DTIC FILE COPY

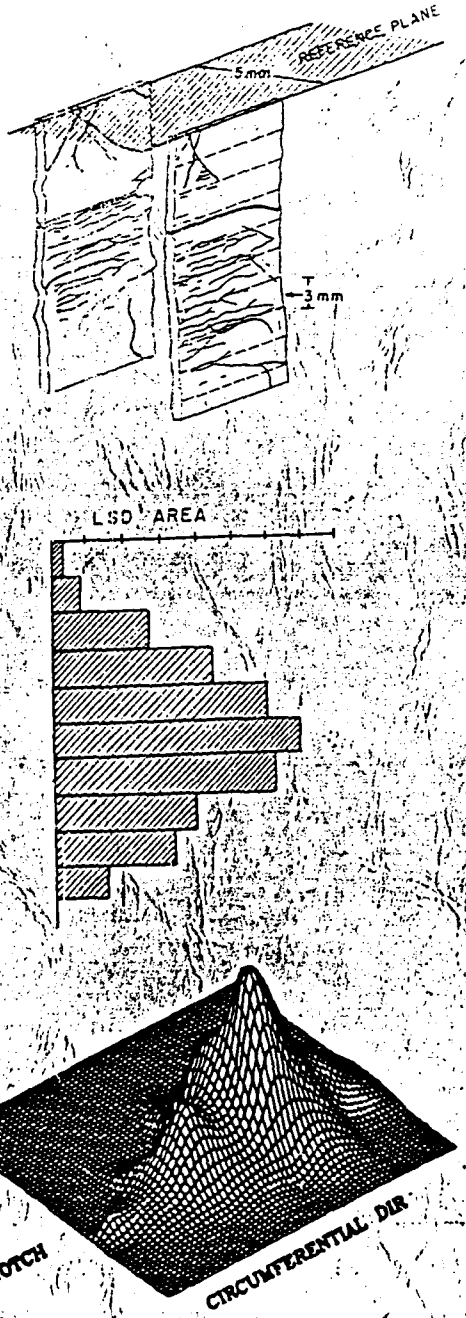
ARO 22588.2-GS

(2)

AD-A204 053

A FUNDAMENTAL INVESTIGATION OF CRACK AND SURROUNDING DAMAGE IN STIFF CLAYS

FINAL TECHNICAL REPORT SFRAC-2



By

A. Saada, A. Chudnovsky and A. Lesser

September 1988

DTIC ELECTE NOV 07 1988 S D

U.S. ARMY RESEARCH OFFICE
Contract No. DAAG29-85-K-0221

GEOTECHNICAL ENGINEERING LABORATORIES
CASE INSTITUTE OF TECHNOLOGY
CASE WESTERN RESERVE UNIVERSITY

Approved for Public Release
Distribution Unlimited

88 11 07 055

MASTER COPY - FOR REPRODUCTION PURPOSES

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ADA204053

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: ARO 22588.2-GS	2. GOVT ACCESSION NO.: N/A
3. RECIPIENT'S CATALOG NUMBER: N/A	
4. TITLE (and Subtitle): A FUNDAMENTAL INVESTIGATION OF CRACK AND SURROUNDING DAMAGE IN STIFF CLAYS	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report 8/12/85 - 8/12/88
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): Prof. Adel S. Saada; Prof. Alexander Chudnovsky	8. CONTRACT OR GRANT NUMBER(s): DAAG29-85-K-0221
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Dept. of Civil Engineering, Case Western Reserve Universi., Cleveland, Ohio 44106	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: N/A
11. CONTROLLING OFFICE NAME AND ADDRESS: U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709	12. REPORT DATE: September 1988
	13. NUMBER OF PAGES:
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA fr block

18. SUPPLEMENTARY NOTES

The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Stiff Clay,
~~Fracture,~~
Damage Propagation,
Strip Densification,
Stress Analysis,
Energy Balance,
Constitutive Model, (Soil) +

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The geotechnical engineering profession is quite familiar with stiff fissured and jointed clays where discontinuities are found on a macroscopic scale. However, there has not been, until recently, a systematic investigation of the crack propagation phenomena involved in the failure of such clays.

This study presents the first phase of an investigation whose aim is to

(over)

understand, describe and quantify the mechanisms involved in crack propagation. The experimental investigation indicates that in a mode II type of fracture a well defined damage zone propagates in front of the crack. In this zone the structure of the clay is shown to be substantially altered. An image analyser is used to quantify the damage which appears as densified strips. Such strips result from the coalescence of clusters of clay particles which are uniformly distributed in the virgin material.

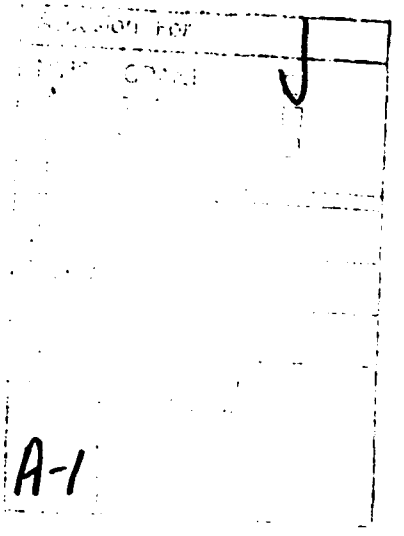
Fracture is treated as an irreversible process within the framework of thermodynamics. A damage parameter is introduced and an equation of state is written in terms of entropy production. A generalized flux is identified as the centroidal movement of the damage zone, and the active part of the thermodynamic force is compared to the conventional J integral. Stress and energy analyses for the damage zone are performed. A constitutive model for describing the fracture is proposed. Keywords:

FORWARD

The study reported herein was made by the faculty and students of Case Western Reserve University (CWRU) during the period August 12, 1985 through August 12, 1988. The investigation was sponsored by the U.S. Army Research Office, under Research Agreement No. DAAG29-85-K-0221 and the supervision of Dr. Steven J. Mock of the Geosciences Division. The Co-Principal Investigators were Professors Adel S. Saada and Alexander Chudnovsky.

Allan Lesser and Boris Kunin participarted in this research, the first as a graduate student and a Ph.D. candidate and the second as a staff researcher. The enclosed report is the doctoral thesis that Mr. Lesser defended at Case in August of 1988.





Theoretical and Experimental Studies of Cooperative Fracture in Overconsolidated Clays

by

Alan J. Lesser

Submitted in partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

Dissertation Advisors:

Dr. Adel Saada
and
Dr. Alexander Chudnovsky

Department of Civil Engineering

CASE WESTERN RESERVE UNIVERSITY

January 1989

Theoretical and Experimental Studies of Cooperative Fracture in Overconsolidated Clays

Abstract

by

Alan J. Lesser

The geotechnical engineering profession is quite familiar with stiff fissured and jointed clays, where discontinuities are found on a macroscopic scale. Until recently, however, there has not been a systematic investigation of the crack propagation phenomena involved in the failure of such clays. This thesis presents a fundamental investigation in an effort to understand, describe and quantify the mechanisms involved in the process; the ultimate goal being to develop a constitutive model which depicts the fracture process in these clays.

The study concentrates on a Mode II type of fracture that the author has been able to induce in a specimen with the configuration of a notched, thin, long, hollow cylinder subjected to a hydrostatic stress and pure torsion. Within the stress concentration field, a damage zone (process zone) is identified. The kaolinite clay is quantifiably characterized in both its undamaged and damaged states. Micromechanisms responsible for macroscopic irreversible deformation are identified. A comprehensive stress and energy analysis is performed in the

vicinity of the damage zone. And finally, a methodology for evaluating a constitutive law is introduced and applied to the case of overconsolidated clays. The methodology describes the fracture process through the laws of thermodynamics of irreversible processes in an elastic continuum with damage.

TABLE OF CONTENTS

ABSTRACT v

TABLE OF CONTENTS vii

LIST OF FIGURES xi

LIST OF TABLES xvi

CHAPTER 1 INTRODUCTION

1.1 Progressive Failure in Overconsolidated Clays 1
1.2 Previous Work 8
1.3 The Concept of Cooperative Fracture and its Propagation 9
1.4 Thesis Outline 15
References 16

CHAPTER 2 EXPERIMENTAL PROCEDURES

2.1 Introduction 27
2.2 Material Properties 28
2.3 Initial Consolidation 28
2.4 Cylinder Preparation 29
2.5 Final Consolidation 30
2.6 The Fatigue Test 32
2.7 Impregnation 32
2.8 Cylinder Sectioning 34
2.9 Section Polishing 35
2.10 General Use of the Image Analyzer 36
References 38

CHAPTER 3 MICROMECHANISMS OF DEFORMATION IN FRACTURE OF OVERCONSOLIDATED CLAYS

3.1 Introduction 48
3.2 External Observations 48
3.3 Internal Observations 50
3.3.1 Undamaged Phase 50
3.3.2 Damaged Phase 52
3.4 Other Observations 54
3.5 Elements of Deformation and Fracture 55
References 56

CHAPTER 4 KAOLINITE CLUSTER CHARACTERIZATION

4.1 Introduction 66
4.2 The Representative Volume 66
4.2.1 Concept of a Representative Volume 68
4.2.2 General Evaluation of a Representative Volume 70
4.2.3 Representative Volume for Stiff Clays 72
4.3 Reconstruction of the Cluster Spatial Distributions 75
4.3.1 The Reconstruction Model 77
4.3.1.1 3–D Notation 78
4.3.1.2 2–D Notation 79
4.3.1.3 Size Distribution Relations 80
4.3.2 Particle Distributions 84
4.3.2.1 Error Analysis 85
4.3.2.2 Cluster Volumetric Concentration (Saltykov's Approximation) 86
References 87

CHAPTER 5 KAOLINITE DAMAGE CHARACTERIZATION

5.1 Introduction 101
5.2 LSD Characterization 102
5.3 LSD Evolution 104
5.4 Integral Description of LSD Zone Evolution 105
5.4.1 Resistive Moment Evaluation 107

CHAPTER 6 THEORETICAL CONSIDERATIONS

6.1 Introduction 120
6.2 Thermodynamics of Fracture in an Elastic Medium 121
6.3 Damage Zone Propagation 124
6.4 Impellent Forces 127
6.5 Energetic Forces 129
6.6 Discussion 131
References 132

CHAPTER 7 STRESS AND ENERGY ANALYSIS OF THE CRACK – DAMAGE INTERACTION

7.1 Introduction 133
7.2 Solution via Superposition 134
7.3 The Crack–Microcrack Interaction Problem 136
7.4 Semi–Empirical Approach via Second Greens Tensor 137
7.5 Semi–Empirical Approach via Complex Potentials 139
7.5.1 Remote and Nominal Potentials 140
7.5.2 Complex Potentials for a Crack and a Dislocation Dipole 141
7.6 Stress Field Evaluation 145
7.7 Stress Intensity Factor Analysis 149
7.7.1 Stress Intensity Factor Analysis for Continuously Distributed Dislocations 154
7.8 Energy Release Rate Analysis 157
7.9 Summary 159
References 160

CHAPTER 8 A CONSTITUTIVE MODEL

8.1 Introduction 195
8.2 Experimental Results and Observations 196
8.3 Phenomenological Relationship and Specific Energy Evaluation 197
References 200

CHAPTER 9 FINAL DISCUSSION

9.1 Summary 205
9.2 Conclusions 209

APPENDIX A	Program Listings	211
APPENDIX B	Representative Volume Data Base	307
APPENDIX C	LSD Histogram Plots	309
APPENDIX D	LSD Evolution Plots for Cylinder 5–6	324

List of Figures

Figure 1.1	Basic types of mass movements constituting landslides.
Figure 1.2:	Successive stages in the Development of Shear Zones.
Figure 1.3:	Three Fundamental Modes of Fracture.
Figure 1.4:	Mode I Test Apparatus and Specimen Geometry
Figure 1.5:	Damage preceding a main crack in polystyrene under Mode I fatigue loading.
Figure 1.6:	Damage zone preceding notch in Mode II fatigue of a hollow cylinder.
Figure 1.7:	Crack Layer damage zone identification.
Figure 2.1:	Initial Consolidation of Kaolinite Block
Figure 2.2:	Specimen Geometry
Figure 2.3:	Description of the soil test cell
Figure 2.4:	Schematic for Specimen Impregnation.
Figure 2.5:	Photographs of Vacuum Chamber used in the impregnation process.
Figure 2.6:	Buehler Isomet saw used for sectioning the impregnated clay samples.
Figure 2.7:	Sectioning Procedure.
Figure 2.8:	Schematic layout of the Image Analyzer
Figure 3.1:	Damage zone (*LSD* zone) evolution on hollow clay cylinder illustrating the localized discontinuities for various cycles. Initially the grid lines were unbroken and orthogonally oriented at 1/4" spacings. However, during the fatigue test they provide a means of *measuring* the deformation in the damage
Figure 3.2:	Photograph of one of the cycle photographs shown in Figure 3.1 illustrating the localized discontinuities in front of the notch tip.
Figure 3.3:	Observed discontinuity lines in an un–notched fatigue

specimen.

Figure 3.4: Photograph illustrating the internal microstructure of the undamaged clay after impregnation. The Kaolinite is white and the bee's wax is dark brown. Notice the undamaged material is made up of two phases; the dense regions of clay called *clusters*, and the loosely densified region which has now been replaced by wax.

Figure 3.5: Comparison photograph of radial sections in undamaged (left) and damaged (right) regions of a cylinder. Notice that the damaged region shows stress induced morphological transformations of material from a previously undamaged state (clusters), to a damaged state (Localized Strip Densifications).

Figure 3.6; Axial section of a fatigued cylinder showing; (a) Network Densifications closest to the notch tip (b) Localized Strip Densifications intermixed with Clusters (c) Other densified regions.

Figure 3.7: Phothgraphs of a slip surface profiles emanating from the notch tip.

Figure 3.8: Photographs at selected stages from the high speed test.

Figure 4.1: Scale hierarchy in overconsolidated clay.

Figure 4.2 A plot of the distributions $f_k(c)$, for (k=1,...,6) and L_k=k*0.62 mm.

Figure 4.3 A plot of the standard deviation (σ_v) as the characteristic dimension L_k increases.

Figure 4.4 Three dimensional plot of the concentration distributions as the characteristic dimension is increased.

Figure 4.5 A prolate ellipsoid of revolution, showing the range of distances from the center to a sectioning plane which produce intersections in the class i.

Figure 4.6: Spatial distribution used in the aspect ratio error analysis study.

Figure 4.7 Plot of resulting error in the mean and standard deviations of the spatial distributions for induced error on the aspect ratio.

Figure 4.8 Data extracted from the image analyzer; a) Contours of

traced clusters. b) Distribution of equivalent radii c) Aspect ratio distribution.

Figure 4.9 Plot of two and three dimensional distributions for 5 classes using Saltykov's solution.

Figure 4.10 Plot of two and three dimensional distributions for 10 classes using Saltykov's solution.

Figure 4.11 Plot of two and three dimensional distributions for 15 classes using Saltykov's solution.

Figure 5.1 Schematic illustrating the *LSD* measurement procedure.

Figure 5.2 *LSD* histogram results for Cylinder 2–1. Note that Slide 2–1–A is nearest the notch and Slide 2–1–O is furthest away.

Figure 5.3 Three dimensional plot of *LSD* densities as related to their measurement locations along the axial and circumferential directions.

Figure 5.4 *LSD* area density contour plot for; a) Cylinder 2–1, notch at (0,100). b) Cylinder 2–3, notch at (0,100). c) Cylinder 3–1, notch at (0,150).

Figure 5.5 Three dimensional sequence of *LSD* zone evolution as a function of cycles.

Figure 5.6 Plot of *LSD* zone centroid distance vs. cycles for Cylinder 3–1.

Figure 5.7 Plot of *LSD* zone centroid distance vs. cycles for Cylinder 5–6.

Figure 5.8 *LSD* zone volume vs. cycles for Cylinder 3–1

Figure 5.9 *LSD* zone volume vs. cycles for Cylinder 5–6

Figure 5.10 Schematic of calculation procedure for R_1.

Figure 5.11 Plot of R_1 vs. cycles for Cylinder 5–6.

Figure 7.1: Three Cases which when superimposed produce a combined stress/displacement field observed in the experiment.

Figure 7.2: Plot of K. K. Lo's solution for a crack with an edge dislocation in the complex plane.

Figure 7.3: Plot of a crack with a dislocation dipole in the complex plane.

Figure 7.4: Mohr's circle diagrams defining the components of stress.

Figure 7.5: Plots of stress components in the vicinity of the crack tip resulting from a remotely applied shear stress.

Figure 7.6: Schematic of a crack interacting with a single dislocation dipole located directly in front of the crack. The dipole is assigned a burgess vector which creates a unit shear discontinuity.

Figure 7.7: Contour plot of σ_{xx} component for the case in Figure 7.6.

Figure 7.8: Contour plot of σ_{yy} component for the case in Figure 7.6.

Figure 7.9: Contour plot of τ_{xy} component for the case in Figure 7.6.

Figure 7.10: Contour plot of σ_{p1} component for the case in Figure 7.6.

Figure 7.11: Contour plot of τ_{max} component for the case in Figure 7.6.

Figure 7.12: Schematic of a crack interacting with a single dislocation dipole located at a 20° inclination in front of the crack. The dipole is assigned a burgess vector which creates a unit shear discontinuity.

Figure 7.13: Contour plot of σ_{xx} component for the case in Figure 7.12.

Figure 7.14: Contour plot of σ_{yy} component for the case in Figure 7.12.

Figure 7.15: Contour plot of τ_{xy} component for the case in Figure 7.12.

Figure 7.16: Contour plot of σ_{p1} component for the case in Figure 7.12.

Figure 7.17: Contour plot of τ_{max} component for the case in Figure 7.12

Figure 7.18: Photograph of *dogbone* cracks developed in a cylinder during testing.

Figure 7.19: Sketch orienting the crack and the distribution of dipole dislocations for the case photographed in Figure 7.18.

Figure 7.20: Contour plot of τ_{xy} stress component for the dogbone crack case (Figures 7.18 and 7.19).

Figure 7.21: Contour plot of σ_{p1} stress component for the dogbone

crack case (Figures 7.18 and 7.19).

Figure 7.22: Contour plot of τ_{max} stress component for the dogbone crack case (Figures 7.18 and 7.19).

Figure 7.23: Plot of non–dimensionalized SIF (K_I^* and K_{II}^*) vs. the angle locating the unit dipole (the angle $\beta = tan^{-1}(y_0/x_0)$). The case for b_y denotes a dipole with a unit opening, and the case for b_x denotes a unit shear discontinuity.

Figure 7.24: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit opening dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is parallel with the x–axis ($\Theta = 0^o$).

Figure 7.25: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit shear dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is parallel with the x–axis ($\Theta = 0^o$).

Figure 7.26: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit opening dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is inclined with the x–axis ($\Theta = +20^o$).

Figure 7.27: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit shear dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is inclined with the x–axis ($\Theta = +20^o$).

Figure 7.28: Photograph of a crack and preceding damage (below) for Cylinder 5–6 cycle # 262 and the approximation of the damage zone (above) with descrete slip lines.

Figure 7.29: Comparison plot of the SIF caused by the remote loading on a main crack (K_{crack}), due to the interaction with the damage (K_{damage}), and their respective total (K_{total}) vs an imaginary crack tip location.

Figure 7.30: An element of damaged material illustrating the internal and external damage.

Figure 7.31: Generic damage zone configuration.

Figure 7.32: Plot of ΔK_{sdz} (SIF due to damage only) vs crack tip location within the damage zone and a scattergram plot of the elemental *LSD* densities

Figure 8.1: Plot of the total measured potential energy and irreversible work vs cycle number for Cylinder 5–6.

Figure 8.2: Plot of the total measured potential energy and irreversible work vs. centroid distance for Cylinder 5–6.

Figure 8.3: Plots of thermodynamic force vs. flux in log–log format for various values of γ ranging between 1.0 and 10.0 kJ/M^3.

Figure 8.4: Plot of the thermodynamic force vs. flux with γ=1.0 kJ/M^3 for Cylinder 5–6 together with regression line fit.

List of Tables

Table 1.1: List of case histories of landslides classified by types of slides.

Table 2.1: Composition of Kaolinite Powder

Table 4.1: Representative volume study results.

Chapter 1

Introduction

1.1 Progressive Failure in Overconsolidated Clays

A characteristic common to many stiff clays is that they are overconsolidated. The rebound that takes place during a stress release often results in small fissures or cracks. As a consequence, many clays exist in a fissured state (1,2,19,21). The initiation of cracks in clay can be chemically induced as in the case of syneresis cracks (16). In general, any depositional process followed by a diagenetic or tectonic process can give rise to macrostructural features like fissures (13,14).

Once a crack is initiated, the stress concentration in the vicinity of the crack tip may cause a substantial reduction in available strength of the intact clay. This has been understood by the geotechnical profession for years. In 1936, Terzaghi (22) gave the some of the first quantitative data on the influence of joints and fissures on the strength of clays. He illustrated that such features are characteristic of overconsolidated clays and that the overall strength of the "stiff fissured clays" could be as low as one fifth to one tenth the strength of the intact clay. Similar findings were published by Skempton, Schuster, and Petley

when they studied the joints and fissures in London Clay (21).

Studies describing the behavior of fissured clays and, in general, overconsolidated clays with any type dislocations are warranted when reviewing the case histories of progressive slope failures. In Bjerrum's publication on the "Progressive Failure in Slopes of Overconsolidated Plastic Clay and Clay Shales" (2) he categorized failures in 60 individual case histories with respect to the type of the clay in which the failure occurred. In his paper, the term weathering was used to describe all changes in the upper layer of clay, including physical changes that do not originate from climatic conditions. Two phases of weathering were identified; the first, called disintegration, involves the breakdown of the diagenitic bonds. The second phase involves chemical changes and decomposition of the minerals. Bjerrum concentrated his attention on the first of these two phases. From his reported observations of the weathered clay he noted : *This zone in general will have a system of open cracks.*

Bjerrum's findings showed that the greatest number of slides (about 55%) had occurred in a zone of weathering in overconsolidated clays with strong diagenitic bonds. The next largest group (about 35%) proved to be in overconsolidated clays with weak bonds. In the second largest group Bjerrum found it difficult to differentiate whether the slides occurred in unweathered or weathered material. The remaining 10% of the study sample were slides that occurred in unweathered material of overconsolidated clay which has high diagenitic bonds.

In general, slope failure can be categorized into basic types of mass movement. Eleven basic types including multiple and complex landslides are shown in Fig. 1.1 (1). Researchers have also found that different types of clays tended to fail in different types of mass movements. A table which includes the type of movement, the type of clay and location, and reporting author(s) was summarized in (1) and is reproduced here in Table 1.1. According to Attewell and Farmer (1), Skempton and Hutchinson found "that the rotational form of multiple retrogressive slides occur most frequently on actively eroding, high relief slopes in which a thick stratum of overconsolidated fissured clay or clay shale is overlain by a thick bed of more competent rock.

Ultimately, geotechnical engineers realized the importance of incorporating cracks and other types of dislocations into the conventional constitutive models. Conventional failure criteria such as Tresca, Mises, or Coulomb may be appropriate for yield dominant failures, however they are not appropriate for describing the brittle mechanisms of failure.

In order to understand the mechanisms involved in progressive failure of overconsolidated clays, various types of research evolved. Much of the published work appears in the form of observational data; some appears in the form proposed models. In the next few pages some of these approaches will be introduced and commented on.

In 1969, Skempton and his coworkers (21) defined a classification system for discontinuity types. In this study, five types of discontinuities were distinguished. These five classifications are briefly listed and described as:

1) Bedding – a discontinuity with a gently undulating surface having a somewhat rough or bumpy texture.

2) Joints – Predominantly vertical and usually between 1 and 4 ft. high, and up to 18 ft. long with a pronounced trend to orthogonal directions.

3) Sheeting – Similar to Bedding except they dip (usually between 5° and 15°). They are also classified as low angle joints.

4) Fissures – Planar or conchoidal fractures, rarely more than 6 in. in size with a matt surface or texture. They also show scarcely any preferred orientations apart from the clear tendency to concentrate in sub–horizontal planes more or less parallel to bedding. The number of fissures per unit volume increases and their size correspondingly decreases as the upper surface of the clay is approached.

5) Faults – Large discontinuities with slightly polished surfaces. Usually, 5–10 mm of *gouge* clay evident on the shear planes.

It should be noted that this classification system is predominantly designed for site investigation information and, accordingly, all features described here are macroscopical anomalies.

In Laboratory observations, Skempton (20) also identified five stages in

the shearing deformation of stiff clay (see Fig 1.2). The first is defined as continuous nonhomogeneous strain. The second is the creation of inclined surfaces ranging between 10° and 30°. The third acknowledged the creation of Reidel shears. With further movement, the Reidel shear is no longer kinematically admissible and the new parallel or subparallel surface is created. And finally, the slip surface undergoes appreciable flattening as a result of still greater movements.

In 1970, Lo (10) proposed a technique to accommodate for size effect in fissured clay and presented complementary test results. Also in his paper, Lo described a computational procedure to determine the operational strength of fissured clay. This computational procedure generally relates to the probability of encountering a critically oriented crack.

Lo contributed another approach in 1973 (11) with a finite element analysis solution to progressive slope failure. For his soil model, Lo used a strain–softening behavior similar to that expected from a shear box test. A typical slope was analyzed repetitively while certain parameters were selectively varied and general comments were made on the results.

Other researchers focused their attention on the micromechanisms responsible for failure. In 1967, Morgenstern and Tchalenko (14) provided an excellent report on the microscopic structures observed in kaolin subjected to direct shear. They identified two distinct regions in the clay. The first is

referred to as the original soil fabric and describes the microstructure of the material subjected to environmental conditions and its sedimentary history. The second region is referred to as the shear–induced fabric. This region describes the microstructure of the material after post depositional shear strains have been induced. Morgenstern and Tchalenko included micrographs from laboratory experiments describing characteristics of the original fabric and the shear induced fabric resulting from shear box tests.

More recent investigations include the application of classical fracture mechanics as applied to a linear elastic material. Of interest in such studies is information about the crack propagation rate, direction, and critical crack length for various modes of fracture. A fracture mode designates geometrically the separation between crack surfaces. In general, a crack propagating in any given direction can be described as a combination of three fundamental modes of fracture. These fundamental modes are shown in Fig. 1.3.

Based on the qualitative descriptions of the failure mechanisms responsible for progressive failure, Palmer and Rice (15) suggested that sliding occurs on concentrated slip surfaces. Using concepts from fracture mechanics (i.e. the J–integral), attempts were made to assess the time dependence governing the propagation rate of a particular shear band. The assumed model used for the shear band was very similar to the cohesive force models (i.e. similar the models proposed by Barenblatt and Dugdale).

Their model asserts that there is a fixed linear relationship between the shear stress in the material and the displacement required to produce it(e.g. the stress–displacement relationships obtained from shear box tests). Since displacement is the integral of the strains over a given region, an immediate consequence is that size effects will occur (i.e. the assumption of a relation between shear stress and shear displacement introduces a characteristic length into the material description). Specifically, the size effect appears in the resistive part of the crack driving force equation. In addition, Palmer and Rice give advice for experimentally obtaining the energy release rate in an approximate manner by computing the energy under the unloading portion of the stress–strain curve in a particular shear box test. Other assumptions made in the shear band analysis include the application of the asymptotic stress distribution in the vicinity of the slip surface (this assumption implies a negligibly small zone of deformed material in the vicinity of the slip surface tip).

In 1985, Saada, Chudnovsky, and Kennedy (17) published results for the determination of the critical stress intensity factors (fracture toughness) K_{Ic} and K_{IIc}. The samples used in the for the Mode I investigation were hollow disks and the specimens used for the Mode II investigation were long hollow cylinders. The Mode I specimens were tested under fatigue loading conditions and the Mode II specimens were tested monotonically.

From these tests information regarding critical crack lengths was

obtained and their results were applied to the example of a critical crack length for an infinite slope at various angles of orientation. Also included was a brief outline for a thermodynamic description of the fracture process.

Other recent contributions include the investigations performed by Vallejo (23,24). His examinations concentrated on defining the direction of crack propagation under various modes (including mixed modes) of fracture. He determined that the crack propagation generally followed the direction of the maximum shear stress irrespective of the notch orientation.

1.2 Previous Work

The initial stage of this research began in 1983 with experimental evaluation of the critical stress intensity factors for both Modes I and II (K_{Ic} and K_{IIc}). The research work was performed by Mark Kennedy under the direction of Drs. Saada and Chudnovsky. Results following this research were published in (17).

In Kennedy's examination of Mode I behavior, either static or dynamic air pressure was applied by way of a membrane inserted along the inner surface of a hollow clay disk (see Fig. 1.4). A crack, initiated from a notch on the inner surface, was allowed to propagate in a stable manner until a critical crack length was reached. Records of the critical crack lengths were documented.

This information was then used with a Green's function to compute the critical stresses and stress intensity factor. However, the Green's function used was initially intended for a strip (not a hollow disk)
having the same stress distribution of a hollow disk.

In the Mode II study, Kennedy used a sample configuration in the form of a notched long hollow cylinder. A monotonic torsional couple was applied to selected samples and test results were documented. The test findings were processed via the solution proposed by Erdogan and Ratwani (8) for K_{IIc}.

In August of 1985, Majd Sharaf (18) completed additional research focused on the description of the kinematics of fatigue crack propagation of stiff clays. He augmented, his study by also considering the effects of variable overconsolidation ratios. His experimental studies were performed on specimens similar to Kennedy's for both Modes I and II.

1.3 The Concept of Cooperative Fracture and its Propagation

The significant role played by microdefects (damage) in the process of crack formation and growth is commonly recognized. In general, two extreme cases regarding the influence of defects in the fracture process are distinguished. Modeling of these two cases requires essentially different formalisms.

Case 1 – A crack propagates through a pre–existing field of defects (i.e., a pre–existing strength field) causing negligible changes in the field. The fluctuation of the microdefect field is directly reflected in the stochastic features of the fracture surfaces and also leads to the scatter of experimentally observed fracture parameters (critical crack length, critical load, etc.). The fracture is characterized by a *single path* phenomenon and a probabilistic approach seems most adequate under these circumstances.

Case 2 – The intensity of damage formed as a response to the stress concentration at the tip of a propagating crack is much greater than the intensity of the pre–existing damage. The crack propagation is then inseparable from the evolution of damage accompanying the crack. The damage accompanying the crack is often referred to as the active zone, the process zone, or the damage zone. This strongly cooperative phenomenon is modeled with a theory based on thermodynamics of irreversible processes.

A common mistake is to refer to non–cooperative fracture as *brittle* and similarly, to refer to cooperative fracture as *ductile*. However, the words brittle and ductile are commonly used in strength of materials to describe the micromechanisms of failure in a given material. That is, a brittle material fails in tension and a ductile material fails in shear. For example, if a crack propagates symmetrically in a *single path* while being tested in Mode II, then the failure mechanism is predominantly ductile while the fracture process is non–cooperative. Similarly, many materials like polystyrene exhibit highly cooperative fracture, while the failure mechanism is predominantly brittle (3,6,12).

Cooperative fracture is identified by a crack preceded by intensive

damage. The term damage is used here in a generic sense and it should be expected that the *nature* of damage varies between materials. When the damage ahead of the crack tip reaches a critical level, local instability takes place and the crack jumps within the damage zone to a new stable configuration. This crack advance, in turn, leads to the changes in the stress field. The latter causes further damage accumulation until a new local instability is reached. Thus we visualize the crack and surrounding damage propagation as a sequence of local instabilities and slow damage accumulation processes. Consequently, a repetition of the above described events on a microscopic scale results in a continuous stable crack propagation process on a macroscopic scale. The concept of *self similar* events constituting the fracture process allows one to develop the mathematical model of the phenomena.

Chudnovsky has shown that a description of the fracture process within a thermodynamic framework ultimately leads to an elegant model which realistically depicts stable crack propagation as well as provide a criteria for crack stability. He outlined these concepts in the context of different materials ranging from polymers to high alloy steels (3,4,5,6,7). The model generated from this framework is commonly referred to as the Crack Layer (CL) approach.

The crack together with its surrounding damage are referred to as a Crack Layer. Within the CL, microdefects can be observed. Depending on the material in which the CL is formed, these defects may be observed in the form

of shear bands, crazes, microcracks, voids or material densifications. For example, Fig. 1.5 exhibits a layer of extensive crazing accompanying fatigue crack propagation in polystyrene (12). Similar features of fracture propagation have been observed recently in stiff clays tested in Mode II. Fig. 1.6 shows the damage surrounding a propagating crack in a notched hollow cylinder subjected to cyclic torsional stresses. More will be said about Fig. 1.6 which is presented at this stage only to demonstrate the applicability of the concept to O.C. clay.

There are two complementary approaches to characterize crack layer propagation. One which can be called *the micromechanics of crack layer*, deals with modeling the stress–strain fields due to the interaction of the main crack with the surrounding damage. This approach requires detailed description of the surrounding damage as well as the geometry of the crack and stress–strain fields. Also, knowledge of the conditions for local instability and a detailed description of the crack jump from one stable configuration to another, are required. This approach is extremely tedious both from a theoretical and experimental point of view.

The second is a thermodynamic approach which describes the system in global terms based on the first principles, and pays no attention to the details of the fracture process. This phenomenological way is based on the general framework of the thermodynamics of irreversible processes.

For the model proposed and the material under consideration, damage is

represented by densified regions (these densified regions will be described in detail later), which can be visualized as three dimensional defects. Specifically we use the total volume of damaged material within a unit volume as the damage density ρ with the dimension mm^3/mm^3. As will be shown later, the damage density in the vicinity of the main crack can be directly evaluated from optical observations using an image analyzer. Fig. 1.7 shows schematically the damage zone surrounding and preceding a generic crack trajectory. The front zone of the CL within which damage accumulation is non zero is defined as the active zone. In this zone $\rho > 0$ and $\dot{\rho} > 0$.

During unloading conditions the rate of damage change is negligible and as a result a wake zone appears as a trace of the active zone propagation. In this zone $\rho > 0$ but $\dot{\rho} = 0$. The active zone is confined by the leading edge Γ_ℓ and the trailing edge Γ_t as shown in Fig. 1.7. If the increments of CL advance are small compared to the CL size, affine transformation of the active zone can reasonably approximate the actual evolution of damage. Accordingly, for an active zone of length ℓ, the rate of translation $\dot{\ell}$ can be considered as a thermodynamic flux. The law of CL propagation can be established by relating the fluxes to the reciprocal forces (the causes) within the framework of irreversible thermodynamics.

Chudnovsky has shown (3,4,6,7) that the cooperative fracture process can be modeled thermodynamically and is based on the following governing equation.

[1.1] $$T\dot{S}_i = \underline{\dot{\ell}} \cdot \underline{X}^c + \underline{\dot{\omega}} \cdot \underline{X}^{rot} + \underset{\sim}{\dot{d}} \cdot \underset{\sim}{X}^{def} = 0$$

where $\dot{S}_i$ is the global entropy production, $\underline{\dot{\ell}}$, $\underline{\dot{\omega}}$, and $\underset{\sim}{\dot{d}}$ are thermodynamic fluxes, and the X's are their reciprocal forces. The fluxes, describe the elementary movements of the Crack Layer shown in Fig. 1.7 and are defined as:

$\underline{\dot{\ell}}$	–	Rate of Translation
$\underline{\dot{\omega}}$	–	Rate of Rotation
$\underset{\sim}{\dot{d}}$	–	Rate of deformation

The conjugate forces X decompose into two parts:

[1.2] $$X = A - \gamma R$$

A is referred to as the active part and R is called the resistive part of a particular thermodynamic force. The active part of a thermodynamic force is evaluated by assessing the potential energy release associated with it's particular elementary movement. The resistive part, R, describes the amount of energy consumed during the movement, and γ is referred to as the specific energy for damage formation.

The thermodynamic forces (Equation 1.2) resemble the criterion for crack instability in a Griffiths crack (9). The Griffiths crack instability

condition can be expressed as:

$$[1.3] \qquad \mathcal{G} - 2\gamma = \left\{ \begin{array}{l} > 0 \ \text{unstable} \\ < 0 \ \text{stable} \end{array} \right\}$$

$\mathcal{G}$ in Equation 1.3 is the elastic energy release rate and 2γ is the surface energy of the new crack surface. It should be noted that Equation 1.3 represents only the necessary condition for instability (i.e., a sufficient condition must also be met). When comparing Equation 1.2 with 1.3, one can appreciate the duality that exists between the Crack Layer theory and other theories describing crack instability. That is, $\gamma R - A\,(2\gamma - \mathcal{G})$ represents the energy barrier which must be overcome for movement of the damage zone (propagation of an ideal crack).

1.4 Thesis Outline

Consequently, the ultimate goal of this research is to:

> Develop a constitutive model which describes the Mode II fracture process in overconsolidated clays. This is completed by describing the thermodynamic forces thereby enabling predictions of the fluxes (elementary movements).

In order to achieve this goal, we identify five main tasks which must be completed.

1) Identify the nature of defects. That is, we must identify *damage* in overconsolidated clays. This requires characterization of both the damaged and undamaged states of the material.

2) Characterize the distribution of defects. The distribution characterization is required before estimates of the active and resistive parts of the thermodynamic forces can be evaluated. The active part requires knowledge of the damage configuration, and the resistive part requires information about the rate of damage accumulation.

3) Stress and Energy Analysis on Crack–Damage Interaction. A semi–empirical stress analysis should be performed to estimate the amount of energy release associated with particular movements observed in the experiment.

4) Integral Characterization of the zone of damage. This is performed experimentally with techniques in quantitative stereology and is used to compute the resistive parts of the forces.

5) Integral Evolution of Damage with respect to time. Once steps 3 and 4 are accomplished at various time intervals, the thermodynamic forces can be evaluated whereby computing γ, the specific enthalpy of damage. γ is considered a material property in the Crack Layer theory. Next, the constitutive law can be expressed in terms of the fluxes and forces.

The principal objective of this thesis is to investigate the necessary aspects of the above outlined research as applied to overconsolidated clay subjected to Mode–II cyclic loading.

References

1) Attewell, P. B., Farmer, I. W. (1976). "Principles of Engineering Geology," John Wiley and Sons, Inc. New York pp 633 – 696.

2) Bjerrum, L. (1974). "Progressive Failure of Overconsolidated Plastic

Clay and Clay Shales," Terzaghi Lectures No. 3 (ASCE) New York.

3) Chudnovsky, A. (1983). "Crack Layer Theory,", NASA Report, Grant NAG –3–23.

4) Chudnovsky, A. (1984). "Statistics and Thermodynamics of Fracture," Journal of Engineering Science, Vol. 22, No 8–10 pp 989 – 997.

5) Chudnovsky, A., Moet, A., Bankert, R. J., Takemori, M. T., (1983). "Effect of Damage Dissemination on Crack Propagation in Polystyrene," Journal of Applied Physics, Vol. 54, No 10 pp 5562 – 5567.

6) Chudnovsky, A., Moet, A. (1985). "Thermodynamics of Translational Crack Layer Propagation," Journal of Material Science, Vol. 20, pp 630 – 635.

7) Chudnovsky, A. (1987). "Crack Layer Theory," Journal of Applied Mechanics. ASME Transactions, In press.

8) Erdogan, F. E., and Ratwani, M. (1973). "A Circumferential Crack in a Cylindrical Shell Under Torsion," Int. Journal of Fract. Mech., Vol. 8, pp 87 – 95.

9) Hellan, Kare, (1984). "Introduction to Fracture Mechanics," McGraw Hill, New York pp 5,73

10) Lo, K. Y. (1970). "The Operational Strength of Fissured Clays," Geotechnique, Vol. 20, No. 10, pp 57 – 74.

11) Lo, K. Y., Lee, C. F. (1973). "Analysis of Progressive Failure in Clay Slopes," Int. Conf. Soil Mech. Found. Engg., Vol. 1.1, pp 251 – 258.

12) Morelli, T. A., Takemori, M. T. (1983). "Fatigue Advance Mechanisms in Polymers: Rubber Toughening Mechanisms in Blends of Poly 12, 6–Dimethyl–1, 4–Phenylene Oxide and Polystyrene," Journal of Material Science, Vol. 18, pp 576 – 581.

13) Morgenstern, N. R. (1977). "Slopes and Excavations in Heavily Overconsolidated Clays," Proc. Ninth Int. Conf. Soil Mech. Found. Eng., Vol. 2, pp 576 – 581.

14) Morgenstern, N. R., Tchalenko, J. S. (1967). "Microscopic Structures in Kaolin Subjected to Direct Shear," Geotechnique, Vol. 17, pp 309 – 328.

15) Palmer, A. C., Rice, J. R. (1973). "The Growth of Slip Surfaces in the Progressive Failure of Overconsolidated Clays," Proc. of Royal Soc. of London Assoc., Vol. 332, pp 527 – 548.

16) Read, H. N., Watson, J. (1962). "Introduction to Geology," Macmillan

and Co., pp 637 – 640.

17) Saada, A. S., Chudnovsky, A., Kennedy, M. (1985). "A Fracture Mechanics Study of Stiff Clays," Proc. Eleventh Int. Conf. Soil Mech. Found. Engg., San Francisco, Vol. 2, pp 637 – 640.

18) Sharaf, M. (1985). "A Fracture Mechanics and Crack Propagation Approach to the study of Overconsolidated Clay," M. S. Thesis, Dept. Civil Engg., Case Western Reserve University

19) Skempton, A. W. (1966). "Slope Stability in Cuttings in Brown London Clay," Proc. Ninth Int. Conf. Soil Mech. Found. Eng., Tokyo, Vol. 13, pp 261 – 270.

20) Skempton, A. W. (1966). "Some Observations On Tectonic Shear Zones," Proc. First Int. Congress Rock Mech., Vol. 1, pp 329 – 335 (Lisbon).

21) Skempton, A. W., Schuster, R. L., Petley, D. J. (1969). "Joints and Fissures in the London Clay at Wraysbury and Edgeware," Geotechnique, Vol. 19, pp 205 – 217.

22) Terzaghi, K. (1936). "Stability of Slopes in Natural Clay," Proc. First Int. Conf. Soil Mech. and Found. Eng., Vol. 1, PP 161 – 165, Harvard.

23) Vallejo, L. E. (1985). "Fissure Interaction and Progressive Failure of Slopes," Proc. Eleventh Int. Conf. Soil Mech. Found. Eng., San Francisco, Vol. 4, pp 2353 –2356.

24) Vallejo, L. E. (1986). "Mechanics of Crack Propagation in Stiff Clays," Proc. from Sym. on Geotech. Aspects of Stiff and Hard Clays, ASCE, Seattle Washington.

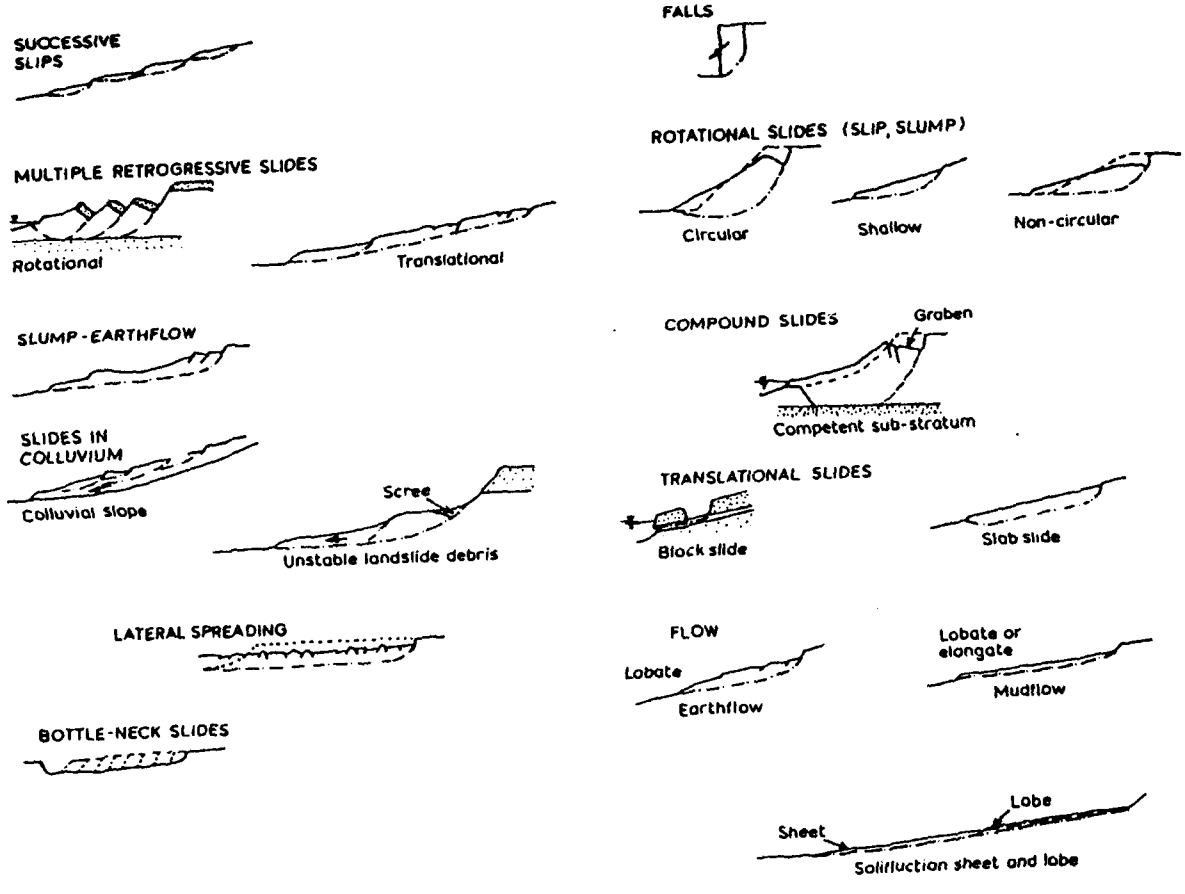


Figure 1.1 Basic types of mass movements constituting landslides.

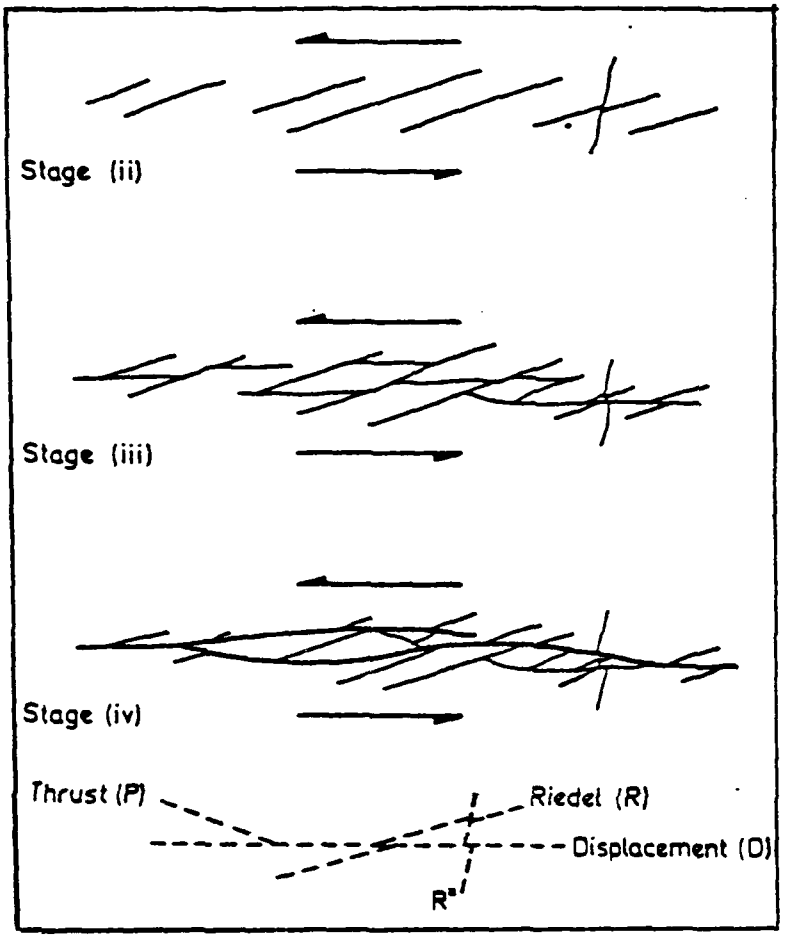


Figure 1.2: Successive stages in the Development of Shear Zones.

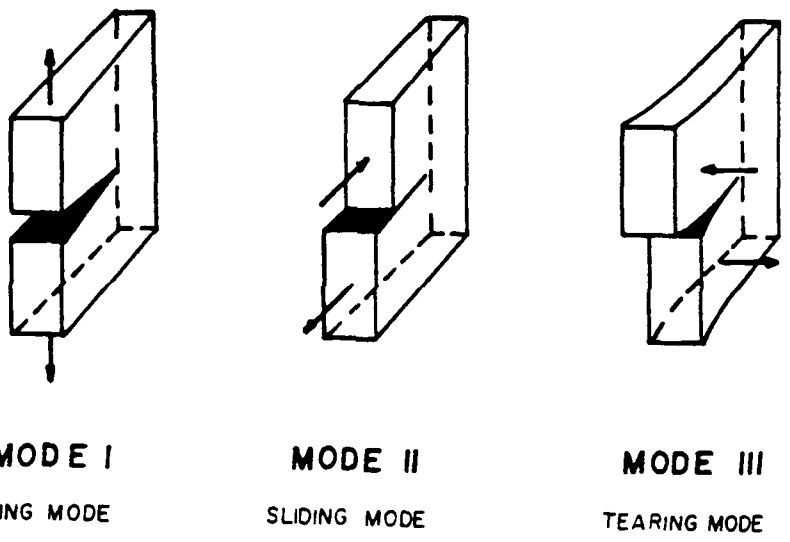


Figure 1.3: Three Fundamental Modes of Fracture.

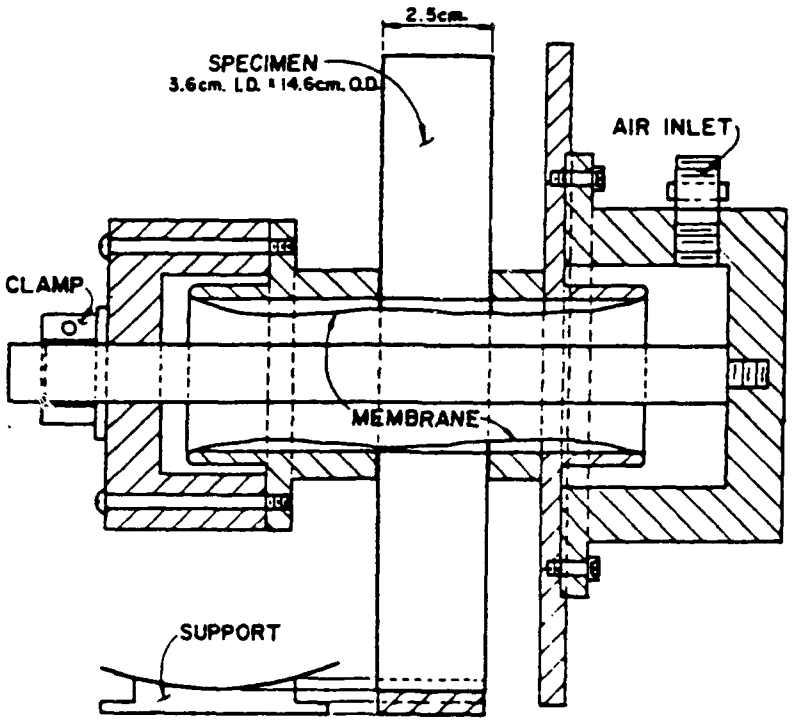


Figure 1.4: Mode I Test Apparatus and Specimen Geometry

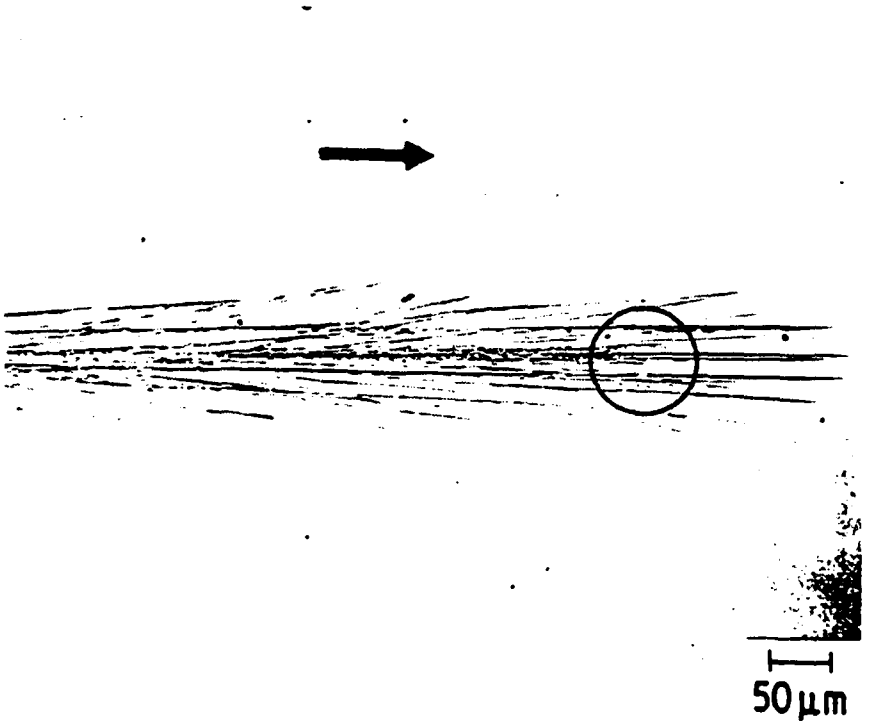


Figure 1.5: Damage preceding a main crack in polystyrene under Mode I fatigue loading.

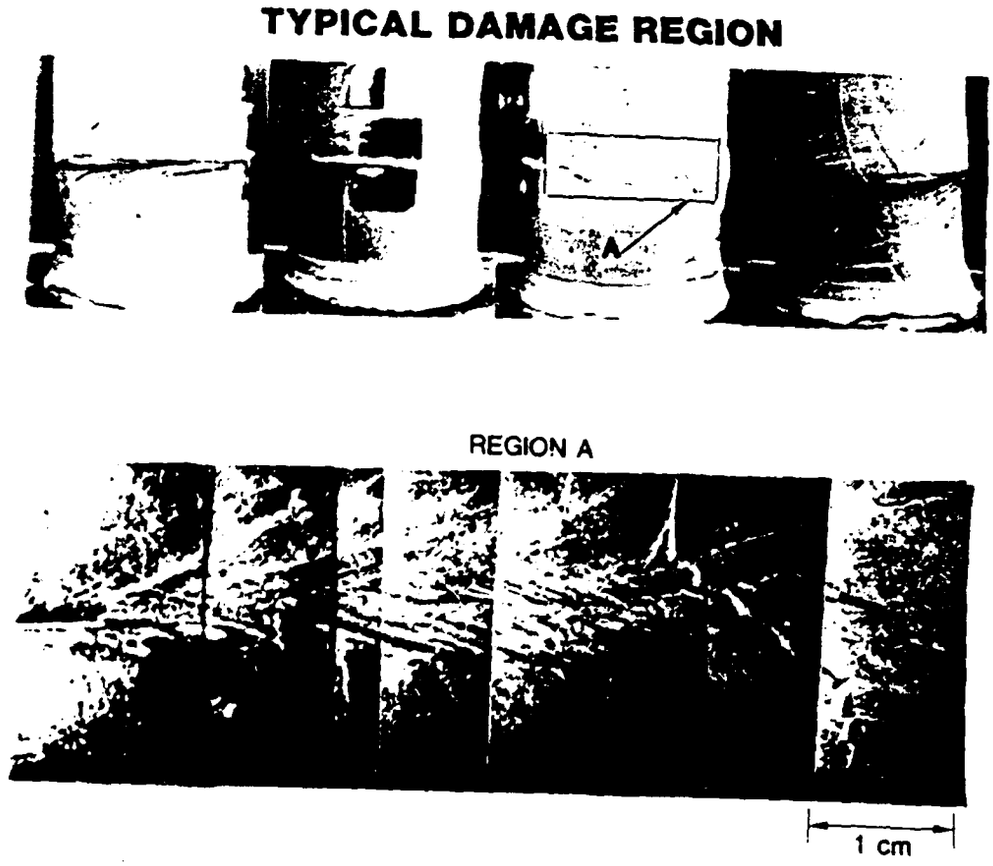


Figure 1.6: Damage zone preceding notch in Mode II fatigue of a hollow cylinder.

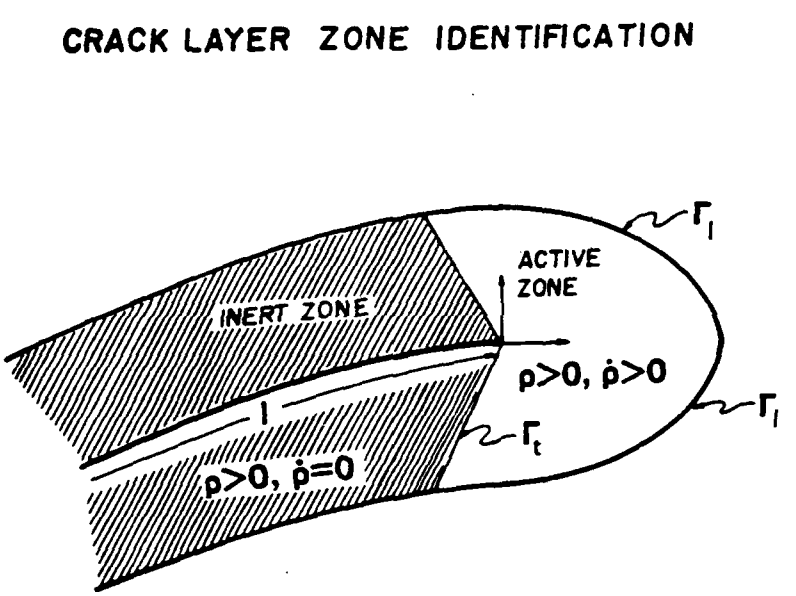


Figure 1.7: Crack Layer damage zone identification.

Type of Movement	*Clay and Location*	*Authors*
Falls	Leda Clay (St. Lawrence Valley)	Bazett *et al.* (1961)
	Brown London Clay (Bradwell, Essex)	Skempton and La Rochelle (1965)
	Residual soil (Hong Kong) (decomposed granite)	Lumb (1962)
Rotational slides		
Circular	Slightly overconsolidated (Lodalen, Oslo)	Sevaldson (1956)
	Stiff and medium blue clay (Chicago)	Ireland (1954)
	Fen silt and soft plastic Fen clay (Esu Brink Cut, Norfolk)	Skempton (1945)
	Stiff, fissured, overconsolidated London Clay (Warden Point, Kent)	Hutchinson (1968)
	Intact clay till (Selset, Yorkshire)	Skempton and Brown (1961)
Non-circular	Jurassic clay (Barrage de Grosbois, France)	Collin (1864)
	Glaciolacustrine sand, silt and clay, glaciofluvial deposits, fluvial sand and gravel, alluvial fan deposits, windblown sand (nr. Fort Spokane, Washington)	Jones *et al.* (1961)
	Weald Clay (Sevenoaks, Kent)	Toms (1948)
Shallow	London Clay slopes	Hutchinson (1967a)
Compound slides		
	London Clay (Kent coast, England)	Hutchinson and Hughes (1968)
	Soft quick clay and weathered crust (Bekkelaget, Oslo)	Eide and Bjerrum (1954)
	Pliocene lacustrine clay, silt, gravels and capping of Pleistocene andesite tuff (Gradot, Yugoslavia)	Suklje and Vidmar (1961)
Translational slides	Unweathered, overconsolidated and jointed clays (Valdermo, Italy)	Esu (1966)
	London Clay low-angle slopes	Hutchinson (1967a)
	Residual soil (Canaleira, nr. Santos, Brazil)	Vargas and Pichler (1957)
	Quick clay (Furre, Norway)	Hutchinson (1961)

Type of Movement	*Clay and Location*	*Authors*
Flows		
Rapid flows (mass transport)	Volcanic mud – 'Lahars' (Gunong Keloet, Java)	Scrivenor (1929)
	Volcanic mudflow (Mt. Bandai, Japan)	Iida (1938)
	Mudflow (Wrightwood, Southern California)	Sharp and Nobles (1953)
	Mudflows (Tenmile Range, Central Colorado)	Curry (1966)
Mudflow	Varved or laminated clays (Steep Rock Lake, Ontario)	Legget and Bartley (1953)
	Elongate Slumgullion mudflow (nr. Lake City, Colorado)	Crandell and Varnes (1961)
	Elongate mudflow (Stoss, St. Gall, Switzerland)	von Moos (1953)
	Elongate mudflow (Mt. Chausu, (Japan)	Fukuoka (1953)
Solifluction Lobes and Sheets	Hythe Beds (nr. Sevenoaks, Kent)	Skempton and Petley (1967); Weeks (1969)
	London Clay (Boughton Hill, Kent, England) and Wadhurst Clay	Weeks (1969)
Successive slips	Irregular slips in London Clay (Hadleigh Castle, Essex, England)	Hutchinson (1967a)
	Irregular slips in London Clay (High Halstow, Kent, England)	Hutchinson (1967b)
Multiple Retrogressive slips		
Rotational	Oligocene clays (Seagrove Bay, Isle of Wight)	Skempton (1946)
	Chalk and Gault Clay (Folkestone Warren, Kent, England)	Hutchinson (1969); Toms (1953)
	Heavily overconsolidated, stiff, fissured, lacustrine silts and clays (Meikle River, Alberta, Canada)	Nasmith (1964)
	Overconsolidated, stiff, fissured interstadial or interglacial clay (Sandnes, Norway)	Bjerrum (1967)
Translational	Overconsolidated clay (Jackfield, Shropshire, England)	Henkel and Skempton (1954)
	Miocene shale and bentonitic tuff, (Portuguese Bend, California)	Merriam (1960)
	Silty deposits (Vibstad, Norway)	Hutchinson (1965)

Table 1.1: List of case histories of landslides classified by types of slides.

Chapter 2

Experimental Procedures

2.1 Introduction

The information contained herein describes the laboratory procedures used in the sample preparation and testing as well as some of the data processing performed on this project. It should be noted that only descriptions pertaining to the successful completion of a laboratory test are included and not information about particular procedures involved in the data processing. For example, information regarding a description of the image analyzer, its capabilities and its general use are described in this chapter, but information regarding the measurement of particular features using this tool are dealt with individually.

A numbering system used throughout this report was adopted for specimen identification. Each cylinder identification number contains two different numbers separated by a slash (–). The first number identifies the batch from which the cylinder was cut and the second number identifies the cylinder within that batch. For example, Cylinder 2–3 was the third specimen

tested from batch # 2. Typically, four cylinders can be fashioned from one batch. However, there are instances where a particular cylinder was lost in the manufacturing, consolidating, or impregnation process. Mention of these mishaps will be made when appropriate.

2.2 Material Properties

The kaolinite used in these experiments has the following properties:

Liquid Limit = 56.3 %

Plastic Limit = 37.5%

Other pertinent information regarding a spectral analysis for the chemical composition of this material are listed in Table 2.1.

2.3 Initial Consolidation

The first step is to produce a large block of clay (referred to herein as a *batch*) strong enough to withstand a machining process. Specifically, a machining process is used to manufacture the notched hollow cylinders. Typically, each batch is large enough to provide four cylinders.

Seventeen pounds of the kaolinite clay powder (described in Section 2.2) are mixed with an equal weight of de–aired distilled water to form a slurry.

This corresponds to a water content of approximately 100% (about twice the liquid limit). The kaolin powder is placed in a 20 cm. diameter by 43 cm. high (8.0 in. diameter by 17.0 in. high) consolidometer. Next, the water is slowly and steadily drawn into the consolidometer by using a vacuum. After the proper amount of clay powder and water mixture is attained, mixing is performed manually until a consistent slurry is achieved.

Once the mixing is complete, the consolidation pressure is increased at a constant rate of 4.7 kPa/hour (0.68 psi/hour) while allowing drainage on both the top and bottom of the sample (see Fig. 2.1). After a five day waiting period, a final vertical pressure of 630 kPa (90 psi) is reached. This maximum pressure is then allowed to remain on the batch for an additional 48 hours. Afterward, another 24 hours is allowed for rebound.

2.4 Cylinder Preparation

Four 7.2 cm (2.83 inch) diameter solid cylinders are cut from the batch. Each cylinder is then wrapped in cellophane and placed in a humidity chamber for storage.

When a test is scheduled, a solid cylinder is removed from the humidity chamber for further sample preparation. A 5.1 cm (2.0 inch) diameter core is removed using a wire cutter and a special mold. Next, the ends are trimmed to

give an 11.4 cm (4.5 inch) high hollow cylinder. The cylinder is then placed in a special lathe and the outer wall is carved to the configuration shown in Fig. 2.2.

The next step involves cutting a 3.8 cm (1.5 inch) circular notch at the mid–height position (see Fig. 2.2). Once the notch is cut, two thin teflon sheets cut to the same geometry are inserted into the notch. This is done to prevent the notch from closing and healing during the next consolidation stage.

At this stage, a grid consisting of orthogonal lines is stamped on the cylinder. This grid is located in front of one of the notch tips. It is important here to use an alcohol based ink since the ink grid must dry on the clay sample while the sample remains moist. During the fatigue test the lagrangian grid is photographed and the configuration of subsequent *microcracks* can be determined.

2.5 Final Consolidation

After a particular specimen has been prepared (i.e. cut to the specified geometry, notched, etc.), it is placed between two rubber membranes and inserted into a special testing cell shown schematically in Fig. 2.4. The cell is filled with silicon oil and positioned in the test frame of a pneumatic analog computer which consolidates the soil under K_o conditions (2). This means that consolidation takes place with no changes in cross section. Basically, the

computer measures the volume of water expelled from the specimen through the porous stones and drainage lines and moves the crosshead of the loading frame down to compensate for the volume loss. Independently, the cell pressure is steadily increased at a constant rate of 50 kPa (7.25 psi) per hour starting at 207 kPa (30 psi) and ending at 620 kPa (90 psi). For this clay, the ratio of the cell to vertical pressure is 0.47. This ratio is called the coefficient of earth pressure at rest, K_o. Since the maximum cell pressure attained is 620 kPa (90 psi), the maximum vertical pressure applied on all samples should be approximately 1320 kPa (192 psi). Once the maximum pressure is achieved, it is maintained constant for an additional 24 hours, and afterward, the consolidation load is removed and the cell pressure is reduced to the test pressure 24 hours prior to the test. The hydrostatic rebound pressure selected was either 276 or 207 kPa (40 or 30 psi) depending on the overconsolidation ratio selected for the particular test. The 276 and 207 kPa (40 and 30 psi) rebound pressures correspond to vertical overconsolidation ratios of 4.78 and 6.38 respectively.

It should be noted that the specimens are tapered to a larger cross sectional area at the edges (see Fig. 2.2). This is done to prevent failure near the grips. As a result, their cross sectional areas varied between 13.78 cm^2 and 20.29 cm^2 (2.13 in^2 and 3.15 in^2). Consequently an average cross sectional area of 15.41 cm^2 (2.39 in^2) was used when calibrating the analog computer displacement setting.

2.6 The Fatigue Test

After the consolidation and rebound process, the specimens are ready to be tested for fracture studies. The test cell is placed in frame which provides the researcher with the capability of applying a sophisticated load pattern (1). For all tests reported herein, however, this system was used to apply a cyclic torsional couple exclusively. The loading frequency used was typically selected at 1.0 cycle/minute and applied in a sinusoidal fashion. Nevertheless, some tests were performed at 0.5 cycles/minute to investigate the effects of the stress/strain rate on the damage characterization (discussed in later). Similarly, the amplitude of the applied couple was typically chosen such that a maximum of 6.0 Newton–Meters (approximately 50 in–lbs) was reached, but some tests were performed at different amplitudes. The amplitude of the applied couple chosen was based on a percentage of the required torque to monotonically fail a similar cylinder prepared under identical conditions. Details about the loading frequency and magnitude will be provided later in this thesis prior to discussing the results obtained for each particular test.

2.7 Impregnation

The sample impregnation is performed for two reasons; The first is that it enables the researcher the capability of cutting and polishing relatively small

sections without greatly disturbing the particle orientation. The second reason is that the impregnated sections show the features of interest more readily. That is, since the kaolin powder is white and the bees wax is dark brown a natural contrast barrier is defined when identifying which features are clay and which are wax. Also, it is believed that the wax only replaces the larger voids (air pockets) in the dried sample and simply surrounds the more densified regions of clay. Thus what results from a properly impregnated and polished section is essentially a *contrast map* of densified features in the material.

After the cylinder is tested under the cyclic torsional couple, it is removed from the test cell and allowed to air dry at room temperature for a minimum period of four days. Once the specimen is sufficiently dry, the impregnation sequence can commence.

Initially, the bee's wax is raised to a temperature of approximately 110° *C* in a melting pot. Once the temperature is stabilized the level of the molten wax is adjusted such that the specimen height and the depth of the wax are approximately the same. Next the specimen is carefully immersed in the wax and the entire setup is placed in a vacuum chamber. After the vacuum bell is secured, an applied vacuum of 104 kPa (15.0 psi) is applied for a period of approximately 2 to $2^1/_2$ hours. This is shown schematically in Fig. 2.4 and in the photograph sequence of Fig. 2.5. Next, the specimens are removed and allowed to cool for a period of approximately 24 hours before sectioning.

2.8 Cylinder Sectioning

Sectioning is performed in order to reveal characteristic features in the interior of the specimen. Once identified, the morphology of these features can be studied with various techniques available in microscopy. In this report the majority of studies were performed with a stereoscope.

Prior to sectioning, the specimens were usually photographed and the section locations were marked with a soft extra–fine marker. Next, three separate coats of epoxy were applied individually to the inner surface of the hollow cylinder. The epoxy was applied to reinforce the clay shell so it would better withstand the stress levels imposed on it during sectioning.

The rough cutting is achieved with the use of a band saw. The band saw is only used to reduce the cylinder into pieces small enough to mount in the Buehler Isomet Saw grips. Note, that the lowest speed was used to cut with the band saw and a light steady pressure was applied during cutting.

Refined sections were all made with a Buehler Isomet Saw and a high concentration diamond tipped wafering blade. Cutting was performed at a blade speed setting of 4.5 and ethyl–glycol was used for the saw blade lubricating fluid (see Fig. 2.6). These sections are called "analysis sections" in this report. The analysis sections were generally cut in either a radial of

circumferential fashion shown in Fig. 2.7. Typically, the axial and circumferential sections were all made in the vicinity of the notch tip. However, some sections were made in regions away from the notch tip either to define a reference level characterization of the clay or for other related studies (see Chapters 3 and 4).

Ideally, one half of a cylinder would be sectioned circumferentiallyand the other half would be sectioned in a radial fashion at 5 mm (0.197 in) spacings. However, in the event that shrinkage cracks or sectioning decimation eliminate the possibility to obtain both types of sections, a higher priority is devoted to the radial sections. The radial sections are deemed more important since they are the type used to reconstruct a quantitative damage evolution for a given cylinder test (elaborated on in Chapter 5).

2.9 Section Polishing

Once the selected sections are made with the Buehler saw, the sections are adhered to either a glass or aluminum slide with an epoxy cement. After the epoxy was given enough time to dry, the polishing can begin.

Polishing is performed on Buehler polishing wheels. The polishing sequence is performed in two separate stages. The first stage of polishing consists of a coarse stage and is done to remove some of the cutting scratches as

well as provide a uniform surface for the microscopic inspection. The second stage is a fine polishing stage and simply improves the quality of the first stage. During the first stage, a 800 grit water–resistant polishing paper is used and during the second stage polishing a billiard cloth polishing cloth is used. In both cases, cool tap water was used as lubricating fluid during the polishing.

Other helpful hints include the following:

1) Polish the section by applying light pressure and maintain a steady hand without rotating the wrist.

2) Keep the polishing duration brief (i.e. only allow the section and the wheel to be in contact for no more than a one second interval) and maintain cool water on the wheel. If the wax becomes to warm from the frictional heat distortion of the features can result from creep of the wax.

2.10 General Use of the Image Analyzer

This section is devoted to providing the reader with a general description of the Image Analyzer used on this research project. The system was originally developed for use in the medical research laboratory. The system purchased for our use consists of the following major components:

1) Nikon Stereoscope with trinocular head
2) 35 mm auto/fixed exposure camera and shell

3) Video Camera
4) Panasonic VHF/CRT Hi–resolution black and white monitor.
5) Nikon Exposure box
6) Houston Instruments Digitizing Tablet
7) HP Vectra (IBM PCAT Compatible).
8) Interfacing card
9) HP 7470A plotter
10) Epson FX286 Dot Matrix Printer.
11) Bioquant Image Analyzer software.

A schematic drawing of the system configuration is shown in Figure 2.8.

This system essentially enables the researcher to accurately measure irregular features (to 100 microns) as well as to statistically quantify stereological information. The Bioquant software provides a selection of various types of measurements and computations which can be made readily while tracing the shapes of the particles under the microscope. In Addition, the information is stored in selected arrays for later reprocessing. The reprocessing portion of this software includes a statistical package which allows the researcher the ability to extract various test statistics and observe measurement distributions. Typical information processed during this study include area, and aspect ratio measurements.

To start the image analyzer, first make sure the hardware (equipment) is interfaced correctly as shown in Fig. 2.8. The Vectra should be shut off and the switch located on the back side of the Vectra should be switched to the upward position. This switch adjusts the video monitor frequency so as to be compatible with the Panasonic Monitor. Next, the HP monitor should be

disconnected and the Panasonic monitor should be installed. The third step involves simply turning on all equipment. At this time the Vectra should boot up on the Panasonic monitor. After successfully reaching this stage, the user should simply type "im" for image and press the enter key. This will automatically start up the Bioquant software. A suggestion is to try the tutorial package included with the software until a certain level of confidence is reached.

References

1) Saada A. S. (1968). " A Pneumatic Computer for Testing Cross Anisotropic Materials," Mater. Res. and Stanps., American Society for Testing and Materials, Vol. 8, No. 1, pp 17 – 23.

2) Saada, A. S. (1970). "One Dimensional Consolidation in a Triaxial Cell," Journal of Soil Mech. and Founds. Division, ASCE, Vol. 96, No SM3, pp 1085 – 1087.

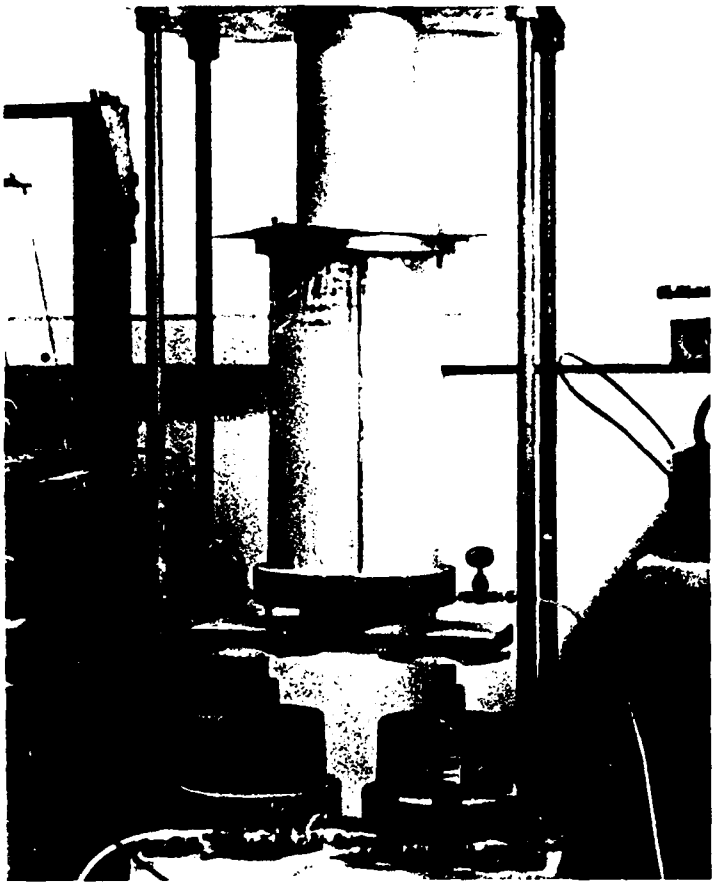


Figure 2.1: Initial Consolidation of Kaolinite Block

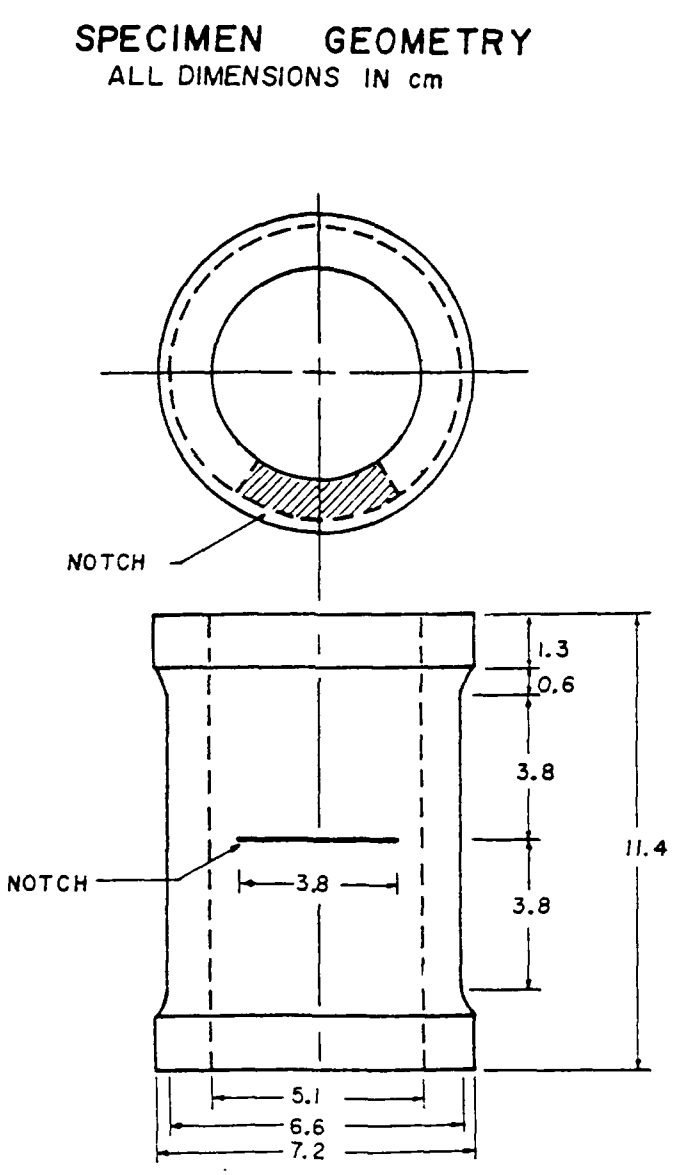


Figure 2.2: Specimen Geometry

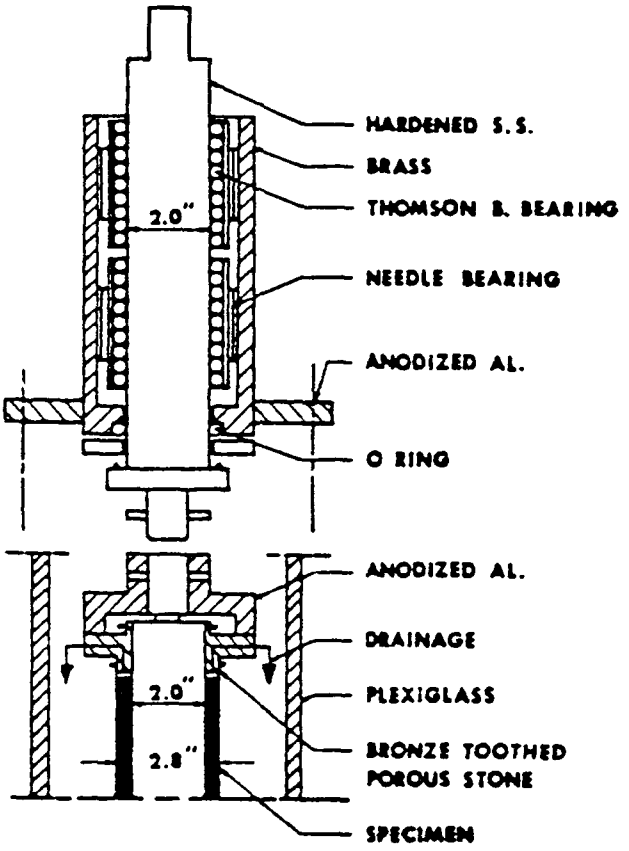


Figure 2.3: Description of the soil test cell

IMPREGNATION PROCEDURE

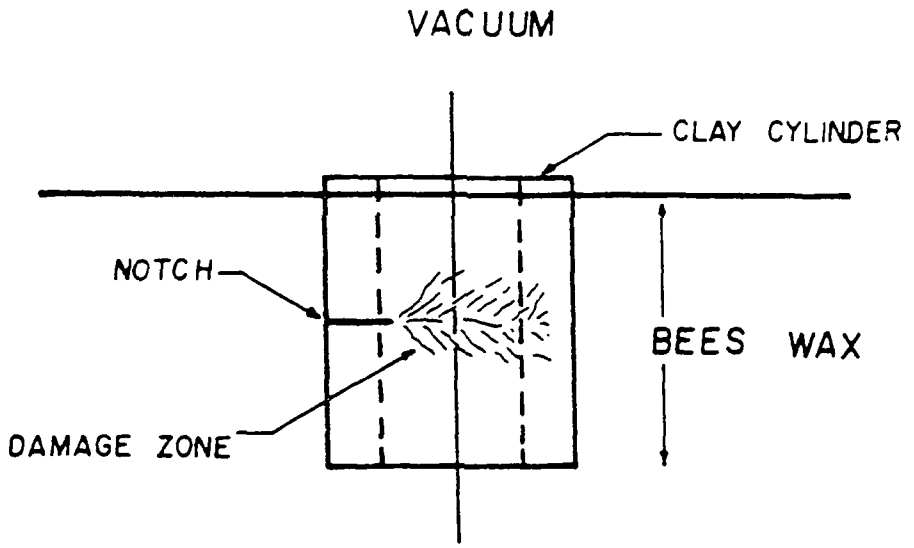


Figure 2.4: Schematic for Specimen Impregnation.

