN FOUND FOR KEY = ',I5)
```

```
C
      RETURN
      END
C
C
```

```
C
C  ***************************************************************
C  *
C  *
C  *       SUBROUTINE FILLIN
C  *
C  *
C  ***************************************************************
C
      SUBROUTINE FILLIN(KEY,LR,ARRAY,VECT)
C
C  Fills VECT with the values given by ARRAY according to rules
C  defined by KEY
C
C  Parameters
C     I/    KEY  : Abaqus  file-8-reading key
C     I/    LR   : Record length of ARRAY
C     I/    ARRAY: Input record read from file 8
C     O     VECT : internal variable input array
C
      INCLUDE 'domain_common'
      DOUBLE PRECISION ARRAY
      DIMENSION ARRAY(513),VECT(9)
C
      IF(KEY.GE.100) GO TO 400
C
C  Element variable
C
      IF(KEY.EQ.2.OR.KEY.EQ.14) GO TO 100
      IF(KEY.EQ.11) GO TO 200
      IF(KEY.GE.21.AND.KEY.LE.25) GO TO 300
C
      KERROR = KERROR + 1
      WRITE (NOWC,1000) KEY
      GO TO 700
C
C  1 component-variable
C
 100  CONTINUE
C
      VECT(1) = ARRAY(3)
      GO TO 700
C
C  Stress tensor
C
 200  CONTINUE
```

```
C
      VECT(1) = ARRAY(3)
      VECT(2) = ARRAY(4)
      VECT(3) = ARRAY(5)
      VECT(4) = ARRAY(6)
      VECT(7) = ARRAY(6)
      IF(NDIM.LT.3) GO TO 700
      VECT(5) = ARRAY(7)
      VECT(8) = ARRAY(7)
      VECT(6) = ARRAY(8)
      VECT(9) = ARRAY(8)
      GO TO 700
C
C     Strain tensor
C
  300 CONTINUE
C
      VECT(1) = ARRAY(3)
      VECT(2) = ARRAY(4)
      VECT(3) = ARRAY(5)
      VECT(4) = ARRAY(6)*0.5
      VECT(7) = ARRAY(6)*0.5
      IF(NDIM.LT.3) GO TO 700
      VECT(5) = ARRAY(7)*0.5
      VECT(8) = ARRAY(7)*0.5
      VECT(6) = ARRAY(8)*0.5
      VECT(9) = ARRAY(8)*0.5
      GO TO 700
C
C     Node variable
C
  400 CONTINUE
C
      IF(KEY.EQ.1991) GO TO 110
      IF(KEY.EQ.104.OR.KEY.EQ.107) GO TO 130
C
      DO 500 J = 4,LR
         IC = J-3
         IF(IC.LT.9) GO TO 500
           KERROR = KERROR + 1
           WRITE(NOWC,2000)
           GO TO 700
  500 VECT(IC) = ARRAY(J)
      GO TO 700
C
C     J-integral
C
  110 CONTINUE
C
      DO 120 I=1,NCONJ
        VECT(I) = ARRAY(I+5)
  120 CONTINUE
      GO TO 700
C
C     Reaction-force
C
  130 CONTINUE
C
      VECT(1) = ARRAY(4)
      VECT(2) = ARRAY(5)
C
  700 CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. FILLIN  * * *',//,
     & 10X,'ONLY KEY = 2,11,14,21,22,23,24,25 OR KEY GT. 100',
     & ' ARE IMPLEMENTED OPTIONS',//,10X,'KEY = ',I4)
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. FILLIN  * * *',//,
     & 10X,'FOR NODE VARIABLES ONLY 9 COMPONENTS CAN BE READ ')
C
      RETURN
      END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *       SUBROUTINE FINDIE
C   *
C   *
C   ****************************************************************
C
    SUBROUTINE FINDIE(NNLC1,NNLC2,NNLC3,NBEL,IELOC,NSID)
C
C   Find the elements to which nodes nnlc belong and on what side
C
C   Parameters
C       I/   NNLC1 NNLC2 NNLC3 : local numbers for 3 nodes
C       O/    NBEL   : number of elements to which all the three
C                      nodes belong (max 2)
C       O/    IELOC(2): local numb. for the elts to which nnlc belong
C       O/    NSID(2) : side of IELC to which nnlc belong (must be
C                      consec. nodes on a side - otherwise nsid=0)
C                      it is NSID = +/- 1 <-> +/- 4
C
    INCLUDE 'domain_common'
    DIMENSION NSLC(3),IELOC(2),NSID(2),NP(3)
C
    NBEL = 0
    DO 10 I = 1,2
    IELOC(I) = 0
 10  NSID(I) = 0
    NSLC(1) =NNLC1
    NSLC(2) =NNLC2
    NSLC(3) =NNLC3
C
    DO 200 IELC = 1, NTELT
C
      NTN = 0
      DO 50 N = 1,3
 50    NP(N) = 0
      DO 100 IP = 1,NNELT
      DO 100 NN = 1,3
C
        KP = (IP-1) *2 +3
        IF(IELTOP(KP,IELC).EQ.NSLC(NN)) THEN
          NP(NN) = IP
          NTN = NTN +1
        ENDIF

 100    CONTINUE
C
      IF (NTN.EQ.3) THEN
        NBEL = NBEL+1
        IELOC(NBEL) =IELC
        NP1 = NP(1)
        NP2 = NP(2)
        NP3 = NP(3)
        CALL SIDNOD(1,NP1,NP2,NP3,NSIDE)
        NSID(NBEL) = NSIDE
      ENDIF
 200  CONTINUE
C
    RETURN
    END
C
C
```

```
C
C     ****************************************************************
C     *
C     *
C     *       SUBROUTINE FINDPN
C     *
C     *
C     ****************************************************************
C
C
      SUBROUTINE FINDPN(NSIDE,IPONE,IPHALF,IPZERO)
C
C     Find the nodes IPHALF,IPZERO to be perturbed by the q field
C
C     Parameters
C          I/    NSIDE : side (-1/-4 +1/+4) to which IPONE belongs
C          I/    IPONE : node with perturbation = 1
C          O/    IPHALF: node with perturbation = 1/2
C          O/    IPZERO: node with perturbation = 0
C
C     NOTE!!!! ONLY FOR 2nd ORDER - 8 NODES - ISOPARAMETRIC 2D
ELEMENTS
C
================================================================
C
      INCLUDE 'domain_common'
C
      GO TO (100,200,300,400,500,600,700,800)IPONE
C
         KERROR = KERROR + 1
         WRITE(NOWC,1000) IPONE
         GO TO 900
C
  100 CONTINUE
      IF(ABS(NSIDE).EQ.1) GO TO 130
      IF(ABS(NSIDE).EQ.4) GO TO 160
C
         KERROR = KERROR + 1
         WRITE(NOWC,2000) IPONE,NSIDE
         GO TO 900
C
  130 CONTINUE
      IPHALF = 8
      IPZERO = 4
      GO TO 900
C
  160 CONTINUE
      IPHALF = 5
      IPZERO = 2
      GO TO 900
C
  200 CONTINUE
      IF(ABS(NSIDE).EQ.2) GO TO 230
      IF(ABS(NSIDE).EQ.1) GO TO 260
C
         KERROR = KERROR + 1
         WRITE(NOWC,2000) IPONE,NSIDE
         GO TO 900
C
  230 CONTINUE
      IPHALF = 5
      IPZERO = 1
      GO TO 900
C
  260 CONTINUE
      IPHALF = 6
      IPZERO = 3
      GO TO 900
C
  300 CONTINUE
      IF(ABS(NSIDE).EQ.3) GO TO 330
      IF(ABS(NSIDE).EQ.2) GO TO 360
C
         KERROR = KERROR + 1
         WRITE(NOWC,2000) IPONE,NSIDE
         GO TO 900
C
  330 CONTINUE
      IPHALF = 6
      IPZERO = 2
      GO TO 900
C
  360 CONTINUE
      IPHALF = 7
      IPZERO - 4
      GO TO 900
C
  400 CONTINUE
      IF(ABS(NSIDE).EQ.4) GO TO 430
      IF(ABS(NSIDE).EQ.3) GO TO 460
C
         KERROR = KERROR + 1
         WRITE(NOWC,2000) IPONE,NSIDE
```

```
      GO TO 900
C
 430  CONTINUE
      IPHALF = 7
      IPZERO = 3
      GO TO 900
C
 460  CONTINUE
      IPHALF = 8
      IPZERO = 1
      GO TO 900
C
 500  CONTINUE
      IF(ABS(NSIDE).EQ.1) GO TO 550
C
       KERROR = KERROR + 1
       WRITE(NOWC,2000) IPONE,NSIDE
       GO TO 900
C
 550  CONTINUE
      IPHALF = 0
      IPZERO = 7
      GO TO 900
C
 600  CONTINUE
      IF(ABS(NSIDE).EQ.2) GO TO 650
C
       KERROR = KERROR + 1
       WRITE(NOWC,2000) IPONE,NSIDE
       GO TO 900
C
 650  CONTINUE
      IPHALF = 0
      IPZERO = 8
      GO TO 900
C
 700  CONTINUE
      IF(ABS(NSIDE).EQ.3) GO TO 750
C
       KERROR = KERROR + 1
       WRITE(NOWC,2000) IPONE,NSIDE
       GO TO 900
C
 750  CONTINUE
      IPHALF = 0
      IPZERO = 5
      GO TO 900
C
 800  CONTINUE
      IF(ABS(NSIDE).EQ.4) GO TO 850
C
       KERROR = KERROR + 1
       WRITE(NOWC,2000) IPONE,NSIDE
       GO TO 900
C
 850  CONTINUE
      IPHALF = 0
      IPZERO = 6
C
 900  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. FINDPN  * * *',//,
     &  10X,'THE NODE NUMBER MUST BE BETW.1 AND 8',//,
     &  10X,'NODE NUMBER = ',I4)
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. FINDPN  * * *',//,
     &  10X,' NODE ',I4,' DOES NOT BELONG TO SIDE ',I4)
C
      RETURN
      END
C
C
```

```
C   ****************************************************************
C   *
C   *
C   *      S U B R O U T I N E   F L A G
C   *
C   *
C   ****************************************************************
C
C
      SUBROUTINE FLAG
C
C     Input flags and material data
C
      INCLUDE 'domain_common'
C
      READ(NORI,600) NSETT
C
      DO 100 N=1,NSETT
  100   READ(NORI,700) (NTDIS(N,I),I=1,16)

      READ(NORI,800) ITDIS, ITEMP, NCRACK, ISI
C
      READ(NORI,900) NDOMT, IPFLAG, IPRINT, IRING, ITEN
C
      IF(NDOMT.LE.40) GO TO 110
      KERROR = KERROR +1
      WRITE(NOWC,2100)NDOMT
  110 CONTINUE
C
      READ(NORI,910) ITYPE,IRDINT,KPSTOP,IDOUBL,FLINE
C
      READ(NORI,919) CRACK
      READ(NORI,920) POISS
      READ(NORI,921) YOUNG
      READ(NORI,922) WIDTH
      READ(NORI,923) THI
      READ(NORI,924) YSTR
      READ(NORI,925) COTHER
C
      IF(ISI.EQ.1) THEN
        WRITE(NOWO,930) POISS,YOUNG,WIDTH,THI,CRACK
      ELSE
        WRITE(NOWO,931) POISS,YOUNG,WIDTH,THI,CRACK
      END IF
C
      READ(NORI,940) RHO
      READ(NORI,941) CSPEC
      READ(NORI,942) CONDUC
      READ(NORI,943) TINIT
      READ(NORI,944) TAMB
      READ(NORI,945) HFILM
C
      DIFFU = CONDUC/(RHO * CSPEC)
      BIOT = (HFILM*WIDTH)/CONDUC
C
      IF(ITEMP.EQ.1) THEN
        IF(ISI.EQ.1) THEN
          WRITE(NOWO,950) RHO,CSPEC,CONDUC,DIFFU,
     &       COTHER,TINIT,TAMB,HFILM,BIOT
        ELSE
          WRITE(NOWO,951) RHO,CSPEC,CONDUC,DIFFU,
     &       COTHER,TINIT,TAMB,HFILM,BIOT
        END IF
      END IF
C
      IF(ITYPE.EQ.8) GO TO 200
      KERROR = KERROR +1
      WRITE(NOWC,2000)ITYPE
  200 CONTINUE
C
      READ(NORI,3000) NTOUT
C
      IF(NTOUT.LE.40) GO TO 210
      KERROR = KERROR +1
      WRITE(NOWC,2200)NTOUT
  210 CONTINUE
C
      DO 350 NOUT = 1,NTOUT
  350   READ(NORI,5000)NKSTEP(NOUT),NKINCR(NOUT)
C
  600 FORMAT(I5)
  700 FORMAT(16(I5))
  800 FORMAT(4(I5))
  900 FORMAT(5(I5))
  910 FORMAT(4(I5),F5.2)
  919 FORMAT(F7.4)
  920 FORMAT(F5.4)
  921 FORMAT(E11.4)
  922 FORMAT(F7.4)
  923 FORMAT(F7.4)
  924 FORMAT(E11.4)
  925 FORMAT(E11.4)
  930 FORMAT(1H1,////,20X,'* * * M A T E R I A L  D A T A * * *',//,
```

```
     & 'Poissons ratio: ',F5.4,//,
     & 'Youngs modulus: ',E11.4,1X,'Pa',//,
     & 'Specimen width: ',F7.4,1X,'m',//,
     & 'Specimen thickness: ',F7.4,1X,'m',//,
     & 'Crack length     : ',F7.4,1X,'m',/)
  931 FORMAT(1H1,////,20X,'* * * M A T E R I A L  D A T A * * *',//,
     & 'Poissons ratio: ',F5.4,//,
     & 'Youngs modulus: ',E11.4,1X,'psi',//,
     & 'Specimen width: ',F7.4,1X,'in.',//,
     & 'Specimen thickness: ',F7.4,1X,'in.',//,
     & 'Crack length     : ',F7.4,1X,'in.',/)
  940 FORMAT(F10.5)
  941 FORMAT(F10.5)
  942 FORMAT(E11.4)
  943 FORMAT(F9.3)
  944 FORMAT(F9.3)
  945 FORMAT(E11.4)
  950 FORMAT(1H1,////,20X,'* * * H E A T  T R A N S F E R '
     & 'D A T A * * *',//,
     & 'Density: ',F10.5,1X,'kg/m*3'//,
     & 'Specific heat: ',F10.5,1X,'J/kg K',//,
     & 'Thermal conductivity: ',E11.4,1X,'W/m K',//,
     & 'Diffusivity: ',E11.4,1X,'m*2/sec',//,
     & 'Coeff. of thermal expansion: ',E11.4,1X,'',//,
     & 'Initial Temperature: ',F7.3,1X,'K',//,
     & 'Ambient Temperature: ',F7.3,1X,'K',//,
     & 'Film coefficient: ',E11.4,1X,'W/m*2 K',//,
     & 'Biot number: ',F5.2,/)
  951 FORMAT(1H1,////,20X,'* * * H E A T  T R A N S F E R '
     & 'D A T A * * *',//,
     & 'Density: ',F10.5,1X,'lb/in*3'//,
     & 'Specific heat: ',F10.5,1X,'BTU/lb F',//,
     & 'Thermal conductivity: ',E11.4,1X,'BTU/in sec F',//,
     & 'Diffusivity: ',E11.4,1X,'in*2/sec',//,
     & 'Coeff. of thermal expansion: ',E11.4,1X,'',//,
     & 'Initial Temperature: ',F7.3,1X,'F',//,
     & 'Ambient Temperature: ',F7.3,1X,'F',//,
     & 'Film coefficient: ',E11.4,1X,'BTU/in*2 sec F',//,
     & 'Biot number: ',F5.2,/)
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR.FLAG  * * *',//,
     & 10X,'ONLY ELEMENT-TYPE 8 IS IMPLEMENTED',//,
     & 10X,'ELEMENT-TYPE = ',I4)
 2100 FORMAT(1H1,///,20X,'* * * ERROR IN SBR.FLAG  * * *',//,
     & 10X,'TOTAL NUMER OF DOMAINS EXCEEDED 40',//,
     & 10X,'NUMBER OF DOMAINS = ',I4)
 2200 FORMAT(1H1,///,20X,'* * * ERROR IN SBR.FLAG  * * *',//,
     & 10X,'TOTAL NUMBER OF INCREMENTS EXCEEDED 40',//,
     & 10X,'NUMBER OF INCREMENTS = ',I4)
 3000 FORMAT(I5)
 5000 FORMAT(2(I5))
C
      RETURN
      END
C
C
```