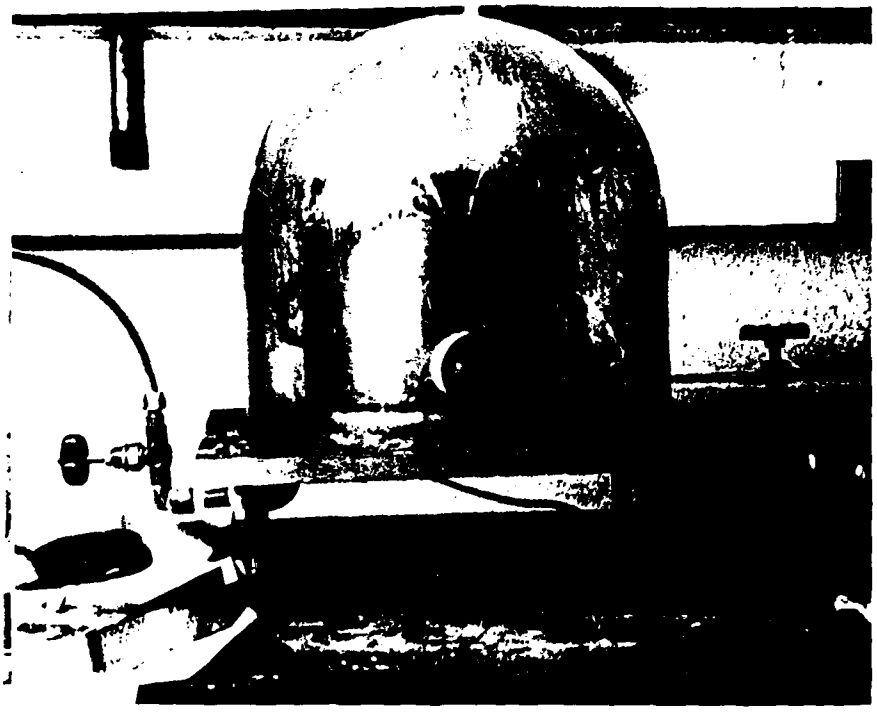


Figure 2.5: Photographs of Vacuum Chamber used in the impregnation process.

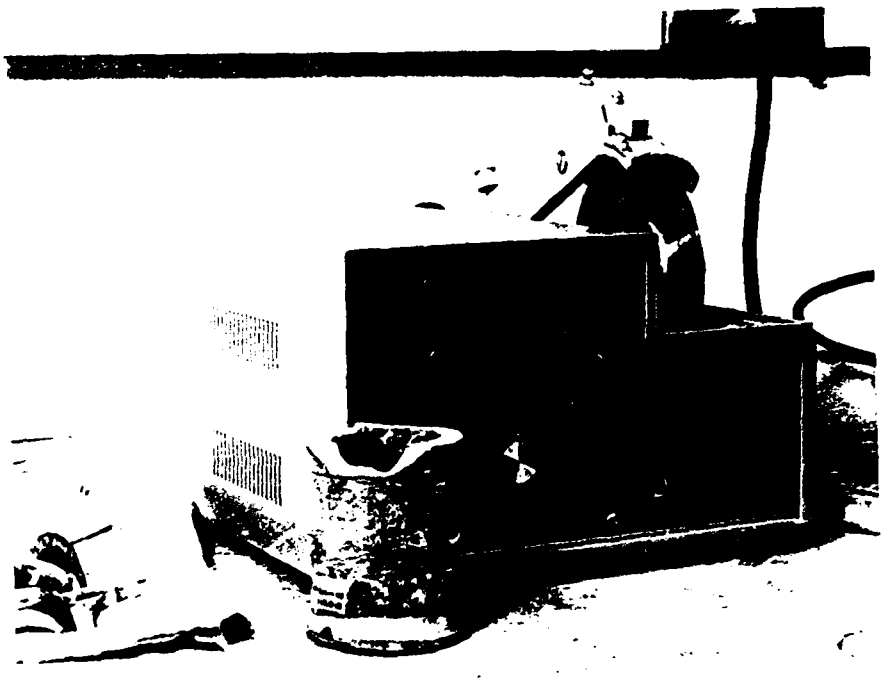


Figure 2.6: Buehler Isomet saw used for sectioning the impregnated clay samples.

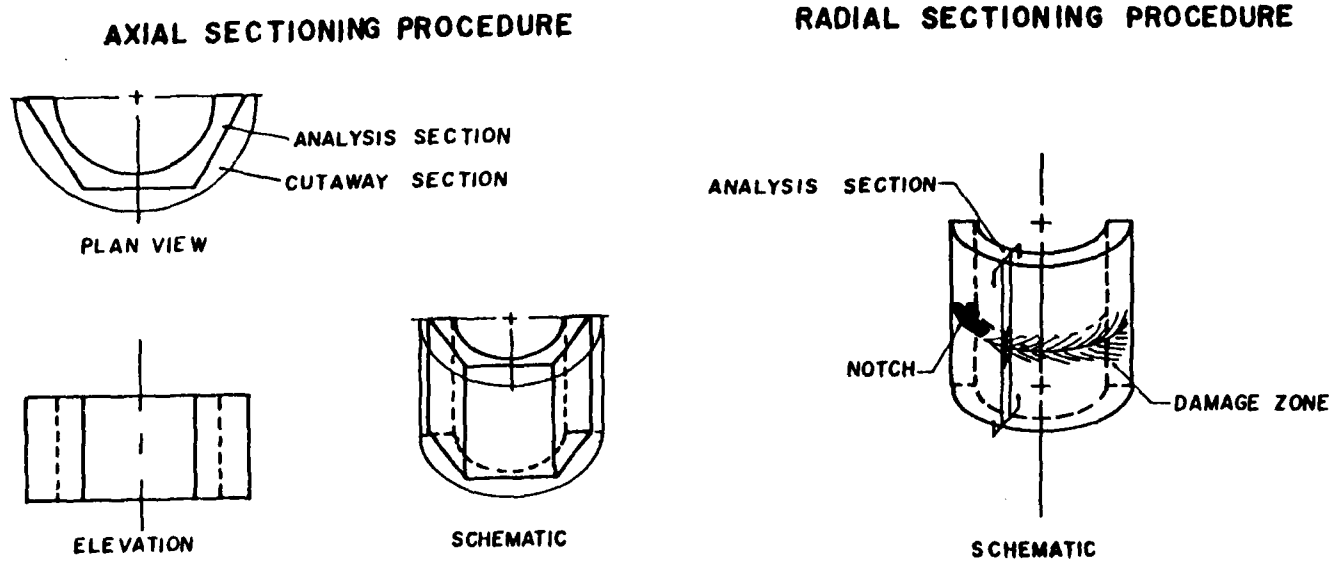


Figure 2.7: Sectioning Procedure

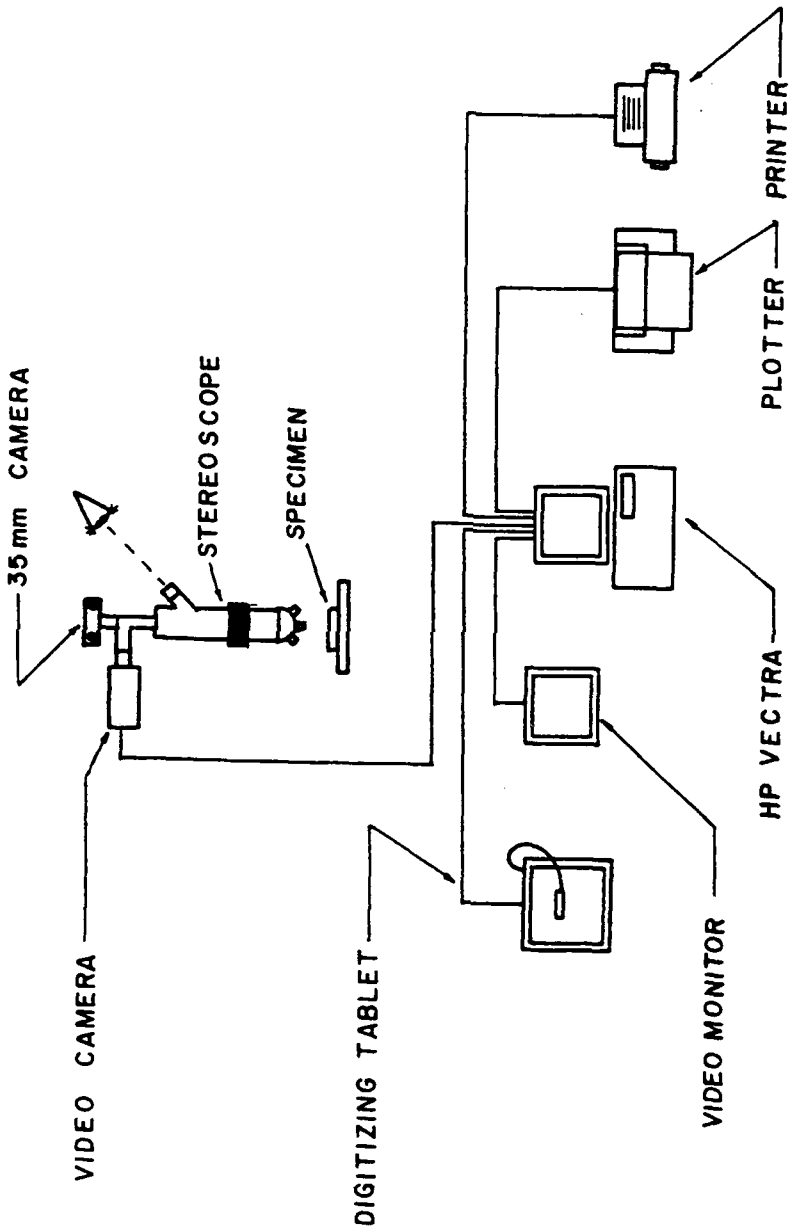


Figure 2.8: Schematic layout of the Image Analyzer

THE feldspar CORPORATION

P.O. Box 98
Spruce Pine, N.C. 28777
Telephone (704) 765-9051

EPK - KAOLIN

Chemical Composition (Dry Basis)

SiO_2	46.50%	MgO	0.16%
Al_2O_3	37.62%	Na_2O	0.02%
Fe_2O_3	0.31%	K_2O	0.40%
TiO_2	0.36%	F_2	0.08%
P_2O_5	0.19%	SO_3	0.21%
CaO	0.25%	V_2O_5	< 0.001%

Loss on Ignition - 13.77%

Free Moisture - 1.43%

Specific Gravity - 2.50 g/cc

Particle Size Distribution

Cumulative % Undersize

40 microns - 100
10 microns - 90
5 microns - 78
3 microns - 68
1 micron - 49
.5 micron - 40
.2 micron - 20

Surface Area

28.1 M^2/gm

Screen Analysis

	Wet* (Lump)	Wet* (Airfloated)	Dry (Airfloated)
+60 Mesh (% Max)	-	-	0.01
+80 Mesh "	0.0	0.0	0.05
+100 Mesh "	0.1	0.1	0.15
+200 Mesh "	0.8	0.8	-
+325 Mesh "	1.5	1.5	-

*Water-washed through screens

Settling Characteristics (10 Wt. % Solids - added 0.1% alum)

Clear liquor 10 Min. - 0.36 in.
Clear liquor 15. Min. - 0.68 in
% Solids (vol.) 48. Hrs. - 28%

Bulk Density

Loose - 32.4 lbs/ft^3

Tapped - 46.2 lbs/ft^3 (6.2 lbs/gal)

Oil Absorption

47.3 gm. oil/100 gm. clay

pH

5 wt. % solids - 6.05
10 wt. % solids - 6.07
20 wt. % solids - 5.85
30 wt. % solids - 5.89

Electrical Conductivity

20 wt. % solids - 110 micro Mho's

Mineral Content (X-ray Diffraction)

Kaolinite ($Al_2O_3 \cdot 2SiO_2 \cdot 2H_2O$) - 97%

Viscosity

30 wt. % solids (TAPPI) - 27 cps
19 wt. % solids (FIRESTONE) - 44 cps

Ceramics Properties — 100% Clay

Water of Plasticity %	38.0
Dry Modulus of Rupture psi	365.0
Linear Dry Shrinkage	6.4
Fired Shrinkage Cone 11 %	11.5
Absorption Cone 11 %	8.0

Casting Properties

Drainage - Excellent
Type of Cast - Plastic
Water Retention - 25.5 %

Cation Exchange Capacity - Methylene Blue

Acid Treated - 5.2 Meq 100 gms
Alkali Response - 5.5 Meq 100 gms
H_2O_2 Reaction - 6.5 Meq 100 gms

Color Values

	L	+a	+b
As received	75.5	1.2	9.0
As received - Compacted	59.9	1.5	7.0
Compacted Fired Core 11	69.2	1.7	4.3

Table 2.1: Composition of Kaolinite Powder

Chapter 3

Micromechanisms of Deformation in Fracture of Overconsolidated Clays

3.1 Introduction

This chapter is devoted to presenting and describing experimental observations. These include photographs, optical micrographs and other related information which illustrate and describe the micromechanisms involved in the fracture process of overconsolidated clays.

3.2 External Observations

Under the cyclic torsional couple, a zone of *damage* steadily propagated around the cylinder. The zone generally consisted of observable discontinuity lines emanating from the horizontal notch plane (the direction of the in plane shearing force) as shown in Figure 1.6. The lines constituting this zone generally deviated at an average orientation of 20° from the notch plane. Note

that these lines represent slip surfaces through the cylinder thickness propagating circumferentially. The envelope of damage could be generally described as a band symmetrically located on both sides of the notch plane. Nearly all the rotational deformation and slip occurs in this band.

It should be noted that, in general, no single slip surface *dominated* enough to be considered a main crack. Rather, a each surface contributed in a fashion such that the damage zone appeared to be made up of a complex arrangement of *branching cracks.*

In order to measure the amount of deformation localized within the damage zone, an experimental technique was developed. This technique involves stamping a grid of orthogonal lines on to the specimen prior to testing. An alcohol based ink is used so that the grid dries while the clay specimen remains moist (refer to Section 2.4). During the cyclic test photographs (taken through the plexiglas wall, silicon oil, and the rubber membrane) are taken during selected stages of the fatigue test.

Photographs illustrating the damage zone evolution through the lagrangian grid system is shown typically in Figures 3.1 and 3.2. All photographs in Fig.s 3.1 and 3.2 are taken while the maximum torque is applied. Notice that the majority of the deformation occurs within the damage zone. Figure 3.3 illustrates similar localized deformation lines in an un–notched specimen. That is, the results from Figure 3.3 show that the localized

deformation can occur in specimens which are fatigued without a prefabricated notch as a focal point for stress concentration.

3.3 Internal Observations

Except for a few cases, most of the internal observations were performed after specimens were impregnated. Some of these exceptions include some preliminary studies performed on a scanning electron microscope (SEM), and a polarizing optical microscope. However, the majority of microscopic studies were optical. These observations were generally performed after the clay specimens were impregnated with bee's wax (Section 2.7), sectioned with a Buehler Isomet Saw (Section 2.8), and polished (Section 2.9).

3.3.1 Undamaged Phase

Before damage can be identified and characterized in the kaolinite, the material must be characterized in its natural state. Actual characterization studies will be presented in Chapter 4, but it is worthwhile to present observations at this point.

Kaolinite particles are *platelet shaped* with typical thicknesses ranging between 50 and 2000 nm and typical diameters ranging between 300 and 4000

nm (5). Of the four most common clay minerals; montmorillonite, illite, chlorite, and kaolinite, kaolinite is the largest.

Clay minerals are formed by one or more of the following processes (4):

1) Crystallization from solutions.
2) Weathering of silicate minerals and rocks.
3) Diagenesis, reconstitution, and ion exchange.
4) Hydrothermal alterations of minerals and rocks.
5) Laboratory synthesis.

The clay soil fabric has been observed and studied by others (1,2,4,5) and a scale hierarchy in the mircostructure has been acknowledged. This hierarchy describes the fashion in which the material fabric is assembled from microscopic particles. Scale hierarchy has been recognized in other materials as well.

Initially, the clay particles (10^{-6} m) are usually aggregated or flocculated together in submicroscopic fabric units called *domains* (10^{-5} m). The domains, in turn, form together what are called *clusters* (10^{-5} to 10^{-4} m). And similarly, the clusters group together to form *peds* or *large clusters* (10^{-4} m). The large clusters are large enough to be seen without a microscope.

A polished section from an untested hollow cylinder after impregnation is presented in Figure 3.4. In this micrograph, the kaolinite appears white and the

impregnation material (bee's wax) appears dark brown). Figure 3.4 shows kaolinite clusters homgenously distributed within a more loosly densified matrix (water intermixed with loose particles of clay) which is now presumed to be replaced by wax. Notice that the observed clusters have a characteristic size on the order of 10^{-4} meters which compares well with that reported by others(1,4,5).

3.3.2 Damaged Phase

Figure 3.5 compares two radial sections (described in Section 2.8) cut from a fatigued cylinder after impregnation; one was sectioned in an undamaged region, and the other in front of the notch tip. The undamaged section in Fig. 3.5a shows the previously reported clusters homogeneously distributed in the wax matrix.

The damaged section in Fig. 3.5b indicates substantial morphological changes due to the applied loading condition. In this region one notices a change in the size distribution of clusters. A few coalesce to produce larger and others split into smaller units. Also, one can see the creation of horizontal strips along which the clusters of clay have aglomerated. Note that these strips constitute surfaces within the cylinder having the orientations shown in Figures 1.6, 3.1, 3.2. The name adopted by the author for the process producing these surfaces is *Localized Strip Densification* or LSD for short.

Additional tests have indicated that when the loading rate is reduced, a higher concentration of damage is attained. This has been experimentally observed in other materials and is in agreement with theoretical considerations (5). For the case considered here, Figure 3.6 shows a series of photographs taken of an axially sectioned specimen which was fatigued at a proportionally lower cyclic load. Three morphologically distinct areas can be identified within the damage zone. The area closest to the notch tip (denoted as Section A in Figure 3.6) shows a network of interwoven densified surfaces. Note that no observable clusters are contained in this region. The second region shows intermixed clusters and densified lines (LSD's) as witnessed earlier in Figure 3.5b. Finally, the third region exhibits vertically oriented densified regions.

All of the above changes in the soil fabric are commonly referred to as damage. Strictly speaking, they are stress induced, morphological transformations of material at the cluster level.

The *broken* grid lines in Figures 3.1 and 3.2 provide experimental evidence that localized irreversible deformation occurs along what appear to be *slip surfaces* emanating from the notch tip. Figures 3.5 and 3.6 present material transformations that occur within the soil fabric inside of the *LSD* zone. In order to provide experimental evidence linking the observed damage on the outside of the cylinder to that occurring within the interior the cylinder, a photograph was taken of the corner showing the exterior and interior fabric.

This photograph is included in Figure 3.7 of this thesis. Note that the interior fabric consists of clusters and loose matrix away from the *slip lines* and consists of *LSD*'s intermixed with clusters within the *slip line* field.

3.4 Other Observations

While preparing the damage zone in Cylinder 3–3 for sectioning, two surfaces at approximately 60° from the horizontal notch plane broke free during handling. The surfaces revealed from this incident indicate what the surface profile along the LSD's may be like. Photographs of these surfaces are included in Figure 3.8 of this report.

Other experimental investigations included fracture studies under a non cyclic fast loading rate. Observations of the stamped grid deformation at various time intervals (every 0.5 msec) were aquired with a Spin Physics SP2000 high speed video camera. The rate of loading (applied torque) was applied in a approximately linear fashion. The maximum torque applied to the cylinder was about 18 Nm and the period over which the load was applied was about 0.25 sec. Results from this study showed a more non cooperative (defined in Chapter 1) fracture behavior than that observed under cyclic fatigue conditions. Photographs of selected cycles from this test are given in Figure 3.9.

Investigations were also performed with the use of an optical polarizing microscope. Various types of slides were prepared for microscopic studies in either transmission or reflection. The optical properties of kaolin have been shown to be bifringent (3). This was reaffirmed with studies performed herein. That is, depending on the relative orientations of the polarizer and analyzer, the kaolin fabric would polarize or become extinct.

Studies were performed on impregnated and un–impregnated samples, as well as fracture surface observations. In general, no difference was noted between the clusters and LSD's. However, both showed a preferred anisotropic orientation.

3.5 Elements of Deformation and Fracture

In summary, the observations to date have shown that undamaged saturated Kaolinite can be considered as made of two phases. The first phase is the continuous water medium (which has been replaced by the wax). The second phase is made of randomly distributed clusters with sizes on the order of 0.1 mm. While it is well known that the particle size is about 0.001 mm, it appears that the stress induced morphological changes can be studied at the level of the cluster; even though it is aknowledged that changes may occur within the clusters. This has implications when one chooses the smallest unit on which experiments can be conducted to study the mechanical behavior of

clay (see Section 4.2).

In the vicinity of cracks and under fatigue loading, morphological changes occur that result in the joining of clusters to form strips (Figs. 3.1 through 3.7). Based on these observations, four micromechanisms responsible for macroscopic deformation can be recognized:

a) Elastic deformation of both the continuous phase and the clusters.

b) Stress induced morphological transformations resulting in the growth or decay of clusters in size and in number within the damage zone.

c) Stress induced morphological transformations resulting in the creation of LSD's and networks.

d) Slippage along the *LSD*'s.

References

1) Attewell, P. B., Farmer, I. W. (1976)."Principles of Engineering Geology," John Wiley and Sons, Inc., New York pp 633 – 696.

2) Chudnovsky, A. (1984). "Statistics and Thermodynamics of Fracture," Journal of Engineering Science, Vol. 22, No 8–10, pp 989–997.

3) Leitch, H. C., Yong, R. N. (1967). "The Rate Dependent Mechanism of Shear Failure in Clay Soils," Soil Mech. Series, No. 21, McGill Univ., Report No. D. Phys. R. (G) Misc. 28, pp 58 – 67.

4) Mitchell, J. K. (1976). "Fundamentals of Soil Behavior," John Wiley and Sons, Inc., New York.

5) Read, H. N., Watson, J. (1962). "Introduction to Geology," Macmillan and Co. pp 637 – 640.

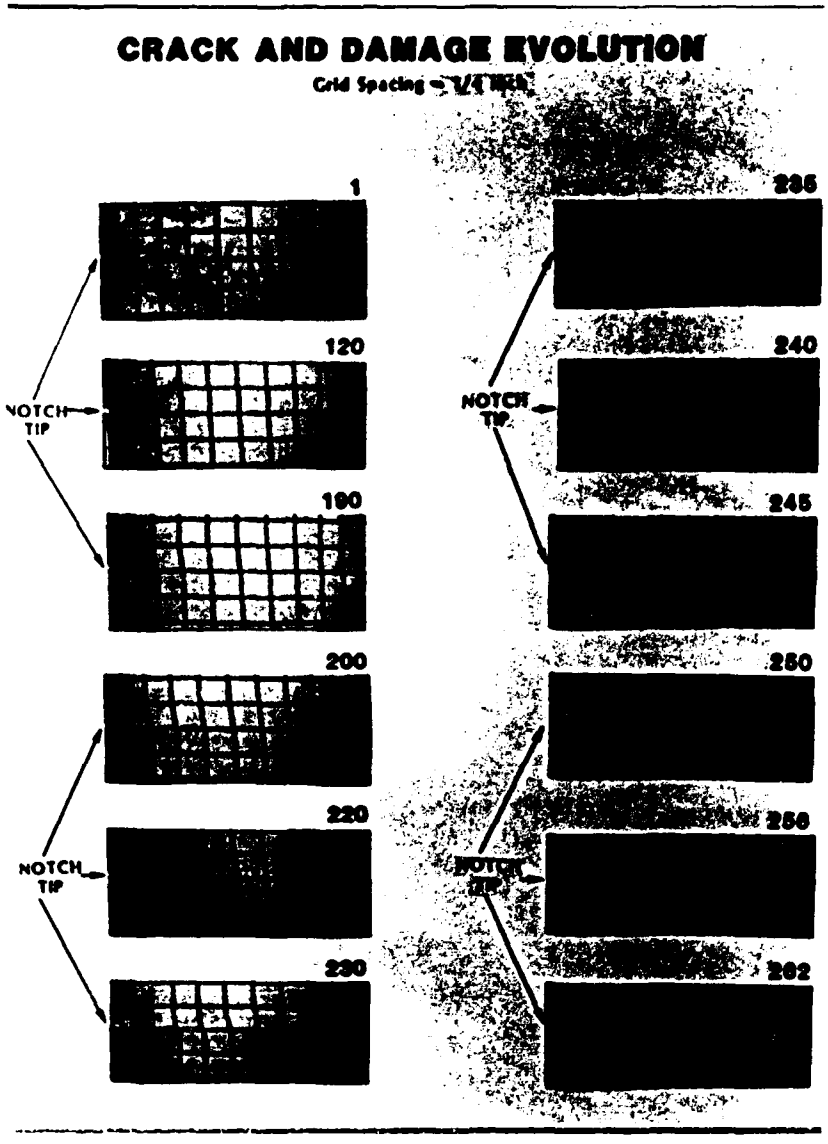


Figure 3.1: Damage zone (*LSD* zone) evolution on hollow clay cylinder illustrating the localized discontinuities for various cycles. Initially the grid lines were unbroken and orthogonally oriented at 1/4" spacings. However, during the fatigue test they provide a means of *measuring* the deformation in the damage

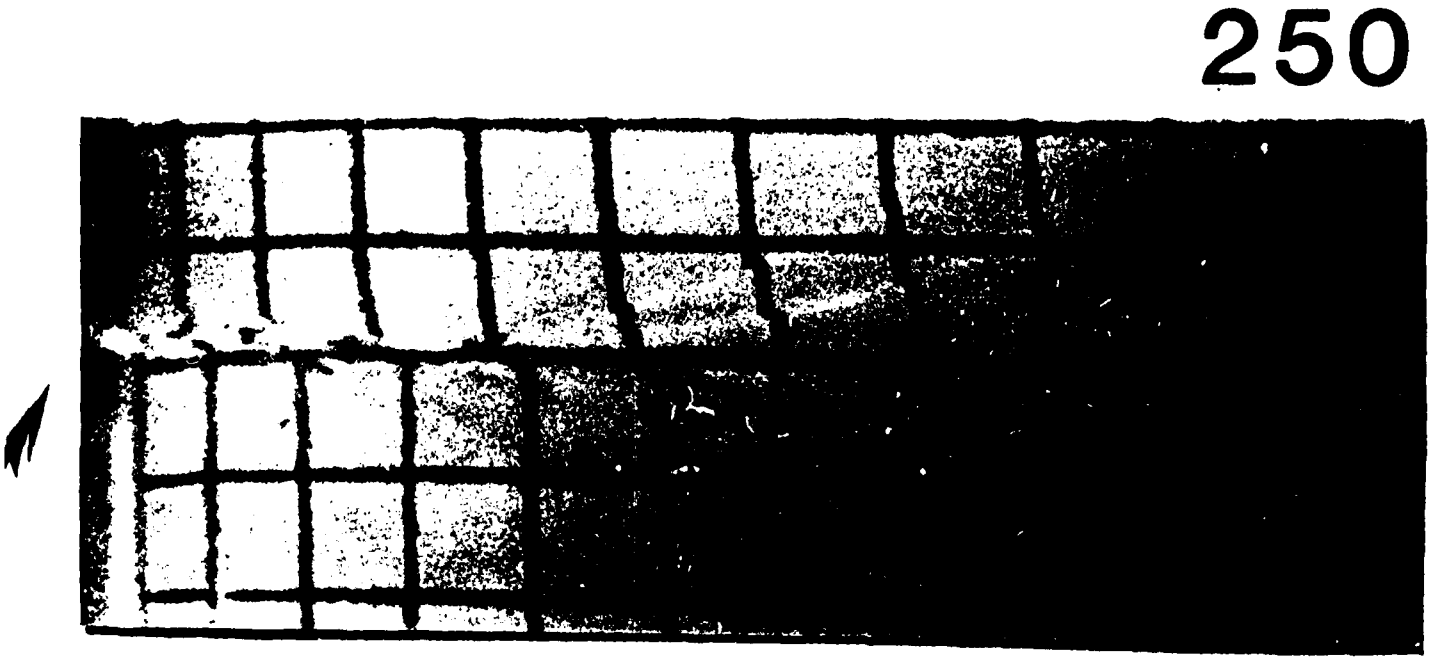


Figure 3.2: Photograph of one of the cycle photographs shown in Figure 3.1 illustrating the localized discontinuities in front of the notch tip.

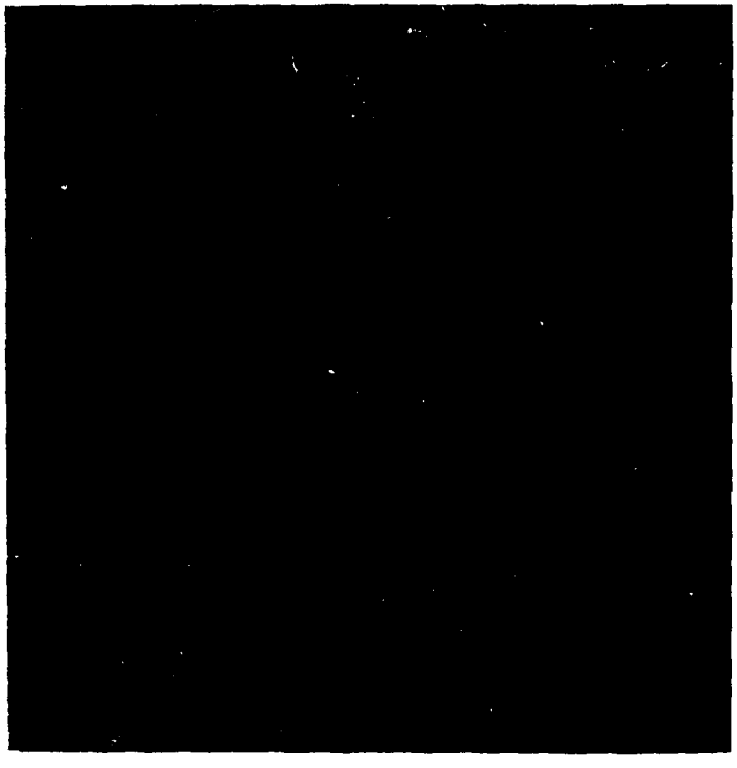


Figure 3.3: Observed discontinuity lines in an un–notched fatigue specimen.

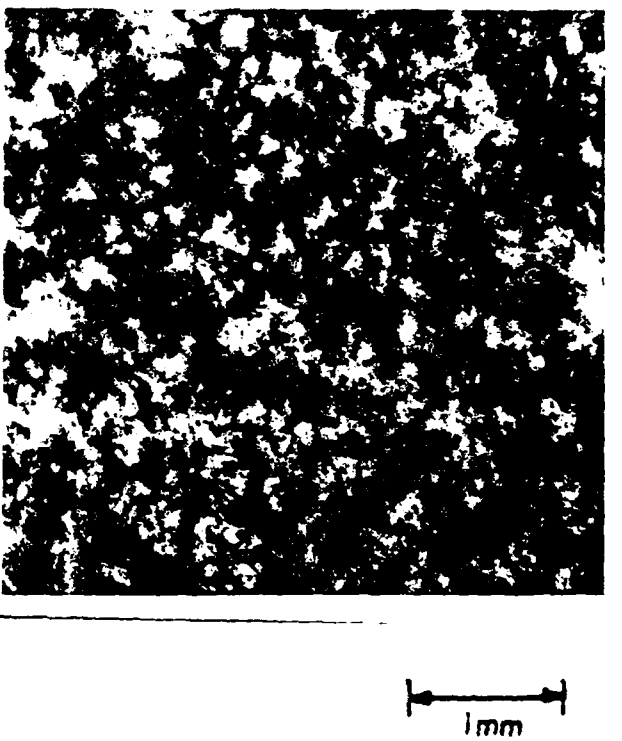


Figure 3.4: Photograph illustrating the internal microstructure of the undamaged clay after impregnation. The Kaolinite is white and the bee's wax is dark brown. Notice the undamaged material is made up of two phases; the dense regions of clay called *clusters*, and water matrix which has been replaced by wax.

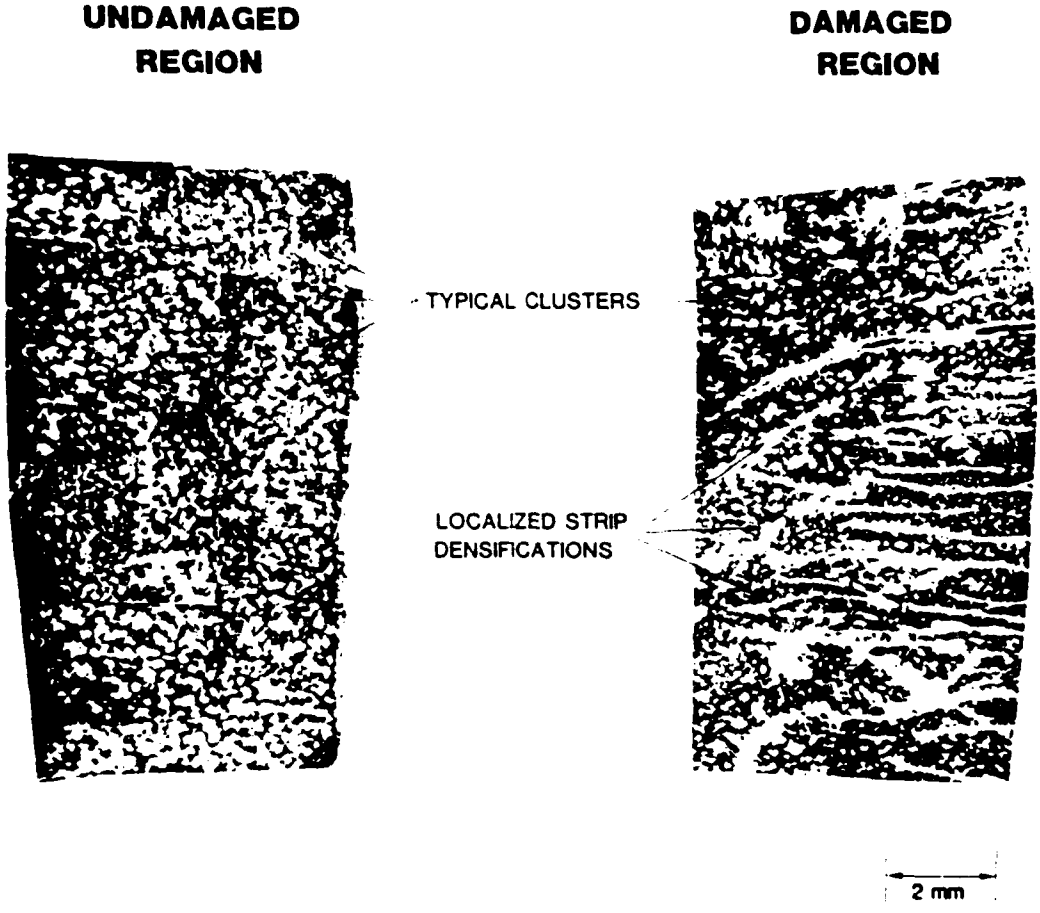


Figure 3.5: Comparison photograph of radial sections in undamaged (left) and damaged (right) regions of a cylinder. Notice that the damaged region shows stress induced morphological transformations of material from a previously undamaged state (clusters), to a damaged state (Localized Strip Densifications).

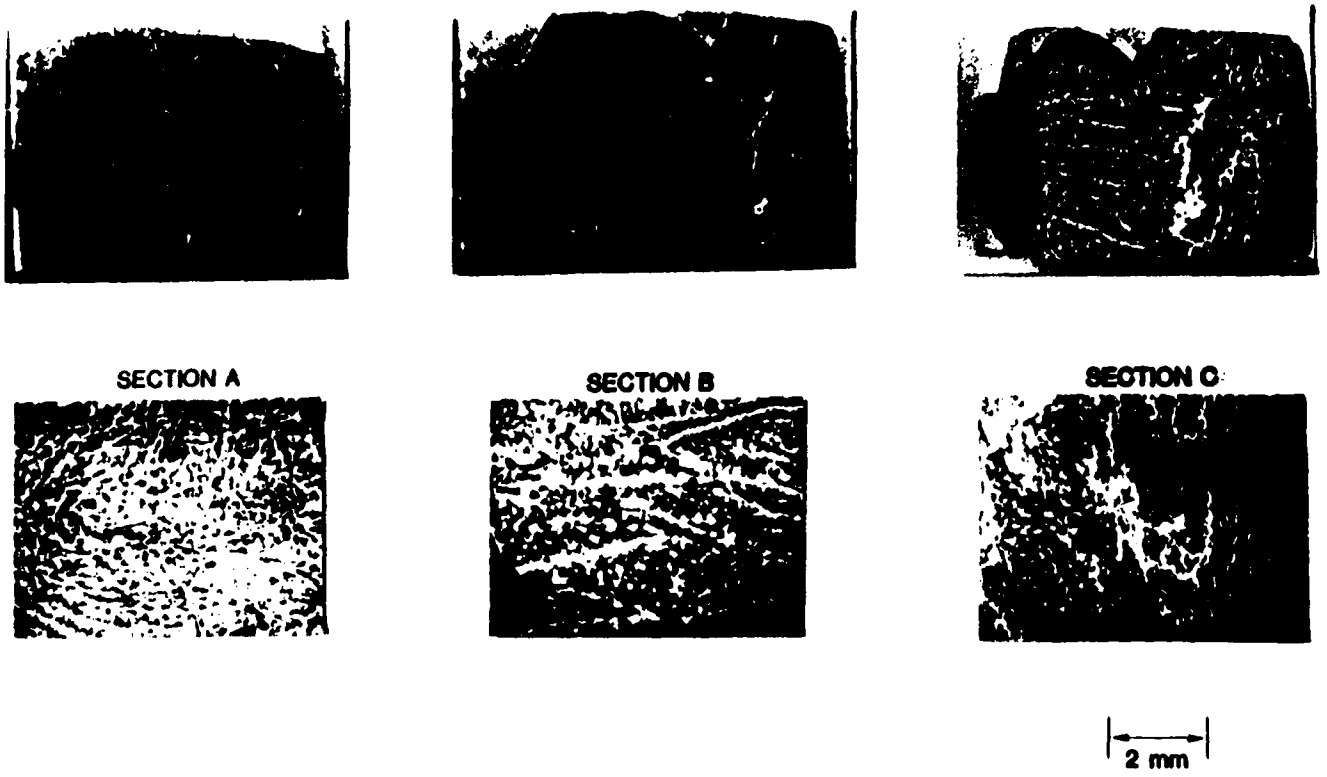


Figure 3.6: Axial section of a fatigued cylinder showing; (a) Network Densifications closest to the notch tip (b) Localized Strip Densifications intermixed with Clusters (c) Other densified regions.

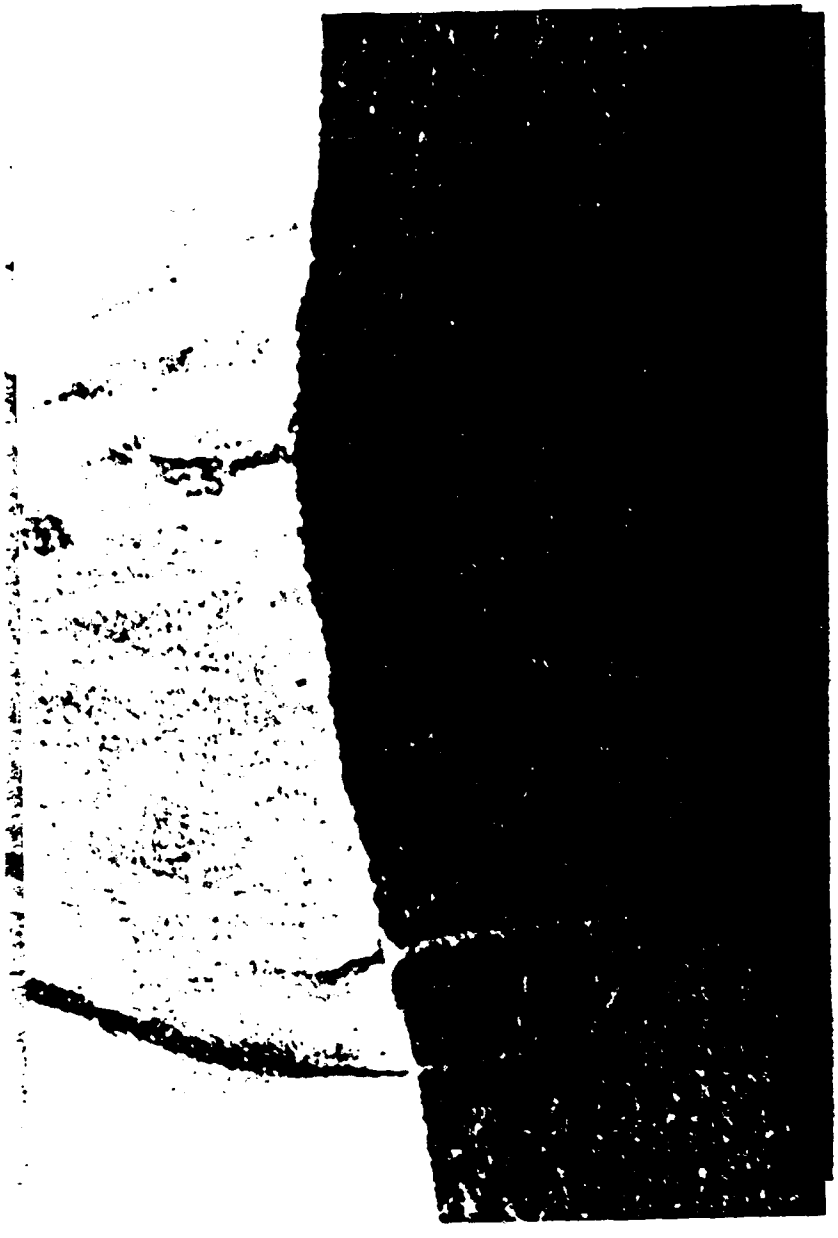


Figure 3.7: Photograph of a corner of an impregnated specimen illustrating slip lines on the outside of the cylinder (left) and *LSD*'s intermixed with clusters in the interior of the cylinder (right).

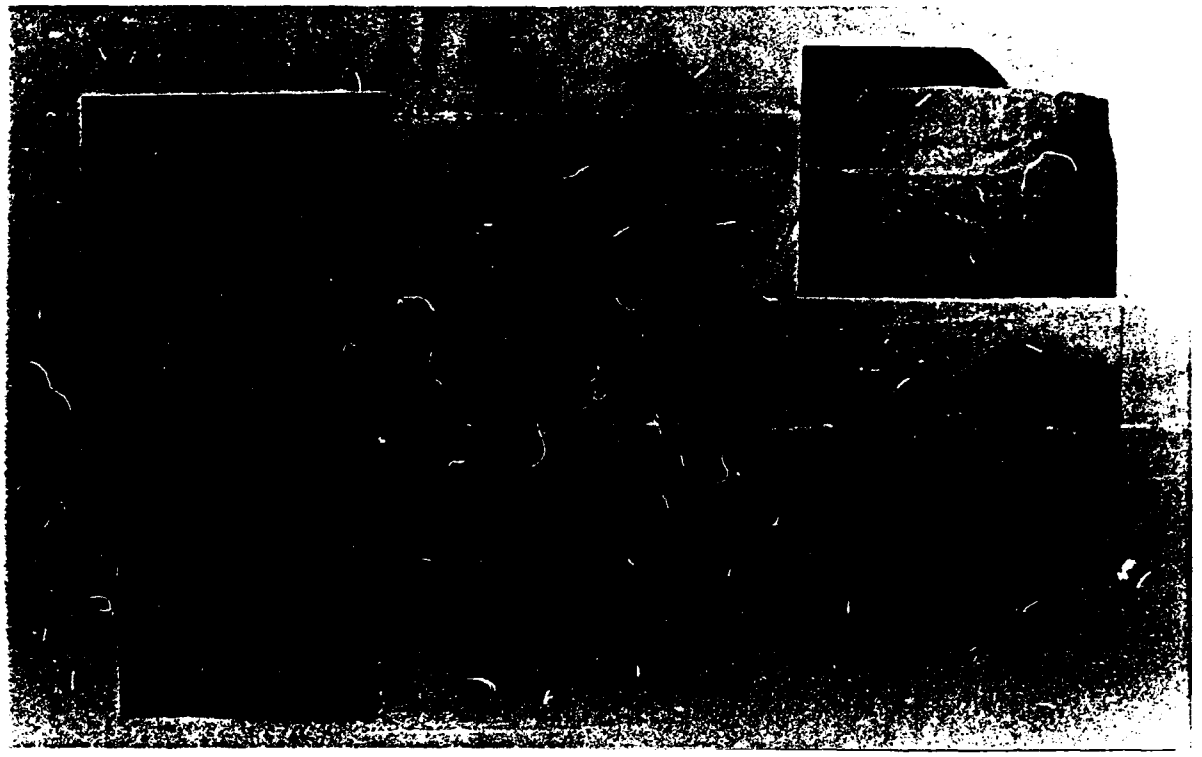


Figure 3.8: Photographs of a slip surface profiles emanating from the notch tip.

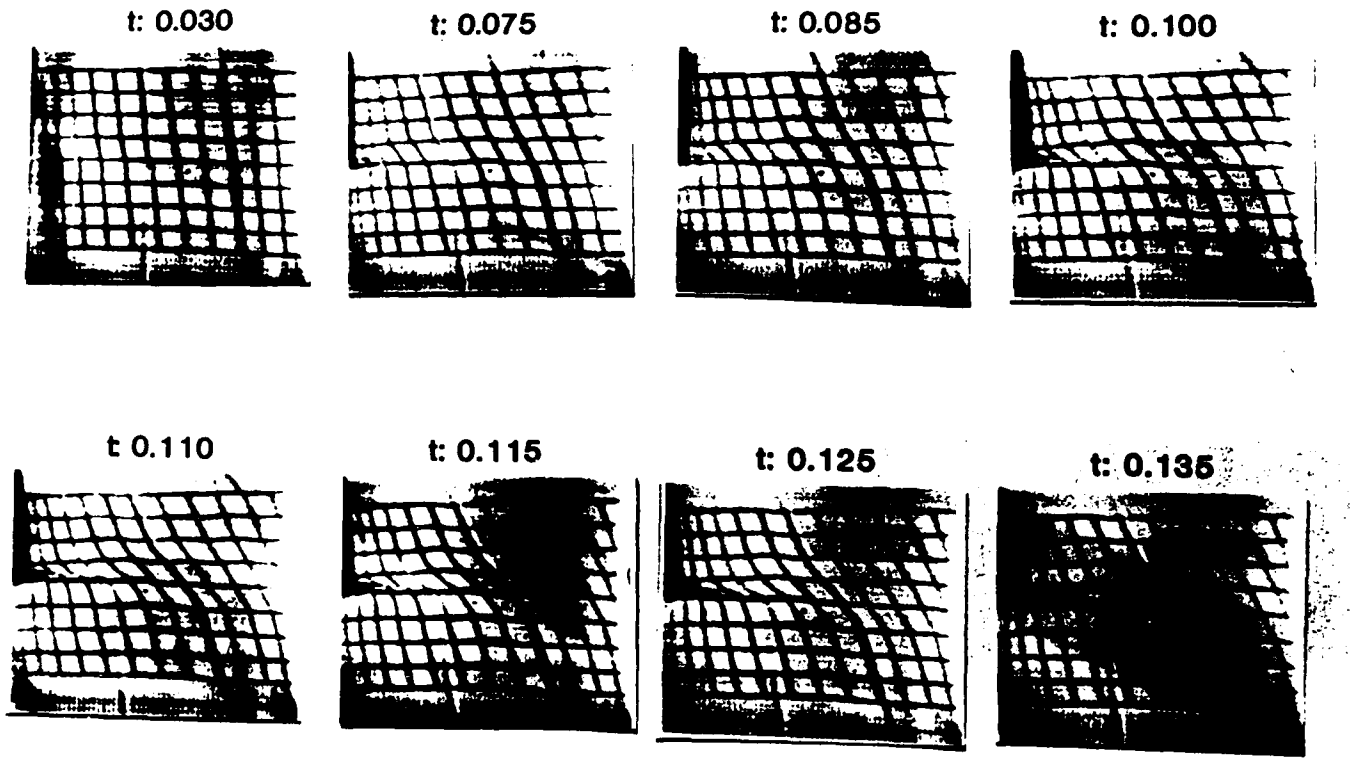


Figure 3.9: Photographs at selected stages from the high speed test.

Chapter 4

Kaolinite Cluster Characterization

4.1 Introduction

Before the damage illustrated in Chapter 3 can be characterized, a complete characterization of the material in its natural state is required. This characterization, in turn, can be used as a reference when observing and analyzing a section of material where damage is expected.

4.2 The Representative Volume

Many continuum theories are based of the assumption that an idealized material can be modeled as a homogeneous continuous medium. In the context of continuum mechanics, the material is typically defined as follows:

> *We disregard the molecular structure of matter and picture it as being without gaps or empty spaces. We further suppose that all mathematical functions entering the theory are continuous functions, except possibly at a finite number of interior surfaces separating regions of continuity. This statement implies that the derivatives of the functions entering the theory*

> *are assumed continuous. This hypothetical continuous material we call a* ***continuous medium or continuum*** (13).

From a material scientists point of view, all real materials exist as an ensemble of individual components at one scale or another. A typical description is given as follows:

> *Two or more atoms, either of the same kind or of different kinds, are, in the case of most elements, capable of uniting with one another to form a higher order of distinct particles called molecules. If the molecules or atoms of which any given material is composed of are alike, the material is a* ***pure substance****. If they are not alike, the material is a* ***mixture*** (1).

In addition to the separate views by which material is defined, further developments in materials analysis and testing have shown two basic types of material properties to exist. These can be described as structurally insensitive and structurally sensitive material properties.

Structurally insensitive properties are those properties that can be accepted as true material properties. This type of property is identified by its average response to a particular flux. Examples of such properties are; permeability, conductivity, density, elastic modulus, etc.

Structurally sensitive properties can be identified as those properties which depend on an extreme response to a particular flux. And since extreme responses are structurally dependant, these properties become sensitive to the

structure. Examples of this type of property are; material strength,etc.(5,6).

The objective of this section is to bridge the two definitions of material by defining the minimum volume (representative volume) of real material that can be considered when continuum mechanics is applied for structurally insensitive material properties. We propose a formal definition of a representative volume (RV) of material and a working definition for the case of overconsolidated clays.

4.2.1 Concept of a Representative Volume

In a continuous medium, material properties such as mass density, permeability, elastic compliance, etc., are ascribed to each point within a body. Similarly, mechanical quantities like stress, strain and energy density can also be ascribed to single point. If, however, the material is heterogeneous, these properties become average values ascribed to a corresponding volume of material. Different philosophies have been proposed to determine effective properties of heterogeneous materials (2,3,10,11,12,14). Some of the more common techniques are known as homogenization and smoothing.

Each of these techniques appreciate the association between material properties and a corresponding minimum volume. Hill (11), provides a rigorous formalism describing a *general free volume*. Klimontovich (12) defines a

correlation volume and radius by introducing a function which is smoothed out over a volume.

In general, a representative volume (RV) for a particular material and with respect to a particular material property can be described as the minimum volume for which the average value of this property does not change when comparison is made with another volume of the same size in the vicinity of the original volume. The size of a RV is determined from the statistical homogeneity of the material. That is, the RV should be large in comparison to any distinct aggregates within the material.

With regard to mechanical properties, it is an accepted rule of thumb that the diameter of a RV of material is approximately one order of magnitude larger than the characteristic inhomogeneity (e.g.. the characteristic aggregate size). In many cases, a rigorous assessment of an RV lacks immediate motivation because the test specimens are many orders of magnitude larger that the aggregates of the material they are made of.

The author believe that the representative volume for a real material is, in effect, that which defines the materials existence in terms of continuum mechanics (7). Thus it is expected that all parameters that are extracted as material invariants (i.e. density, elastic compliance, Poisson's ratio etc.) will have similar distributions and will give rise to the same RV size. This expectation is based on the following concept: the entity which defines the

heterogeneous behavior is dictated by the material of each of the individual components and their respective configuration. Since at the size of the RV, the configuration is essentially indistinguishable and the individual components remain unchanged, all global behaviors will be identically invariant. Therefore, if an RV size for density is defined in clay, it is expected that this same RV size would be representative for elastic compliance or Poisson's ratio, etc.

Rationale for computing an operational definition of an RV involves the relationship between the RV and the size of the test specimen. If the specimen size is not a *prescribed* amount larger than the RV, statistical convergence of various measured data may not transpire. In the following subsections, we propose an approach for determining the size of an RV and report its application to overconsolidated clay.

4.2.2 General Evaluation of a Representative Volume

Let λ be a parameter that characterizes a certain property Λ of the material. For a given volume size V, measurement for λ should be obtained for different elements each of size V. Note that the size of V should be on the order of the characteristic inhomogeneity (i.e. average cluster size). Next, derivation of the corresponding distribution $f_V(\lambda)$ can be ascertained. After the initial distribution $f_V(\lambda)$ is computed for a particular V, the size of V should be increased and its corresponding distribution computed.

It is natural to expect that for a small V the scatter in the values of λ will be high. That is, the distribution $f_v(\lambda)$ will be wide. However, as the size of V increases, $f_v(\lambda)$ *might be* getting narrower with the mean value defined as:

$$[4.1] \qquad <\lambda>_v = \int \lambda \, f_v(\lambda) \, d\lambda$$

tending toward a limit λ^* ($<\lambda> \rightarrow \lambda^*$, as V increases).

If this is the case then Λ can be referred to as a *material property* while taking λ^* for the value of Λ. It should also be noted that when Λ is accepted as a material parameter (i.e. invariant), the corresponding minimum volume of material, V^*, with respect to the specimen shape and size is considered as a **Representative Volume.**

To define the representative volume size V^*, it remains to choose tolerances $\Delta\lambda^*$ and σ^* such that:

a) if $|<\lambda>_v - \lambda^*| \leq \Delta\lambda^*$ then, for practical purposes, $<\lambda>_v$ is indistinguishable from λ^*.

b) if the $\sigma_v \leq \sigma^*$ where σ_v is defined as:

$$[4.2] \qquad \sigma_v = \left\{ \int (\lambda - <\lambda>_v)^2 \, f_v(\lambda) \, d\lambda \right\}^{\frac{1}{2}}$$

If the conditions listed above are met then, for practical purposes, $f_V(\lambda)$ is indistinguishable from a delta–function distribution. Now V^* is defined as the smallest V, for which the conditions in listed in a and b are met.

4.2.3 Representative Volume for Stiff Clays

We now apply the formalism of Subsection 4.2.2 to finding the representative volume for stiff clays. It is found that the average cluster size has a characteristic dimension on the order of 10^{-4} m and that a scale hierarchy exists in the material microstructure (refer to Subsection 3.3.1,9) and that all of the stress induced morphological transformations occur at the cluster level (Section 3.5). Since this hierarchy exists, it is evident that the RV size should be based on the average cluster size and not the size of individual clay particles (see Figure 4.1). As we intend to stay at a level above the cluster level (i.e. the level where individual clusters are not distinguishable), we begin constructing the $f_V(\lambda)$ distributions starting from $V = (0.6 \text{ mm})^3$. At this size of V it is expected that σ_V will be wide.

Next, we select a parameter for λ, with respect to which a RV is sought. For this particular case we choose concentration c of densified clay in the form of clusters (this is observed in the plane of polish and therefore is not the volumetric concentration). The concentration c is computed as the area of

clusters within a region divided by the area of that region: $c = c(L)$ = (percent of cluster area in a square of dimension LxL). The reader should note that what will be computed herein is the representative volume directly associated with cluster concentration. From now on the volume elements are understood to be $LxLxL$ cubes and the letter L will be used in place of V since $V = L^3$.

It is the belief of the author that the observed cluster concentration is directly related to the density of the kaolinite material. That is, the wax penetrates only the loose regions during the impregnation process and simply surrounds the more densely packed regions. Thus, a measurement of concentration is essentially a measure of some arbitrary density.

In a macroscopically homogenous region on a plane of polish, a 12.4 mm x 12.4 mm area was subdivided into an L_1xL_1 square mesh with $L = 0.62$ mm (i.e. subdivided into 20 squares by 20 squares). Next, six successive sizes for L were defined as $L_k = k^*L$ $(k = 1,...,6)$. For each of the six sizes, nine non–overlapping L_kxL_k squares were chosen (each consisting of k^2 elementary L_1xL_1 squares of the mesh).

Cluster concentration was measured and recorded for each of 400 squares in a mesh pattern using the image analyzer (see Section 2.10). A computer program was written to perform the following computations: For each k=1,...,6, the value of c was computed for nine neighboring squares and their distribution $f_k(c) = f_{L_k}(c)$ was constructed. A Fortran listing of the

program written to process this information for the six successive sizes (L_k) is provided in Appendix A and is named REPVOL. Gaussian approximations for the six cases are shown in Fig. 4.2. Figure 4.3 shows illustrates how the standard deviation in concentration decreases as the size of the volume analyzed is increased. Also the interpolation determined in Fig. 4.3 was used to sketch the 3–D plot in Fig. 4.4 Table 4.1 gives more numerical details from the six successive analyses. The cluster concentration data base (obtained with the image analyzer) used in this analysis is listed in Appendix B.

Since the measurements obtained from the image analyzer themselves contain a certain amount of fluctuation, a true *delta function* is unachievable. In order to assess the magnitude of the error associated with these particular measurements, a typical cluster was singled out and 25 successive area measurements were performed on it. The successive measurements resulted in a 6% variation about the mean (i.e. the ratio of the standard deviation to the mean measurement was approximately equal to 0.06) Therefore, a cluster concentration distribution with a standard deviation less than 3% could never be expected regardless of the size of the volume analyzed (assuming a mean concentration value equal to 0.5).

In order to assess the size of the representative volume, the tolerances for $\Delta\lambda^*$ and σ^* must be chosen. The results from Table 4.1 indicate that if a tolerance in the mean concentration is selected as $\Delta c^* \leq 3\%$ and the variation of c is on the order of the measurement error, then the characteristic dimension of

the representative volume for this material is somewhere in the neighborhood of 3 to 4 mm (i.e. V^* is somewhere between 27 mm^3 and 64 mm^3). Alternatively, if a volume smaller than V^* is chosen, the information in Table 4.1 and Figure 4.3 can be used to estimate the associated error which corresponds to the *smaller volume.*

4.3 Reconstruction of the Cluster Spatial Distributions

In this section we will investigate techniques for reconstructing the spatial distributions of clusters. Studies of this type motivate attention for the following reasons:

- Comparison of spatial distributions of clusters between an area away from clusters in an undamaged region to that of a damaged region may show changes resulting from stress concentrations.
- Comparisons of spatial distributions of clusters on orthogonal planes may indicate whether the material behaves isotropically or whether some anisotropy exists at the cluster level. Even if the individual clusters depict some anisotropic behavior, the material may behave isotropic if the clusters are randomly oriented. Such a study may define if the material is statistically isotropic at the cluster level.

Spatial reconstruction involves using specified information from a *plane of polish* together with certain assumptions (usually regarding the particle shape and orientation) to make predictions about the volumetric distributions. For our particular case, the objective is to reconstruct the spatial distribution of a

pclydispersed system of clusters homogeneously distributed in an opaque medium (bee's wax) given information from a planar section randomly passing through the medium. Before any relationships can be formulated linking the spatial distribution of a particular volume to the observed distribution of *spots* obtained from a planar section, the cluster shapes must be approximated with some regular shape.

The first and simplest type of polydispersed system consists of a system of spherical particles. Cahn and Fullman (4) derived a method for obtaining distribution of sphere diameters and plate thicknesses from the size distribution functions obtained along randomly oriented lines. Saltykov (15) derived and tabulated a set of coefficients which linearly relates the size distribution of circles in a planar section to that of spherical particles homogeneously distributed in the material.

Saltykov assumed the spatial distribution to consist of a prescribed number of classes in a given volume. Each class represents a specific particle size in the 3–D distribution. With each class, he associates an *unknown* volumetric density from this class ($\#/m^3$). In the paper (15), the particle sizes are divided into any number of classes up to 15. Next, Saltykov relates the above (up to 15) to the diametrical distribution of *circular spots* obtained by cutting the volume of material with a plane in a random fashion.

Dehoff (8) provides a shape factor correction which can be applied to the

coefficients Saltykov furnished. Dehoff arrives at a shape factor correction by considering the spatial distribution to consist of a mixture of either prolate or oblate spheroids instead of spheres. Prolate and oblate spheroids are both classes of ellipsoids generated by revolving ellipses about either the major (for prolate) or minor (for oblate) axis. Use of Dehoff's solution requires that the researcher know the particle type (either prolate or oblate), and the aspect ratio (the ratio of the minor divided major axis) a priori.

4.3.1 The Reconstruction Model

Outlined here, is the formalism for computing the relation between a polydispersed system of prolate and oblate spheroids homogeneously distributed and randomly oriented, and a two dimensional distribution of elliptical spots resulting from a plane of polish randomly passed through the medium. This formalism follows closely to that of Dehoff's except that it's generalized to include the mixture of both prolate and oblate ellipsoids and some notation changes were made to accommodate for the mixture.

In Saltykov's method, the researcher is required to sort data obtained from the planar section and compute the frequency of intersecting a spherical particle and obtaining a circular spot between radii r and $r+\Delta r$. Similarly, Dehoff requires that the data be sorted according to the frequency of intersecting a prolate (oblate) particle and obtaining an elliptical spot with a

minor axis between b and $b+\Delta b$ (a and $a+\Delta a$).

For the case considered here, we require that the data be sorted according to their *equivalent radii* obtained from elliptical spots between a range r and $r+\Delta r$. In general, lower case letters will be used to describe 2–D features in the plane of polish and upper case will be used for 3–D features in the spatial distribution.

4.3.1.1 3–D Notation

This subsection defines the variables which describe the spatial distribution of particles. A more detailed description revealing how these variables relate will be given in a later derivation.

- K^{ϵ} – Number of discrete classes of ellipsoids of the "ϵ" type.
  ϵ=0 → oblate
  ϵ=1 → prolate
- A – Semi–major axis of an ellipsoidal particle.
- B – Semi–minor axis of the ellipsoidal particle.
- Q – Aspect ratio of ellipsoidal particle ($Q = B/A$).
- R_j^{ϵ} (j=1,...,K^{ϵ}) – Equivalent radius of an ellipsoid of the "ϵ" type and the "j" class (size).

  For ϵ=0 (oblate) type ellipsoid
  $4/3\ \pi R^3 = 4/3\ \pi A^2 B$

$$A = R/Q^{\frac{1}{3}}$$

$$B = Q^{\frac{2}{3}}R$$

For ϵ=1 (prolate) type ellipsoid
$4/3\ \pi R^3 = 4/3\ \pi AB^2$

$$A = R/Q^{\frac{2}{3}}$$

$$B = Q^{\frac{1}{3}}R$$

- $N_j^{\epsilon}(R)$ – Number of centers of "ϵ" type ellipsoidal particles with an equivalent radius R_j^{ϵ} in a unit volume.

4.3.1.2 2-D Notation

Here we define the variables which describe the information which can be extracted from a plane of polish (test plane).

- a – Semi–Major axis of an elliptical spot
- b – Semi–Minor axis of an elliptical spot
- q – Aspect ratio ($a = b/a$)
- k^t – Total number of classes upon which a histogram of equivalent radii is constructed. This is a user selected parameter which generally reflects the accuracy of the reconstruction. In general, $k^t \geq \Sigma\ K^{\epsilon}$ (ϵ=0,1). If $k^t = \Sigma\ K^{\epsilon}$, then a linear system of equations will be solved in the solution process. Otherwise a least squares error approach will be used to solve the overdefined system of equations.
- r_i (i=0,...,k^t) – Equivalent radius of the elliptical spot of the "i^{th}" size.

- $n_i(r)$ (i=0,...,k^t) — Number of centers of elliptical spots with equivalent radii greater than r_{i-1} and less than or equal to r_i in a unit area.

4.3.1.3 Size Distribution Relations

Consider an aggregate mixture of prolate and oblate particles dispersed in an opaque matrix. Let the index j refer to the size of the aggregates for a particular type (ϵ) of ellipsoid. Similarly, let the index i refer to the size of ellipse resulting from the intersection of a test plane and a particle. This is shown for the case of a prolate particle in Figure 4.5.

Next, let the particles be divided into K^ϵ (ϵ=0,1) classes for the oblates and prolates respectively. Also let the increment between each size of oblate and prolate particle be given by Δ^ϵ with Δ^0 not necessarily equal to Δ^1. If R^ϵ_{max} depicts the maximum *sizes* for the distributions then:

[4.3] $$\Delta^\epsilon = \frac{R^\epsilon_{max}}{K^\epsilon} \quad (\epsilon=0,1)$$

It is assumed that all sections are represented in a test plane (i.e. we will analyze a test plane no smaller than a *representative volume* size). Thus it is assumed that R^ϵ_{max} can be computed from the dimensions of the *largest ellipses* in the test plane. After R^ϵ_{max} are established, the number of classes of prolates and oblates can be assigned and Equation 4.3 can be evaluated.

The centers of ellipsoids which produce the subclass of sections lie in a volume which has a cross sectional area equal to the area of the test plane and a thickness equal to $\rho_{i-1} - \rho_i$ on both sides of the test plane. For a unit area of test plane, this volume is given by:

[4.4] $$V(i,j,\phi) = 2(\rho_{i-1} - \rho_i)(1 \text{ unit})^2$$

Where ρ denotes the distance from the center of the ellipsoid to the intersecting test plane. The relationship between ρ and r (the equivalent radius of the ellipse resulting from the intersection of the ellipsoid and test plane) can be determined from pure analytic geometry. That is, for an ϵ type ellipsoid of size R_j^ϵ, and a shape Q, there exists one value for the distance ρ_i from the center of the ellipsoid to the test plane.

Let the $X'-Y'-Z'$ coordinate system be oriented in such a way that the Y' axis coincides with the axis of revolution and the origin is located at the center of the ellipsoid. Next let the $X-Y-Z$ coordinate system be located such that it's origin is also located at the ellipsoid center but oriented so that the $X-Z$ plane is parallel to the test plane. Denote the angle between the Y and Y' axes by ϕ. This is shown in Figure 4.5 for the prolate case. The equation for a prolate ellipsiod is given by:

[4.4] $$\frac{x'^2}{B_j^2} + \frac{y'^2}{A_j^2} + \frac{z'^2}{B_j^2} = 1$$

The coordinate transformation equations are given by:

$$x' = x\cos\phi - y\sin\phi$$
$$y' = x\sin\phi - y\cos\phi \qquad [4.5]$$
$$z' = z$$

And the equation defining the test plane is

$$y = -\rho_i \qquad [4.6]$$

Next combining equations 4.5 and 4.6 with 4.4 and describing A_j and B_j in terms of R_j and Q, along with describing a_i and b_i in terms of r_i and q yeilds:

$$\rho_i = \sqrt{\frac{r_i^2/\sqrt{l} - (Q^*)^{\frac{2}{3}}R_j^2/l}{(n - ml/l^2}} \qquad [4.7]$$

where:

$$l = \cos^2\phi + Q^*\sin^2\phi$$
$$m = \sin^2\phi + Q^*\cos^2\phi$$
$$n = (1-Q^*)^2\sin^2\phi\cos^2\phi$$

and:

$$Q^* = \begin{Bmatrix} Q & \rightarrow \epsilon=1 \\ 1/Q & \rightarrow \epsilon=0 \end{Bmatrix}$$

The planar density $n(\epsilon,\phi,j,i)$ is related to the spatial density N_j^ϵ by using substituting the results from Equation 4.7 into Equation 4.3 to obtain:

[4.8] $$n(\epsilon,\phi,j,i) = N_j^\epsilon V(i,j,\phi)=2(\rho_{i-1} - \rho_i)$$

Next, we average over all possible orientations to obtain:

[4.9] $$n_i = \sum_{\epsilon=0}^{1} \sum_{j=1}^{K^\epsilon} C_j^\epsilon N_j^\epsilon$$

where C_j^ϵ is defined as:

[4.10] $$C_j^\epsilon = \int_0^{\pi/2} 2(\rho_{i-1} - \rho_i)\ sin\phi\ d\phi$$

The coefficients C_j^ϵ can be determined for a specified ellipsoid type (ϵ) and class (j^{th}) using by integrating Equation 4.10 and computing the necessary values of ρ from Equation 4.7.

4.3.2 Particle Distributions

Note that Equation 4.9 relates the densities of a volumetric distribution (N_j^ϵ's) to that of a planar distribution (n_i's). In theory, if a large enough area of spots (clusters) is used to construct the 2–D distribution (i.e. a representative sample of data), and if the coefficients C_j^ϵ in Equation 4.10 can be inverted, the unknown volumetric densities can be written in terms of the known planar densities. Futher, if the planar densities are refined into a larger number of classes than the corresponding spatial distribution (i.e. $k^* > K^\epsilon$), an overdefined system of equations results and a least squares approach can be used to solve for the unknown spatial densities.

A fortran program named MORPHOL was written to perform such analyses. A listing is included in Appendix A of this report. Required input data for this routine includes; a distribution of planar spots with specified equivalent radii (n_i's), a number of classes for each ellipsoid type (K^ϵ), the aspect ratio of each ellipsoid type (Q^ϵ), and the size range of each ellipsoid type (R_{max}^ϵ). The output from MORPHOL includes spatial densities for each class of the particle distribution (N_j^ϵ), distribution plots, and total volumetric cluster concentration.

4.3.2.1 Error Analysis

Before a spatial reconstruction scheme can be used with confidence, an understanding of the reconstruction accuracy must be investigated. The purpose of this subsection is not to perform an elaborate error analysis study. Instead, the objective here is to outline how such an analysis could be performed and to present some preliminary results.

In this type of reconstruction scheme, different types of errors can be introduced. Some of these include;

- Error in approximating a *cluster* with a regular shape (ellipsoid, or sphere).
- Error in prescribed variables (R^{ϵ}_{max}, K^{ϵ}, Q^{ϵ}).
- Error in assigning a descrete spatial distribution for a distribution which may be more appropriately modeled as continuous.
- Measurement error in $n_i(r)$.
- Error in assuming random particle orientation (not applicable to spheres).

Some of these error types can be investigated by first defining some *artificial* spatial distribution of particles of some regular shape (spheres or ellipsoids). A two dimensional distribution of elliptical spots can then be generated directly from the relation presented in Equation 4.10 (using the coefficients C^{ϵ}_j's directly). The 2–D distribution, in turn, can be used to

reconstruct a 3–D distribution while introducing *error* in some prescribed variables. Afterward, the new 3–D distribution can be compared to the original one and a resulting error in the 3–D distribution can be computed.

This technique is illustrated for the case of approximating particle shapes. An initial particle distribution consisting of 4 classes of spheres was selected and is shown in Figure 4.6. Next, a corresponding 2–D distribution of spots of equivalent radii was generated. Afterward, the 2–D distribution was used along with either prolate or oblate spheroids with varying aspect ratios to reconstruct a new 3–D distribution. Resulting errors in the mean and standard deviation of the 3–D distributions were tabulated. These analyses were performed on for three systems of equations; a linear system (4 equations – 4 unknowns), 50% overdefined system (6 equations – 4 unknowns) and 100% overdefined (8 equations – 4 unknowns). These results are plotted in Figure 4.7

4.3.2.2 Cluster Volumetric Concentration (Saltykov's Approximation)

A 4.0 mm x 4.0 mm cross section of material (i.e., a representative sample) of material was axially sectioned and polished from a cylinder which was not fatigued (*undamaged* specimen). This region is very similar to that photographed in Figure 3.4. Clusters within the 4.0 mm x 4.0 mm section were traced with the image analyzer and planar densities of cluster size and aspect ratio were recorded. This information is given in Figure 4.8. Next, the cluster

planar information was input into the MORPHOL routine and spatial distributions were constructed. These distributions were generated for 5, 10, and 15 classes and results are presented in Figures 4.9, 4.10, and 4.11, respectively.

It should be noted that the results presented in Figures 4.9 through 4.11 essentially show that a planar cluster concentration of about 50% translates to a volumetric cluster concentration of about 35%.

References

1) Bean, Howard S. (1978). "General Properties of Materials", Mark's Standard Handbook for Mechanical Engineers, 8th ed., McGraw Hill, New York, pp 6–2.

2) Beran, Mark J. (1968). "Statistical Continuum Theories," Monographs in Statistical Physics and Thermodynamics, Vol. 9, John Wiley and Sons, New York.

3) Burgess, R. (1965). "Fluctuation Phenomena in Solids," Academic Press, New York

4) Cahn, J. W., Fullman, R. L. (1956). "On the Use of Lineal Analysis for Obtaining Particle Distribution Functions in Opaque Samples, " Trans. AIME, pp 610 – 612

5) Chudnovsky, A. (1984). "Statistics and Thermodynamics of Fracture," Journal of Engineering Science, Vol. 22, No 8–10, pp 989–997.

6) Chudnovsky, A., Perdikaris, P. C. (1983). "On the Scale Effect of Fracture In Concrete," Proc. 4th Intl. Conf. on Applications of Statistics and Probability in Soil and Structural Engineering, Vol. 1, pp 407–419, Florence Italy

7) Chudnovsky A., Saada, A., Lesser, A. J. (1988) "Micromechanisms of

Deformation and Fracture in Overconsolidated Clays – Part 1," Canadian Geotechnical Journal, May, 1988 pp –

8) Dehoff, R. T. (1962) "The Determination of the Size Distribution of Ellipsoidal Particles from Measurements made on Random Plane Sections," Trans. AIME 224, pp. 474 – 477.

9) Eberhardt, J. J., Hay, J. P., Carpenter, J. A. (1985). "Materials by Design – A Hierarchial Approach to the Design of New Materials," Proc. Mat. Res. Soc. Sym. Vol. 63, Boston Mass.

10) Fishman, L., McCoy, J. J. (1980). "Homogenization and Smoothing: A unified View of Two Derivatives of Effective Properties and Extensions," Journal of Applied Mechanics, ASME, Vol. 47, pp. 679 – 682.

11) Hill, Terrel L. (1956). "Statistical Mechanics Principles and Selected Applications," McGraw Hill, New York pp. 679 – 682.

12) Klimontovich, Yu. L. (1986). "Statistical Physics", Howard Academic Press, New York, pp 152 – 163.

13) Malvern, L. E. (1982). "Introduction to the Mechanics of a Continuous Medium", Prentice Hall Inc., Englewood Cliffs, N. J.

14) McCoy, John J. (1979). "On the Calculation of Bulk Properties of Heterogeneous Material ", Quarterly of Applied Mathematics, July, pp. 137 – 149.

15) Saltykov, S. A. (1967). "The Determination of the Size Distribution of Particles in an Opaque Material from Measurements made on Random Plane Sections," Proc. Second Int. Congress for Stereology, New York: Springer – Verlag 163

16) Underwood, E. E. (1970). "Quantitative Stereology," Addison Wesley Publishing Co., Reading Mass.

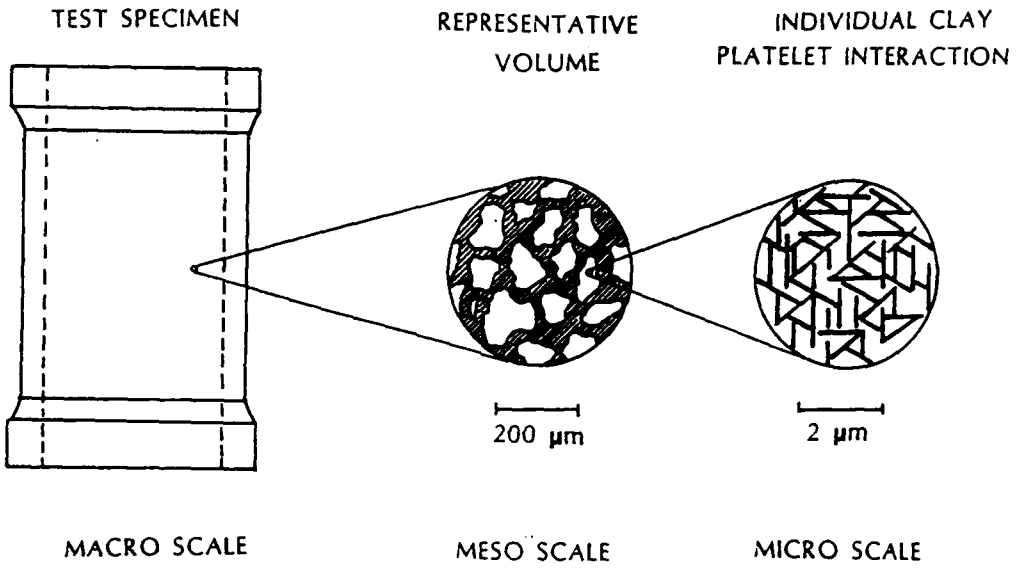


Figure 4.1: Scale hierarchy in overconsolidated clay.