```
C   ****************************************************************
C   *
C   *
C   *      SUBROUTINE GEOINP
C   *
C   *
C   ****************************************************************
C
    SUBROUTINE GEOINP(NDOM,IDONE)
C
C   Manages the geometric input
C
    INCLUDE 'domain_common'
C
    DIMENSION IFISNO(20000),IDONE(20000)
C
    DO 499 N=1,20000
      IFISNO(N)=0
      IDONE(N)=0
 499  CONTINUE
C
    NNELT = 8
    NNLELT = 3
    NDIM = 2
    NINTP = 9
    IF (IRDINT.EQ.1) NINTP = 4
C
    READ(NORI,4000) NLAYER(NDOM)
C
C   Check the maximum number of contour and layers
C
    DO 200 I = 1,NDOMT
        IF(NLAYER(I).LE.15) GO TO 200
        KERROR = KERROR +1
        WRITE(NOWC,2000) I,NLAYER(I)
 200  CONTINUE
C
C   Interface definition
C
    CALL ITFDEF(IFISNO)
C
    IF (KERROR.GT.0) GO TO 300
C
C   Connectivity matrices
C
    CALL CONNEC(IFISNO,IDONE,NDOM)
C
    IF (KERROR.GT.0) GO TO 300
C
    CALL ITFCON(NDOM)
C
    IF (KERROR.GT.0) GO TO 300
C
C   Node cartesian components
C
    IF(NDOM.EQ.1) CALL NCARCO(IDONE)
C
 300  CONTINUE
C
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. GEOINP  * * *',//,
     &  10X,'MAXIMUM NMB. OF LAY. IN DOMAIN EXCEEDED (MAX = 15)',//,
     &  10X,'DOMAIN',I4,' REQUIRED NUMBER OF LAYERS = ',I4)
 4000 FORMAT(I5)
C
    RETURN
    END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE GRADIE
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE GRADIE(IELC,IDIM,F,OU)
C
C  Evaluates the gradient of a scalar field or of a vector field F at
C  the integration points of the element IELC
C
C  Parameters
C     I/  IDIM : Dimension flag of the input field
C                                0 : scalar field
C                                1 : vector field
C     I/  F    : Input field at all nodes of the elt
C     O/  OU   : Output gradient field at the int.points
C           OU(I,J) =  comp.I at int.pt. J
C
      INCLUDE 'domain_common'
C
      DIMENSION  F(3,8),OU(9,9),GR(3,3)
      DIMENSION AM(3,3),AU(3,3),AL(3,3),B(3),X(3), INEW(3)
C
C  Local array :
C     AM : coeff. matrix -> AM(I,J) = [d N(k)/d c(I)] * X(k)(J)
C
C     AL,AU : Triang. fact. matr. AM = AL*AU
C
C     B : RHS -> Bj(I) = [d N(k)/d c(I)] * Fj(k)
C
C     X : Unknown vect -> Xj(I) = d Fj/d x(I)
C
C     GR : Gradient at the int. point  ->
C                      GR(J,I) = d Fj/ d x(I)
C                         C
C              N(k)  : Kth shape function
C              c(I)  : Ith isop. coord.
C              X(k)(J): Jth cart coord of the Kth node
C              x(I)  : Ith cartesian coordinate
C              Fj(k) :Jth comp. of input field at node k
C
C
      DO 10 N = 1,9
      DO 10 I = 1,9
 10   OU(I,N) = 0.d+00
C
      DO 20 I = 1,3
      DO 20 J = 1,3
 20   GR(I,J) = 0.d+00
C
      NCOMP = NDIM
      IF(IDIM.EQ.0) NCOMP = 1
C
C
      DO 400 IP = 1,NINTP
C
C       Evaluate the coefficient matrix AM
C
        CALL GRAMAT(IELC,IP,AM)
C
C       LU factorization
C
        KER = KERROR
        CALL LUFACT(KER,3,IMOVE,INEW,AM,AU,AL)
        KERROR = KER
        IF(KERROR.GT.0) GO TO 500
C
C       Solving the system for all the components of F
C
        DO 100 ID = 1,NCOMP
C
C         Evaluating the RHS
C
          CALL GRARHS (IELC,IP,F,ID,B)
C
C         solving the system
C
          CALL SOLVER (3,IMOVE,INEW,AU,AL,B,X)
C
C       Storing the gradient
C
        DO 100 J = 1,NDIM
C
 100      GR(ID,J) = X(J)
C
C       Storing the result in the output array
C
        IF (IDIM.EQ.1) GO TO 300
C
```

```
C      F is a scalar field -> OU is a vector with NDIM components
C
       OU(1,IP) = GR(1,1)
       OU(2,IP) = GR(1,2)
       OU(3,IP) = GR(1,3)
C
      GO TO 400
C
C      F is a vector field -> OU is a tensor with NDIM*NDIM comp's
C
 300    CONTINUE
C
       OU(1,IP) = GR(1,1)
       OU(2,IP) = GR(2,2)
       OU(3,IP) = GR(3,3)
       OU(4,IP) = GR(1,2)
       OU(5,IP) = GR(1,3)
       OU(6,IP) = GR(2,3)
       OU(7,IP) = GR(2,1)
       OU(8,IP) = GR(3,1)
       OU(9,IP) = GR(3,2)
C
 400  CONTINUE
 500  CONTINUE
C
      RETURN
      END
C
C
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE GRADIQ
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE GRADIQ(IELC)
C
C  Evaluates the gradient of the perturbation  field Q at
C  the integration points of the element IELC
C
C  Parameters
C    I/   IELC : Local elt number
C    O/   QGRAD: Gradient of perturbation field at the integration points
C
C         QGRAD(1,J,IELC) :  dQ/dX1 at integration point J
C         QGRAD(2,J,IELC) :  dQ/dX2 at integration point J
C         QGRAD(3,J,IELC) :  dQ/dX3 at integration point J
C
C
      INCLUDE 'domain_common'
C
      DIMENSION  F(3,8),OU(9,9)

      DO 10 N = 1,3
      DO 10 I = 1,8
 10   F(N,I) = 0.d+00
C
      DO 200 NN = 1,NNELT
C
        F(1,NN) = RPERT(NN,IELC,1)
        F(2,NN) = 0.d+00
        F(3,NN) = 0.d+00
C
 200  CONTINUE
C
      CALL GRADIE(IELC,1,F,OU)
      IF (KERROR.NE.0) GO TO 400
C
      DO 300 J = 1,9
      DO 300 I = 1,9
 300  QGRAD(I,J,IELC) = OU(I,J)
C
```

```
      IQGRAD(IELC) = 1
C
 400  CONTINUE
C
      RETURN
      END
C
C
```



```
C
C     ***************************************************************
C     *
C     *
C     *       SUBROUTINE GRADIU
C     *
C     *
C     ***************************************************************
C
      SUBROUTINE GRADIU(IELC)
C
C     Evaluates the gradient of the displacement field U at
C     the integration points of the element IELC
C
C     Parameters
C          I/   IELC : Local elt number
C
      INCLUDE 'domain_common'
C
      DIMENSION  F(3,8),OU(9,9)
C
      DO 10 N = 1,3
      DO 10 I = 1,8
 10   F(N,I) = 0.
C
      DO 200 NN = 1,NNELT
C
        NLCP = (NN-1)*2+3
        NNLC = IELTOP(NLCP,IELC)
        DO 100 ND = 1,NDIM
 100      F(ND,NN) = UNODE(ND,NNLC)
C
 200  CONTINUE
C
      CALL GRADIE(IELC,1,F,OU)
      IF (KERROR.NE.0) GO TO 400
C
      DO 300 I = 1,9
      DO 300 J = 1,9
 300  UGRAD(I,J,IELC) = OU(I,J)
C
      IUGRAD(IELC) = 1
C
 400  CONTINUE
C
      RETURN
```

```
      END
C
C  *****************************************************************
C  *
C  *
C  *       SUBROUTINE GRAMAT
C  *
C  *
C  *****************************************************************
C
      SUBROUTINE GRAMAT(IELC,IP,AM)
C
C  Evaluate the coefficient matrix AM for integration point IP
C  of element IELC
C
C  Paramweters
C     I/    IELC : local elt id number
C     I/    IP   : position of the integr.point in the elt
C     O/    AM   : coeff. matrix ->
C                      AM(I,J) = [d N(k)/d c(I)] * X(k)(J)
C
C             N(k)  : Kth shape function
C             c(I)  : Ith isop. coord.
C             X(k)(J): Jth cart coord of the Kth integr.point
C
      INCLUDE 'domain_common'
      DIMENSION AM(3,3)
C
      DO 20 NR = 1,3
      DO 10 NC = 1,3
 10   AM(NR,NC) = 0.d+00
 20   AM(NR,NR) = 1.d+00
C
      DO 200 NR = 1,NDIM
      DO 200 NC = 1,NDIM
         SUM = 0.
         DO 100 K = 1, NNELT
            INLC = 3 +(K-1)*2
            NNLC = IELTOP(INLC,IELC)
 100        SUM = SUM+ SFDITP(IP,K,NR) * COORDS(NC,NNLC)
 200  AM(NR,NC) = SUM
C
      RETURN
      END
C
C
```

```
C
C  *****************************************************************
C  *
C  *
C  *       SUBROUTINE GRARHS
C  *
C  *
C  *****************************************************************
C
C
      SUBROUTINE GRARHS(IELC,IP,F,ID,B)
C
C  Evaluates the Right Hand Side of the system to be solved in order
C  to find the gradient of the IDth component of the field F
C
C  Parameters
C     I/    IELC : local elt id number
C     I/    IP   : position of the int. point in the elt
C     I/    F    : input field at each node
C     I/    ID   : component of F to be considered
C     O/    B    : RHS -> B(I) = [d N(k)/d c(I)] * F(ID)(k)
C
C             N(k)  : Kth shape function
C             c(I)  : Ith isop. coord.
C             F(ID)(k):IDth comp. of input field at int. point k
C
      INCLUDE 'domain_common'
      DIMENSION F(3,8),B(3)
C
      DO 10 I = 1,3
 10   B(I) = 0.d+00
C
      DO 200 J = 1,NDIM
         SUM = 0.
         DO 100 K = 1,NNELT
 100        SUM = SUM + SFDITP(IP,K,J) * F(ID,K)
 200  B(J) = SUM
C
      RETURN
      END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *      SUBROUTINE IDTCAL
C   *
C   *
C   ****************************************************************
C
      SUBROUTINE IDTCAL(NNIS,ISIDE,LAYMAX,IDOTOP)
C
C     Evaluates the topology matrix IDOTOP for the domain integral
C
C     Parameters
C        I/ LAYMAX: farthest layer from the interface for which
C                   IDOTOP must be evaluated
C        I/ NNIS  : interface number for which IDOTOP must be eval.
C        I/ ISIDE : side of the intf. for which IDOTOP must be eval.
C        O/ IDOTOP: Output topology matrix
C                   IDOTOP(I,J) =  topology of layer J
C                   the component of each row are:
C     I =  1   2   3    4    5   6   7   8    9
C
NELLAY,IELC1,IPONE1,IPHALF1,IPZERO1,IELC2,IPONE2,IPHALF2,IPZERO2
C
C
      INCLUDE 'domain_common'
      DIMENSION IDOTOP(9,15),IELOC(2),NSID(2)
C
      DO 10 N = 1,9
      DO 10 I = 1,15
 10   IDOTOP(N,I) = 0
C
      NELT = LNOCON(1,NNIS)
C
      DO 700 LE = 1,NELT
C
        LEP = (LE-1)*2+2
        LELT = LNOCON(LEP  ,NNIS)
        IPL  = LNOCON(LEP+1,NNIS)
C
        NNLC = LELCON(IPL,ISIDE,LELT)
        IELC = LELCON( 4 ,ISIDE,LELT)
        NSIDE= LELCON( 5 ,ISIDE,LELT)
C
        DO 100 NN = 1,NNELT
          KP = (NN-1)*2+3
 100      IF(IELTOP(KP,IELC).EQ.NNLC) IPONE = NN
C
        CALL FINDPN(NSIDE,IPONE,IPHALF,IPZERO)
        IF (KERROR.GT.0) GO TO 900
C
        IDOTOP(1,1) = IDOTOP(1,1) +1
         KP = (IDOTOP(1,1)-1)*4 +2
        IDOTOP(KP  ,1) = IELC
        IDOTOP(KP+1,1) = IPONE
        IDOTOP(KP+2,1) = IPHALF
        IDOTOP(KP+3,1) = IPZERO
C
        DO 600 NLAY = 2,LAYMAX
C
          NLCZP = (IPZERO-1)*2 +3
          NNLCZ = IELTOP(NLCZP,IELC)
C
          NSOP = NSIDOP(NSIDE)
          IF (KERROR.GT.0) GO TO 900
C
          CALL SIDNOD(0,IP1,IP2,IP3,NSOP)
          IF (KERROR.GT.0) GO TO 900
C
          N1P = (IP1-1)*2+3
          NNLC1 = IELTOP(N1P,IELC)
          N2P = (IP2-1)*2+3
          NNLC2 = IELTOP(N2P,IELC)
          N3P = (IP3-1)*2+3
          NNLC3 = IELTOP(N3P,IELC)
C
          CALL FINDIE(NNLC1,NNLC2,NNLC3,NBEL,IELOC,NSID)
C
          IF (NBEL.GT.1) GO TO 200
C
            KERROR = KERROR +1
            WRITE(NOWC,1000) NNLC1,NNLC2,NNLC3,IELC
            GO TO 900
C
 200      CONTINUE
C
          DO 300 NB = 1, NBEL
C
           IF(IELOC(NB).EQ.IELC) GO TO 300
            IELC = IELOC(NB)
            NSIDE = NSID(NB)
            IF(NSIDE.NE.0) GO TO 400
```