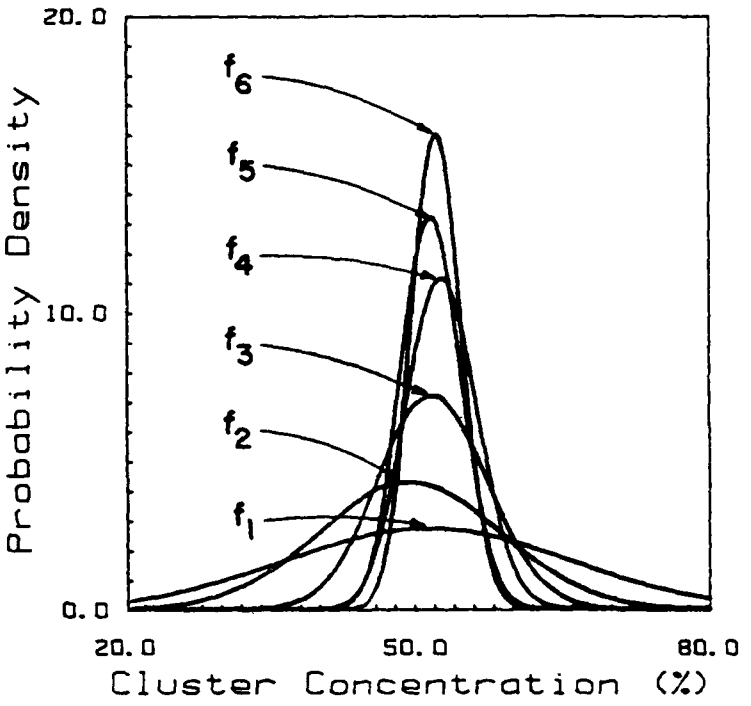


Figure 4.2 A plot of the distributions $f_k(c)$, for (k=1,...,6) and L_k=k*0.62 mm.

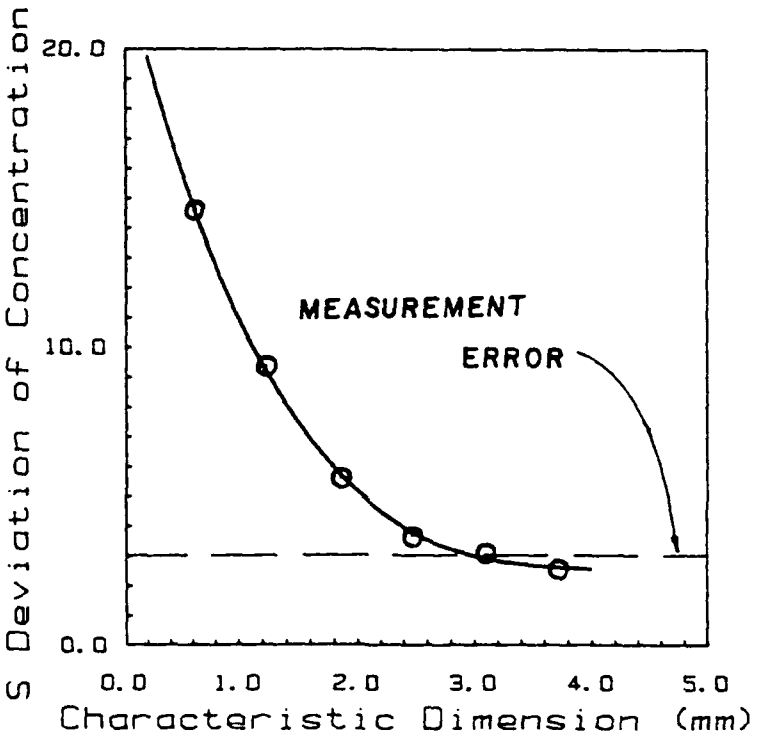


Figure 4.3 A plot of the standard deviation (σ_v) as the characteristic dimension L_k increases.

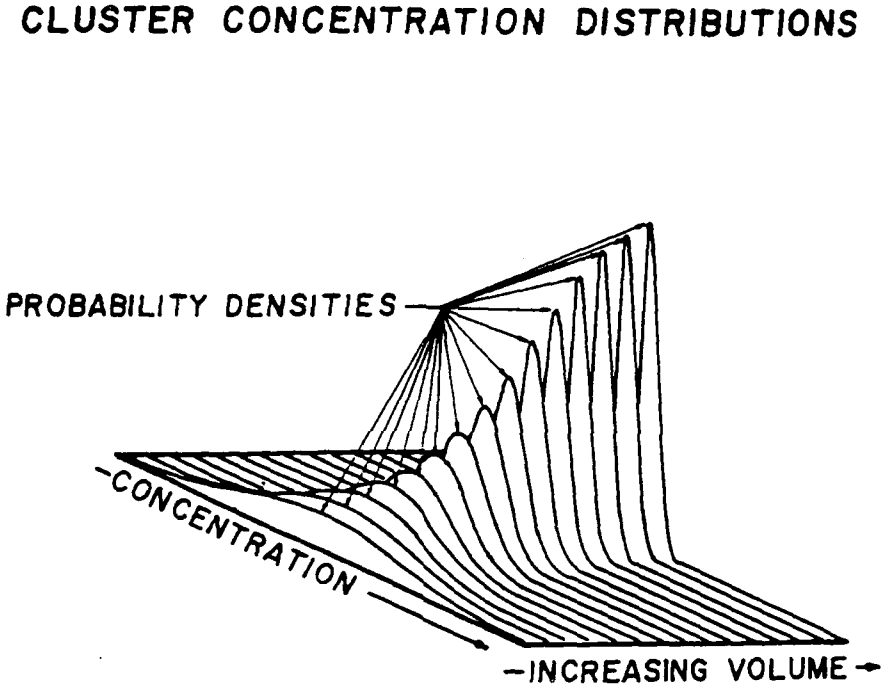


Figure 4.4 Three dimensional plot of the concentration distributions as the characteristic dimension is increased.

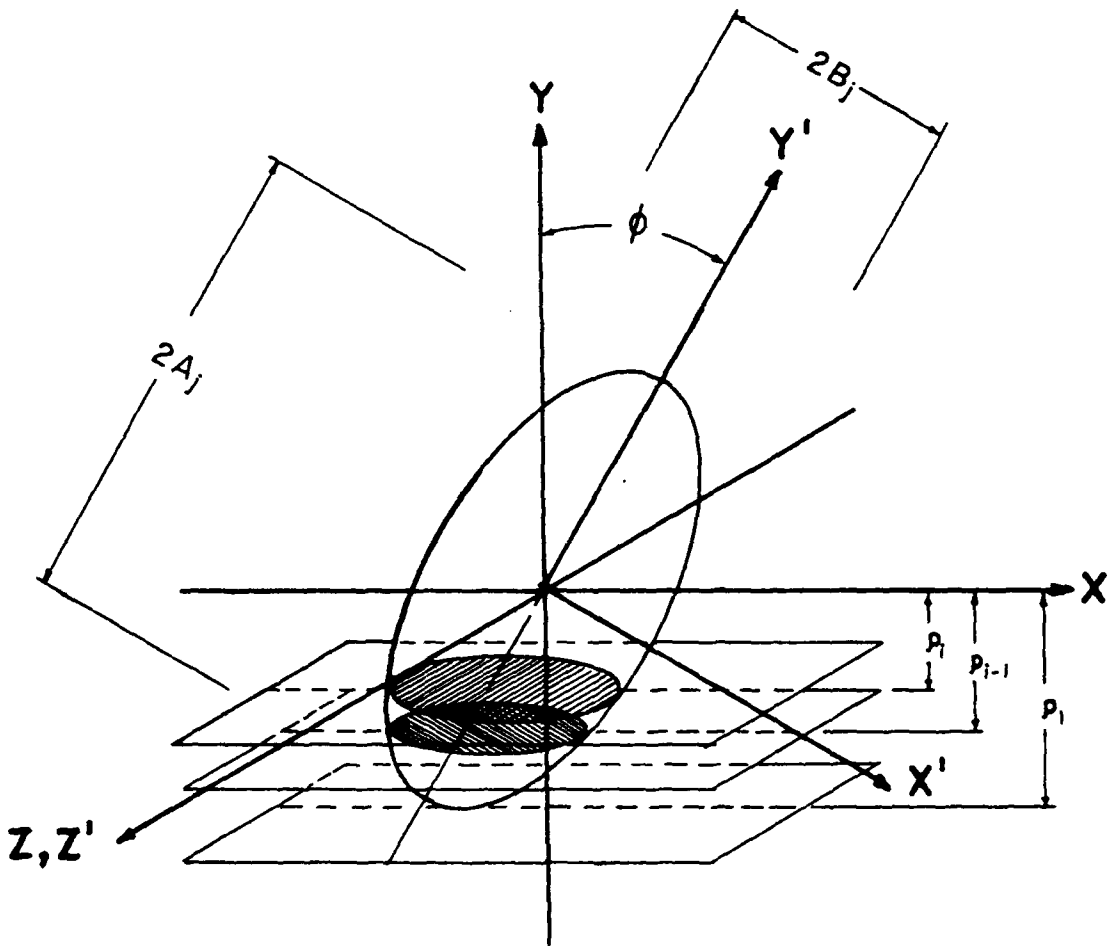


Figure 4.5 A prolate ellipsoid of revolution, showing the range of distances from the center to a sectioning plane which produce intersections in the class *i*.

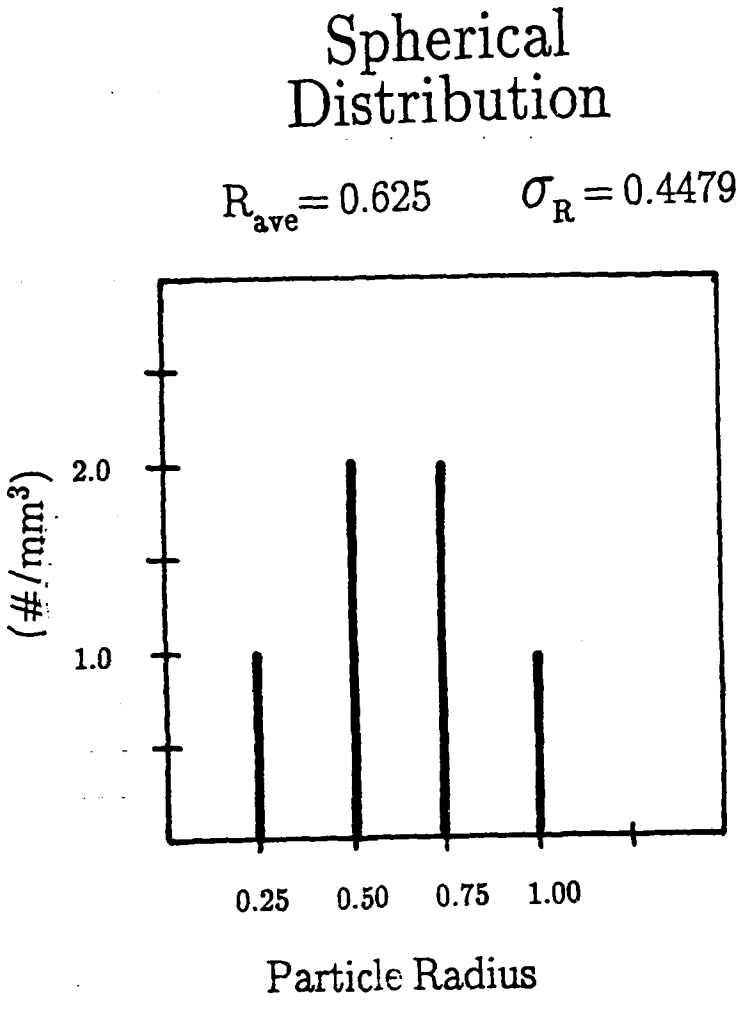
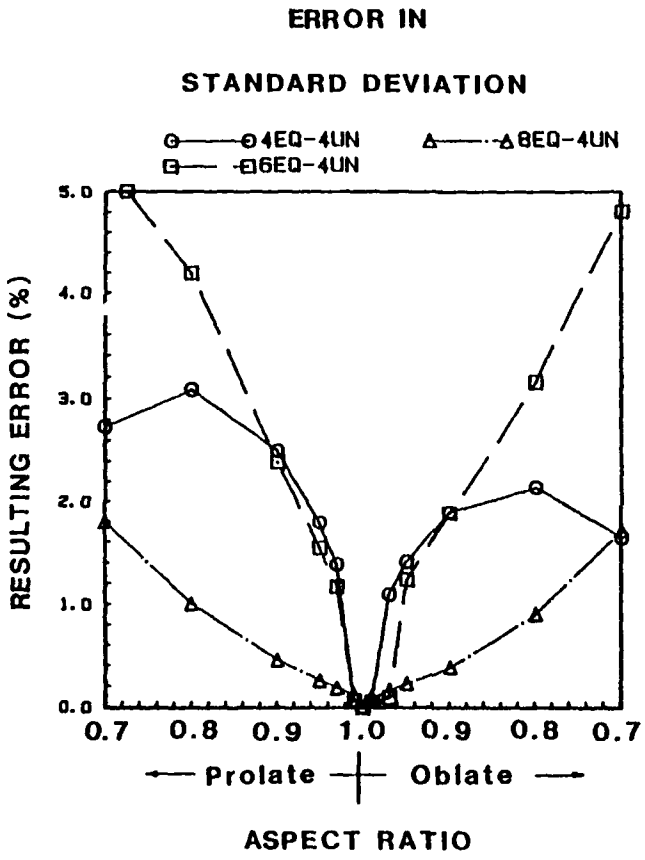


Figure 4.6: Spatial distribution used in the aspect ratio error analysis study.



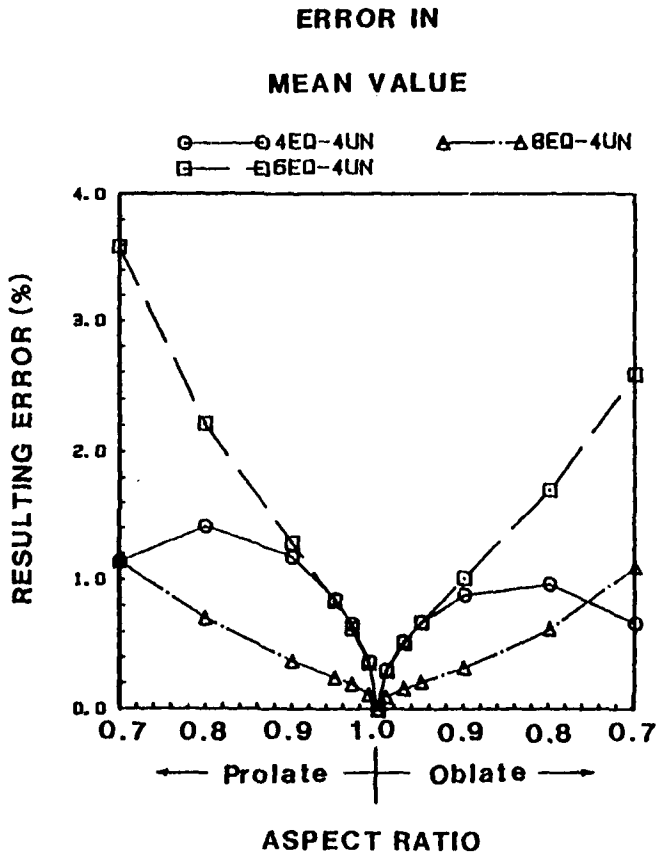


Figure 4.7 Plot of resulting error in the mean and standard deviations of the spatial distributions for induced error on the aspect ratio.

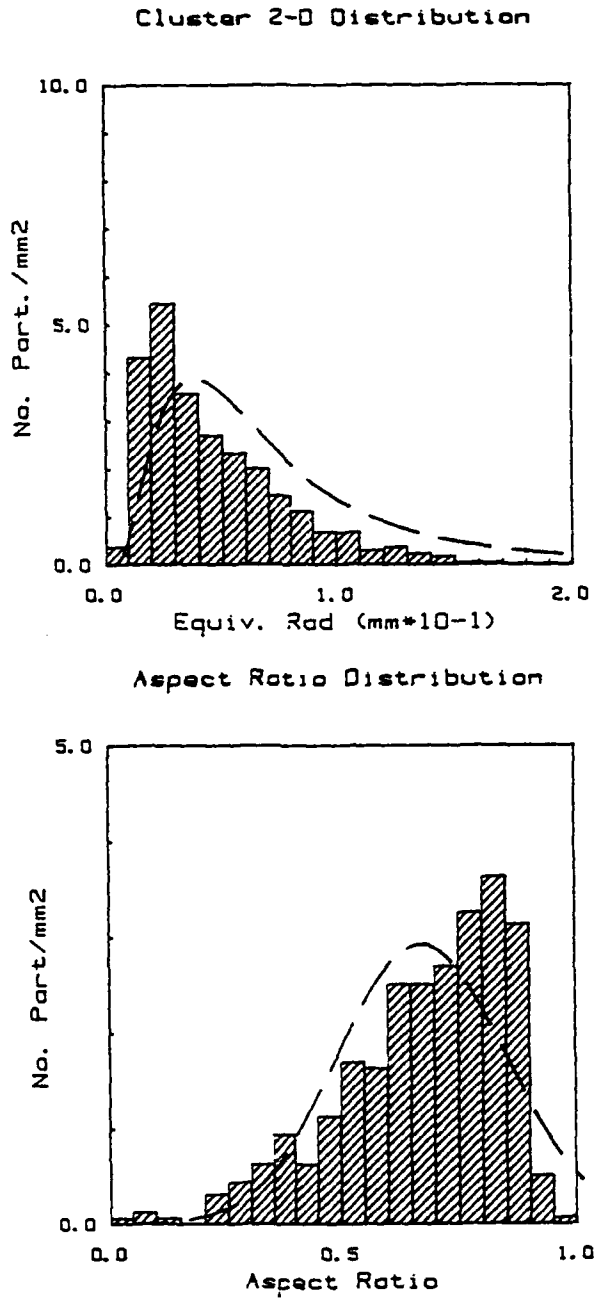


Figure 4.8 Data extracted from the image analyzer; a) Distribution of equivalent radii b) Aspect ratio distribution

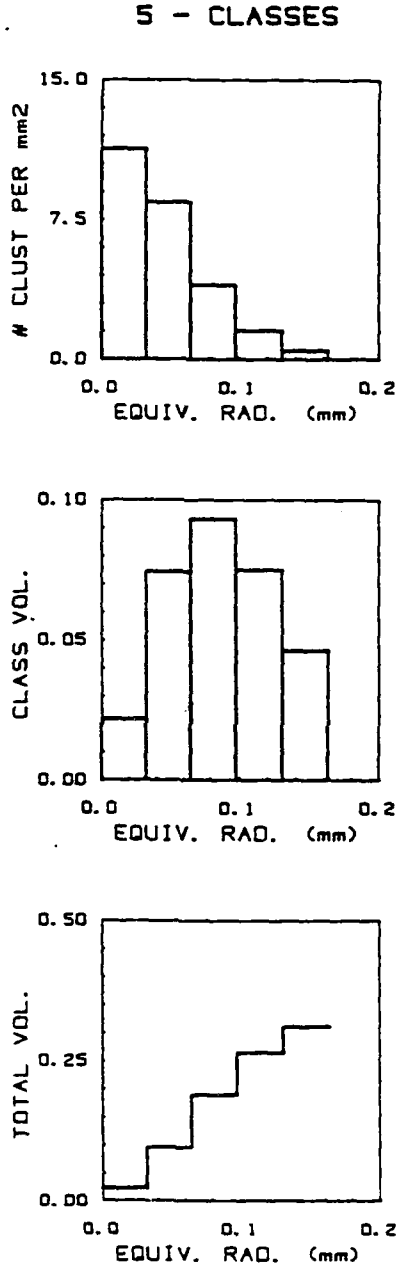


Figure 4.9 Plot of two and three dimensional distributions for 5 classes using Saltykov's solution.

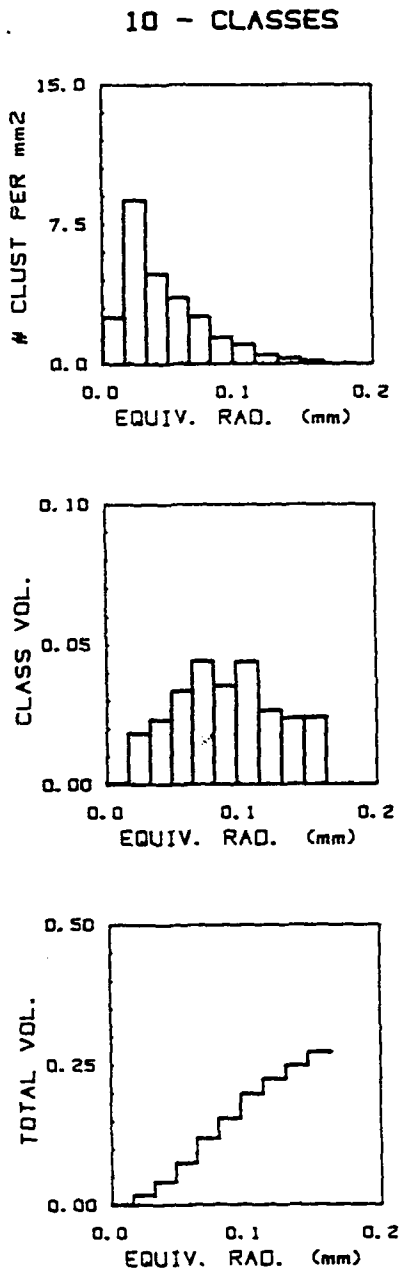


Figure 4.10 Plot of two and three dimensional distributions for 10 classes using Saltykov's solution.

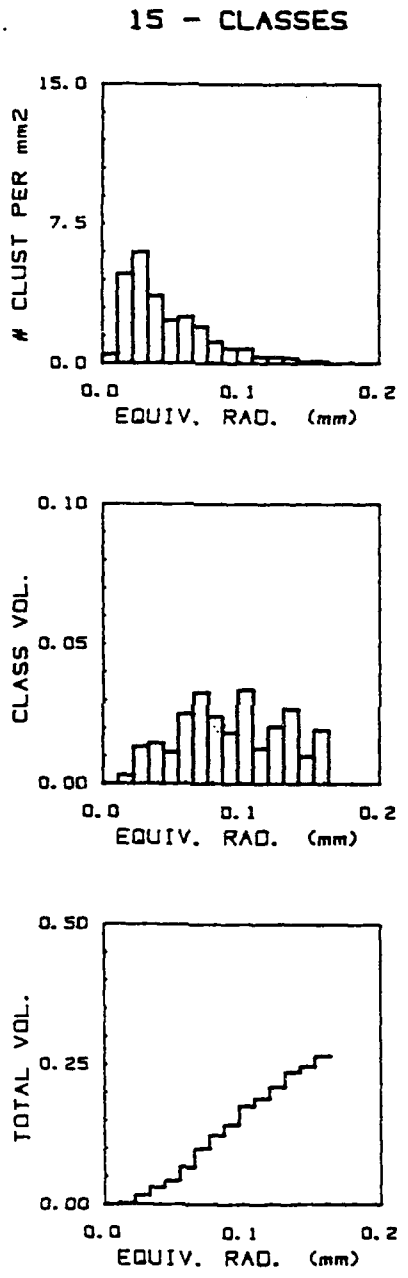


Figure 4.11 Plot of two and three dimensional distributions for 15 classes using Saltykov's solution.

REPRESENTATIVE VOLUME ANALYSIS			
Anal No.	Dimension (M*E-4)	Mean (%)	Std. Dev. (%)
1	6.20	50.47	14.54
2	12.40	48.71	9.29
3	18.60	51.23	5.54
4	24.80	52.24	3.58
5	31.00	51.20	3.02
6	37.20	51.79	2.49

Table 4.1 Representative volume study results.

Chapter 5

Kaolinite Damage Characterization

5.1 Introduction

In Chapter 3 of this thesis, evidence of stress induced morphological transformations (damage) of material at the cluster level is presented. Included is evidence of localized densification processes referred to as *Localized Strip Densifications* (*LSD*'s) and *Network Densifications*.

In general, characterization of damage involves both identification of characteristic features and some quantitative description of these features. Chapter 4 presents techniques which can be used to characterize the soil fabric in its natural state (i.e., cluster and water matrix) which essentially defines a reference level upon which the damage can be identified.

This chapter is devoted to illustrating techniques for characterization of the most common of the material transformations; the *LSD*'s. The *LSD* characterization procedure presented in this chapter is based on experimental techniques. In other words, experimental techniques are employed to

reconstruct the volumetric distribution of *LSD*'s.

5.2 LSD Characterization

The *LSD*'s (Localized Strip Densifications) are shown in a comparison photograph between two radial sections in Figure 3.5 and in Section b of Figure 3.6 (axial section). The sectioning procedure for both radial and axial sections is described in Section 2.8 and illustrated in Figure 2.7. The *LSD*'s can be thought of as densified surfaces emanating from the notch tip.

The first step in the characterization process involves making a series of radial sections through a fatigued and impregnated cylinder. These sections are usually equally spaced at approximately 5 mm intervals and start near the notch tip and continue through the entirety of the damage zone. Next, each of the sections is mounted on slides and polished (refer to Section 2.9).

Afterward, each slide is segmented into 3mm thick bands below a known reference plane as shown in Figure 5.1. Each slide is then individually placed under the stereoscope and the image analyzer (Section 2.10) is used to compute the total *LSD* area in each band. These areas are then used to construct a histogram for each slide as illustrated in Figure 5.1

It has been observed that the *LSD* area distributions are generally bell

shaped, thus a Gaussian curve is used to approximate the distributions in the axial directions for further analyses. These histograms and their approximating distributions are illustrated in Figure 5.2 for Cylinder 2–1 and is shown for for others (Cylinders 2–3, 3–1, and 5–6) in Appendix C. In Figure 5.3, the density distributions are sketched at their respective locations along the circumferential direction for the case of Cylinder 2–1.

The next step in the *LSD* characterization involves constructing a contour map of the damage zone. To accomplish this, the *LSD* density was approximated with the Gaussian distributions in the axial direction and a spline interpolation scheme was used in the circumferential direction. A fortran routine was developed to perform this task and generate a contour map of *LSD* area densities. The program is named CONTOUR and a listing is provided in Appendix A. Contour maps of *LSD* densities are illustrated for Cylinders 2–1, 2–3, and 3–1 in Figure 5.4. All of the contour maps in Figure 5.4 were constructed with contour lines starting at 0.1 mm levels and incrementing by 0.1 mm. levels.

Notice that the volume represented by the contour maps represent the total volume of *LSD* transformed during each of the respective tests. Thus, the information contained in these contour maps allows one to compute the total volume of material transformed from clusters to *LSD*'s during the cyclic fatigue test.

Also notice that the amount of transformed material increases as one moves away from the notch tip. This is evidenced in Figures 5.2 through 5.4. In Figure 5.2, Slide 2–1–A is nearest the notch tip and Slide 2–1–O is furthest. In Figure 5.4, the contour maps show that the highest *LSD* densities occur between 4.and 6 centimeters in front of the notch tip.

5.3 *LSD* Evolution

Observations of the radial sections did not identify a *main crack* within the densified regions. Moreover, external observations during the cyclic test revealed a propagating *zone of damage* but no one surface could be identified as a main crack. This is evidenced in Figure 1.6, and in the damage evolution photographs in Figures 3.1 and 3.2.

Photographs of the damage evolution similar to that shown in Figure 3.1 were taken for specified cycles during the fatigue of Cylinders 3–1 and 5–6. Note that these photographs were taken during the actual test (i.e., the cylinders were photographed through the plexiglas shell, silicon oil, and rubber membrane). Since the nature of the damage on the outside and the inside of the specimens have been experimentally correlated (i.e., in Figure 3.7 evidence showing the correlation between the external slip lines and the internal LSD's is illustrated), templates reflecting the evolution of damage for specified cycles was constructed from photographs. The basis for constructing the templates

involved using a photograph of the last cycle as a *final state* and proportioning the evolutionary development of earlier cycles to it. It is assumed that a direct correlation exists between the amount of damage produced on the outside of the specimen to that of its interior. Each photograph corresponding to a specified cycle was divided into regions. Next, percentages reflecting the amount each region has evolved during that cycle were assigned to each region thereby creating a grid of numbers reflecting the overall evolution of the damage zone called a template.

A computer program was written to multiply the template for a particular cycle onto the the contour map at the final stage. The program written to perform this task is called PZONE and a listing is included in Appendix A. Results from this program include a 3–D simulation of the LSD zone for various cycles. Processed results for Cylinders 3–1 and 5–6 are shown in Figures 5.5 and Appendix D, respectively. The processed results for Cylinder 3–1 include a 3–*D* plot of of the damage density for specified cycles of the test (Figure 5.5). The results for Cylinder 5–6 include both 3–*D* plots and corresponding contour plots for specified cycles (Appendix D).

5.4 Integral Description of *LSD* Zone Evolution

Due to the nature of the damage zone evolution reconstruction technique, it is not expected that minute details of a particular damage zone

description at some specified cycle (see Figure 5.5 and Appendix D) are necessarily accurate. Moreover, a constitutive law which describes this motion should not be sensitive to these same perturbations (recall the representative volume considerations in Section 4.2). Thus, the objective in this section is to present integral characteristics of the *LSD* zone evolution.

It is natural to decompose the *LSD* zone evolution into a combination of elementary movements. This is in agreement with the theoretical considerations outlined in Section 1.5 and described in detail in Chapter 6. Examples of such movements for the case under consideration may be:

$\dot{\underline{x}}_c$	–	Rate of Translation of *LSD* zone with respect to the centroid.
$\dot{\underline{\omega}}$	–	Rate of rotation.
$\dot{\underset{\sim}{d}}$	–	Rate of deformation.

Symmetry of the specimen geometry and the loading conditions eliminates $\dot{\underline{\omega}}$ from these studies.

In addition to the 3–D simulation of the *LSD* evolution, PZONE computes the centroid distance from the notch tip and the total volume of *LSD* for each specified cycle. Results from these analyses are presented in Figures 5.7 through 5.9 for Cylinders 3–1 and 5–6.

5.4.1 Resistive Moment Evaluation

Other integral descriptions of the damage zone relate to the amount of damage accumulated during each of the above described elementary movements. In the thermodynamic model proposed in Chapter 6, reference is made to *resistive moments* when describing thermodynamic forces. These resistive moments are related to the amount of damage accumulated during damage zone movements.

The resistive moment due to translation of the damage zone can be expressed as γR_1 where R_1 is functionally related to the the damage zone size and shape and is computed from:

[5.1] $$R_1 = \frac{1}{t} \int_{V_{sdz}} \left(\frac{\partial \rho}{\partial x_1}\right) dV$$

where V_{sdz} is the volume containing the Localized Strip Densification zone and ρ is the damage density and t is the cylinder thickness. This expression will be derived in Chapter 6 and is presented here only to describe the computational details.

Before equation 5.1 can be evaluated, a formal definition of the damage density density ρ must be presented. For the case of overconsolidated clays, it is found that microdefects (damage) are 3 dimensional (see Chapter 3).

Consequently, ρ can be defined as:

[5.2] $$\rho \equiv \lim_{\Delta V \to 0} \left[\frac{\Delta V_t}{\Delta V}\right]$$

ΔV_t – amount of material transformed within an incremental volume.

ΔV – incremental volume of material.

The incremental volume V is defined by the area of the damage zone where the Strip Densifications are detected, multiplied by the wall thickness.

Experimentally, R_1 is evaluated between any two cycles by the approximation given in Equation 5.3.

[5.3] $$R_1 \approx \frac{1}{t}\frac{1}{\Delta x_c}\int_{\Delta V_{sdz}} \rho \, dV$$

where Δx_c is the change in centroidal distances and ΔV_{sdz} is the change in volume of the damage zone associated with translating the existing damage zone to the new centroid location. This calculation is shown schematically in Figure 5.10. Figure 5.11 presents the results for Cylinder 5–6. It can be seen in Figure 5.10 that Equation 5.3 expresses the amount of new damage accumulated from pure translation of the damage zone per unit thickness of material.

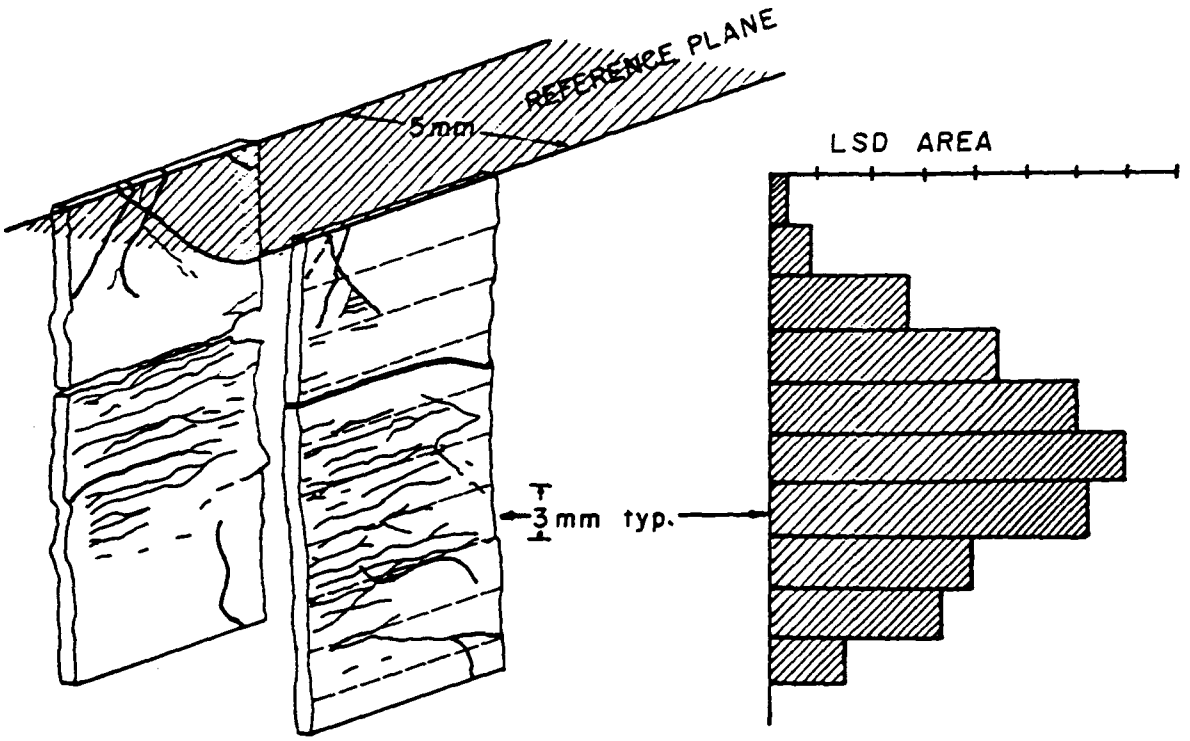


Figure 5.1: Schematic illustrating the *LSD* measurement procedure.

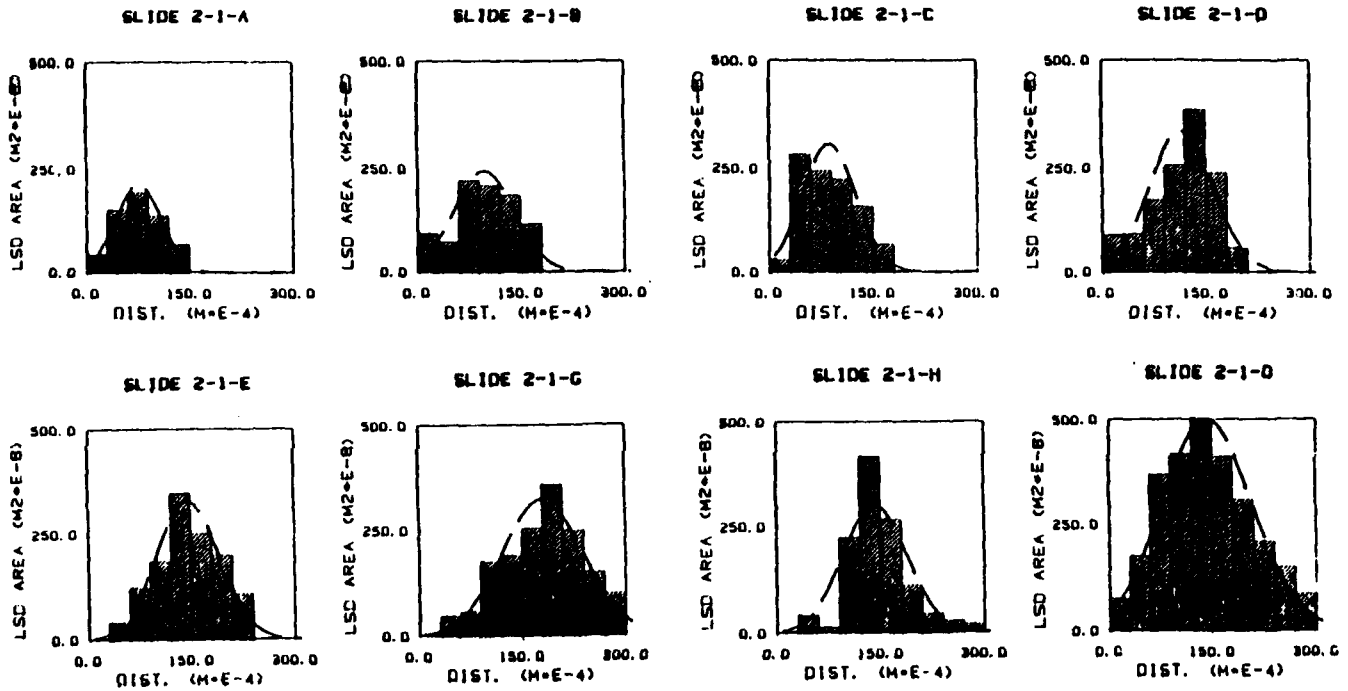


Figure 5.2: *LSD* histogram results for Cylinder 2–1. Note that Slide 2–1–A is nearest the notch and Slide 2–1–O is furthest away.

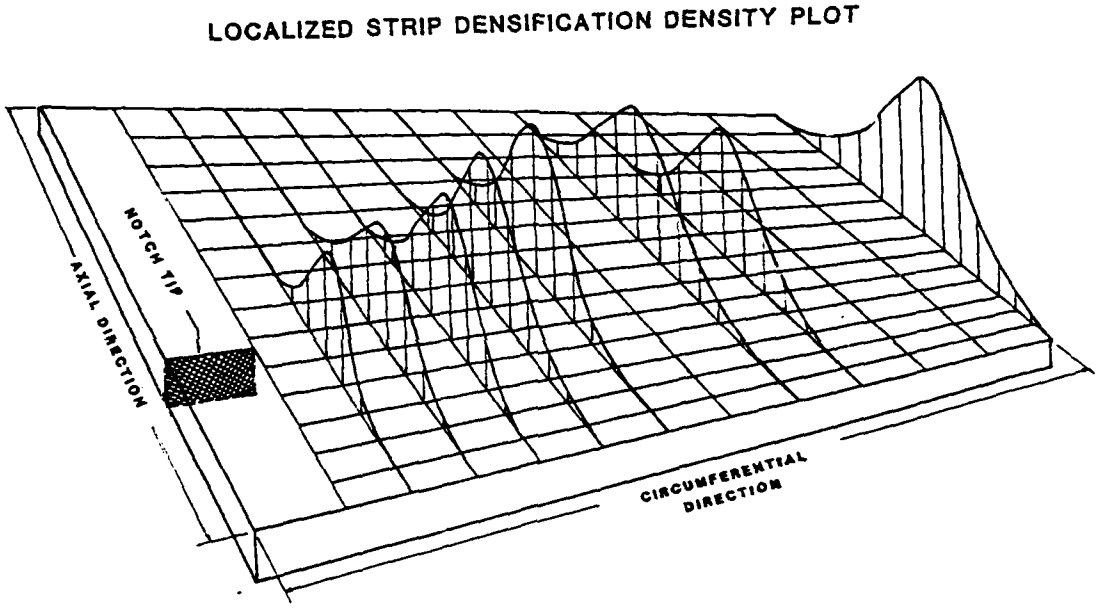


Figure 5.3: Three dimensional plot of *LSD* densities as related to their measurement locations along the axial and circumferential directions.

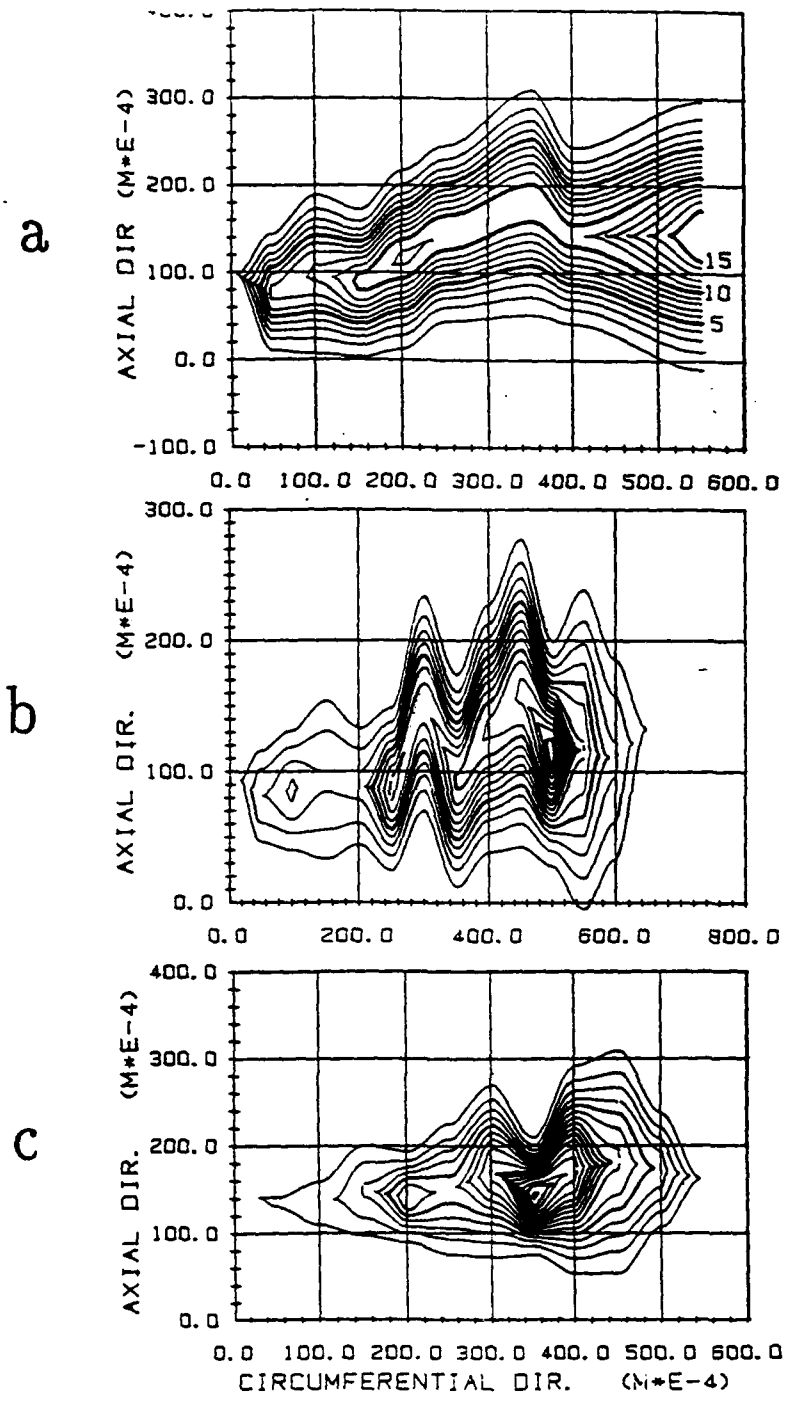


Figure 5.4: *LSD* area density contour plot for; a) Cylinder 2–1, notch at (0,100). b) Cylinder 2–3, notch at (0,100). c) Cylinder 3–1, notch at (0,150).

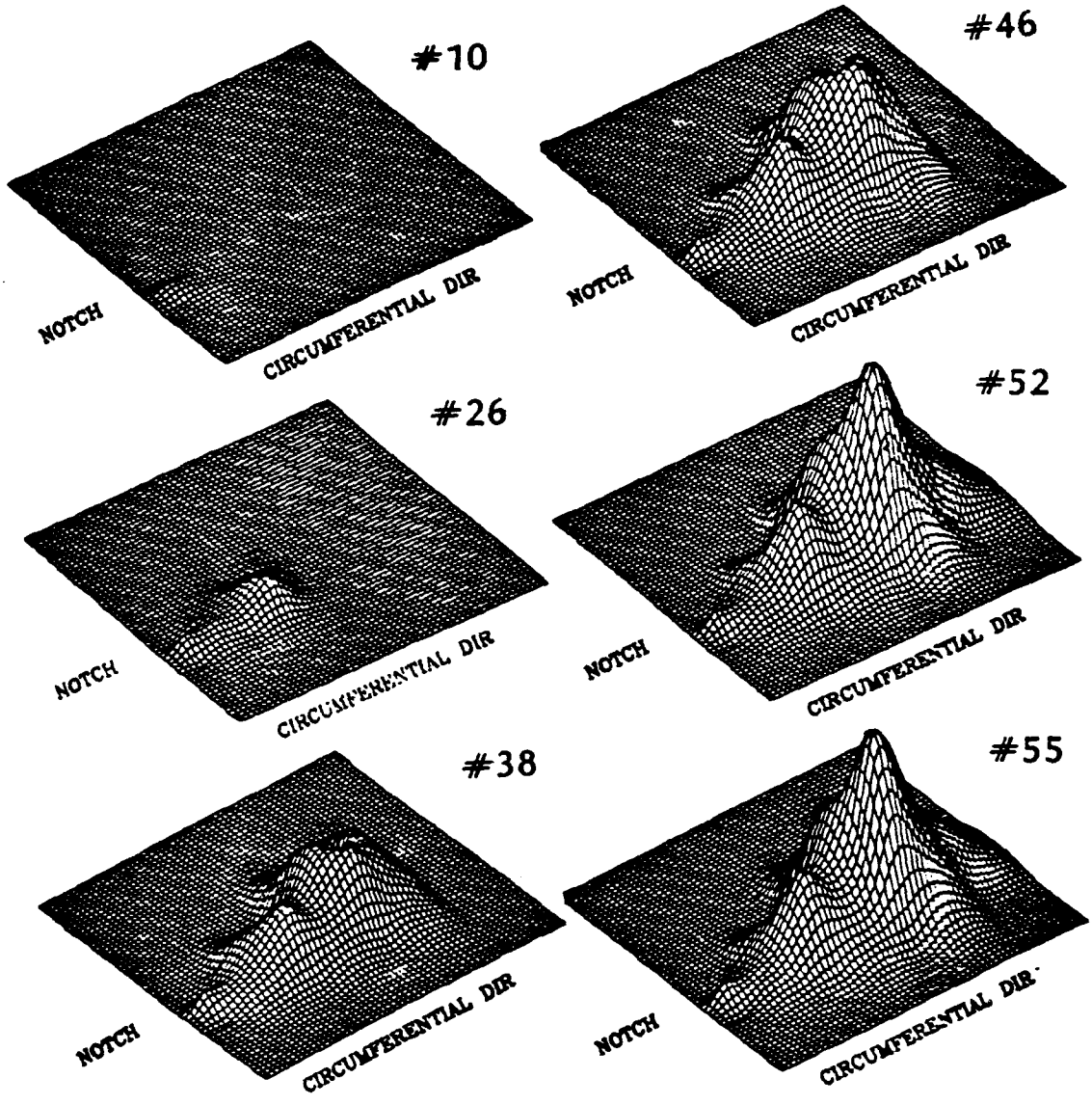


Figure 5.5: Three dimensional sequence of *LSD* zone evolution as a function of cycles.

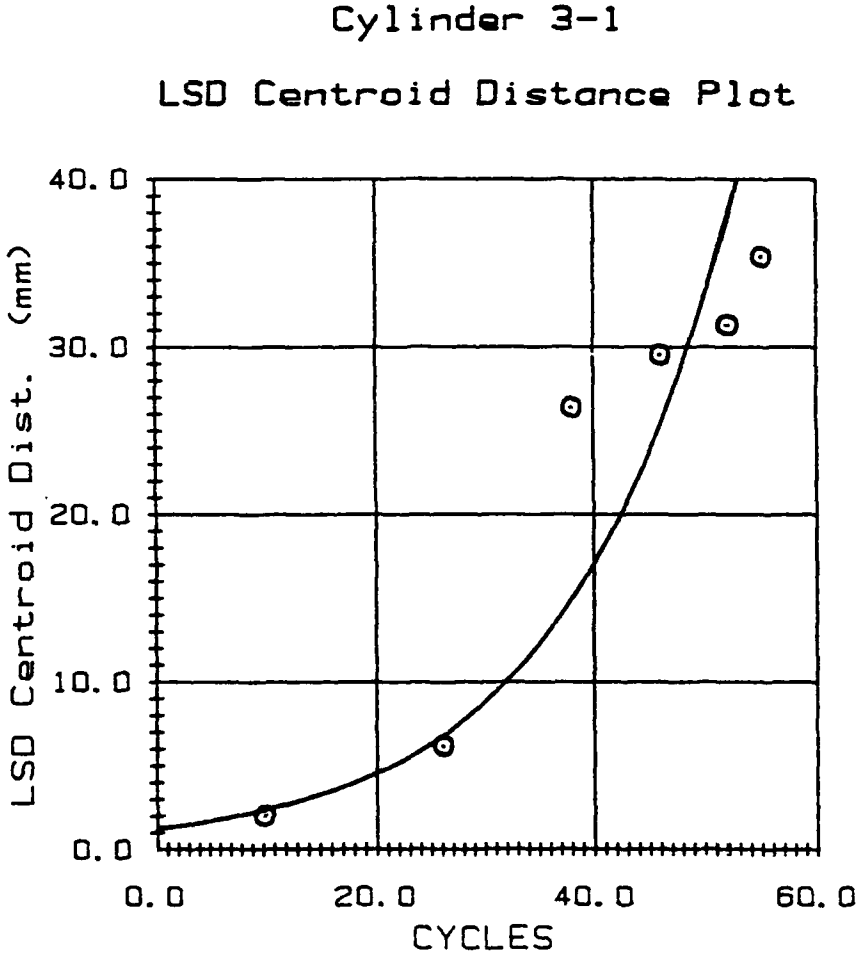


Figure 5.6: Plot of *LSD* zone centroid distance vs. cycles for Cylinder 3–1.

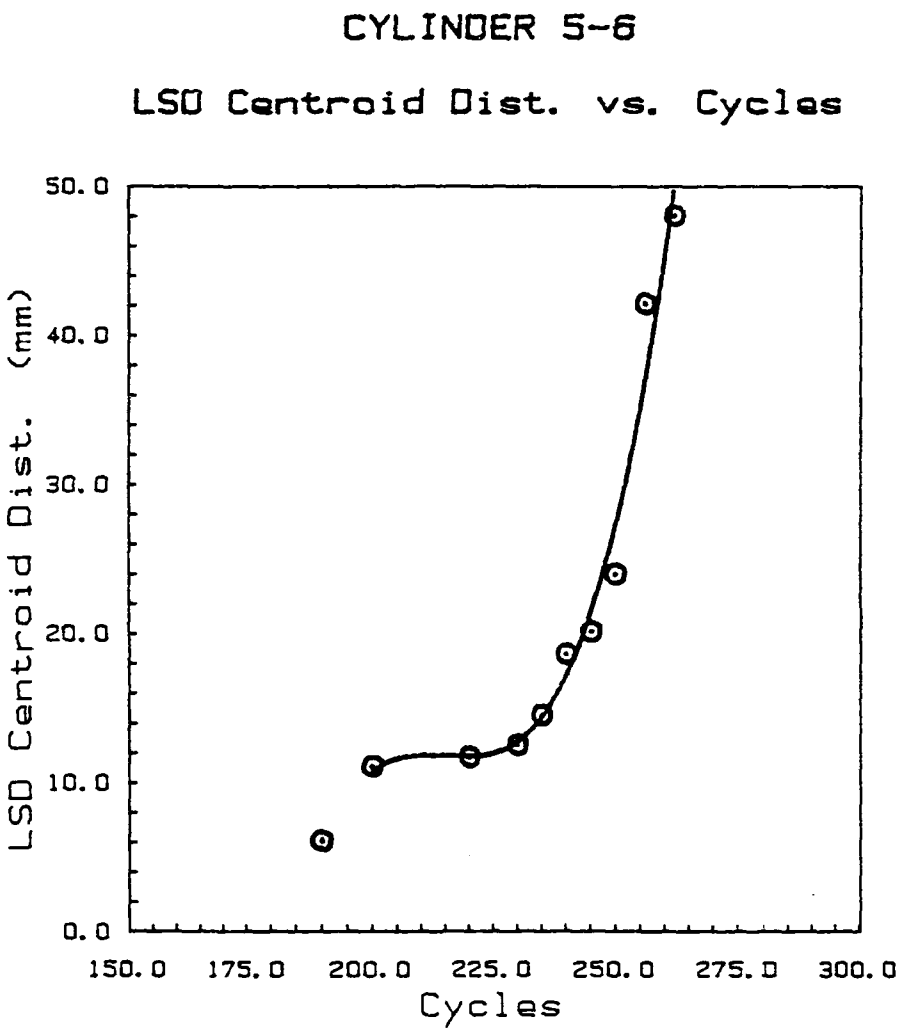


Figure 5.7: Plot of *LSD* zone centroid distance vs. cycles for Cylinder 5–6.

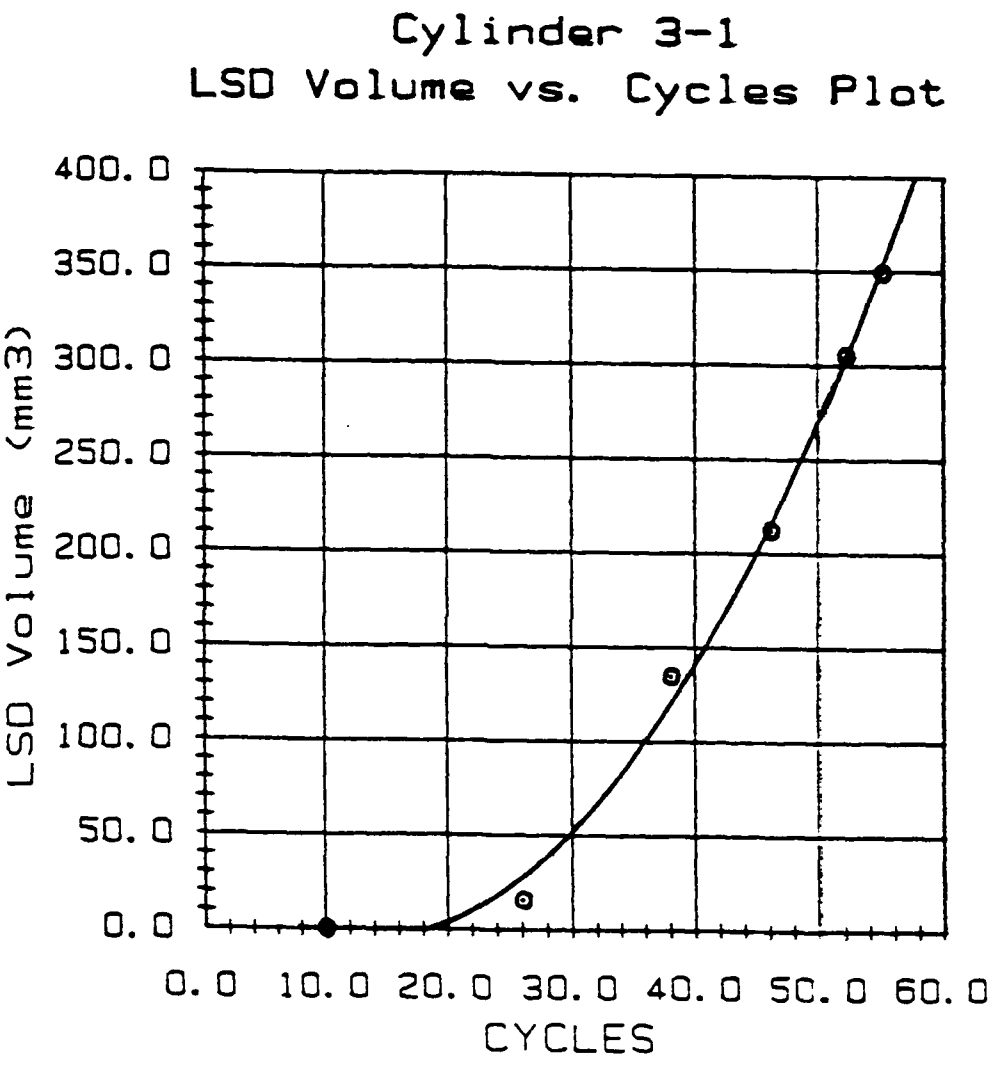


Figure 5.8: *LSD* zone volume vs. cycles for Cylinder 3–1.

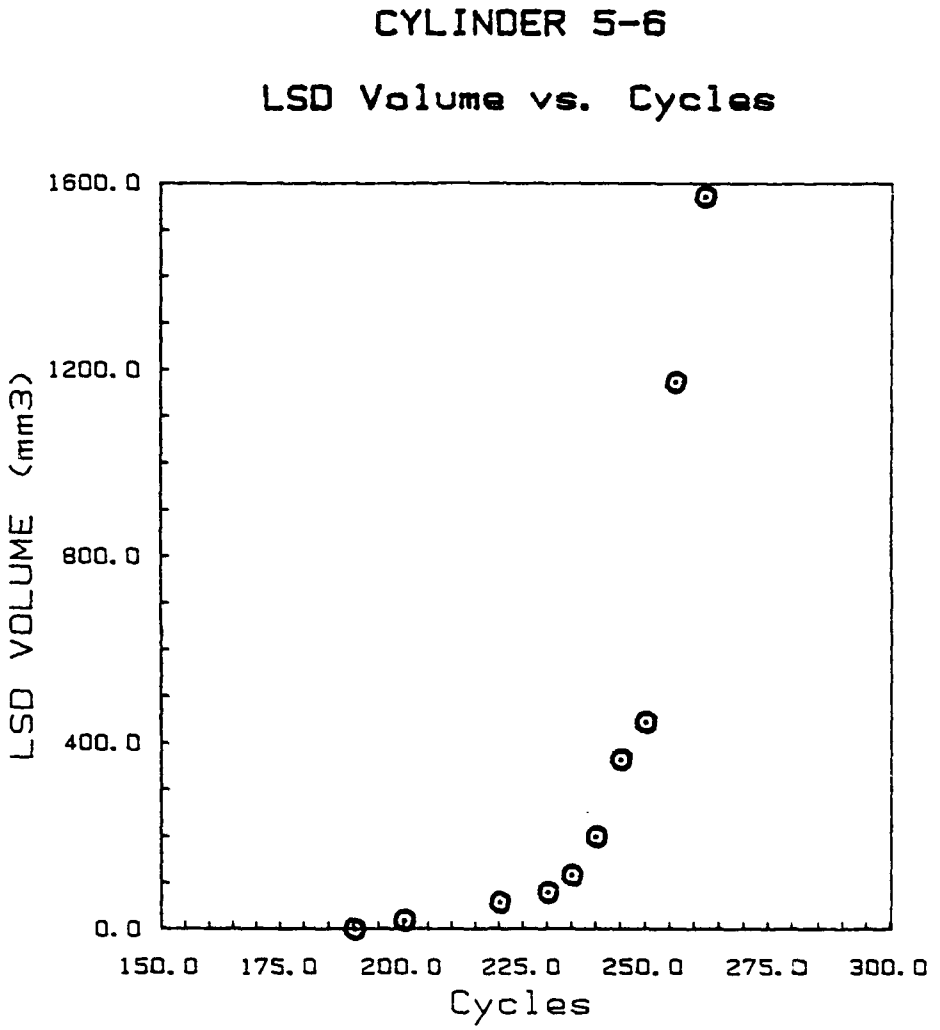
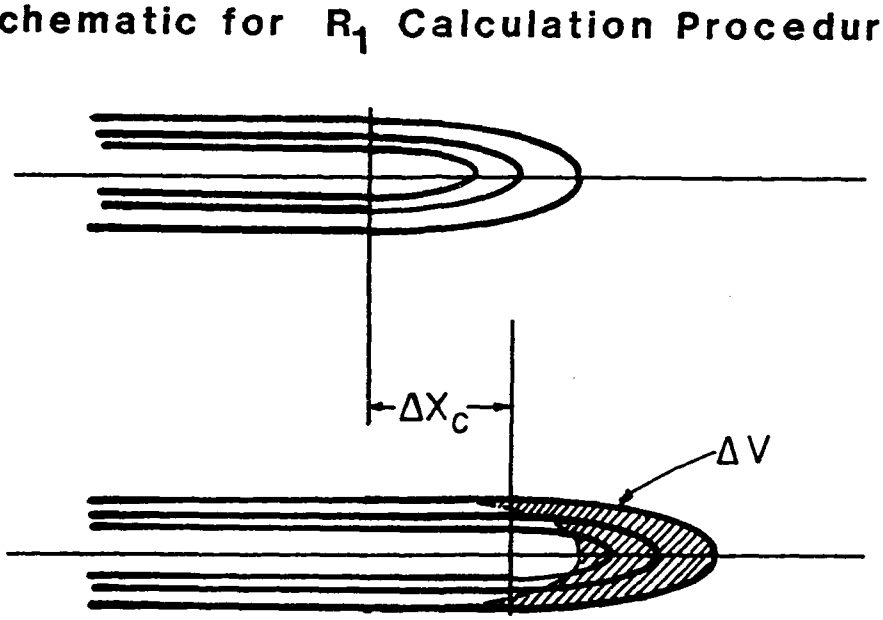


Figure 5.9: *LSD* zone volume vs. cycles for Cylinder 5–6.



$$R_1 \approx \frac{1}{t}\frac{1}{\Delta x_c}\int_{\Delta V} \rho \, dV$$

Figure 5.10: Schematic of calculation procedure for R_1.

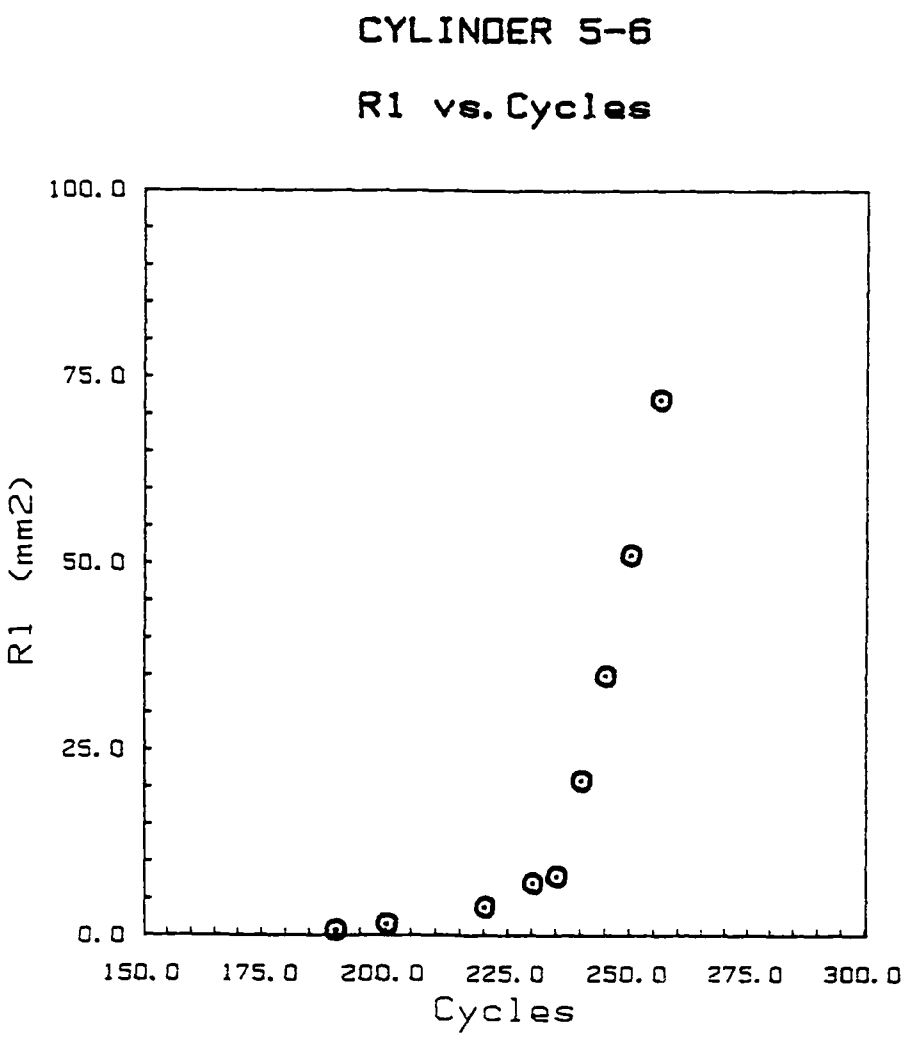


Figure 5.11: Plot of R_1 vs. cycles for Cylinder 5–6.

Chapter 6

Theoretical Considerations

6.1 Introduction

In Section 1.5 we introduce the governing equation (Equation 1.1) which describes the *crack–damage* evolution for a cooperative fracture process. This equation results from treating the crack and surrounding damage as a single thermodynamic entity and applying the laws of thermodynamics of irreversible processes.

Usually fracture is an irreversible process which is often coupled with other physicochemical processes like phase and chemical transformations, heat and mass transfer, etc. Thermodynamics of irreversible processes offers a general framework to study these phenomena, in particular, by introducing the thermodynamic forces. It is demonstrated herein that thermodynamic forces are distinct from conventional energetic forces like the J–integral. In this chapter thermodynamics and energetic forces for thermoelasticity are derived following the thermodynamic approach (3,4).

6.2 Thermodynamics of Fracture in an Elastic Medium

It is demonstrated in Chapters 3 and 5 that the fracture process is strongly cooperative. In this chapter, damage (*LSD*) is considered as a system of material inhomogeneities, and damage nucleation and growth as a material transformation. Damage can be characterized by its density (Chapter 5) and average orientation. It is assumed that the orientation of inhomogeneities does not vary in the fracture process. Therefore, the scalar damage density is incorporated as the only thermodynamic state parameter characterizing damage.

The stress tensor σ_{ij} and the absolute temperature T conventionally constitute a set of state parameters for thermoelasticity. This set is extended by the damage density ρ in an elastic medium:

[6.1] $$\{\sigma_{ij},\ T,\ \rho\}$$

In what follows, we derive the thermodynamic forces associated with damage. The local energy balance is taken as

[6.2] $$\dot{u} = \sigma_{ij}\dot{\epsilon}_{ij} - \frac{\partial \dot{q}_k}{\partial x_k}$$

Here $\dot{u}$ stands for the rate of internal energy density, $\dot{\epsilon}_{ij}$ is the strain rate

tensor, the product $\sigma_{ij}\dot{\epsilon}_{ij}$ represents the work rate density, $\dot{q}_k$ stands for the heat flux, and the usual summation convention over the repeated indicies is employed.

It is convenient to express the internal energy density u in terms of Gibb's potential density and the entropy density s:

[6.3] $$u = g + Ts + \sigma_{ij}\epsilon_{ij}$$

Here, ϵ_{ij} is the total strain component. The entropy production $\dot{s}_i$ introduced in thermodynamics of irreversible processes is defined as a portion of the total entropy production rate $\dot{s}$.

$$\dot{s}_i = \dot{s} - \dot{s}_e$$

[6.4] with:

$$\dot{s}_e = -\frac{\partial}{\partial x_k}\left[\frac{\dot{q}_k}{T}\right]$$

where $\dot{s}_e$ is the entropy density rate due to heat exchange (for a closed system where no mass is transposed).

Employing the energy balance in Equation 6.2, decomposition of the internal energy density in Equation 6.3 and the definition of the entropy production in Equation 6.4 we arrive at:

[6.5] $$\dot{s}_i = \frac{1}{T}\left[-\dot{g} + s\dot{T} + \epsilon_{ij}\dot{\sigma}_{ij} + \frac{\dot{q}_k}{T}\left[\frac{\partial T}{\partial x_k}\right]\right]$$

In conventional thermoelasticity, the Gibb's potential density g is often designated as the thermoelastic potential energy density π, and is expressed as a Taylor's decomposition with respect to the state parameters σ_{ij} and T. The coefficients of decomposition represent material properties such as elastic compliance, thermal expansion coefficient, etc., and the first term of the decomposition represents the reference level of g.

For the damaged thermoelastic medium, the rate of Gibb's potential density $\dot{g}$ is affected by the damage growth $\dot{\rho}$ in two ways, through changes in (i) the reference level of g (Gibb's Potential), and (ii) the material property coefficients included in π. Hence,

[6.6] $$\dot{g}(\sigma_{ij}, T, \rho) = \gamma(\sigma_{ij}, T)\dot{\rho} + \dot{\pi}(\sigma_{ij}, T, \rho)$$

Here, γ is the difference between Gibb's potential densities in the damaged and undamaged states per unit damage density; and π is taken in the same form as in conventional thermoelasticity, but with the material property coefficients being functions of the damage parameter ρ.

The assumption of local equilibrium yields thermoelastic constitutive equations:

[6.7] $$\left.\frac{\partial \pi}{\partial T}\right|_{\rho, \varepsilon_{ij}} = -s$$

[6.8] $$\left.\frac{\partial \pi}{\partial \sigma_{ij}}\right|_{T, \rho} = -\epsilon_{ij}$$

Accounting for Equations 6.6 and 6.8 the local entropy production in Equation 6.5 takes the form:

[6.9] $$\dot{s}_i = -\frac{1}{T}\left[\gamma + \frac{\partial \pi}{\partial \rho}\right]\dot{\rho} - \frac{1}{T^2}\left[\frac{\partial T}{\partial x_k}\right]\overset{q}{J_k}$$

In the outline of thermodynamics of irreversible processes, the entropy production is conventionally presented as a bilinear form of generalized fluxes and forces, such as $\overset{q}{J_k}$ and $-\partial_k T/T^2$ for heat transfer. Similarly, if the rate of damage density $\dot{\rho}$ is taken as a flux, the reciprocal force is $-(1/T)[\gamma + (\partial \pi/\partial \rho)]$.

6.3 Damage Zone Propagation

It has been observed that the damage zone (*LSD* zone) in overconsolidated clays propagates and ultimately leads specimen failure (see Section 5.3). In the approach we introduce herein, the damage zone movements will be decomposed into a combination of elementary movements; translation, deformation). In this thesis, we apply the thermodynamic model to the translation of the *LSD* zone.

The global entropy production is the integral of Equation 6.9 over the entire volume of the solid:

$$[6.10] \quad \dot{S}_i = \int_V \dot{s}_i \, dV = -\int_V \frac{\gamma}{T} \dot{\rho} \, dV - \int_V \frac{\partial \pi}{\partial \rho} \frac{\dot{\rho}}{T} \, dV - \int_V \frac{1}{T^2} \frac{\partial T}{\partial x_k} \dot{q}_k \, dV$$

The first two integrals on the right hand side of Equation 6.10 are reduced to the those over the damage zone V_{sdz} since $\dot{\rho}$ in nonzero only within this zone and they represent the global entropy production due to **Localized Strip Densification**. The first integral is associated with the energy consumed by material transformation from an undamaged state into a damaged state (i.e. from clusters to *LSD*'s). It reflects the materials resistance to the strip densification zone propagation. The second integral evidently represents the the entropy production associated with the potential energy release rate, and is defined as the impellent of the *LSD* zone. The last integral in Equation 6.10 reflects the entropy rate due to heat transfer. For the case considered here, an isothermal condition will be assumed ($\nabla T = 0$). Thus the last integral in Equation 6.10 reduces to zero.

It is assumed that the damage zone moves maintaining a self–similar distribution of damage such that the rate of damage density $\dot{\rho}$ at a given point x_i in an Euler system of coordinates, can be expressed as

$$[6.11] \quad \dot{\rho}(x_i) = -V_k(\xi_i) \frac{\partial \rho(\xi_i)}{\partial x_k}$$

where ξ_i is a position vector in the movable system of Cartesian coordinates with the origin at the crack tip and the k=1 axis is chosen along the tangent to the notch and $V_k(\xi_i)$ is the k^{th} component of the velocity vector. Reducing the general motion of the Strip Densification Zone to simple translation of the centroid ($\dot{x}^c$):

[6.12] $$V_k(\xi_i) = x^c \delta_{1k}$$

provided $i,k = 1,2$ and δ_{ik} is Kronecker's delta symbol. The damage density $\dot{\rho}$ in Equation 6.11 with 6.12 becomes:

[6.13] $$\dot{\rho}(x_i) = -\dot{x}_1^c \frac{\partial \rho(\xi_i)}{\partial x_1}$$

Upon substitution of Equation 6.13 into Equation 6.10 we can write:

[6.14] $$\dot{S}_i = \frac{1}{T}(\underline{\dot{x}}^c \cdot \underline{X}_1)$$

Above, $\underline{\dot{x}}^c$ is considered to be a generalized flux and stands for the rate of translation of the damage zone as measured from its centroid (see Section 5.3) and consequently, $\underline{X}_1$ is the reciprocal thermodynamic force. The thermodynamic force $\underline{X}_1$ can be written as a sum of an active and resistive part. That is:

[6.15] $$\underline{X}_1 = \underline{A}_1 - \underline{R}_1$$

where:

[6.16] $$\underline{A}_1 = -\int_{V_{sdz}} \frac{1}{T} \frac{\partial \pi}{\partial \rho} \frac{\partial \rho}{\partial x_1} dV$$

and

[6.17] $$\underline{R}_1 = \int_{V_{sdz}} \frac{\gamma}{T} \frac{\partial \rho}{\partial x_1} dV$$

Equations 6.16 and 6.17 represent the active and resistive components of the thermodynamic forces, respectively. For the case considered here, R_1 (excluding the γ/T term) is evaluated in Subsection 5.3.1. In the next section we will concentrate our efforts to re–expressing Equation 6.17 into a recognizable form.

6.4 Impellent Forces

For a homogeneous medium the density of elastic potential energy π is a function of the state parameters and does not depend on coordinates explicitly:

[6.18] $$\pi = \pi\left[\sigma_{ij}(\xi_i), \rho(\xi_i)\right]$$

Consequently,

[6.19] $$\frac{\partial \pi}{\partial \rho}\frac{\partial \rho}{\partial x_k} = \frac{\partial \pi}{\partial x_k} - \frac{\partial \pi}{\partial \sigma_{ij}}\frac{\partial \sigma_{ij}}{\partial x_k}$$

where for a two–dimensional case , i,j, and $k = 1,2$.

Using constitutive Equations 6.7, 6.8, and 6.19 can be written as

[6.20] $$\frac{\partial \pi}{\partial \rho}\frac{\partial \rho}{\partial x_k} = \frac{\partial \pi}{\partial x_k} + \epsilon_{ij}\frac{\partial \sigma_{ij}}{\partial x_k}$$

The elastic potential energy density is defined as

[6.21] $$\pi = f - \sigma_{ij}\epsilon_{ij}$$

where f represents the strain energy density which is conventionally expressed in linear elasticity as follows:

[6.22] $$f = \frac{1}{2}\sigma_{ij}\epsilon_{ij}$$

Upon substitution of Equation of Equation 6.20, 6.21, and 6.22 into 6.16 and accounting for local equilibrium equations

[6.23] $$\frac{\partial \sigma_{ij}}{\partial x_j} = 0$$

A_1 can be written as:

$$[6.24] \qquad A_1 = \int_{V_{sdz}} \frac{1}{T}\left[\frac{\partial f}{\partial x_1} - \frac{\partial}{\partial x_j}(\sigma_{ij}\frac{\partial u_i}{\partial x_1})\right] dV$$

Finally, applying Gauss's Theorem we arrive at:

$$[6.25] \quad A_1 = \int_{\partial V_{sdz}} \frac{1}{T}(f\cdot n_1 - \sigma_{ij}n_j u_{i,1})\, d\Gamma - \int_{V_{s\,dz}} \frac{1}{T^2}\left[f\cdot\frac{\partial T}{\partial x_1} - \sigma_{ij}u_{i,1}\left[\frac{\partial T}{\partial x_j}\right]\right] dV$$

In Equation 6.25, n_j is the j^{th} component of the unit outward normal to an integration path ∂V_{sdz}. In the case of an isothermal condition ($\nabla T = 0$) the second integral in Equation 6.25 vanishes. Therefore the expression for A_1 reduces to :

$$[6.26] \qquad A_1 = \frac{1}{T}\int_{\partial V_{sdz}} (f\cdot n_1 - \sigma_{ij}n_j\frac{\partial u_i}{\partial x_1})\, d\Gamma$$

Notice that Equation 6.26 resembles the well known *J*–integral introduced by James Rice (2,10).

6.5 Energetic Force

In the previous section the thermodynamic *crack driving forces* in an elastic medium are presented. In this section, the conventional energetic forces

as derived by Eshelby (1,5,6,7,9) and Rice (2,10) will be introduced for comparison. Rice presents the *J*–integral as:

[6.27] $$J_1 = \int_{\Gamma} (f \cdot n_1 - \sigma_{ij} n_j \frac{\partial u_i}{\partial x_1})\, d\Gamma$$

Note that Equations 6.26 and 6.27 only differ by the absolute temperature term (T) for the isothermal condition. Although these two integrals are similar, identifying which is the *true* force driving a crack is at this time speculative (3,4).

The energy release rate can also be expressed in terms of Eshelby's Energy Momentum Tensor (1,5,6,7,9) as:

[6.28] $$J_1 = \int_{\partial V_{sdz}} (P_{i1} n_i)\, d\Gamma$$

where:

[6.29] $$P_{ij} = f\delta_{ij} - \sigma_{ik} u_{k,j}$$

and

f – strain energy density

σ_{ik} – stress tensor components

u_k – displacement vector

δ_{ij} – Kroneker's delta

6.6 Discussion

In this chapter we presented an expression illustrating that the global entropy production involved in the fracture process of an elastic medium with damage under isothermal conditions (Equation 6.14) can be expressed as a bilinear form of generalized flux and a thermodynamic force. This equation considers only the translation of the damage zone. In order to convert the expression in 6.14 into a constitutive model, we must first evaluate the thermodynamic force (Equations 6.15, 6.16 and 6.17) which contains an unknown value for γ (the specific energy for densification), and determine evaluate the relationship between the force and flux (e.g. the phemonological relationship).

Methods for evaluating the thermodynamic flux ($\dot{\underline{x}}_c$– rate of translation of the centroid of the damage zone) and the resistive part of the thermodynamic force (Equation 6.17) have been developed and are illustrated in Chapter 5.

Chapter 7 introduces methods for evaluating the active part of the thermodynamic force through a semi–empirical stress and energy analysis. Chapter 8 will tie the results together and present a constitutive model for overconsolidated clays.

References

1) Bilby, B. A. (1985). "Introductory Lecture in: Fundamentals of Deformation and Fracture, Eshelby Memorial Symposium", eds B. A. Bilby, K. J. Miller, J. R. Willis, Int. Union of Theorectical Applied Mechanics, Cambridge University Press, pp 11 – 32.

2) Budiansky, B., Rice, J. (1973). "Conservation Laws and Energy Release Rates", Journal of Applied Mechanics, Vol. 40, pp 201 – 203.

3) Chudnovsky, A. (1984). "Crack Layer Theory", NASA Report, N174634

4) Chudnovsky A., Gommerstadt, B. (1986). "Thermodynamic and Energetic Forces in Thermoelasticity", Int. Journal Solids Structures, Vol. 1, No.1, pp 1 – 7

5) Eshelby, J. (1951). "The Force on an Elastic Singularity", Phil. Trans., Royal Soc. of London, Vol. A244, pp 87 – 112.

6) Eshelby, J. (1970). "Energy Relations and the Energy–Momentum Tensor in Continuum Mechanics in Inelastic Behavior of Solids", eds. M. Kanninen, W. Adler, A. Rosenfield, and R. Jaffee, McGraw Hill, pp 77 – 114.

7) Eshelby, J. (1975). "The Elastic Energy–Momentum Tensor", Journal Elasticity, Vol. 5, No. 4–5, pp 321 – 335.

8) Fung, Y. C. (19). "Foundations of Solid Mechanics", Prentice–Hall, Inc., pp 341 – 407.

9) Hill, R. (1986). "Energy–Momentum Tensors in Elastostatics: Some Reflections on the the General Theory" Journal of Mech. Phys. Solids, Vol. 34, No. 3, pp. 305 – 317.

10) Rice, J. (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.

Chapter 7

Stress and Energy Analysis of the Crack – Damage Interaction

7.1 Introduction

In Chapter 6, a constitutive law based on thermodynamics of irreversible processes is outlined. In order to apply this formalism, knowledge of the dominant thermodynamic fluxes, and forces is required. For the case of overconsolidated clays, the fluxes and the resistive part of the thermodynamic forces for a translational model are evaluated and presented in Chapter 5.

This chapter describes an approach for evaluation of the active part of the thermodynamic forces. That is, the potential energy release rates associated with each of the fluxes (elementary movements).

During the cyclic fatigue tests, measurements of applied torque vs. rotation are recorded for specified cycles. From these measurements, values for the potential energy (obtained by using the unloading portion of the

torque–rotation curve) and irreversible work can be computed. These energy values can then be used to assess the total potential energy release rate between successive cycles. Recall, however, that potential energy is released during any of the four elementary movements (translation, rotation, expansion, distorsion) of the damage zone. The information presented in this chapter isolates the potential energy release rate for pure translation of the damage zone.

7.2 Solution via Superposition

A rigorous stress and energy analysis of this problem would require the reconstruction of the stress and displacement fields for the case of a hollow cylinder with a circumferential crack interacting with the *microcrack array* while being loaded under torsion. The nature of this study, however, is fundamental and thus a highly accurate analysis is not a goal.

In order the simplify the problem, we will restrict our analysis to the case of an infinite plate (i.e., plane stress or plane strain) made of a linear elastic material. For this case, the general problem can be decomposed by the law of superposition into three separate problems. These will be identified herein as:

Case 1 – A uniform plate free of all cracks and microdefects subjected to a uniform shear and compressive stress.

Case 2 – A plate with the main crack only (i.e. the notch) and a traction equal to the shear stress in Case 1 applied to the crack surfaces.

Case 3 – A plate with both the main crack and the array of *microcracks* with the measured displacements applied along the microcrack array

These cases are illustrated in Figure 7.1. By the laws of superposition, the combined stress, strain, and displacement fields, along with the stress intensity factor can be expressed as a sum of their respective parts:

$$\begin{aligned} \underset{\sim}{u}_{tot} &= \underset{\sim}{u}_1 + \underset{\sim}{u}_2 + \underset{\sim}{u}_3 \\ \underset{\sim}{\sigma}_{tot} &= \underset{\sim}{\sigma}_1 + \underset{\sim}{\sigma}_2 + \underset{\sim}{\sigma}_3 \\ K_I^{tot} &= K_I^2 + K_I^3 \\ K_{II}^{tot} &= K_{II}^2 + K_{II}^3 \end{aligned} \qquad [7.1]$$

Case 2 is the classical case and the solution is in every elementary book on fracture mechanics (14,18). If stresses σ_∞ and τ_∞ are the tensile and shear stresses applied at the remote locations on an infinite plate which contains a crack of length $2c$, the stress intensity factor for Case 2 is given by:

$$\begin{aligned} K_I &= \sigma_\infty \sqrt{\pi c} \\ K_{II} &= \tau_\infty \sqrt{\pi c} \end{aligned} \qquad [7.2]$$

In the next sections of this chapter we will concentrate our efforts on developing a method to solve the problem in Case 3.

7.3 The Crack–Microcrack Interaction Problem

In recent years, the problem of multiple crack interaction has been addressed by various authors. This problem can formally be expressed as a system of singular integral equations. The equations represent boundary conditions on the crack and micro–crack surfaces (the usual boundary conditions require traction free surfaces). Rigorous solutions to this problem have been formulated for only a select set of micro–crack configurations

Rice and Tompson (26) analyzed the interaction between a gliding dislocation and a crack using an energy method (i.e., using a Maxwell relation for the strain energy). Similarly, Shiue and Lee (32) studied the effect of a climb dislocation. They included solutions for both a single dislocation and a dislocation dipole. This problem was also addressed by Lo (21) and Ballarini (2,3) using complex potentials and by Erdogan (14) for a point force and moment.

The problem of collinear micro–cracks interacting with a semi–infinite crack was analyzed by Rubinstein (30). Similarly, Rose (27) studied the case of a semi–infinite crack interacting with a collinear mico–crack. The list continues with various techniques on other select problems in (1 through 32), just to name a few.

7.4 Semi–Empirical Approach via Second Green's Tensor

Due to the nature of the crack–damage interaction problem (i.e., solving a system of singular integral equations), solutions to many realistic problems becomes impractical, even with the advent of supercomputers. Alternatively, certain authors have proposed semi–empirical techniques to compute the SIF (Stress Intensity Factor) induced on main crack from an array of microdefects.

Chudnovsky et al. (5,11,12) propose a technique based on a self consistent method utilizing a double layer potential formulation. In this analysis, the microcracks are considered as a continuous distribution of dislocations and the problem of feedback is solved by approximating the traction on the microcracks as a polynomial distribution (using Willis's (34) polynomial conservation theorem). The interaction between the main crack as well as between the microcracks is considered. More recently, Kachanov (17,18) proposed as alternative approach by considering the average tractions on the individual microcracks.

In general, for a random configuration of a large number of microcracks the solution implies an extremely tedious and time consuming numerical procedure (8,9,10). In 1987, Chudnovsky and Ouezdou (12), employed a technique of using experimentally *observed* COD's (Crack Opening Displacements) as the solution to the microcrack problem. The problem is

formulated by using the second Green's tensor (5,11,12) $\underset{\sim}{\Phi}(\underset{\sim}{\xi},\underset{\sim}{x})$ which is defined as the displacement response at a point $\underset{\sim}{x}$ due to a unit discontinuity at a point $\underset{\sim}{\xi}$. For the case of plane stress, the second Green's tensor is given by :

$$[7.3] \qquad \underset{\sim}{\Phi}(\underset{\sim}{\xi},\underset{\sim}{x}) = -\frac{(1+\nu)}{4\pi R^2}\Big[(1-2\nu)(\underset{\sim}{n}_{\xi}\underset{\sim}{R}-\underset{\sim}{R}\underset{\sim}{n}_{\xi} + \underset{\sim}{n}_{\xi}\underset{\sim}{R}\underset{\sim}{I}) + 2\frac{\underset{\sim}{n}_{\xi}\underset{\sim}{R}}{R^2}\underset{\sim}{R}\underset{\sim}{R}\Big]$$

where $\underset{\sim}{n}_{\xi}$ is the unit normal vector to the surface across which the discontinuity takes place, ν is poissons ratio, $\underset{\sim}{I}$ is the unit second rank tensor and $\underset{\sim}{R}$ is the position vector (i.e., $\underset{\sim}{R} = \underset{\sim}{\xi}-\underset{\sim}{x}$).

The second Greens tensor in Equation 7.3 is then used to to describe the displacement response on the main crack by integrating over the observed displacements in the microcrack array. Afterward the displacement response is converted to stress by applying an appropriate stress operator T_x which transforms the displacement field u_k into stress σ_{ij}. Once the stress field is known, an effective stress intensity factor can evaluated.

Even though this approach was applied to the 2–D problem for the case of Ouezdou (12), and Chabat(5), it can be easily extended to the 3–D case.

In the next sections of this chapter, a semi–empirical stress and energy analysis will be outlined and illustrated for the case of overconsolidated clays. This analysis will be based on a method of complex potentials but will follow the same general philosophy described above.

7.5 Semi–Empirical Approach via Complex Potentials

Many problems in the theory of elasticity can be solved with great mathematical simplicity by using *complex variables* (13,23,33). A complex variable z is formed by two real variables x and y such that:

[7.4] $$z = x + iy$$

where i represents $\sqrt{-1}$ and is called the *imaginary unit.* Muskhelishvili (23) has shown that, for plane problems, many solutions can be expressed in terms of a pair of complex potentials. Once the potentials are defined for a particular problem, the stresses and displacements can be expressed as functions of the potentials (i.e., analogous to an Airy's Stress Function). If the complex potentials are denoted by ϕ and ψ, then the displacements can be defined as:

[7.5] $$2G(u + iv) = \kappa\phi - \bar{z}\,\overline{\phi}' - \overline{\psi}$$

Here u and v are the Cartesian components of displacement, G is the shear modulus, and κ is related to poissons ratio as defined in Equation 7.6.

[7.6] $$\kappa = \begin{Bmatrix} 3-4\nu \rightarrow \text{plane strain} \\ \frac{3-\nu}{1+\nu} \rightarrow \text{plane stress} \end{Bmatrix}$$

Also, the overbar "$\bar{}$" denotes the conjugate of the variable or function and the prime "$'$" denotes differentiaton of the function with respect to the complex variable z.

Similarly, the components of stress can be expressed as:

[7.7]
$$\sigma_{xx} + \sigma_{yy} = 2(\phi' + \overline{\phi}')$$
$$\sigma_{yy} - \sigma_{xx} + 2i\tau_{xy} = 2(\bar{z}\phi'' + \psi')$$

In the following subsections (i.e., Subsections 7.5.1, 7.5.2, and 7.5.3) solutions to the three problems outlined in Section 7.2 will be presented in complex potential form. In these sections, the following notation will be used.

[7.8]
$$\Phi(z) = \phi'(z)$$
$$\Psi(z) = \psi'(z)$$

7.5.1 Remote and Nominal Potentials

This Subsection presents the potentials for the stress/displacement fields caused by the remote (Case 1 in Section 7.2) and the nominal (Case 2 in Section 7.2) loading conditions.

From the methods in (23), the potentials for the remote loading

condition are given in Equation 7.9.

[7.9]
$$\Phi_\infty(z) = \tfrac{1}{4}\sigma_\infty$$
$$\Psi_\infty(z) = \tfrac{1}{2}\sigma_\infty + i\tau_\infty$$

Similarly, the stresses and displacements produced from the main crack as expressed in Case 2 is given in (24,3) by:

[7.10]
$$\Phi_n(z) = \frac{1}{2\sqrt{2\pi z}}\left(K_I^n + iK_{II}^n\right)$$
$$\Psi_n(z) = \overline{\Phi_n(\overline{z})} - \Phi_n(z) - z\Phi_n'(z)$$

Substituting the results of Equation 7.2 for K_I^n and K_{II}^n into Equations 7.10 gives:

[7.11]
$$\Phi_n(z) = \frac{\sqrt{c}}{2\sqrt{2z}}\left(\sigma_\infty + i\tau_\infty\right)$$
$$\Psi_n(z) = \frac{\sqrt{c}}{2\sqrt{2}}\left[\sigma_\infty(\overline{z}^{-\frac{1}{2}} - \tfrac{1}{2}z^{-\frac{1}{2}}) - i\tau_\infty(\overline{z}^{-\frac{1}{2}} + \tfrac{1}{2}z^{-\frac{1}{2}})\right]$$

7.5.2 Complex Potentials for a Crack and a Dislocation Dipole

In the complex potential formulation, discontinuities (damage) are

modeled as a distribution of dislocations. In 1978, K. K Lo (20) presented a solution (in the form of a pair of complex potentials) for a finite length crack and a single edge dislocation embedded in infinite plate. The edge dislocation essentially models a sudden change in the displacement field (either a sudden jump, a sudden shift, or a combination of the two).

In Lo's solution, the crack is located on the real/imaginary plane horizontally with its tip at the origin. The crack has a length of $2c$ and the edge dislocation is located at coordinates z_0 as shown in Figure 7.2. With this solution, the researcher can compute the stress field at a point z due to an edge dislocation at a point z_0 interacting with a crack.

The pair of potentials which make up Lo's solution, Φ and Ψ, can be written as:

[7.12]
$$\Phi_e = \Phi_1 + \Phi_2$$
$$\Psi_e = \Psi_1 + \Psi_2$$

where:

[7.13]
$$\Phi_1(z) = \frac{\alpha}{(z-z_0)}$$
$$\Psi_1(z) = \frac{\bar{\alpha}}{(z-z_0)} + \frac{\alpha \bar{z}_0}{(z-z_0)^2}$$

and

[7.14]
$$\Phi_2(z) = -\Big[\, \alpha F(z,z_0) + \alpha F(z,\bar{z}_0) + \bar{\alpha}(z_0-\bar{z}_0)G(z,\bar{z}_0) - \alpha X(z)\Big]$$

$$\Psi_2(z) = \overline{\Phi_2(z)} - \Phi_2(z) - z\Phi_2'(z)$$

with

$$F(z,z_0) = \frac{\frac{1}{2}\left[1 - \frac{X(z)}{X(z_0)}\right]}{(z-z_0)}$$

[7.15]
$$G(z,z_0) = \frac{\partial}{\partial z_0}\Big[F(z,z_0)\Big]$$

$$= \frac{\frac{1}{2}\left[1 - \frac{X(z)}{X(z_0)} + (z-z_0)\frac{X(z)X'(z_0)}{X(z_0)^2}\right]}{(z - z_0)^2}$$

[7.16]
$$X(z) = \frac{1}{\sqrt{z}\ \sqrt{z+2c}}$$

The dislocation data is included in the complex constant α which is defined by the relation in Equation 7.17.

[7.17]
$$\alpha = \frac{\mu b}{\pi i(\kappa+1)}$$

In Equation 7.17, μ is the shear modulus, κ is related to Poisson's ratio (Equation 7.6), and b is a Burgers vector which denotes the jump in the displacement at z_0 (Equation 7.18).

[7.18] $$b = e^{i\theta}\Big\{[u_r] + i[u_\theta]\Big\}$$

The quantities $[u_r]$ and $[u_\theta]$ denote the jumps in the tangential and normal displacements across the dislocation line. It should be understood that this solution (Equations 7.12 to 7.18) models a finite length crack ($2c$) in an infinite plate interacting with a semi–infinite jump and/or shift in the displacement field occurring z_o (defined by Equations 7.16 and 7.17).

In order to model the observed slip lines and, in general, finite sized cracks and micro–cracks, a solution for a finite jump and/or shift in displacement interacting with a crack is required.

In 1988, Ballarini and Denda (2) derived an analytical solution for a dislocation dipole interacting with a semi–infinite crack. In their derivation, they used Lo's solution (20) and superimposed the effects of a pair of edge dislocations, with Burgers vectors equal in magnitude but opposite in direction separated by an infinitesimal distance.

This solution (2), is limiting in our studies because the the crack length (notch length) is in many cases smaller than the preceding damage (see Chapter 4). Thus, a solution for the *finite sized* dislocation dipole will be introduced using the philosophy introduced by Ballarini and Denda (2) along with the potentials presented by Lo (20). For the finite sized dislocation dipole, the principle of superposition will be invoked and a pair of edge dislocations will be

used. In this approach the spacing dz_0 is not required to be infinitesimally small. This is illustrated in Figure 7.3. Consequently, the complete solution for the dislocation dipole interacting with a crack is given by:

$$\Phi_d(z,z_0) = \Phi_e(z,z_0,\alpha) + \Phi_e(z,z_0+dz_0,-\alpha)$$
$$\Psi_d(z,z_0) = \Psi_e(z,z_0,\alpha) + \Psi_e(z,z_0+dz_0,-\alpha) \qquad [7.19]$$

where Φ_e and Ψ_e are given by Equation 7.12. A similar expression can be written for an array of dipoles. This problem is the one of most practical interest since slip lines (see Chapter 3), cracks, microcracks and other discontinuities within a continuum can be modeled with an array of dislocations and/or dislocation dipoles. Since the law of superposition is valid for potentials, the complex potentials for an array of dipoles can be written as:

$$\Phi_d^{tot}(z) = \sum_{i=1}^{n} \Phi_d(z,z_{0_i})$$
$$\Psi_d^{tot}(z) = \sum_{i=1}^{n} \Psi_d(z,z_{0_i}) \qquad [7.20]$$

7.6 Stress Field Evaluation

The information presented in Sections 7.2 through 7.5 provide the

necessary tools to compute the stress field in the vicinity of the crack and damage. In principle, the slip surfaces shown in Figures 3.1 and 3.2 can be modeled with a distribution of dislocation dipoles. Thus, measured slippage along the discontinuity lines whose orientation and magnitude can be extracted from photographs can be modeled with an array of dipoles (see Equation 7.20).

A fortran routine was written to generate contour plots of selected components of stress using the information Sections 7.2 through 7.5. The program was named STRESS and a listing is included in Appendix A. The program STRESS, utilizes the complex operations available in the Mircosoft Version 3.2 Fortran Compiler to compute the stress field from the potentials. Input for this routine includes the slip line configuration and the Crack Sliding Displacement (CSD) along each slip line. Other input information includes the remote loading stress (σ_∞, τ_∞), crack length, elastic modulus, condition of plane stress or plane strain.

The output of program STRESS is a contour map of equal stress component magnitude for the component selected. Six components of stress are available. They include the Cartesian components of stress, σ_{xx}, σ_{yy}, and τ_{xy}, as well as the principle stresses (maximum and minimum) σ_{p1}, and σ_{p2}, and the maximum shear stress, τ_{max}. These components are defined in the Mohr's diagrams in Figure 7.4.

It should be noticed that the maximum principle stress σ_{p1} is defined

herein as the principle stress component which is largest in magnitude. Thus, as the Mohr's circle translates through a zone of pure shear, the σ_{p1}, component will change sign (see Figure 7.4). When the σ_{p1} contour plot is inspected, zones of pure shear are readily identified since the sign change in σ_{p1} produces a sharp transition from dashed (compressive) to solid (tensile) lines or vise versa. This phenomena will be illustrated in some following examples.

For all of the of the stress and energy analyses performed in this chapter, the following parameters were used:

$E = 11200$ psi

$\nu = 0.48$

plane strain

$\tau_\infty = 20.0$ psi

$\sigma_\infty = -30.0$ psi or 0.0 psi

The first case analyzed was to plot contours of equal stress for the case of a single crack loaded under pure shear (mode K_{II}). Five components of stress are illustrated in Figure 7.5; the three Cartesian stress components (σ_{xx}, σ_{yy}, τ_{xy}), the maximum principle stress σ_{p1}, and the maximum shear stress τ_{max}. Notice that the maximum shear stress need not be zero on the crack face since only two of the three stress components (σ_{yy}, τ_{xy}) are specified to zero by the crack face boundary conditions.

Next, stress field contour plots were generated illustrating the

interaction between the main crack (notch), and a single dislocation dipole symmetrically located in front of the crack. A unit of sliding in the same direction as observed in Figures 3.1 and 3.2 is was assigned to the dipole. For this case, no remote loading was considered (i.e., $\sigma_\infty = \tau_\infty = 0$) since the objective here is to isolate the effects due to the localized displacement. A schematic illustrating this problem is included in Figure 7.6. The contour plots for σ_{xx}, σ_{yy}, τ_{xy}, σ_{p1}, and τ_{max} are included in Figures 7.7 through 7.11, respectively. In these plots, the asymptotic stress field around the crack tip is *magnified* placed in the lower left corner of the plots. The asymptotic plots in the lower left corners for Figures 7.7 through 7.11 are plotted at the same scale as those in Figure 7.5. This was done so the effects of the dislocation dipole on the crack can be readily compared to those produce by remote loading.

The next case analyzed was the case of a single dislocation dipole interacting with the main crack (similar to the above case), but the dipole was oriented so as to produce a 20° angle with the x–axis (see Figure 7.12). The 20° angle was selected since it represented the typical orientation of the slip surfaces (see Figures 1.6, 3.1, 3.2). Again, five contour plots illustrating the crack–dipole interaction are included in Figures 7.13 through 7.17.

Next we will illustrate how this crack–mircocrack interaction can be investigated. The microcracks are to be modeled with an array of dislocation dipoles.

During the *damage–zone initiation* a dog–bone cracking configuration was observed and is shown in Figure 7.18. The dog–bone cracks are modeled with a distribution of dipoles such that they produce an elliptical crack sliding distribution (this corresponds to a constant traction assigned to each dog–bone crack). The configuration for this case is illustrated in Figure 7.19, and stress contour plots for τ_{xy}, σ_{p1}, and τ_{max} are included in Figures 7.20, 7.21, and 7.22, respectively.

7.7 Stress Intensity Factor Analysis

The stress intensity factors can be defined in complex form (2,3,25) as:

$$[7.21] \qquad K_I + iK_{II} = \lim_{\sqrt{z}\to 0} \sqrt{2\pi z}\,(\sigma_{yy} + i\tau_{xy})$$

In order to compute the SIFs (Stress Intensity Factors) caused by the dislocation dipole, we start by determining the SIF for a single edge dislocation. Here, we substitute substitute Equations 7.7, and 7.8 into 7.12 to yield:

$$[7.22] \qquad K_I + iK_{II} = \lim_{\sqrt{z}\to 0} \sqrt{2\pi z}\left[\Phi_e(z,z_0) + \overline{\Phi_e(z,z_0)} + \bar{z}\Phi'_e(z,z_0) + \Psi_e(z,z_0)\right]$$

Substituting Equations 7.12 through 7.18 into 7.22 and taking the limit results in Equation 7.23

$$[7.23] \qquad K_{\mathrm{I}}^{\mathrm{e}} - iK_{\mathrm{II}}^{\mathrm{e}} = 2\sqrt{\frac{\pi}{c}} \left\{ -\alpha \left[\Re \left(\sqrt{\frac{z_0+2c}{z_0}} \right) - 1 \right] + \frac{i\,\bar{\alpha}\,c\,\Im(z_0)}{(\sqrt{\bar{z}})^3 (\sqrt{\bar{z}_0+2c})} \right\}$$

In Equation 7.23, $\Re()$ denotes the *real* part of a complex value and $\Im()$ denotes the *imaginary* part.

Since the law of superposition is valid for stress intensity factors, the SIFs obtained for a single edge dislocation can be summed in a similar manner to that for the potentials (see Equations 7.19 and 7.20). Consequently, the SIFs for a single dislocation dipole and an array of dipoles can be expressed in Equations 7.24 and 7.25.

$$[7.24] \qquad \begin{aligned} K_{\mathrm{I}}^{\mathrm{d}} &= K_{\mathrm{I}}^{\mathrm{e}}(z_0,\alpha) + K_{\mathrm{I}}^{\mathrm{e}}(z_0+dz_0,-\alpha) \\ K_{\mathrm{II}}^{\mathrm{d}} &= K_{\mathrm{II}}^{\mathrm{e}}(z_0,\alpha) + K_{\mathrm{II}}^{\mathrm{e}}(z_0+dz_0,-\alpha) \end{aligned}$$

$$[7.25] \qquad K_{\mathrm{I}}^{\mathrm{dtot}} = \sum_{i=1}^{n} K_{\mathrm{I}i}^{\mathrm{d}}$$

$$K_{\mathrm{II}}^{\mathrm{dtot}} = \sum_{i=1}^{n} K_{\mathrm{II}i}^{\mathrm{d}}$$

Three different fortran programs were written to perform various analyses related to the stress intensity factor study. These routines, and their

respective results will be discussed in the following paragraphs.

The first of the three routines is called *KIJ* (listing included in Appendix A) which plots a non–dimensionalized SIF vs. angular variation β (for $z_0 = x_0 + iy_0$; $\beta = tan^{-1}(y_0/x_0)$) denoting the coordinates of the dislocation dipole. The SIF's are non–dimensionalized by the relation expressed in Equation 7.26.

[7.26] $$K_I^* - iK_{II}^* = \sqrt{\frac{\pi}{2}} \frac{(\kappa+1)\rho^{\frac{3}{2}}}{\mu \; d\rho \, |b|} (K_I - iK_{II})$$

where

$$|b| = \sqrt{b_x^2 + b_y^2}$$

$$z_0 = x_0 + iy_0$$

[7.27] $$z_0 + dz_0 = (x_0 + dx_0) + i(y_0 + dy_0)$$

$$\rho = \sqrt{x_0^2 + y_0^2}$$

$$dp = \sqrt{(dx_0)^2 + (dy_0)^2}$$

The effective SIF's for two different cases were plotted; one due to an opening dislocation dipole (Case for b_y) oriented with $\Theta = 0^o$ (Θ is defined in Equation 7.18), and another shear dislocation dipole (Case for b_x) with $\Theta = 0^o$. Plots for these two cases are presented in Figure 7.23. These results agree with those presented by Ballarini (2,3) for the case of an infinitely long main crack. Figure 7.23 shows the effect (i.e., amplification or shielding) the dislocation dipole has on the main crack. These are summarized as:

$$\text{For the Case of } b_y \begin{cases} \text{Mode–I} \begin{cases} \beta < 69^o & \rightarrow \text{Amplification} \\ \beta > 69^o & \rightarrow \text{Shielding} \end{cases} \\ \text{Mode–II} \begin{cases} \beta < 36^o,\ \beta > 110^o & \rightarrow \text{Amplification} \\ 36^o < \beta < 110^o & \rightarrow \text{Shielding} \end{cases} \end{cases}$$

and

$$\text{For the Case of } b_x \begin{cases} \text{Mode–I} \begin{cases} \beta < 36^o,\ \beta > 110^o & \rightarrow \text{Amplification} \\ 36^o < \beta < 110^o & \rightarrow \text{Shielding} \end{cases} \\ \text{Mode–II} \begin{cases} \beta < 30^o, 83^o < \beta < 125^o & \rightarrow \text{Amplification} \\ 83^o < \beta < 125^o,\ \beta > 125^o & \rightarrow \text{Shielding} \end{cases} \end{cases}$$

Another fortran program called SIF (listing in Appendix A) was written to produce contour plots of equal levels of Green's function for the stress intensity factor denoted G_{sif}. In particular, $G_{\text{sif}}(x_1,x_2)$ defines the stress intensity factor at the crack tip caused by a dipole dislocation at coordinates (x_1,x_2). The magnitude for the stress intensity factor is non–dimensionalized by the following relation:

$$[7.28] \qquad G_{\text{sif}_I} - iG_{\text{sif}_{II}} = K_I^{**} - iK_{II}^{**} = \frac{1}{\sqrt{\pi c}\ |b|\, d\rho}(K_I - iK_{II})$$

Results from SIF are included in Figures 7.24 through 7.27. Figure 7.24 shows plots of equal stress intensity factor for both Modes I and II due to a single dislocation dipole with an applied unit opening displacement. Similarly, Figure 7.25 shows these results for Modes I and II due to a single dislocation with an applied unit shear displacement. These results (Figures 7.24 and 7.25)

compare with those presented by others (2,3,12). Figures 7.26 and 7.27 reproduce the same analyses performed in 7.24 and 7.25 except the dislocation dipole is oriented at 20°. The orientation $\Theta = 20°$ is analyzed since this is the typical orientation of the *LSD* lines. The information contained in these plots show where the zones of amplification and shielding occur and how these zones are affected when the discontinuities are inclined at 20°.

A third analysis on was performed on a particular configuration of crack and damage. Detailed photographs of the last cycle from the Cylinder 5–6 fatigue test were used to create the composite photograph of the crack and damage. The zone of damage (consisting of slip surfaces) was approximated with 27 descrete slip lines. A photograph of the zone and the approximating slip lines is illustrated in Figure 7.28. Measurements of CSD (Crack Sliding Displacement) were recorded for each of the 27 slip lines. Ten dislocation dipoles per slip line for a total of 270 dipoles were distributed to model the zone of damage. A Program call KMIN (listing in Appendix A) computed the Mode–II stress intensity factors caused by the damage array (Equations 7.24 and 7.25) and the remotely applied load (Equations 7.2) along with a cumulative result denoted K_{damage}, K_{crack}, and K_{total}, respectively. KMIN computed these values for various *crack tip locations* starting at the notch tip and extending through the damage zone. The SIF magnitudes are normalized by the K_{IIc} values obtained for this material by Kennedy (31). Plots of K_{crack}, K_{damage}, and K_{total} vs. *imaginary crack tip location* are illustrated in Figure 7.29. The reader should note that the total SIF (K_{total}) is approximately equal

to zero at the same location where the centroid of damage was determined for this cycle. This finding suggests that the Mode–II fracture in overconsolidated clays resembles that of a Dugdale–Barenblatt model (15,19) if an imaginary crack tip is located at the centroid of damage.

Although not explicitly reported, *KMIN* also computed the same SIF results for Mode–I. In general, Mode–I effects can be induced the unsymmetrical development of the damage zone (Figure 7.28). However, K_{Idamage} never exceeded more that 4% of K_{IIdamage} and the hydrostatic effects ($\sigma_\infty = -30$ psi) cancel any Mode–I propagation (i.e., $K_{\text{Itotal}} = 0$).

7.7.1 Stress Intensity Factor Analysis for Continuously Distributed Dislocations

It is recognized that the above results are only valid for the particular configuration shown in Figures 7.28 and 7.29 and that a more general approach would may rendure more worthy results. In this subsection we will outline a method for computing the SIF's caused from the distributed damage illustrated in Chapter 5.

Consider a square element of material extending through the cylinder thickness within the *LSD* damage zone as illustrated in Figure 7.30. It is expected that some of the *LSD* lines (surfaces) do not all extend to the outer

wall of the cylinder and thus the actual discontinuity density may be more realistically depicted by utilizing the *LSD* density measurements.

In general, the total SIF can be written as a combination of that due to the remotely applied load (K_o) along with that of the *LSD* array (ΔK_{sdz}). This can be expressed as:

[7.29] $$K_{tot} = K_o + \Delta K_{sdz}$$

where:

[7.30] $$K_o = \tau_\infty \sqrt{\pi c}$$

and:

[7.31] $$\Delta K_{sdz} = \mu \sqrt{\pi c} \int_{V_{s\,dz}} G_{s\,if}(\rho,\Theta)\ \mathscr{C}(x,y)\ dV$$

In Equation 7.31, $G_{sif}(\rho,\Theta)$ denotes the Green's Function for the stress intensity factor for a unit discontinuity located at ρ and oriented by Θ. The expression for the Green's Function is written in Equation 7.28 and is plotted for in Figures 7.24 through 7.27. $\mathscr{C}(x,y)$ in Equation 7.31 denotes the concentration of *LSD*'s within the damage zone and has a dimension of and can be written for a particular *LSD* orientation as:

[7.32] $$\mathscr{A}(x,y) = \frac{|b|\ d\rho}{dx\ dy}\, t$$

where t is the cylinder thickness.

Thus, the total SIF as expressed in Equation 7.31 can be written in the following form:

[7.33] $$K_{tot} = K_o\left[\ 1 + \frac{\mu}{\tau_\infty}\iint\limits_{V_{s\,dz}} G(x,y)\ \mathscr{A}(x,y)\ dx\ dy\right]$$

In order to employ Equation 7.33, we need only to define the *LSD* concentration $\mathscr{A}(x,y)$. A method will be proposed herein as an illustrative example for constructing the *LSD* concentration from the distributions constructed in Chapter 5.

If we assume that the number of elemental discontinuity lines are of average orientation (i,e, $\Theta = \pm\ 20^o$ depending if above or below the x–axis), and that the discontinuous displacement within a given element is equal to the sum dipole displacements within that element, the concentration for this element can be written determined.

In the example proposed in this subsection, a constant dipole displacement was assumed, which implies that the total discontinuous

displacement within across the damage zone is proportional to the volume of damage created. Further, the total number of dipoles within an element are assumed to be proportional to the damage density.

For simplicity, an *LSD* zone is assumed to have the simplified form of that shown in Figure 7.31. A series of programs called KMSM and PLDS (listing in Appendix A) performed the analysis. Figure 7.32 illustrates the results for the SIF due to damage only (ΔK_{sdz} in Equation 7.31) as well as a dislocation density scattergram in a comparable analysis to that performed for a particular configuration in Figure 7.29.

Notice that this generalized analysis also illustrates that a location where all of the stress field singularities cansel appears within the damage zone. This is similar to that shown in Figure 7.29 and illustrates in a more general sense that a phenonima of a modified Dugdale–Barenblatt model with a tip located within the $K_{tot} = 0$ vicinity may actually occur in our material.

7.8 Energy Release Rate Analysis

In this section we will introduce methods for computing the energy release rates for both the near (at the notch tip) and far (around the entire damage zone) fields. From LEFM (Linear Elastic Fracture Mechanics) the energy release rate of the notch tip is related to the SIF by Equation 7.34.

[7.34] $$J_{\text{near}} = \frac{(K_{\text{I}}^{\text{tot}})^2 + (K_{\text{II}}^{\text{tot}})^2}{E'}$$

where

[7.35] $$E' = \begin{Bmatrix} E & \rightarrow \text{plane stress} \\ \frac{E}{1-\nu^2} & \rightarrow \text{plane strain} \end{Bmatrix}$$

The energy release of the entire damage zone will be referred to herein as the far field ERR (Energy Release Rate) and denoted by J_{far}. Budiansky and Rice (4) provide expressions for the ERR in complex form as:

[7.36] $$J_{\text{far}} = \frac{2}{E'} \oint_{\Gamma} \left[(\Phi')^2 + 2\Phi'\Psi' \right] d\Gamma$$

Thus, J_{far} can be evaluated by performing a numerical contour integration around the microdefect array by using the potentials in Equations 7.9, 7.10, and 7.19.

Again, a fortran routine named RENG was written to perform such an analysis. This routine reads in the microdefect file and evaluates the near and far field ERR's. RENG utilizes a rhomberg integration technique to perform the contour integration and a listing is included in Appendix A.

The configuration shown in Figure 7.25 (Cylinder 5–6) was analyzed and the following results were obtained:

$J_{near} = 1528$ J/M^2

$J_{far} = 149$ J/M^2

7.9 Summary

The information contained in this chapter formulates the stress and energy analysis of the crack–damage interaction problem. The solutions are expressed in terms complex potentials and the results obtained from these analyses provide insight in:

- The asymptotic stress fields in the vicinity of the main crack, the stress fields for the crack–dipole and the stress fields for the dogbone crack configuration as applied to Mode–II conditions.
- Zones of shielding and amplification are identified through Green's Function for the SIF plots. These plots were produced for both Modes I and II for the case of the horizontal and inclined dipoles.
- An effective crack propagation within the damage zone is suggested through an elaborate crack–damage SIF study on a particular configuration as well as a generic zone constructed of continuously distributed dipoles.
- Far field energy release rates which relate to the translation of both crack and entire *LSD* is computed for a particular configuration and the results from this analysis can be used to properly proportion the measured ERR. For the case considered in Section 7.8, it appears that the energy release for *rigid* translation of the damage zone is approximately 10% of the total measured ERR. This factor will be used in Chapter 8 to adjust the measured ERR accordingly in the constitutive model.

References

1) Atkinson, C. (1962),"The Interaction Between a Dislocation and a Crack", International Journal of Mechanics, Vol. 2, pp 567–574

2) Ballarini, R., Mitsunori, D. (1988),"The interaction between a crack and a dislocation dipole" International Journal of Fracture, Vol. 37, pp 61–71.

3) Ballarini, R. "A semi–empirical analysis of micro–cracking in concrete', Proc. of th Int. Conf. of Fracture Mechanics , Houston ,In press.

4) Budiansky, B., Rice, J. R. (1973),"Conservation Laws and Energy Release Rates", Journal of Applied Mechanics, Trans. of ASME, March, pp. 201–203.

5) Chaabat, M., (1987),"Stress analysis in Polystyrene", PhD. Dissertation, Dept. of Civil Engineering, Case Western Reserve University

6) Cherepanov, G. P., Kuliev, V. D. (1975), "On Crack Twinning",Int. Journal of Fracture, Vol. 11, No. 1, pp 29–38

7) Chu, S., "Elastic Interaction Between Screw Dislocation and Surface Crack", J. Applied Physics, Vol. 53 (1982)

8) Chudnowsky, A., Dolgopolsky, A., Kachanov, M. (1984), "Advances in Fracture Research", Proc. of Int. Conf. Frac. 6, Vol. 2, ed. S.R. Valluri et al., Pergamon Press, Oxford, pp. 825–832.

9) Chudnowsky, A., Dolglopolsky, A., Kachanov, M. (1987) "Elastic Interaction of a Crack with a Microcrack Array –I. Formulation of the Problem and General form of the Solution", Int. Journal of Solid Structures, Vol. 23, No.1, pp 1–10

10) Chudnowsky, A., Kachanov, M. (1983), "Interaction of a Crack with a Field of Microcracks", Letters in Applied Eng. Sci., Vol. 21, No. 8, pp 1009–1018.

11) Chudnowsky, A., Wu, S. (1988), "Crack Layer Translational Energy Release Rates"

12) Chudnowsky, A., Ouezdou B. M. (1988), "Semi–empirical Crack Tip Analysis', has been submitted to the International Journal of Fracture.

13) Churchill, R. V., Brown, J. W. (1984), "Complex Variables and Applications", McGraw Hill Book Co., New York

14) Erdogan, F. (1962), "On the Stress Distributions in Plates With Collinear Cuts Under Arbitrary Loads ",Proceedings of the Fourth Int. U.S. National Congress of Applied Mechanics, Vol. I, pp 547–554.

15) Hellan, K. (1984), "Introduction to Fracture Mechanics", McGraw Hill Book Co., New York

16) Hoagland, R. G., Embury, J. D. (1980), "A Treatment of the Inelastic Deformation Around a Crack Tip due to Microcracking", Journal of the American Ceramic Soc., Vol. 63, pp 404–410

17) Kachanov, M. (1985), "A Simple Technique of Stress Analysis in Elastic Solids with many Cracks", Int. Journal of Fracture, Vol. 28, pp R11–R19

18) Kachanov, M., Montagut, E. (1986), "Interaction of a Crack with Certain Microcrack Arrays", Engineering Fracture Mechanics, Vol. 25, No. 5/6, pp 625–636

19) Kanninen, M. F., Popelar, C. H. (1985), "Advanced Fracture Mechanics", Oxford Eng. Sci. Series 15, Oxford University Press, New York

20) Lee, S. (1985), "A New analysis of Elastic interaction between a Surface Crack and a Parallel Screw Dislocation", Engineering Fracture Mechanics, Vol. 22, No. 6, pp 429–435

21) Lo, K. K. (1978), "Analysis of Branched Cracks", Journal of Applied Mechanics, Vol. 45, pp 797–802

22) Mendelson, A. (1973), "Boundary–Integral Methods in Elasticity and Plasticity", NASA Technical Note, TN D–7418,

23) Mendelson, A. (1986), "Advanced Elasticity", Class Notes, Dept. of Civil Eng., Case Western Reserve University

24) Muskhelishvili, N. L. (1958), "Some Basic Problems of Mathematical Theory of Elasticity", P. Noordhoff

25) Rice, J. R. (1968), "Mathematical Analysis in the Mechanics of Fracture", Chapter 3 in Fracture, Vol II, edited by Liebowitz, H., Academic Press, pp 192–311

26) Rice, J. R., Thompson, R. (1974), "On the Ductile verses Brittle Behavior of Crystals", Phil. Mag., Vol. 29, pp 73–97

27) Rose, L. R. F. (1986), "Microcrack interaction with a main crack", Int. Journal of Fracture, Vol. 31, pp 233–242

28) Rose, L. R. F.(1986), "A Kinematic Model for Stress–Induced Transformations around Cracks", Mechanics of Transformation Toughening, Journal of American Ceramic Soc, Vol. 69, pp 212–214

29) Rose, L. R. F., Swain, M. V.(1986), "Two R Curves for Partially Stabilized Zirconia", Mechanics of Transformation Toughening, Journal of American Ceramics Soc., Vol. 69, pp 203–207

30) Rubenstien, A. A., (1985), "Macrocrack Interaction with a Semi–infinite Microcrack Array", Int. Journal of Fracture, Vol. 27, pp 113–119

31) Saada, A. S., Chudnovsky, A., Kennedy, M. (1985)."A Fracture Mechanics Study of Stiff Clays," Proc. Eleventh Int. Conf. Soil Mech. Found. Engg., San Francisco, Vol. 2, pp 637 – 640.

32) Shiue, S., Lee, S., (1985), "A Thermodynamic Approach to the Interaction Between Dislocation and Crack and its Applications", Engineering Fracture Mechanics, Vol. 22, No. 6, pp 1105–1115

33) Wang, C. T. (1953), "Applied Elasticity", McGraw Hill Book Co., New York, pp 171–208

34) Willis, J. R. (1968), "The Stress Field Around an Elliptical Crack in an Elastic Medium", Int. Journal of Engineering Science, Vol. 5, pp 5

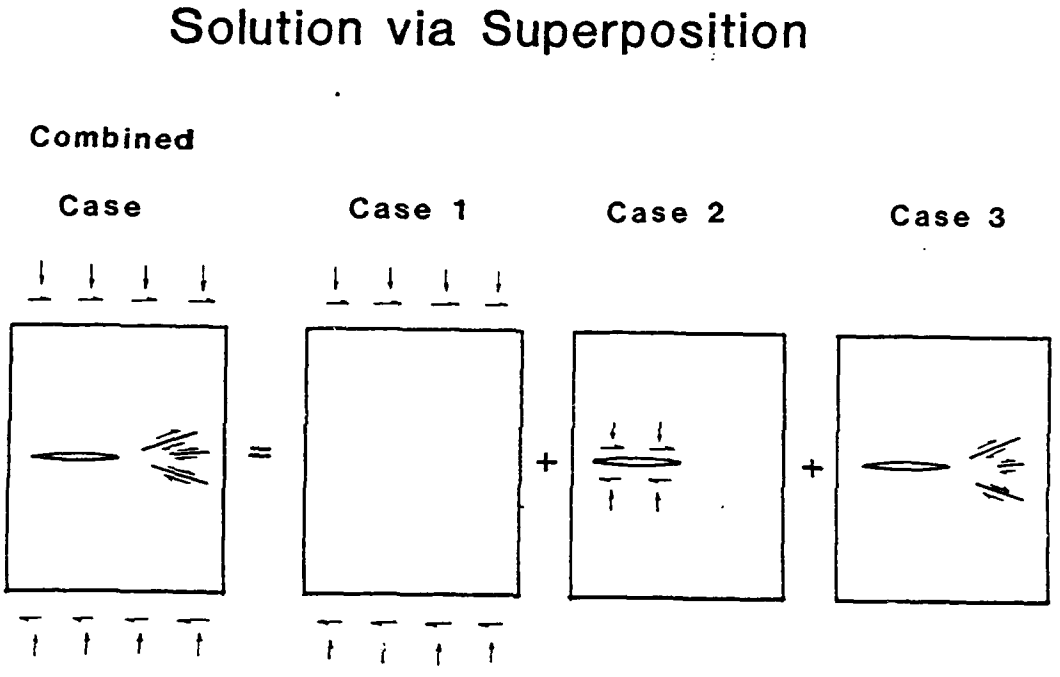


Figure 7.1: Three Cases which when superimposed produce a combined stress/displacement field observed in the experiment.

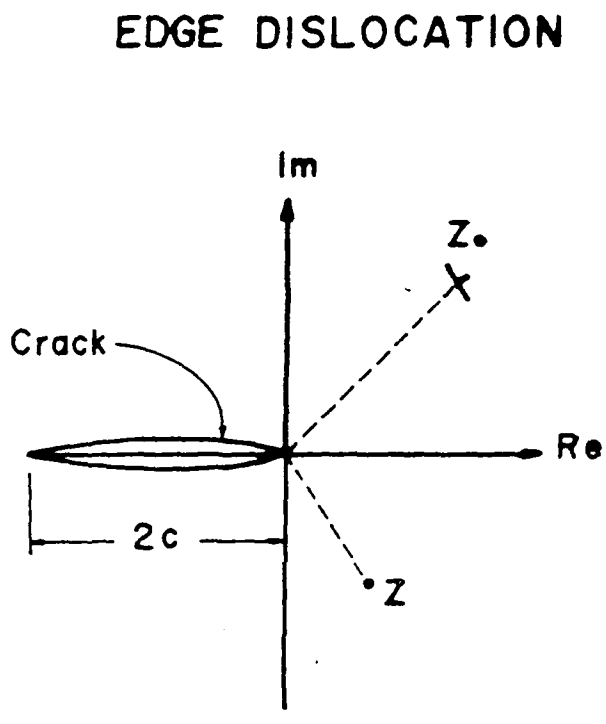


Figure 7.2: Plot of K. K. Lo's solution for a crack with an edge dislocation in the complex plane.

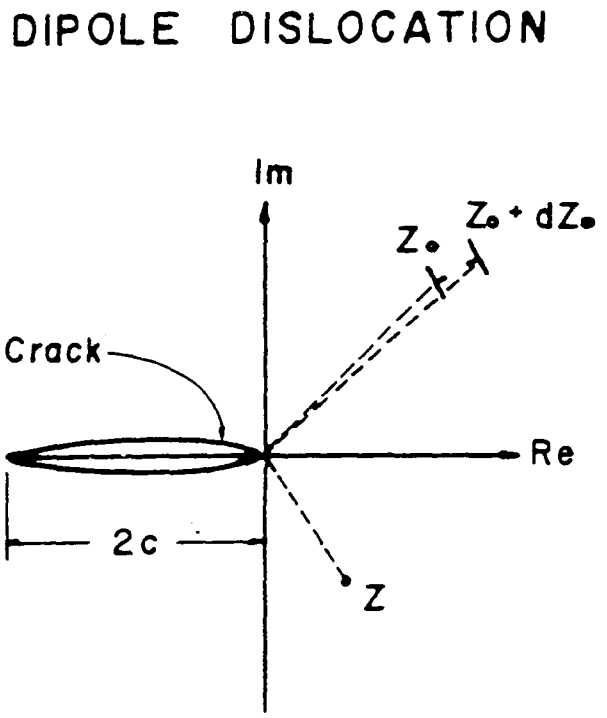


Figure 7.3: Plot of a crack with a dislocation dipole in the complex plane.

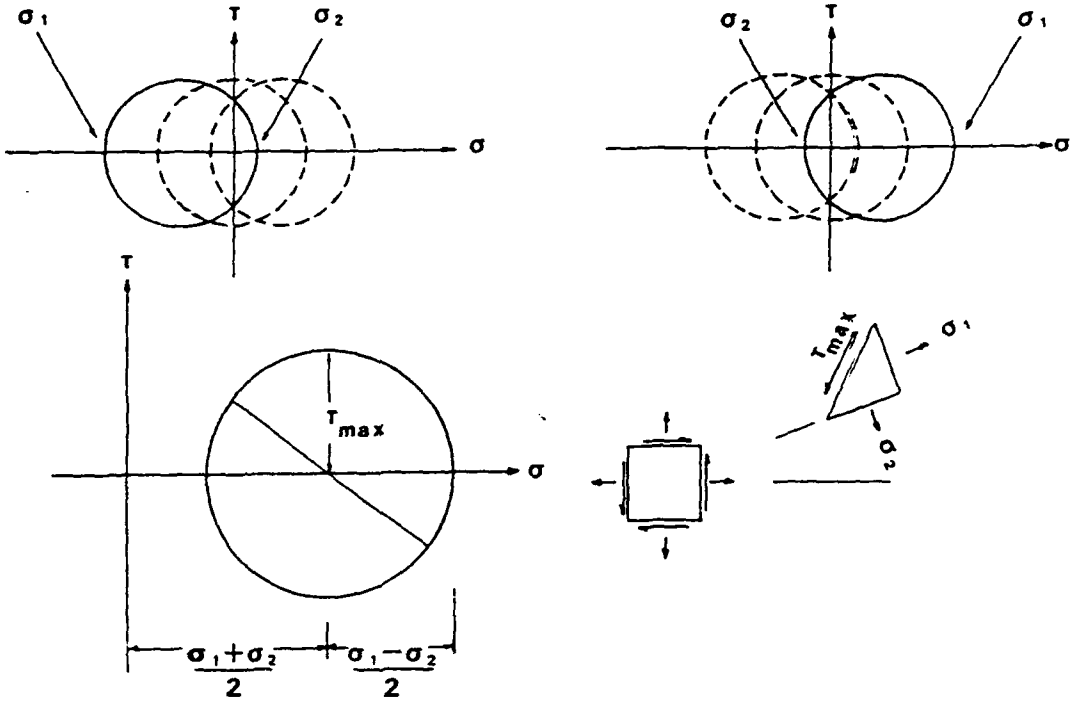


Figure 7.4: Mohr's circle diagrams defining the components of stress.

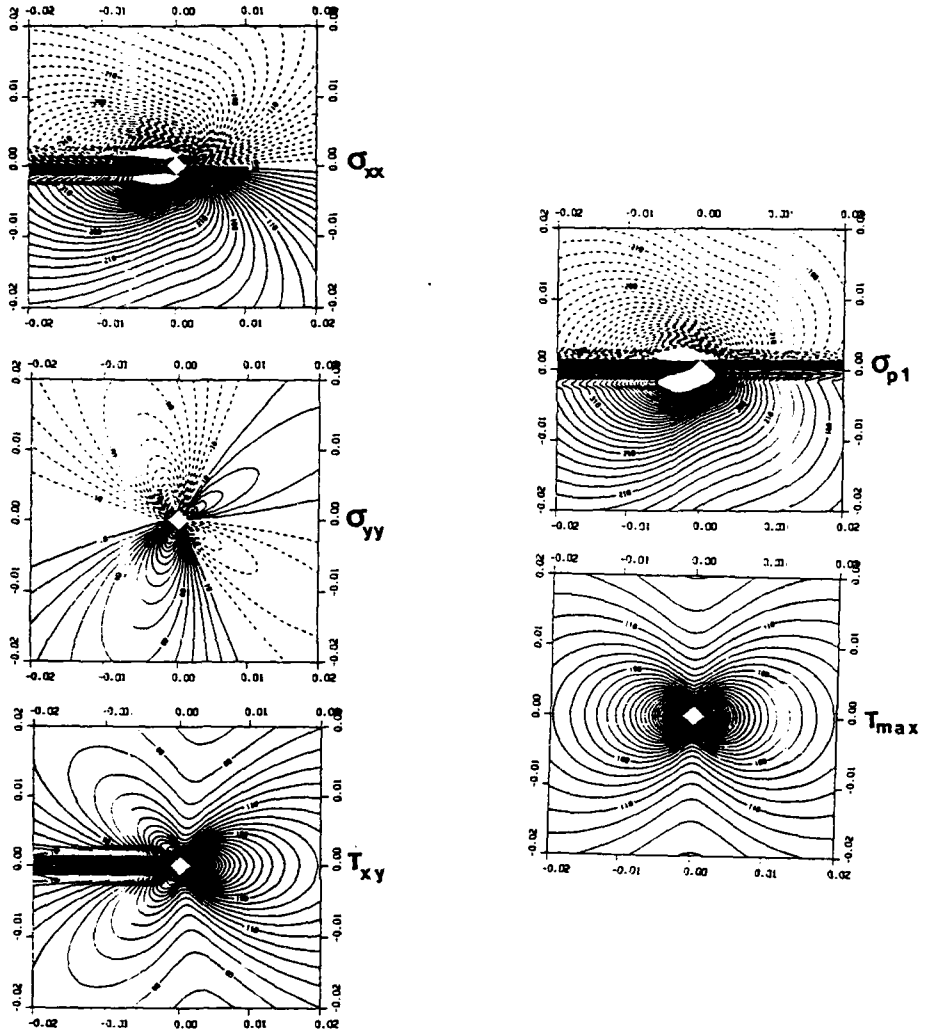


Figure 7.5: Plots of stress components in the vicinity of the crack tip resulting from a remotely applied shear stress.

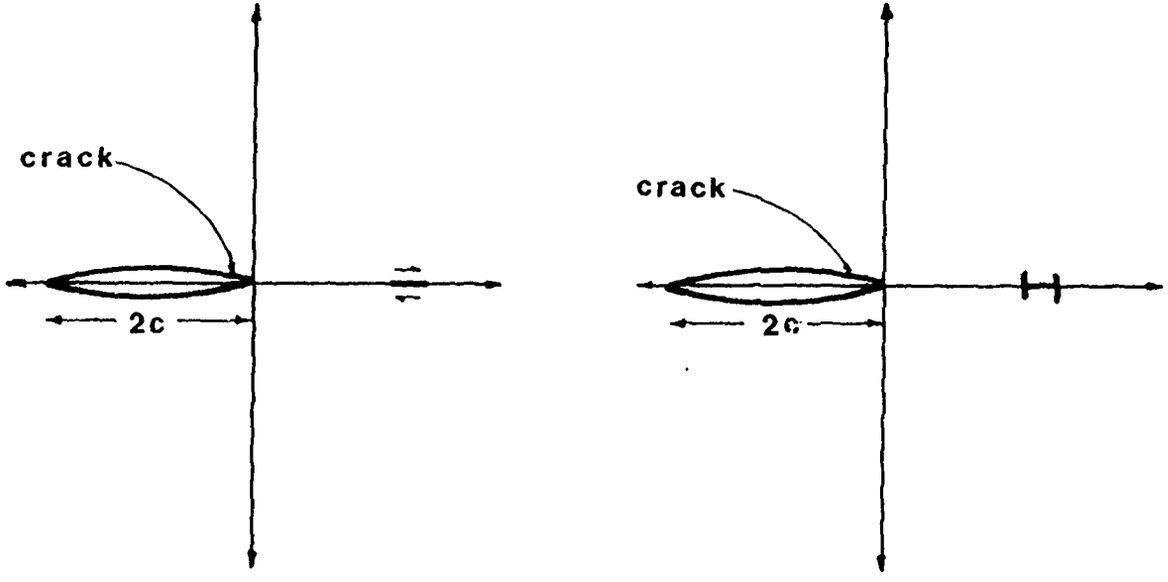


Figure 7.6: Schematic of a crack interacting with a single dislocation dipole located directly in front of the crack. The dipole is assigned a burgess vector which creates a unit shear discontinuity.

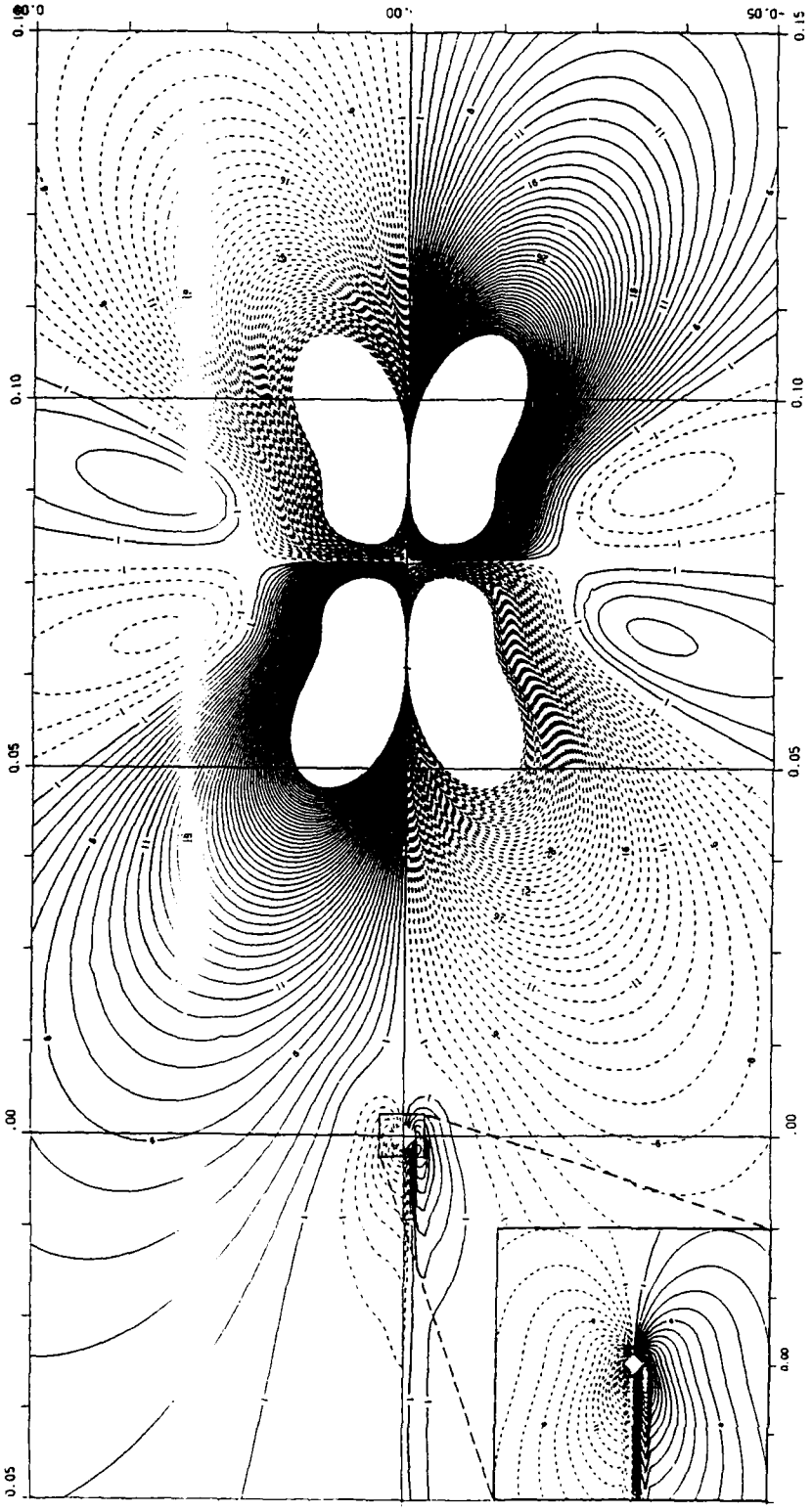


Figure 7.7: Contour plot of σ_{xx} component for the case in Figure 7.6.

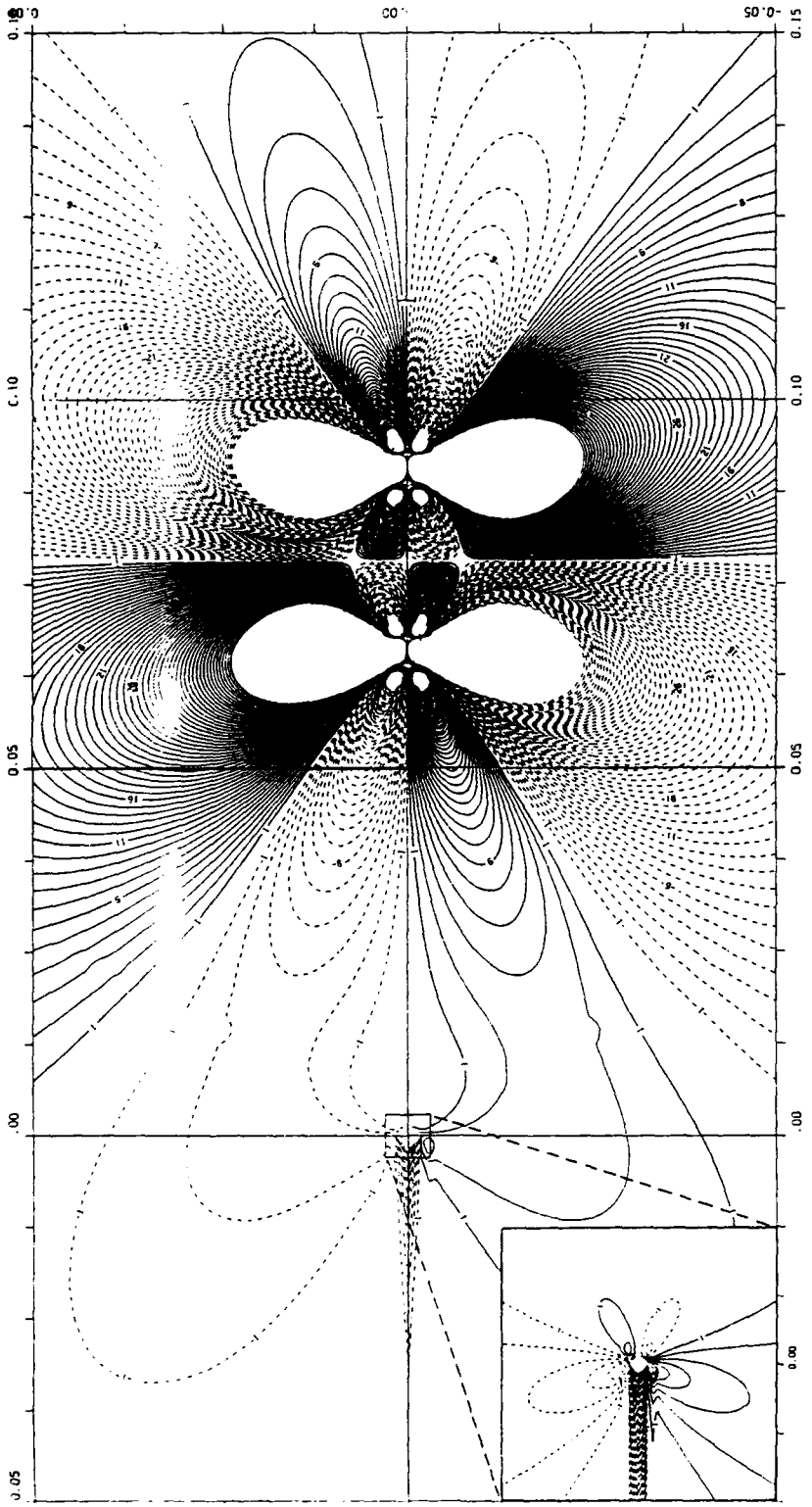


Figure 7.8: Contour plot of σ_{yy} component for the case in Figure 7.6.

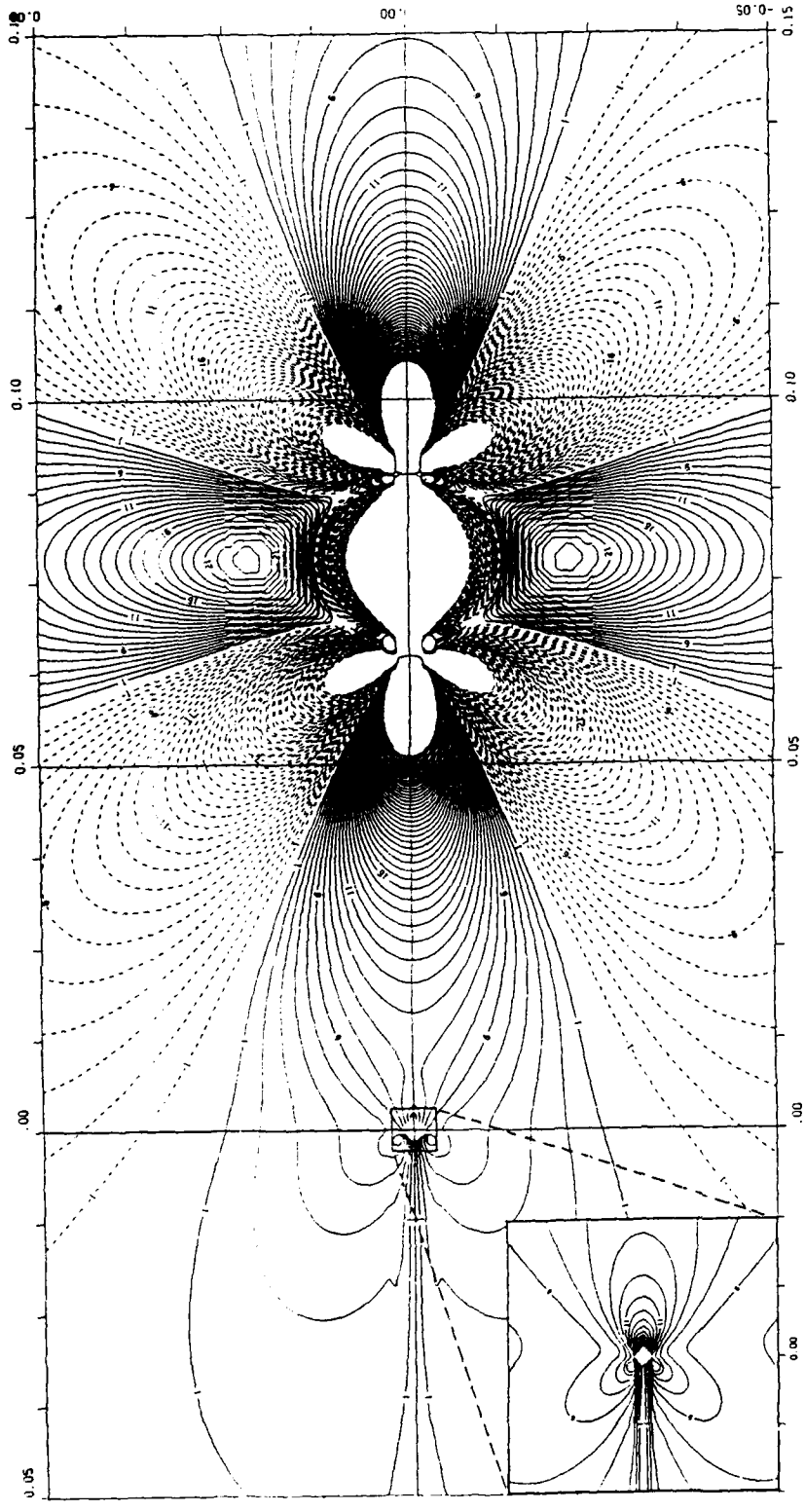


Figure 7.9: Contour plot of τ_{xy} component for the case in Figure 7.6.

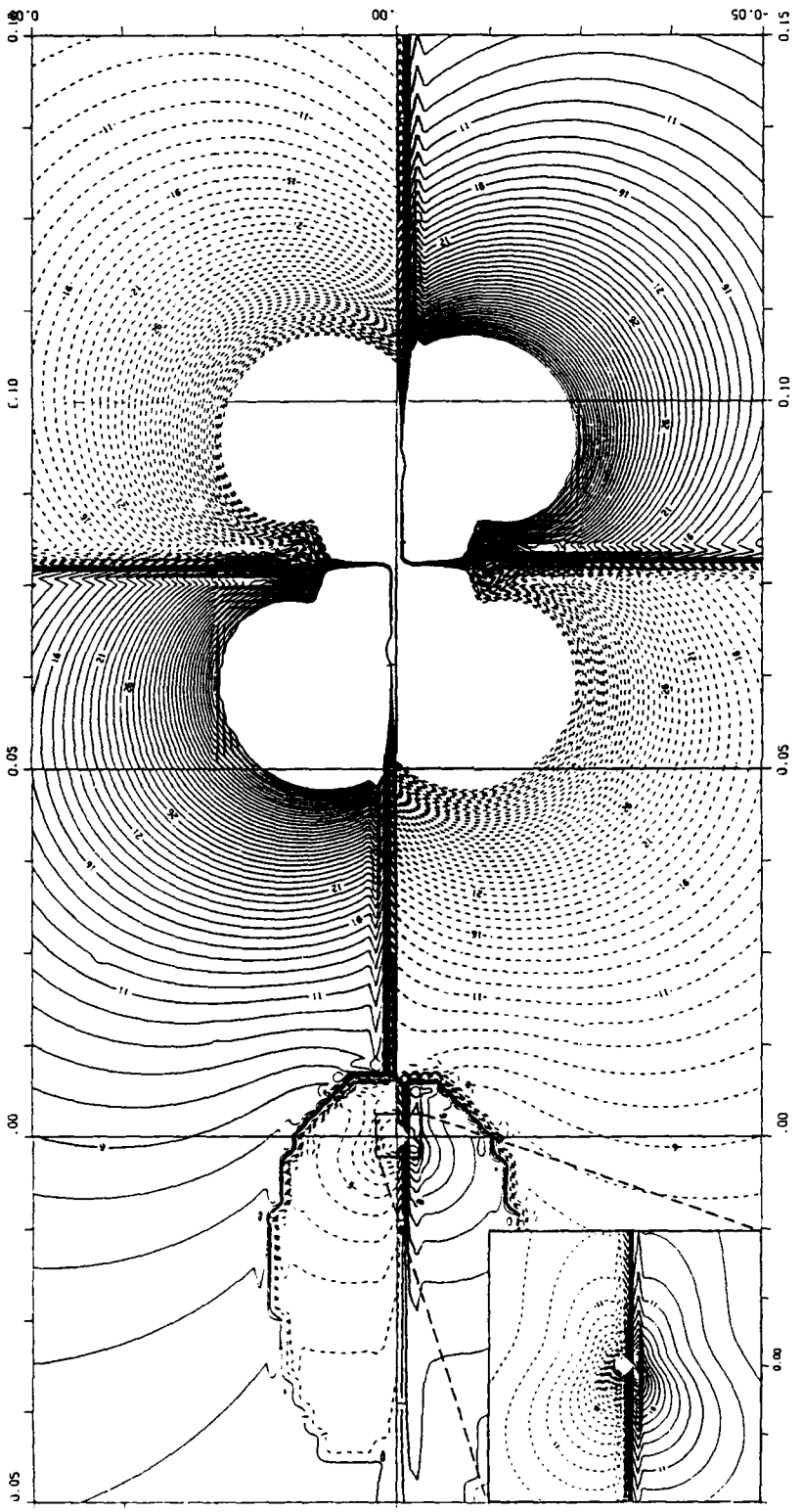


Figure 7.10: Contour plot of σ_{p1} component for the case in Figure 7.6.

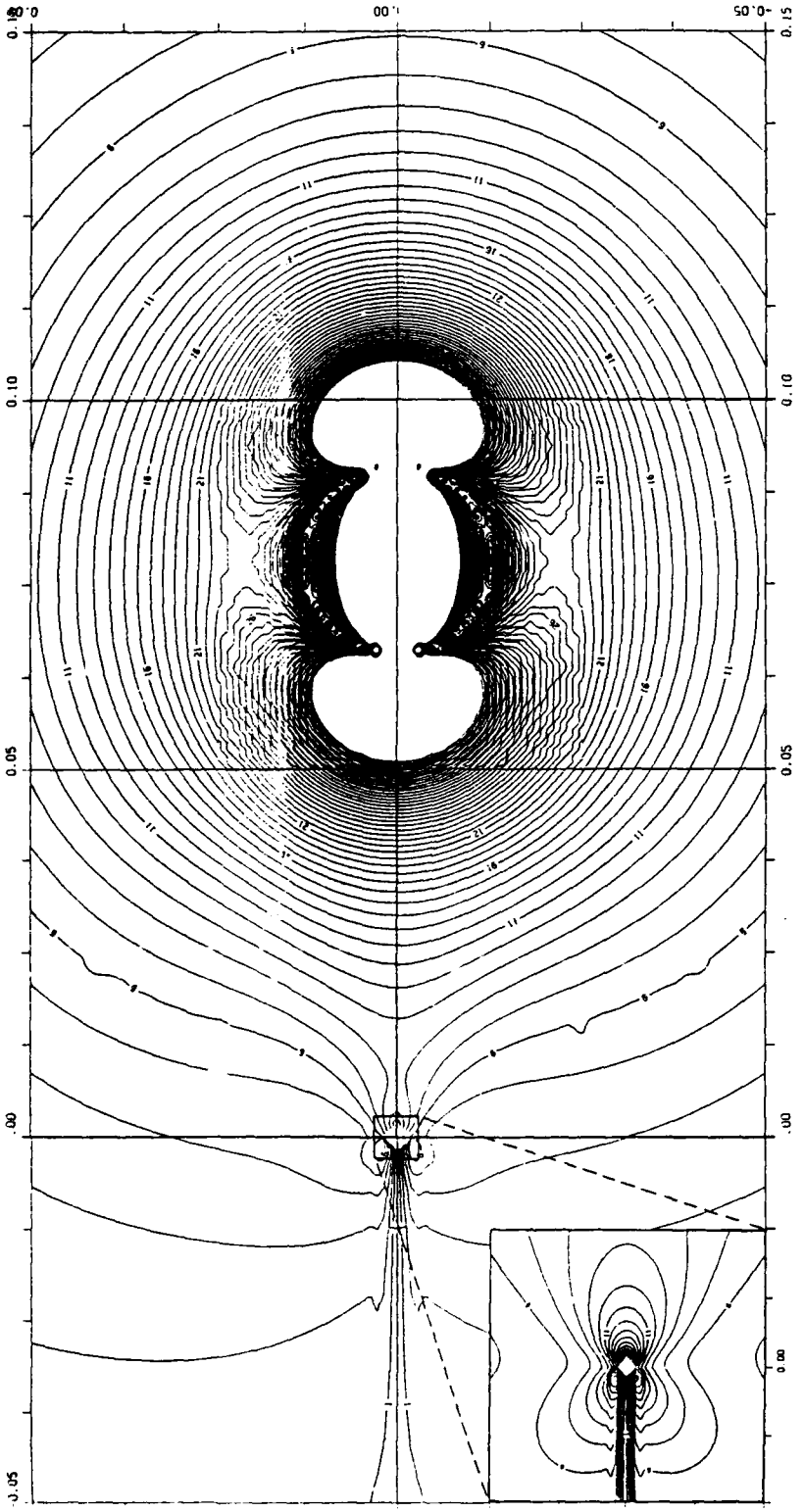


Figure 7.11: Contour plot of τ_{max} component for the case in Figure 7.6.

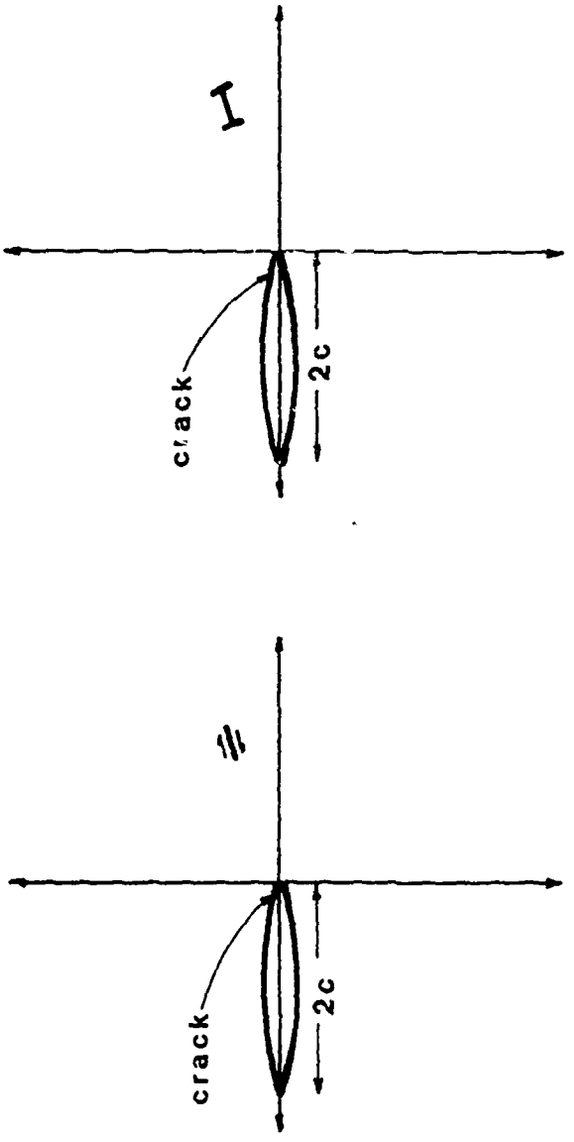


Figure 7.12: Schematic of a crack interacting with a single dislocation dipole located at a 20° inclination in front of the crack. The dipole is assigned a burgess vector which creates a unit shear discontinuity.

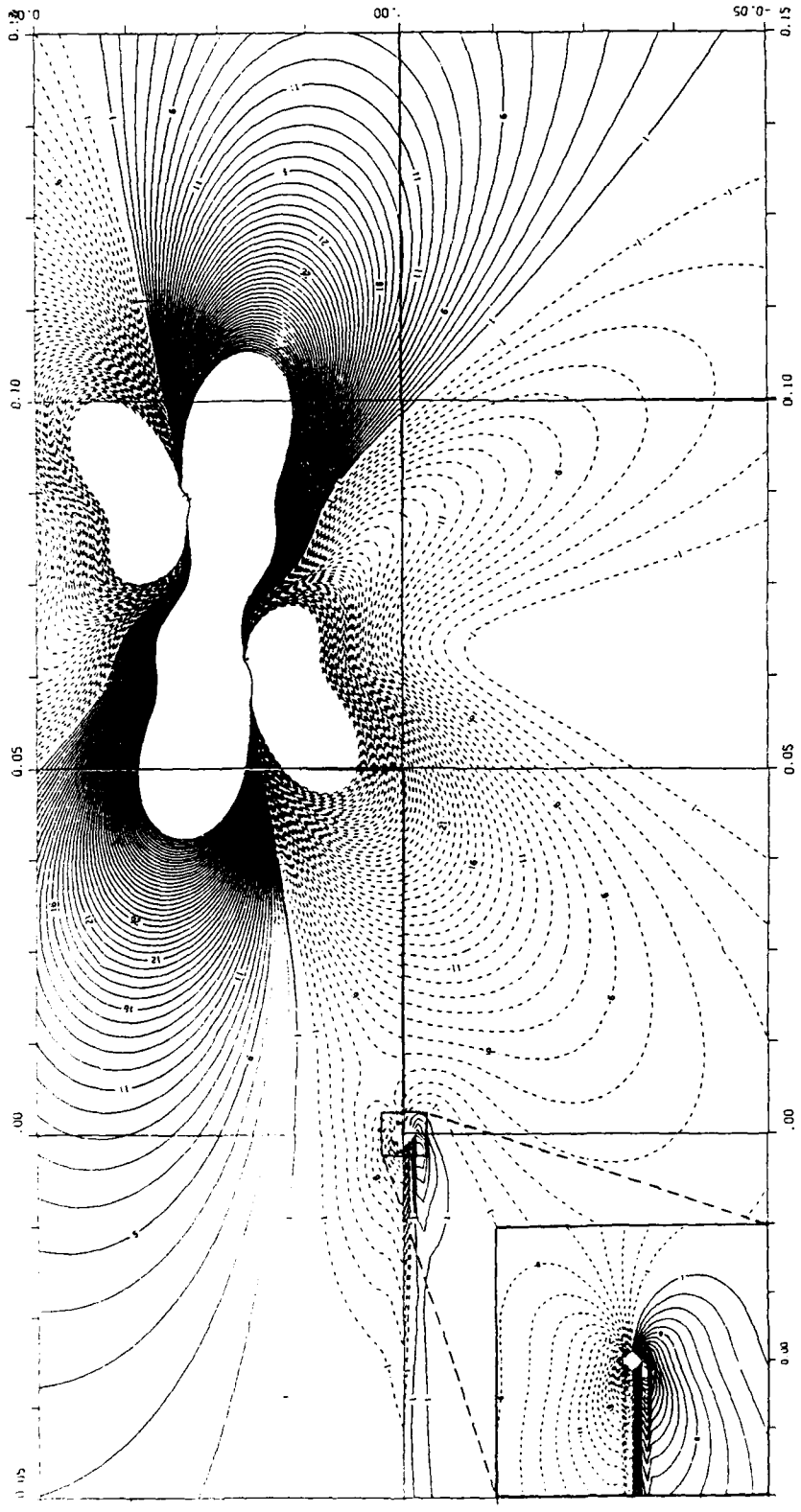


Figure 7.13: Contour plot of σ_{xx} component for the case in Figure 7.12.