```
          KERROR = KERROR +1
          WRITE(NOWC,2000) NNLC1,NNLC2,NNLC3,IELC
 300      CONTINUE
 400      CONTINUE
C
        DO 500 NN = 1,NNELT
          KP = (NN-1)*2+3
 500      IF(IELTOP(KP,IELC).EQ.NNLCZ) IPONE = NN
C
        CALL FINDPN(NSIDE,IPONE,IPHALF,IPZERO)
        IF (KERROR.GT.0) GO TO 900
C
        IDOTOP(1,NLAY) = IDOTOP(1,NLAY) +1
         KP = (IDOTOP(1,NLAY)-1)*4 +2
        IDOTOP(KP  ,NLAY) = IELC
        IDOTOP(KP+1,NLAY) = IPONE
        IDOTOP(KP+2,NLAY) = IPHALF
        IDOTOP(KP+3,NLAY) = IPZERO
C
 600    CONTINUE
 700  CONTINUE
C
 900  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR.IDTCAL  * * *',//,
     & 10X,'CANNOT FIND ANOTHER ELEMENT TO WHICH NODES',3(2X,I4),
     & 1X,'BELONG. START FROM ELT NUMBER = ',I4)
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR.IDTCAL  * * *',//,
     & 10X,' NODES',3(2X,I4),'BELONG TO ELT ',I4,'BUT THEY DO NOT',
     & 10X,'BELONG. TO THE SAME SIDE ')
C
     RETURN
     END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *       S U B R O U T I N E  I N T A C T
C  *
C  *
C  ****************************************************************
C
   SUBROUTINE INTACT(IELC,BINTG,NOUT)
C
C  Find the integral over the element IELC of
C
C          - W*Q1,1 + SIGij*Uj,1*Q1,i
C
C  by means of Gauss quadrature
C
C  Parameters
C    I/   IELC  : Local numb. of elt
C    O/   BINTG : Integral over IELC of F=- W*Q1,1+SIGij*Uj,1*Q1,i
C               = sum|ip of F[ip]*Jac[ip]*Weight[ip]
C
   INCLUDE 'domain_common'
   DIMENSION F(9), QGR(3,3), UGR(3,3), SIG(3,3), TGR(3)
   DIMENSION AJACOB(9), X(2), SIGL(3,3), EPSL(3,3), UGRL(3,3)
C
   BINTG = 0.d+00
   ONENE = -1.d+00
   PI = DACOS(ONENE)
   OOPI = 1.d+00/PI
   OMPS = 1.d+00 - POISS**2
   OPP = 1.d+00 + POISS
   OOY = 1.d+00/YOUNG
   ZERO = 0.d+00
C
   CALL JACALC(IELC,AJACOB)
C
   DO 400 IP = 1,NINTP
C
     IF(ITEMP.EQ.1) THEN
C
       TGR(1) = TGRAD(1,IP,IELC)
       TGR(2) = TGRAD(2,IP,IELC)
       TGR(3) = TGRAD(3,IP,IELC)
C
     END IF
```

```
C
      QGR(1,1) = QGRAD(1,IP,IELC)
      QGR(2,2) = QGRAD(2,IP,IELC)
      QGR(3,3) = QGRAD(3,IP,IELC)
      QGR(1,2) = QGRAD(4,IP,IELC)
      QGR(1,3) = QGRAD(5,IP,IELC)
      QGR(2,3) = QGRAD(6,IP,IELC)
      QGR(2,1) = QGRAD(7,IP,IELC)
      QGR(3,1) = QGRAD(8,IP,IELC)
      QGR(3,2) = QGRAD(9,IP,IELC)
C
      UGR(1,1) = UGRAD(1,IP,IELC)
      UGR(2,2) = UGRAD(2,IP,IELC)
      UGR(3,3) = UGRAD(3,IP,IELC)
      UGR(1,2) = UGRAD(4,IP,IELC)
      UGR(1,3) = UGRAD(5,IP,IELC)
      UGR(2,3) = UGRAD(6,IP,IELC)
      UGR(2,1) = UGRAD(7,IP,IELC)
      UGR(3,1) = UGRAD(8,IP,IELC)
      UGR(3,2) = UGRAD(9,IP,IELC)
C
      SIG(1,1) = SIGMIP(1,IP,IELC)
      SIG(2,2) = SIGMIP(2,IP,IELC)
      SIG(3,3) = SIGMIP(3,IP,IELC)
      SIG(1,2) = SIGMIP(4,IP,IELC)
      SIG(1,3) = SIGMIP(5,IP,IELC)
      SIG(2,3) = SIGMIP(6,IP,IELC)
      SIG(2,1) = SIGMIP(7,IP,IELC)
      SIG(3,1) = SIGMIP(8,IP,IELC)
      SIG(3,2) = SIGMIP(9,IP,IELC)
C
C     Calculate the coordinates of the IP
C
      DO 100 NC=1,NDIM
        SUM = 0.
        DO 110 K=1,NNELT
          INLC = 3+(K-1)*2
          NNLC = IELTOP(INLC,IELC)
110       SUM = SUM + SFNITP(IP,K) * COORDS(NC,NNLC)
100     X(NC) = SUM
C
      FRAC = X(2)/X(1)
      THETA = DATAN(FRAC)
      IF(X(1).GT.ZERO) GO TO 115
       THETA = PI - DABS(THETA)
115    CONTINUE
      THDEG = THETA*180/PI
      ARG = X(1)**2 + X(2)**2
      RAD = DSQRT(ARG)
      PIRA = PI * RAD
      UARG = -FLINE/(PI*RAD*YOUNG) * OPP
C
C     Print the coordinates of the IPs
C
      IF(NOUT.EQ.1) THEN
        WRITE(NOWN,1000) IP,IELC,X(1),X(2),THDEG
      END IF
C
C     Calculate Q1
C
      SUM = 0.d+00
      DO 120 K=1,NNELT
120       SUM = SUM + SFNITP(IP,K) * RPERT(K,IELC,1)
      Q1 = SUM
C
C     Line load solutions
C
      SIGL(1,1) = - FLINE * DCOS(THETA)**3/PIRA
      SIGL(2,2) = - FLINE * DSIN(THETA)**2 * DCOS(THETA)/PIRA
      SIGL(3,3) = - FLINE * POISS * DCOS(THETA)/PIRA
      SIGL(1,2) = - FLINE * DSIN(THETA) * DCOS(THETA)**2/PIRA
      SIGL(1,3) = 0.d+00
      SIGL(2,3) = 0.d+00
      SIGL(2,1) = SIGL(1,2)
      SIGL(3,1) = 0.d+00
      SIGL(3,2) = 0.d+00
C
      EPSL(1,1) = OOY*(OMPS*SIGL(1,1)-POISS*OPP*SIGL(2,2))
      EPSL(2,1) = OOY*OPP*SIGL(1,2)
      EPSL(1,2) = EPSL(2,1)
      EPSL(2,2) = OOY*(OMPS*SIGL(2,2)-POISS*OPP*SIGL(1,1))
C
      UGRL(1,1) = EPSL(1,1)
      UGRL(2,1) = UARG*DSIN(THETA)*(POISS-DSIN(THETA)**2)
      UGRL(1,2) = -UARG*DSIN(THETA)*(POISS-DSIN(THETA)**2-
     &            2.d+00*DCOS(THETA)**2)
C
      DIVQ = QGR(1,1)
C
C       WRITE(NOWM,4000) IELC, IP
      SUM = 0.d+00
      DO 300 J = 1,NDIM
        AMULT = 0.
        DO 200 I = 1,NDIM
```

```
            ADD1 = SIGL(I,J)*UGR(I,1)
            ADD2 = SIG(I,J)*UGRL(I,1)
            ADD = ADD1 + ADD2
            AMULT = AMULT + ADD
C             WRITE(NOWM,3000) I, J, ADD1, ADD2, ADD, AMULT
 200          CONTINUE
C             WRITE(NOWM,2000) QGR(1,J)
            ADDSUM = AMULT*QGR(1,J)
C             WRITE(NOWM,2000) ADDSUM
          SUM = SUM + ADDSUM
C         WRITE(NOWM,2000) SUM
 300      CONTINUE
C         WRITE(NOWM,2000) SUM
C
          TOT = SIG(1,1)*EPSL(1,1) + SIG(2,1)*EPSL(2,1) +
     &          SIG(1,2)*EPSL(1,2) + SIG(2,2)*EPSL(2,2)
C
C         WRITE(NOWM,2000) TOT
          F(IP) = SUM - TOT*DIVQ
C         WRITE(NOWM,2000) F(IP)
C
          TOTT = SIGL(1,1) + SIGL(2,2) + SIGL(3,3)
          THPART = COTHER * TGR(1) * TOTT * Q1
C         WRITE(NOWM,2000) TOTT
C         WRITE(NOWM,2000) THPART
          IF(ITEMP.EQ.1) F(IP) = F(IP) + THPART
C         WRITE(NOWM,2000) F(IP)
C
        ADDINT = WEIGHT(IP)*F(IP)*AJACOB(IP)
        BINTG = BINTG + ADDINT
C       WRITE(NOWM,2000) ADDINT
C       WRITE(NOWM,2000) BINTG
C
C       WRITE(NOWM,5000) IELC, IELTOP(1,IELC),IP
C       DO 10 I=1,3
C  10 WRITE(NOWM,6000) (SIGL(I,J),J=1,3)
C       DO 11 I=1,2
C  11 WRITE(NOWM,7000) (EPSL(I,J),J=1,2)
C       WRITE(NOWM,7000) UGRL(1,1), UGRL(2,1)
C
 400 CONTINUE
C
1000 FORMAT(1X,2(I4),2(1X,E12.4),3X,F7.2)
2000 FORMAT(E12.4)
3000 FORMAT(2(I4),4(E12.4))
4000 FORMAT(2(I4))
5000 FORMAT(3(I5))
6000 FORMAT(3(E12.4))
7000 FORMAT(2(E12.4))
C
        RETURN
        END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *     SUBROUTINE INTPRIN
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE INTPRIN
C
C     Orders the J-integral and T-stress values according to the
C     number of elements in the domain and writes them in a plot file
C
      INCLUDE 'domain_common'
      DIMENSION NELTO(40),NELO(40)
C
C     J-integral
C
      IF(NDOMT.EQ.1) GO TO 999
C
      DO 15 N=1,NTOUT
      DO 212 M=1,NDOMT
        NELO(M)=NELEM(M)
  212   NELTO(M)=NTLELT(M)
      DO 12 J=2,NDOMT
        NA=NELO(J)
        B=TJ(J,N)
        C=TNORM(J,N)
        ND=NELTO(J)
        E=TSTR(J,N)
        F=VTINTA(J,N)
        G=VKONE(J,N)
        H=VKONOR(J,N)
        DO 11 I=J-1,1,-1
          IF(NELO(I).LE.NA)GO TO 10
          NELO(I+1)=NELO(I)
          TJ(I+1,N)=TJ(I,N)
          TNORM(I+1,N)=TNORM(I,N)
          NELTO(I+1)=NELTO(I)
          TSTR(I+1,N)=TSTR(I,N)
          VTINTA(I+1,N)=VTINTA(I,N)
          VKONE(I+1,N)=VKONE(I,N)
          VKONOR(I+1,N)=VKONOR(I,N)
   11   CONTINUE
        I=0
   10   NELO(I+1)=NA
        TJ(I+1,N)=B
        TNORM(I+1,N)=C
        NELTO(I+1)=ND
        TSTR(I+1,N)=E
        VTINTA(I+1,N)=F
        VKONE(I+1,N)=G
        VKONOR(I+1,N)=H
   12   CONTINUE
   15   CONTINUE
C
      DO 21 N=1,NDOMT
        NELEM(N)=NELO(N)
   21   NTLELT(N)=NELTO(N)
C
  999 CONTINUE
C
      IF(NTOUT.EQ.1) THEN
        WRITE(NOWP(1),1000) IPFLAG, ITEMP, IRDINT
        WRITE(NOWP(1),4000)
        WRITE(NOWP(2),2000) IPFLAG, ITEMP, IRDINT
        WRITE(NOWP(2),4000)
        WRITE(NOWP(9),2100) IPFLAG, ITEMP, IRDINT
        WRITE(NOWP(9),4000)
        WRITE(NOWP(10),2200) IPFLAG, ITEMP, IRDINT
        WRITE(NOWP(10),4000)
        WRITE(NOWD,1000) IPFLAG, ITEMP, IRDINT
        DO 100 I=1,NDOMT
          WRITE(NOWD,3000) NELEM(I),TJ(I,NTOUT)
          WRITE(NOWP(1),3000) NELEM(I),TJ(I,NTOUT)
          WRITE(NOWP(2),3000) NELEM(I),TNORM(I,NTOUT)
          WRITE(NOWP(9),3000) NELEM(I),VKONE(I,NTOUT)
  100     WRITE(NOWP(10),3000) NELEM(I),VKONOR(I,NTOUT)
C
        WRITE(NOWD,2000) IPFLAG, ITEMP, IRDINT
        DO 200 I=1,NDOMT
  200     WRITE(NOWD,3000) NELEM(I),TNORM(I,NTOUT)
      END IF
C
      IF(ITEMP.EQ.1) THEN
C
      DELTH = TINIT - TAMB
      VJNF = (1.d+00 -POISS)**2/(CRACK*YOUNG*COTHER**2*DELTH**2)
      WRITE(NOWP(4),5000) IPFLAG, ITEMP, IRDINT, BIOT
      WRITE(NOWP(5),6000) IPFLAG, ITEMP, IRDINT, BIOT
      WRITE(NOWP(9),6100) IPFLAG, ITEMP, IRDINT, BIOT
      WRITE(NOWP(10),6200) IPFLAG, ITEMP, IRDINT, BIOT
```

```
      WRITE(NOWP(11),6300) IPFLAG, ITEMP, IRDINT, BIOT
      DO 16 I=1,NDOMT
        WRITE(NOWP(4),4000)
        WRITE(NOWP(5),4000)
        WRITE(NOWP(9),4000)
        WRITE(NOWP(10),4000)
        WRITE(NOWP(11),4000)
      DO 13 N=1,NTOUT
        VJNOR = TJ(I,N) * VJNF
        WRITE(NOWP(4),7100) FOU(N),VJNOR,NKINCR(N),TIM(N),TJ(I,N)
        WRITE(NOWP(5),7000) FOU(N),TNORM(I,N),NKINCR(N),TIM(N)
        WRITE(NOWP(9),7000) FOU(N),VKONE(I,N),NKINCR(N),TIM(N)
        WRITE(NOWP(10),7000) FOU(N),VKONOR(I,N),NKINCR(N),TIM(N)
        WRITE(NOWP(11),7000) TNORM(I,N),VKONOR(I,N),NKINCR(N),TIM(N)
 13   CONTINUE
 16   CONTINUE
C
      DO 500 J=1,NCONJ
        WRITE(NOWP(4),4100)
        DO 500 N=1,NTOUT
          VARJN = VARJ(J,N) * VJNF
 500      WRITE(NOWP(4),7100) FOU(N),VARJN,NKINCR(N),TIM(N),
     &                 VARJ(J,N)
C
      ELSE
C
      WRITE(NOWP(4),8100) IPFLAG, ITEMP, IRDINT
      WRITE(NOWP(5),8200) IPFLAG, ITEMP, IRDINT
      WRITE(NOWP(9),8300) IPFLAG, ITEMP, IRDINT
      DO 17 I=1,NDOMT
        WRITE(NOWP(4),4000)
        WRITE(NOWP(5),4000)
        WRITE(NOWP(9),4000)
      DO 18 N=1,NTOUT
        WRITE(NOWP(4),3000) NKINCR(N),TJ(I,N)
        WRITE(NOWP(5),3000) NKINCR(N),TNORM(I,N)
        WRITE(NOWP(9),3000) NKINCR(N),VKONE(I,N)
 18   CONTINUE
 17   CONTINUE
C
      DO 510 J=1,NCONJ
        WRITE(NOWP(4),4100)
        DO 510 N=1,NTOUT
 510      WRITE(NOWP(4),3000) NKINCR(N),VARJ(J,N)
C
      END IF
C
      WRITE(NOWO,990)
      DO 300 N=1,NDOMT
        LAYMAX = NELEM(N)/NTLELT(N)
 300    WRITE(NOWO,991) N, NELEM(N), NTLELT(N),LAYMAX
C
      DO 400 N=1,NTOUT
        WRITE(NOWO,992)NKSTEP(N),NKINCR(N),TIM(N)
        IF(ITDIS.EQ.1) GO TO 400
        WRITE(NOWO,993)
        WRITE(NOWO,994) (VARJ(I,N),I=1,NCONJ)
        WRITE(NOWO,995) (TJ(I,N),I=1,NDOMT)
        WRITE(NOWO,770)
        IF(ITEMP.EQ.1) THEN
          WRITE(NOWO,771) (VTINTA(I,N),I=1,NDOMT)
          WRITE(NOWO,772) (TSTR(I,N),I=1,NDOMT)
          WRITE(NOWO,773) (TNORM(I,N),I=1,NDOMT)
        ELSE
          WRITE(NOWO,771) (VTINTA(I,N),I=1,NDOMT)
          WRITE(NOWO,772) (TSTR(I,N),I=1,NDOMT)
          WRITE(NOWO,774) SFAR(N)
          WRITE(NOWO,775) (TNORM(I,N),I=1,NDOMT)
        END IF
 400  CONTINUE
C
C
 990 FORMAT(1H1,////,20X,'* * * D O M A I N  I N F O * * *',/)
 991 FORMAT('Domain No. ',I4,//,
     & 'Number of elements in domain: ',I4,1X,'(',I3,1X,'x',I3,')',/)
 992 FORMAT(1H1,//,'STEP NO. ',I4,//,
     & 'INCREMENT NO. ',I4,//,
     & 'TIME: ',E10.4,/)
 993 FORMAT(1H1,////,20X,'* * *  J - I N T E G R A L  '
     & 'E S T I M A T E S  * * *',//,
     & 20X,'     C O N T O U R S',//,
     & 1X,'            1       2       3       4'
     & '       5       6',/)
 994 FORMAT(1H1,/,1X,'ABAQUS:    ',6(1X,E14.7))
 995 FORMAT(1H1,/,1X,'Calculated: ',6(1X,E14.7))
 770 FORMAT(1H1,////,20X,'* * *  I N T E R A C T I O N - '
     & 'I N T E G R A L  * * *',//,
     & 1X,'            1       2       3       4'
     & '       5       6',/)
 771 FORMAT(1H1,/,1X,' Value:   ',6(E14.7,2X),/)
 772 FORMAT(1H1,/,1X,' T-stress: ',6(E14.7,2X),//,
     & 1X,'  (psi)  ',/)
 773 FORMAT(1H1,/,1X,' Tau:     ',6(E14.7,2X),/)
 774 FORMAT(1H1,/,1X,' Far-field      ',//,
```

```
     &  1X,'stress (psi): ',E14.7,/)
  775 FORMAT(1H1,/,1X,' Normalized     ',//,
     &  1X,' T-stress: ',6(E14.7,2X))
 1000 FORMAT(1X,'number of elements',/,1X,'J-integral',/,'type 3',/,
     &      'labl 3',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      'end')
 2000 FORMAT(1X,'number of elements',/,1X,'T-stress',/,'type 3',/,
     &      'labl 3',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      'end')
 2100 FORMAT(1X,'number of elements',/,1X,'K\I',/,'type 3',/,
     &      'labl 3',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      'end')
 2200 FORMAT(1X,'number of elements',/,1X,
     &      'K\I (1-@n) / E@a(Tinit-Tamb) (@pa)1/2 ',/,
     &      'type 3',/,
     &      'labl 3',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      'end')
 3000 FORMAT(1X,I5,2X,E17.10)
 4000 FORMAT(1X,'1.0E32  101')
 4100 FORMAT(1X,'1.0E32  102')
 5000 FORMAT(1X,' @at/w^2 ',/,
     &      1X,'J(1-@n^2)/aE@a^2(Tinit-Tamb)^2',/,'type 3',/,
     &      'labl 4',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      '1.7 3. .13',/,'BIOT= ',F6.2,/,
     &      'end')
 6000 FORMAT(1X,' @at/w^2 ',/,1X,' T / E@a(Tinit-Tamb) ',/,
     &      'type 3',/,
     &      'labl 4',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      '1.7 3. .13',/,'BIOT= ',F6.2,/,
     &      'end')
 6100 FORMAT(1X,' @at/w^2 ',/,1X,' K\I ',/,'type 3',/,
     &      'labl 4',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      '1.7 3. .13',/,'BIOT= ',F6.2,/,
     &      'end')
 6200 FORMAT(1X,' @at/w^2 ',/,1X,
     &      ' K\I (1-@n) / E@a(Tinit-Tamb) (@pa)1/2 ',/,
     &      'type 3' ,/,
     &      'labl 4',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      '1.7 3. .13',/,'BIOT= ',F6.2,/,
     &      'end')
 6300 FORMAT(1X,'T / E@a (Tinit - Tamb)',/,1X,
     &      'K\I (1-@n) / E@a(Tinit - Tamb) (@pa)1/2 ',/,
     &      'type 3' ,/,
     &      'labl 4',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      '1.7 3. .13',/,'BIOT= ',F6.2,/,
     &      'end')
 7000 FORMAT(1X,2(2X,E17.10),2X,I5,2X,E17.10)
 7100 FORMAT(1X,2(2X,E17.10),2X,I5,2X,(2X,E17.10))
 8000 FORMAT(1X,2(2X,E17.10),2X,I5)
 8100 FORMAT(1X,'increment #',/,1X,'J-integral',/,'type 3',/,
     &      'labl 3',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      'end')
 8200 FORMAT(1X,'increment #',/,1X,' @t ',/,'type 3',/,
     &      'labl 3',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      'end')
 8300 FORMAT(1X,'increment #',/,1X,'K\I',/,'type 3',/,
     &      'labl 3',/,'1.7 4.5 .13',/,'IPFLAG = ',I5,/,
     &      '1.7 4. .13',/,'ITEMP= ',I5,/,
     &      '1.7 3.5 .13',/,'IRDINT= ',I5,/,
     &      'end')
C
      RETURN
      END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *      SUBROUTINE ITFCON
C   *
C   *
C   ****************************************************************
C
C
      SUBROUTINE ITFCON(NDOM)
C
C
C     Evaluates the connectivity matrices  for itf. elts LELCON & LNOCON
C     Evaluates the total number of linear  element NTLELT
C
      INCLUDE 'domain_common'
      DIMENSION IELC(2),NSIDE(2)
C
      NTLELT(NDOM) = (NTOTIS-1)/2
C
      DO 300 LELT = 1,NTLELT(NDOM)
C
      DO 100 NN = 1,NNLELT
C
        NNIS = (LELT-1)*2+NN
        LNOCON (1,NNIS)=LNOCON (1,NNIS)+1
        K =  (LNOCON (1,NNIS)-1)*2 +2
        LNOCON (K,NNIS)=LELT
        LNOCON (K+1,NNIS)=NN
C
      DO 100 IS = 1,2
        ISIDE = IS
        IF (IDOUBL.EQ.1) ISIDE = 1
  100 LELCON (NN,IS,LELT) = NCONN (2,ISIDE,NNIS)
C
      DO 200 IS = 1,2
C
      CALL FINDIE(LELCON(1,IS,LELT),LELCON(2,IS,LELT),LELCON(3,IS,LELT),
     &            NBEL,IELC,NSIDE)
      DO 200 NBE = 1,NBEL
      IF((IS.EQ.1.AND.NSIDE(NBE).GT.0).OR.(IS.EQ.2.AND.NSIDE(NBE).LT.0))
     &  THEN
        LELCON (4,IS,LELT)=IELC(NBE)
        LELCON (5,IS,LELT)=NSIDE(NBE)
      ENDIF
  200 CONTINUE
  300 CONTINUE
C
C
      IF (IPRINT.EQ.0) GO TO 600
        WRITE(NOWG,1000)
        DO 400 LELT = 1,NTLELT(NDOM)
  400   WRITE(NOWG,2000) ((LELCON(I,J,LELT),I=1,5),J=1,2)
C
        WRITE(NOWG,3000)
        DO 500 NNIS = 1, NTOTIS
  500   WRITE(NOWG,4000) (LNOCON(I,NNIS),I = 1,5)
C
  600 CONTINUE
C
 1000 FORMAT(1H1,///,40X,'* * * L E L C O N * * *',//,
     & 14X,'ISIDE = 1',28X,'ISIDE = 2 ',//,
     & 1X,'NNLC1 NNLC2 NNLC3  IELC NSIDE   NNLC1 NNLC2 NNLC3',
     &   '  IELC NSIDE',//)
 2000 FORMAT(5(1X,I5,1X),2X,5 (1X,I5,1X))
 3000 FORMAT(1H1,///,40X,'* * * L N O C O N * * *',//,
     & 1X,' NELT  LELT1  IP1  LELT2  IP2',//)
 4000 FORMAT(5(1X,I5,1X))
C
      RETURN
      END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE ITFDEF
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE ITFDEF(IFISNO)
C
C     Reads input set nodes. Find total number of input set nodes
C     Fills NCONN(1,ISIDE,NNIS)
C
C     Parameters
C       O/    IFISNO(20000) :NNIS input-set-number for abaqus nodes
C                    if >0 -> node on outer interface
C                    if <0 -> node on the inner interface
C
      INCLUDE 'domain_common'
      DIMENSION IFISNO(20000),NINP(15)
C
      ISIGN = 1
      DO 300 ISIDE = 1,IDOUBL
        ISIGN = -ISIGN
        NNIS = 0
        READ(NORI,1000) NSET
        DO 100 NS=1,NSET
        READ(NORI,2000) IN,(NINP(K),K=1,15)
        IF(IN.NE.0)THEN
          NF = NINP(1)
          NL = NINP(2)
          NT = 1+(NL-NF)/IN
          DO 30 NN = 1,NT
            NNIS = NNIS+1
            NNAB = NF + (NN-1)*IN
            NCONN(1,ISIDE,NNIS) = NNAB
            IFISNO(NNAB) = NNIS*ISIGN
 30       CONTINUE
        ELSE
          DO 60 NN=1,15
          IF(NINP(NN).EQ.0) GO TO 60
            NNAB = NINP(NN)
            NNIS = NNIS +1
            NCONN(1,ISIDE,NNIS) = NNAB
            IFISNO(NNAB) = NNIS*ISIGN
 60       CONTINUE
        ENDIF
 100    CONTINUE
        NTOTIS = NNIS
C
C       Check the max number of nodes
C
        IF (NTOTIS.LE.400) GO TO 200
          KERROR = KERROR +1
          WRITE(NOWC,3000) ISIDE,NTOTIS
 200    CONTINUE
 300  CONTINUE
C
 1000 FORMAT(I5)
 2000 FORMAT(16(I5))
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. ITFDEF  * * *',//,
     & 10X,'MAX NUMBER OF INPUT SET NODES  EXCEEDED ON SIDE ',I2,//,
     & 10X,'TOTAL NUMBER OF I.S. NODES = ',I4,' (MAX: 400)')
C
      RETURN
      END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE JACALC
C  *
C  *
C  ****************************************************************
C
   SUBROUTINE JACALC(IELC,AJACOB)
C
C  Evaluate the Jacobian at the integration points  of element IELC
C
C  Paramweters
C     I/    IELC : local elt id number
C     O/    AJACOB: Jacobian:
C                AJACOB(I) = jacobian at int. pt. I
C
C           AJACOB= DX/DG*DY/DH-DY/DG*DX/DH
C
   INCLUDE 'domain_common'
   DIMENSION AJACOB(9)
C
   DO 10 NC = 1,9
10  AJACOB(NC) = 0.d+00
C
   DO 200 IP = 1,NINTP
C
     DXDG = 0.d+00
     DXDH = 0.d+00
     DYDG = 0.d+00
     DYDH = 0.d+00
     DO 100 K = 1, NNELT
C
       INLC = 3 +(K-1)*2
       NNLC = IELTOP(INLC,IELC)
C
       DXDG = DXDG +SFDITP(IP,K,1)*COORDS(1,NNLC)
       DXDH = DXDH +SFDITP(IP,K,2)*COORDS(1,NNLC)
       DYDG = DYDG +SFDITP(IP,K,1)*COORDS(2,NNLC)
       DYDH = DYDH +SFDITP(IP,K,2)*COORDS(2,NNLC)
100    CONTINUE
     AJACOB(IP) = DXDG*DYDH-DYDG*DXDH
200 CONTINUE
C
   RETURN
   END
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE JINTGR
C  *
C  *
C  ****************************************************************
C
   SUBROUTINE JINTGR(IELC,AINTG)
C
C  Find the integral over the element IELC of
C
C          - W*Q1,1 + SIGij*Uj,1*Q1,i
C
C  by means of Gauss quadrature
C
C  Parameters
C    I/   IELC  : Local numb. of elt
C    O/   AINTG : Integral over IELC of F=- W*Q1,1+SIGij*Uj,1*Q1,i
C              = sum|ip of F[ip]*Jac[ip]*Weight[ip]
C
   INCLUDE 'domain_common'
   DIMENSION F(9), QGR(3,3), UGR(3,3), SIG(3,3), TGR(3)
   DIMENSION AJACOB(9)
C
   DO 399 N=1,9
399      F(N)=0.d+00
C
   AINTG = 0.d+00
C
   CALL JACALC(IELC,AJACOB)
C
   DO 400 IP = 1,NINTP
C
    IF(ITEMP.EQ.1) THEN
C
      TGR(1) = TGRAD(1,IP,IELC)
      TGR(2) = TGRAD(2,IP,IELC)
      TGR(3) = TGRAD(3,IP,IELC)
C
    END IF
C
     QGR(1,1) = QGRAD(1,IP,IELC)
     QGR(2,2) = QGRAD(2,IP,IELC)
     QGR(3,3) = QGRAD(3,IP,IELC)
```