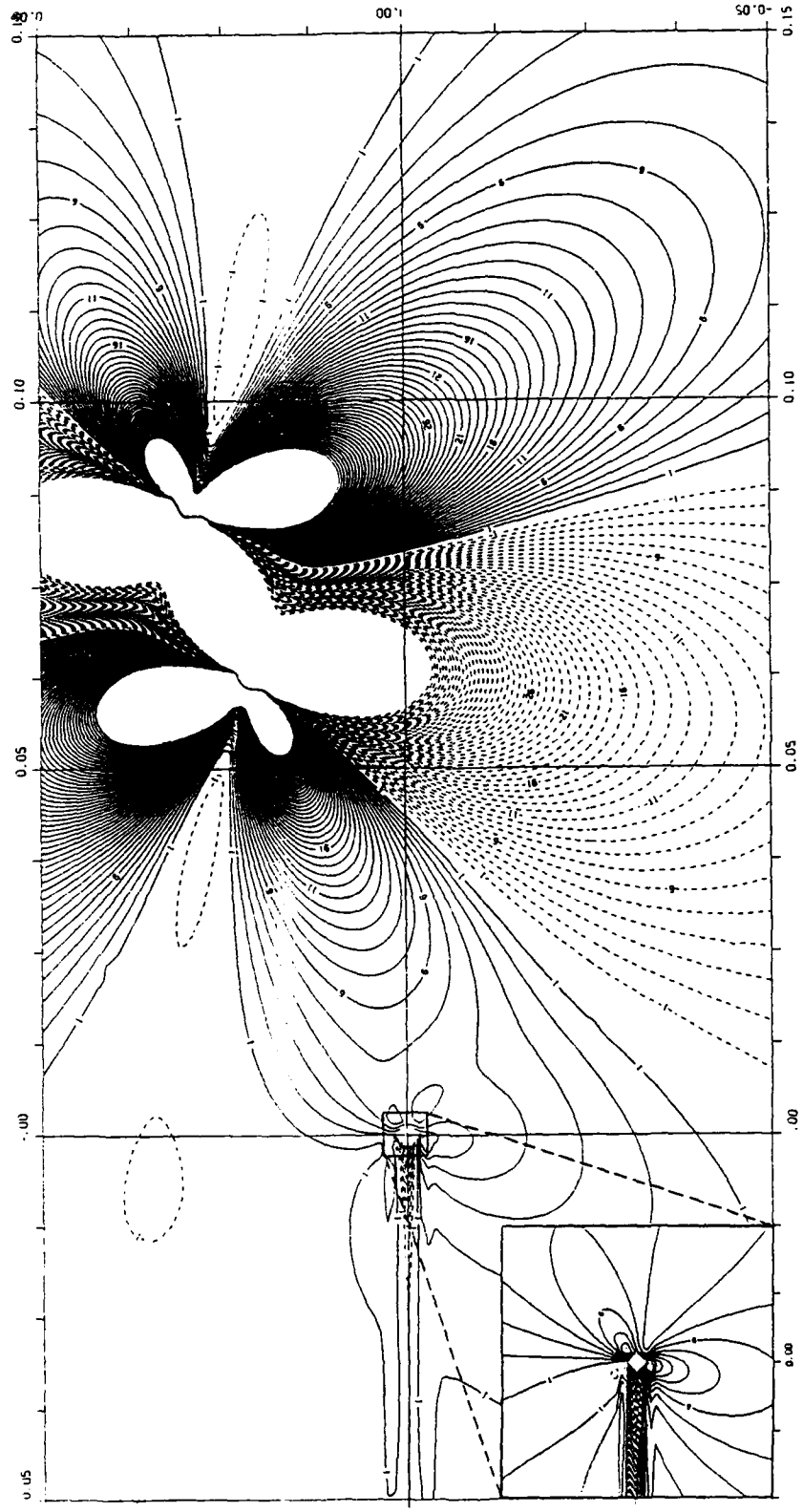


Figure 7.14: Contour plot of σ_{yy} component for the case in Figure 7.12.

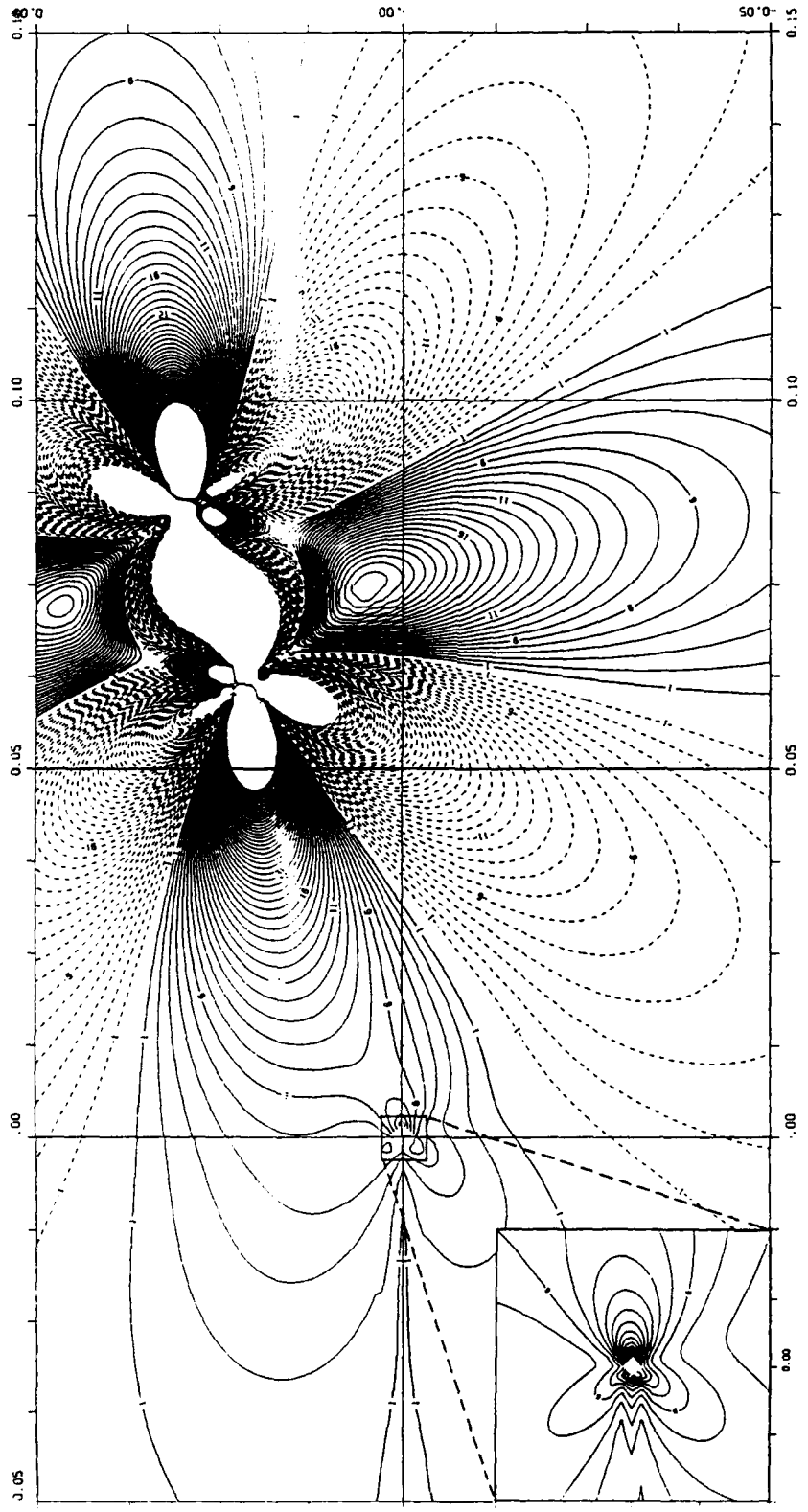


Figure 7.15: Contour plot of τ_{xy} component for the case in Figure 7.12.

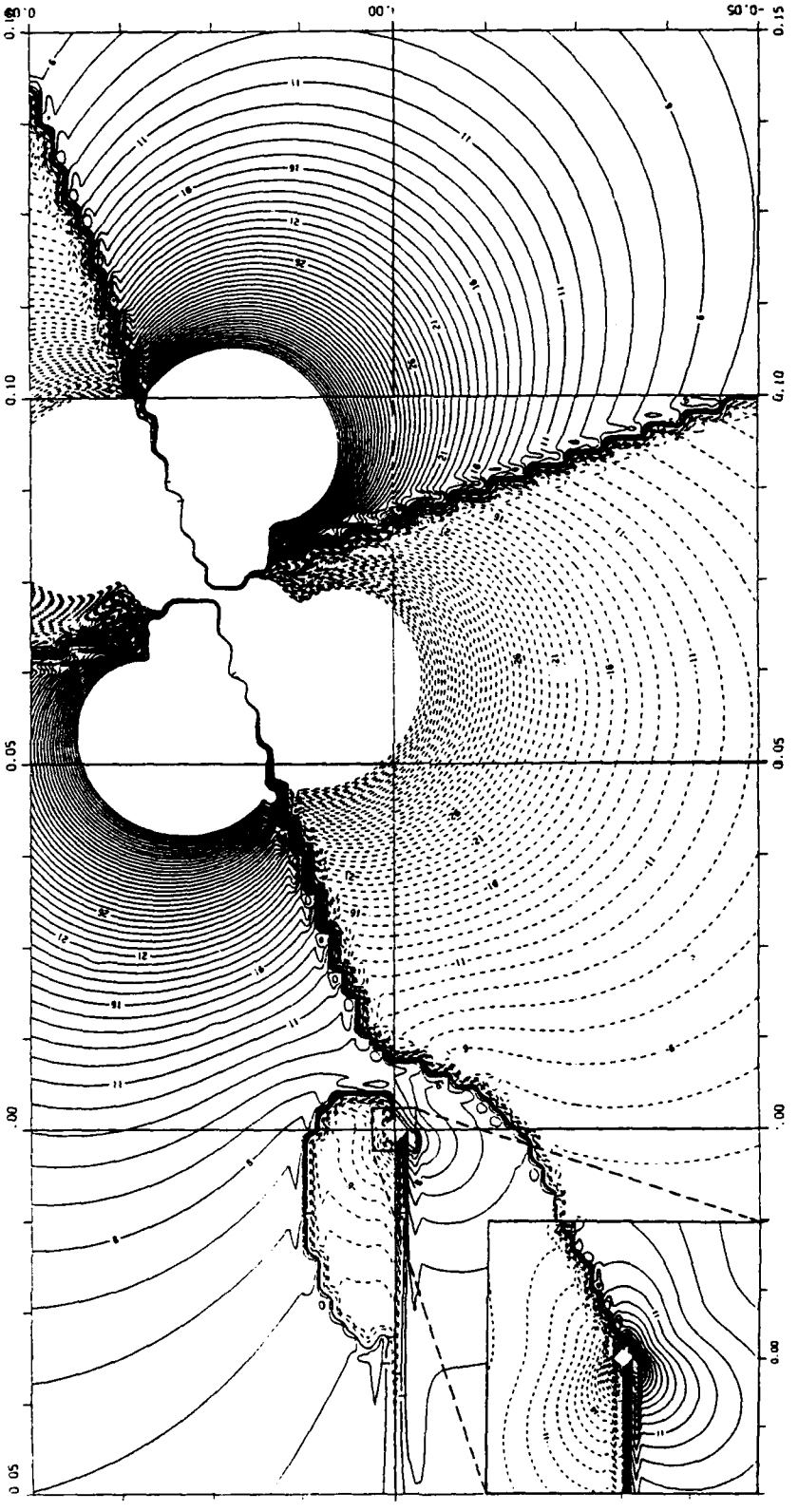


Figure 7.16: Contour plot of σ_{p1} component for the case in Figure 7.12.

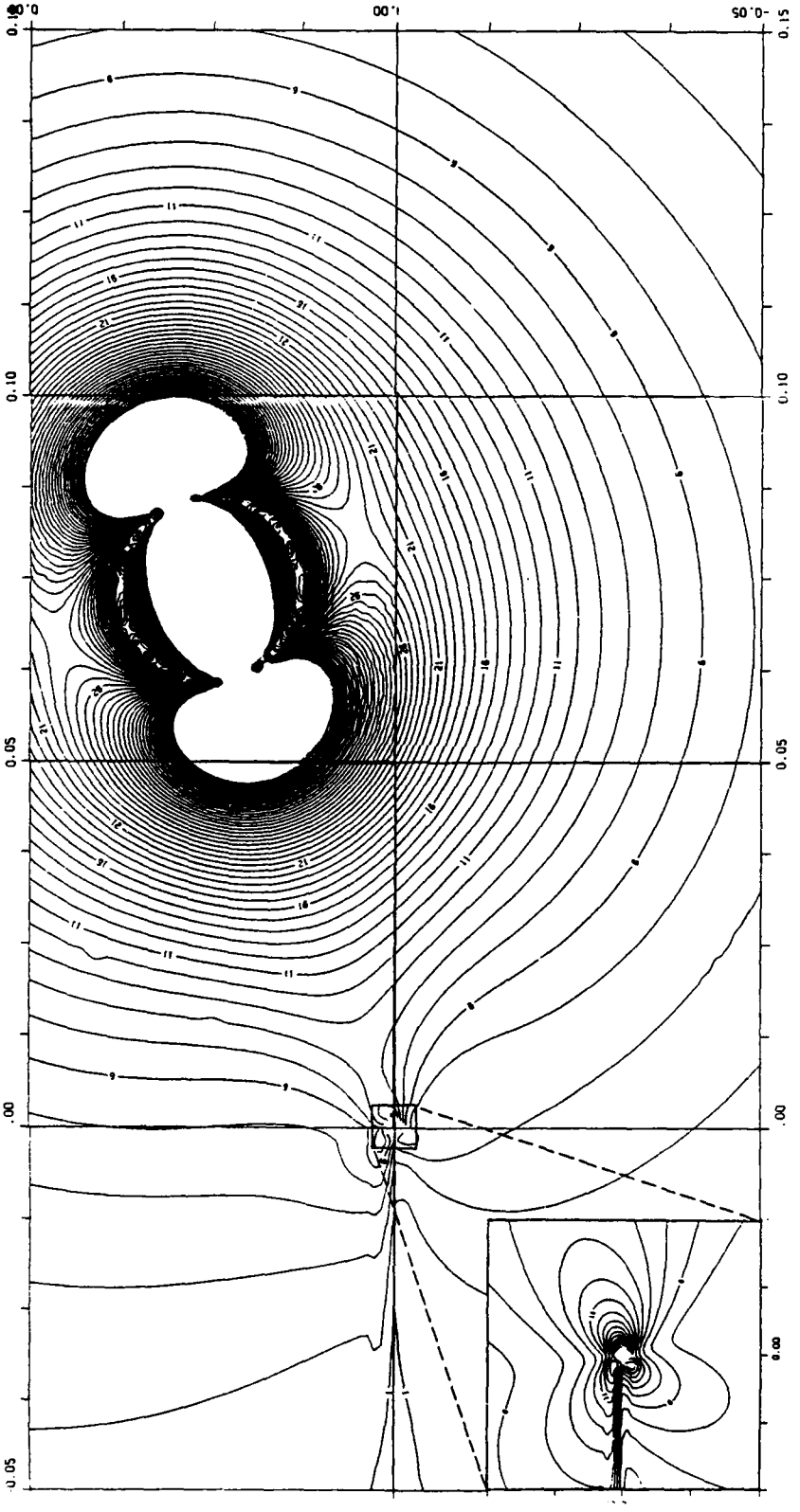


Figure 7.17: Contour plot of τ_{max} component for the case in Figure 7.12



Figure 7.18: Photograph of *dogbone* cracks developed in a cylinder during testing.

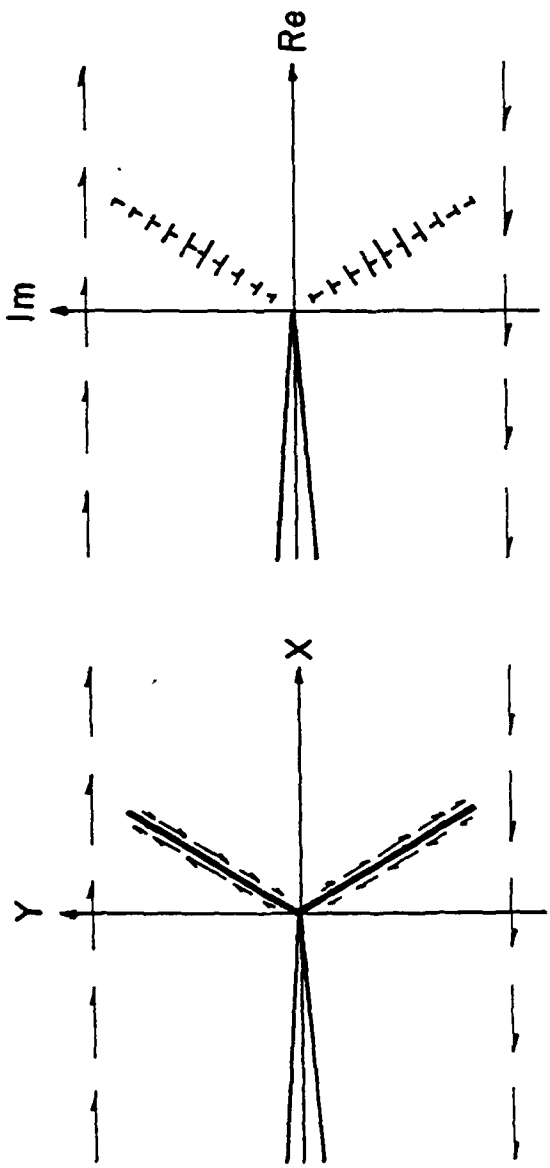


Figure 7.19: Sketch orienting the crack and the distribution of dipole dislocations for the case photographed in Figure 7.18.

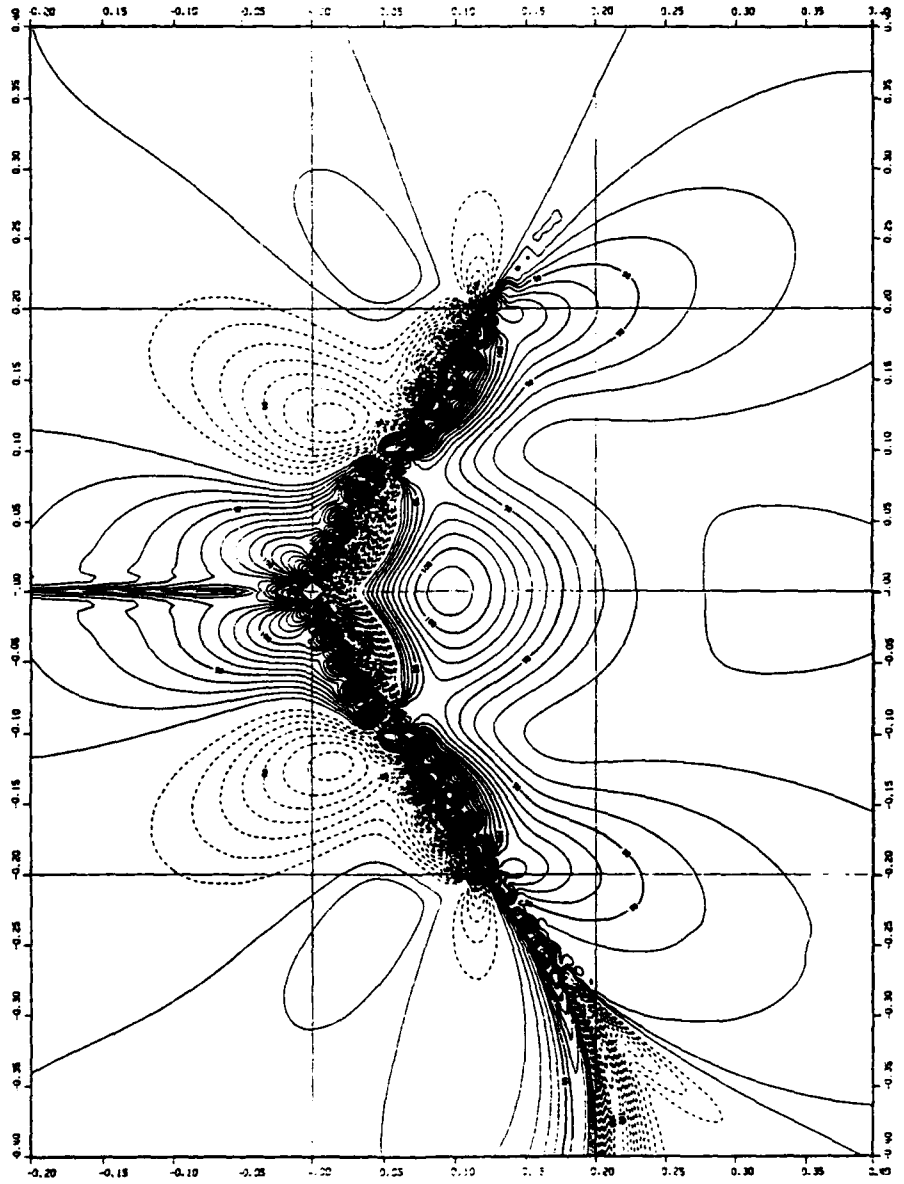


Figure 7.20: Contour plot of τ_{xy} stress component for the dogbone crack case (Figures 7.18 and 7.19).

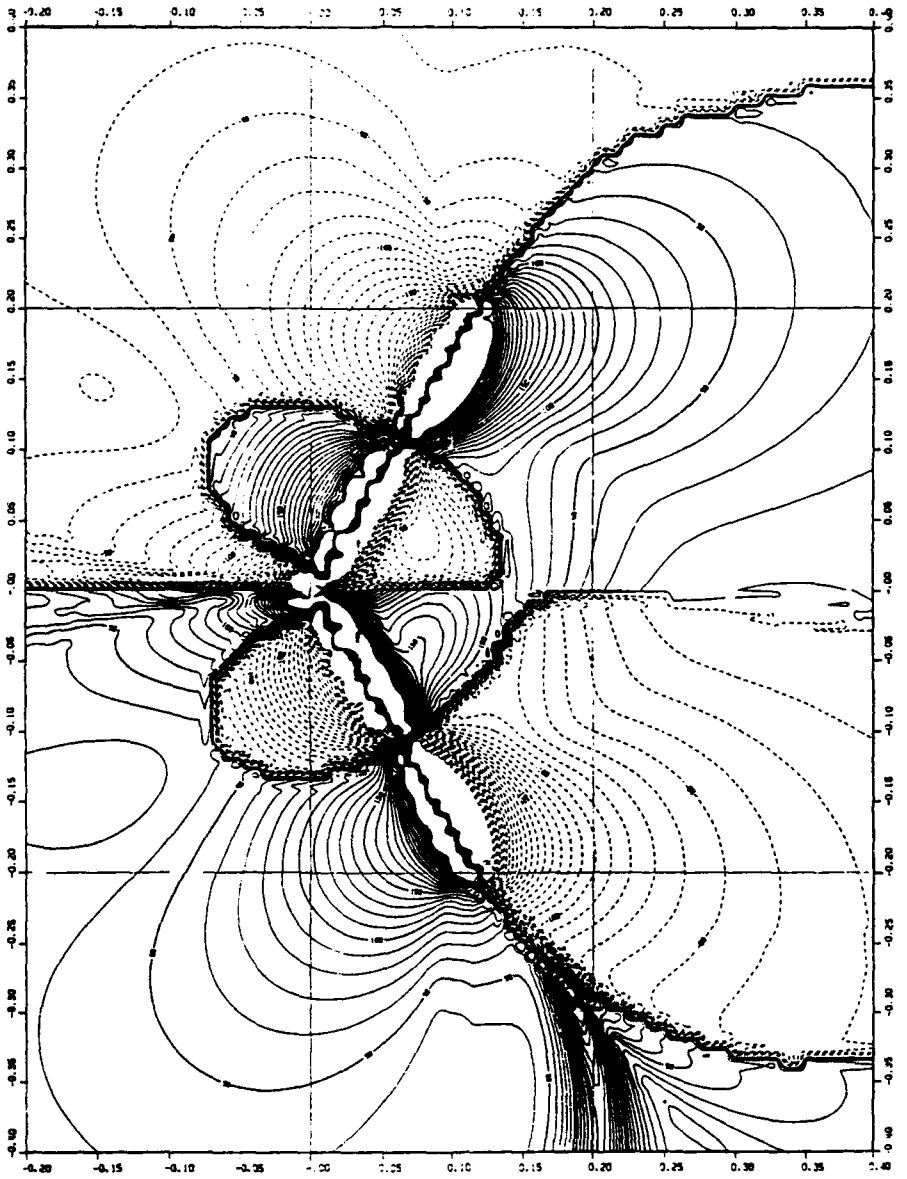


Figure 7.21: Contour plot of σ_{p1} stress component for the dogbone crack case (Figures 7.18 and 7.19).

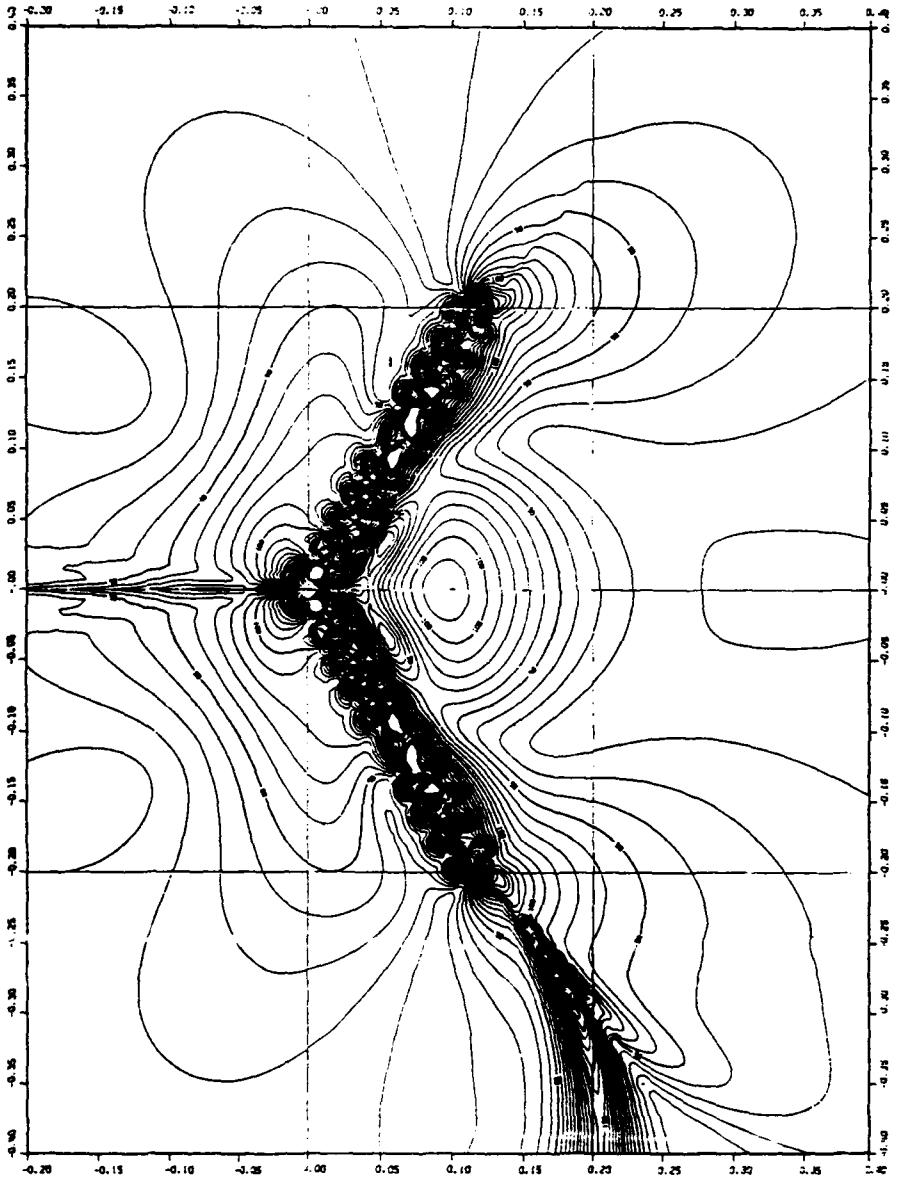


Figure 7.22: Contour plot of τ_{max} stress component for the dogbone crack case (Figures 7.18 and 7.19).

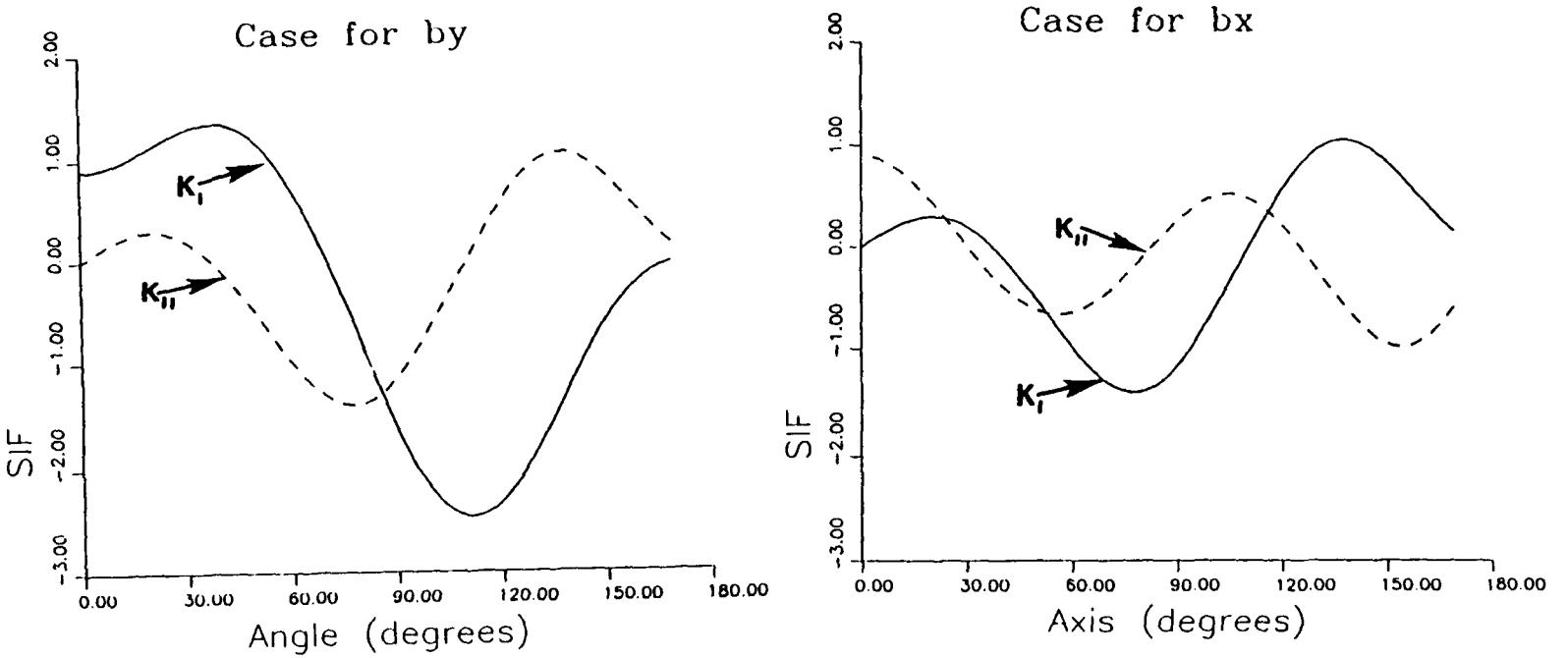
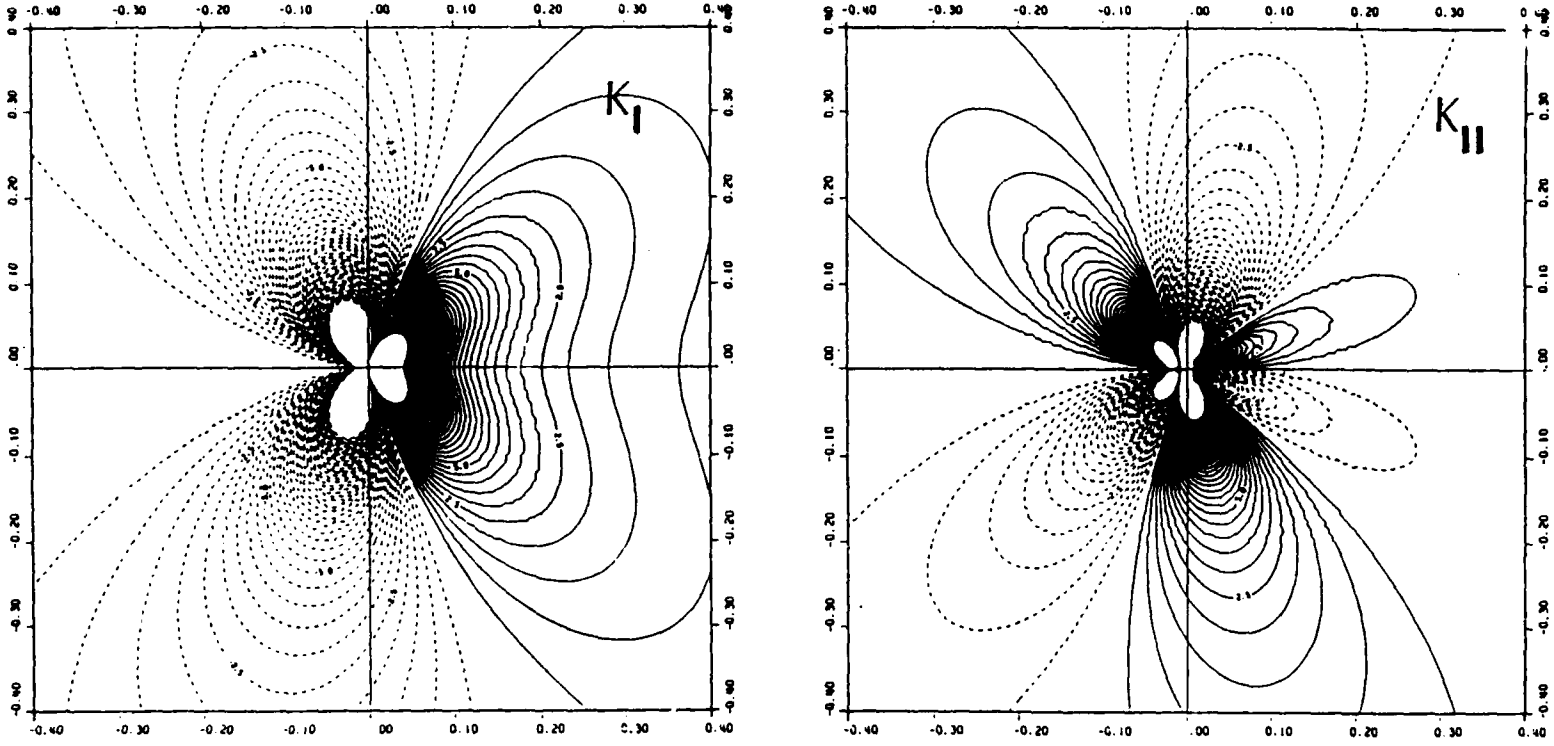
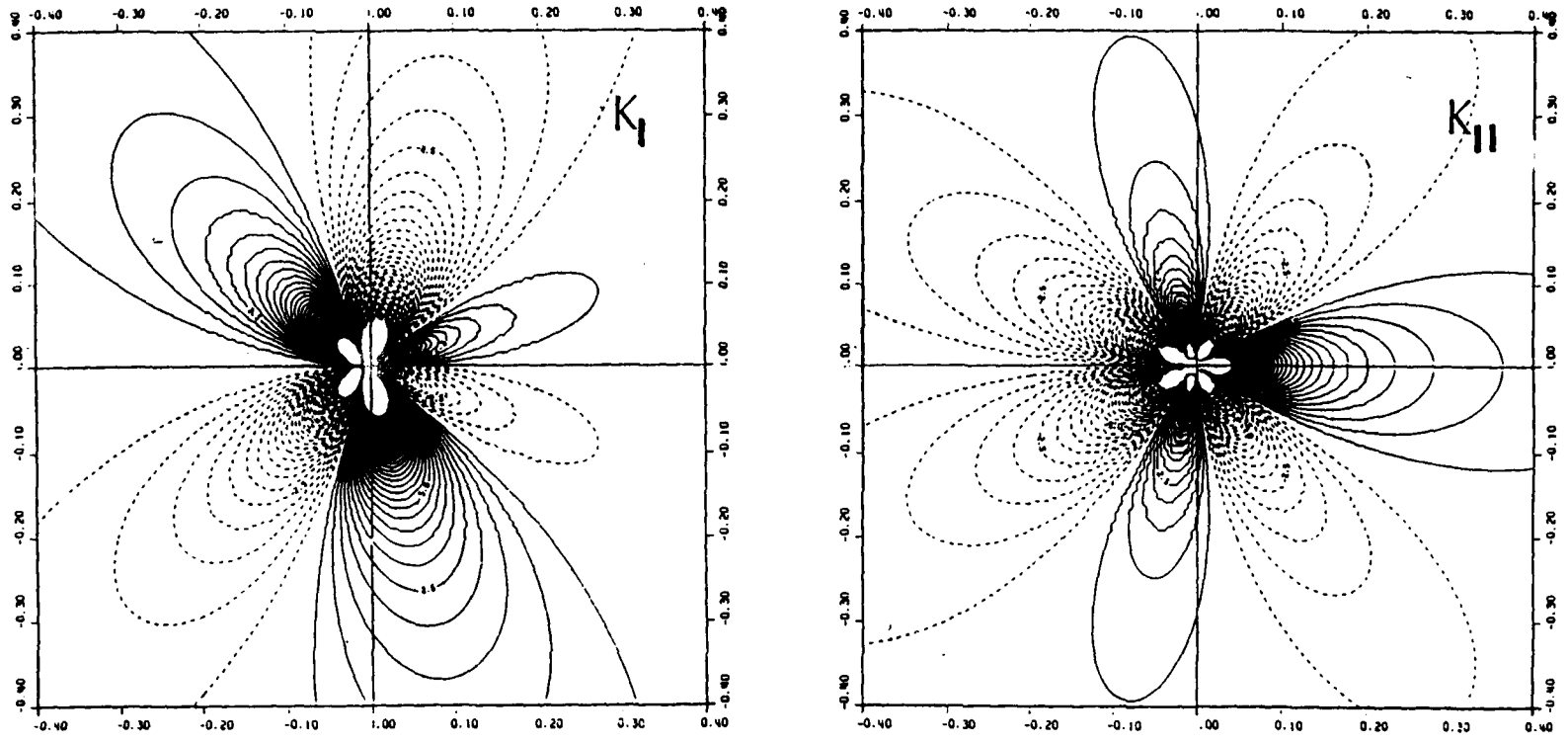


Figure 7.23: Plot of non–dimensionalized SIF (K_I^* and K_{II}^*) vs. the angle locating the unit dipole (the angle $\beta = tan^{-1}(y_0/x_0)$). The case for b_y denotes a dipole with a unit opening, and the case for b_x denotes a unit shear discontinuity.



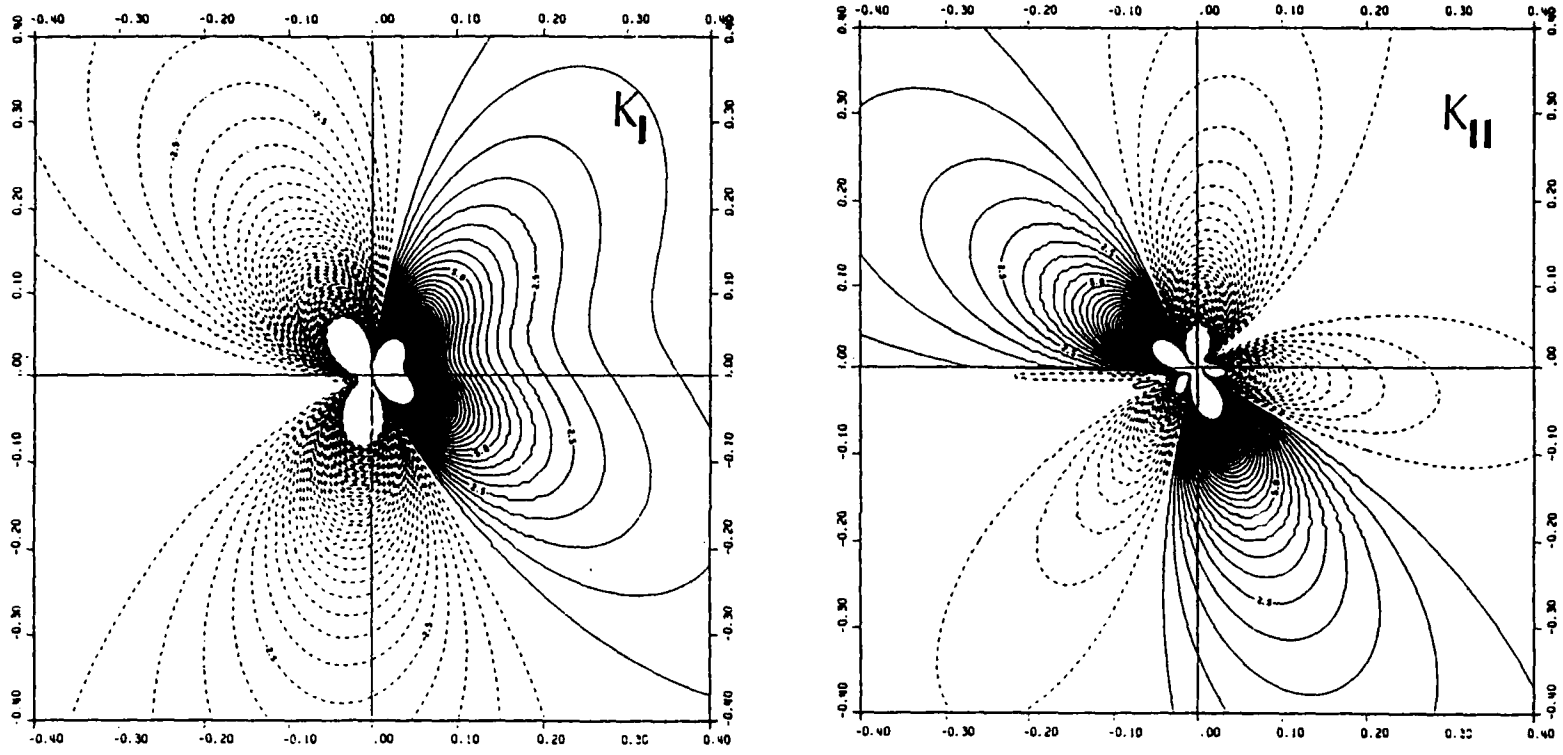
$$G_{sif_I} - iG_{sif_{II}} = K_I^{**} - iK_{II}^{**} = \frac{1}{\sqrt{\pi c}\ |b|\, d\rho}(K_I - iK_{II})$$

Figure 7.24: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit opening dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is parallel with the x–axis ($\Theta = 0^o$).



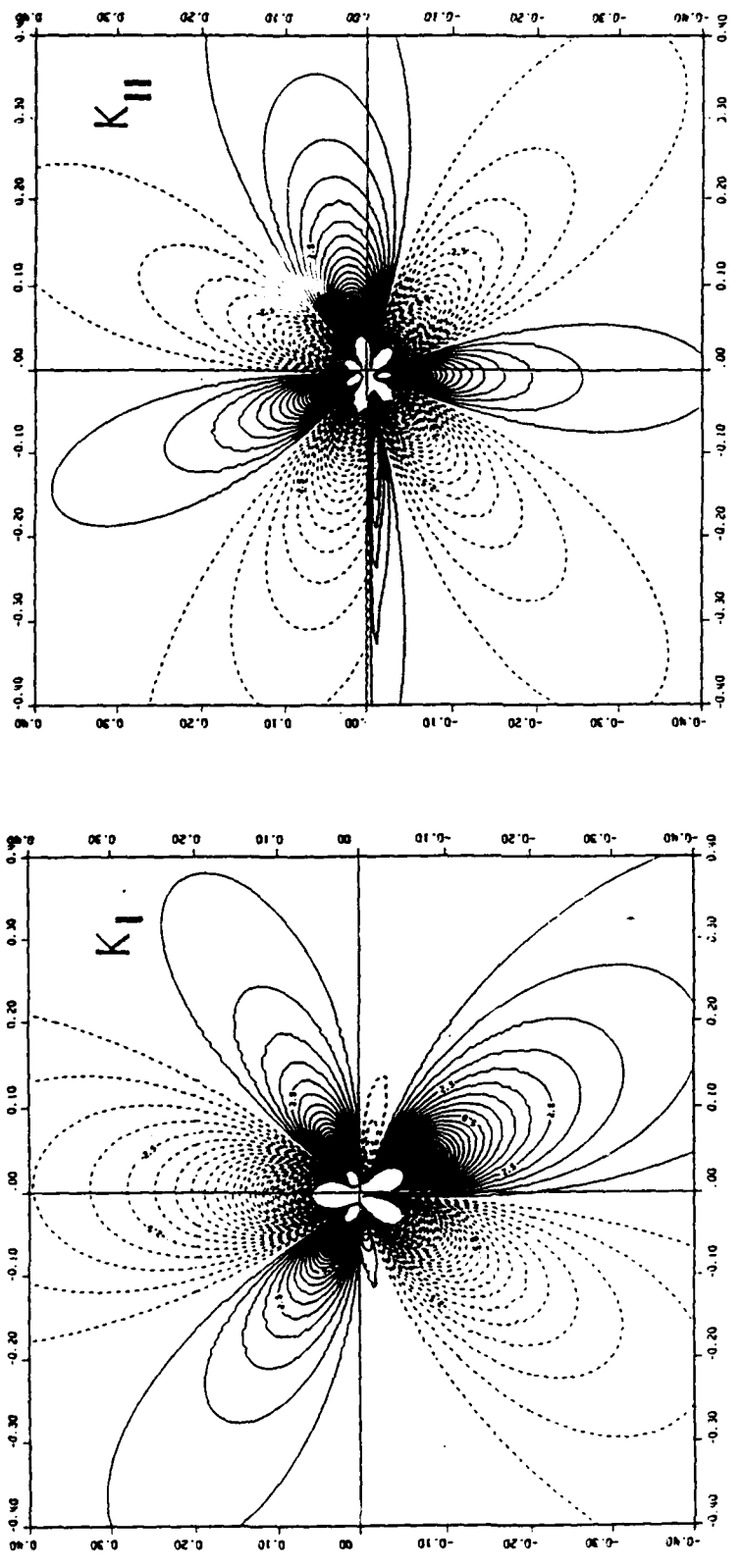
$$G_{\mathrm{sif}_{\mathrm{I}}} - iG_{\mathrm{sif}_{\mathrm{II}}} = K_{\mathrm{I}}^{**} - iK_{\mathrm{II}}^{**} = \frac{1}{\sqrt{\pi c}\ |b|\, d\rho}(K_{\mathrm{I}} - iK_{\mathrm{II}})$$

Figure 7.25: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit shear dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is parallel with the x–axis ($\Theta = 0^{\circ}$).



$$G_{\mathrm{sif}_{\mathrm{I}}} - iG_{\mathrm{sif}_{\mathrm{II}}} = K_{\mathrm{I}}^{**} - iK_{\mathrm{II}}^{**} = \frac{1}{\sqrt{\pi c}\ \ |b|\, d\rho}(K_{\mathrm{I}} - iK_{\mathrm{II}})$$

Figure 7.26: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit opening dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is inclined with the x–axis ($\Theta = +20^{\circ}$).



$$G_{\mathrm{sif}_{\mathrm{I}}} - iG_{\mathrm{sif}_{\mathrm{II}}} = K_{\mathrm{I}}^{**} - iK_{\mathrm{II}}^{**} = \frac{1}{\sqrt{\pi c}\ |b|\, d\rho}(K_{\mathrm{I}} - iK_{\mathrm{II}})$$

Figure 7.27: Plot of nondimensionalized Green's Function for SIF denoted G_{sif} for a unit shear dislocation causing Mode–I and Mode–II effects. The dipole is oriented so that it is inclined with the x–axis ($\Theta = +20°$).

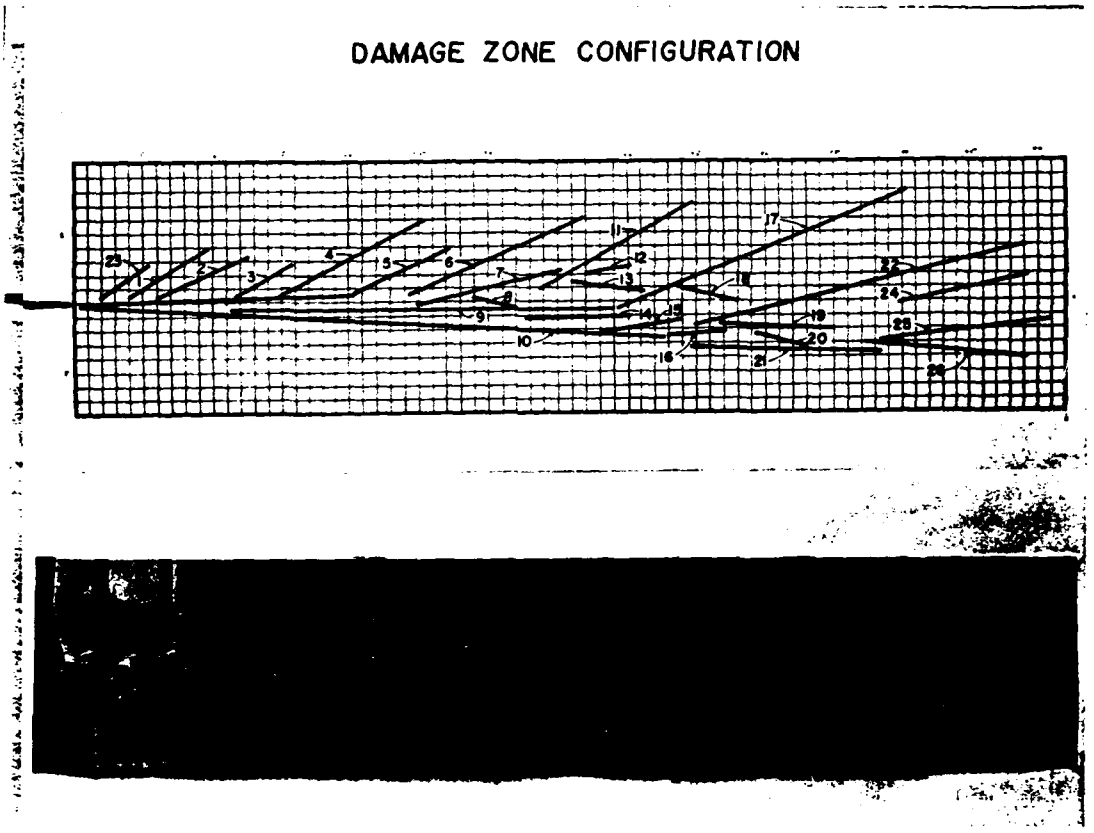


Figure 7.28: Photograph of a crack and preceding damage (below) for Cylinder 5—6 cycle # 262 and the approximation of the damage zone (above) with descrete slip lines.

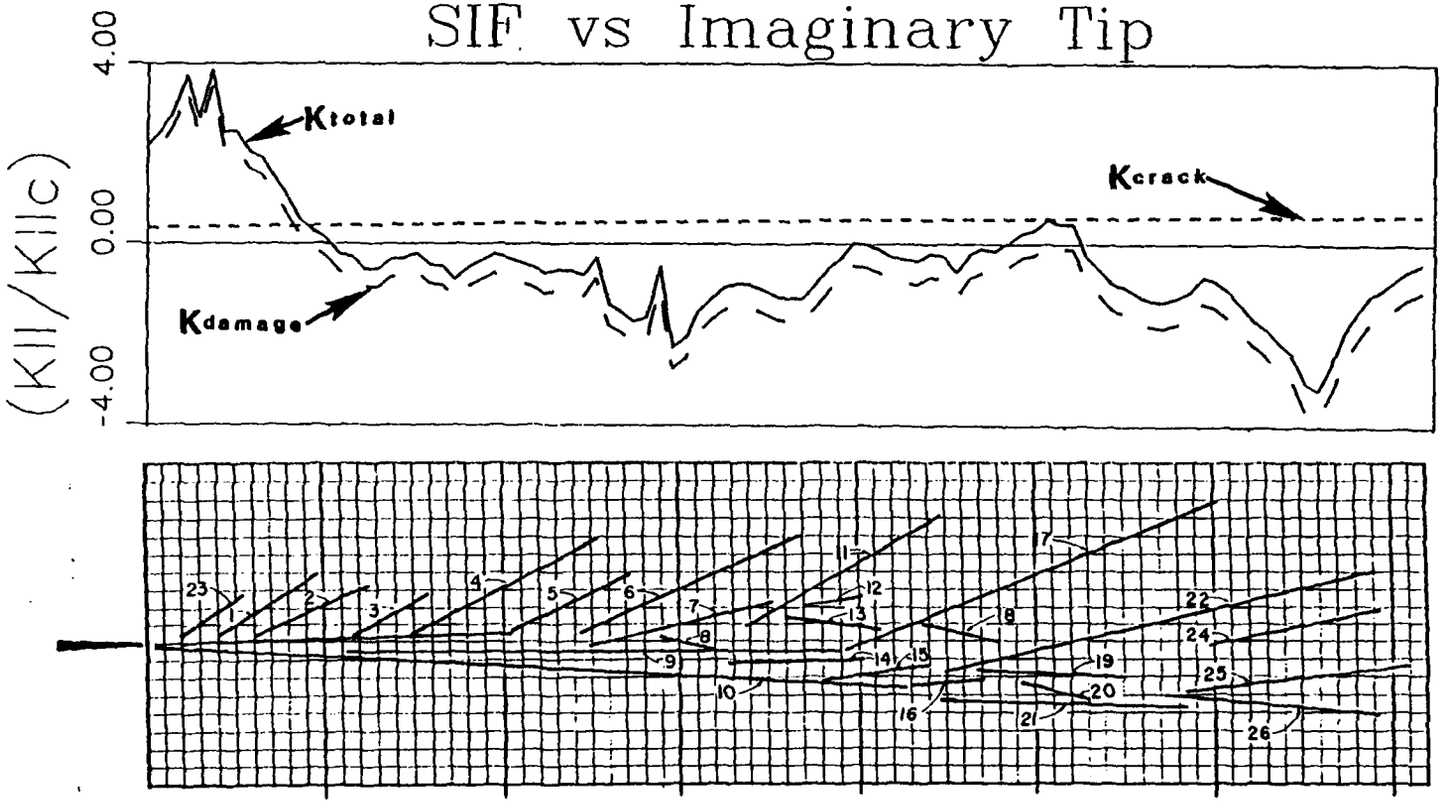


Figure 7.29: Comparison plot of the SIF caused by the remote loading on a main crack (K_{crack}), due to the interaction with the damage (K_{damage}), and their respective total (K_{total}) vs an imaginary crack tip location.

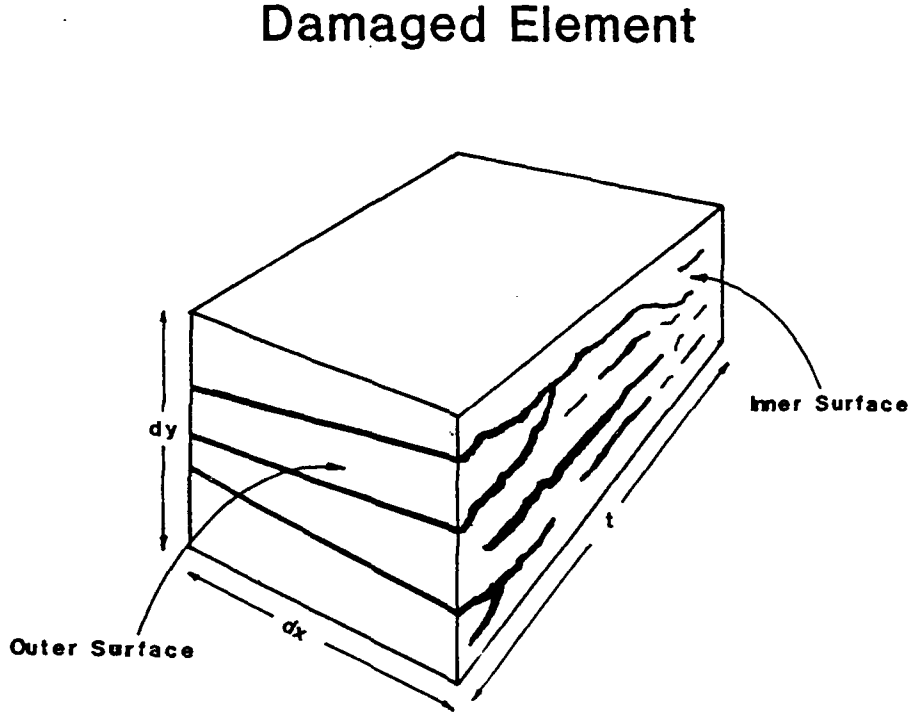
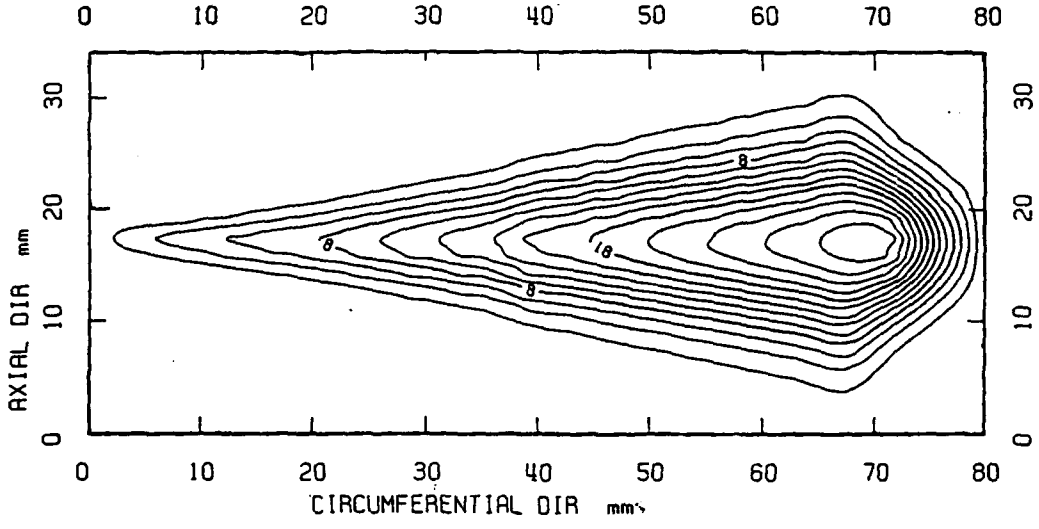


Figure 7.30: An element of damaged material illustrating the internal and external damage.



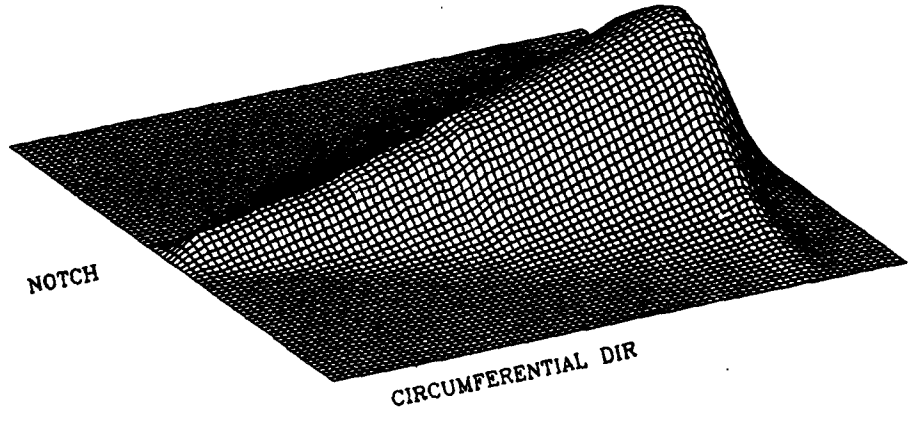


Figure 7.31: Generic damage zone configuration.

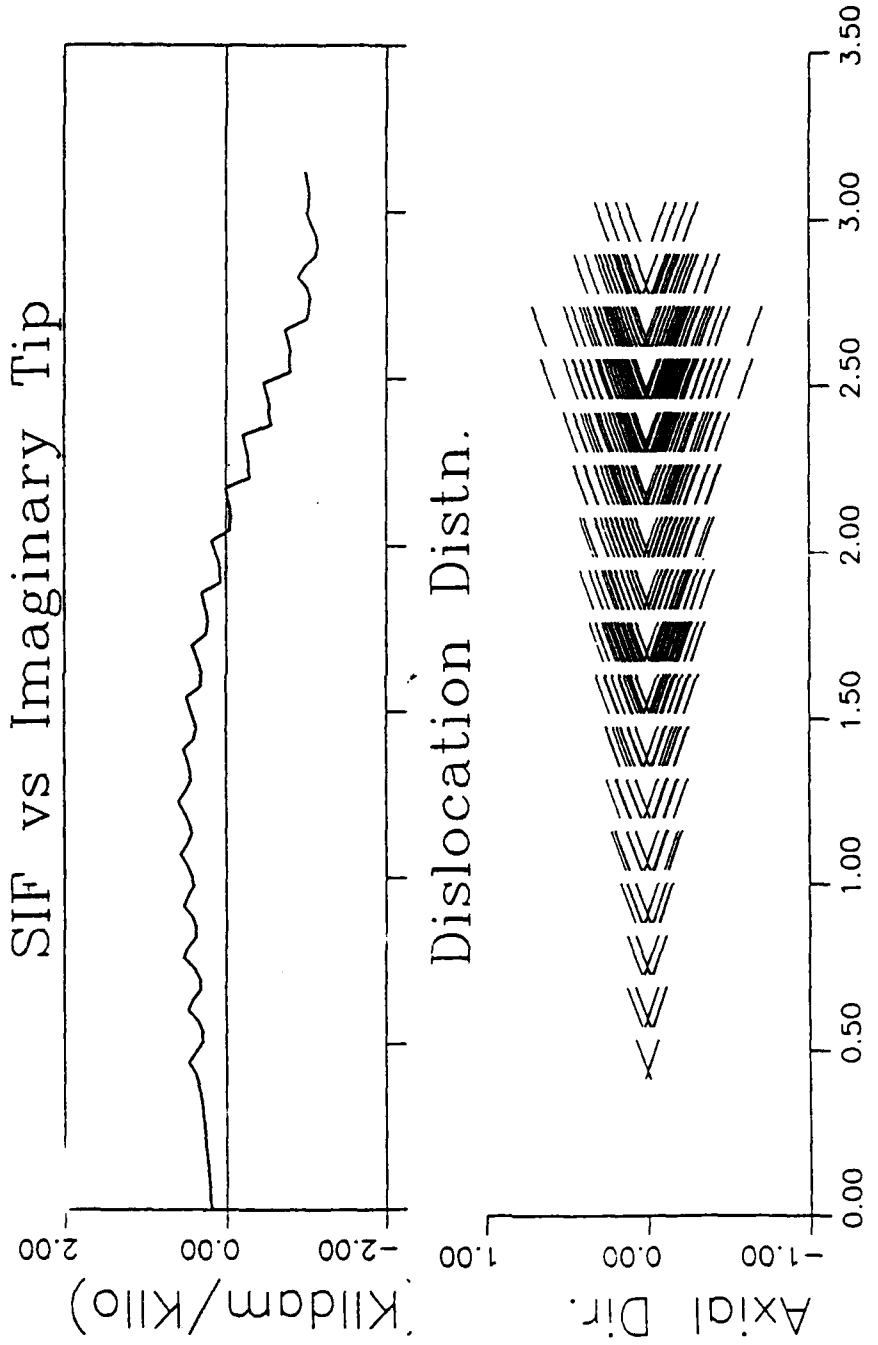


Figure 7.32: **Plot of ΔK_{sdz} (SIF due to damage only) vs crack tip location within the damage zone and a scattergram plot of the elemental *LSD* densities**

Chapter 8

A Constitutive Model

8.1 Introduction

In Chapter 6, the fracture process is treated as a thermodynamic irreversible process. Therein, the global entropy production is described as a bilinear form of generalized fluxes and thermodynamic forces (Equation 6.14). This type of relationship has also proven to be true for other types of irreversible processes such as diffusion and chemical reactions (1,2,3,4).

In order to develop a constitutive model describing the Mode II fracture process in overconsolidated clays using the relationship in Equation 6.14, two tasks must be accomplished:

- The phenomenological relationship between the generalized flux ($\underline{\dot{x}}_c$) and the thermodynamic force ($\underline{X}_1$) must be evaluated.
- Evaluation of the specific energy for the Localized Strip Densification Process (i.e., compute γ). γ is considered to be a material property in the theory and is contained in the thermodynamic force (see Equations 6.15, 6.16, and 6.17).

In this chapter the above two tasks will be addressed using experimentally obtained results from Cylinder 5–6.

8.2 Experimental Results and Observations

Cylinder 5–6 was tested in a manner described in Chapter 2. During the cyclic test, specified torque–rotation cycles were recorded and corresponding photographs of the LSD zone evolution were taken (see Figures 3.1, 3.2, 7.28). From the torque–rotation curves, the total potential energy and irreversible work values vs cycles were plotted. These plots are shown in Figure 8.1.

Other processed results for this test included the damage zone evolution (LSD evolution) performed in Sections 5.3, and 5.4. Results from the Cylinder 5–6 damage evolution reconstruction include a plot of the centroid distance (x_c) vs. cycle number (Figure 5.7) and a plot of the translational resistive moment (R_1) vs cycle number (Figure 5.11).

The data in the plot of centroid distance vs. cycle number in Figure 5.7 was fit by a least squares approach with a third order polynomial. Differentiating the curve fit equation produced a second equation describing the rate of centroid movement (i.e., an equation for $\dot{x}_c$ for any cycle number).

The stress and energy analysis in Chapter 7 showed that the energy release for pure translation of the damage zone amounted to approximately 10% of the total measured energy release rate (see Section 7.8). Thus, the 10% factor was used to reduce the total measured energy release rate in Figure 8.1 to compute the active part of the thermodynamic force (A_1). Application of the above described approach was done in by plotting the potential energy vs. centroid distance and fitting the data with an approximating curve (see Figure 8.2). Again, differentiating the curve fit equation produced another equation representing the total ERR (energy release rate). These results were reduced to 10% of their total value in accordance with the findings in Section 7.8 to reflect the ERR due to translation of the damage zone.

Note that by processing the information as described above allows us to obtain the generalized flux ($\dot{x}_c$), the active part of the thermodynamic force (A_1), and the resistive part of the force (R_1) for any particular cycle if a value for γ is specified. Thus, we are now ready to investigate an appropriate phenomenological relation between the force and flux and to estimate a value for γ, the specific energy for Localized Strip Densification.

8.3 Phenomenological Relationship and Specific Energy Evaluation

In this section we evaluate the phenomenological relation between the generalized flux $\dot{x}_c$ and the thermodynamic force X_1. For convenience we

remind that:

[8.1] $$X_1 = A_1 - \gamma R_1$$

$\dot{x}_c \equiv$ rate of translation of the LSD zone centroid.

At thermodynamic equilibrium all processes stop and we have simultaneously for all irreversible processes:

[8.2] $$\dot{x}_c = 0 \quad X_1 = 0 \qquad \text{at equilibrium.}$$

It is quite natural to assume, at least in the neighborhood of an equilibrium condition (i.e., stable crack and damage propagation), the relation between the force and flux can be approximated by a linear relationship. This type of relation is assumed in many other processes studied in nature. Fourier's Law for heat conduction and Fick's law for diffusion are examples of such processes (1,2,3,4). This type of phenomenological relation is known as the Onsager Principle (2,3).

Initially, the effect of γ on the force/flux relationship was investigated by plotting on a *log–log* scale the data of forces vs. fluxes for various values of γ=constant. In other words, a range of plots illustrating the force–flux data for differing values of γ were produced. These plots for γ ranging between 1.0 kJ/M^3 and 10.0kJ/M^3 is included in Figure 8.3.

In Figure 8.3, it is noticed that their is an apparent insensitivity for a chosen γ in the early portion of the data and that the significant influence only occurs near the end of the record (i.e., the value of γ only significantly affects the last four data points). The last points correspond to cycles greater than or equal to 245. When investigating the *LSD* zone evolution for cycles earlier than 245 cycles (Appendix D) it can be noticed that the relative rate of *LSD* nucleation is significant when compared later portions of the record. Since the thermodynamic model presented within this study did not account for this effect, the application of the translational model to the *LSD* zone to the earlier portion of the record is not applicable.

An Onsager type relation (linear relation) is proposed for the data with cycles greater than or equal to 245 from the Cylinder 5–6 data set. In order to assess the proper value for γ and the phenomological coefficients a regression analysis on the force/flux data for ranging values of γ. The combination which produced the best correlation (e.g., according to Pearson's Correlation Coefficient) were chosen as appropriate parameters to describe the fracture process in this material. The resulting fit is shown in Figure 8.4 and the corresponding equation describing the fracture process is given by:

[8.3] $$\dot{x}_c = \frac{1}{\beta}\Big[A_1 - \gamma R_1 - \alpha\Big]$$

where, for the case of overconsolidated kaolinite clay:

$$\begin{aligned} \gamma &= 1.0 \ (kJ/M^3) \\ \beta &= 45\text{x}10^3 \ (J \text{ cyc}/M) \\ \alpha &= 32 \ (J/M^2) \end{aligned} \qquad [8.4]$$

References

1) Aris, R. (1962) "Vectors, Tensors, and the Basic Equations of Fluid Mechanics," Prentice–Hall, Englewood Cliffs, N.J.

2) Fahien, R. W. (1983). "Fundamentals of Transport Phenomena," McGraw Hill, N.Y.

3) Fung, Y. C. (19). "Foundations of Solid Mechanics", Prentice–Hall, Englewood Cliffs, N.J,

4) Moon, P., Spencer, D. E. (1961). "Field Theory for Engineers," Van Nostrand, Princeton, N.J.

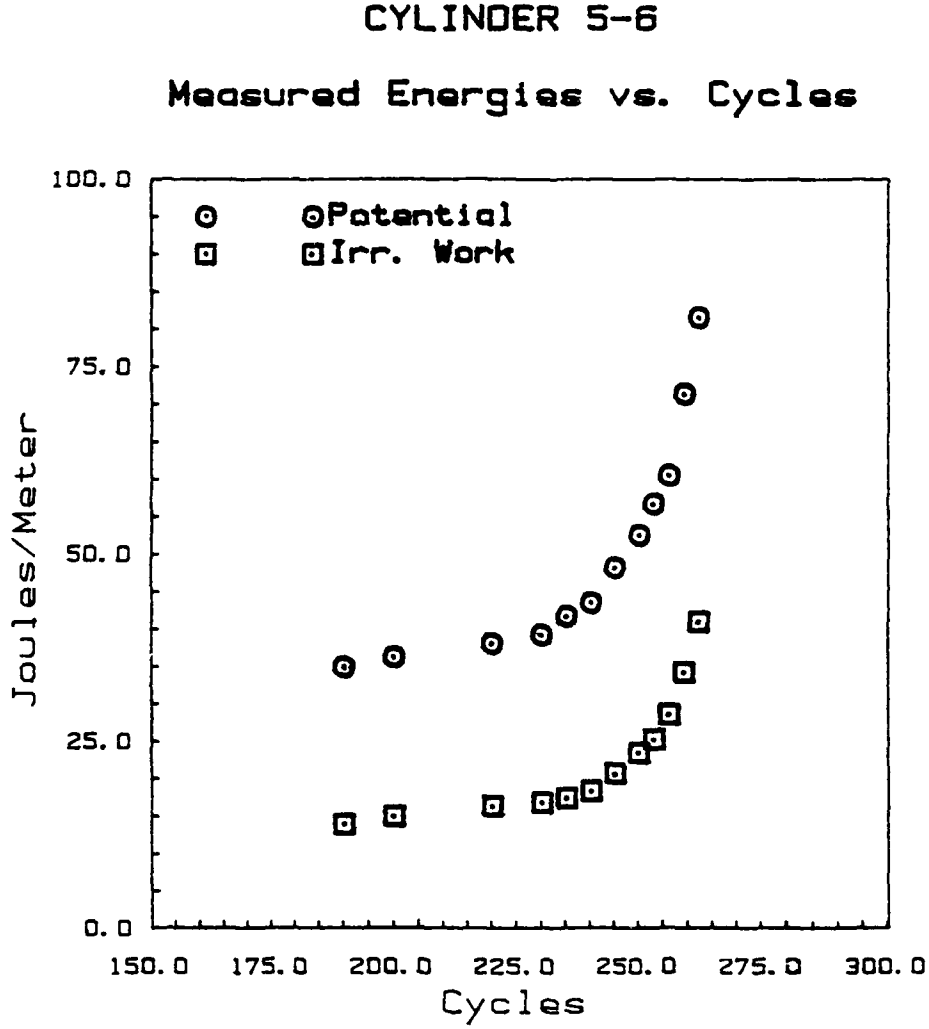


Figure 8.1: Plot of the total measured potential energy and irreversible work vs cycle number for Cylinder 5–6.

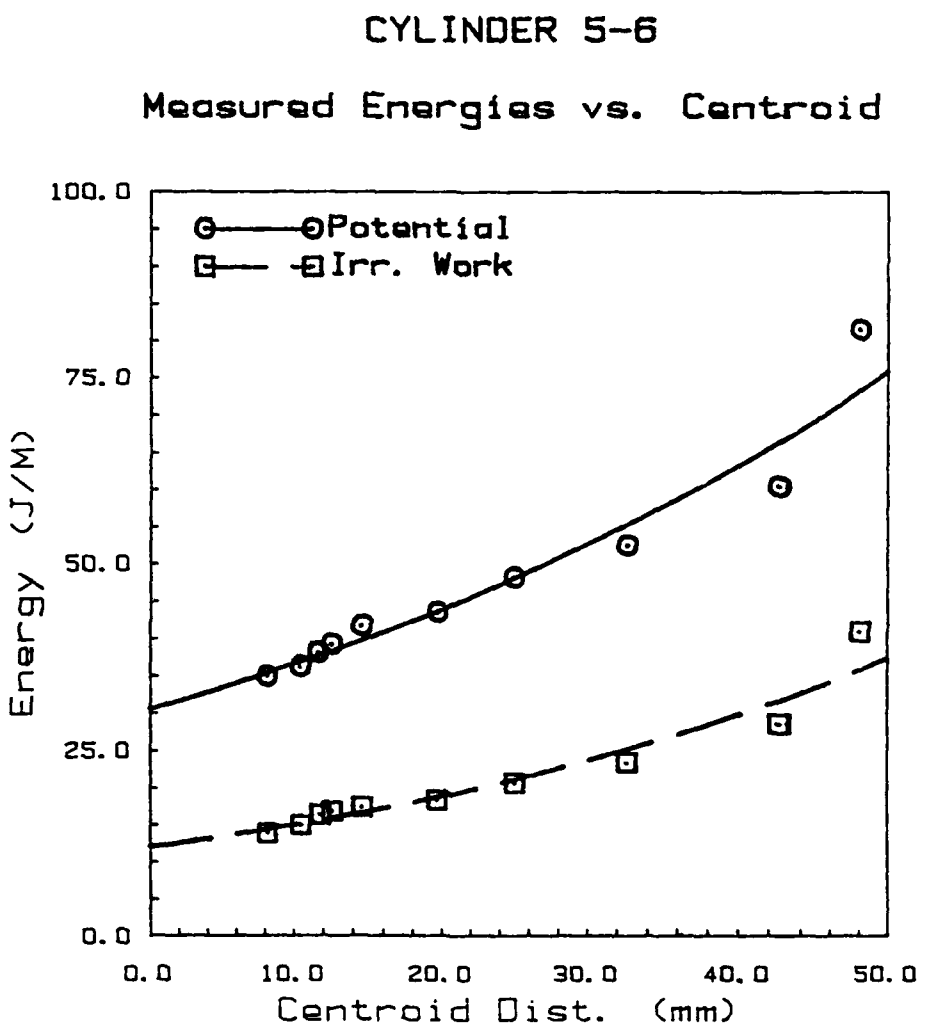


Figure 8.2: Plot of the total measured potential energy and irreversible work vs. centroid distance for Cylinder 5–6.

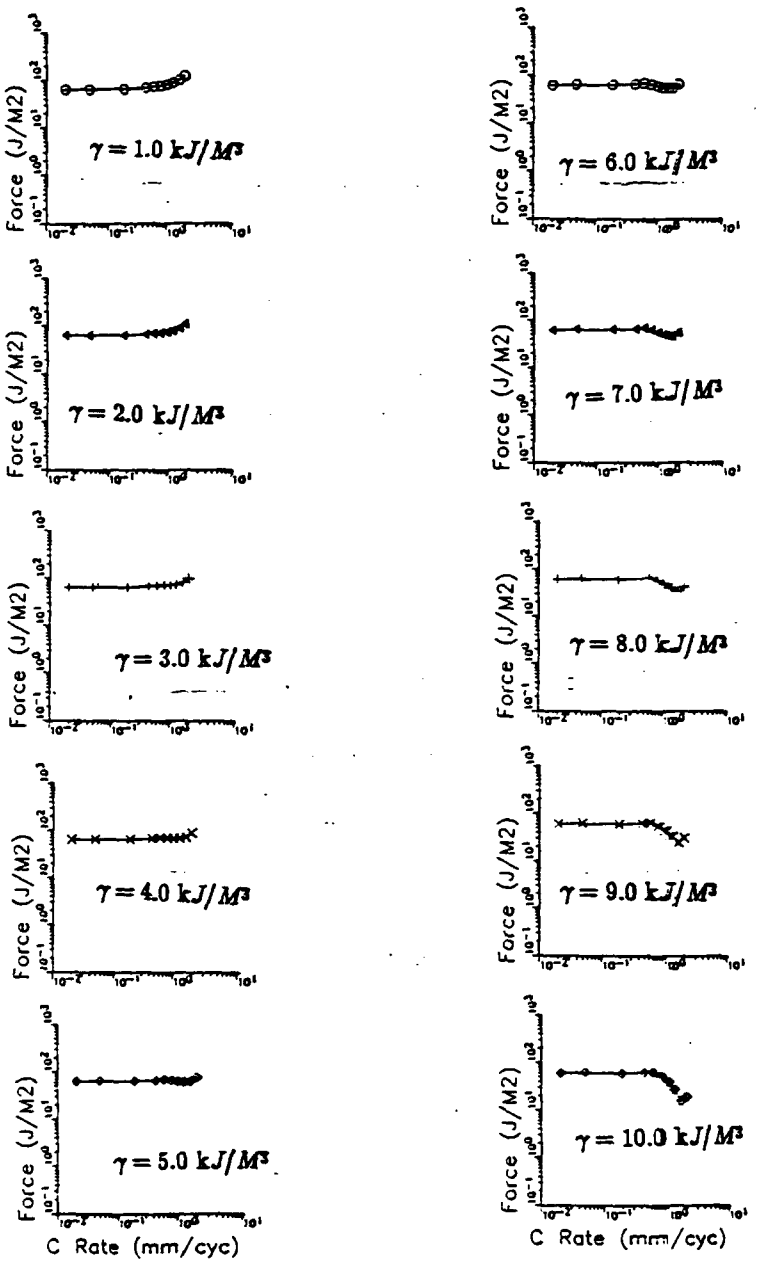


Figure 8.3: Plots of thermodynamic force vs. flux in *log–log* format for various values of γ ranging between 1.0 and 10.0 kJ/M³.