```
      QGR(1,2) = QGRAD(4,IP,IELC)
      QGR(1,3) = QGRAD(5,IP,IELC)
      QGR(2,3) = QGRAD(6,IP,IELC)
      QGR(2,1) = QGRAD(7,IP,IELC)
      QGR(3,1) = QGRAD(8,IP,IELC)
      QGR(3,2) = QGRAD(9,IP,IELC)
C
      UGR(1,1) = UGRAD(1,IP,IELC)
      UGR(2,2) = UGRAD(2,IP,IELC)
      UGR(3,3) = UGRAD(3,IP,IELC)
      UGR(1,2) = UGRAD(4,IP,IELC)
      UGR(1,3) = UGRAD(5,IP,IELC)
      UGR(2,3) = UGRAD(6,IP,IELC)
      UGR(2,1) = UGRAD(7,IP,IELC)
      UGR(3,1) = UGRAD(8,IP,IELC)
      UGR(3,2) = UGRAD(9,IP,IELC)
C
      SIG(1,1) = SIGMIP(1,IP,IELC)
      SIG(2,2) = SIGMIP(2,IP,IELC)
      SIG(3,3) = SIGMIP(3,IP,IELC)
      SIG(1,2) = SIGMIP(4,IP,IELC)
      SIG(1,3) = SIGMIP(5,IP,IELC)
      SIG(2,3) = SIGMIP(6,IP,IELC)
      SIG(2,1) = SIGMIP(7,IP,IELC)
      SIG(3,1) = SIGMIP(8,IP,IELC)
      SIG(3,2) = SIGMIP(9,IP,IELC)
C
      IF(ITEMP.EQ.1) THEN
        SUM = 0.d+00
        DO 111 K=1,NNELT
          INLC = 3 + (K-1) * 2
          NNLC = IELTOP(INLC,IELC)
  111     SUM = SUM + SFNITP(IP,K) * TEMP(NNLC)
        TEMPIP = SUM
        THERM = (TEMPIP - TINIT) * COTHER
        GAMM12 = UGR(1,2) + UGR(2,1)
        STRENER = 0.5d+00*( (SIG(1,1)*(UGR(1,1)-THERM)) +
     &            (SIG(2,2)*(UGR(2,2)-THERM)) +
     &            (SIG(3,3)*(UGR(3,3)-THERM)) +
     &            (SIG(1,2)*GAMM12) )
      ELSE
        GAMM12 = UGR(1,2) + UGR(2,1)
        STRENER = 0.5d+00*( (SIG(1,1)*UGR(1,1)) +
     &            (SIG(2,2)*UGR(2,2)) +
     &            (SIG(1,2)*GAMM12) )
      ENDIF
C
C     Calculate Q1
C
      SUM = 0.d+00
      DO 120 K=1,NNELT
  120   SUM = SUM + SFNITP(IP,K) * RPERT(K,IELC,1)
      Q1 = SUM
C
      DIVQ = QGR(1,1)
C
      SUM = 0.d+00
      DO 300 I = 1,NDIM
        AMULT = 0.d+00
        DO 200 J = 1,NDIM
  200     AMULT = AMULT + SIG(I,J)*UGR(J,1)
  300   SUM = SUM +AMULT*QGR(1,I)
C
      F(IP) = SUM - WENIP(IP,IELC)*DIVQ
C     F(IP) = SUM - STRENER*DIVQ
C
      COM = SIG(1,1) + SIG(2,2) + SIG(3,3)
      THPART = COTHER*TGR(1)*COM*Q1
C     WRITE(NOWM,1000) IELC,IP,STRENER,WENIP(IP,IELC),
C    &            F(IP),THPART
C
      IF(ITEMP.EQ.1) F(IP) = F(IP) + THPART
C
      ADDINT = WEIGHT(IP)*F(IP)*AJACOB(IP)
C
C     WRITE(NOWM,2000) WEIGHT(IP), AJACOB(IP), ADDINT
C
      AINTG = AINTG + ADDINT
C
C     WRITE(NOWM,3000) AINTG
C
  400 CONTINUE
C
 1000 FORMAT (2(I5,2X),4(E12.3))
 2000 FORMAT (3(E12.3))
 3000 FORMAT (E12.3)
C
      RETURN
      END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *      SUBROUTINE LUFACT
C   *
C   *
C   ****************************************************************
C
      SUBROUTINE LUFACT(KER,ID,IMOVE,INEW,AM,AU,AL)
C
C     Factorize a square matrix AM(ID,ID) into a lower triang. matrix
C     AL(ID,ID) and an upper triang. matrix AU(ID,ID) . AM = AL*AU
C     If elements on the main diagonal of AM are equal to zero,
C     rows of AM are permuted and a track is kept in array INEW.
C     Then (AM)perm. is factorized.
C     Parameters
C        I/O   KER : error count.
C        I/    ID : matrix dimension
C        O/    IMOVE : row permutation code (=1 : row permuted )
C        O/    INEW : perm. record :INEW(J)= old # of new row #J
C        I/    AM : square input matrix
C        O/    AL : lower triang matrix
C        O/    AU : upper triang.matrix
C
      INCLUDE 'domain_common'
C
      DIMENSION AM(ID,ID),AU(ID,ID),AL(ID,ID),INEW(ID)
C
C     Check if matrix AM has zero elements on the main diagonal
C     and permutation of rows if it is needed.
C
      CALL RWPERM (ID,IMOVE,INEW,AM,AU,AL)
C
C     Check if AM has any zero pivot
C
      IF(IMOVE.GE.0) GO TO 5
       KER = KER+1
       GO TO 200
 5    CONTINUE
C
      PIVMIN = 1.D-10
C
      DO 20 NC = 1,ID
      DO 10 NR = 1,ID
       AU(NR,NC) = AM(NR,NC)
 10    AL(NR,NC) = 0.d+00
 20    AL(NC,NC) = 1.d+00
C
      DO 100 NFR = 1,ID-1
         PIVOT = AU(NFR,NFR)
         IF(DABS(PIVOT).GT.PIVMIN) GO TO 50
          KER = KER +1
          WRITE (13,1000) NFR,PIVOT
          WRITE (13,*) AM
          IF(IMOVE.GT.0) WRITE(13,2000)
          IF(IMOVE.GT.0) WRITE(13,*) INEW
          GO TO 200
 50       CONTINUE
      DO 100 NSR = NFR+1,ID
         AL(NSR,NFR) = AU(NSR,NFR)/PIVOT
      DO 100 NC = NFR,ID
 100  AU(NSR,NC) = AU(NSR,NC)-AU(NFR,NC)*AL(NSR,NFR)
C
 200  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. LUFACT  * * *',//,
     & 10X,'ROW ',I4,' PIVOT TOO SMALL!!!! PIVOT = ',E12.5,//,
     & 10X,'INPUT MATRIX :',//)
 2000 FORMAT(////,20X,'* * * N O T E ! ! ! * * *',//,
     & 10X,'ROWS HAVE BEEN PERMUTED . PERMUTATION RECORD VECTOR ',
     & 10X,'INEW FOLLOWS: (INEW(J) = OLD # OF ACTUAL ROW # J)',/)
C
      RETURN
      END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *       S U B R O U T I N E   N C A R C O
C   *
C   *
C   ****************************************************************
C
C
      SUBROUTINE NCARCO(IDONE)
C
C     Reads the cartesian coordinates of the nodes
C
C     Parameters
C        I/     IDONE(20000) :local number for abaqus nodes
C
      INCLUDE 'domain_common'
C
      DOUBLE PRECISION ARRAY
      DIMENSION ARRAY(513),JRRAY(2,513),IDONE(20000)
      EQUIVALENCE (ARRAY(1), JRRAY(1,1))
C
C     Rewind file 8
C
      CALL DBFILE(2,ARRAY,JRCD)
C
C     Scanning file 8
C
      NCHECK = 0
      DO 200 K = 1,99999
         CALL DBFILE(0,ARRAY,JRCD)
         IF (JRCD.NE.0) GO TO 300
         LR = JRRAY(1,1)
         KEY = JRRAY(1,2)
         IF(KEY.NE.1901) GO TO 200
         NNAB = JRRAY(1,3)
         NNLC = IDONE(NNAB)
         IF(NNLC.EQ.0) GO TO 200
         NCHECK = NCHECK + 1
         DO 100 ND = 1, NDIM
  100       COORDS(ND,NNLC) = ARRAY(3+ND)
  200 CONTINUE
  300 CONTINUE
C
C     Check if all the nodes have been found
C
      IF(NCHECK.EQ.NTNOD) GO TO 400
        KERROR = KERROR +1
        WRITE(NOWC,1000) NCHECK,NTNOD
  400 CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. NCARCO  * * *',//,
     &  10X,'TOTAL NUMBER OF READ NODES :',I4,'IS NOT CONSISTENT WITH ',
     &  'THE TOTAL NUMBER OF NODES : ',I4)
 2000 FORMAT(1H1,////,40X,'* * *  C O O R D I N A T E S * * *',//
     &  'NNLC NNAB   X1       X2       X3',/)
 3000 FORMAT (2(1X,I4),3(2X,F6.3,2X))
C
      RETURN
      END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *      FUNCTION   NSIDOP
C   *
C   *
C   ****************************************************************
C
C
      FUNCTION NSIDOP(NSIDE)
C
C     Gives the identity of the element side opposite to nside
C
C     Parameters
C           NSIDE : input side of elt
C
C     NOTE!!!! ONLY FOR 2nd ORDER - 8 NODES - ISOPARAMETRIC 2D
ELEMENTS
C
================================================================
C
      NSIDOP = 0
C
      NS = NSIDE +5
C
      GO TO (100,200,300,400,500,600,700,800,900) NS
C
         KERROR = KERROR + 1
         WRITE(NOWC,1000) NSIDE
         GO TO 990
C
  100 CONTINUE
      NSIDOP = 2
      GO TO 990
  200 CONTINUE
      NSIDOP = 1
      GO TO 990
  300 CONTINUE
      NSIDOP = 4
      GO TO 990
  400 CONTINUE
      NSIDOP = 3
      GO TO 990
  500 CONTINUE
      KERROR = KERROR + 1
      WRITE(NOWC,1000) NSIDE
      GO TO 990
  600 CONTINUE
      NSIDOP = -3
      GO TO 990
  700 CONTINUE
      NSIDOP = -4
      GO TO 990
  800 CONTINUE
      NSIDOP = -1
      GO TO 990
  900 CONTINUE
      NSIDOP = -2
C
  990 CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.NSIDOP  * * *',//,
     & 10X,'THE SIDE NUMBER MUST BE BETW.-4 AND -1 OR 1 AND 4',//,
     & 10X,'SIDE NUMBER = ',I4)
C
      RETURN
      END
C
```

```
C
C  ******************************************************************
C  *
C  *
C  *      FUNCTION  PDSHFN
C  *
C  *
C  ******************************************************************
C
      FUNCTION PDSHFN(G,H,R,K,J,I)
C
C     Evaluate partial derivative of sh. function dN(k)/dc(j) at(G,H,R)
C
C     Parameters
C           G : 1st|  isoparametric coordinate of the location at
C           H : 2nd|  which the partial derivative must be
C           R : 3rd|  evaluated
C
C           K : number of the shape function
C           J : isoparametric coordinate with respect to which
C              the shape function must be differentiated
C           I : Element type
C                    8 : 2D - 8 nodes  isop. element
C
      INCLUDE 'domain_common'
C
      PDSHFN = 0.
C
      IF(I.EQ.8) GO TO 10
        KERROR = KERROR+1
        WRITE (NOWC,1000) I
        GO TO 900
   10 CONTINUE
C
      IF(J.EQ.1.OR.J.EQ.2) GO TO 20
        KERROR = KERROR+1
        WRITE (NOWC,2000) J
        GO TO 900
   20 CONTINUE
C
      GO TO (100,200,300,400,500,600,700,800) K
        KERROR = KERROR +1
        WRITE (NOWI,3000) K
        GO TO 900
C
  100 CONTINUE
      IF(J.EQ.1) PDSHFN = 0.25d+00*(1-H)*(2*G+H)
      IF(J.EQ.2) PDSHFN = 0.25d+00*(1-G)*(2*H+G)
      GO TO 900
C
  200 CONTINUE
      IF(J.EQ.1) PDSHFN = 0.25d+00*(1-H)*(2*G-H)
      IF(J.EQ.2) PDSHFN = 0.25d+00*(1+G)*(2*H-G)
      GO TO 900
C
  300 CONTINUE
      IF(J.EQ.1) PDSHFN = 0.25d+00*(1+H)*(2*G+H)
      IF(J.EQ.2) PDSHFN = 0.25d+00*(1+G)*(2*H+G)
      GO TO 900
C
  400 CONTINUE
      IF(J.EQ.1) PDSHFN = 0.25d+00*(1+H)*(2*G-H)
      IF(J.EQ.2) PDSHFN = 0.25d+00*(1-G)*(2*H-G)
      GO TO 900
C
  500 CONTINUE
      IF(J.EQ.1) PDSHFN = -(1-H)*G
      IF(J.EQ.2) PDSHFN = -0.5d+00*(1-G)*(1+G)
      GO TO 900
C
  600 CONTINUE
      IF(J.EQ.1) PDSHFN =  0.5d+00*(1-H)*(1+H)
      IF(J.EQ.2) PDSHFN = -(1+G)*H
      GO TO 900
C
  700 CONTINUE
      IF(J.EQ.1) PDSHFN = -(1+H)*G
      IF(J.EQ.2) PDSHFN =  0.5d+00*(1-G)*(1+G)
      GO TO 900
C
  800 CONTINUE
      IF(J.EQ.1) PDSHFN = - 0.5d+00*(1-H)*(1+H)
      IF(J.EQ.2) PDSHFN = -(1-G)*H
C
C
  900 CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.PDSHFN  * * *',//,
     &  10X,'ONLY ELEMENT-TYPE 8 IS IMPLEMENTED',//,
     &  10X,'ELEMENT-TYPE = ',I4)
 2000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.PDSHFN  * * *',//,
     &  10X,'THE REQUIRED-COORDINATE-CODE MUST BE 1 (G) OR 2(H)',//,
     &  10X,'REQUIRED COORDINATE CODE = ',I4)
```

```
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.PDSHFN  * * *',//,
    & 10X,'FOR THIS EL-TYPE THE NODE NUMBER MUST BE BETW. 1 AND 8',//,
    & 10X,'NODE NUMBER = ',I4)
C
     RETURN
     END
C
C
C
C   ****************************************************************
C   *
C   *
C   *      SUBROUTINE PERCAL
C   *
C   *
C   ****************************************************************
C
     SUBROUTINE PERCAL(NNIS,ISIDE,LAYMAX,IDOTOP,NTELD,NTND)
C
C    Evaluates the perturbation field of the domain
C
C    Parameters
C      I/ NNIS  : interface  node number
C      I/ ISIDE : side of the intface
C      I/ LAYMAX: farthest layer from the interface for which
C                 perturbation field must be established
C      I/ IDOTOP:  topology matrix
C                 IDOTOP(I,J) =  topology of layer J
C      O/ IDOEL :  Mapping of the domain integral element numbering
C                 to the local numbering
C      O/ NTELD :  Total number of elements in the domain
C
C      O/ NTND  :  Total number of nodes in the domain
C
C
     INCLUDE 'domain_common'
     DIMENSION IDOTOP(9,15)
C
     DO 300 NLAY = 1,LAYMAX
C
       NELLAY = IDOTOP(1,NLAY)
       DO 200 NE = 1,NELLAY
C
         KP = (NE-1)*4 +2
         IELC = IDOTOP(KP  ,NLAY)
         NONE = IDOTOP(KP+1,NLAY)
         NHALF= IDOTOP(KP+2,NLAY)
         NZERO= IDOTOP(KP+3,NLAY)
C
         IF(IECNT(IELC).NE.1) CALL DOMTOP(IELC,NTELD,NTND)
C
         IF(IPFLAG.EQ.1) THEN
           CALL PYRAMID(NNIS,IELC,NONE,NHALF,NZERO,NLAY,
    &             LAYMAX,IDOTOP,RPERT)
```

```
        IF(KERROR.NE.0) GO TO 900
      ELSE IF(IPFLAG.EQ.0) THEN
        CALL PLATEAU(NNIS,IELC,NONE,NHALF,NZERO,NLAY,
     &             LAYMAX,IDOTOP,RPERT)
        IF(KERROR.NE.0) GO TO 900
      END IF
C
 200  CONTINUE
 300 CONTINUE
C
 900 CONTINUE
C
    RETURN
    END
C
C
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE PLATEAU
C  *
C  *
C  ****************************************************************
C
    SUBROUTINE PLATEAU(NNIS,IELC,IPONE,IPHALF,IPZERO,
     &           NLAY,LAYMAX,IDOTOP)
C
C   Evaluates the perturbation field Q at
C   the nodes of the element IELC
C
C   Parameters
C    I/ IPONE
C       IPHALF : nodes to be perturbed
C       IPZERO
C
C    I/   NNIS : ref. intf. node : Q = unit normal to intf at NNIS
C    I/   IELC : Local elt number
C    O/   RPERT: perturbation field vector
C
    INCLUDE 'domain_common'
C
    DIMENSION  F(3,8),IDOTOP(9,15),NTR(3)
C
    DO 10 N = 1,3
    DO 10 I = 1,8
 10  F(N,I) = 0.d+00
C
    IF(NNIS.EQ.1.OR.NNIS.EQ.NTOTIS) THEN
     DO 20 I=1,3
 20   F(1,I)=0.d+00
C
    ELSE IF(NNIS.EQ.2.AND.NLAY.NE.LAYMAX) THEN
     FLEFT=COORDS(1,IELTOP(5,IELC))-COORDS(1,IELTOP(3,IELC))
     FMLEFT=1.d+00/FLEFT
     YLEFT=FMLEFT*(COORDS(1,IELTOP(11,IELC))-
     &             COORDS(1,IELTOP(3,IELC)))
     DO 30 I=1,3
 30   F(1,I)=DABS(YLEFT)
C
    ELSE IF(NNIS.EQ.NTOTIS-1.AND.NLAY.NE.LAYMAX) THEN
     FRIGHT=COORDS(1,IELTOP(5,IELC))-COORDS(1,IELTOP(3,IELC))
```