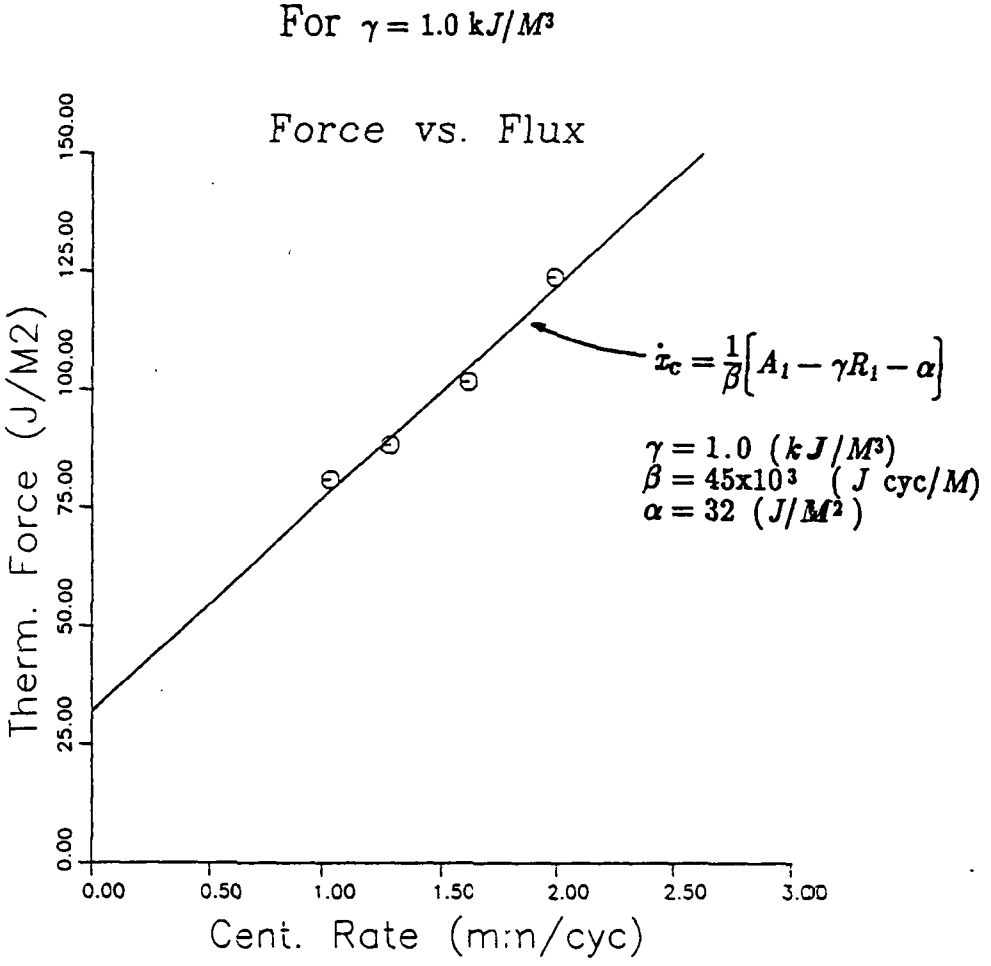


Figure 8.4: Plot of the thermodynamic force vs. flux with γ=1.0 kJ/M^3 for Cylinder 5–6 together with regression fit line.

Chapter 9

Final Discussion

9.1 Summary

The work contained in this dissertation represents a fundamental investigation of the Mode II fracture process in overconsolidated clays. A methodology is presented for characterizing cooperative fracture with an ultimate goal to develop a constitutive model based on the framework of thermodynamics of irreversible processes.

While performing this study, new developments in theorectical aspects, and experimental techniques arose. Some of these developments will be summarized herein.

Chapter 2 presents experimental techniques. Some of the new experimental contributions include using an image analysis system for the quantitative characterization of observed damage in this material. Also, a technique for measuring localized irreversible deformation (slippage) during the testing sequence was developed. It involves photographing and analyzing a grid

system stamped on the cylinder prior to testing.

Chapter 3 presents experimental evidence of stress induced morphological transformations that occur in the soil fabric during fracture. It should be noted that this is the first time damage has been observed in the microstructure of this material and the processes responsible are named Localized Strip Densifications and Network Densifications. In addition, four micromechanisms responsible for macroscopic deformation are identifyed.

Chapter 4 introduces techniques for characterizing the kaolinite cluster distributions (i.e., the soil fabric in it's natural state). Included, is an introduction to the concept of a *representative volume* of material which bridges the gap between material science and applied mechanics. A formal definition is given for the representative volume and a working definition of a representative volume for the case of overconsolidated clays is given.

Chapter 4 also provides new theorectical and numerical techniques in the field of quantitative stereology by presenting a new scheme for reconstructing the spatial distribution of a polydispersed system of particles from information obtained from a planar section. The new reconstruction scheme is a hybrid of Dehoff's and Saltykov's methods except that it is generalized to accomodate a mixture of both prolate and oblate ellipsoids. In addition, the presented scheme allows for an overdefined system of equations improve accuracy over the conventional linear system. Finally, an error analysis technique is presented

and some preliminary results for the spatial distribution of clusters is presented.

Chapter 5 presents an experimentally based scheme for quantitatively characterizing most common of the material transformations (i.e., the Localized Strip Densifications). The technique involves a combination of elaborate specimen sectioning along with an intensive numerical processing scheme to reconstruct contour maps of the damage zone. Further, a technique for reconstructing the evolution of damage is presented and gross (elementary) movements of the entire zone are identified.

Chapter 6 introduces the concept of treating fracture as an irreversible process within the framework of thermodynamics. Within, the conventional set of state parameters is extended to include a damage parameter ρ and an equation of state is written in terms of entropy production. Both the local and global entropy production is expressed as a bilinear form of thermodynamic forces and generalized fluxes. Also, a generalized flux is identified as the centroidal movement of the zone of damage and the active part of the thermodynamic force is compared to the conventional energetic force ($J-$ integral) in fracture for an isothermal condition.

Chapter 7 provides a comprehensive stress and energy analysis for the crack and damage zone configuration. The combined problem of crack, damage and applied load is decomposed into three separate problems. A semi–empirical stress analysis based on a complex potential formulation is introduced.

Contours of effective stress fields are shown for specified cases. Also, the effective stress intensity from each of the separate cases is investigated. Zones of amplification and shielding are identified for a microcrack (dislocation dipole) oriented so as to be parallel to the main crack and for the case of an inclined (20°) microcrack for both Modes I and II. These zones are identified through contour plots of the Green's Function for the stress intensity factor. Also, the combined stress intensity factor caused from both remote loading and a particular (observed) configuration of damage is analyzed and a plot of the total SIF vs imaginary crack tip suggests that the fracture in this material resembles that of a sophisticated Dugdale/Barenblatt model. Further, a generic approach for analyzing the damage zone as distributed damage is introduced and illustrated for a particular case. Finally, an energy release rate analysis of the entire damage zone is performed.

Chapter 8 brings together all of the information from the previous chapters and proposes a constitutive model for describing the fracture in this material. The model is based on a bilinear form of a generalized flux (centroidal movement of the damage zone) and a thermodynamic force. The model presented consists of a phenomological relation between flux and force and the evaluation of a material property γ (the specific energy for cluster densification).

9.2 Conclusions

Many conclusions can be extracted from the work reported in this thesis. Most of these have been commented on in respective sections of their chapters. These will not be reiterated in this section and only those that are considered most significant will be discussed.

The first conclusion that can be extracted from this work is the micromechanisms responsible for macroscopic deformation which are evidenced in Chapter 3 and explicitly listed therein.

Second is that the overall methodology presented, provides a constitutive law which derives its basis from the thermodynamic studies of transport phenomena (heat conduction, diffusion, etc.). Thus, a bilinear form describing the fracture process contains sound scientific reasoning and is not simply an empirical formula.

Finally, it is my belief that further studies of this nature would greatly benefit if a specimen of larger diameter is used. The basis for this finding can be found explicitly in three different chapters. First of all, the minimum representative volume size determined in Chapter 4 indicates that statistical convergence of specimen response may not occur unless the specimen size is at least an order of magnitude larger that the RV size. Secondly, close inspection of Figure 3.5 b shows a higher frequency of LSD's emanating to the outer

surface and relatively few along the inner surface. This observation indicates that the curvature affects the nucleation and growth of damage. Thirdly, Figure 8.3 shows a relative insensitivity to the value for γ except at the latter data points. This indicates that the early portion (less than 245 cycles) majority of the test sequence can be considered an *initiation stage* and the applicability of this model (constitutive model) only becomes reasonable after 245 cycles. A larger diameter specimen would provide more data within this realm.

Appendix A

Program Listings

```
                                                                        Page   1
                                                                        06-24-86
                                                                        12:21:45
D Line# 1      7                    IBM Personal Computer FORTRAN Compiler V2.00
      1         PROGRAM PROBLY
      2 C*************************************************************************
      3 C***** THIS PROGRAM IS DESIGNED TO COMPUTE THE CUMULATIVE    *******
      4 C***** PROBABILITY USING A GAUSSIAN DENSITY DISTRIBUTION FOR *******
      5 C***** A SPECIFIED INTERVAL .                                *******
      6 C*************************************************************************
      7 C
      8 C     Input Required Variables
      9 C
     10       WRITE(*,*)'  --- INPUT PARAMETERS ---'
     11  0005 WRITE(*,*)'  MEAN,    SDEV,    SUM'
     12       READ(*,*,ERR=0005)BARX,SDEV,SUM
     13  0010 WRITE(*,*)'   GIVE LOWER AND UPPER LIMIT'
     14       READ(*,*,ERR=0010)BL,UL
     15 C
     16 C     Set Integration variable
     17 C
     18       NINT=1000
     19       WDTH=(BL-LL)/FLOAT(NINT)
     20       XC=BL
     21       COEFF=SUM/(SDEV*2.5066)
     22       SUM1=0.0
     23 C
     24       DO 0100 I=1,NINT
1    25
1    26       EX=-0.5*(((XC-BARX)/SDEV)**2)
1    27       HGHT=EXP(EX)
1    28       HGHT=COEFF*HGHT
1    29       SUM1=SUM1+(HGHT*WDTH)
1    30       XC=XC+WDTH
1    31  0100 CONTINUE
     32 C
     33 C     Write final results
     34 C
     35       WRITE(*,0020)BARX,SDEV,BL,UL,SUM1
     36  0020 FORMAT(//,' -----  SUMMARY OF INPUT AND RESULTS ----',/,
     37      1 '      MEAN=',G10.4,'       SDEV=',G10.4,/,
     38      2 ' LOWER LIM=',G10.4,'  UPPER LIM=',G10.4,/,
     39      3 '    CUMULATIVE PROBILITY =',G10.4)
     40       END

Name   Type         Offset P Class

BARX   REAL            2
BL     REAL           14
COEFF  REAL           38
EX     REAL           54
EXP                       INTRINSIC
FLOAT                     INTRINSIC
HGHT   REAL           58
I      INTEGER*4      46
LL     INTEGER*4      10
NINT   INTEGER*4      22
SDEV   REAL            6
SUM    REAL           10
SUM1   REAL           42

UL     REAL           18
WDTH   REAL           26
XC     REAL           34
```

D Line# 1 7 IBM Personal Computer FORTRAN Compiler V2.00
41

Name	Type	Size	Class
PROBLY			PROGRAM

Pass One No Errors Detected
41 Source Lines

```
D Line#  1      7                                  Microsoft FORTRAN77 V3.20 02/84
      1 $DEBUG
      2         PROGRAM CYCIMP
      3 C*************************************************************************
      4 C***** THE PURPOSE OF THIS PROGRAM IS TO RECIEVE INPUT FROM THE HP *
      5 C***** PLOTTER INFORMATION AND STORE THE INFO IN AN ARRAY FOR PLOT-*
      6 C***** TING AND OTHER REPROCESSING.                                 *
      7 C*************************************************************************
      8         DIMENSION X(1000),Y(1000)
      9         CHARACTER*10 DATFIL
     10         CHARACTER*40 TITLE
     11         XINC=0.0
     12         YINC=0.0
     13  0010 WRITE(*,*)'    --- GIVE FILE NAME --- '
     14         WRITE(*,*)' XXXXXXXXXX'
     15         READ(*,0020,ERR=0010)DATFIL
     16  0020 FORMAT(A10)
     17         OPEN(10,FILE=DATFIL,STATUS='NEW')
     18  0030 WRITE(*,*)'    --- GIVE DATA TITLE ---'
     19         WRITE(*,*)' XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
     20         READ(*,0040,ERR=0030)TITLE
     21  0040 FORMAT(A40)
     22  0035 WRITE(*,*)' GIVE NUMBER OF POINTS'
     23         READ(*,*,ERR=0035)NPTS
     24  0025 WRITE(*,*)'      IF AUTO INCREMENTING  X OR Y  DATA '
     25         WRITE(*,*)'        GIVE  XINC AND YINC RESPECTIVELY'
     26         READ(*,*,ERR=0025)DUMX,DUMY
     27         IF(DUMX.NE.0.0)XINC=DUMX
     28         IF(DUMY.NE.0.0)YINC=DUMY
     29         DUMX=0.0
     30         DUMY=0.0
     31 C-------------------------------------------------------------------------
     32 C       TERMINAL INPUT LOOP
     33         DO 0100 I1=1,NPTS
1    34  0105 WRITE(*,0110)I1,DUMX,DUMY
1    35  0110 FORMAT(///
1    36      1 ' ------------------ DEFAULT VALUES ----------------------',/
1    37      2 '     PT. NO=',I5,'    X =',F12.5,'   Y=',F12.5,//
1    38      3 '       GIVE  X  OR  Y  IF NOT SATISFACTORY',//
1    39      4 '          -->-999. TO CHANGE INCREMENT VALUES',/
1    40      5 ' ------------------------------------------------------',/)
1    41         DMX=0.0
1    42         DMY=0.0
1    43         READ(*,*,ERR=0105)DMX,DMY
1    44 C
1    45         IF(DMX.EQ.-999.)THEN
1    46  0115     WRITE(*,*)' GIVE NEW XINC AND YINC'
1    47           READ(*,*,ERR=0115)XINC,YINC
1    48           GO TO 0105
1    49         ENDIF
1    50 C
1    51         IF(DMX.NE.0.0)DUMX=DMX
1    52         IF(DMY.NE.0.0)DUMY=DMY
1    53         X(I1)=DUMX
1    54         Y(I1)=DUMY
1    55         DUMX=DUMX+XINC
1    56         DUMY=DUMY+YINC

1    57  0100 CONTINUE
     58 C-------------------------------------------------------------------------
     59 C
```

```
D Line# 1      -                                    Microsoft FORTRAN77 V3.20 02/84
     60  0120 WRITE(*,*)'  CONTINUE --> 0   LIST INPUT --1'
     61       IDUM=0
     62       READ(*,*,ERR=0120)IDUM
     63       IF(IDUM.GT.0)THEN
     64  0133    WRITE(*,*)' |--- PT NO --|---X (value) ---|---Y (value)---|'
     65          DO 0125 I1=1,NPTS
1    66  0125    WRITE(*,0130)I1,X(I1),Y(I1)
     67  0130    FORMAT('  |   ',I5,'      |   ',F12.5,'  |  ',F12.5,'  |')
     68  0127    WRITE(*,*)'  CHANGE INPUT GIVE POINT NO'
     69          IDUM=0
     70          READ(*,*,ERR=0127)IDUM
     71          IF(IDUM.GT.0)THEN
     72  0129  WRITE(*,*)' GIVE    X (value), Y (value)'
     73        READ(*,*,ERR=0129)X(IDUM),Y(IDUM)
     74              GO TO 0133
     75          ENDIF
     76       ENDIF
     77 C-------------------------------------------------------------------
     78 C
     79 C     GENERATE INPUT
     80  0140 WRITE(*,*)'  GENERATE OTHER HALF -->1'
     81       IDUM=0
     82       READ(*,*,ERR=0140)IDUM
     83       IF(IDUM.GT.0)THEN
     84          NPM1=NPTS-1
     85          WRITE(*,*)' NPM1=',NPM1
     86          DO 0150 I1=1,NPM1
1    87          X(NPTS+I1)=X(NPTS)-X(I1+1)
1    88          Y(NPTS+I1)=-1.0*Y(I1+1)
1    89  0150    CONTINUE
     90          NPTS=NPTS*2-1
     91          WRITE(*,*)' NPTS=',NPTS
     92       ENDIF
     93       WRITE(*,*)'    WRITING FILE'
     94       DO 0200 I1=1,NPTS
1    95  0200 WRITE(10,*)X(I1),Y(I1)
     96       STP=-999.
     97       WRITE(10,*)STP,STP
     98       WRITE(10,0040)TITLE
     99       CLOSE(10)
    100       END

Name    Type         Offset P Class

DATFIL  CHAR*10        8024
DMX     REAL           8490
DMY     REAL           8494
DUMX    REAL           8090
DUMY    REAL           8094
I1      INTEGER*4      8098
IDUM    INTEGER*4      8498
NPM1    INTEGER*4      8556
NPTS    INTEGER*4      8086
STP     REAL           8568
TITLE   CHAR*40        8040
X       REAL             16

XINC    REAL           8016
Y       REAL           4016
YINC    REAL           8020
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

Name	Type	Size	Class
CYCIMP			PROGRAM

Pass One No Errors Detected
100 Source Lines

```
D Line# 1      7                          Microsoft FORTRAN77 V3.20 02/84
      1 $DEBUG
      2       PROGRAM CONTOUR
      3 C*************************************************************************
      4 C***** This program is designed to create a series of files for   ***
      5 C***** plotting a an active zone contour map.  It assumes the     ***
      6 C***** phenomina can be represented with a normal distribution.   ***
      7 C*************************************************************************
      8
      9       COMMON PT(100,2),YBAR(100),YDEV(100),YCUT(100,2),YMAX(100),
     10      1 XCRD(100)
     11       CHARACTER*10 FILENM,FINP
     12 C
     13 C     Set origin (crack tip) values
     14 C
     15       IN=0
     16  0200 WRITE(*,*)' INPUT   FILE ->1  TERMINAL ->0'
     17       READ(*,*,ERR=0200)IN
     18       IF(IN.NE.0)THEN
     19  0205  WRITE(*,*)' GIVE INPUT FILE NAME'
     20        READ(*,0055,ERR=0205)FINP
     21        OPEN(7,FILE=FINP,STATUS='OLD')
     22        READ(7,*)NPRF
     23        DO 0230 I=1,NPRF
1    24        READ(7,*)YBAR(I),YDEV(I),YMAX(I)
1    25        READ(7,*)YCUT(I,1),YCUT(I,2),XCRD(I)
1    26  0230  CONTINUE
     27        CLOSE(7)
     28        GO TO 0046
     29       ENDIF
     30       YMAX(1)=0.0
     31       YDEV(1)=1.0
     32       YCUT(1,1)=1.0
     33       YCUT(1,2)=2.0
     34       XCRD(1)=0.0
     35  0005 WRITE(*,*)'  GIVE CRACK LOCATION  '
     36       READ(*,*,ERR=0005)YBAR(1)
     37  0010 WRITE(*,*)'  ALIGN MEAN VALUES ?  YES ->1  NO ->0'
     38       READ(*,*,ERR=0010)IMOVE
     39 C
     40 C     Read in remaining input
     41 C
     42  0015 WRITE(*,*)' GIVE  TOTAL # OF PROFILES TO INPUT'
     43       READ(*,*,ERR=0015)NPRF
     44       NPRF=NPRF+1
     45  0017 WRITE(*,*)' GIVE SEARCH LIMITS  (lower,  upper)'
     46       READ(*,*,ERR=0017)SL,SU
     47 C
     48 C     Start input loop here
     49 C
     50       DO 0022 I=2,NPRF
1    51       I1=I-1
1    52  0025 WRITE(*,0030)I1
1    53  0030 FORMAT(//' GIVE  MEAN,  SDEV,  MAX,  FOR ',I3,'th PROFILE')
1    54       READ(*,*,ERR=0025)YBAR(I),YDEV(I),YMAX(I)
1    55  0035 WRITE(*,*)' GIVE CUTOFF LIMITS   (lower,  upper)'
1    56       DUM1=0.0

1    57       DUM2=0.0
1    58       READ(*,*,ERR=0035)DUM1,DUM2
1    59 C
```

```
D Line# 1
1     60          IF(DUM1.NE.0.0)THEN
1     61           YCUT(I,2)=DUM1
1     62          ELSE
1     63           YCUT(I,2)=SL
1     64          ENDIF
1     65 C
1     66          IF(DUM2.NE.0.0)THEN
1     67           YCUT(I,1)=DUM2
1     68          ELSE
1     69           YCUT(I,1)=SU
1     70          ENDIF
1     71 C
1     72          IF(IMOVE.EQ.1)THEN
1     73           YCUT(I,1)=YCUT(I,1)-YBAR(I)
1     74           YCUT(I,2)=YCUT(I,2)-YBAR(I)
1     75           YBAR(I)=0.0
1     76          ENDIF
1     77 C
1     78 C
1     79   0033 WRITE(*,*)'  GIVE X-COORDINATE FOR PROFILE '
1     80          READ(*,*,ERR=0033)XCRD(I)
1     81   0022 CONTINUE
      82          IN=0
      83   0210 WRITE(*,*) '  STORE INPUT DATA IN FILE  YES ->1  NO ->0'
      84          READ(*,*,ERR=0210)IN
      85          IF(IN.NE.0)THEN
      86   0215    WRITE(*,*)'  GIVE FILE NAME'
      87           READ(*,0055,ERR=0215)FINP
      88           OPEN(7,FILE=FINP,STATUS='NEW')
      89           WRITE(7,*)NPRF
      90           DO 0220 I=1,NPRF
1     91           WRITE(7,*)YBAR(I),YDEV(I),YMAX(I)
1     92           WRITE(7,*)YCUT(I,1),YCUT(I,2),XCRD(I)
1     93    0220   CONTINUE
      94           CLOSE(7)
      95          ENDIF
      96 C
      97 C      Start contour input loop here
      98 C
      99   0046 CONTINUE
     100   0045 WRITE(*,0040)
     101   0040 FORMAT(//'   CONTINUE ->0   STOP ->1')
     102          ISTP=0
     103          READ(*,*,ERR=0045)ISTP
     104          IF(ISTP.GT.0)GO TO 0999
     105 C
     106 C      Read in contour magnitude and file output name
     107 C
     108   0050 WRITE(*,*)'    GIVE MAGNITUDE OF DESIRED CONTOUR'
     109          READ(*,*,ERR=0050)CMAG
     110          N1=1
     111   0060 WRITE(*,*)'    GIVE OUTPUT FILE NAME'
     112          READ(*,0055,ERR=0060)FILENM
     113   0055 FORMAT(A10)
     114 C
     115 C      Check to see which profile exceeds the contour magnitude
     116 C
     117          DO 0065 I=1,NPRF
1.   118   0065 IF(YMAX(I).GE.CMAG)GO TO 0070
```

```
    119        WRITE(*,*)'  VALUE OF CONTOUR EXCEEDS DENSITY PEAK'
    120        GO TO 3046
    121  0070  CONTINUE
    122 C
    123 C      Compute the number of points required for contour
    124 C
    125        IPRST=I
    126  0120  CONTINUE
    127        IPRND=0
    128        DO 0075 I=IPRST,NPRF
1   129        IF(YMAX(I).LT.CMAG)THEN
1   130         IPRND=I-1
1   131         GO TO 3076
1   132        ENDIF
1   133  0075  CONTINUE
    134  0076  CONTINUE
    135        IF(IPRND.GT.0)THEN
    136         NPTS=2*(IPRND-IPRST+1)+3
    137        ENDIF
    138        IF(IPRND.EQ.0)THEN
    139         IPRND=NPRF
    140         NPTS=2*(IPRND-IPRST+1)+1
    141        ENDIF
    142        WRITE(*,*)'  NPTS =',NPTS
    143        WRITE(*,*)' IPRST, IPRND=',IPRST,IPRND
    144 C
    145 C      Compute the coordinates for the points
    146 C
    147        IF(IPRND.NE.NPRF)THEN
    148         IPR=IPRND+1
    149         CALL FINTNT(IPR,CMAG,VALX,VALY)
    150         PT(1,1)=VALX
    151         PT(1,2)=VALY
    152         PT(NPTS,1)=VALX
    153         PT(NPTS,2)=VALY
    154        ELSE
    155         PT(1,1)=XCRD(IPRND)
    156         PT(NPTS,1)=XCRD(IPRND)
    157         IPR=IPRND
    158         CALL FINDPT(IPR,CMAG,VALY1,VALY2)
    159         PT(1,2)=VALY1
    160         PT(NPTS,2)=VALY2
    161        ENDIF
    162        WRITE(*,*)'  COMPUTED FIRST 2 PTS'
    163        I1=1
    164        I2=NPTS
    165        IF(IPRND.NE.NPRF)THEN
    166         NSTOP=IPRND-IPRST+1
    167         IPR=IPRND+1
    168        ELSE
    169         NSTOP=IPRND-IPRST
    170         IPR=IPRND
    171        ENDIF
    172        DO 0080 I=1,NSTOP
1   173        I1=I1+1
1   174        I2=I2-1

1   175        IPR=IPR-1
1   176        PT(I1,1)=XCRD(IPR)
1   177        PT(I2,1)=XCRD(IPR)
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
1   178        IF(YMAX(IPR).EQ.CMAG)THEN
1   179         PT(I1,2)=YBAR(IPR)
1   180         PT(I2,2)=YBAR(IPR)
1   181        ELSE
1   182         WRITE(*,*)'  CALLING FINDPT'
1   183         WRITE(*,*)IPR,CMAG,VALY1,VALY2
1   184         CALL FINDPT(IPR,CMAG,VALY1,VALY2)
1   185         WRITE(*,*)'  RETURNING FROM FINDPT',VALY1,VALY2
1   186         PT(I1,2)=VALY1
1   187         PT(I2,2)=VALY2
1   188        ENDIF
1   189  0080  CONTINUE
    190        CALL LINPT(IPR,CMAG,VALX,VALY)
    191        PT(I1+1,1)=VALX
    192        PT(I1+1,2)=VALY
    193 C
    194 C      Write contour to file
    195 C
    196        OPEN(10,FILE=FILENM,STATUS='NEW')
    197        DO 0090 I=1,NPTS
1   198        WRITE(10,*)PT(I,1),PT(I,2)
1   199  0090  CONTINUE
    200        STP=-999.
    201        WRITE(10,*)STP,STP
    202        WRITE(10,0100)CMAG
    203  0100  FORMAT(' FILE FOR CONTOUR MAG=',G10.3)
    204        CLOSE(10)
    205 C
    206 C      Check for additional curve at the corresponding magnitude
    207 C
    208        FLG=0
    209        IF(IPRND.LT.NPRF)THEN
    210         IC=IPRND+1
    211         IPRST=0
    212         DO 0110 I=IC,NPRF
1   213         IF(YMAX(I).GE.CMAG)THEN
1   214         IPRST=I
1   215         GO TO 0111
1   216         ENDIF
1   217  0110   CONTINUE
    218  0111   CONTINUE
    219         IF(IPRST.GE.IC.AND.IPRST.LE.NPRF)THEN
    220         WRITE(*,*)' NEW STARTING PROFILE AT',IPRST
    221          WRITE(*,*)'  ANOTHER POSSIBLE CONTOUR AT',CMAG
    222  0130    WRITE(*,*)'  GIVE NEW FILE NAME'
    223          READ(*,2055,ERR=0130)FILENM
    224          FLG=1
    225         ENDIF
    226        ENDIF
    227        IF(FLG.EQ.1)GO TO 0120
    228        GO TO 0046
    229  0999  CONTINUE
    230        END

Name    Type         Offset P Class

CMAG    REAL            166
DUM1    REAL            118
DUM2    REAL            122
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

FILENM	CHAR*10	174	
FINP	CHAR*10	20	
FLG	REAL	336	
I	INTEGER*4	34	
I1	INTEGER*4	58	
I2	INTEGER*4	284	
IC	INTEGER*4	340	
IMOVE	INTEGER*4	42	
IN	INTEGER*4	16	
IPR	INTEGER*4	264	
IPRND	INTEGER*4	252	
IPRST	INTEGER*4	248	
ISTP	INTEGER*4	162	
N1	INTEGER*4	170	
NPRF	INTEGER*4	30	
NPTS	INTEGER*4	260	
NSTOP	INTEGER*4	288	
PT	REAL	0	/COMMQQ/
SL	REAL	46	
STP	REAL	300	
SU	REAL	50	
VALX	REAL	268	
VALY	REAL	272	
VALY1	REAL	276	
VALY2	REAL	280	
XCRD	REAL	2800	/COMMQQ/
YBAR	REAL	800	/COMMQQ/
YCUT	REAL	1600	/COMMQQ/
YDEV	REAL	1200	/COMMQQ/
YMAX	REAL	2400	/COMMQQ/

```
    231 C
    232 C
    233       SUBROUTINE LININT(I,CMAG,VALX,VALY)
    234       COMMON PT(100,2),YBAR(100),YDEV(100),YCUT(100,2),YMAX(100),
    235      1 XCRD(100)
    236       D=(YBAR(I-1)-YBAR(I))**2
    237       D=D+(XCRD(I-1)-XCRD(I))**2
    238       D=SQRT(D)
    239       DPR=(D*(CMAG-YMAX(I-1)))/(YMAX(I)-YMAX(I-1))
    240       VALX=(DPR*(XCRD(I)-XCRD(I-1)))/D
    241       VALY=(DPR*(YBAR(I)-YBAR(I-1)))/D
    242       VALX=VALX+XCRD(I-1)
    243       VALY=VALY+YBAR(I-1)
    244       RETURN
    245       END
```

Name	Type	Offset	P	Class
CMAG	REAL	4	*	
D	REAL	348		
DPR	REAL	352		
I	INTEGER*4	0	*	
PT	REAL	0		/COMMQQ/
SQRT				INTRINSIC
VALX	REAL	8	*	
VALY	REAL	12	*	
XCRD	REAL	2800		/COMMQQ/

```
D Line# 1      7
YBAR    REAL              800   /COMMQQ/
YCUT    REAL             1600   /COMMQQ/
YDEV    REAL             1200   /COMMQQ/
YMAX    REAL             2400   /COMMQQ/


   246 C
   247 C
   248       SUBROUTINE FINDPT(I,CMAG,VALY1,VALY2)
   249       COMMON PT(100,2),YBAR(100),YDEV(100),YCUT(100,2),YMAX(100),
   250      1 XCRD(100)
   251 C
   252 C     Check first to see if CMAX falls within the cutoff
   253 C
   254       FLAG=0
   255       LL=1
   256       Y1=YBAR(I)
   257       Y2=YCUT(I,LL)
   258  0100 CONTINUE
   259       YG=YCUT(I,LL)
   260       CALL FX(I,YG,YMAG)
   261       IF(CMAG.LE.YMAG)THEN
   262        YVAL=YCUT(I,LL)
   263       ELSE
   264 C
   265 C     Perform bisecting convergence routine
   266 C
   267
   268  0010  CONTINUE
   269        YG=(Y1+Y2)/2.0
   270        CALL FX(I,YG,YMAG)
   271        CON=((YMAG-CMAG)*100.0)/CMAG
   272        IF(ABS(CON).LE.1.0)GO TO 0020
   273        IF(FLAG.EQ.0)THEN
   274        IF(YMAG.GT.CMAG)THEN
   275         Y1=YG
   276        ELSE
   277         Y2=YG
   278        ENDIF
   279        ELSE
   280        IF(YMAG.GT.CMAG)THEN
   281         Y2=YG
   282        ELSE
   283         Y1=YG
   284        ENDIF
   285        ENDIF
   286        GO TO 0010
   287  0020  CONTINUE
   288        YVAL=YG
   289       ENDIF
   290       IF(FLAG.EQ.0)THEN
   291        VALY1=YVAL
   292       ELSE
   293        VALY2=YVAL
   294       ENDIF
   295       IF(FLAG.EQ.0)THEN

   296        FLAG=1
   297        Y2=YBAR(I)
   298        LL=2
```

```
D Line# 1     7
    299        Y1=YCUT(I,LL)
    300        GO TO 0100
    301       ENDIF
    302       RETURN
    303       END
```

Name	Type	Offset	P	Class
ABS				INTRINSIC
CMAG	REAL	4	*	
CON	REAL	384		
FLAG	REAL	356		
I	INTEGER*4	0	*	
LL	INTEGER*4	360		
PT	REAL	0		/COMMQQ/
VALY1	REAL	8	*	
VALY2	REAL	12	*	
XCRD	REAL	2800		/COMMQQ/
Y1	REAL	364		
Y2	REAL	368		
YBAR	REAL	800		/COMMQQ/
YCUT	REAL	1600		/COMMQQ/
YDEV	REAL	1200		/COMMQQ/
YG	REAL	372		
YMAG	REAL	376		
YMAX	REAL	2400		/COMMQQ/
YVAL	REAL	380		

```
    304
    305 C
    306 C
    307 C
    308       SUBROUTINE FX(I,YG,YMAG)
    309       COMMON PT(100,2),YBAR(100),YDEV(100),YCUT(100,2),YMAX(100),
    310      1 XCRD(100)
    311       EX=(YG-YBAR(I))/YDEV(I)
    312       EX=-0.5*(EX*EX)
    313       YMAG=EXP(EX)*YMAX(I)
    314       RETURN
    315       END
```

Name	Type	Offset	P	Class
EX	REAL	388		
EXP				INTRINSIC
I	INTEGER*4	0	*	
PT	REAL	0		/COMMQQ/
XCRD	REAL	2800		/COMMQQ/
YBAR	REAL	800		/COMMQQ/
YCUT	REAL	1600		/COMMQQ/
YDEV	REAL	1200		/COMMQQ/
YG	REAL	4	*	
YMAG	REAL	8	*	
YMAX	REAL	2400		/COMMQQ/

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

Name	Type	Size	Class
COMMQQ		3200	COMMON
CONTOU			PROGRAM
FINDPT			SUBROUTINE
FX			SUBROUTINE
LININT			SUBROUTINE

Pass One No Errors Detected
315 Source Lines

```
D Line# 1      7                                Microsoft FORTRAN77 V3.20 02/84
      1 $DEBUG
      2         PROGRAM REPVOL
      3 C*************************************************************************
      4 C***** This program is designed to help the user determine the    ***
      5 C***** representative volume for a specified sample by :          ***
      6 C*****  1) Computing the spread of diversity in a          ***
      7 C*****        by creating histograms for a certian          ***
      8 C*****     dimension.                                      ***
      9 C*****  2) Increasing the size of the dimension           ***
     10 C*****     recomputing the effective spread in the        ***
     11 C*****     sample.                                        ***
     12 C*************************************************************************
     13        DIMENSION GR(20,20),CREP(3,3),BAREP(20),SDREP(20)
     14        CHARACTER*10 FILE1
     15        WRITE(*,*)'   GIVE INPUT FILE NAME'
     16        READ(*,0005)FILE1
     17  0005  FORMAT(A10)
     18        OPEN(10,FILE=FILE1,STATUS='OLD')
     19        READ(10,*)GRDM
     20        DO 0010 I1=1, 20
1    21          DO 0010 I2=1, 20
2    22  0010    READ(10,*)GR(I1,I2)
     23  0015  WRITE(*,*)'   GIVE NO. OF EVALUATIONS TO PERFORM'
     24        READ(*,*,ERR=0015)NAN
     25        IF(NAN.LE.0)NAN=1
     26 C
     27 C      Input is complete,  Thus start Analysis
     28 C
     29        DO 0020 I1=1, NAN
1    30        WRITE(*,*)'  WORKING ON ',I1,'th  DIMENSION'
1    31        AREP=(FLOAT(I1) GRDM)*(FLOAT(I1)*GRDM)
1    32        DO 0030 I2=1, 3
2    33        DO 0030 I3=1, 3
3    34 C
3    35        ISTR1=((I2-1)*I1)+1
3    36        ISTP1=ISTR1+I1-1
3    37        ISTR2=((I3-1)*I1)+1
3    38        ISTP2=ISTR2+I1-1
3    39        CREP(I2,I3)=0.0
3    40        DO 0040 I4=ISTR1,ISTP1
4    41        DO 0040 I5=ISTR2,ISTP2
5    42  0040  CREP(I2,I3)=CREP(I2,I3)+GR(I4,I5)
3    43        CREP(I2,I3)=(CREP(I2,I3)*100.0)/AREP
3    44  0030  CONTINUE
1    45 C
1    46 C      COMPUTE THE MEAN AND VARIATION FOR PARTICULAR DIMENSION
1    47 C
1    48        BAREP(I1)=0.0
1    49        SDREP(I1)=0.0
1    50        DO 0050 I2=1,3
2    51        DO 0050 I3=1,3
3    52        RFI=CREP(I2,I3)
3    53        SDREP(I1)=SDREP(I1)+((RFI*RFI)/9.0)
3    54        BAREP(I1)=BAREP(I1)+(RFI/9.0)
3    55  0050  CONTINUE
1    56        SDREP(I1)=SQRT(SDREP(I1)-(BAREP(I1)*BAREP(I1)))

1    57  0020  CONTINUE
     58 C
     59 C      WRITE OUT THE FINAL RESULTS
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
      60 C
      61  0060 WRITE(*,*)'   TERMINAL ->0  PRINTER ->1'
      62       READ(*,*,ERR=0060)IW
      63       IF(IW.NE.0)THEN
      64        OPEN(7,FILE='PRN')
      65        IW=7
      66       ENDIF
      67       IWA=0
      68  0100 WRITE(IWA,0110)
      69  0110 FORMAT('1',//,
      70      1 ' |-------------------------------------------------------------|'/
      71      2 ' |                REPRESENTATIVE VOLUME ANALYSIS               |'/
      72      3 ' |---------|---------------|--------------|-------------------|'/
      73      4 ' |  Anal   |   Dimension   |     Mean     |     Std. Dev.     |'/
      74      5 ' |   No.   |    (M*E-4)    |     (%)      |       (%)         |'/
      75      6 ' |---------|---------------|--------------|-------------------|')
      76       DO 0120 I1=1,NAN
1     77       DIM=GRDM*FLOAT(I1)
1     78       WRITE(IWA,0130)I1,DIM,BAREP(I1),SDREP(I1)
1     79  0130 FORMAT(
1     80      1 '  |   ',I5,'   |   ',F10.2,'    |   ',F10.2,'    |   ',F10.2,'    |',/,
1     81      2 '  |---------|---------------|--------------|-------------------|')
1     82  0120 CONTINUE
      83       WRITE(IWA,0125)
      84  0125 FORMAT('1',//,20X,'--- CLUSTER AREA DATA BASE ---')
      85       DO 0135 I1=1,20
1     86  0135 WRITE(IWA,0140)(GR(I1,I2),I2=1,20)
      87  0140 FORMAT(/,20(1XF3.0))
      88       IF(IW.EQ.7.AND.IWA.EQ.0)THEN
      89        IWA=7
      90        GO TO 0100
      91       ENDIF
      92       END

Name     Type         Offset P Class

AREP     REAL           1348
BAREP    REAL           1732
CREP     REAL           1696
DIM      REAL           2360
FILE1    CHAR*10        1812
FLOAT                         INTRINSIC
GR       REAL             16
GRDM     REAL           1828
I1       INTEGER*4      1832
I2       INTEGER*4      1836
I3       INTEGER*4      1852
I4       INTEGER*4      1872
I5       INTEGER*4      1880
ISTP1    INTEGER*4      1860
ISTP2    INTEGER*4      1868
ISTR1    INTEGER*4      1856
ISTR2    INTEGER*4      1864
IW       INTEGER*4      1892
IWA      INTEGER*4      1896
NAN      INTEGER*4      1840

RFI      REAL           1888
SDREP    REAL           1616
SQRT                          INTRINSIC
```

0 Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

Name	Type	Size	Class
REPVOL			PROGRAM

Pass One No Errors Detected
92 Source Lines

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
      1 $debug
      2       Program Morphol
      3 C*************************************************************************
      4 C***** This program is designed to compute the number of clusters **
      5 C***** in a given class per unit volume of material.  This is to  **
      6 C***** be performed using an algorithm similar to that used in De-**
      7 C***** Hoffs ellipsiod analysis.  This algorithm, however, has    **
      8 C***** been generalized to compute the number of both oblate and  **
      9 C***** prolate spheroids within a given class as well as a poss-  **
     10 C***** ible aspect ratio for each class.                          **
     11 C*************************************************************************
     12       real*8 rj,ra,ram1,phi,qs,rho,rhoa,rhom1,dphi,c(40,40)
     13       real*4 ri(1000),s(1000),na(40),nj(40)
     14       dimension drj(2),q(2),k(2)
     15       character*10 fil
     16       character*3 label(2)
     17       character*40 title
     18       data label/'Pro','Obl'/
     19 c
     20 c     Input the necessary Information
     21 c
     22  0005 write(*,*)'   Give filename '
     23       read(*,0010,err=0005)fil
     24  0010 format(a10)
     25       open(5,file=fil,status='old')
     26       read(5,*)kt,delr
     27       do 0013 i=1,kt
  1  28  0013 read(5,*)na(i)
     29       close(5)
     30  0015 write(*,*)'   Give no. of prolate and oblate classes'
     31       read(*,*,err=0015)k(1),k(2)
     32       if(k(1).gt.0)then
     33  0020  write(*,*)'   Give prmax and aspect ratio'
     34        read(*,*,err=0020)prmax,q(1)
     35        drj(1)=prmax/float(k(1))
     36       endif
     37       if(k(2).gt.0)then
     38  0025  write(*,*)'   Give ormax and aspect ratio'
     39        read(*,*,err=0025)ormax,q(2)
     40        drj(2)=ormax/float(k(2))
     41       endif
     42       kut=k(1)+k(2)
     43       write(*,*)'   Give title'
     44       read(*,0030)title
     45  0030 format(a40)
     46 c
     47 c     Integrate the probability coefficients
     48 c
     49       do 0100 ia=1,kt
  1  50       ra=delr*float(ia)
  1  51       ram1=delr*float(ia-1)
  1  52       do 0110 j=1,kut
  2  53       c(ia,j)=0.0
  2  54       if(j.le.k(1))then
  2  55        rj=dble(drj(1)*float(j))
  2  56        qs=dble(q(1))

  2  57       else
  2  58        rj=dble(drj(2)*float(j-k(1)))
  2  59        qs=dble(1.0/q(2))
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
2     60       endif
2     61       dphi=1.570796327/180.0
2     62       do 0120 i1=1,180
3     63       phi=1.5707963*dble(i1)/180.0
3     64       call comrho(rj,ram1,phi,qs,rho)
3     65       rhom1=rho
3     66       call comrho(rj,ra,phi,qs,rho)
3     67       rhoa=rho
3     68       c(ia,j)=c(ia,j)+((rhom1-rhoa)*dsin(phi)*dphi)
3     69  0120 continue
2     70       c(ia,j)=2.0*c(ia,j)
2     71       if(c(ia,j).lt.0.00001)c(ia,j)=0.0
2     72       write(*,*)'  '
2     73       write(*,0109)ia,j,c(ia,j)
2     74  0109 format(///'     C(',i3,',',i3,')=',g15.9)
2     75  0110 continue
1     76  0100 continue
      77 c
      78 c     solve for the Nj's
      79 c
      80       iwrong=0
      81 c
      82       call prereg(c,na,kut,kt,iwrong)
      83       if(iwrong.eq.1)go to 0999
      84 c
      85       call solve(kut,c,nj,na)
      86 c
      87 c     Write out the output for the cases
      88 c
      89  0200 write(*,*)'  Terminal ->0   Printer ->1   File ->2'
      90       read(*,*,err=0200)iflg
      91       if(iflg.eq.1)open(7,file='prn')
      92       if(iflg.eq.2)open(7,file='table.dat',status='new')
      93       iw=0
      94  0210 continue
      95       write(iw,0220)title
      96  0220 FORMAT('1',///,
      97      1 ' |-------------------------------------------------------------|'/
      98      2 ' |                                                             |'/
      99      3 ' |          Kunin-Lesser  ELLIPSOID ANALYSIS                   |'/
     100      4 ' |          for VOLUMETRIC CLUSTER DISTRIBUTION                |'/
     101      5 ' |                                                             |'/
     102      6 ' |       ',A40,'       |'/
     103      7 ' |------|---------|--------|---------|----------|---------|'/
     104      8 ' |      |         |        |         |          |         |'/
     105      9 ' |Partcl| Equiv.  |        | Partcle | No. of   |         |'/
     106      A ' | Type | Partcle | Aspct. |  Vol.   | Particle | Class   |'/
     107      B ' |      |  Rad.   | Ratio  | (mm3)   |  per     | Volume  |'/
     108      C ' |      |  (mm)   |        |         | (mm3)    | (mm3)   |'/
     109      D ' |------|---------|--------|---------|----------|---------|')
     110       tvol=0.0
     111       do 0250 i=1,kt
1    112       if(i.le.k(1))then
1    113        it=1
1    114        j=i
1    115       else
1    116        it=2
1    117        j=i-k(1)
1    118       endif
1    119       rad=drj(it)*float(j)
1    120       pvol=4.188790204*(rad**3)
```

```
D Line# 1     7                                  Microsoft FORTRAN77 V3.20 02/84
1   119       cvol=nj(i)*pvol
1   120       tvol=tvol+cvol
1   121       write(iw,0230)label(it),rad,q(it),pvol,nj(i),cvol
1   122   0230 format(
1   123      1 ' |   ',a3,' |',f9.5,'|'f8.4,'|',f9.5,'|',f10.6,'|',
1   124      2 f9.6,'|')
1   125       write(iw,0240)
1   126   0240 format(
1   127      1 ' |-----|---------|--------|---------|---------|---------|')
1   128   0250 continue
    129       write(iw,0270)tvol
    130   0270 format(
    131      1 ' |                  Total Volume = ',f12.6,'              |'/
    132      2 ' |-------------------------------------------------------|')
    133
    134       if(iflg.eq.2.and.iw.eq.7)close(7)
    135       if(iflg.ne.0.and.iw.eq.0)then
    136        iw=7
    137        go to 0210
    138       endif
    139 c
    140 c     Write output for file information
    141 c
    142       if(k(1).gt.0)then
    143         sum=0.0
    144       do 0300 i=1,k(1)
1   145       sum=sum+nj(i)
1   146   0300 continue
    147       rmp=0.0
    148       sdp=0.0
    149       do 0310 i=1,k(1)
1   150       rad=float(i)*drj(1)
1   151       rmp=rmp+(rad*nj(i)/sum)
1   152       sdp=sdp+(rad*rad*nj(i)/sum)
1   153   0310 continue
    154       open(7,file='lpt1')
    155       write(7,0320)k(1),rmp,sdp
    156   0320 format('   *** ESTIMATED DISTN ***',/
    157      1 '     No. Prolate Classes =',i5,/
    158      2 '     MEAN=',G10.4,'   SDEV=',G10.4)
    159       close(7)
    160       endif
    161       if(k(2).gt.0)then
    162         sum=0.0
    163       ktot=k(1)+k(2)
    164       do 0330 i=k(1),ktot
1   165       sum=sum+nj(i)
1   166   0330 continue
    167       rmo=0.0
    168       sdo=0.0
    169       do 0340 i=k(1),ktot
1   170       j=k(1)-i
1   171       rad=float(j)*drj(2)
1   172       rmo=rmo+(rad*nj(i)/sum)
1   173       sdo=sdo+(rad*rad*nj(i)/sum)
1   174   0340 continue

    175       open(7,file='lpt1')
    176       write(7,0350)k(2),rmo,sdo
    177   0350 format('   *** ESTIMATED DISTN ***',/
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
    178       1 '     No. Oblate Classes=',i5,/
    179       2 '     OBLATE MEAN=',G10.4,'   SDEV=',G10.4)
    180        close(7)
    181        endif
    182  0999  continue
    183        end
```

Name	Type	Offset	P	Class
C	REAL*8	8362		
CVOL	REAL	22250		
DBLE				INTRINSIC
DELR	REAL	21182		
DPHI	REAL*8	21354		
DRJ	REAL	8340		
DSIN				INTRINSIC
FIL	CHAR*10	21162		
FLOAT				INTRINSIC
I	INTEGER*4	21186		
I1	INTEGER*4	21362		
IA	INTEGER*4	21306		
IFLG	INTEGER*4	21438		
IT	INTEGER*4	22238		
IW	INTEGER*4	21442		
IWRONG	INTEGER*4	21434		
J	INTEGER*4	21330		
K	INTEGER*4	8348		
KT	INTEGER*4	21178		
KTOT	INTEGER*4	22890		
KUT	INTEGER*4	21202		
LABEL	CHAR*3	8356		
NA	REAL	8020		
NJ	REAL	8180		
ORMAX	REAL	21198		
PHI	REAL*8	21366		
PRMAX	REAL	21194		
PVOL	REAL	22246		
Q	REAL	8012		
QS	REAL*8	21346		
RA	REAL*8	21314		
RAD	REAL	22242		
RAM1	REAL*8	21322		
RHO	REAL*8	21374		
RHOA	REAL*8	21390		
RHOM1	REAL*8	21382		
RI	REAL	4012		
RJ	REAL*8	21338		
RMO	REAL	22898		
RMP	REAL	22714		
S	REAL	12		
SDO	REAL	22902		
SDP	REAL	22718		
SUM	REAL	22706		
TITLE	CHAR*40	21206		
TVOL	REAL	22230		

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
    186 c
    187         Subroutine comrho(Rj,ri,phi,qs,rho)
    188 C*********************************************************************
    189 C***** This routine is designed to compute the value of rho in   ***
    190 C***** Dehoff's solution for a prolate spheriod given a value of ***
    191 C***** Rj, ri, Q, and angle phi.  (phi is in radians)            ***
    192 C*****     QS=Qj     (PROLATE)                 ***
    193 C*****     QS=1.0/Qj (OBLATE)                  ***
    194 C*********************************************************************
    195         real*8 l,m,n,Rj,ri,phi,qs,rho,dc,ds
    196         dc=dcos(phi)
    197         ds=dsin(phi)
    198         l=(dc*dc)+(qs*qs*ds*ds)
    199         m=(ds*ds)+(qs*qs*dc*dc)
    200         n=(ds*ds*dc*dc)*(1.0-qs*qs)
    201 c
    202 c       Compute Rho
    203 c
    204         rho=(ri*ri)/dsqrt(l)
    205         rho=rho-((((qs)**1.666666666)*Rj*Rj)/l)
    206         rho=(rho*l*l)/(n-(m*l))
    207         if(rho.lt.0.0)the
    208          rho=0.0
    209         else
    210          rho=dsqrt(rho)
    211         endif
    212         return
    213         end
```

Name	Type	Offset	P	Class
DC	REAL*8	23082		
DCOS				INTRINSIC
DS	REAL*8	23090		
DSIN				INTRINSIC
DSQRT				INTRINSIC
L	REAL*8	23098		
M	REAL*8	23106		
N	REAL*8	23114		
PHI	REAL*8	8	*	
QS	REAL*8	12	*	
RHO	REAL*8	16	*	
RI	REAL*8	4	*	
RJ	REAL*8	0	*	

```
    214 C
    215 C
    216 C
    217         SUBROUTINE PREREG(A,B,M,MM,IWRONG)
    218        REAL*8 A(40,40),ASQ(40,40)
    219        DIMENSION B(40),BSQ(40)
    220        IF(M.GT.MM)THEN
    221         WRITE(*,*)'   SYSTEM OF EQS CANT BE SOLVED'
    222         IWRONG=1
    223         GO TO 0999
    224        ENDIF
    225        DO 0010 I=1,M
  1 226        DO 0020 J=1,M
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
2    227       ASQ(I,J)=0.0
2    228       DO 0030 K=1,MM
3    229       ASQ(I,J)=ASQ(I,J)+(A(K,I)*A(K,J))
3    230  0030 CONTINUE
2    231  0020 CONTINUE
2    232  0010 CONTINUE
     233 C
     234       DO 0040 I=1,M
1    235       BSQ(I)=0.0
1    236       DO 0050 K=1,MM
2    237       ADUM=SNGL(A(K,I))
2    238       BSQ(I)=BSQ(I)+(ADUM*B(K))
2    239  0050 CONTINUE
1    240  0040 CONTINUE
     241 C
     242       DO 0060 I=1,M
1    243       DO 0070 J=1,M
2    244       A(I,J)=ASQ(I,J)
2    245  0070 CONTINUE
1    246       B(I)=BSQ(I)
1    247  0060 CONTINUE
     248  0999 RETURN
     249       END

Name     Type         Offset P Class

A        REAL*8            0 *
ADUM     REAL          36114
ASQ      REAL*8        23282
B        REAL              4 *
BSQ      REAL          23122
I        INTEGER*4     36082
IWRONG   INTEGER*4        16 *
J        INTEGER*4     36090
K        INTEGER*4     36098
M        INTEGER*4         8 *
MM       INTEGER*4        12 *
SNGL                       INTRINSIC


     250
     251 C
     252         SUBROUTINE SOLVE(N,A,X1,B1)
     253 C*************************************************************************
     254 C***** This routine solves a linear system of equations ( Ax=b)  ***
     255 C***** by using Gauss reduction.  This method does not require  ***
     256 C*****  that the matrix A be symmetric.                          ***
     257 C*************************************************************************
     258       real*8 A(40,40),X(40),B(40)
     259       dimension X1(40),B1(40)
     260       do 0010 i=1,40
1    261       B(I)=OBLE(B1(I))
1    262  0010 CONTINUE
     263 C
     264 C     Triangularize the A matrix
     265 C
     266       NN=N-1
     267       DO 0100 K=1,NN
1    268       KK=K+1
```

```
D Line# 1      7                                   Microsoft FORTRAN77 V3.20 02/84
1   269        DO 0100 I=KK,N
2   270        C=A(I,K)/A(K,K)
2   271        B(I)=B(I)-B(K)*C
2   272        DO 0100 J=K,N
3   273  0100  A(I,J)=A(I,J)-A(K,J)*C
    274 C
    275 C      Solve for x vector (solution vector)
    276 C
    277        X(N)=B(N)/A(N,N)
    278        DO 0300 K=1,NN
1   279        I=N-K
1   280        II=I+1
1   281        C=0.0
1   282        DO 0400 J=II,N
2   283  0400  C=C+A(I,J)*X(J)
1   284        X(I)=(B(I)-C)/A(I,I)
1   285  0300  CONTINUE
    286        DO 0030 I=1,40
1   287  0030  X1(I)=SNGL(X(I))
    288        RETURN
    289        END
```

Name	Type	Offset	P	Class
A	REAL*8	4	*	
B	REAL*8	36446		
B1	REAL	12	*	
C	REAL	36790		
DBLE				INTRINSIC
I	INTEGER*4	36766		
II	INTEGER*4	36806		
J	INTEGER*4	36794		
K	INTEGER*4	36774		
KK	INTEGER*4	36782		
N	INTEGER*4	0	*	
NN	INTEGER*4	36770		
SNGL				INTRINSIC
X	REAL*8	36126		
X1	REAL	8	*	

Name	Type	Size	Class
COMRHO			SUBROUTINE
MORPHO			PROGRAM
PREREG			SUBROUTINE
SOLVE			SUBROUTINE

```
 Pass One     No Errors Detected
             289 Source Lines
```

```
D Line# 1     7                                  Microsoft FORTRAN77 V3.20 02/84
      1 $debug
      2         Program Morphol
      3 C*************************************************************************
      4 C***** This program is designed to compute the number of clusters **
      5 C***** in a given class per unit volume of material.  This is to  **
      6 C***** be performed using an algorithm similar to that used in De-**
      7 C***** Hoffs ellipsiod analysis.  This algorithm, however, has    **
      8 C***** been generalized to compute the number of both oblate and  **
      9 C***** prolate spheroids within a given class as well as a poss-  **
     10 C***** ible aspect ratio for each class.                          **
     11 C*************************************************************************
     12       real*8 rj,ra,ra1,phi,qs,rho,rhoa,rho1,dphi,c(40,40)
     13       real*4 ri(1000),s(1000),na(40),nj(40)
     14       dimension drj(2),q(2),k(2)
     15       character*10 afil,sfil
     16       character*3 label(2)
     17       character*40 title
     18       data label/'Pro','Obl'/
     19 c
     20 c     Input the necessary Information
     21 c
     22  0005 write(*,*)'    Give the filename for the AREA ARRAY'
     23       read(*,0010,err=0005)afil
     24  0010 format(a10)
     25       open(5,file=afil,status='old')
     26       rmax=0.0
     27       do 0015 i=1,1000
1    28       read(5,*)dum
1    29       if(dum.eq.-999.)go to 0020
1    30       ri(i)=(sqrt(dum/3.14159))/10.0
1    31       if(ri(i).gt.rmax)rmax=ri(i)
1    32  0015 continue
     33  0020 continue
     34       npts=i-1
     35       close(5)
     36 c
     37  0025 write(*,*)'    Give the filename for the SHAPE FACT ARRAY'
     38       read(*,0010,err=0025)sfil
     39       open(5,file=sfil,status='old')
     40       amin=1.0
     41       amax=0.0
     42       aave=0.0
     43       do 0026 i=1,1000
1    44       read(5,*)dum
1    45       if(dum.eq.-999.)go to 0030
1    46       s(i)=dum
1    47       if(s(i).lt.amin)amin=s(i)
1    48       if(s(i).gt.amax)amax=s(i)
1    49       aave=aave+s(i)
1    50  0026 continue
     51  0030 continue
     52       close(5)
     53       aave=aave/float(i-1)
     54       write(*,*)'     No. of Points =',npts
     55       write(*,*)'  '
     56  0035 write(*,*)'     Give desired no of classes   1<->20'
     57       write(*,*)'     for the prolate and oblate respectively'
     58       read(*,*,err=0035)k(1),k(2)
     59       write(*,*)'  '
```

```
D Line# 1     7                                  Microsoft FORTRAN77 V3.20 02/84
     60       if(k(1).gt.20)k(1)=10
     61       if(k(2).gt.20)k(2)=10
     62       write(*,*)'   ASPECT RATIO INFORMATION'
     63       write(*,*)'   Max=',amax,'  Min=',amin,'  Ave=',aave
     64       if(k(1).gt.0)then
     65         write(*,*)'  Prolate give aspect ratio, maximum radius'
     66         read(*,*)q(1),prmax
     67 c       prmax=rmax/(q(1)**0.3333)
     68 c       prmax=rmax
     69         drj(1)=prmax/float(k(1))
     70         write(*,*)' Prolate Equiv. Rad inc= =',drj(1)
     71       endif
     72       if(k(2).gt.0)then
     73         write(*,*)'  Oblate give aspect ratio, maximum radius'
     74         read(*,*)q(2),ormax
     75 c       ormax=rmax*(q(2)**0.1667)
     76 c       ormax=rmax
     77         drj(2)=ormax/float(k(2))
     78         write(*,*)' Oblate Equiv. Rad. inc =',drj(2)
     79       endif
     80       write(*,*)'     Give Title for data set'
     81       read(*,0040)title
     82  0040 format(a40)
     83  0045 write(*,*)'     Give size of ANALYSIS AREA    (mm2)'
     84       read(*,*,err=0045)tarea
     85 c
     86 c     Compute the maximum radius, ra, and na
     87 c
     88       kt=k(1)+k(2)
     89       delr=rmax/float(kt)
     90       open(5,file='td.dat',status='new')
     91       write(*,0040)title
     92       do 0060 i=1,kt
1    93       bl=delr*float(i-1)
1    94       ul=delr*float(i)
1    95       nct=0
1    96       do 0055 j=1,npts
2    97  0055 if(ri(j).gt.bl.and.ri(j).le.ul)nct=nct+1
1    98       na(i)=float(nct)/tarea
1    99       write(*,*)'   na(',i,')=',na(i)
1   100       write(5,*)bl,0.0
1   101       write(5,*)bl,na(i)
1   102       write(5,*)ul,na(i)
1   103       write(5,*)ul,0.0
1   104  0060 continue
    105       write(5,*)-999.,-999.
    106       close(5)
    107 c
    108 c     Integrate the probability coefficients
    109 c
    110       open(6,file='car.dat',status='new')
    111       do 0100 ia=1,kt
1   112       ra=delr*float(ia)
1   113       ral=delr*float(ia-1)
1   114       do 0110 j=1,kt
2   115       c(ia,j)=0.0
2   116       if(j.le.k(1))then
2   117        rj=dble(drj(1)*float(j))
2   118        qs=dble(q(1))
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
2   119        else
2   120         rj=dble(drj(2)*float(j-k(1)))
2   121         qs=dble(1.0/q(2))
2   122        endif
2   123        dphi=1.570796327/180.0
2   124        do 0120 il=1,180
3   125        phi=1.5707963*dble(il)/180.0
3   126        call comrho(rj,ram1,phi,qs,rho)
3   127        rhom1=rho
3   128        call comrho(rj,ra,phi,qs,rho)
3   129        rhoa=rho
3   130        c(ia,j)=c(ia,j)+((rhom1-rhoa)*dsin(phi)*dphi)
3   131  0120  continue
2   132        c(ia,j)=2.0*c(ia,j)
2   133        if(c(ia,j).lt.0.00001)c(ia,j)=0.0
2   134        write(*,*)'  '
2   135        write(*,0109)ia,j,c(ia,j)
2   136        write(6,0109)ia,j,c(ia,j)
2   137  0109  format(///'     C(',i3,',',i3,')=',g15.9)
2   138  0110  continue
1   139  0100  continue
    140        close(6)
    141  c
    142  c     solve for the Nj's
    143  c
    144        call solve(kt,c,nj,na)
    145  c     do 0150 i=1,kt
    146  c     if(nj(i).lt.0.0)nj(i)=0.0
    147  c 0150 continue
    148  c
    149  c     Write out the output for the cases
    150  c
    151   0200 write(*,*)'  Terminal ->0   Printer ->1   File ->2'
    152        read(*,*,err=0200)iflg
    153        if(iflg.eq.1)open(7,file='prn')
    154        if(iflg.eq.2)open(7,file='table.dat',status='new')
    155        iw=0
    156   0210 continue
    157        write(iw,0220)title
    158   0220 FORMAT('1',///,
    159       1 ' |-------------------------------------------------------------|'/
    160       2 ' |            Kunin-Lesser  ELLIPSOID ANALYSIS                |'/
    161       3 ' |            for VOLUMETRIC CLUSTER DISTRIBUTION              |'/
    162       4 ' |                                                             |'/
    163       5 ' |         ',A40,'       |'/
    164       6 ' |------|---------|--------|--------|---------|--------|'/
    165       7 ' |      | Equiv.  |        |        |         |        |'/
    166       8 ' |Partcl| Partcle |Aspct.  |Partcle | No. of  |        |'/
    167       9 ' | Type | Rad.    | Ratio  | Vol.   |Particle | Class  |'/
    168       A ' |      | (mm)    |        | (mm3)  | per     | Volume |'/
    169       B ' |      |         |        |        | (mm3)   | (mm3)  |')
    170        tvol=0.0
    171        do 0250 i=1,kt
1   172        if(i.le.k(1))then
1   173         it=1
1   174         j=i

1   175        else
1   176         it=2
1   177         j=i-k(1)
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1   178        endif
1   179        rad=drj(it)*float(j)
1   180        pvol=4.188790204*(rad**3)
1   181        cvol=nj(i)*pvol
1   182        tvol=tvol+cvol
1   183        write(iw,0230)label(it),rad,q(it),pvol,nj(i),cvol
1   184   0230 format(
1   185       1 ' |  ',a3,' |',f9.5,'|'f8.4,'|',f9.5,'|',f10.6,'|',
1   186       2 f9.6,'|')
1   187        write(iw,0240)
1   188   0240 format(
1   189       1 ' |------|---------|--------|---------|---------|--------|')
1   190   0250 continue
     191        write(iw,0270)tvol
     192   0270 format(
     193       1 ' |                Total Volume = ',f12.6,'             |'/
     194       2 ' |-----------------------------------------------------|')
     195
     196        if(iflg.eq.2.and.iw.eq.7)close(7)
     197        if(iflg.ne.0.and.iw.eq.0)then
     198         iw=7
     199         go to 0210
     200        endif
     201  c
     202  c     Write output for file information
     203  c
     204        stp=-999.
     205        if(k(1).gt.0)then
     206          open(8,file='cvolp.dat',status='new')
     207          open(9,file='tvolp.dat',status='new')
     208          tvol=0.0
     209          do 0280 i=1,k(1)
1    210          rad=drj(1)*float(i)
1    211          rdm1=drj(1)*float(i-1)
1    212          pvol=4.188790204*(rad**3)
1    213          cvol=nj(i)*pvol
1    214          tvol=tvol+cvol
1    215          write(8,*)rdm1,0.0
1    216          write(8,*)rdm1,cvol
1    217          write(8,*)rad,cvol
1    218          write(8,*)rad,0.0
1    219          write(9,*)rdm1,tvol
1    220          write(9,*)rad,tvol
1    221   0280   continue
     222          write(8,*)stp,stp
     223          write(9,*)stp,stp
     224          close(8)
     225          close(9)
     226        endif
     227        if(k(2).gt.0)then
     228          open(8,file='cvolo.dat',status='new')
     229          open(9,file='tvolo.dat',status='new')
     230          tvol=0.0
     231          do 0290 i=1,k(2)
1    232          rad=drj(2)*float(i)
1    233          rdm1=drj(2)*float(i-1)
1    234          pvol=4.188790204*(rad**3)
1    235          cvol=nj(i)*pvol
1    236          tvol=tvol+cvol
```

```
D Line# 1     7
1   237           write(8,*)rdm1,0.0
1   238           write(8,*)rdm1,cvol
1   239           write(8,*)rad,cvol
1   240           write(8,*)rad,0.0
1   241           write(9,*)rdm1,tvol
1   242           write(9,*)rad,tvol
1   243   0290    continue
    244           write(8,*)stp,stp
    245           write(9,*)stp,stp
    246           close(8)
    247           close(9)
    248         endif
    249         end
```

Name	Type	Offset	P	Class
AAVE	REAL	21212		
AFIL	CHAR*10	21162		
AMAX	REAL	21208		
AMIN	REAL	21204		
BL	REAL	21286		
C	REAL*8	8362		
CVOL	REAL	22242		
DBLE				INTRINSIC
DELR	REAL	21278		
DPHI	REAL*8	21350		
DRJ	REAL	8340		
DSIN				INTRINSIC
DUM	REAL	21186		
FLOAT				INTRINSIC
I	INTEGER*4	21182		
I1	INTEGER*4	21358		
IA	INTEGER*4	21306		
IFLG	INTEGER*4	21430		
IT	INTEGER*4	22230		
IW	INTEGER*4	21434		
J	INTEGER*4	21298		
K	INTEGER*4	8348		
KT	INTEGER*4	21274		
LABEL	CHAR*3	8356		
NA	REAL	8020		
NCT	INTEGER*4	21294		
NJ	REAL	8180		
NPTS	INTEGER*4	21190		
ORMAX	REAL	21220		
PHI	REAL*8	21362		
PRMAX	REAL	21216		
PVOL	REAL	22238		
Q	REAL	8012		
QS	REAL*8	21342		
RA	REAL*8	21314		
RAD	REAL	22234		
RAM1	REAL*8	21322		
RDM1	REAL	22706		
RHO	REAL*8	21370		
RHOA	REAL*8	21386		
RHOM1	REAL*8	21378		
RI	REAL	4012		
RJ	REAL*8	21334		

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
RMAX    REAL            21178
S       REAL               12
SFIL    CHAR*10         21194
SQRT                            INTRINSIC
STP     REAL            22698
TAREA   REAL            21270
TITLE   CHAR*40         21224
TVOL    REAL            22222
UL      REAL            21290


    250 c
    251 c
    252 c
    253         Subroutine comrho(Rj,ri,phi,qs,rho)
    254 C*************************************************************************
    255 C***** This routine is designed to compute the value of rho in   ***
    256 C***** Dehoff's solution for a prolate spheriod given a value of ***
    257 C***** Rj, ri, Q, and angle phi.  (phi is in radians)             ***
    258 C*****     QS=Qj      (PROLATE)                 ***
    259 C*****     QS=1.0/Qj  (OBLATE)                  ***
    260 C*************************************************************************
    261         real*8 l,m,n,Rj,ri,phi,qs,rho,dc,ds
    262         dc=dcos(phi)
    263         ds=dsin(phi)
    264         l=(dc*dc)+(qs*qs*ds*ds)
    265         m=(ds*ds)+(qs*qs*dc*dc)
    266         n=(ds*ds*dc*dc)*(1.0-qs*qs)
    267 c
    268 c       Compute Rho
    269 c
    270         rho=(ri*ri)/dsqrt(l)
    271         rho=rho-((((qs)**0.666666666)*Rj*Rj)/l)
    272         rho=(rho*l*l)/(n-(m*l))
    273         if(rho.lt.0.0)then
    274          rho=0.0
    275         else
    276          rho=dsqrt(rho)
    277         endif
    278         return
    279         end

Name     Type         Offset P Class

DC      REAL*8         22714
DCOS                           INTRINSIC
DS      REAL*8         22722
DSIN                           INTRINSIC
DSQRT                          INTRINSIC
L       REAL*8         22730
M       REAL*8         22738
N       REAL*8         22746
PHI     REAL*8             8 *
QS      REAL*8            12 *
RHO     REAL*8            16 *
RI      REAL*8             4 *

RJ      REAL*8             0 *
    280 C
    281 C
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
    282 C
    283         SUBROUTINE SOLVE(N,A,X1,B1)
    284 C*************************************************************************
    285 C***** This routine solves a linear system of equations ( Ax=b)  ***
    286 C***** by using Gauss reduction.  This method does not require   ***
    287 C*****  that the matrix A be symmetric.                           ***
    288 C*************************************************************************
    289         real*8 A(40,40),X(40),B(40)
    290         dimension X1(40),B1(40)
    291         do 0010 i=1,40
1   292         B(I)=DBLE(B1(I))
1   293  0010 CONTINUE
    294 C
    295 C       Triangularize the A matrix
    296 C*
    297         NN=N-1
    298         DO 0100 K=1,NN
1   299         KK=K+1
1   300         DO 0100 I=KK,N
2   301         C=A(I,K)/A(K,K)
2   302         B(I)=B(I)-B(K)*C
2   303         DO 0100 J=K,N
3   304  0100   A(I,J)=A(I,J)-A(K,J)*C
    305 C
    306 C       Solve for x vector (solution vector)
    307 C
    308         X(N)=B(N)/A(N,N)
    309         DO 0300 K=1,NN
1   310         I=N-K
1   311         II=I+1
1   312         C=0.0
1   313         DO 0400 J=II,N
2   314  0400   C=C+A(I,J)*X(J)
1   315         X(I)=(B(I)-C)/A(I,I)
1   316  0300 CONTINUE
    317         DO 0030 I=1,40
1   318  0030 X1(I)=SNGL(X(I))
    319         RETURN
    320         END
```

Name	Type	Offset	P	Class
A	REAL*8	4	*	
B	REAL*8	23074		
B1	REAL	12	*	
C	REAL	23418		
DBLE				INTRINSIC
I	INTEGER*4	23394		
II	INTEGER*4	23434		
J	INTEGER*4	23422		
K	INTEGER*4	23402		
KK	INTEGER*4	23410		
N	INTEGER*4	0	*	
NN	INTEGER*4	23398		
SNGL				INTRINSIC
X	REAL*8	22754		
X1	REAL	8	*	

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

Name	Type	Size	Class
COMRHO			SUBROUTINE
MORPHO			PROGRAM
SOLVE			SUBROUTINE

Pass One No Errors Detected
320 Source Lines

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
      1 $STORAGE: 2
      2 $DEBUG
      3         PROGRAM PZONE
      4 C*************************************************************************
      5 C     This routine is designed to process information regarding
      6 C     the damage zone.  The infomation to be processed is read in
      7 C     the form of a series of normal distribution curves.  These
      8 C     curves are used to produce a 3-D mesh and the corresponding
      9 C     results are then used to compute the centroid of damage as
     10 C     well as the volume of accumulated damage for each interval.
     11 C*************************************************************************
     12 C
     13 C     Start by reading in the input information
     14 C
     15       Real*8 VOL(30),XVOL(30),YVOL(30),R1(30)
     16       COMMON/WORK1/Z(40,40),XP(450),YP(450),ZP(450),FZ(15,5),
     17      1 ZPIJ(600),VERTEX(16),ZLEV(40),YBAR(30),YDEV(30),YCUT(30,2)
     18       COMMON/WORK2/YMAX(30),XCRD(30),KNXT(450),MASK(6000),LDIG(40),
     19      1 LWGT(40)
     20       CHARACTER*10 FILENM,FINP
     21         CHARACTER*30 XLABEL,YLABEL
     22         CHARACTER* 1 XLABEB(30),YLABEB(30)
     23         EQUIVALENCE (XLABEB(1),XLABEL),(YLABEB(1),YLABEL)
     24       DATA XLABEL/'  CIRCUMFERENTIAL DIR (mm-1)  '/
     25       DATA YLABEL/'       AXIAL DIR (mm-1)       '/
     26       NNX=40
     27       NNY=40
     28       NXX=40
     29       NYY=40
     30       ITT=93
     31       THICK=75.0
     32  0006 write(*,*)'  *** Welcome to P-Zone  ***'
     33       write(*,*)'  '
     34       write(*,*)'  Give Location for plotting output'
     35       write(*,*)'             0 -> Monocrome Monitor'
     36       write(*,*)'             1 -> Color Monitor    '
     37       write(*,*)'             2 ->  Printer '
     38       write(*,*)'             3 ->  HP Plotter'
     39       read(*,*,err=0006)iout
     40       if(iout.eq.0)then
     41        ioport=93
     42        model=93
     43       elseif(iout.eq.1)then
     44        ioport=99
     45        model=99
     46       elseif(iout.eq.2)then
     47        ioport=0
     48        model=64
     49       else
     50        ioport=9602
     51        model=20
     52       endif
     53 C
     54 C     Set origin (crack tip) values
     55 C
     56       IN=0
     57  0050 WRITE(*,*)'  INPUT   FILE ->1  TERMINAL ->0'
     58       READ(*,*,ERR=0050)IN
     59       IF(IN.NE.0)THEN
```

```
D Line# 1     7                                  Microsoft FORTRAN77 V3.20 02/84
     60  0056  WRITE(*,*)'  GIVE INPUT FILE NAME'
     61        READ(*,0055,ERR=0056)FINP
     62        OPEN(7,FILE=FINP,STATUS='OLD')
     63        READ(7,*)NPRF
     64        DO 0057 I=1,NPRF
1    65        READ(7,*)YBAR(I),YDEV(I),YMAX(I)
1    66        READ(7,*)YCUT(I,1),YCUT(I,2),XCRD(I)
1    67  0057  CONTINUE
     68        CLOSE(7)
     69        GO TO 0046
     70       ENDIF
     71       YMAX(1)=0.0
     72       YDEV(1)=1.0
     73       YCUT(1,1)=1.0
     74       YCUT(1,2)=2.0
     75       XCRD(1)=0.0
     76  0005 WRITE(*,*)'   GIVE CRACK LOCATION  '
     77       READ(*,*,ERR=0005)YBAR(1)
     78  0010 WRITE(*,*)'   ALIGN MEAN VALUES ?   YES ->1   NO ->0'
     79       READ(*,*,ERR=0010)IMOVE
     80 C
     81 C     Read in remaining input
     82 C
     83  0015 WRITE(*,*)'  GIVE  TOTAL # OF PROFILES TO INPUT'
     84       READ(*,*,ERR=0015)NPRF
     85       NPRF=NPRF+1
     86  0017 WRITE(*,*)'  GIVE SEARCH LIMITS   (lower,   upper)'
     87       READ(*,*,ERR=0017)SL,SU
     88 C
     89 C     Start input loop here
     90 C
     91       DO 0022 I=2,NPRF
1    92       I1=I-1
1    93  0025 WRITE(*,0030)I1
1    94  0030 FORMAT(//'  GIVE  MEAN,  SDEV,  MAX,  FOR ',I3,'th PROFILE')
1    95       READ(*,*,ERR=0025)YBAR(I),YDEV(I),YMAX(I)
1    96  0035 WRITE(*,*)'  GIVE CUTOFF LIMITS    (lower,   upper)'
1    97       DUM1=0.0
1    98       DUM2=0.0
1    99       READ(*,*,ERR=0035)DUM1,DUM2
1   100 C
1   101       IF(DUM1.NE.0.0)THEN
1   102        YCUT(I,2)=DUM1
1   103       ELSE
1   104        YCUT(I,2)=SL
1   105       ENDIF
1   106 C
1   107       IF(DUM2.NE.0.0)THEN
1   108        YCUT(I,1)=DUM2
1   109       ELSE
1   110        YCUT(I,1)=SU
1   111       ENDIF
1   112 C
1   113       IF(IMOVE.EQ.1)THEN
1   114        YCUT(I,1)=YCUT(I,1)-YBAR(I)
1   115        YCUT(I,2)=YCUT(I,2)-YBAR(I)
1   116        YBAR(I)=0.0
1   117       ENDIF
1   118 C
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1   119 C
1   120  0033 WRITE(*,*)'  GIVE X-COORDINATE FOR PROFILE '
1   121       READ(*,*,ERR=0033)XCRD(I)
1   122  0022 CONTINUE
    123       IN=0
    124  0058 WRITE(*,*) '  STORE INPUT DATA IN FILE  YES ->1  NO ->0'
    125       READ(*,*,ERR=0058)IN
    126       IF(IN.NE.0)THEN
    127  0059  WRITE(*,*)'  GIVE FILE NAME'
    128        READ(*,0055,ERR=0059)FINP
    129  0055  FORMAT(A10)
    130        OPEN(7,FILE=FINP,STATUS='NEW')
    131        WRITE(7,*)NPRF
    132        DO 0060 I=1,NPRF
1   133        WRITE(7,*)YBAR(I),YDEV(I),YMAX(I)
1   134        WRITE(7,*)YCUT(I,1),YCUT(I,2),XCRD(I)
1   135  0060  CONTINUE
    136        CLOSE(7)
    137       ENDIF
    138  0046 CONTINUE
    139  0070 WRITE(*,*)'     Give the # of grid increments'
    140       READ(*,*,ERR=0070)ITEMP
    141       XLOL=1.0
    142       YLOL=1.0
    143 C     ****************************************************************
    144 C     Start the big loop for all computations
    145 C     ****************************************************************
    146       OPEN(12,FILE='TABLE.DAT',STATUS='NEW')
    147       DO 0080 I1=1,ITEMP
1   148       IF(I1.EQ.1)THEN
1   149        IPROJ=0
1   150       ELSE
1   151        IPROJ=1
1   152       ENDIF
1   153 C
1   154 C     Read in the information for the overlay grid
1   155 C
1   156       WRITE(*,*)'     ----- Overlay Grid Information -----'
1   157       WRITE(*,*)' '
1   158  0083 WRITE(*,*)'  Terminal ->0     File ->1'
1   159       READ(*,*,ERR=0083)IFT
1   160       IF(IFT.GT.0)THEN
1   161        WRITE(*,*)'     Give File Name'
1   162        READ(*,0055)FINP
1   163        OPEN(7,FILE=FINP,STATUS='OLD')
1   164        READ(7,*)NTX,NTY,XMAX
1   165        DO 0081 J=1,NTY
2   166        DO 0082 I=1,NTX
3   167  0082  READ(7,*)FZ(I,J)
2   168  0081  CONTINUE
1   169        CLOSE(7)
1   170       ELSE
1   171  0090   WRITE(*,*)'  Give the refine res Xno.,  Yno. and the Xlimit'
1   172         READ(*,*,ERR=0090)NTX,NTY,XMAX
1   173         DO 0100 J=1,NTY
2   174         DO 0110 I=1,NTX

3   175  0105 WRITE(*,*)' Xno=',I,' Yno=',J,'  Give block height fact (0->1)'
3   176         READ(*,*,ERR=0105)FZ(I,J)
3   177  0110   CONTINUE
```

```
D Line# 1     7                                  Microsoft FORTRAN77 V3.20 02/84
2   178  0100   CONTINUE
1   179  0091   WRITE(*,*)'     Store Template Give ->1'
1   180         READ(*,*,ERR=0091)IN
1   181         IF(IN.NE.0)THEN
1   182        WRITE(*,*)'  GIVE FILE NAME'
1   183        READ(*,0055)FINP
1   184        OPEN(7,FILE=FINP,STATUS='NEW')
1   185        WRITE(7,*)NTX,NTY,XMAX
1   186        DO 0092 J=1,NTY
2   187        DO 0093 I=1,NTX
3   188  0093  WRITE(7,*)FZ(I,J)
2   189  0092  CONTINUE
1   190        CLOSE(7)
1   191         ENDIF
1   192       ENDIF
1   193 C
1   194 C     1....  Compute the XP, YP, and the ZP arrays
1   195 C
1   196       YDMAX=0.0
1   197       DO 0125 I=1,NPRF
2   198       IF(YDEV(I).GT.YDMAX)THEN
2   199        YDMAX=YDEV(I)
2   200        YBMAX=YBAR(I)
2   201        YL=YBAR(I)-(3.0*YDEV(I))
2   202        YU=YBAR(I)+(3.0*YDEV(I))
2   203       ENDIF
2   204  0125 CONTINUE
1   205       DELY=(YU-YL)/20.0
1   206       XL=XCRD(1)
1   207       XU=XCRD(NPRF)
1   208  0126 WRITE(*,*)'  XL=',XL,'YL=',YL,'XU=',XU,'YU=',YU
1   209       WRITE(*,*)'  TO CHANGE DEFAULTS GIVE  ->1'
1   210       READ(*,*,ERR=0126)IC
1   211       IF(IC.NE.0)THEN
1   212        WRITE(*,*)'  GIVE XL,YL,XU,YU'
1   213        READ(*,*,ERR=0126)XL,YL,XU,YU
1   214       ENDIF
1   215       TDX=(XMAX-XL)/FLOAT(NTX)
1   216       TDY=(YU-YL)/FLOAT(NTY)
1   217       NP=0
1   218 C
1   219       DO 0120 I=1,NPRF
2   220       XVAL=XCRD(I)
2   221       IF(XVAL.LT.XMAX)THEN
2   222        DO 0130 J=1,20
3   223        NP=NP+1
3   224        YVAL=YL+(DELY*(FLOAT(J-1)))
3   225        EX=(YVAL-YBAR(I))/YDEV(I)
3   226          EX=-0.5*(EX*EX)
3   227        IF(EX.LT.-15.)THEN
3   228         HI=0.0
3   229        ELSE
3   230            HI=EXP(EX)*YMAX(I)
3   231           ENDIF
3   232 C
3   233 C     Compute the proper template Value
3   234 C
3   235        DO 0160 J1=1,NTY
4   236        YTVM=YL+(TDY*(FLOAT(J1-1)))
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
4    237         YTVP=YL+(TDY*(FLOAT(J1)))
4    238         DO 0170 II1=1,NTX
5    239         XTVM=XL+(TDX*(FLOAT(II1-1)))
5    240         XTVP=XL+(TDX*(FLOAT(II1)))
5    241         IF(XVAL.GE.XTVM.AND.XVAL.LE.XTVP)ICHK=I1
5    242  0170   CONTINUE
4    243         IF(YVAL.GE.YTVM.AND.YVAL.LE.XTVP)JCHK=J1
4    244  0160   CONTINUE
3    245         FCT=FZ(ICHK,JCHK)
3    246         ZP(NP)=HI*FCT
3    247         XP(NP)=XVAL
3    248         YP(NP)=YVAL
3    249  0130   CONTINUE
2    250        ELSE
2    251         DO 0140 J=1,20
3    252         NP=NP+1
3    253         YVAL=YL+(DELY*(FLOAT(J-1)))
3    254         ZP(NP)=0.0
3    255         XP(NP)=XMAX
3    256         YP(NP)=YVAL
3    257  0140   CONTINUE
2    258        ENDIF
2    259  0120  CONTINUE
1    260           DO 0200 I=1,NNX
2    261           DO 0200 J=1,NNY
3    262           Z(I,J)=0.0
3    263  0200     CONTINUE
1    264        XST=0.0
1    265        YST=0.0
1    266 C      YST=YBMAX-(3.0*YDMAX)
1    267 C      NYST=INT((YST/50.0)+.5)
1    268 C      YST=FLOAT(NYST*50)
1    269        DELTA=100.0
1    270 C
1    271        XLENG=XCRD(NPRF)/100.0
1    272        YLENG=5.0*YDMAX/100.0
1    273        FACT=6.0/XLENG
1    274 C      NSM=2
1    275        CAY=5.
1    276        NDIV=2
1    277        IDIR=3
1    278        IEDGE=0
1    279        IFRAME=1
1    280        ZLOW=1.0E35
1    281        ICUT=1
1    282        ITRIM=1
1    283        NRNG=10
1    284         XUPR = XLENG + XLOL
1    285         YUPR = YLENG + YLOL
1    286        DX=(XU-XL)/FLOAT(NXX-1)
1    287        DY=(YU-YL)/FLOAT(NYY-1)
1    288        WRITE(*,*)'      Generating the Grid'
1    289        CALL ZGRID(Z,NNX,NNY,NXX,NYY,XL,YL,XU,YU,XP,YP,ZP,NP,CAY,NRNG,
1    290       1 ZPIJ,XNXT)
1    291 C      WRITE(*,*)'      Smoothing the Grid'
1    292 C      CALL ZSMTH(Z,NNX,NNY,NXX,NYY,NSM)

1    293 C
1    294 C      Compute the Volume, and the Centriod of the P-Zone
1    295 C
```

```
1   296         WRITE(*,*)'        Computing the Volume and Centroid'
1   297         VOL(I1)=0.0
1   298         XVOL(I1)=0.0
1   299         YVOL(I1)=0.0
1   300         R1(I1)=0.0
1   301         ZMAX=0.0
1   302         AREA=DX*DY
1   303 C
1   304         DO 0220 J=1,NYY
2   305         YVAL=YL+(DY*(FLOAT(J-1)))
2   306         DO 0230 I=1,NXX
3   307         IF(Z(I,J).LT.0.0)Z(I,J)=0.0
3   308         IF(Z(I,J).GT.35.0)Z(I,J)=0.0
3   309         IF(Z(I,(J+1)).LT.0.0)Z(I,(J+1))=0.0
3   310         IF(Z(I,(J+1)).GT.35.0)Z(I,(J+1))=0.0
3   311         IF(Z((I+1),J).LT.0.0)Z((I+1),J)=0.0
3   312         IF(Z((I+1),J).GT.35.0)Z((I+1),J)=0.0
3   313         IF(Z((I+1),(J+1)).LT.0.0)Z((I+1),(J+1))=0.0
3   314         IF(Z((I+1),(J+1)).GT.35.0)Z((I+1),(J+1))=0.0
3   315         XVAL=XL+(DX*(FLOAT(I-1)))
3   316         Z1=Z(I,J)
3   317         Z2=Z((I+1),J)
3   318         Z3=Z(I,(J+1))
3   319         Z4=Z((I+1),(J+1))
3   320 C
3   321 C       Compute the Volume, and Centroid
3   322 C
3   323         HGHT=0.25*(Z1+Z2+Z3+Z4)
3   324         IF(HGHT.LE.0.01)GO TO 0230
3   325         VOL(I1)=VOL(I1)+(AREA*HGHT)
3   326         XVOL(I1)=XVOL(I1)+(XVAL+(DX/2.0))*(AREA*HGHT)
3   327         YVOL(I1)=YVOL(I1)+(YVAL+(DY/2.0))*(AREA*HGHT)
3   328  0230 CONTINUE
2   329  0220 CONTINUE
1   330         XVOL(I1)=XVOL(I1)/(VOL(I1)*10.)
1   331         YVOL(I1)=YVOL(I1)/(VOL(I1)*10.)
1   332         VOL(I1)=VOL(I1)/1000.
1   333 C
1   334         write(*,*)'  I1=',I1
1   335         IF(I1.GT.1)THEN
1   336          DELXC=10.0*(XVOL(I1-1)-XVOL(I1))
1   337          IXC=(DELXC/(XU-XL))*NXX+1
1   338          WRITE(*,*)'  Shifted by ',IXC,'  Boxes'
1   339          DO 0236 J=1,NYY
2   340          DO 0237 I=IXC,NXX
3   341          IO=I-IXC+1
3   342          ZO1=Z(IO,J)
3   343          ZO2=Z((IO+1),J)
3   344          ZO3=Z(IO,(J+1))
3   345          ZO4=Z((IO+1),(J+1))
3   346          ZN1=Z(I,J)
3   347          ZN2=Z((I+1),J)
3   348          ZN3=Z(I,(J+1))
3   349          ZN4=Z((I+1),(J+1))
3   350          HOLD=0.25*(ZO1+ZO2+ZO3+ZO4)
3   351          HNEW=0.25*(ZN1+ZN2+ZN3+ZN4)
3   352          DELH=ABS(HOLD-HNEW)
3   353          R1(I1)=R1(I1)+DELH
3   354  0237    CONTINUE
```

```
D Line# 1     7                                   Microsoft FORTRAN77 V3.20 02/84
2   355  0236  CONTINUE
1   356        R1(I1)=(R1(I1)*AREA)/DELXC
1   357       ELSE
1   358        R1(I1)=0.0
1   359       ENDIF
1   360       R1(I1)=R1(I1)/100.0
1   361 C
1   362       WRITE(*,*)'  '
1   363       WRITE(*,*)'     LSD Volume = ',VOL(I1),'mm3'
1   364       WRITE(*,*)'  '
1   365       WRITE(*,*)'   Xcen = ',XVOL(I1),'mm      Ycen = ',YVOL(I1),'mm'
1   366       WRITE(*,*)'  '
1   367       WRITE(*,*)'              R1 =',R1(I1),'mm2'
1   368       WRITE(*,*)'  '
1   369       WRITE(12,*)'  '
1   370       WRITE(12,*)'     LSD Volume = ',VOL(I1),'mm3'
1   371       WRITE(12,*)'  '
1   372       WRITE(12,*)'   Xcen = ',XVOL(I1),'mm      Ycen = ',YVOL(I1),'mm'
1   373       WRITE(12,*)'  '
1   374       WRITE(12,*)'              R1=',R1(I1),'mm2'
1   375       WRITE(12,*)'  '
1   376 C
1   377  0280 WRITE(*,*)'  '
1   378       WRITE(*,*)'              3-D PLOT'
1   379       WRITE(*,*)'       No Plot          -> 0'
1   380       WRITE(*,*)'       For 3-D Plot     -> 1'
1   381       WRITE(*,*)'  '
1   382       READ(*,*,ERR=0280)IPL
1   383       IF(IPL.EQ.0)GO TO 0080
1   384  0250 WRITE(*,*)'   Give Horiz, and Vert Angles'
1   385       READ(*,*,ERR=0250)PHI,THETA
1   386        CALL PLOTS(0,IOPORT,MODEL)
1   387       CALL COLOR(0,IERR)
1   388        CALL COMPLX
1   389        CALL FACTOR(FACT)
1   390       CALL MESHS(Z, NNX,NNY, NXX, NYY, PHI, THETA, XLOL, YLOL,
1   391      1  XUPR, YUPR, NDIV, IEDGE, IDIR, IPROJ, IFRAME, ZLOW,
1   392      2  ICUT, ITRIM, MASK, VERTEX)
1   393 C
1   394 C     Provide labeling
1   395 C
1   396       CALL CVT(0.,0.,0.,XR,YR,XL,YL,DX,DY)
1   397       X1=XR
1   398       Y1=YR
1   399       XA=XCRD(NPRF)
1   400       CALL CVT(XA,0.,0.,XR,YR,XL,YL,DX,DY)
1   401       X2=XR
1   402       Y2=YR
1   403       VANG=ATAN((Y1-Y2)/(X1-X2))*(180/3.14159)
1   404       WRITE(*,*)'   VERT ANG=',VANG,' degrees'
1   405       IF(VANG.LT.0)THEN
1   406        XA=XCRD(NPRF)-25.0
1   407        YA=YU+50.
1   408        XN=-60.
1   409        YN=YBAR(1)
1   410       ELSE
1   411        XA=25.0
1   412        YA=YL-50.
1   413        XN=-200.
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1   414         YN=YBAR(1)
1   415        ENDIF
1   416        CALL CVT(XN,YN,0.,XR,YR,XL,YL,DX,DY)
1   417        CALL SYMBOL(XR,YR,.125,'NOTCH',VANG,5)
1   418        CALL CVT(XA,YA,0.,XR,YR,XL,YL,DX,DY)
1   419        CALL SYMBOL(XR,YR,.125,'CIRCUMFERENTIAL DIR',VANG,19)
1   420        CALL PLOT(0.0,0.0,999)
1   421  0270  WRITE(*,*)' '
1   422        WRITE(*,*)'       CONTOUR MAP'
1   423        WRITE(*,*)'     No Plot      -> 0'
1   424        WRITE(*,*)'     Contour Plot -> 1'
1   425        WRITE(*,*)' '
1   426        READ(*,*,ERR=0270)IPL
1   427        if(IPL.eq.0)go to 0080
1   428  0260  CONTINUE
1   429 C
1   430 C      Draw the contour map next.
1   431 C
1   432        WRITE(*,*)' '
1   433  0290  WRITE(*,*)'     The max height is ',ZMAX
1   434        WRITE(*,*)' '
1   435        WRITE(*,*)'    Enter Min and Max for Contour range'
1   436        READ(*,*,ERR=0290)ZMN,ZMX
1   437        WRITE(*,*)' Enter No. Cntrs, No. per Label, Istart'
1   438        READ(*,*,ERR=0290)NLEV,NLAB,ICK
1   439        ZINC=(ZMX-ZMN)/FLOAT(NLEV)
1   440        DO 0300 I=1,NLEV
2   441        ZLEV(I)=ZMN+ZINC*FLOAT(I-1)
2   442        IF(ICK.EQ.NLAB)THEN
2   443         LWGT(I)=2
2   444         LDIG(I)=-1
2   445         ICK=1
2   446        ELSE
2   447         LWGT(I)=1
2   448         LDIG(I)=-2
2   449         ICK=ICK+1
2   450        ENDIF
2   451  0300  CONTINUE
1   452        NDIV=4
1   453        NARC=4
1   454         ZMAXLN=AMAX1(XLENG,YLENG)
1   455         HGT = ZMAXLN/80.
1   456         CALL PLOTS(0,IOPORT,MODEL)
1   457         CALL FACTOR(FACT)
1   458 C
1   459 C      Scale the X and Y axis of the plot
1   460 C
1   461 C
1   462 C        CALL SCALE(XP,XLENG,NP, 1)
1   463 C        CALL SCALE(YP,YLENG,NP, 1)
1   464 C
1   465 C
1   466         CALL ZCSEG(Z,NNX,NNY,NXX,NYY,XLOL,YLOL,XUPR,YUPR,ZLEV,LDIG,
1   467       1            LWGT,NLEV,HGT,NDIV,NARC)
1   468         CALL NEWPEN(0)
1   469         CALL STAXIS(.15,.15,.15,.15,-1)

1   470        CALL AXIS(XLOL,YLOL,XLABEB,-30,-XLENG,0.0,
1   471       1          XST,DELTA)
1   472         CALL AXIS(XLOL,YLOL,YLABEB,30,-YLENG, 90.0,
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
1   473        1          YST,DELTA)
1   474 C
1   475 C      PLACE TIC ON OPPOSITE AXES
1   476 C
1   477        CALL AXIS(XLOL,YUPR,' ',0,-XLENG,00.0,
1   478        1            XST,DELTA)
1   479        CALL AXIS(XUPR,YLOL,' ',-1,-YLENG,90.0,
1   480        1            YST,DELTA)
1   481 C
1   482 C      Begin graphics output.
1   483 C
1   484        CALL PLOT(0.0,0.0,999)
1   485   0080 CONTINUE
    486        CLOSE(12)
    487 C
    488 C      Write out the centroid data to file
    489 C
    490        OPEN(8,FILE='TAB.OUT',STATUS='NEW')
    491        WRITE(8,0410)
    492 0410   FORMAT('          DAMAGE ZONE  DEVELOPMENT TABLE',//
    493       1 ' File          Volume            Xcen          Ycen',/
    494       2 '               (mm3)             (mm)          (mm)')
    495        DO 0400 I=1,ITEMP
1   496        IF(I.EQ.ITEMP)THEN
1   497         J=1
1   498        ELSE
1   499         J=I+1
1   500        ENDIF
1   501        WRITE(8,0420)J,VOL(J),XVOL(J),YVOL(J)
1   502   0420 FORMAT(I5,3G15.3)
1   503   0400 CONTINUE
    504        CLOSE(8)
    505    999 CONTINUE
    506        END
```

Name	Type	Offset	P	Class
ABS				INTRINSIC
AMAX1				INTRINSIC
AREA	REAL	1424		
ATAN				INTRINSIC
CAY	REAL	1382		
DELH	REAL	1520		
DELTA	REAL	1366		
DELXC	REAL	1460		
DELY	REAL	1260		
DUM1	REAL	1152		
DUM2	REAL	1156		
DX	REAL	1412		
DY	REAL	1416		
EX	REAL	1298		
EXP				INTRINSIC
FACT	REAL	1378		
FCT	REAL	1342		
FILENM	CHAR*10	*****		
FINP	CHAR*10	1058		
FLOAT				INTRINSIC
FZ	REAL	11800		/WORK1 /
HGHT	REAL	1456		

D Line# 1 7

HGT	REAL	1610	
HI	REAL	1302	
HNEW	REAL	1516	
HOLD	REAL	1512	
I	INTEGER*2	1070	
I1	INTEGER*2	1094	
IC	INTEGER*2	1272	
ICHK	INTEGER*2	1338	
ICK	INTEGER*2	1592	
ICUT	INTEGER*2	1398	
IDIR	INTEGER*2	1388	
IEDGE	INTEGER*2	1390	
IERR	INTEGER*2	1534	
IFRAME	INTEGER*2	1392	
IFT	INTEGER*2	1190	
II1	INTEGER*2	1322	
IMOVE	INTEGER*2	1078	
IN	INTEGER*2	1056	
IO	INTEGER*2	1478	
IOPORT	INTEGER*2	1052	
IOUT	INTEGER*2	1050	
IPL	INTEGER*2	1524	
IPROJ	INTEGER*2	1188	
ITEMP	INTEGER*2	1172	
ITRIM	INTEGER*2	1400	
ITT	INTEGER*2	1044	
IXC	INTEGER*2	1464	
J	INTEGER*2	1200	
J1	INTEGER*2	1306	
JCHK	INTEGER*2	1340	
KNXT	INTEGER*2	240	/WORK2 /
LDIG	INTEGER*2	13140	/WORK2 /
LWGT	INTEGER*2	13220	/WORK2 /
MASK	INTEGER*2	1140	/WORK2 /
MODEL	INTEGER*2	1054	
NARC	INTEGER*2	1604	
NDIV	INTEGER*2	1386	
NLAB	INTEGER*2	1590	
NLEV	INTEGER*2	1588	
NNX	INTEGER*2	1036	
NNY	INTEGER*2	1038	
NP	INTEGER*2	1282	
NPRF	INTEGER*2	1068	
NRNG	INTEGER*2	1402	
NTX	INTEGER*2	1192	
NTY	INTEGER*2	1194	
NXX	INTEGER*2	1040	
NYY	INTEGER*2	1042	
PHI	REAL	1526	
R1	REAL*8	796	
SL	REAL	1080	
SU	REAL	1084	
TDX	REAL	1274	
TDY	REAL	1278	
THETA	REAL	1530	
THICK	REAL	1046	
VANG	REAL	1564	
VERTEX	REAL	14500	/WORK1 /
VOL	REAL*8	16	

D Line# 1 7

X1	REAL	1544	
X2	REAL	1556	
XA	REAL	1552	
XCRD	REAL	120	/WORK2 /
XL	REAL	1264	
XLABEB	CHAR*1	766	
XLABEL	CHAR*30	766	
XLENG	REAL	1370	
XLOL	REAL	1174	
XMAX	REAL	1196	
XN	REAL	1572	
XP	REAL	6400	/WORK1 /
XR	REAL	1536	
XST	REAL	1358	
XTVM	REAL	1330	
XTVP	REAL	1334	
XU	REAL	1268	
XUPR	REAL	1404	
XVAL	REAL	1290	
XVOL	REAL*8	256	
Y1	REAL	1548	
Y2	REAL	1560	
YA	REAL	1568	
YBAR	REAL	14724	/WORK1 /
YBMAX	REAL	1248	
YCUT	REAL	14964	/WORK1 /
YDEV	REAL	14844	/WORK1 /
YDMAX	REAL	1238	
YL	REAL	1252	
YLABEB	CHAR*1	736	
YLABEL	CHAR*30	736	
YLENG	REAL	1374	
YLOL	REAL	1178	
YMAX	REAL	0	/WORK2 /
YN	REAL	1576	
YP	REAL	8200	/WORK1 /
YR	REAL	1540	
YST	REAL	1362	
YTVM	REAL	1314	
YTVP	REAL	1318	
YU	REAL	1256	
YUPR	REAL	1408	
YVAL	REAL	1294	
YVOL	REAL*8	496	
Z	REAL	0	/WORK1 /
Z1	REAL	1440	
Z2	REAL	1444	
Z3	REAL	1448	
Z4	REAL	1452	
ZINC	REAL	1594	
ZLEV	REAL	14564	/WORK1 /
ZLOW	REAL	1394	
ZMAX	REAL	1420	
ZMAXLN	REAL	1606	
ZMN	REAL	1580	
ZMX	REAL	1584	
ZN1	REAL	1496	
ZN2	REAL	1500	
ZN3	REAL	1504	

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
ZN4     REAL          1508
ZO1     REAL          1480
ZO2     REAL          1484
ZO3     REAL          1488
ZO4     REAL          1492
ZP      REAL         10000   /WORK1 /
ZPLJ    REAL         12100   /WORK1 /


    507 C
    508 C
    509         SUBROUTINE CVT(X,Y,Z, XR, YR,XL,YL,DX,DY)
    510 C...    TRANSLATE WORLD COORDINATES TO GRID UNITS.
    511         XP = 1.0 + (X - XL) /DX
    512         YP = 1.0 + (Y - YL) /DY
    513 C...    CONVERT WORLD COORDINATES TO PLOTTER COORDINATES.
    514         CALL P3D2D(XP,YP,Z, XR, YR)
    515         RETURN
    516         END

Name    Type         Offset P Class

DX      REAL            28 *
DY      REAL            32 *
X       REAL             0 *
XL      REAL            20 *
XP      REAL          1812
XR      REAL            12 *
Y       REAL             4 *
YL      REAL            24 *
YP      REAL          1816
YR      REAL            16 *
Z       REAL             8 *


    517 C

Name    Type          Size   Class

AXIS                       SUBROUTINE
COLOR                      SUBROUTINE
COMPLX                     SUBROUTINE
CVT                        SUBROUTINE
FACTOR                     SUBROUTINE
MESHS                      SUBROUTINE
NEWPEN                     SUBROUTINE
P3D2D                      SUBROUTINE
PLOT                       SUBROUTINE
PLOTS                      SUBROUTINE
PZONE                      PROGRAM
STAXIS                     SUBROUTINE
SYMBOL                     SUBROUTINE
WORK1                15204 COMMON
WORK2                13300 COMMON
ZCSEG                      SUBROUTINE
ZGRID                      SUBROUTINE
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.20 02/84
 Pass One      No Errors Detected
               517 Source Lines
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.20 02/84
      1 $Storage: 2
      2 $Debug
      3         Program Energy
      4 c*************************************************************************
      5 c***** This program is designed to compute the near and far field   *
      6 c***** energy release rates using a complex potential formulation.  *
      7 c***** The near field is evaluated via the stress intensity factors *
      8 c***** and the far field is computed through a numerical integration*
      9 c*************************************************************************
     10 c
     11       Complex*8 z,zst,znd,zinc,zc,z01(200),z02(200),
     12      1 a(200),dum1,dum,dum2,zs,ze,dz,an,Ktot,K12,Jc,Jc1,ft3
     13       Real*4 k,sig(30,30,3),emod,nu,zp(30,30),zlev(50),vertex(16),
     14      1 Jnear,Jfar,Lc
     15       Integer*2 ldig(50),lwgt(50),mask(3000)
     16       Character*10 mfil
     17       Common /mat/ emod,gmod,nu,k
     18       Common /dam/z01,z02,a,iz
     19       Common /crack/ c
     20       Common/remote/sigyy,sigxy
     21       Data nnx,nny,xlow,ylow /30,30,1.0,1.0/
     22 c
     23       pi=3.141592654
     24       write(*,*)'    Give material properties'
     25  0005 write(*,*)'   emod,nu,  0-> pl. stress  1-> pl. strain'
     26       read(*,*,err=0005)emod,nu,ipl
     27       if(ipl.eq.0)then
     28        k=(3.0-nu)/(1.0+nu)
     29       else
     30        k=3.0-4.0*nu
     31       endif
     32       gmod=emod/(2.0*(1.0+nu))
     33  0010 write(*,*)'    Give main crack length'
     34       read(*,*,err=0010)c
     35  0007 write(*,*)'   Give Centroid Distance (mm)'
     36       read(*,*,err=0007)xcen
     37       c=c+xcen
     38       write(*,*)'  '
     39       write(*,*)'   Give File Name for Microcrack Data'
     40       read(*,0011)mfil
     41  0011 format(a10)
     42 c
     43       open(5,file=mfil,status='old')
     44       write(*,*)'   '
     45       write(*,*)'  --- Reading the Microcrack Data --- '
     46 c
     47 c     nmic=# of microcracks   ndis=# of dislocations/micro
     48 c
     49       read(5,*)nmic,ndis
     50       if(ndis.le.0.or.ndis.gt.10)ndis=1
     51       ic=0
     52       dum=cmplx(0,1)
     53       do 0012 i=1,nmic
1    54 c
1    55 c     Next read coords of the microcrack (xs,ys) to (xe,ye)
1    56 c
1    57       write(*,*)'  Currently reading the ',i,'th data'
1    58       read(5,*)xs,ys,xe,ye
1    59 c     ************************************************************
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1    60        xe=xe-xcen
1    61        xs=xs-xcen
1    62 c      ********************************************************
1    63        den=xe-xs
1    64        if(den.lt.0.000001)then
1    65         theta=pi/2.0
1    66        else
1    67         theta=atan((ye-ys)/den)
1    68        endif
1    69        xd=cos(theta)
1    70        yd=sin(theta)
1    71        dum1=cmplx(xd,yd)
1    72 c
1    73 c      Give  the CSD, and the COD'
1    74 c
1    75        read(5,*)b1,b2
1    76        b1=b1/float(ndis)
1    77        b2=b2/float(ndis)
1    78        xdis=(xe-xs)/(2.0*float(ndis+1))
1    79        ydis=(ye-ys)/(2.0*float(ndis+1))
1    80 c
1    81        do 0013 j=1,ndis
2    82        ic=ic+1
2    83        xsc=xs+(xdis*float(j))
2    84        ysc=ys+(ydis*float(j))
2    85        xec=xe-(xdis*float(j))
2    86        yec=ye-(ydis*float(j))
2    87        z01(ic)=cmplx(xsc,ysc)
2    88        z02(ic)=cmplx(xec,yec)
2    89        a(ic)=cmplx(b1,b2)
2    90        a(ic)=(qmod/(pi*(k+1)))*dum1*a(ic)
2    91        a(ic)=a(ic)/dum
2    92  0013 continue
1    93  0012 continue
     94        close(5)
     95 c
     96 c      Start computation of the Near Field Energy Release Rates
     97 c
     98        write(*,*)'   Total # dipoles =',ic
     99        Ktot=cmplx(0.0,0.0)
    100        Do 0030 i=1,ic
1   101        an=a(i)
1   102        ze=z01(i)
1   103        dum=csqrt((ze+c)/ze)
1   104        dum=-1.0*an*(real(dum)-1.0)
1   105        dum2=csqrt(conjg(ze))
1   106        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+c)
1   107        dum1=(conjg(an)*aimag(ze)*c)/dum2
1   108        dum2=cmplx(0.0,1.0)
1   109        dum1=dum1*dum2
1   110        K12=-2.0*sqrt((2*pi)/c)*(dum+dum1)
1   111        an=-1.0*a(i)
1   112        ze=z02(i)
1   113        dum=csqrt((ze+c)/ze)
1   114        dum=-1.0*an*(real(dum)-1.0)
1   115        dum2=csqrt(conjg(ze))

1   116        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+c)
1   117        dum1=(conjg(an)*aimag(ze)*c)/dum2
1   118        dum2=cmplx(0.0,1.0)
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1   119        dum1=dum1*dum2
1   120        K12=K12-(2.0*sqrt((2*pi)/c)*(dum+dum1))
1   121        Ktot=Ktot+K12
1   122   0030 continue
    123 c
    124 c      Convert Stress Intensity Factors to Energy Release Rates
    125 c
    126   0038 write(*,*)'    Give Remote Loading Stress'
    127        write(*,*)'              sig-yy, sig-xy '
    128        read(*,*,err=0038)sigyy,sigxy
    129        rKI=real(Ktot)
    130        rKII=-1.0*aimag(Ktot)
    131        rKI=rKI+(sigyy*sqrt(pi*0.5*c))
    132        if(rKI.lt.0.0)rKI=0.0
    133        rKII=rKII+(sigxy*sqrt(pi*0.5*c))
    134        write(*,*)'    The stress intensity factors are :'
    135        write(*,*)'              KI  =',rKI
    136        write(*,*)'              KII =',rKII
    137        Jnear=(rKI*rKI)+(rKII*rKII)
    138        if(ipl.gt.0)then
    139         Jnear=Jnear/Emod
    140        else
    141         Jnear=((1.0-nu*nu)/Emod)*Jnear
    142        endif
    143        write(*,*)'  The near field is '
    144        write(*,*)'     Jnear=',Jnear
    145 c
    146 c      Compute the Far Field Energy Release Rates via Budiansky
    147 c      and Rice Formulation
    148 c
    149   0035 write(*,*)'    Give Contour Size'
    150        write(*,*)'    Length,Width'
    151        read(*,*,err=0035)Lc,Wc
    152        write(*,*)'  '
    153 c
    154 c      Integrate over a rectilinear path
    155 c
    156        Jc=cmplx(0.0,0.0)
    157        Do 0040 i=1,5
1   158        if(i.eq.1)then
1   159         a1=-0.2*c
1   160         b=-0.5*Wc
1   161         zst=cmplx(a1,0.0)
1   162         znd=cmplx(a1,b)
1   163         write(*,*)'  '
1   164         write(*,*)'   Currently Integrating over 1st part'
1   165        elseif(i.eq.2)then
1   166         zst=cmplx(a1,b)
1   167         znd=cmplx(Lc,b)
1   168         write(*,*)'  '
1   169         write(*,*)'   Currently Integrating over 2nd part'
1   170        elseif(i.eq.3)then
1   171         zst=cmplx(Lc,b)
1   172         b=0.5*Wc
1   173         znd=cmplx(Lc,b)
1   174         write(*,*)'  '

1   175         write(*,*)'   Currently Integrating over 3rd part'
1   176        elseif(i.eq.4)then
1   177         zst=cmplx(Lc,b)
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1    178        znd=cmplx(a1,b)
1    179        write(*,*)'  '
1    180        write(*,*)'   Currently Integrating over 4th part'
1    181       else
1    182        zst=cmplx(a1,b)
1    183        znd=cmplx(a1,0.0)
1    184        write(*,*)'  '
1    185        write(*,*)'   Currently Integrating over 5th part'
1    186       endif
1    187 c
1    188 c
1    189       call Qromb(ic,zst,znd,Jc1)
1    190       Jc=Jc+Jc1
1    191  0040 continue
     192 c
     193 c     Adding on extra terms caused by the open ended integration
     194 c     path should be done at this point.
     195 c
     196       Jfar=aimag(Jc)
     197       if(ipl.eq.0)then
     198        Jfar=(2.0/Emod)*Jfar
     199       else
     200        Jfar=2.0*((1.0-nu*nu)/Emod)*Jfar
     201       endif
     202       open(5,file='j.out',status='new')
     203       write(5,*)'     Crack Input File:'
     204       write(5,0050)mf'l
     205  0050 format(15x,a10)
     206       write(5,*)'     Crack Length =',c
     207       write(5,*)'  '
     208       write(5,*)'     Contour Dimensions:'
     209       write(5,*)'       Lc=',Lc,'  Wc=',Wc
     210       write(5,*)'  '
     211       write(5,*)'      Material Properties:'
     212       write(5,*)'       Emod=',Emod,'  nu=',nu
     213       if(ipl.eq.0)then
     214        write(5,*)'       Plane Stress'
     215       else
     216        write(5,*)'       Plane Strain'
     217       endif
     218       write(5,*)'  '
     219       write(5,*)'     Stress Intensity Factors:'
     220       Write(5,*)'     KI=',rKI,'     KII=',rKII
     221       write(5,*)'  '
     222       write(5,*)'      Energy Release Rates:'
     223       write(5,*)'      Jnear= ',Jnear
     224       write(5,*)'      Jfar  =',Jfar
     225       close(5)
     226       write(*,*)'    Jfar=',Jfar
     227       end

Name     Type         Offset P Class

A        COMPLEX        3200   /DAM   /
A1       REAL          21468
AIMAG                          INTRINSIC
AN       COMPLEX       21112
ATAN                           INTRINSIC
B        REAL          21472
```

D Line# 1 7

B1	REAL	21034	
B2	REAL	21038	
C	REAL	0	/CRACK /
CMPLX			INTRINSIC
CONJG			INTRINSIC
COS			INTRINSIC
CSQRT			INTRINSIC
DEN	REAL	21010	
DUM	COMPLEX	20978	
DUM1	COMPLEX	21026	
DUM2	COMPLEX	21168	
DZ	COMPLEX	*****	
EMOD	REAL	0	/MAT /
FLOAT			INTRINSIC
FT3	COMPLEX	*****	
GMOD	REAL	4	/MAT /
I	INTEGER*2	20986	
IC	INTEGER*2	20976	
IPL	INTEGER*2	20896	
IZ	INTEGER*2	4800	/DAM /
J	INTEGER*2	21050	
JC	COMPLEX	21460	
JC1	COMPLEX	21492	
JFAR	REAL	21500	
JNEAR	REAL	21448	
K	REAL	12	/MAT /
K12	COMPLEX	21272	
KTOT	COMPLEX	21098	
LC	REAL	21452	
LDIG	INTEGER*2	14680	
LWGT	INTEGER*2	14780	
MASK	INTEGER*2	14880	
MFIL	CHAR*10	20902	
NDIS	INTEGER*2	20974	
NMIC	INTEGER*2	20972	
NNX	INTEGER*2	20880	
NNY	INTEGER*2	20882	
NU	REAL	8	/MAT /
PI	REAL	20892	
REAL			INTRINSIC
RKI	REAL	21440	
RKII	REAL	21444	
SIG	REAL	3816	
SIGXY	REAL	4	/REMOTE/
SIGYY	REAL	0	/REMOTE/
SIN			INTRINSIC
SQRT			INTRINSIC
THETA	REAL	21014	
VERTEX	REAL	14616	
WC	REAL	21456	
XCEN	REAL	20898	
XD	REAL	21018	
XDIS	REAL	21042	
XE	REAL	21002	
XEC	REAL	21066	
XLOW	REAL	20884	
XS	REAL	20994	
XSC	REAL	21058	
YD	REAL	21022	

```
D Line# 1     7
YDIS    REAL          21046
YE      REAL          21006
YEC     REAL          21070
YLOW    REAL          20888
YS      REAL          20998
YSC     REAL          21062
Z       COMPLEX       *****
Z01     COMPLEX           0   /DAM   /
Z02     COMPLEX        1600   /DAM   /
ZC      COMPLEX       *****
ZE      COMPLEX       21120
ZINC    COMPLEX       *****
ZLEV    REAL           3616
ZND     COMPLEX       21484
ZP      REAL             16
ZS      COMPLEX       *****
ZST     COMPLEX       21476


    228 c
    229 c ---------------------- Romberg Integration Routines ------------
    230 c
    231         Subroutine Qromb(ic,zst,znd,IIz)
    232       Parameter(eps=5.e-5,jmax=20,jmaxp=jmax+1,k=5,km=4)
    233       Complex*8 Iz1(jmaxp),dIz,IIz
    234       Real*4 h(jmaxp)
    235       h(1)=1.0
    236       do 0010 j=1,jmax
1   237       call Trapzd(ifunc,ic,zst,znd,Iz1(j),j)
1   238       if(j.ge.k)then
1   239        l=j-km
1   240        call Polint(h(l),Iz1(l),k,0.0,IIz,dIz)
1   241        if(cabs(dIz).lt.(eps*cabs(IIz)))return
1   242       endif
1   243       Iz1(j+1)=Iz1(j)
1   244       h(j+1)=0.25*h(j)
1   245  0010 continue
    246       write(*,*)'    Too many steps for reqd accuracy'
    247       end

Name    Type         Offset P Class

CABS                         INTRINSIC
DIZ     COMPLEX       21772
EPS                          PARAMETER
H       REAL          21682
IC      INTEGER*2         0 *
IFUNC   INTEGER*2     21768
IIZ     COMPLEX          12 *
IZ1     COMPLEX       21514
J       INTEGER*2     21766
JMAX                         PARAMETER
JMAXP                        PARAMETER
K                            PARAMETER
KM                           PARAMETER
L       INTEGER*2     21770

ZND     REAL              8 *
ZST     REAL              4 *
    248 c
```

```
    249 c
    250        SUBROUTINE POLINT(XA,YA,N,X,Y,DY)
    251       PARAMETER (NMAX=10)
    252       Complex*8 YA(N),C(NMAX),D(NMAX),Y,DY,W,WDEN
    253       DIMENSION XA(N)
    254        NS=1
    255        DIF=ABS(X-XA(1))
    256        DO 11 I=1,N
1   257          DIFT=ABS(X-XA(I))
1   258          IF (DIFT.LT.DIF) THEN
1   259           NS=I
1   260             DIF=DIFT
1   261          ENDIF
1   262          C(I)=YA(I)
1   263          D(I)=YA(I)
1   264 11     CONTINUE
    265       Y=YA(NS)
    266       NS=NS-1
    267       DO 13 M=1,N-1
1   268         DO 12 I=1,N-M
2   269           HO=XA(I)-X
2   270           HP=XA(I+M)-X
2   271           W=C(I+1)-D(I)
2   272           DEN=HO-HP
2   273           WDEN=W/DEN
2   274           D(I)=HP*WDEN
2   275           C(I)=HO*WDEN
2   276 12      CONTINUE
1   277         IF (2*NS.LT.N-M)THEN
1   278           DY=C(NS+1)
1   279         ELSE
1   280           DY=D(NS)
1   281           NS=NS-1
1   282         ENDIF
1   283         Y=Y+DY
1   284 13     CONTINUE
    285       RETURN
    286       END
```

Name	Type	Offset	P	Class
ABS				INTRINSIC
C	COMPLEX	21804		
D	COMPLEX	21884		
DEN	REAL	22012		
DIF	REAL	21966		
DIFT	REAL	21978		
DY	COMPLEX	20	*	
HO	REAL	21996		
HP	REAL	22000		
I	INTEGER*2	21970		
M	INTEGER*2	21982		
N	INTEGER*2	8	*	
NMAX				PARAMETER
NS	INTEGER*2	21964		
W	COMPLEX	22004		
WDEN	COMPLEX	22016		
X	REAL	12	*	
XA	REAL	0	*	

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
Y       COMPLEX          16 *
YA      COMPLEX           4 *


    287 c
    288 c
    289         Subroutine Trapzd(ic,zst,znd,IIz,n)
    290        Complex*8 zst,znd,IIz,del,z,Iz1,Iz2
    291        if(n.eq.1)then
    292         Call Fnct(ic,zst,Iz1)
    293         Call Fnct(ic,znd,Iz2)
    294         IIz=0.5*(znd-zst)*(Iz1+Iz2)
    295         it=1
    296        else
    297         tnm=float(it)
    298         del=(znd-zst)/tnm
    299         z=zst+0.5*del
    300         Iz2=cmplx(0.0,0.0)
    301         Do 0010 j=1,it
1   302         Call Fnct(ic,z,Iz1)
1   303         Iz2=Iz2+Iz1
1   304         z=z+del
1   305  0010   continue
    306         IIz=0.5*(IIz+(znd-zst)*Iz2/tnm)
    307         it=2*it
    308        endif
    309        return
    310        end

Name     Type         Offset P Class

CMPLX                          INTRINSIC
DEL      COMPLEX       22086
FLOAT                          INTRINSIC
IC       INTEGER*2         0 *
IIZ      COMPLEX          12 *
IT       INTEGER*2     22080
IZ1      COMPLEX       22048
IZ2      COMPLEX       22056
J        INTEGER*2     22118
N        INTEGER*2        16 *
TNM      REAL          22082
Z        COMPLEX       22102
ZND      COMPLEX           8 *
ZST      COMPLEX           4 *


    311 c
    312 c
    313         Subroutine Fnct(ic,z,IIz)
    314        Complex*8 z,fz1,fz2,ft1,ft2,IIz,z01(200),z02(200),a(200),
    315       1 phin,ksin,dum
    316        Common /dam/z01,z02,a,iz
    317        Common/remote/sigyy,sigxy
    318        ft1=cmplx(0.0,0.0)
    319        ft2=cmplx(0.0,0.0)

    320 c
    321 c      Sum over all Dislocations
    322 c
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
     323       do 0010 iz=1,ic
1    324       call PHI(z,fz1)
1    325       call KSI(z,fz2)
1    326       ft1=ft1-fz1
1    327       ft2=ft2-fz2
1    328  0010 continue
     329 c
     330 c     Add effects of remote loading in the potentials
     331 c
     332       if(sigyy.lt.0.0)then
     333        sigy1=0.0
     334       else
     335        sigy1=sigyy
     336       endif
     337       fz1=cmplx(sigy1,sigxy)
     338       phin=(sqrt(c)/4.0)*(fz1/csqrt(z))
     339       fz1=cmplx((0.25*sigyy),0.0)
     340       phin=phin+fz1
     341       fz1=(1.0/csqrt(conjg(z)))-(0.5/csqrt(z))
     342       fz2=(1.0/csqrt(conjg(z)))+(0.5/csqrt(z))
     343       dum=cmplx(0.0,1.0)
     344       ksin=(sqrt(c)/4.0)*((sigy1*fz1)-(sigxy*dum*fz2))
     345       dum=cmplx((0.5*sigyy),sigxy)
     346       ksin=ksin+dum
     347 c
     348       ft1=ft1+phin
     349       ft2=ft2+ksin
     350 c
     351       IIz=(ft1*ft1)+(2.0*ft1*ft2)
     352       return
     353       end

Name    Type         Offset P Class

A       COMPLEX        3200   /DAM   /
C       REAL          22200
CMPLX                        INTRINSIC
CONJG                        INTRINSIC
CSQRT                        INTRINSIC
DUM     COMPLEX       22308
FT1     COMPLEX       22150
FT2     COMPLEX       22158
FZ1     COMPLEX       22172
FZ2     COMPLEX       22180
IC      INTEGER*2         0 *
IIZ     COMPLEX           8 *
IZ      INTEGER*2      4800   /DAM   /
KSIN    COMPLEX       22316
PHIN    COMPLEX       22192
SIGXY   REAL              4   /REMOTE/
SIGY1   REAL          22188
SIGYY   REAL              0   /REMOTE/
SQRT                         INTRINSIC
Z       COMPLEX           4 *
Z01     COMPLEX           0   /DAM   /
Z02     COMPLEX        1600   /DAM   /
     354 c
```

```
D Line# 1      7                                Microsoft FORTRAN77 V3.20 02/84
    355 c ---------------------- Complex Function Routines -----------------
    356 c
    357        Subroutine PHI(z,fzt)
    358       Complex*8 z,z01(200),z02(200),fz,fzt,a(200),an
    359       Common /dam/z01,z02,a,iz
    360       an=a(iz)
    361       call Phi2(an,z,z01(iz),fz)
    362       fzt=(an/(z-z01(iz)))+fz
    363 c
    364 c     Add Negative Burgess Vector
    365 c
    366       an=-1.0*a(iz)
    367       call Phi2(an,z,z02(iz),fz)
    368       fzt=fzt+fz+(an/(z-z02(iz)))
    369       return
    370       end

Name     Type         Offset P Class

A       COMPLEX        3200   /DAM   /
AN      COMPLEX       22380
FZ      COMPLEX       22388
FZT     COMPLEX           4 *
IZ      INTEGER*2      4800   /DAM   /
Z       COMPLEX           0 *
Z01     COMPLEX           0   /DAM   /
Z02     COMPLEX        1600   /DAM   /


    371 c
    372 c
    373 c
    374        Subroutine Phi2(an,z,zo,fzt)
    375       Complex*8 an,ac,z,zo,fzt,fz1,fz2,fz3,Xz
    376       ac=conjg(an)
    377       zoc=conjg(zo)
    378       call fX(z,Xz)
    379       call F(z,zo,fz1)
    380       call F(z,zoc,fz2)
    381       call G(z,zoc,fz3)
    382 c     fzt=-1.0*((an*fz1)+(an*fz2)+(ac*(zo-zoc)*fz3))
    383       fzt=-1.0*((an*fz1)+(an*fz2)+(ac*(zo-zoc)*fz3)-(an*Xz))
    384       return
    385       end

Name     Type         Offset P Class

AC      COMPLEX       22420
AN      COMPLEX           0 *
CONJG                       INTRINSIC
FZ1     COMPLEX       22440
FZ2     COMPLEX       22448
FZ3     COMPLEX       22456
FZT     COMPLEX          12 *
XZ      COMPLEX       22432
Z       COMPLEX           4 *

ZO      COMPLEX           8 *
ZOC     REAL          22428
    386 c
```

D Lines 1 7 Microsoft FORTRAN77 V3.20 02/84

```
    387 c
    388 c
    389         Subroutine F(z,zo,fz)
    390        Complex*8 z,zo,fz,Xz,Xzo
    391        call fX(z,Xz)
    392        call fX(zo,Xzo)
    393        fz=(1.0-(Xz/Xzo))/(2.0*(z-zo))
    394        return
    395        end
```

Name	Type	Offset	P	Class
FZ	COMPLEX	8	*	
XZ	COMPLEX	22512		
XZO	COMPLEX	22520		
Z	COMPLEX	0	*	
ZO	COMPLEX	4	*	

```
    396 c
    397 c
    398 c
    399         Subroutine G(z,zo,fz)
    400        Complex*8 z,zo,fz,Xz,Xzo,XPzo
    401        call fX(z,Xz)
    402        call fX(zo,Xzo)
    403        call fXP(zo,XPzo)
    404        fz=1.0-(Xz/Xzo)+(((z-zo)*Xz*XPzo)/(Xzo*Xzo))
    405        fz=fz/(2.0*(z-zo)*(z-zo))
    406        return
    407        end
```

Name	Type	Offset	P	Class
FZ	COMPLEX	8	*	
XPZO	COMPLEX	22568		
XZ	COMPLEX	22552		
XZO	COMPLEX	22560		
Z	COMPLEX	0	*	
ZO	COMPLEX	4	*	

```
    408 c
    409 c
    410 c
    411         Subroutine fX(z,Xz)
    412        Complex*8 Xz,z,zpc
    413        Common /crack/ c
    414        zpc=(z+c)*z
    415        Xz=1.0/csqrt(zpc)
    416        return
    417        end
```

Name	Type	Offset	P	Class
C	REAL	0		/CRACK /
CSQRT				INTRINSIC
XZ	COMPLEX	4	*	
Z	COMPLEX	0	*	

```
D Line# 1     7
ZPC    COMPLEX       22640


    418 c
    419 c
    420 c
    421 c
    422         Subroutine fXP(z,XPz)
    423        Complex*8 XPz,z,zpc,za,zb
    424        Common /crack/ c
    425        zpc=z+c
    426        za=csqrt(z)
    427        za=za*za*za*csqrt(zpc)
    428        zb=csqrt(z)*csqrt(zpc*zpc*zpc)
    429        XPz=-0.5*((1.0/za)+(1.0/zb))
    430        return
    431        end

Name     Type         Offset P Class

C       REAL              0   /CRACK /
CSQRT                         INTRINSIC
XPZ     COMPLEX           4 *
Z       COMPLEX           0 *
ZA      COMPLEX       22680
ZB      COMPLEX       22728
ZPC     COMPLEX       22672


    432 c
    433 c
    434         Subroutine KSI(z,fzt)
    435       Complex*8 z,zc,zoc,z01(200),z02(200),fz1,fz2,fz3,fzt,a(200),
    436      1 an,ac
    437        Common /dam/z01,z02,a,iz
    438        an=a(iz)
    439        zc=conjg(z)
    440        call Phi2(an,zc,z01(iz),fz1)
    441        fz1=conjg(fz1)
    442        call Phi2(an,z,z01(iz),fz2)
    443        call Phi2P(an,z,z01(iz),fz3)
    444        fzt=fz1-fz2-(z*fz3)
    445        ac=conjg(an)
    446        zoc=conjg(z01(iz))
    447        fzt=fzt+(ac/(z-z01(iz)))+((an*zoc)/((z-z01(iz))*(z-z01(iz))))
    448 c
    449 c      Add Negative of Second Burgess Vector
    450 c
    451        an=-1.0*a(iz)
    452        call Phi2(an,zc,z02(iz),fz1)
    453        fz1=conjg(fz1)
    454        call Phi2(an,z,z02(iz),fz2)
    455        call Phi2P(an,z,z02(iz),fz3)
    456        fzt=fzt+fz1-fz2-(z*fz3)
    457        ac=conjg(an)
    458        zoc=conjg(z02(iz))
    459        fzt=fzt+(ac/(z-z02(iz)))+((an*zoc)/((z-z02(iz))*(z-z02(iz))))
    460        return
    461        end
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

Name	Type	Offset	P	Class
A	COMPLEX	3200		/DAM /
AC	COMPLEX	22856		
AN	COMPLEX	22808		
CONJG				INTRINSIC
FZ1	COMPLEX	22824		
FZ2	COMPLEX	22832		
FZ3	COMPLEX	22840		
FZT	COMPLEX	4	*	
IZ	INTEGER*2	4800		/DAM /
Z	COMPLEX	0	*	
Z01	COMPLEX	0		/DAM /
Z02	COMPLEX	1600		/DAM /
ZC	COMPLEX	22816		
ZOC	COMPLEX	22864		

```
462 c
463 c
464 c
465         Subroutine Phi2P(an,z,zo,fzt)
466        Complex*8 an,ac,z,zo,fzt,fz1,fz2,fz3,XPz
467        ac=conjg(an)
468        zoc=conjg(zo)
469        call fXP(z,XPz)
470        call FP(z,zo,fz1)
471        call FP(z,zoc,fz2)
472        call GP(z,zoc,fz3)
473        fzt=-1.0*(an*fz1+an*fz2+ac*(zo-zoc)*fz3-an*XPz)
474 c      fzt=-1.0*(an*fz1+an*fz2+ac*(zo-zoc)*fz3)
475        return
476        end
```

Name	Type	Offset	P	Class
AC	COMPLEX	22952		
AN	COMPLEX	0	*	
CONJG				INTRINSIC
FZ1	COMPLEX	22972		
FZ2	COMPLEX	22980		
FZ3	COMPLEX	22988		
FZT	COMPLEX	12	*	
XPZ	COMPLEX	22964		
Z	COMPLEX	4	*	
ZO	COMPLEX	8	*	
ZOC	REAL	22960		

```
477 c
478 c
479 c
480         Subroutine FP(z,zo,fz)
481        Complex*8 z,zo,fz,Xz,Xzo,XPz
482        call fX(z,Xz)
483        call fX(zo,Xzo)
484        call fXP(z,XPz)
485        fz=1.0-(Xz/Xzo)+(((z-zo)*XPz)/Xzo)
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
    486         fz=fz/(-2.0*(z-zo)*(z-zo))
    487         return
    488         end
```

Name	Type	Offset	P	Class
FZ	COMPLEX	8	*	
XPZ	COMPLEX	23060		
XZ	COMPLEX	23044		
XZO	COMPLEX	23052		
Z	COMPLEX	0	*	
ZO	COMPLEX	4	*	

```
    489 c
    490 c
    491 c
    492          Subroutine GP(z,zo,fz)
    493         Complex*8 z,zo,fz,fz1,fz2,Xz,Xzo,XPz,XPzo
    494         call fX(z,Xz)
    495         call fX(zo,Xzo)
    496         call fXP(z,XPz)
    497         call fXP(zo,XPzo)
    498         fz1=1.0-(Xz/Xzo)+(((z-zo)*Xz*XPzo)/(Xzo*Xzo))
    499         fz1=fz1/((z-zo)*(z-zo)*(z-zo))
    500         fz2=((z-zo)*XPzo*XPz)/(Xzo*Xzo)
    501         fz2=fz2+((XPzo*Xz)/(Xzo*Xzo))-(XPz/Xzo)
    502         fz2=fz2/(2.0*(z-zo)*(z-zo))
    503         fz=fz2-fz1
    504         return
    505         end
```

Name	Type	Offset	P	Class
FZ	COMPLEX	8	*	
FZ1	COMPLEX	23148		
FZ2	COMPLEX	23220		
XPZ	COMPLEX	23132		
XPZO	COMPLEX	23140		
XZ	COMPLEX	23116		
XZO	COMPLEX	23124		
Z	COMPLEX	0	*	
ZO	COMPLEX	4	*	

```
    506 c
    507 c
    508          Subroutine PHIP(z,fzt)
    509         Complex*8 z,z01(200),z02(200),fz,fzt,a(200),an
    510         Common /dam/z01,z02,a,iz
    511         an=a(iz)
    512         call Phi2P(an,z,z01(iz),fz)
    513         fzt=fz-(an/((z-z01(iz))*(z-z01(iz))))
    514         an=-1.0*a(iz)
    515         call Phi2P(an,z,z02(iz),fz)
    516         fzt=fzt+fz-(an/((z-z02(iz))*(z-z02(iz))))
    517         return
    518         end
```

D Line# 1 7

Name	Type	Offset	P	Class
A	COMPLEX	3200		/DAM /
AN	COMPLEX	23316		
FZ	COMPLEX	23324		
FZT	COMPLEX	4	*	
IZ	INTEGER*2	4800		/DAM /
Z	COMPLEX	0	*	
Z01	COMPLEX	0		/DAM /
Z02	COMPLEX	1600		/DAM /

```
519 c
520 c
521
```

Name	Type	Size	Class
CRACK		4	COMMON
DAM		4802	COMMON
ENERGY			PROGRAM
F			SUBROUTINE
FNCT			SUBROUTINE
FP			SUBROUTINE
FX			SUBROUTINE
FXP			SUBROUTINE
G			SUBROUTINE
GP			SUBROUTINE
KSI			SUBROUTINE
[illegible]		16	COMMON
PHI			SUBROUTINE
PHI2			SUBROUTINE
PHI2P			SUBROUTINE
PHIP			SUBROUTINE
POLINT			SUBROUTINE
QROMB			SUBROUTINE
REMOTE		8	COMMON
TRAPZD			SUBROUTINE

Pass One No Errors Detected
521 Source Lines

```
D Line# 1     7                                  Microsoft FORTRAN77 V3.20 02/84
      1 $Storage: 2
      2 $Debug
      3         Program kmin
      4 c*************************************************************************
      5 c***** This program is designed to compute the stress intensity     *
      6 c***** factor at various locations in front of the crack tip for a  *
      7 c***** particular damage configuration.                             *
      8 c*************************************************************************
      9 c
     10       Complex*8 z,zst,znd,zinc,zc,z01(200),z02(200),
     11      1 a(200),dum1,dum,dum2,zs,ze,dz,an,Ktot,K12,ft3
     12       Real*4 K,nu,xs(50),xe(50),ys(50),ye(50),b1(50),b2(50)
     13       Character*10 mfil
     14 c
     15       pi=3.141592654
     16       write(*,*)'    Give material properties'
     17  0005 write(*,*)'   emod,nu,  0-> pl. stress  1-> pl. strain'
     18       read(*,*,err=0005)emod,nu,ipl
     19       if(ipl.eq.0)then
     20        k=(3.0-nu)/(1.0+nu)
     21       else
     22        k=3.0-4.0*nu
     23       endif
     24       gmod=emod/(2.0*(1.0+nu))
     25  0010 write(*,*)'    Give main crack length'
     26       read(*,*,err=0010)c
     27  0007 write(*,*)'   Damage zone length'
     28       read(*,*,err=0007)xmax
     29  0038 write(*,*)'   Give  sig-xy '
     30       read(*,*,err=0038)sigxy
     31       rK2c=sigxy*sqrt(pi*0.5*c)
     32       write(*,*)'  '
     33       write(*,*)'   Give File Name for Microcrack Data'
     34       read(*,0011)mfil
     35  0011 format(a10)
     36 c
     37       open(5,file=mfil,status='old')
     38 c
     39 c     nmic=# of microcracks   ndis=# of dislocations/micro
     40 c
     41       read(5,*)nmic,ndis
     42       if(ndis.le.0.or.ndis.gt.10)ndis=1
     43 c
     44 c     Next read coords of the microcrack (xs,ys) to (xe,ye)
     45 c
     46       do 0012 i=1,nmic
1    47       read(5,*)xs(i),ys(i),xe(i),ye(i)
1    48       read(5,*)b1(i),b2(i)
1    49  0012 continue
     50       close(5)
     51 c
     52 c     Start the computation for the SIF
     53 c
     54       open(5,file='kcrak.out',status='new')
     55       open(6,file='kdam.out',status='new')
     56       open(7,file='ktot.out',status='new')
     57       do 0016 j=1, 100
1    58       write(*,*)' working on ',j,'th point'
1    59       ic=0
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1    60        xcen=(xmax*float(j-1))/100.0
1    61        do 0017 i=1,nmic
2    62        xes=xe(i)-xcen
2    63        xss=xs(i)-xcen
2    64        yes=ye(i)
2    65        yss=ys(i)
2    66        den=xes-xss
2    67        if(den.lt.0.000001)then
2    68         theta=pi/2.0
2    69        else
2    70         theta=atan((yes-yss)/den)
2    71        endif
2    72        xd=cos(theta)
2    73        yd=sin(theta)
2    74        dum1=cmplx(xd,yd)
2    75        b1s=b1(i)/float(ndis)
2    76        b2s=b2(i)/float(ndis)
2    77        xdis=(xes-xss)/(2.0*float(ndis+1))
2    78        ydis=(yes-yss)/(2.0*float(ndis+1))
2    79 c
2    80        dum=cmplx(0,1)
2    81        do 0013 l=1,ndis
3    82        ic=ic+1
3    83        xsc=xss+(xdis*float(l))
3    84        ysc=yss+(ydis*float(l))
3    85        xec=xes-(xdis*float(l))
3    86        yec=yes-(ydis*float(l))
3    87        z01(ic)=cmplx(xsc,ysc)
3    88        z02(ic)=cmplx(xec,yec)
3    89        a(ic)=cmplx(b1s,b2s)
3    90        a(ic)=(gmod/(pi*(k+1)))*dum1*a(ic)
3    91        a(ic)=a(ic)/dum
3    92  0013 continue
2    93  0017 continue
1    94 c
1    95 c     Start computation of the Stress Intensity Factor
1    96 c
1    97        cl=c+xcen
1    98        Ktot=cmplx(0.0,0.0)
1    99        Do 0030 i=1,ic
2   100        an=a(i)
2   101        ze=z01(i)
2   102        dum=csqrt((ze+cl)/ze)
2   103        dum=-1.0*an*(real(dum)-1.0)
2   104        dum2=csqrt(conjg(ze))
2   105        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+cl)
2   106        dum1=(conjg(an)*aimag(ze)*cl)/dum2
2   107        dum2=cmplx(0.0,1.0)
2   108        dum1=dum1*dum2
2   109        K12=-2.0*sqrt((2*pi)/cl)*(dum+dum1)
2   110        an=-1.0*a(i)
2   111        ze=z02(i)
2   112        dum=csqrt((ze+cl)/ze)
2   113        dum=-1.0*an*(real(dum)-1.0)
2   114        dum2=csqrt(conjg(ze))
2   115        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+cl)
2   116        dum1=(conjg(an)*aimag(ze)*cl)/dum2
2   117        dum2=cmplx(0.0,1.0)
2   118        dum1=dum1*dum2
```

```
D Line# 1     7                                    Microsoft FORTRAN77 V3.20 02/84
2   119       K12=K12-(2.0*sqrt((2*pi)/cl)*(dum+dum1))
2   120       Ktot=Ktot+K12
2   121  0030 continue
1   122 c
1   123 c     Write out the Stress Intensity Factors
1   124 c
1   125       rKII=-1.0*aimag(Ktot)
1   126       rKII=rKII/rk2c
1   127       rkcrak=sigxy*sqrt(pi*0.5*cl)
1   128       rkcrak=rkcrak/rk2c
1   129       write(5,*)xcen,rkcrak
1   130       write(6,*)xcen,rKII
1   131       tot=rkcrak+rKII
1   132       write(7,*)xcen,tot
1   133       write(*,*)'   '
1   134       write(*,*)' Stress Intensity Factors:  '
1   135       write(*,*)'   Kdam=',rKII,'   Kcrack=',rkcrak,'   Ktot=',tot
1   136  0016 continue
    137       close(5)
    138       close(6)
    139       close(7)
    140       end
```

Name	Type	Offset	P	Class
A	COMPLEX	3216		
AIMAG				INTRINSIC
AN	COMPLEX	6280		
ATAN				INTRINSIC
B1	REAL	5616		
B1S	REAL	6190		
B2	REAL	5816		
B2S	REAL	6194		
C	REAL	6038		
CL	REAL	6262		
CMPLX				INTRINSIC
CONJG				INTRINSIC
COS				INTRINSIC
CSQRT				INTRINSIC
DEN	REAL	6166		
DUM	COMPLEX	6206		
DUM1	COMPLEX	6182		
DUM2	COMPLEX	6336		
DZ	COMPLEX	*****		
EMOD	REAL	6020		
FLOAT				INTRINSIC
FT3	COMPLEX	*****		
GMOD	REAL	6034		
I	INTEGER*2	6128		
IC	INTEGER*2	6138		
IPL	INTEGER*2	6028		
J	INTEGER*2	6136		
K	REAL	6030		
K12	COMPLEX	6440		
KTOT	COMPLEX	6266		
L	INTEGER*2	6214		
MFIL	CHAR*10	6054		
NDIS	INTEGER*2	6126		
NMIC	INTEGER*2	6124		

NU	REAL	6024	
PI	REAL	6016	
REAL			INTRINSIC
RK2C	REAL	6050	
RKCRAK	REAL	6612	
RKII	REAL	6608	
SIGXY	REAL	6046	
SIN			INTRINSIC
SQRT			INTRINSIC
THETA	REAL	6170	
TOT	REAL	6616	
XCEN	REAL	6140	
XD	REAL	6174	
XDIS	REAL	6198	
XE	REAL	5416	
XEC	REAL	6230	
XES	REAL	6150	
XMAX	REAL	6042	
XS	REAL	4816	
XSC	REAL	6222	
XSS	REAL	6154	
YD	REAL	6178	
YDIS	REAL	6202	
YE	REAL	5216	
YEC	REAL	6234	
YES	REAL	6158	
YS	REAL	5016	
YSC	REAL	6226	
YSS	REAL	6162	
Z	COMPLEX	*****	
Z01	COMPLEX	16	
Z02	COMPLEX	1616	
ZC	COMPLEX	*****	
ZE	COMPLEX	6288	
ZINC	COMPLEX	*****	
ZND	COMPLEX	*****	
ZS	COMPLEX	*****	
ZST	COMPLEX	*****	

Name	Type	Size	Class
KMIN			PROGRAM

Pass One No Errors Detected
140 Source Lines

D Lines 1 7 Microsoft FORTRAN77 V3.20 02/84

```
      1 $Storage: 2
      2 $Debug
      3           Program SIF
      4 c*************************************************************************
      5 c***** This program is designed to generate a greens function contour
      6 c***** plot using the complex potential results.                        *
      7 c*************************************************************************
      8 c
      9       Complex*8 z,zc,zck1,zck2,z01,z02,fz1,fz2,fz3,trans,
     10      1 a,dum1,dum,dum2,dum3,dum4,zs,ze,dz,an,Ktot,K12
     11       Real*4 k,nu
     12       Common/Work1/sif(50,50,2),zp(50,50),zlev(50),vertex(16)
     13       Integer*2 ldig(50),lwgt(50),mask(3000)
     14       Data nnx,nny,xlow,ylow /50,50,1.0,1.0/
     15 c
     16       pi=3.141592654
     17       write(*,*)'    Give material properties'
     18  0005 write(*,*)'   emod,nu,  0-> pl. stress  1-> pl. strain'
     19       read(*,*,err=0005)emod,nu,ipl
     20       if(ipl.eq.0)then
     21        k=(3.0-nu)/(1.0+nu)
     22       else
     23        k=3.0-4.0*nu
     24       endif
     25       gmod=emod/(2.0*(1.0+nu))
     26  0010 write(*,*)'    Give main crack length'
     27       read(*,*,err=0010)c
     28       write(*,*)'  Give the microcrack orientation,csd,cod'
     29       read(*,*)theta,bx,by
     30       theta=(pi*theta)/180.0
     31 c
     32 c     Compute the Burgess Vector for the values above
     33 c
     34       ci=cos(theta)
     35       si=sin(theta)
     36       trans=cmplx(ci,si)
     37       dum=cmplx(0,1)
     38       a=cmplx(bx,by)
     39       a=trans*a
     40       a=(gmod/(pi*(k+1)))*a
     41       a=a/dum
     42       ax=real(a)
     43       ay=aimag(a)
     44       write(*,*)'  a =',ax,'   + i',ay
     45       bmag=sqrt(bx*bx+by*by)
     46 c
     47 c     Start grid loop here
     48 c
     49  0025 write(*,*)'   Give starting and ending points xs,ys,xe,ye'
     50       read(*,*,err=0025)xs,ys,xe,ye
     51 c     write(*,*)'   Give grid resolution nx,ny  (max = 30)'
     52 c     read(*,*,err=0025)nx,ny
     53       nx=49
     54       ny=49
     55       dx=(xe-xs)/float(nx-1)
     56       dy=(ye-ys)/float(ny-1)
     57 c
     58 c     Define the grid for the SIF computation
     59 c
```

```
     60        sif1=0.0
     61        sif2=0.0
     62 c
     63 c
     64        do 0035 i=1,nx
1    65        xd=dx*float(i-1)
1    66        do 0045 j=1,ny
2    67        yd=dy*float(j-1)
2    68        yval=ys+yd
2    69        xval=xs+xd
2    70        z=cmplx(xval,yval)
2    71        zck1=z-c
2    72        cz=0.01*cos(theta)
2    73        sz=0.01*sin(theta)
2    74        zc=cmplx(cz,sz)
2    75        zck2=z-zc
2    76        if(cabs(z).lt.1.0e-6)then
2    77         write(*,*)'   Singularity at x=',xval,'y=',yval
2    78         do 0043 m=1,2
3    79         sif(i,j,m)=1.0e35
3    80  0043   continue
2    81         go to 0045
2    82        endif
2    83        if(cabs(zck1).lt.1.0e-6)then
2    84         write(*,*)'   Singularity at x=',xval,'y=',yval
2    85         do 0044 m=1,2
3    86         sif(i,j,m)=1.0e35
3    87  0044   continue
2    88         go to 0045
2    89        endif
2    90        if(cabs(zck2).lt.1.0e-6)then
2    91         write(*,*)'   Singularity at x=',xval,'y=',yval
2    92         do 0042 m=1,2
3    93         sif(i,j,m)=1.0e35
3    94  0042   continue
2    95         go to 0045
2    96        endif
2    97        xd2=0.01*cos(theta)
2    98        yd2=0.01*sin(theta)
2    99        x2=xval+xd2
2   100        y2=yval+yd2
2   101        z01=z
2   102        z02=cmplx(x2,y2)
2   103        xave=0.5*(xval+x2)
2   104        yave=0.5*(yval+y2)
2   105        rad=sqrt(xave*xave+yave*yave)
2   106 c
2   107 c      Compute Nondimensionalizing factor via Chudnovsky approach
2   108 c
2   109        fact=gmod*sqrt(pi*0.5*c)*bmag*(0.01)
2   110 c      fact=(sqrt(pi/2.0)*(k+1.0)*sqrt(bmag))/(gmod*.01)
2   111 c
2   112 c      Compute the Stress Intensity Factor for the dipole
2   113 c
2   114        an=a
2   115        ze=z01
2   116        dum=csqrt((ze+c)/ze)
2   117        dum=-1.0*an*(real(dum)-1.0)
2   118        dum2=csqrt(conjg(ze))
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
2   119        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+c)
2   120        dum1=(conjg(an)*aimag(ze)*c)/dum2
2   121        dum2=cmplx(0.0,1.0)
2   122        dum1=dum1*dum2
2   123        K12=-2.0*sqrt((2.0*pi)/c)*(dum+dum1)
2   124        an=-1.0*a
2   125        ze=z02
2   126        dum=csqrt((ze+c)/ze)
2   127        dum=-1.0*an*(real(dum)-1.0)
2   128        dum2=csqrt(conjg(ze))
2   129        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+c)
2   130        dum1=(conjg(an)*aimag(ze)*c)/dum2
2   131        dum2=cmplx(0.0,1.0)
2   132        dum1=dum1*dum2
2   133        K12=K12-(2.0*sqrt((2*pi)/c)*(dum+dum1))
2   134 c
2   135        sif1=real(K12)/fact
2   136        sif2=(-1.0*aimag(K12))/fact
2   137        sif(i,j,2)=sif2
2   138        sif(i,j,1)=sif1
2   139 c      write(*,*)'    '
2   140 c      write(*,*)'    '
2   141 c      write(*,*)'    Location:      X=',xval,'    Y=',yval
2   142 c      write(*,*)'    SIFs:          KI=',sif1,'      KII=',sif2
2   143        if(abs(sm2).lt.abs(sif2))smy=sif(i,j,2)
2   144        if(abs(sm1).lt.abs(sif1))smx=sif(i,j,1)
2   145  0045  continue
1   146  0035  continue
    147 c
    148 c
    149 c      Prepare to plot the results
    150 c
    151  0050  write(*,*)'     Give Ioport,Model,Fact'
    152        read(*,*,err=0050)ioport,model,fact
    153  0055  write(*,*)'    KI -> 1,      KII -> 2'
    154        read(*,*,err=0055)ip
    155        do 0090 i=1,nx
1   156        do 0095 j=1,ny
2   157        zp(i,j)=sif(i,j,ip)
2   158  0095  continue
1   159  0090  continue
    160        if(ip.eq.1)then
    161         write(*,*)'   The maximum value is =',sif1
    162        else
    163         write(*,*)'   The maximum value is =',sif2
    164        endif
    165  0070  write(*,*)'  -- Give Contour Information --'
    166        write(*,*)'  #levs, min lev, max lev, # betw labels, # first '
    167        read(*,*,err=0070)nlev,vmin,vmax,nbl,nst
    168        dlev=(vmax-vmin)/float(nlev)
    169        il=nst
    170        do 0075 i=1,nlev
1   171        zlev(i)=vmin+(dlev*float(i))
1   172        if(il.eq.nbl)then
1   173         ldig(i)=1
1   174         lwgt(i)=3
1   175         il=0
1   176        else
1   177         ldig(i)=-2
```

```
D Line# 1      7
1   178         lwgt(i)=1
1   179        endif
1   180        il=il+1
1   181   0075 continue
      182        xhigh=5.0
      183        yhigh=5.0
      184        hgt=0.07
      185        narc=3
      186        ndiv=2
      187        call plots(0,ioport,model)
      188        call factor(fact)
      189        call zcseg(zp,nnx,nny,nx,ny,xlow,ylow,xhigh,yhigh,zlev,ldig,
      190       1     lwgt,nlev,hgt,ndiv,narc)
      191        do 0130 i=1,nlev
1     192        zlev(i)=-1.0*zlev(i)
1     193        lwgt(i)=-1.0*lwgt(i)
1     194   0130 continue
      195 c      call stline(-1,.1389,0.)
      196        call zcseg(zp,nnx,nny,nx,ny,xlow,ylow,xhigh,yhigh,zlev,ldig,
      197       1     lwgt,nlev,hgt,ndiv,narc)
      198 c      call stline(1,.1389,0.)
      199        xln=xhigh-xlow
      200        yln=yhigh-ylow
      201        xdel=(xe-xs)/(xhigh-xlow)
      202        ydel=(ye-ys)/(yhigh-ylow)
      203        hgt1=.1
      204        hgt2=.1
      205        call staxis(hgt1,hgt2,hgt1,hgt2,2)
      206        call axis(xlow,ylow,' ',-1,xln,0.0,xs,xdel)
      207        call axis(xlow,ylow,' ',1,yln,90.0,ys,ydel)
      208        call axis(xlow,yhigh,' ',1,xln,0.0,xs,xdel)
      209        call axis(xhigh,ylow,' ',-1,yln,90.0,ys,ydel)
      210        call plot(0.,0.,999)
      211   0100 write(*,*)'   3-D Plot  -> 1'
      212        read(*,*,err=0100)i3d
      213        if(i3d.gt.0)then
      214   0110  write(*,*)'   Give horiz, vert, angles'
      215         read(*,*,err=0110)ahor,aver
      216         call plots(0,ioport,model)
      217         call factor(fact)
      218         iproj=0
      219         itrim=0
      220         idir=3
      221         zlow=1.0e35
      222         iframe=1
      223         icut=3
      224         call mesh(zp,nnx,nny,nx,ny,ahor,aver,xlow,ylow,xhigh,
      225        1      yhigh,iedge,idir,iproj,iframe,zlow,icut,
      226        2      itrim,mask,vertex)
      227          call plot(0.,0.,999)
      228        endif
      229   0080 write(*,*)'   Try a different plot  -> 1'
      230        read(*,*,err=0080)inew
      231        if(inew.gt.0)go to 0050
      232        end

Name    Type        Offset P Class

A       COMPLEX       6286
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

ABS			INTRINSIC
AHOR	REAL	6968	
AIMAG			INTRINSIC
AN	COMPLEX	6524	
AVER	REAL	6972	
AX	REAL	6318	
AY	REAL	6322	
BMAG	REAL	6326	
BX	REAL	6254	
BY	REAL	6258	
C	REAL	6246	
CABS			INTRINSIC
CI	REAL	6262	
CMPLX			INTRINSIC
CONJG			INTRINSIC
COS			INTRINSIC
CSQRT			INTRINSIC
CZ	REAL	6414	
DLEV	REAL	6908	
DUM	COMPLEX	6278	
DUM1	COMPLEX	6652	
DUM2	COMPLEX	6580	
DUM3	COMPLEX	*****	
DUM4	COMPLEX	*****	
DX	REAL	6350	
DY	REAL	6354	
DZ	COMPLEX	*****	
EMOD	REAL	6228	
FACT	REAL	6520	
FLOAT			INTRINSIC
FZ1	COMPLEX	*****	
FZ2	COMPLEX	*****	
FZ3	COMPLEX	*****	
GMOD	REAL	6242	
HGT	REAL	6928	
HGT1	REAL	6958	
HGT2	REAL	6962	
I	INTEGER*2	6366	
I1	INTEGER*2	6912	
IJD	INTEGER*2	6966	
ICUT	INTEGER*2	6988	
IDIR	INTEGER*2	6980	
IEDGE	INTEGER*2	6990	
IFRAME	INTEGER*2	6986	
INEW	INTEGER*2	6992	
IOPORT	INTEGER*2	6876	
IP	INTEGER*2	6880	
IPL	INTEGER*2	6236	
IPROJ	INTEGER*2	6976	
ITRIM	INTEGER*2	6978	
J	INTEGER*2	6378	
K	REAL	6238	
K12	COMPLEX	6692	
KTOT	COMPLEX	*****	
LDIG	INTEGER*2	12	
LWGT	INTEGER*2	112	
M	INTEGER*2	6450	
MASK	INTEGER*2	212	
MODEL	INTEGER*2	6878	