```
      FMRIG=-1.d+00/FRIGHT
      YRIGHT=FMRIG*(COORDS(1,IELTOP(11,IELC))-
     &               COORDS(1,IELTOP(5,IELC)))
      DO 40 I=1,3
 40    F(1,I)=DABS(YRIGHT)
C
     ELSE IF(NNIS.EQ.2.AND.NLAY.EQ.LAYMAX) THEN
      FLEFT=COORDS(1,IELTOP(5,IELC))-COORDS(1,IELTOP(3,IELC))
      FMLEFT=1.d+00/FLEFT
      YLEFT=FMLEFT*(COORDS(1,IELTOP(11,IELC))-
     &               COORDS(1,IELTOP(3,IELC)))
      F(1,1)=DABS(YLEFT)
      F(1,3)=0.d+00
C
     ELSE IF(NNIS.EQ.NTOTIS-1.AND.NLAY.EQ.LAYMAX) THEN
      FRIGHT=COORDS(1,IELTOP(5,IELC))-COORDS(1,IELTOP(3,IELC))
      FMRIG=-1.d+00/FRIGHT
      YRIGHT=FMRIG*(COORDS(1,IELTOP(11,IELC))-
     &               COORDS(1,IELTOP(5,IELC)))
      F(1,1)=DABS(YRIGHT)
      F(1,3)=0.d+00
C
     ELSE IF (NLAY.EQ.LAYMAX) THEN
      FDOWN=COORDS(2,IELTOP(15,IELC))-COORDS(2,IELTOP(11,IELC))
      FMDOWN=-1.d+00/FDOWN
      YDOWN=FMDOWN*(COORDS(2,IELTOP(17,IELC))-
     &               COORDS(2,IELTOP(3,IELC)))
      F(1,1)=1.d+00
      IF(IPHALF.NE.0) F(1,2)=DABS(YDOWN)
      F(1,3)=0.d+00
C
     ELSE
      DO 50 I=1,3
 50    F(1,I)=1.d+00
     END IF
C
     NTR(1) = IPONE
     NTR(2) = IPHALF
     NTR(3) = IPZERO
C
     DO 200 NN=1,NNELT
       DO 300 I=1,3
         IF(NTR(I).EQ.NN.AND.RPERT(NN,IELC,1).EQ.0) THEN
          IF(I.EQ.2.AND.NTR(2).EQ.0) GO TO 300
            NNLC=IELTOP(KP,IELC)
            RPERT(NN,IELC,1)=F(1,I)
         END IF
 300 CONTINUE
 200 CONTINUE
C
     RETURN
     END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *      SUBROUTINE PRESFN
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE PRESFN
C
C     Evaluates partial derivative of the shape function at the nodes
C     (SFDNOD(8,8,2)) and at the integration points (SFDITP(9,8,2)) .
C     Evaluates also the gauss integration weights WEIGHTS(9).
C
      INCLUDE 'domain_common'
      DIMENSION C(3)
C
      DO 50 I = 1,3
   50 C(I) = 0.d+00
C
      IR = IRDINT
      IT = ITYPE
C
      CALL WEICAL(IT,IR)
C
C     Derivatives of the shape functions at the nodes
C
      DO 200 I = 1,NNELT
         DO 100 M = 1,NDIM
  100       C(M) = DLISCO(0,I,M,IT,IR)
      DO 200 K = 1,NNELT
      DO 200 J = 1,NDIM
         SFDNOD(I,K,J) = PDSHFN(C(1),C(2),C(3),K,J,IT)
  200 CONTINUE
C
C     Derivatives of the shape functions and shape functions
c     at the integration points
C
      DO 400 I = 1,NINTP
         DO 300 M = 1,NDIM
  300       C(M) = DLISCO(1,I,M,IT,IR)
      DO 400 K = 1,NNELT
      SFNITP(I,K)=SHPFNC(C(1),C(2),C(3),K,IT)
      DO 400 J = 1,NDIM
         SFDITP(I,K,J) = PDSHFN(C(1),C(2),C(3),K,J,IT)
  400 CONTINUE
C
C
      IF (IPRINT.EQ.1) THEN
        WRITE(NOWN,1000)
        DO 500 I = 1,NNELT
        DO 500 K = 1,NNELT
        DO 500 J = 1,NDIM
           WRITE(NOWN,2000) I,K,J,SFDNOD(I,K,J)
  500 CONTINUE
        WRITE(NOWN,3000)
        DO 600 I = 1,NINTP
        DO 600 K = 1,NNELT
        DO 600 J = 1,NDIM
           WRITE(NOWN,2000) I,K,J,SFDITP(I,K,J)
  600 CONTINUE
      END IF
C
 1000 FORMAT(1H1,////,20X,'* * * SHAPE FUNCTION DERIVATIVE AT '
     & 'NODES * * *',//,
     & 1X,'NODE  SH.FN  J  D(SH.NF)/D(COORD.J)',//)
 2000 FORMAT(1X,I4,3X,I2,4X,I1,4X,F10.5)
 3000 FORMAT(1H1,////,20X,'* * * SHAPE FUNCTION DERIVATIVE AT '
     & 'IPS * * *',//,
     & 1X,'IT.P  SH.FN  J  D(SH.NF)/D(COORD.J)',//)
C
      RETURN
      END
C
C
```

```
C
C  ******************************************************************
C  *
C  *
C  *       SUBROUTINE PYRAMID
C  *
C  *
C  ******************************************************************
C
      SUBROUTINE PYRAMID(NNIS,IELC,IPONE,IPHALF,IPZERO,
     &           NLAY,LAYMAX,IDOTOP)
C
C     Evaluates the perturbation field Q at
C     the nodes of the element IELC
C
C     Parameters
C      I/ IPONE
C         IPHALF : nodes to be perturbed
C         IPZERO
C
C      I/   NNIS : ref. intf. node : Q = unit normal to intf at NNIS
C      I/   IELC : Local elt number
C      O/   RPERT: perturbation field vector
C
      INCLUDE 'domain_common'
C
      DIMENSION  F(3,8),C(2,3),IDOTOP(9,15),NTR(3)
C
      DO 10 N = 1,3
      DO 10 I = 1,8
 10   F(N,I) = 0.d+00
C
      ZERO = 0.d+00
      KP1 = (IPONE - 1) * 2 + 3
      IF(IPHALF.NE.0) THEN
        KP2 = (IPHALF - 1) * 2 + 3
        C(1,2) = COORDS(1,IELTOP(KP2,IELC))
        C(2,2) = COORDS(2,IELTOP(KP2,IELC))
      ENDIF
      KP3 = (IPZERO - 1) * 2 + 3
C
      C(1,1) = COORDS(1,IELTOP(KP1,IELC))
      C(1,3) = COORDS(1,IELTOP(KP3,IELC))
      C(2,1) = COORDS(2,IELTOP(KP1,IELC))
      C(2,3) = COORDS(2,IELTOP(KP3,IELC))
C
      FMLEFT = COORDS(2,NCORL)/COORDS(1,IFNN1)
      FMRIG = COORDS(2,NCORR)/COORDS(1,IFNNL)
C
      DO 100 I=1,3
C
       IF(C(1,I).LT.ZERO) THEN
C
        IF(I.EQ.2.AND.IPHALF.EQ.0) GO TO 100
C
        YLEFT = FMLEFT * C(1,I)
        IF(C(2,I).LE.YLEFT) THEN
         FNOM1A = COORDS(2,IFNN1)-COORDS(2,NCORL)
         FNOM1B = C(1,I) - COORDS(1,IFNN1)
         FNOM2A = COORDS(1,NCORL) - COORDS(1,IFNN1)
         FNOM2B = C(2,I) - COORDS(2,IFNN1)
         DEN1 = COORDS(1,IFNN1) * COORDS(2,NCORL)
         DEN2 = COORDS(1,NCORL) * COORDS(2,IFNN1)
         FNOM = FNOM1A * FNOM1B + FNOM2A * FNOM2B
         DEN = DEN1 - DEN2
         F(1,I) = FNOM/DEN
        ELSE
         FNOM1A = COORDS(2,NCORL)-COORDS(2,NCORR)
         FNOM1B = C(1,I) - COORDS(1,NCORL)
         FNOM2A = COORDS(1,NCORR) - COORDS(1,NCORL)
         FNOM2B = C(2,I) - COORDS(2,NCORL)
         DEN1 = COORDS(1,NCORL) * COORDS(2,NCORR)
         DEN2 = COORDS(1,NCORR) * COORDS(2,NCORL)
         FNOM = FNOM1A * FNOM1B + FNOM2A * FNOM2B
         DEN = DEN1 - DEN2
         F(1,I) = FNOM/DEN
        ENDIF
C
       ELSE IF(C(1,I).GE.ZERO) THEN
        YRIG = FMRIG * C(1,I)
        IF(C(2,I).LE.YRIG) THEN
         FNOM1A = COORDS(2,IFNNL)-COORDS(2,NCORR)
         FNOM1B = C(1,I) - COORDS(1,IFNNL)
         FNOM2A = COORDS(1,NCORR) - COORDS(1,IFNNL)
         FNOM2B = C(2,I) - COORDS(2,IFNNL)
         DEN1 = COORDS(1,IFNNL) * COORDS(2,NCORR)
         DEN2 = COORDS(1,NCORR) * COORDS(2,IFNNL)
         FNOM = FNOM1A * FNOM1B + FNOM2A * FNOM2B
         DEN = DEN1 - DEN2
         F(1,I) = FNOM/DEN
        ELSE
         FNOM1A = COORDS(2,NCORL)-COORDS(2,NCORR)
         FNOM1B = C(1,I) - COORDS(1,NCORL)
```

```
        FNOM2A = COORDS(1,NCORR) - COORDS(1,NCORL)
        FNOM2B = C(2,I) - COORDS(2,NCORL)
        DEN1 = COORDS(1,NCORL) * COORDS(2,NCORR)
        DEN2 = COORDS(1,NCORR) * COORDS(2,NCORL)
        FNOM = FNOM1A * FNOM1B + FNOM2A * FNOM2B
        DEN = DEN1 - DEN2
        F(1,I) = FNOM/DEN
       ENDIF
      ENDIF
C
 100 CONTINUE
C
      NTR(1) = IPONE
      NTR(2) = IPHALF
      NTR(3) = IPZERO
C
      DO 200 NN=1,NNELT
        DO 300 I=1,3
          IF(NTR(I).EQ.NN.AND.RPERT(NN,IELC,1).EQ.0) THEN
           IF(I.EQ.2.AND.NTR(2).EQ.0) GO TO 300
            NNLC=IELTOP(KP,IELC)
            RPERT(NN,IELC,1)=F(1,I)
          ENDIF
 300 CONTINUE
 200 CONTINUE
C
      RETURN
      END
C
```



```
C
C   ****************************************************************
C   *
C   *
C   *       SUBROUTINE RESETV
C   *
C   *
C   ****************************************************************
C
C
      SUBROUTINE RESETV(NOUT)
C
C     Reset to zero the memory used in the loop over the step/increment
C
      INCLUDE 'domain_common'
C
      WRITE(NOWD,1000) NKSTEP(NOUT), NKINCR(NOUT)
C
C
      DO 300 N = 1,2000
          IUGRAD(N) = 0
          ITGRAD(N) = 0
          DO 200 J=1,9
              WENIP(J,N)=0.d+00
          DO 200 K=1,9
              SIGMIP(J,K,N)=0.d+00
              UGRAD(J,K,N) = 0.d+00
              DO 100 I=1,3
 100            TGRAD(I,K,N)  =0.d+00
 200        CONTINUE
 300  CONTINUE
C
      DO 400 N=1,8000
          TEMP(N) = 0.d+00
          DO 410 I=1,3
 410        UNODE(I,N)=0.d+00
 400  CONTINUE
C
1000  FORMAT(1X,'STEP NO. ',I4,' INCREMENT NO. ',I4,/)
C
      RETURN
      END
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *       S U B R O U T I N E   R W P E R M
C   *
C   *
C   ****************************************************************
C
    SUBROUTINE RWPERM(ID,IMOVE,INEW,AM,AP,AN)
C
C   Change the order of the rows of AM if ZERO elements are on the
C   main diagonal.
C
C   Parameters:
C       I/   ID : matrix dimension
C       O/   IMOVE : row permutation code (=1 : row permuted )
C       O/   INEW : perm. record :INEW(J)= old # of new row #J
C       I/O  AM : square input/output matrix
C        /   AP : matrix used locally to keep record of non-zero
C                 pivots
C        /   AN : matrix used locally to store the new perm. matr.
C
    INCLUDE 'domain_common'
    DIMENSION AM(ID,ID),AP(ID,ID),AN(ID,ID),INEW(ID)
C
    DO 10 NR = 1,ID
       INEW(NR) = 0
    DO 10 NC = 1,ID
       AP(NR,NC) = 0.d+00
       AN(NR,NC) = 0.d+00
 10 CONTINUE
C
    IMOVE = 0
    PIVMIN = 0.00001
C
C   Check if matrix AM has zero elements on the main diagonal
C
    DO 50 NR = 1,ID
       IF(DABS(AM(NR,NR)).GE.PIVMIN) GO TO 50
       IMOVE = 1
 50 CONTINUE
C
    IF(IMOVE.EQ.0) GO TO 900
C
C   Permutation required
C
C   Prepare matrix AP with the position of non-zero pivots and
C   vector INEW with the # of rows which have non-zero pivots
C
    DO 100 NPIV = 1,ID
    DO 100 NR   = 1,ID
       IF(DABS(AM(NR,NPIV)).LT.PIVMIN) GO TO 100
        IPOS  = INEW(NPIV)+1
        INEW(NPIV) = IPOS
        AP(NPIV,IPOS) = DFLOAT(NR)
100 CONTINUE
C
C   Permutation
C
    NP = 0
C
200 CONTINUE
C
    DO 700 NPIV = 1,ID
C
     IF(INEW(NPIV).NE.NP) GO TO 700
C
     IF(NP.EQ.0) THEN
      IMOVE = -1
      WRITE(13,1000) NPIV
C
      DO 210 JJ = 1,ID
210     WRITE(13,*) (AM(JJ,KK),KK=1,ID)
      GO TO 900
     ENDIF
C
     PIVMAX = PIVMIN
C
     DO 300 NCP = 1,NP
       NR = INT(AP(NPIV,NCP))
       PIVOT = DABS(AM(NR,NPIV))
       IF(PIVOT.GE.PIVMAX) THEN
         PIVMAX = PIVOT
         NRCH   = NR
       ENDIF
300    CONTINUE
C
     INEW(NPIV) = -NRCH
     DO 350 NC = 1,ID
350    AN(NPIV,NC) = AM(NRCH,NC)
C
C     Packing of AP
```

```
C
      DO 600 NR = 1,ID
        IF(INEW(NR).LT.0) GO TO 600
        DO 500 NC = 1,ID
          IF(INT(AP(NR,NC)).EQ.NRCH) THEN
            INEW(NR) = INEW(NR) -1
            DO 400 NLC = NC,ID-1
400           AP(NR,NLC) = AP(NR,NLC+1)
            AP(NR,ID)  = 0.d+00
            GO TO 600
          ENDIF
500       CONTINUE
600     CONTINUE
C
      NP = NP-1
      GO TO 200
700   CONTINUE
C
      NP = NP+1
C
C     Check if all the pivot have been found
C
      INEG = 1
      DO 750 NR = 1, ID
750   IF(INEW(NR).GT.0) INEG = 0
C
      IF(INEG.EQ.1) GO TO 800
C
      IF(NP.LE.ID) GO TO 200
C
      IMOVE = -1
      WRITE(13,2000)
      WRITE(13,*) INEW
      WRITE(13,3000)
      WRITE(13,*) AM
      WRITE(13,4000)
      WRITE(13,*) AP
      WRITE(13,5000)
      WRITE(13,*) AN
C
800   CONTINUE
C
      DO 850 NR = 1,ID
        INEW(NR) = -INEW(NR)
      DO 850 NC = 1,ID
        AM(NR,NC) = AN(NR,NC)
850   CONTINUE
C
900   CONTINUE
C
1000  FORMAT(1H1,///,20X,'* * * ERROR IN SBR. RWPERM  * * *',//,
     &  20X,'NO PIVOT AVAILABLE FOR ROW N.',I4,//,
     &  20X,'INPUT MATRIX AM FOLLOWS ',//)
2000  FORMAT(1H1,///,20X,'* * * ERROR IN SBR. RWPERM  * * *',//,
     &  20X,'NP GREATER THAN ID.  NP = ',I4,' ID = ',I4,//,
     &  20X,'PERMUTATION RECORD VECTOR INEW  FOLLOWS ',//)
3000  FORMAT(1X,//,20X,'INPUT MATRIX AM FOLLOWS',//)
4000  FORMAT(1X,//,20X,'PIVOT MATRIX AP FOLLOWS',//)
5000  FORMAT(1X,//,20X,'OUTPUT MATRIX AN FOLLOWS',//)
      RETURN
      END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *      FUNCTION SHPFNC
C   *
C   *
C   ****************************************************************
C
    FUNCTION SHPFNC(G,H,R,K,I)
C
C   Evaluate  shape function N(k) at(G,H,R)
C
C   Parameters
C           G : 1st|  isoparametric coordinate of the location at
C           H : 2nd|  which the shape function must be
C           R : 3rd|  evaluated
C
C           K : number of the shape function
C           I : Element type
C                       8 : 2D - 8 nodes  isop. element
C
    INCLUDE 'domain_common'
C
    SHPFNC= 0.d+00
C
    IF(I.EQ.8) GO TO 10
      KERROR = KERROR+1
      WRITE (NOWC,1000) I
      GO TO 900
 10  CONTINUE
C
    GO TO (100,200,300,400,500,600,700,800) K
      KERROR = KERROR +1
      WRITE (NOWI,3000) K
      GO TO 900
C
 100  CONTINUE
    SHPFNC = -0.25d+00*(1-G)*(1-H)*(1+G+H)
    GO TO 900
C
 200  CONTINUE
    SHPFNC = -0.25d+00*(1+G)*(1-H)*(1-G+H)
    GO TO 900
C
 300  CONTINUE
    SHPFNC = -0.25d+00*(1+G)*(1+H)*(1-G-H)
    GO TO 900
C
 400  CONTINUE
    SHPFNC = -0.25d+00*(1-G)*(1+H)*(1+G-H)
    GO TO 900
C
 500  CONTINUE
    SHPFNC = 0.5d+00*(1-G)*(1+G)*(1-H)
    GO TO 900
C
 600  CONTINUE
    SHPFNC = 0.5d+00*(1+G)*(1-H)*(1+H)
    GO TO 900
C
 700  CONTINUE
    SHPFNC = 0.5d+00*(1-G)*(1+G)*(1+H)
    GO TO 900
C
 800  CONTINUE
    SHPFNC = 0.5d+00*(1-G)*(1+H)*(1-H)
C
C
 900  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.PDSHFN  * * *',//,
    &  10X,'ONLY ELEMENT-TYPE 8 IS IMPLEMENTED',//,
    &  10X,'ELEMENT-TYPE = ',I4)
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN FNC.PDSHFN  * * *',//,
    &  10X,'FOR THIS EL-TYPE THE NODE NUMBER MUST BE BETW. 1 AND 8',//,
    &  10X,'NODE NUMBER = ',I4)
C
    RETURN
    END
C
C
```

```
C
C   ****************************************************************
C   *
C   *
C   *       SUBROUTINE SIDNOD
C   *
C   *
C   ****************************************************************
C
C
      SUBROUTINE SIDNOD(IFLAG,N1,N2,N3,NSIDE)
C
C     Find the side to which N1,N2,N3 belong (if IFLAG = 1) or
C     the nodes N1,N2,N3 that belong to NSIDE (if IFLAG = 0)
C
C     Parameters
C          I/    L : identity flag: 0-> find node  1-> find side
C          I/O   N1,N2,N3 : node position (1-8)
C          I/O   NSIDE : side (-1/-4 +1/+4)(nodes must be consecutive)
C
C     NOTE!!!! ONLY FOR 2nd ORDER - 8 NODES - ISOPARAMETRIC 2D
ELEMENTS
C
================================================================
C
      INCLUDE 'domain_common'
C
      IF(IFLAG.EQ.0) GO TO 500
C
C     NSIDE has to be found
C
      NSIDE = 0
      GO TO (100,200,300,400)N1
      GO TO 600
C
  100 CONTINUE
      IF(N2.EQ.5.AND.N3.EQ.2) NSIDE = 1
      IF(N2.EQ.8.AND.N3.EQ.4) NSIDE =-4
      GO TO 600
C
  200 CONTINUE
      IF(N2.EQ.6.AND.N3.EQ.3) NSIDE = 2
      IF(N2.EQ.5.AND.N3.EQ.1) NSIDE =-1
      GO TO 600
C
  300 CONTINUE
      IF(N2.EQ.7.AND.N3.EQ.4) NSIDE = 3
      IF(N2.EQ.6.AND.N3.EQ.2) NSIDE =-2
      GO TO 600
C
  400 CONTINUE
      IF(N2.EQ.8.AND.N3.EQ.1) NSIDE = 4
      IF(N2.EQ.7.AND.N3.EQ.3) NSIDE =-3
      GO TO 600
C
  500 CONTINUE
C
C     N1,N2,N3 have to be found
C
      NS = NSIDE +5
C
      GO TO (510,520,530,540,550,560,570,580,590) NS
C
        KERROR = KERROR + 1
        WRITE(NOWC,1000) NSIDE
        GO TO 600
C
  510 CONTINUE
      N1 = 1
      N2 = 8
      N3 = 4
      GO TO 600
  520 CONTINUE
      N1 = 4
      N2 = 7
      N3 = 3
      GO TO 600
  530 CONTINUE
      N1 = 3
      N2 = 6
      N3 = 2
      GO TO 600
  540 CONTINUE
      N1 = 2
      N2 = 5
      N3 = 1
      GO TO 600
  550 CONTINUE
      KERROR = KERROR+1
      WRITE(NOWC,1000) NSIDE
      GO TO 600
  560 CONTINUE
      N1 = 1
```

```
      N2 = 5
      N3 = 2
      GO TO 600
  570 CONTINUE
      N1 = 2
      N2 = 6
      N3 = 3
      GO TO 600
  580 CONTINUE
      N1 = 3
      N2 = 7
      N3 = 4
      GO TO 600
  590 CONTINUE
      N1 = 4
      N2 = 8
      N3 = 1
C
  600 CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. SIDNOD  * * *',//,
     & 10X,'THE SIDE NUMBER MUST BE BETW.-4 AND -1 OR 1 AND 4',//,
     & 10X,'SIDE NUMBER = ',I4)
C
      RETURN
      END
C
C
C  ****************************************************************
C  *
C  *
C  *       S U B R O U T I N E   S O L V E R
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE SOLVER(ID,IMOVE,INEW,AU,AL,B,X)
C
C  Solve the systems AL*C = B and AU*X = C
C  If matrix AM has been permuted, the RHS is permuted too
C
C  Parameters
C       I/   ID : matrix/vector dimension
C       I/   IMOVE : row permutation code (=1 : row permuted )
C       I/   INEW : perm. record :INEW(J)= old # of new row #J
C       I/   AL : lower triang matrix AL(ID,ID)
C       I/   AU : upper triang.matrix AU(ID,ID)
C       I/   B  : RHS of the system AL*AU * X = B
C       O/    X  : unknown vector
C
      INCLUDE 'domain_common'
      DIMENSION AU(ID,ID), AL(ID,ID), B(ID), X(ID),INEW(ID)
C
C   Check if rows of AM have been permuted. If YES, permute B
C
      CALL CKPERM(ID,IMOVE,INEW,B,X)
C
      DO 10 NC = 1,ID
   10  X(NC) = 0.d+00
C
C   Evaluate X  -> AL*X = B
C
      DO 200 NR = 1,ID
         SUM = 0.d+00
       IF(NR.EQ.1) GO TO 200
         DO 100 NC = 1,NR-1
  100       SUM = SUM + X(NC) * AL(NR,NC)
  200  X(NR) = B(NR) -SUM
C
C   Copy X in B
C
      DO 250 I = 1,ID
      B(I) = X(I)
```

```
 250  X(I) = 0.d+00

C    Evaluate X  -> AU*X = B
C
     DO 400 NR = ID,1,-1
          SUM = 0.d+00
        IF (NR.EQ.ID) GO TO 400
          DO 300 NC = NR+1,ID
 300        SUM = SUM + X(NC) * AU(NR,NC)
 400  X(NR) = (B(NR)-SUM)/AU(NR,NR)
C
C
     RETURN
     END
C
C
```



```
C
C    ****************************************************************
C    *
C    *
C    *       S U B R O U T I N E   S T R D I S
C    *
C    *
C    ****************************************************************
C
     SUBROUTINE STRDIS(NDOM,NOUT)
C
C    Obtaines the through-thickness stress-distribution
C    of the specimen
C
     INCLUDE 'domain_common'
     DIMENSION NELIF(100),VARNOD(9,8000),SNODE(200)
C
C    Input the element numbers along the symmetry line of the
C    specimen
C
     NELC = 0
C
     DO 100 N=1,NTOTTE
       NNLC = NTEMP(N)
       DO 100 I=1,NTELT
         IF(IELTOP(11,I).EQ.NNLC) THEN
           NELC = NELC + 1
           NELIF(NELC) = IELTOP(1,I)
         END IF
  100 CONTINUE
C
     NELDT = NELC
C
     NCHK = (NTOTTE-1)/2
     IF(NELDT.EQ.NCHK) GO TO 300
      KERROR = KERROR + 1
      WRITE(NOWC,1000) NELDT, NCHK
  300 CONTINUE
C
     CALL F8NOIN(NOUT,11,VARNOD,FILL1,FILL2)
C
     DO 310 N = 1,NTOTTE
       NNLC = NTEMP(N)
  310    SNODE(N) = VARNOD(2,NNLC)
C
     IF(NDOM.EQ.1) THEN
```

```
C
      IF(NDOM.EQ.1.AND.NOUT.EQ.1) THEN
        WRITE(NOWP(7),4000)
        WRITE(NOWP(8),6000)
      END IF
      WRITE(NOWP(7),5000)
      WRITE(NOWP(8),5000)
      WRITE(NOWS,3000)
C
      DO 200 NNTE=1,NTOTTE
        N1 = NTEMP(1)
        NNLC = NTEMP(NNTE)
        XNORM = ((COORDS(1,NNLC) + DABS(COORDS(1,N1)))/WIDTH)
        WRITE(NOWS,900) NTEMP(NNTE),COORDS(1,NNLC),
     &                  SNODE(NNTE)
        WRITE(NOWP(7),2000) COORDS(1,NNLC),SNODE(NNTE)
              IF(ITEMP.EQ.1) THEN
          DELT = TINIT - TAMB
          STNOR = (SNODE(NNTE)*(1.d+00-POISS))/(COTHER*YOUNG*DELT)
          WRITE(NOWP(8),2000) XNORM,STNOR
              END IF
  200 CONTINUE
C
      END IF
C
  900 FORMAT(I5,2X,2(E17.4,3X))
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. STRDIS  * * *',//,
     & 10X,'INCORRECT NUMBER OF ELEMENTS ALONG SYMMETRY LINE',//,
     & 10X,'FOUND: ',I4,'CORRECT: ',I4)
 2000 FORMAT(2(E17.4,3X))
 3000 FORMAT(1H1,///,20X,'* * * S T R E S S  '
     &       'D I S T R I B U T I O N * * *',//,
     & 1X,'NNLC  COORDS   TEMP  ',/)
 4000 FORMAT(1X,'x-coordinate',/,1X,'stress',/,'type 1',/,
     &      'end')
 5000 FORMAT(1X,'1.0E32   101')
 6000 FORMAT(1X,'x/L',/,1X,'(1 - @n)@sT @aE(Tinit-Tamb)'
     & ,/,'type 1',/,'xmin 0.',/,'xmax 1.',/,
     &      'end')
C
      RETURN
      END
C
C
C
C     ****************************************************************
C     *
C     *
C     *       S U B R O U T I N E   T E M P D I S
C     *
C     *
C     ****************************************************************
C
      SUBROUTINE TEMPDIS(NDOM,NOUT,TIME)
C
C     Obtaines the through-thickness temperature-distribution
C     of the specimen
C
      INCLUDE 'domain_common'
      DIMENSION VARNOD(9,8000),ATEMP(200), AXCO(200)
C
C     Input the temperature
C
      CALL F8NOIN(NOUT,201,VARNOD,TIME,TTIME)
C
      DO 110 NNL = 1,NTNDO
        NNLC = IDON(NNL)
  110 TEMP(NNLC) = VARNOD(1,NNLC)
C
      DO 115 NNL = 1,NTOTTE
        NNLC = NTEMP(NNL)
  115 TEMP(NNLC) = VARNOD(1,NNLC)
C
      TIM(NOUT)=TIME
      FOU(NOUT) = DIFFU * TIME/WIDTH**2
      HTIM(NOUT)=TTIME
C
C     Analytical temperature distribution
C
C     CALL ANALYT(TIME,ATEMP,AXCO,NDOM,NOUT)
C
      IF(NOUT.EQ.1.AND.NDOM.EQ.1) THEN
        WRITE(NOWP(3),4000)
        WRITE(NOWP(6),6000)
      END IF
      WRITE(NOWP(3),5000)
      WRITE(NOWS,3000)
C
C     WRITE(NOWP(6),8000)
C     DO 140 NNTE=1,NTOTTE
```

```
C       NNLC = NTEMP(NNTE)
C 140   WRITE(NOWP(6),2000) AXCO(NNLC),ATEMP(NNLC)
C 999  CONTINUE
C
      WRITE(NOWP(6),5000)
C
      DO 200 NNTE=1,NTOTTE
        N1 = NTEMP(1)
        NNLC = NTEMP(NNTE)
        TTHETA = (TEMP(NNLC)-TAMB)/(TINIT-TAMB)
        XNORM = (COORDS(1,NNLC) + DABS(COORDS(1,N1)))/WIDTH
        WRITE(NOWS,1000) NTEMP(NNTE),COORDS(1,NNLC),
     &              TEMP(NNLC)
        IF(NDOM.EQ.1.AND.NNLC.EQ.NCRLC) THEN
          TDIFF = ATEMP(NCRLC)  - TTHETA
C           TPERC = 100.d+00*DABS(TDIFF)/ATEMP(NCRLC)
C           WRITE(NOWM,2000) FOU(NOUT), TPERC
        END IF
        WRITE(NOWP(6),2000) XNORM,TTHETA
 200    WRITE(NOWP(3),2000) COORDS(1,NNLC),TEMP(NNLC)
C
C
 1000 FORMAT(I5,2X,F9.4,2X,F9.4)
 2000 FORMAT(2(E17.4,3X))
 3000 FORMAT(1H1,///,20X,'* * * T E M P E R A T U R E '
     &       'D I S T R I B U T I O N * * *',//,
     & 1X,'NNLC  COORDS    TEMP ',/)
 4000 FORMAT(1X,'x-coordinate',/,1X,'temperature',/,'type 1',/,
     &       'end')
 5000 FORMAT(1X,'1.0E32   101')
 6000 FORMAT(1X,'x/L',/,1X,'(T - Tamb)(Tinit - Tamb)',/,
     &       'type 1',/,
     &       'xmin 0.',/,'xmax 1.',/,'ymin 0.',/,'ymax 1.'
     &       ,/,'end')
 7000 FORMAT(2(F7.3))
 8000 FORMAT(1X,'1.0E32   102')
C
      RETURN
      END
C
C
```

```
C
C  ****************************************************************
C  *
C  *
C  *       S U B R O U T I N E   T E M P G R
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE TEMPGR(IELC)
C
C  Evaluates the gradient of the perturbation  field Q at
C  the integration points of the element IELC
C
C  Parameters
C    I/   IELC : Local elt number
C    O/   TGRAD: Gradient of T at the integration points
C
C          TGRAD(1,J,IELC) :  dT/dX1 at integration point J
C          TGRAD(2,J,IELC) :  dT/dX2 at integration point J
C          TGRAD(3,J,IELC) :  dT/dX3 at integration point J
C
C
      INCLUDE 'domain_common'
C
      DIMENSION  F(3,8),OU(9,9)

      DO 10 N = 1,3
      DO 10 I = 1,8
 10   F(N,I) = 0.d+00
C
      DO 200 NN = 1,NNELT
C
        KP = (NN-1)*2+3
        NNLC=IELTOP(KP,IELC)
        F(1,NN) = TEMP(NNLC)
C
 200  CONTINUE
C
      CALL GRADIE(IELC,0,F,OU)
      IF (KERROR.NE.0) GO TO 400
C
      DO 300 J = 1,9
      DO 300 I = 1,3
 300  TGRAD(I,J,IELC) = OU(I,J)
C
```

```
      ITGRAD(IELC) = 1
C
 400  CONTINUE
C
      RETURN
      END
C
C
```



```
C
C  ****************************************************************
C  *
C  *
C  *       SUBROUTINE TEMPINP
C  *
C  *
C  ****************************************************************
C
      SUBROUTINE TEMPINP(IDONE)
C
C  Input the nodes for the temperature distribution of
C  the specimen
C
      INCLUDE 'domain_common'
      DIMENSION NINP(15),IDONE(20000)
C
      NNTE =0
      DO 100 NS=1,NSETT
        IN = NTDIS(NS,1)
        DO 110 K=2,16
 110      NINP(K-1) = NTDIS(NS,K)
        IF(IN.NE.0)THEN
          NF = NINP(1)
          NL = NINP(2)
          NT = 1+(NL-NF)/IN
          DO 30 NN = 1,NT
            NNTE = NNTE+1
            NNAB = NF + (NN-1)*IN
            NNLC = IDONE(NNAB)
            NTEMP(NNTE) = NNLC
 30       CONTINUE
        ELSE
          DO 60 NN=1,15
            IF(NINP(NN).EQ.0) GO TO 60
            NNAB = NINP(NN)
            NNTE = NNTE +1
            NNLC = IDONE(NNAB)
            NTEMP(NNTE) = NNLC
 60       CONTINUE
        ENDIF
 100  CONTINUE
      NTOTTE = NNTE
C
C  Local crack-tip node number
C
```

```
      NCRLC = IDONE(NCRACK)
C
      WRITE(NOWG,4000)
      DO 300 NNTE=1,NTOTTE
  300   WRITE(NOWG,5000) NNTE, NTEMP(NNTE), NABAQ(NTEMP(NNTE))
C
      IF (NTOTTE.LE.200) GO TO 400
        KERROR = KERROR +1
        WRITE(NOWC,3000) NTOTTE
  400 CONTINUE
C
 3000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. TEMP  * * *',//,
     & 10X,'MAX NUMBER OF INPUT SET NODES  EXCEEDED '//,
     & 10X,'TOTAL NUMBER OF I.S. NODES = ',I4,' (MAX: 200)')
 4000 FORMAT(1H1,///,20X,'* * * N Ns  F O R  '
     &      'T E M P E R A T U R E  '
     &      'D I S T R I B U T I O N * * *',//,
     & 1X,'NNTE  NNLC  NNAB  ',/)
 5000 FORMAT(3(2X,I5,2X))
C
      RETURN
      END
C
C
C  ********************************************************************
C  *
C  *
C  *      SUBROUTINE VARINP
C  *
C  *
C  ********************************************************************
C
C
      SUBROUTINE VARINP(NOUT)
C
C   Parameter
C       I/ NOUT  : Serial number of the required step/increment
C
C
C   Reads Elastic energy and Stress tensor at the integration points
C   Reads also the displacement field and the creep strains at the nodes
C
      INCLUDE 'domain_common'
      DIMENSION VARELT(9,9,2000),VARNOD(9,8000)
C
C   Input the elastic energy
C
      CALL F8IPIN(NOUT,14,VARELT)
C
      DO 110 IE = 1,NTELDO
      IEL = IDOEL(IE)
      DO 110 IP = 1,NINTP
  110  WENIP(IP,IEL) = VARELT(1,IP,IEL)
C
C   Input the strain tensor
C
C     CALL F8IPIN(NOUT,21,VARELT)
C
C     DO 100 IE = 1,NTELDO
C     IEL = IDOEL(IE)
C     DO 100 IP = 1,NINTP
C     DO 100 NC = 1,9
C 100  EPSIP(NC,IP,IEL) = VARELT(NC,IP,IEL)
C
C   Input the stress tensor
C
      CALL F8IPIN(NOUT,11,VARELT)
C
      DO 200 IE = 1,NTELDO
```

```
      IEL = IDOEL(IE)
      DO 200 IP = 1,NINTP
      DO 200 NC = 1,9
  200 SIGMIP(NC,IP,IEL) = VARELT(NC,IP,IEL)
C
C     Input the displacement at the nodes
C
      CALL F8NOIN(NOUT,101,VARNOD,FILL1,FILL2)
C
      DO 310 NNL = 1,NTNDO
      NNLC = IDON(NNL)
      DO 310 ND = 1,NDIM
  310 UNODE(ND,NNLC) = VARNOD(ND,NNLC)
C
C     Input the J-integral
C
      CALL F8JINT(NOUT,1991)
C
      IF(ITEMP.EQ.0) THEN
C
C       Input the far-field stress
C
        CALL F8NOIN(NOUT,104,VARNOD,FILL1,FILL2)
C
      END IF
C
      DO 340 NEL = 1,NTELDO
        IELC = IDOEL(NEL)
        IF(ITEMP.EQ.1.AND.ITGRAD(IELC).NE.1) CALL TEMPGR(IELC)
        IF(IUGRAD(IELC).NE.1) CALL GRADIU(IELC)
  340 CONTINUE
C
      RETURN
      END
C
C
C     ******************************************************************
C     *
C     *
C     *       SUBROUTINE VAROUT
C     *
C     *
C     ******************************************************************
C
C
      SUBROUTINE VAROUT(NOUT)
C
C     Parameter
C         I/ NOUT  : Serial number of the required step/increment
C         I/ NTELDO: Total number of elts in the domain
C
C
C     Prints elastic energy and stress tensor at the integration points
C     Prints also the displacement field at the nodes
C
      INCLUDE 'domain_common'
C
C     Print the elastic energy
C
      WRITE(NOWS,6000)
      DO 770 IE=1,NTELDO
        IELC = IDOEL(IE)
        DO 770 IP=1,NINTP
  770 WRITE(NOWS,7000) IELC,IELTOP(1,IELC),IP,WENIP(IP,IELC)
C
      IF(ITEMP.EQ.1) THEN
C
C       Print the temperature
C
        WRITE(NOWS,8000)
        DO 780 IE=1,NTELDO
        IELC=IDOEL(IE)
        DO 780 NN=1,NNELT
          KP = (NN-1)*2 + 3
          NNLC=IELTOP(KP,IELC)
          WRITE(NOWS,7000) IELC,NNLC,NABAQ(NNLC),TEMP(NNLC)
  780 CONTINUE
C
      ENDIF
C
C     Print the strain tensor
```

```
C
C     WRITE(NOWS,1000)
C     DO 700 IE=1,NTELDO
C       IELC = IDOEL(IE)
C       DO 700 IP=1,NINTP
C  700 WRITE(NOWS,3000) IELC,IELTOP(1,IELC),IP,(EPSIP(I,IP,IELC),I=1,4)
C
C    Print the stress tensor
C
      WRITE(NOWS,2000)
      DO 800 IE=1,NTELDO
        IELC = IDOEL(IE)
        DO 800 IP=1,NINTP
  800 WRITE(NOWS,3000) IELC,IELTOP(1,IELC),IP,(SIGMIP(I,IP,IELC),I=1,4)
C
C    Print the displacement at the nodes
C
      WRITE(NOWS,4000)
      DO 890 IE=1,NTELDO
        IELC=IDOEL(IE)
        DO 900 NN=1,NNELT
          KP = (NN-1)*2 + 3
          NNLC=IELTOP(KP,IELC)
          WRITE(NOWS,5000) IELC,NNLC,NABAQ(NNLC),(UNODE(I,NNLC),I=1,2)
  900 CONTINUE
  890 CONTINUE
C
      IF(NOUT.GT.1) GO TO 999
C
C    Print the nodal coordinates
C
      WRITE(NOWS,9100)
      DO 910 IE=1,NTELDO
        IELC=IDOEL(IE)
        DO 920 NN=1,NNELT
          KP = (NN-1)*2 + 3
          NNLC=IELTOP(KP,IELC)
          WRITE(NOWS,9200) IELC,NNLC,NABAQ(NNLC),
     &          (COORDS(I,NNLC),I=1,3)
  920 CONTINUE
  910 CONTINUE
C
  999 CONTINUE
C
      IF(ITEMP.EQ.0) THEN
C
C      Print the reaction force at node 10000
C
        WRITE(NOWS,9000) RFNOD(1),RFNOD(2)
C
      END IF
C
C
C    Print the displacement gradient
C
      WRITE(NOWS,7100)
      DO 300 NEL = 1,NTELDO
        IELC = IDOEL(NEL)
        DO 300 J = 1,9
  300  WRITE(NOWS,7200) IELC, J, (UGRAD(I,J,IELC),I=1,9)
C
      IF(ITEMP.EQ.1) THEN
C
C    Print the temperature gradient
C
      WRITE(NOWS,7400)
      DO 360 NEL = 1,NTELDO
        IELC = IDOEL(NEL)
        DO 360 J = 1,9
  360    WRITE(NOWS,7500)IELC,J,(TGRAD(I,J,IELC),I=1,3)
C
      END IF
C
C
 1000 FORMAT(1H1,////,40X,'* * *  STRAINS AT IP * * *',//
     &    ' IELC IEAB  IP     E11          E22          E33
     &          E12',/)
 2000 FORMAT(1H1,////,40X,'* * *  STRESSES AT IP * * *',//
     &    ' IELC IEAB  IP     S11          S22          S33
     &          S12',/)
 3000 FORMAT(3(1X,I4),4(2X,E12.4,2X))
 4000 FORMAT(1H1,////,40X,'* * *  DISPLACEMENTS AT NODES * * *',//
     &    ' IELC  NNLC  NNAB     U1           U2',/)
 5000 FORMAT(3(1X,I4),2(2X,E12.4,2X))
 6000 FORMAT(1H1,////,40X,'* * *  STRAIN ENERGY DENSITY AT IP * * *',//
     &    ' IELC  IEAB  IP     WENIP',/)
 7000 FORMAT(3(1X,I4),2X,E12.4)
 7100 FORMAT(1H1,///,40X,'* * * U G R A D * * *',//,
     & 1X,'IELC  IP   U11   U22   U33   U12   U13   U23   '
     & 'U21   U31   U32',/)
 7200 FORMAT(2(1X,I4),9(1X,E12.3))
 7400 FORMAT(1H1,///,40X,'* * * T G R A D * * *',//,
     & 1X,'IELC  IP   T1    T2    T3',/)
 7500 FORMAT(2(1X,I4),3(E12.3))
```

```
 8000 FORMAT(1H1,////,40X,'* * *  TEMPERATURE AT NODES * * *',//
   &    ' IELC  NNLC  NNAB    TEMP',/)
 9000 FORMAT(1H1,////,20X,'* * *  REACTION FORCE AT NODE 10000 * * *',//
   &    'RF in x-direction: ',E11.4,//,
   &    'RF in y-direction: ',E11.4)
 9100 FORMAT(1H1,////,40X,'* * *  C O O R D I N A T E S * * *',//
   & 'IELC NNLC NNAB   X1       X2       X3',/)
 9200 FORMAT (3(1X,I4),3(2X,F6.3,2X))
      RETURN
      END
C
C
C  ********************************************************************
C  *
C  *
C  *       S U B R O U T I N E   W E I C A L
C  *
C  *
C  ********************************************************************
C
C
      SUBROUTINE WEICAL(IT,IR)
C
C   Evaluates the gauss weights at integration points WEIGHT(9)
C
C   Parameters
C       I/     I : Element type
C                      8 : 2D - 8 nodes  isop. element
C       I/     IR: Reduced integration flag
C                      0 : full integration (9 i.p.)
C                      1 : reduced integration (4 i.p.)
C
C   NOTE!!!! ONLY FOR 2nd ORDER - 8 NODES - ISOPARAMETRIC 2D
ELEMENTS
C
=====================================================================
C

      INCLUDE 'domain_common'
C
      FIV = .555555555555556d+00
      EIG = .888888888888889d+00
C
      IF ( IT.EQ.8) GO TO 100
        KERROR = KERROR + 1
        WRITE(NOWC,1000) IT
        GO TO 900
  100 CONTINUE
C
      IF(IR.EQ.1) GO TO 200
C
C    Full integration
C
      WEIGHT(1) = FIV*FIV
      WEIGHT(2) = EIG*FIV
      WEIGHT(3) = FIV*FIV
      WEIGHT(4) = EIG*FIV
```

```
      WEIGHT(5) = EIG*EIG
      WEIGHT(6) = EIG*FIV
      WEIGHT(7) = FIV*FIV
      WEIGHT(8) = EIG*FIV
      WEIGHT(9) = FIV*FIV
C
      GO TO 900
C
 200  CONTINUE
C
C     Reduced integration
C
      WEIGHT(1) = 1.d+00
      WEIGHT(2) = 1.d+00
      WEIGHT(3) = 1.d+00
      WEIGHT(4) = 1.d+00
      WEIGHT(5) = 0.d+00
      WEIGHT(6) = 0.d+00
      WEIGHT(7) = 0.d+00
      WEIGHT(8) = 0.d+00
      WEIGHT(9) = 0.d+00
C
 900  CONTINUE
C
 1000 FORMAT(1H1,///,20X,'* * * ERROR IN SBR. WEICAL  * * *',//,
     & 10X,'ONLY ELEMENT-TYPE 8 IS IMPLEMENTED',//,
     & 10X,'ELEMENT-TYPE = ',I4)
C
      RETURN
      END
C
C
C  *****************************************************************
C  *
C  *
C  *       SUBROUTINE ZERO
C  *
C  *
C  *****************************************************************
C
C
      SUBROUTINE ZERO(NDOM)
C
C     Reset to zero the memory used in the loop over the domain
C
      INCLUDE 'domain_common'
C
      WRITE(NOWD,1000) NDOM
C
      DO 300 N = 1,2000
         IECNT(N)=0
         IQGRAD(N)=0
         DO 200 J=1,17
 200        IELTOP(J,N)=0
 300  CONTINUE
C
      DO 400 N=1,8000
         INCNT(N)=0
         NABAQ(N)=0
 400  CONTINUE
C
      DO 500 N=1,200
         DO 510 I=1,5
         DO 510 J=1,2
         LELCON(I,J,N)=0
 510  CONTINUE
 500  CONTINUE
C
      DO 600 N=1,400
         DO 610 I=1,5
 610        LNOCON(I,N)=0
         DO 630 I=1,21
         DO 630 J=1,2
 630        NCONN(I,J,N)=0
 600  CONTINUE
C
      DO 700 N=1,3200
```

```
 700       IDON(N)=0
C
    DO 750 N=1,800
 750       IDOEL(N)=0
C
    DO 800 N=1,2000
         DO 800 J=1,9
         DO 800 I=1,9
 800       QGRAD(I,J,N) = 0.d+00
C
    DO 900 N=1,2000
         DO 920 I=1,8
         DO 920 J=1,3
 920       RPERT(I,N,J)=0.d+00
 900  CONTINUE
C
1000  FORMAT(1X,'DOMAIN NO. ',I4,/)
C
    RETURN
    END
```