D Line# 1 7

NARC	INTEGER*2	6932	
NBL	INTEGER*2	6904	
NDIV	INTEGER*2	6934	
NLEV	INTEGER*2	6894	
NNX	INTEGER*2	6212	
NNY	INTEGER*2	6214	
NST	INTEGER*2	6906	
NU	REAL	6232	
NX	INTEGER*2	6346	
NY	INTEGER*2	6348	
PI	REAL	6224	
RAD	REAL	6516	
REAL			INTRINSIC
SI	REAL	6266	
SIF	REAL	0	/WORK1 /
SIF1	REAL	6358	
SIF2	REAL	6362	
SIN			INTRINSIC
SM1	REAL	6868	
SM2	REAL	6860	
SMX	REAL	6872	
SMY	REAL	6864	
SQRT			INTRINSIC
SZ	REAL	6418	
THETA	REAL	6250	
TRANS	COMPLEX	6270	
VERTEX	REAL	30200	/WORK1 /
VMAX	REAL	6900	
VMIN	REAL	6896	
X2	REAL	6484	
XAVE	REAL	6508	
XD	REAL	6374	
XD2	REAL	6476	
XDEL	REAL	6950	
XE	REAL	6338	
XHIGH	REAL	6920	
XLN	REAL	6942	
XLOW	REAL	6216	
XS	REAL	6330	
XVAL	REAL	6394	
Y2	REAL	6488	
YAVE	REAL	6512	
YD	REAL	6386	
YD2	REAL	6480	
YDEL	REAL	6954	
YE	REAL	6342	
YHIGH	REAL	6924	
YLN	REAL	6946	
YLOW	REAL	6220	
YS	REAL	6334	
YVAL	REAL	6390	
Z	COMPLEX	6398	
Z01	COMPLEX	6492	
Z02	COMPLEX	6500	
ZC	COMPLEX	6422	
ZCX1	COMPLEX	5406	
ZCX2	COMPLEX	6430	
ZE	COMPLEX	6532	
ZLEV	REAL	30000	/WORK1 /

```
D Line# 1     7
ZLOW    REAL          6982
ZP      REAL         20000   /WORK1 /
ZS      COMPLEX      *****


Name    Type          Size   Class

AXIS                         SUBROUTINE
FACTOR                       SUBROUTINE
MESH                         SUBROUTINE
PLOT                         SUBROUTINE
PLOTS                        SUBROUTINE
SIF                          PROGRAM
STAXIS                       SUBROUTINE
WORK1                30264   COMMON
ZCSEG                        SUBROUTINE

 Pass One     No Errors Detected
             232 Source Lines
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
      1 $Storage: 2
      2 $Debug
      3         Program Stress
      4 c*************************************************************************
      5 c***** This program is designed to compute the stress field in the *
      6 c***** vicinity of a main crack givin a damage in the form of dis- *
      7 c***** locations.  This technique will use the complex potential   *
      8 c***** formulation presented by K.K. Lo.                           *
      9 c*************************************************************************
     10 c
     11       Complex*8 z,zc,zck1,zck2,zck3,z01(50),z02(50),fz1,fz2,fz3,
     12      1 a(50),dum1,dum,zs,ze,dz
     13       Real*4 k,sig(30,30,3),emod,nu,zp(30,30),zlev(50),vertex(16),
     14      1 KI,KII
     15       Integer*2 ldig(50),lwgt(50),mask(3000)
     16       Common /mat/ emod,gmod,nu,k
     17       Common /dam/z01,z02,a,iz
     18       Common /crack/ c
     19       Data nnx,nny,xlow,ylow /30,30,1.0,1.0/
     20 c
     21       pi=3.141592654
     22       write(*,*)'    Give material properties'
     23  0005 write(*,*)'   emod,nu,  0-> pl. stress  1-> pl. strain'
     24       read(*,*,err=0005)emod,nu,ipl
     25       if(ipl.eq.0)then
     26        k=(3.0-nu)/(1.0+nu)
     27       else
     28        k=3.0-4.0*nu
     29       endif
     30       gmod=emod/(2.0*(1.0+nu))
     31  0010 write(*,*)'    Give main crack length'
     32       read(*,*,err=0010)c
     33       write(*,*)'  '
     34       write(*,*)'   ---------- Give Microcrack Information ---------'
     35  0011 write(*,*)'       Give # micros, # dislocations/micro '
     36       read(*,*,err=0011)nmic,ndis
     37       if(ndis.le.0.or.ndis.gt.10)ndis=1
     38       ic=0
     39       dum=cmplx(0,1)
     40 c
     41       do 0012 i=1,nmic
1    42  0015 write(*,*)'  Give starting and ending points ',i,'th micro'
1    43       write(*,*)'   xstart,ystart,xend,yend'
1    44       read(*,*,err=0015)xs,ys,xe,ye
1    45       den=xe-xs
1    46       if(den.lt.0.000001)then
1    47        theta=pi/2.0
1    48       else
1    49        theta=atan((ye-ys)/den)
1    50       endif
1    51       xd=cos(theta)
1    52       yd=sin(theta)
1    53       dum1=cmplx(xd,yd)
1    54  0020 write(*,*)'  Give  the CSD, and the COD'
1    55       read(*,*,err=0020)b1,b2
1    56       b1=b1/float(ndis)
1    57       b2=b2/float(ndis)
1    58       xdis=(xe-xs)/(2.0*float(ndis+1))
1    59       ydis=(ye-ys)/(2.0*float(ndis+1))
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
1    60 c
1    61        do 0013 j=1,ndis
2    62        ic=ic+1
2    63        xsc=xs+(xdis*float(j))
2    64        ysc=ys+(ydis*float(j))
2    65        xec=xe-(xdis*float(j))
2    66        yec=ye-(ydis*float(j))
2    67        z01(ic)=cmplx(xsc,ysc)
2    68        z02(ic)=cmplx(xec,yec)
2    69        a(ic)=cmplx(b1,b2)
2    70        a(ic)=(gmod/(pi*(k+1)))*dum1*a(ic)
2    71        a(ic)=a(ic)/dum
2    72   0013 continue
1    73   0012 continue
     74 c
     75 c      Select points of interest for stress computation
     76 c
     77   0021 write(*,*)'    Superimpose remote loading   Yes ->1'
     78        read(*,*,err=0021)irl
     79        if(irl.gt.0)then
     80   0022  write(*,*)'   Give values for sigyy, sigxy'
     81         read(*,*,err=0022)ryy,rxy
     82         KI=ryy*sqrt(pi*c/2.0)
     83         KII=rxy*sqrt(pi*c/2.0)
     84        endif
     85   0025 write(*,*)'   Give starting and ending points xs,ys,xe,ye'
     86        read(*,*,err=0025)xs,ys,xe,ye
     87        zs=cmplx(xs,ys)
     88        ze=cmplx(xe,ye)
     89        write(*,*)'   Give grid resolution nx,ny  (max = 30)'
     90        read(*,*,err=0025)nx,ny
     91        dx=(xe-xs)/float(nx-1)
     92        dy=(ye-ys)/float(ny-1)
     93 c
     94 c      Define the grid for the stress computation
     95 c
     96        z=zs
     97        smx=0.0
     98        smy=0.0
     99        smxy=0.0
    100        l=0
    101 c
    102 c      Perform Double Loop over the Stress Grid
    103 c
    104        do 0035 i=1,nx
1   105        xd=dx*float(i-1)
1   106        do 0045 j=1,ny
2   107        yd=dy*float(j-1)
2   108        l=l+1
2   109        yval=ys+yd
2   110        xval=xs+xd
2   111        z=cmplx(xval,yval)
2   112        rzx=real(z)
2   113        rzy=aimag(z)
2   114        zc=conjg(z)
2   115        zck1=z-c
2   116        if(cabs(z).lt.1.0e-6)then
2   117         write(*,*)'   Singularity at x=',xval,'y=',yval
2   118         do 0043 m=1,3
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
3   119         sig(i,j,m)=1.0e35
3   120   0043  continue
2   121         go to 0045
2   122        endif
2   123        if(cabs(zck3).lt.1.0e-6)then
2   124         write(*,*)'   Singularity at x=',xval,'y=',yval
2   125         do 0044 m=1,3
3   126         sig(i,j,m)=1.0e35
3   127   0044  continue
2   128         go to 0045
2   129        endif
2   130        write(*,*)'   '
2   131        write(*,*)'   '
2   132        write(*,*)'    Working on Point',i,',',j
2   133        write(*,*)'    Coords =',xval,yval
2   134 c
2   135 c      Start Loop over defined dislocations
2   136 c
2   137        sig1=0.0
2   138        sig2=0.0
2   139        sig3=0.0
2   140        if(nmic.le.0)go to 0024
2   141        do 0023 iz=1,ic
3   142 c
3   143 c
3   144 c      Check if chosen point is too close to singularity
3   145 c
3   146        zck1=z-z01(iz)
3   147        zck2=z-z02(iz)
3   148        if(cabs(zck1).lt.1.0e-6)then
3   149         write(*,*)'   Singularity at x=',xval,'y=',yval
3   150         do 0041 m=1,3
4   151         sig(i,j,m)=1.0e35
4   152   0041  continue
3   153         go to 0045
3   154        endif
3   155        if(cabs(zck2).lt.1.0e-6)then
3   156         write(*,*)'   Singularity at x=',xval,'y=',yval
3   157         do 0042 m=1,3
4   158         sig(i,j,m)=1.0e35
4   159   0042  continue
3   160         go to 0045
3   161        endif
3   162 c
3   163 c      Compute the stresses at the prescribed point
3   164 c
3   165        call PHI(z,fz1)
3   166        call KSI(z,fz2)
3   167        call PHIP(z,fz3)
3   168        dum=zc*fz3+fz2
3   169        sig2d=2.0*real(fz1)+real(dum)
3   170        sig1d=4.0*real(fz1)-sig2d
3   171        sig3d=aimag(dum)
3   172        sig1=sig1-sig1d
3   173        sig2=sig2-sig2d
3   174        sig3=sig3-sig3d
3   175   0023 continue
2   176   0024 continue
2   177 c
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
2   178 c      Compute stresses due to applied loading
2   179 c
2   180        if(irl.gt.0)then
2   181         if(abs(xval).gt.0.00001)then
2   182          th=yval/xval
2   183          th=atan(th)
2   184          if(xval.lt.0.0.and.yval.gt.0.0)th=pi+th
2   185          if(xval.lt.0.0.and.yval.lt.0.0)th=th-pi
2   186          th=th/2.0
2   187         else
2   188          th=pi/4.0
2   189          th=sign(th,yval)
2   190         endif
2   191         th3=3.0*th
2   192         rad=cabs(z)
2   193         fctr=KI/sqrt(2.0*pi*rad)
2   194         s1t=sin(th)
2   195         s3t=sin(th3)
2   196         c1t=cos(th)
2   197         c3t=cos(th3)
2   198         sig1=sig1+(fctr*(c1t*(1.0-s1t*s3t)))
2   199         sig2=sig2+(fctr*(c1t*(1.0+s1t*s3t)))
2   200         sig3=sig3+(fctr*(s1t*c1t*c3t))
2   201 c
2   202         fctr=KII/sqrt(2.0*pi*rad)
2   203         sig1=sig1-(fctr*(s1t*(2.0+c1t*c3t)))
2   204         sig2=sig2+(fctr*s1t*c1t*c3t)
2   205         sig3=sig3+(fctr*(c1t*(1.0-s1t*s3t)))
2   206        endif
2   207        sig(i,j,2)=sig2
2   208        sig(i,j,1)=sig1
2   209        sig(i,j,3)=sig3
2   210        write(*,*)'    Location     X=',xval,'    Y=',yval
2   211        write(*,*)'    Sxx=',sig1,'      Syy=',sig2,'      Sxy=',sig3
2   212        if(abs(smy).lt.abs(sig2))smy=sig(i,j,2)
2   213        if(abs(smx).lt.abs(sig1))smx=sig(i,j,1)
2   214        if(abs(smxy).lt.abs(sig3))smxy=sig(i,j,3)
2   215  0045 continue
1   216  0035 continue
    217 c
    218 c      Prepare to plot the results
    219 c
    220  0050 write(*,*)'     Give Ioport,Model,Fact'
    221        read(*,*,err=0050)ioport,model,fact
    222  0055 write(*,*)'    Sxx -> 1,     Syy -> 2,        Sxy -> 3'
    223        write(*,*)'              or  Principle Stresses'
    224        write(*,*)'    Maximum ->4,  Minimum ->5,  Deviatoric ->6'
    225        read(*,*,err=0055)ip
    226        if(ip.lt.4)then
    227         do 0090 i=1,nx
1   228         do 0095 j=1,ny
2   229         zp(i,j)=sig(i,j,ip)
2   230  0095   continue
1   231  0090   continue
    232        endif
    233
    234 c
    235 c      Compute the Principle Stresses
    236 c
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
    237        if(ip.ge.4)then
    238         smax=0.0
    239         do 0120 i=1,nx
1   240         do 0125 j=1,ny
2   241         sig1=sig(i,j,1)
2   242         sig2=sig(i,j,2)
2   243         sig3=sig(i,j,3)
2   244         call prince(sig1,sig2,sig3,sip1,sip2,sdev)
2   245         if(ip.eq.4)zp(i,j)=sip1
2   246         if(ip.eq.5)zp(i,j)=sip2
2   247         if(ip.eq.6)zp(i,j)=sdev
2   248         if(abs(zp(i,j)).ge.1.0e12)go to 0125
2   249         if(abs(zp(i,j)).ge.smax)smax=abs(zp(i,j))
2   250  0125   continue
1   251  0120   continue
    252        endif
    253        if(ip.eq.1)then
    254         write(*,*)'   The maximum value is =',smx
    255        elseif(ip.eq.2)then
    256         write(*,*)'   The maximum value is =',smy
    257        elseif(ip.eq.3)then
    258         write(*,*)'   The maximum value is =',smxy
    259        else
    260         write(*,*)'   The maximum value is =',smax
    261        endif
    262  0070 write(*,*)'  -- Give Contour Information --'
    263        write(*,*)'  #levs, min lev, max lev, # betw labels, # first '
    264        read(*,*,err=0070)nlev,vmin,vmax,nbl,nst
    265        dlev=(vmax-vmin)/float(nlev)
    266        il=nbl-nst
    267        do 0075 i=1,nlev
1   268        zlev(i)=vmin+(dlev*float(i))
1   269        il=il+1
1   270        if(il.eq.nbl)then
1   271         ldig(i)=-1
1   272         lwgt(i)=2
1   273         il=0
1   274        else
1   275         ldig(i)=-2
1   276         lwgt(i)=1
1   277        endif
1   278  0075 continue
    279        xhigh=5.0
    280        yhigh=5.0
    281        hgt=0.07
    282        narc=3
    283        ndiv=2
    284        call plots(0,ioport,model)
    285        call factor(fact)
    286        call zcseg(zp,nnx,nny,nx,ny,xlow,ylow,xhigh,yhigh,zlev,ldig,
    287       1     lwgt,nlev,hgt,ndiv,narc)
    288        do 0130 i=1,nlev
1   289        zlev(i)=-1.0*zlev(i)
1   290        lwgt(i)=-1.0*lwgt(i)
1   291  0130 continue
    292 c      call stline(-1,.1389,0.)
    293        call zcseg(zp,nnx,nny,nx,ny,xlow,ylow,xhigh,yhigh,zlev,ldig,
    294       1     lwgt,nlev,hgt,ndiv,narc)
    295 c      call stline(1,.1389,0.)
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
    296        xln=xhigh-xlow
    297        yln=yhigh-ylow
    298        xdel=(xe-xs)/(xhigh-xlow)
    299        ydel=(ye-ys)/(yhigh-ylow)
    300        hgt1=.1
    301        hgt2=.1
    302        call staxis(hgt1,hgt2,hgt1,hgt2,2)
    303        call axis(xlow,ylow,' ',-1,xln,0.0,xs,xdel)
    304        call axis(xlow,ylow,' ',1,yln,90.0,ys,ydel)
    305        call axis(xlow,yhigh,' ',1,xln,0.0,xs,xdel)
    306        call axis(xhigh,ylow,' ',-1,yln,90.0,ys,ydel)
    307        call plot(0.,0.,999)
    308   0100 write(*,*)'   3-D Plot  -> 1'
    309        read(*,*,err=0100)i3d
    310        if(i3d.gt.0)then
    311   0110  write(*,*)'   Give horiz, vert, angles'
    312         read(*,*,err=0110)ahor,aver
    313         call plots(0,ioport,model)
    314         call factor(fact)
    315         iproj=0
    316         itrim=0
    317         idir=3
    318         zlow=1.0e35
    319         iframe=1
    320         icut=3
    321         call mesh(zp,nnx,nny,nx,ny,ahor,aver,xlow,ylow,xhigh,
    322        1      yhigh,iedge,idir,iproj,iframe,zlow,icut,
    323        2      itrim,mask,vertex)
    324         call plot(0.,0.,999)
    325        endif
    326   0080 write(*,*)'   Try a different plot  -> 1'
    327        read(*,*,err=0080)inew
    328        if(inew.gt.0)go to 0050
    329        end

Name     Type         Offset P Class

A        COMPLEX        800   /DAM   /
ABS                           INTRINSIC
AHOR     REAL         21432
AIMAG                         INTRINSIC
ATAN                          INTRINSIC
AVER     REAL         21436
B1       REAL         20960
B2       REAL         20964
C        REAL             0   /CRACK /
C1T      REAL         21300
C2T      REAL         21304
CABS                          INTRINSIC
CMPLX                         INTRINSIC
CONJG                         INTRINSIC
COS                           INTRINSIC
DEN      REAL         20936
DLEV     REAL         21372
DUM      COMPLEX      20904
DUM1     COMPLEX      20952
DX       REAL         21062
DY       REAL         21066
DZ       COMPLEX      *****
```

D Line# 1 7

EMOD	REAL	0	/MAT /
FACT	REAL	21312	
FCTR	REAL	21288	
FLOAT			INTRINSIC
FZ1	COMPLEX	21220	
FZ2	COMPLEX	21228	
FZ3	COMPLEX	21236	
GMOD	REAL	4	/MAT /
HGT	REAL	21392	
HGT1	REAL	21422	
HGT2	REAL	21426	
I	INTEGER*2	20912	
I1	INTEGER*2	21376	
I3D	INTEGER*2	21430	
IC	INTEGER*2	20902	
ICUT	INTEGER*2	21452	
IDIR	INTEGER*2	21444	
IEDGE	INTEGER*2	21454	
IFRAME	INTEGER*2	21450	
INEW	INTEGER*2	21456	
IOPORT	INTEGER*2	21308	
IP	INTEGER*2	21316	
IPL	INTEGER*2	20896	
IPROJ	INTEGER*2	21440	
IRL	INTEGER*2	21024	
ITRIM	INTEGER*2	21442	
IZ	INTEGER*2	1200	/DAM /
J	INTEGER*2	20976	
K	REAL	12	/MAT /
KI	REAL	21034	
KII	REAL	21038	
L	INTEGER*2	21090	
LDIG	INTEGER*2	14680	
LWGT	INTEGER*2	14780	
M	INTEGER*2	21148	
MASK	INTEGER*2	14880	
MODEL	INTEGER*2	21310	
NARC	INTEGER*2	21396	
NBL	INTEGER*2	21368	
NDIS	INTEGER*2	20900	
NDIV	INTEGER*2	21398	
NLEV	INTEGER*2	21358	
NMIC	INTEGER*2	20898	
NNX	INTEGER*2	20880	
NNY	INTEGER*2	20882	
NST	INTEGER*2	21370	
NU	REAL	8	/MAT /
NX	INTEGER*2	21058	
NY	INTEGER*2	21060	
PI	REAL	20892	
RAD	REAL	21272	
REAL			INTRINSIC
RXY	REAL	21030	
RYY	REAL	21026	
RZX	REAL	21112	
RZY	REAL	21116	
S1T	REAL	21292	
S3T	REAL	21296	
SDEV	REAL	21354	

D Line# 1 7

SIG	REAL	3816	
SIG1	REAL	21162	
SIG1D	REAL	21256	
SIG2	REAL	21166	
SIG2D	REAL	21252	
SIG3	REAL	21170	
SIG3D	REAL	21260	
SIGN			INTRINSIC
SIN			INTRINSIC
SIP1	REAL	21346	
SIP2	REAL	21350	
SMAX	REAL	21330	
SMX	REAL	21078	
SMXY	REAL	21086	
SMY	REAL	21082	
SQRT			INTRINSIC
TH	REAL	21264	
TH3	REAL	21268	
THETA	REAL	20940	
VERTEX	REAL	14616	
VMAX	REAL	21364	
VMIN	REAL	21360	
XD	REAL	20944	
XDEL	REAL	21414	
XDIS	REAL	20968	
XE	REAL	20928	
XEC	REAL	20992	
XHIGH	REAL	21384	
XLN	REAL	21406	
XLOW	REAL	20884	
XS	REAL	20920	
XSC	REAL	20984	
XVAL	REAL	21108	
YD	REAL	20948	
YDEL	REAL	21418	
YDIS	REAL	20972	
YE	REAL	20932	
YEC	REAL	20996	
YHIGH	REAL	21388	
YLN	REAL	21410	
YLOW	REAL	20888	
YS	REAL	20924	
YSC	REAL	20988	
YVAL	REAL	21104	
Z	COMPLEX	21070	
Z01	COMPLEX	0	/DAM /
Z02	COMPLEX	400	/DAM /
ZC	COMPLEX	21120	
ZCX1	COMPLEX	21180	
ZCX2	COMPLEX	21188	
ZCX3	COMPLEX	21128	
ZE	COMPLEX	21050	
ZLEV	REAL	3616	
ZLOW	REAL	21446	
ZP	REAL	16	
ZS	COMPLEX	21042	

330 c

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
    331 c
    332         Subroutine PHI(z,fzt)
    333        Complex*8 z,z01(50),z02(50),fz,fzt,a(50),an
    334        Common /dam/z01,z02,a,iz
    335        an=a(iz)
    336        call Phi2(an,z,z01(iz),fz)
    337        fzt=(an/(z-z01(iz)))+fz
    338 c
    339 c      Add Negative Burgess Vector
    340 c
    341        an=-1.0*a(iz)
    342        call Phi2(an,z,z02(iz),fz)
    343        fzt=fzt+fz+(an/(z-z02(iz)))
    344        return
    345        end
```

Name	Type	Offset	P	Class
A	COMPLEX	800		/DAM /
AN	COMPLEX	21458		
FZ	COMPLEX	21466		
FZT	COMPLEX	4	*	
IZ	INTEGER*2	1200		/DAM /
Z	COMPLEX	0	*	
Z01	COMPLEX	0		/DAM /
Z02	COMPLEX	400		/DAM /

```
    346 c
    347 c
    348 c
    349         Subroutine Phi2(an,z,zo,fzt)
    350        Complex*8 an,ac,z,zo,fzt,fz1,fz2,fz3,Xz
    351        ac=conjg(an)
    352        zoc=conjg(zo)
    353        call fX(z,Xz)
    354        call F(z,zo,fz1)
    355        call F(z,zoc,fz2)
    356        call G(z,zoc,fz3)
    357 c      fzt=-1.0*((an*fz1)+(an*fz2)+(ac*(zo-zoc)*fz3))
    358        fzt=-1.0*((an*fz1)+(an*fz2)+(ac*(zo-zoc)*fz3)-(an*Xz))
    359        return
    360        end
```

Name	Type	Offset	P	Class
AC	COMPLEX	21498		
AN	COMPLEX	0	*	
CONJG				INTRINSIC
FZ1	COMPLEX	21518		
FZ2	COMPLEX	21526		
FZ3	COMPLEX	21534		
FZT	COMPLEX	12	*	
XZ	COMPLEX	21510		
Z	COMPLEX	4	*	
ZO	COMPLEX	8	*	
ZOC	REAL	21506		

```
    361 c
    362 c
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
    363 c
    364         Subroutine F(z,zo,fz)
    365        Complex*8 z,zo,fz,Xz,Xzo
    366        call fX(z,Xz)
    367        call fX(zo,Xzo)
    368        fz=(1.0-(Xz/Xzo))/(2.0*(z-zo))
    369        return
    370        end
```

Name	Type	Offset	P	Class
FZ	COMPLEX	8	*	
XZ	COMPLEX	21590		
XZO	COMPLEX	21598		
Z	COMPLEX	0	*	
ZO	COMPLEX	4	*	

```
    371 c
    372 c
    373 c
    374         Subroutine G(z,zo,fz)
    375        Complex*8 z,zo,fz,Xz,Xzo,XPzo
    376        call fX(z,Xz)
    377        call fX(zo,Xzo)
    378        call fXP(zo,XPzo)
    379        fz=1.0-(Xz/Xzo)+(((z-zo)*Xz*XPzo)/(Xzo*Xzo))
    380        fz=fz/(2.0*(z-zo)*(z-zo))
    381        return
    382        end
```

Name	Type	Offset	P	Class
FZ	COMPLEX	8	*	
XPZO	COMPLEX	21646		
XZ	COMPLEX	21630		
XZO	COMPLEX	21638		
Z	COMPLEX	0	*	
ZO	COMPLEX	4	*	

```
    383 c
    384 c
    385 c
    386         Subroutine fX(z,Xz)
    387        Complex*8 Xz,z,zpc
    388        Common /crack/ c
    389        zpc=(z+c)*z
    390        Xz=1.0/csqrt(zpc)
    391        return
    392        end
```

Name	Type	Offset	P	Class
C	REAL	0		/CRACK /
CSQRT				INTRINSIC
XZ	COMPLEX	4	*	
Z	COMPLEX	0	*	
ZPC	COMPLEX	21718		

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
    393 c
    394 c
    395 c
    396         Subroutine fXP(z,XPz)
    397        Complex*8 XPz,z,zpc,za,zb
    398        Common /crack/ c
    399        zpc=z+c
    400        za=csqrt(z)
    401        za=za*za*za*csqrt(zpc)
    402        zb=csqrt(zpc)
    403        zb=csqrt(z)*zb*zb*zb
    404        XPz=-0.5*((1.0/za)+(1.0/zb))
    405        return
    406        end
```

Name	Type	Offset	P	Class
C	REAL	0		/CRACK /
CSQRT				INTRINSIC
XPZ	COMPLEX	4	*	
Z	COMPLEX	0	*	
ZA	COMPLEX	21758		
ZB	COMPLEX	21806		
ZPC	COMPLEX	21750		

```
    407 c
    408 c
    409 c
    410         Subroutine KSI(z,fzt)
    411        Complex*8 z,zc,zoc,z01(50),z02(50),fz1,fz2,fz3,fzt,a(50),an,ac
    412        Common /dam/z01,z02,a,iz
    413        an=a(iz)
    414        zc=conjg(z)
    415        call Phi2(an,zc,z01(iz),fz1)
    416        fz1=conjg(fz1)
    417        call Phi2(an,z,z01(iz),fz2)
    418        call Phi2P(an,z,z01(iz),fz3)
    419        fzt=fz1-fz2-(z*fz3)
    420        ac=conjg(an)
    421        zoc=conjg(z01(iz))
    422        fzt=fzt+(ac/(z-z01(iz)))+((an*zoc)/((z-z01(iz))*(z-z01(iz))))
    423 c
    424 c      Add Negative of Second Burgess Vector
    425 c
    426        an=-1.0*a(iz)
    427        call Phi2(an,zc,z02(iz),fz1)
    428        fz1=conjg(fz1)
    429        call Phi2(an,z,z02(iz),fz2)
    430        call Phi2P(an,z,z02(iz),fz3)
    431        fzt=fzt+fz1-fz2-(z*fz3)
    432        ac=conjg(an)
    433        zoc=conjg(z02(iz))
    434        fzt=fzt+(ac/(z-z02(iz)))+((an*zoc)/((z-z02(iz))*(z-z02(iz))))
    435        return
    436        end
```

Name	Type	Offset	P	Class

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
A       COMPLEX          800   /DAM   /
AC      COMPLEX        21926
AN      COMPLEX        21878
CONJG                          INTRINSIC
FZ1     COMPLEX        21894
FZ2     COMPLEX        21902
FZ3     COMPLEX        21910
FZT     COMPLEX            4 *
IZ      INTEGER*2       1200   /DAM   /
Z       COMPLEX            0 *
ZO1     COMPLEX            0   /DAM   /
ZO2     COMPLEX          400   /DAM   /
ZC      COMPLEX        21886
ZOC     COMPLEX        21934
```

```
    437 c
    438 c
    439 c
    440         Subroutine Phi2P(an,z,zo,fzt)
    441        Complex*8 an,ac,z,zo,fzt,fz1,fz2,fz3,XPz
    442        ac=conjg(an)
    443        zoc=conjg(zo)
    444        call fXP(z,XPz)
    445        call FP(z,zo,fz1)
    446        call FP(z,zoc,fz2)
    447        call GP(z,zoc,fz3)
    448        fzt=-1.0*(an*fz1+an*fz2+ac*(zo-zoc)*fz3-an*XPz)
    449 c      fzt=-1.0*(an*fz1+an*fz2+ac*(zo-zoc)*fz3)
    450        return
    451        end
```

```
Name    Type         Offset P Class

AC      COMPLEX       22022
AN      COMPLEX           0 *
CONJG                       INTRINSIC
FZ1     COMPLEX       22042
FZ2     COMPLEX       22050
FZ3     COMPLEX       22058
FZT     COMPLEX          12 *
XPZ     COMPLEX       22034
Z       COMPLEX           4 *
ZO      COMPLEX           8 *
ZOC     REAL          22030
```

```
    452 c
    453 c
    454 c
    455         Subroutine FP(z,zo,fz)
    456        Complex*8 z,zo,fz,Xz,Xzo,XPz
    457        call fX(z,Xz)
    458        call fX(zo,Xzo)
    459        call fXP(z,XPz)
    460        fz=1.0-(Xz/Xzo)+(((z-zo)*XPz)/Xzo)
    461        fz=fz/(-2.0*(z-zo)*(z-zo))
    462        return
    463        end
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
Name    Type        Offset P Class

FZ      COMPLEX          8 *
XPZ     COMPLEX      22130
XZ      COMPLEX      22114
XZO     COMPLEX      22122
Z       COMPLEX          0 *
ZO      COMPLEX          4 *


    464 c
    465 c
    466 c
    467         Subroutine GP(z,zo,fz)
    468        Complex*8 z,zo,fz,fz1,fz2,Xz,Xzo,XPz,XPzo
    469        call fX(z,Xz)
    470        call fX(zo,Xzo)
    471        call fXP(z,XPz)
    472        call fXP(zo,XPzo)
    473        fz1=1.0-(Xz/Xzo)+(((z-zo)*Xz*XPzo)/(Xzo*Xzo))
    474        fz1=fz1/((z-zo)*(z-zo)*(z-zo))
    475        fz2=((z-zo)*XPzo*XPz)/(Xzo*Xzo)
    476        fz2=fz2+((XPzo*Xz)/(Xzo*Xzo))-(XPz/Xzo)
    477        fz2=fz2/(2.0*(z-zo)*(z-zo))
    478        fz=fz2-fz1
    479        return
    480        end

Name    Type        Offset P Class

FZ      COMPLEX          8 *
FZ1     COMPLEX      22218
FZ2     COMPLEX      22290
XPZ     COMPLEX      22202
XPZO    COMPLEX      22210
XZ      COMPLEX      22186
XZO     COMPLEX      22194
Z       COMPLEX          0 *
ZO      COMPLEX          4 *


    481 c
    482 c
    483         Subroutine PHIP(z,fzt)
    484        Complex*8 z,z01(50),z02(50),fz,fzt,a(50),an
    485        Common /dam/z01,z02,a,iz
    486        an=a(iz)
    487        call Phi2P(an,z,z01(iz),fz)
    488        fzt=fz-(an/((z-z01(iz))*(z-z01(iz))))
    489        an=-1.0*a(iz)
    490        call Phi2P(an,z,z02(iz),fz)
    491        fzt=fzt+fz-(an/((z-z02(iz))*(z-z02(iz))))
    492        return
    493        end

Name    Type        Offset P Class

A       COMPLEX        800   /DAM   /
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
AN      COMPLEX       22386
FZ      COMPLEX       22394
FZT     COMPLEX           4 *
IZ      INTEGER*2      1200   /DAM   /
Z       COMPLEX           0 *
Z01     COMPLEX           0   /DAM   /
Z02     COMPLEX         400   /DAM   /


    494 c
    495 c
    496         Subroutine Prince(sig1,sig2,sig3,sip1,sip2,sdev)
    497        if(sig1.ge.1.0e12.or.sig2.ge.1.0e12.or.sig3.ge.1.0e12)then
    498         sip1=1.0e35
    499         sip2=1.0e35
    500         sdev=1.0e35
    501         go to 0999
    502        endif
    503        sdev=((sig1-sig2)/2.0)**2+sig3**2
    504        sdev=sqrt(sdev)
    505        s=(sig1+sig2)/2.0
    506        if(s.ge.0.0)then
    507         sip1=s+sdev
    508         sip2=s-sdev
    509        else
    510         sip1=s-sdev
    511         sip2=s+sdev
    512        endif
    513  0999 return
    514        end

Name    Type        Offset P Class

S       REAL         22442
SDEV    REAL            20 *
SIG1    REAL             0 *
SIG2    REAL             4 *
SIG3    REAL             8 *
SIP1    REAL            12 *
SIP2    REAL            16 *
SQRT                        INTRINSIC


Name    Type          Size   Class

AXIS                       SUBROUTINE
CRACK                 4    COMMON
DAM                1202    COMMON
F                          SUBROUTINE
FACTOR                     SUBROUTINE
FP                         SUBROUTINE
FX                         SUBROUTINE
FXP                        SUBROUTINE
G                          SUBROUTINE
GP                         SUBROUTINE
KSI                        SUBROUTINE
MAT                  16    COMMON
MESH                       SUBROUTINE
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

PHI	SUBROUTINE
PHI2	SUBROUTINE
PHI2P	SUBROUTINE
PHIP	SUBROUTINE
PLOT	SUBROUTINE
PLOTS	SUBROUTINE
PRINCE	SUBROUTINE
STAXIS	SUBROUTINE
STRESS	PROGRAM
ZCSEG	SUBROUTINE

Pass One No Errors Detected
514 Source Lines

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
      1 $Storage: 2
      2 $Debug
      3           Program KIJPlot
      4 c*************************************************************************
      5 c***** This program is designed to compute the near and far field   *
      6 c***** energy release rates using a complex potential formulation.  *
      7 c***** The near field is evaluated via the stress intensity factors *
      8 c***** and the far field is computed through a numerical integration*
      9 c*************************************************************************
     10 c
     11       Complex*8 z,zc,z01,z02,fz1,fz2,fz3,trans,
     12      1 a,dum1,dum,dum2,dum3,dum4,zs,ze,dz,an,Ktot,K12
     13       Real*4 k,sig(30,30,3),emod,nu,zp(30,30),zlev(50),vertex(16)
     14       Common /mat/ emod,gmod,nu,k
     15       Common /dam/z01,z02,a,iz
     16       Common /crack/ c
     17       Data nnx,nny,xlow,ylow /30,30,1.0,1.0/
     18 c
     19       pi=3.141592654
     20       write(*,*)'    Give material properties'
     21  0005 write(*,*)'   emod,nu,  0-> pl. stress  1-> pl. strain'
     22       read(*,*,err=0005)emod,nu,ipl
     23       if(ipl.eq.0)then
     24        k=(3.0-nu)/(1.0+nu)
     25       else
     26        k=3.0-4.0*nu
     27       endif
     28       gmod=emod/(2.0*(1.0+nu))
     29  0010 write(*,*)'    Give main crack length'
     30       read(*,*,err=0010)c
     31       write(*,*)'  Give rad,theta,csd,cod'
     32       read(*,*)rad,theta,bx,by
     33       theta=(theta*pi)/180.
     34 c
     35 c     Compute the Burgess Vector
     36 c
     37       ci=cos(theta)
     38       si=sin(theta)
     39       trans=cmplx(ci,si)
     40       dum=cmplx(0,1)
     41       a=cmplx(bx,by)
     42       a=trans*a
     43       a=(gmod/(pi*(k+1)))*a
     44       a=a/dum
     45 c
     46 c     Compute Nondimensionalizing factor in Ballarini's paper
     47 c
     48       fact=(rad**1.5)*(k+1.0)/(gmod*.01*sqrt(bx*bx+by*by))
     49       fact=fact*sqrt(pi/2.0)
     50       open(5,file='KI.dat',status='new')
     51       open(6,file='KII.dat',status='new')
     52       do 0050 i1=1,170
  1  53       beta=float(i1-1)*pi/180.0
  1  54       dum=cmplx(0,1)
  1  55       xs=rad*cos(beta)
  1  56       ys=rad*sin(beta)

  1  57       ye=ys+(0.01*sin(theta))
  1  58       xe=xs+(0.01*cos(theta))
  1  59       z01=cmplx(xs,ys)
```

```
D Line# 1      7                                   Microsoft FORTRAN77 V3.20 02/84
1    60        z02=cmplx(xe,ye)
1    61 c
1    62 c      Start computation of the Near Field Energy Release Rates
1    63 c
1    64        Ktot=cmplx(0.0,0.0)
1    65        an=a
1    66        ze=z01
1    67        dum=csqrt((ze+c)/ze)
1    68        dum=-1.0*an*(real(dum)-1.0)
1    69 c
1    70        dum2=csqrt(conjg(ze))
1    71        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+c)
1    72        dum1=(conjg(an)*aimag(ze)*c)/dum2
1    73        dum2=cmplx(0.0,1.0)
1    74        dum1=dum1*dum2
1    75        K12=-2.0*sqrt((2.0*pi)/c)*(dum+dum1)
1    76        an=-1.0*a
1    77        ze=z02
1    78        dum=csqrt((ze+c)/ze)
1    79        dum=-1.0*an*(real(dum)-1.0)
1    80 c
1    81        dum2=csqrt(conjg(ze))
1    82        dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+c)
1    83        dum1=(conjg(an)*aimag(ze)*c)/dum2
1    84        dum2=cmplx(0.0,1.0)
1    85        dum1=dum1*dum2
1    86        K12=K12-(2.0*sqrt((2*pi)/c)*(dum+dum1))
1    87        Ktot=Ktot+K12
1    88 c
1    89 c      Nondimensionalize by the factor in Ballarini's Paper
1    90 c
1    91        rKI=fact*real(Ktot)
1    92        rKII=-1.0*fact*aimag(Ktot)
1    93        ri1=float(i1-1)
1    94        write(5,*)ri1,rKI
1    95        write(6,*)ri1,rKII
1    96  0050 continue
     97        close(5)
     98        close(6)
     99        end
```

Name	Type	Offset	P	Class
A	COMPLEX	16		/DAM /
AIMAG				INTRINSIC
AN	COMPLEX	14792		
BETA	REAL	14764		
BX	REAL	14702		
BY	REAL	14706		
C	REAL	0		/CRACK /
CI	REAL	14710		
CMPLX				INTRINSIC
CONJG				INTRINSIC
COS				INTRINSIC
CSQRT				INTRINSIC
DUM	COMPLEX	14726		
DUM1	COMPLEX	14920		
DUM2	COMPLEX	14848		
DUM3	COMPLEX	*****		

Microsoft FORTRAN77 V3.20 02/84

D Line# 1 7

DUM4	COMPLEX	*****	
DZ	COMPLEX	*****	
EMOD	REAL	0	/MAT /
FACT	REAL	14758	
FLOAT			INTRINSIC
FZ1	COMPLEX	*****	
FZ2	COMPLEX	*****	
FZ3	COMPLEX	*****	
GMOD	REAL	4	/MAT /
I1	INTEGER*2	14762	
IPL	INTEGER*2	14692	
IZ	INTEGER*2	24	/DAM /
K	REAL	12	/MAT /
K12	COMPLEX	14960	
KTOT	COMPLEX	14784	
NNX	INTEGER*2	14676	
NNY	INTEGER*2	14678	
NU	REAL	8	/MAT /
PI	REAL	14688	
RAD	REAL	14694	
REAL			INTRINSIC
RI1	REAL	15136	
RKI	REAL	15128	
RKII	REAL	15132	
SI	REAL	14714	
SIG	REAL	3876	
SIN			INTRINSIC
SQRT			INTRINSIC
THETA	REAL	14698	
TRANS	COMPLEX	14718	
VERTEX	REAL	3812	
XE	REAL	14780	
XLOW	REAL	14680	
XS	REAL	14768	
YE	REAL	14776	
YLOW	REAL	14684	
YS	REAL	14772	
Z	COMPLEX	*****	
Z01	COMPLEX	0	/DAM /
Z02	COMPLEX	8	/DAM /
ZC	COMPLEX	*****	
ZE	COMPLEX	14800	
ZLEV	REAL	3612	
ZP	REAL	12	
ZS	COMPLEX	*****	

Name	Type	Size	Class
CRACK		4	COMMON
DAM		26	COMMON
XLJPLO			PROGRAM
MAT		16	COMMON

Pass One No Errors Detected
99 Source Lines

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
      1
      2 $Storage: 2
      3 $Debug
      4           Program ksmin
      5 c*************************************************************************
      6 c***** This program is designed to compute the stress intensity     *
      7 c***** factor at various locations in front of the crack tip for a  *
      8 c***** smoothed damage distribution.                                *
      9 c*************************************************************************
     10 c
     11       Complex*8 z,zst,znd,zinc,zc,z01(400),z02(400),
     12      1 a(200),dum1,dum,dum2,zs,ze,dz,an,Ktot,K12,ft3
     13       Real*4 k,nu,xs(400),xe(400),ys(400),ye(400),b1(400),b2(400),
     14      1 vol(20,8)
     15       pi=3.141592654
     16       write(*,*)'    Give material properties'
     17  0005 write(*,*)'   emod,nu,  0-> pl. stress  1-> pl. strain'
     18       read(*,*,err=0005)emod,nu,ipl
     19       if(ipl.eq.0)then
     20        k=(3.0-nu)/(1.0+nu)
     21       else
     22        k=3.0-4.0*nu
     23       endif
     24       gmod=emod/(2.0*(1.0+nu))
     25  0010 write(*,*)'    Give main crack length'
     26       read(*,*,err=0010)c
     27  0007 write(*,*)'   Damage zone length'
     28       read(*,*,err=0007)xmax
     29  0038 write(*,*)'   Give  sig-xy'
     30       read(*,*,err=0038)sigxy
     31       rinit=sigxy*sqrt(pi*0.5*c)
     32       write(*,*)'  '
     33  0039   write(*,*)'  Give Approximate Dislocation #'
     34       read(*,*,err=0039)nfudge
     35       phi=(20.0*pi)/180.0
     36       open(5,file='vol.out',status='old')
     37       vtot=0.0
     38       do 0006 j=1,8
  1  39       do 0008 i=1,20
  2  40       read(5,*)vol(i,j)
  2  41       vtot=vtot+vol(i,j)
  2  42  0008 continue
  1  43  0006 continue
     44       write(*,*)'  total volume =',vtot
     45       vtot=vtot/float(nfudge)
     46       close(5)
     47 c
     48 c      symmetrize the volume distributions
     49 c
     50       do 0130 i=1,20
  1  51       vol(i,1)=0.5*(vol(i,1)+vol(i,8))
  1  52       vol(i,8)=vol(i,1)
  1  53       vol(i,2)=0.5*(vol(i,2)+vol(i,7))
  1  54       vol(i,7)=vol(i,2)
  1  55       vol(i,3)=0.5*(vol(i,3)+vol(i,6))
  1  56       vol(i,6)=vol(i,3)
  1  57       vol(i,4)=0.5*(vol(i,4)+vol(i,5))
  1  58       vol(i,5)=vol(i,4)
  1  59  0130 continue
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
     60 c
     61 c      Compute an equivalent # of dislocations
     62 c
     63        ic=0
     64        xf=4.0/25.4
     65        yf=5.0/25.4
     66        open(5,file='dis.out',status='new')
     67 c
     68        do 0100 j=1,8
1    69        do 0110 i=1,20
2    70        xlc=xf*float(i-1)
2    71        xuc=xf*float(i)
2    72        ylc=-0.78740+(yf*float(j-1))
2    73        yuc=-0.78740+(yf*float(j))
2    74        ncdis=int(vol(i,j)/vtot)
2    75        if(ncdis.gt.0)then
2    76         ydel=(yuc-ylc)/(float(ncdis))
2    77         do 0115 il=1,ncdis
3    78         ic=ic+1
3    79         xs(ic)=xlc
3    80         xe(ic)=xuc
3    81         ys(ic)=ylc+(ydel*float(il-1))
3    82         if(yuc.lt.0.0.or.ylc.lt.0.0)then
3    83          zphi=-1.0*phi
3    84         else
3    85          zphi=phi
3    86         endif
3    87         ye(ic)=ys(ic)+((xuc-xlc)*tan(zphi))
3    88         if(abs(ys(ic)).lt.0.005)ys(ic)=sign(0.005,ys(ic))
3    89         if(abs(ye(ic)).lt.0.005)ye(ic)=sign(0.005,ye(ic))
3    90         write(5,*)xs(ic),ys(ic),xe(ic),ye(ic)
3    91 c
3    92 c      Enter the function for the Slippage function here
3    93 c
3    94          b1(ic)=0.0005*(400.0/float(nfudge))
3    95          b2(ic)=0.00
3    96  0115   continue
2    97        endif
2    98  0110 continue
1    99  0100 continue
    100        close(5)
    101        write(*,*)ic,' dislocations generated'
    102        open(5,file='kcrak.out',status='new')
    103        open(6,file='kdam.out',status='new')
    104        open(7,file='ktot.out',status='new')
    105        do 0016 j=1, 100
1   106        write(*,*)' working on ',j,'th point'
1   107        dum=cmplx(0,1)
1   108        do 0017 i=1,ic
2   109        xcen=(xmax*float(j-1))/100.0
2   110        xes=xe(i)-xcen
2   111        xss=xs(i)-xcen
2   112        yes=ye(i)
2   113        yss=ys(i)
2   114        den=xes-xss
2   115        if(den.lt.0.000001)then
2   116         theta=pi/2.0
2   117        else
2   118         theta=atan((yes-yss)/den)
```

```
2   119         endif
2   120         xd=cos(theta)
2   121         yd=sin(theta)
2   122         dum1=cmplx(xd,yd)
2   123         z01(i)=cmplx(xss,yss)
2   124         z02(i)=cmplx(xes,yes)
2   125         a(i)=cmplx(b1(i),b2(i))
2   126         a(i)=(gmod/(pi*(k+1)))*dum1*a(i)
2   127         a(i)=a(i)/dum
2   128  0017   continue
1   129 c
1   130 c       Start computation of the Stress Intensity Factor
1   131 c
1   132         cl=c+xcen
1   133         Ktot=cmplx(0.0,0.0)
1   134         Do 0030 i=1,ic
2   135         an=a(i)
2   136         ze=z01(i)
2   137         dum=csqrt((ze+cl)/ze)
2   138         dum=-1.0*an*(real(dum)-1.0)
2   139         dum2=csqrt(conjg(ze))
2   140         dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+cl)
2   141         dum1=(conjg(an)*aimag(ze)*cl)/dum2
2   142         dum2=cmplx(0.0,1.0)
2   143         dum1=dum1*dum2
2   144         K12=-2.0*sqrt((2*pi)/cl)*(dum+dum1)
2   145         an=-1.0*a(i)
2   146         ze=z02(i)
2   147         dum=csqrt((ze+cl)/ze)
2   148         dum=-1.0*an*(real(dum)-1.0)
2   149         dum2=csqrt(conjg(ze))
2   150         dum2=2.0*dum2*dum2*dum2*csqrt(conjg(ze)+cl)
2   151         dum1=(conjg(an)*aimag(ze)*cl)/dum2
2   152         dum2=cmplx(0.0,1.0)
2   153         dum1=dum1*dum2
2   154         K12=K12-(2.0*sqrt((2*pi)/cl)*(dum+dum1))
2   155         Ktot=Ktot+K12
2   156  0030   continue
1   157 c
1   158 c       Write out the Stress Intensity Factors
1   159 c
1   160         rKII=-1.0*aimag(Ktot)
1   161         rKII=rKII/rinit
1   162         rkcrak=sigxy*sqrt(pi*0.5*cl)
1   163         rkcrak=rkcrak/rinit
1   164         write(5,*)xcen,rkcrak
1   165         write(6,*)xcen,rKII
1   166         tot=rkcrak+rKII
1   167         write(7,*)xcen,tot
1   168         write(*,*)'   '
1   169         write(*,*)' Stress Intensity Factors:  '
1   170         write(*,*)'   Kdam=',rKII,'   Kcrack=',rkcrak,'   Ktot=',tot
1   171  0016   continue
    172         close(5)
    173         close(6)
    174         close(7)

    175         end
```

Name	Type	Offset	P	Class
A	COMPLEX	6416		
ABS				INTRINSIC
AIMAG				INTRINSIC
AN	COMPLEX	18452		
ATAN				INTRINSIC
B1	REAL	15056		
B2	REAL	16656		
C	REAL	18278		
CL	REAL	18434		
CMPLX				INTRINSIC
CONJG				INTRINSIC
COS				INTRINSIC
CSQRT				INTRINSIC
DEN	REAL	18386		
DUM	COMPLEX	18352		
DUM1	COMPLEX	18402		
DUM2	COMPLEX	18508		
DZ	COMPLEX	*****		
EMOD	REAL	18260		
FLOAT				INTRINSIC
FT3	COMPLEX	*****		
GMOD	REAL	18274		
I	INTEGER*2	18306		
I1	INTEGER*2	18340		
IC	INTEGER*2	18308		
INT				INTRINSIC
IPL	INTEGER*2	18268		
J	INTEGER*2	18304		
K	REAL	18270		
K12	COMPLEX	18612		
KTOT	COMPLEX	18438		
NCDIS	INTEGER*2	18334		
NFUDGE	INTEGER*2	18294		
NU	REAL	18264		
PHI	REAL	18296		
PI	REAL	18256		
REAL				INTRINSIC
RINIT	REAL	18290		
RKCRAK	REAL	18784		
RKII	REAL	18780		
SIGN				INTRINSIC
SIGXY	REAL	18286		
SIN				INTRINSIC
SQRT				INTRINSIC
TAN				INTRINSIC
THETA	REAL	18390		
TOT	REAL	18788		
VOL	REAL	14416		
VTOT	REAL	18300		
XCEN	REAL	18366		
XD	REAL	18394		
XE	REAL	12816		
XES	REAL	18370		
XF	REAL	18310		
XLC	REAL	18318		
XMAX	REAL	18282		

D Line# 1 7

XS	REAL	8016
XSS	REAL	18374
XUC	REAL	18322
YD	REAL	18398
YDEL	REAL	18336
YE	REAL	11216
YES	REAL	18378
YF	REAL	18314
YLC	REAL	18326
YS	REAL	9616
YSS	REAL	18382
YUC	REAL	18330
Z	COMPLEX	*****
Z01	COMPLEX	16
Z02	COMPLEX	3216
ZC	COMPLEX	*****
ZE	COMPLEX	18460
ZINC	COMPLEX	*****
ZND	COMPLEX	*****
ZPHI	REAL	18348
ZS	COMPLEX	*****
ZST	COMPLEX	*****

Name	Type	Size	Class
XSMIN			PROGRAM

Pass One No Errors Detected
175 Source Lines

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
      1 $DEBUG
      2 $STORAGE: 2
      3       Program Plotxy
      4 c*************************************************************************
      5 c***** This routine is designed to draw up to 5 files on an x-y plot
      6 c*************************************************************************
      7       Common/work/x1(500),y1(500),x2(500),y2(500)
      8       character*10 filenm
      9       character*1 itext,xtt(20),xtt(20),ytt(20)
     10       character*20 xtit,xtit,ytit,legd
     11       equivalence (xtt(1),xtit),(xtt(1),xtit),(ytt(1),ytit)
     12       data xmax/10.0/, ymax/8.0/, xax/7.0/, yax/5.0/, xmt/1.5/
     13 c
     14 c     Input General Information
     15 c
     16       istyle=0
     17  0007 write(*,*)'    Give Xmin,Xmax,Xlbl,Ymin,Ymax,Ylbl'
     18       read(*,*,err=0007)xmn,xmx,xax,ymn,ymx,yax
     19       iforce=1
     20       ymt=yax+0.1
     21       f1=7.0/xax
     22       f2=5.0/yax
     23       if(f1.le.f2)then
     24        fmin=f1
     25       else
     26        fmin=f2
     27       endif
     28  0055 format(' XXXXXXXXXXXXXXXXXXX')
     29       write(*,*)'   Give X - Axis title'
     30       read(*,0050)xtit
     31       write(*,0055)
     32  0050 format(a20)
     33       write(*,*)'   Give Y - Axis title'
     34       write(*,0055)
     35       read(*,0050)ytit
     36       write(*,*)'   Give Main Title'
     37       write(*,0055)
     38       read(*,0050)mtit
     39  0008 write(*,*) '  Select Output Device:'
     40       write(*,*)'          Terminal -> 0'
     41       write(*,*)'          Printer  -> 1'
     42       read(*,*,err=0008)iout
     43       if(iout.eq.0)then
     44        ioport=93
     45        model=93
     46       else
     47        ioport=0
     48        model=64
     49       endif
     50  0009 write(*,*)'   Give Plot Size Factor'
     51       read(*,*,err=0009)fact
     52       fact=fact*fmin
     53 c
     54 c     Start Large Loop for File Input
     55 c
     56       isym=-13
     57       ltf=1
     58       npen=1
     59       if(isym.eq.0)then
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
      60        lt=0
      61       elseif(isym.gt.0.and.isym.le.15)then
      62        lt=1
      63       elseif(isym.lt.0.and.isym.ge.-15)then
      64        isym=-1*isym
      65        lt=-1
      66       endif
      67       open(5,file='dis.out',status='old')
      68       np1=0
      69       np2=0
      70       do 0030 j=1,500
1     71       read(5,*,err=0027)d1x,d1y,d2x,d2y
1     72       if(d1y.gt.0.0)then
1     73        np1=np1+1
1     74        x1(np1)=d1x
1     75        y1(np1)=d1y+.02
1     76        x2(np1)=d1x
1     77        y2(np1)=-1.0*(d1y+.02)
1     78       endif
1     79       if(d2y.gt.0.0)then
1     80        np1=np1+1
1     81        x1(np1)=d2x
1     82        y1(np1)=d2y+.02
1     83        x2(np1)=d2x
1     84        y2(np1)=-1.0*(d2y+.02)
1     85       endif
1     86  0030 continue
      87  0027 continue
      88       close(5)
      89       np2=np1
      90 c
      91 c     Start plotting the distn data
      92 c
      93 c          Initialize input information
      94 c
      95       call plots(0,ioport,model)
      96       call factor(fact)
      97       call window(0.,0.,xmax,ymax)
      98 c
      99 c     Set plot origin at coords (1.25,1.25)
     100 c
     101       call plot(1.25,1.25,-3)
     102 c
     103 c     Determine scaling parameters
     104 c
     105       x1(np1+1)=xmn
     106       x1(np1+2)=(xmx-xmn)/xax
     107       y1(np1+1)=ymn
     108       y1(np1+2)=(ymx-ymn)/yax
     109       x2(np2+1)=xmn
     110       x2(np2+2)=(xmx-xmn)/xax
     111       y2(np2+1)=ymn
     112       y2(np2+2)=(ymx-ymn)/yax
     113 c
     114 c     Draw each of the curves
     115 c
     116       do 0031 il=1,2
1    117       if(il.eq.1)then
1    118        call stline(ltf,0.25,110.)
```

Appendix B

Representative Volume Data Base

"REPRESENTATIVE VOLUME" STUDY

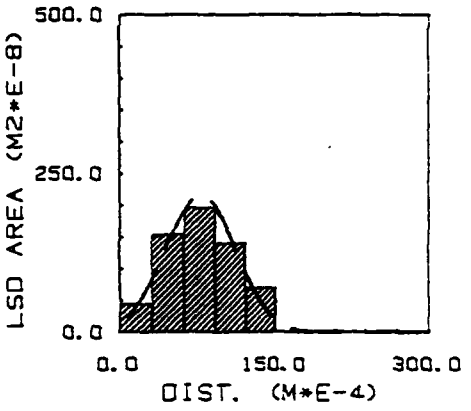
--- CLUSTER AREA DATA BASE ---

16. 18. 31. 20. 9. 6. 21. 36. 27. 25. 21. 34. 18. 21. 9. 15. 18. 22. 18. 16.
12. 18. 19. 20. 19. 17. 11. 15. 12. 8. 16. 14. 14. 16. 18. 11. 8. 15. 31. 28.
16. 18. 28. 19. 14. 11. 20. 18. 18. 27. 8. 20. 20. 30. 21. 16. 18. 20. 20. 26.
26. 12. 18. 15. 18. 18. 33. 6. 20. 23. 21. 27. 12. 25. 18. 15. 27. 35. 33. 22.
20. 13. 35. 18. 19. 15. 12. 25. 31. 14. 16. 20. 30. 22. 21. 19. 20. 18. 21. 16.
25. 17. 23. 25. 27. 20. 21. 18. 11. 21. 18. 10. 26. 33. 9. 18. 16. 19. 19. 16.
14. 15. 27. 13. 19. 12. 31. 33. 18. 21. 14. 18. 20. 36. 15. 20. 11. 10. 21. 17.
18. 18. 20. 25. 17. 32. 16. 17. 33. 34. 16. 21. 21. 14. 27. 19. 15. 27. 31. 16.
31. 19. 15. 23. 17. 16. 34. 20. 15. 27. 25. 17. 30. 16. 14. 17. 25. 18. 17. 15.
20. 32. 18. 19. 15. 17. 19. 36. 15. 24. 19. 22. 23. 11. 17. 29. 25. 33. 19. 15.
17. 19. 36. 15. 24. 21. 16. 25. 17. 23. 25. 27. 20. 21. 18. 11. 21. 18. 10. 18.
21. 16. 25. 17. 23. 25. 27. 20. 21. 18. 11. 21. 18. 10. 26. 33. 9. 18. 16. 19.
19. 16. 14. 9. 15. 17. 31. 13. 18. 23. 18. 30. 25. 19. 15. 21. 15. 18. 33. 16.
18. 22. 8. 16. 14. 14. 16. 18. 11. 8. 15. 31. 28. 16. 18. 28. 19. 14. 11. 20.
18. 18. 27. 8. 20. 20. 30. 21. 16. 18. 20. 20. 26. 7. 16. 18. 21. 20. 9. 6.
21. 33. 35. 16. 31. 19. 15. 23. 17. 16. 34. 20. 15. 27. 25. 20. 9. 6. 21. 33.
35. 29. 26. 25. 20. 18. 17. 30. 16. 14. 17. 25. 18. 17. 15. 16. 19. 22. 23. 11.
17. 29. 25. 33. 19. 15. 17. 19. 36. 15. 24. 20. 9. 6. 21. 33. 35. 29. 26. 25.
20. 18. 21. 16. 25. 33. 19. 15. 17. 19. 36. 15. 25. 19. 18. 16. 12. 18. 19. 20.
19. 17. 22. 24. 18. 11. 10. 3. 31. 16. 23. 19. 33. 35. 29. 26. 25. 19. 3. 25.

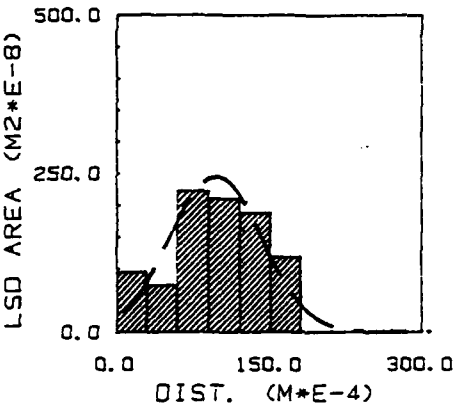
Appendix C

LSD Histogram Plots

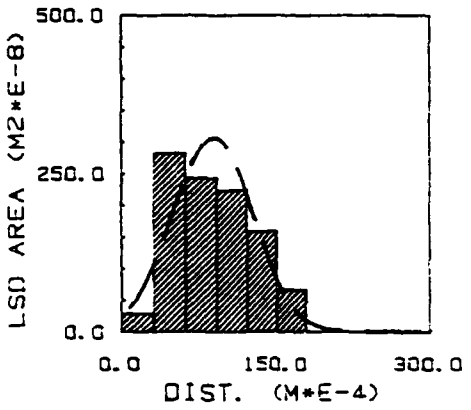
SLIDE 2-1-A



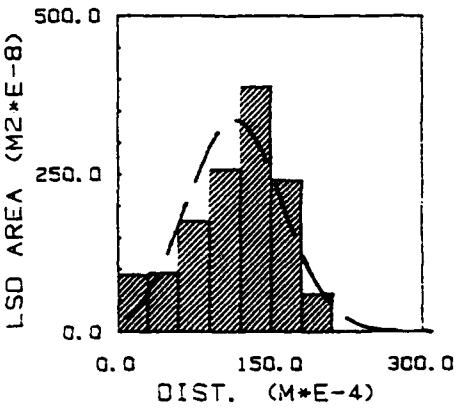
SLIDE 2-1-B



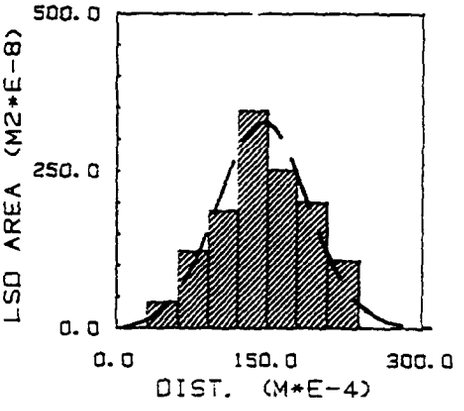
SLIDE 2-1-C



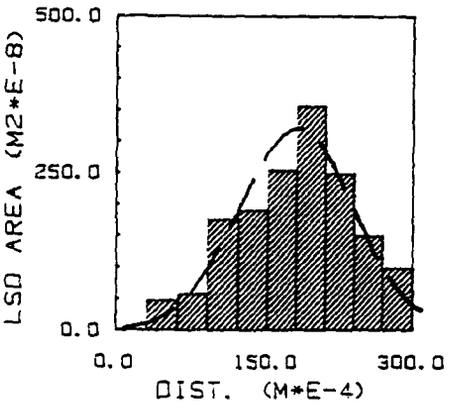
SLIDE 2-1-D



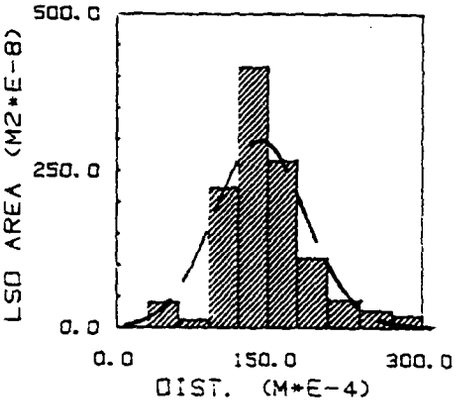
SLIDE 2-1-E



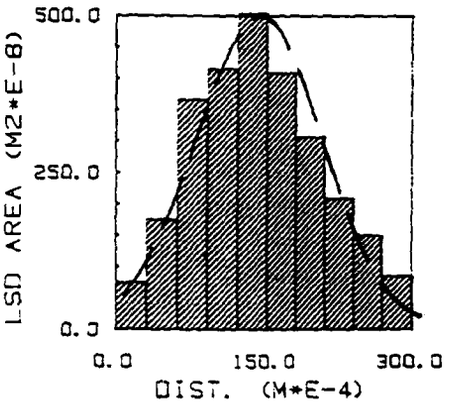
SLIDE 2-1-G



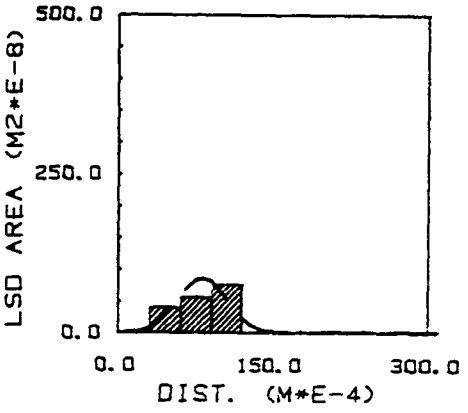
SLIDE 2-1-H



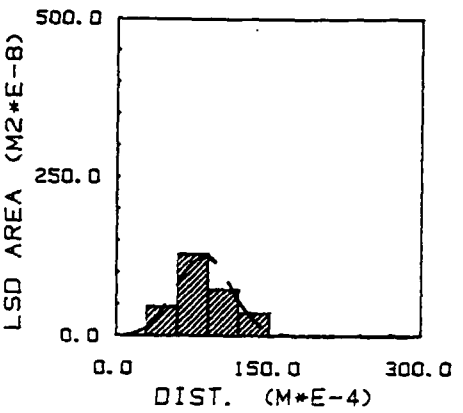
SLIDE 2-1-O



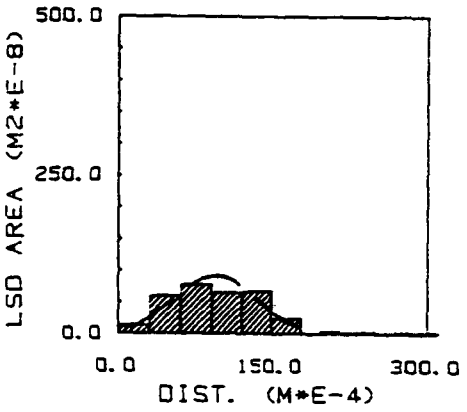
SLIDE 2-3-A



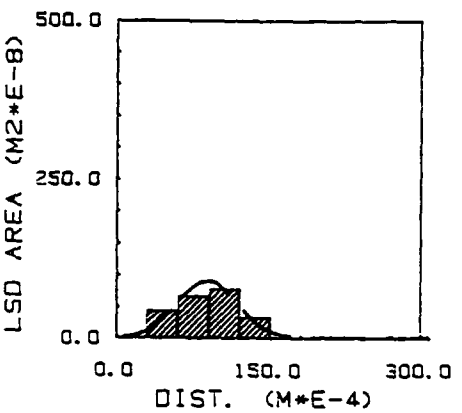
SLIDE 2-3-B



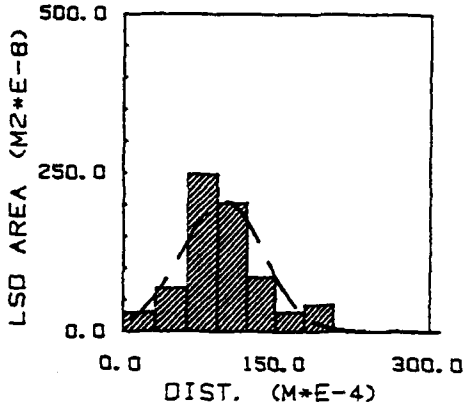
SLIDE 2-3-C



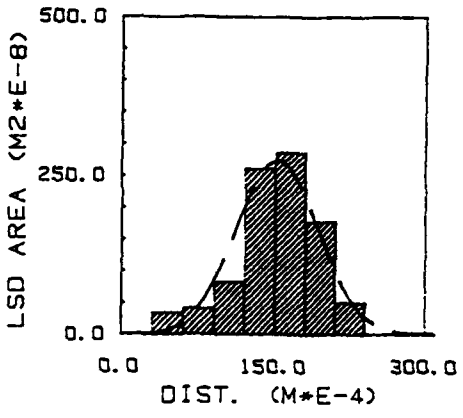
SLIDE 2-3-D



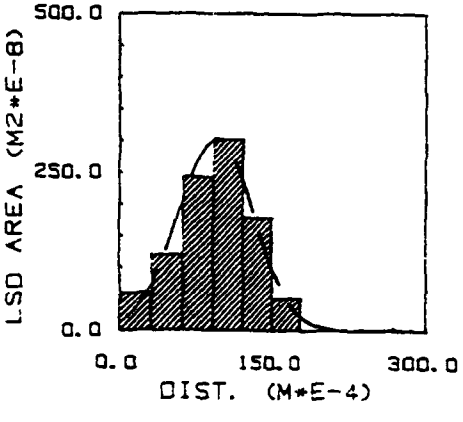
SLIDE 2-3-E



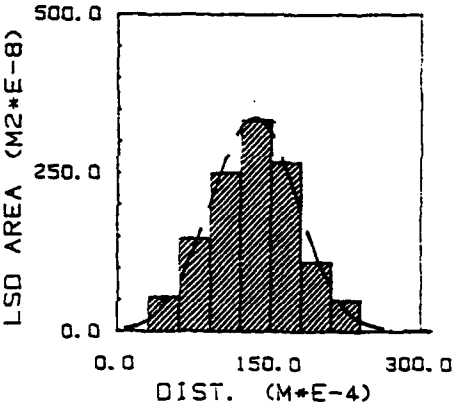
SLIDE 2-3-F



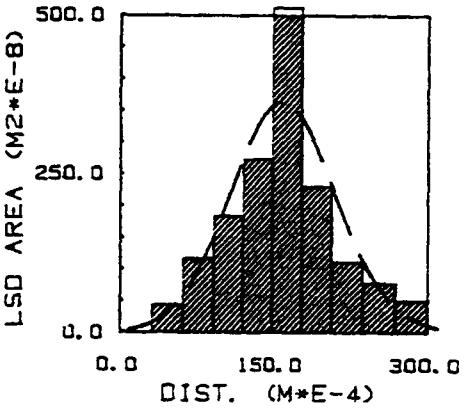
SLIDE 2-3-G



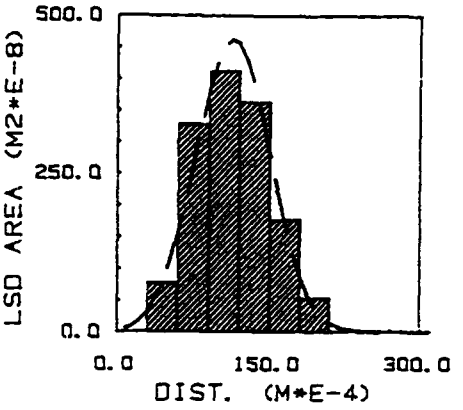
SLIDE 2-3-H



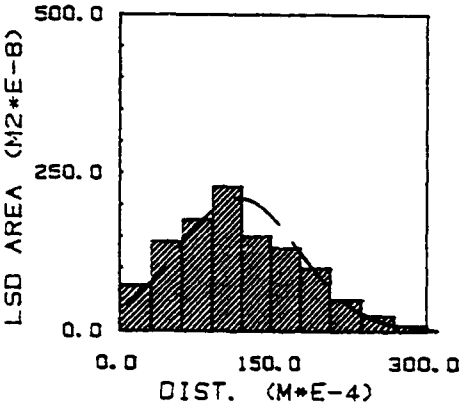
SLIDE 2-3-I



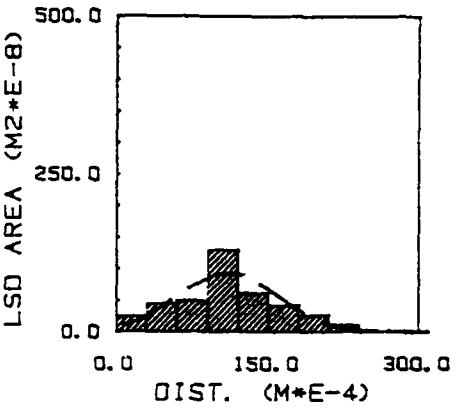
SLIDE 2-3-J



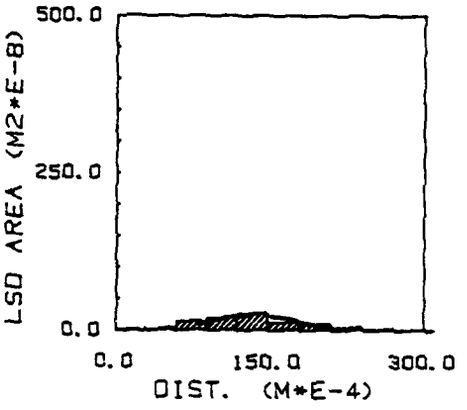
SLIDE 2-3-K



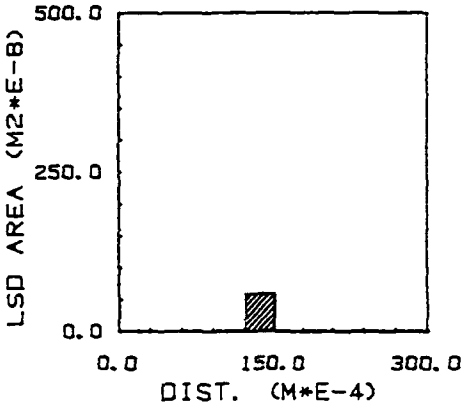
SLIDE 2-3-L



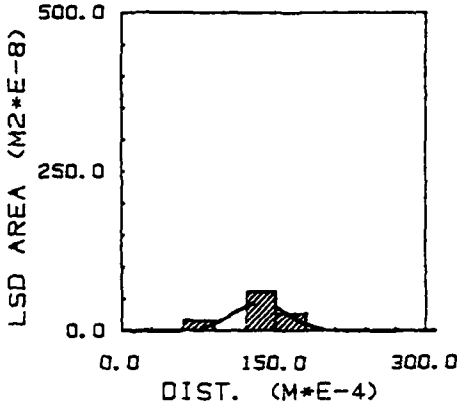
SLIDE 2-3-M



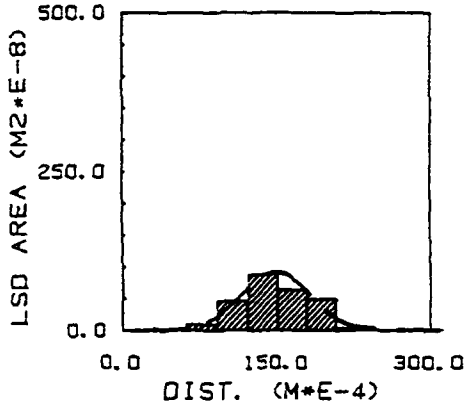
SLIDE 3-1-A



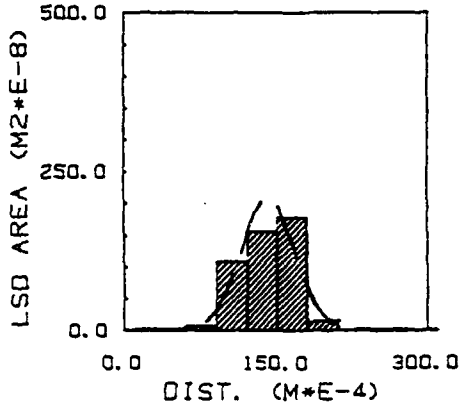
SLIDE 3-1-B



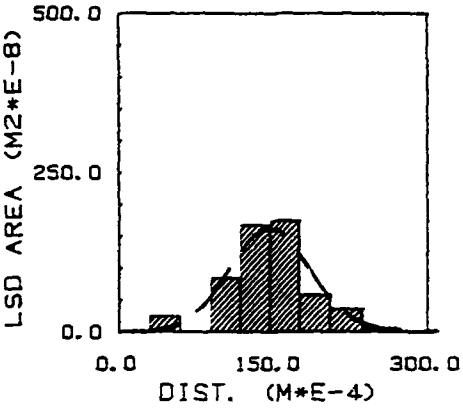
SLIDE 3-1-C



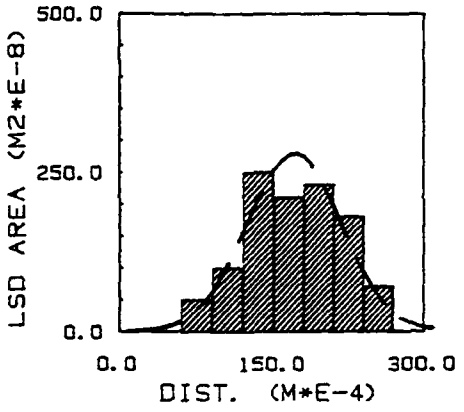
SLIDE 3-1-D



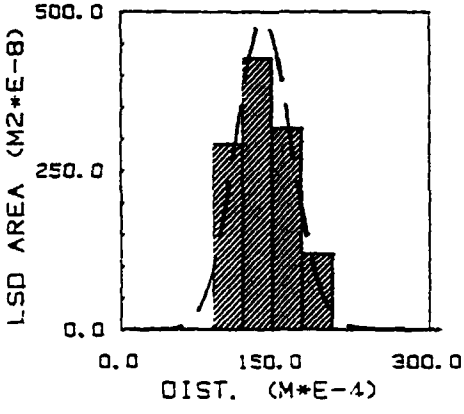
SLIDE 3-1-E



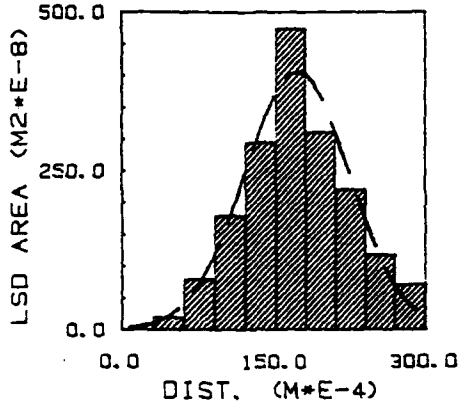
SLIDE 3-1-F



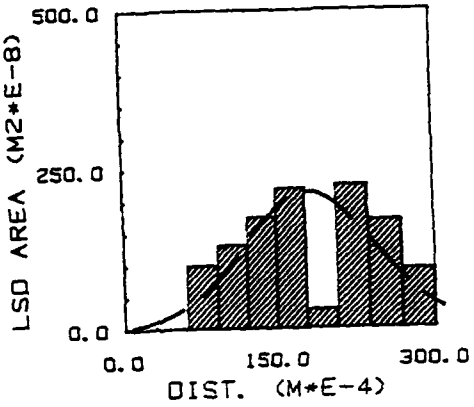
SLIDE 3-1-G



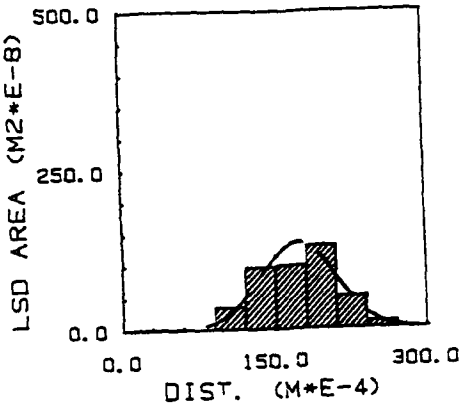
SLIDE 2-1-H



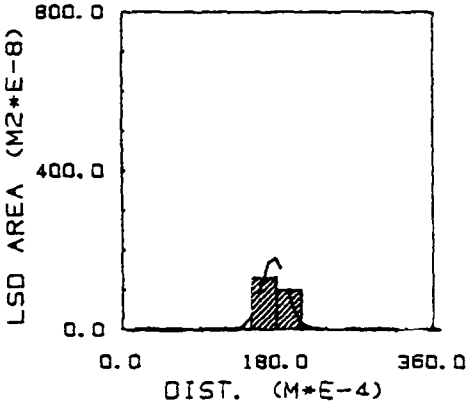
SLIDE 3-1-I



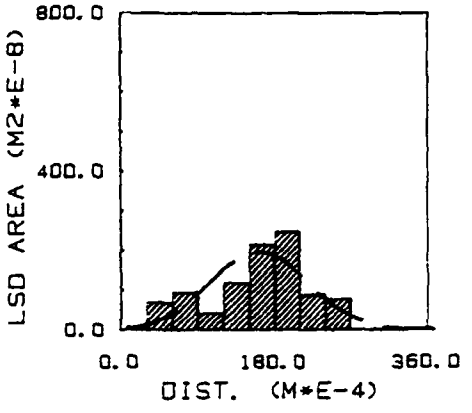
SLIDE 3-1-J



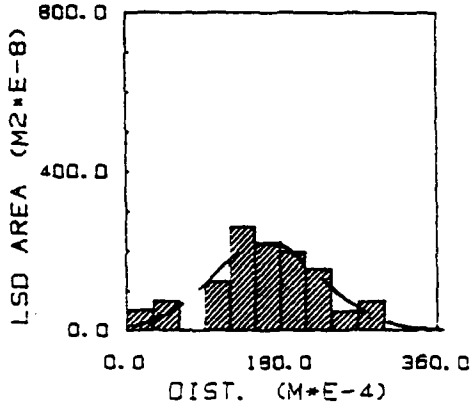
SLIDE 5-6-A



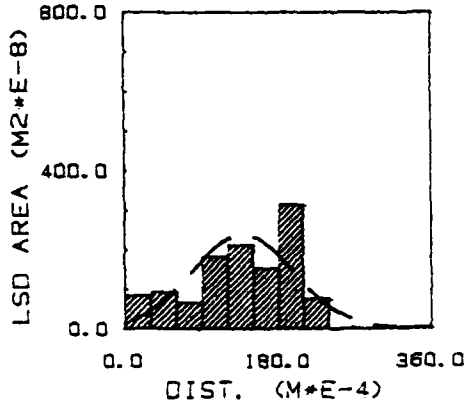
SLIDE 5-6-B



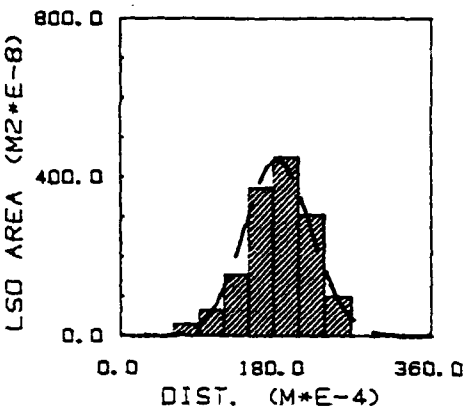
SLIDE 5-6-C



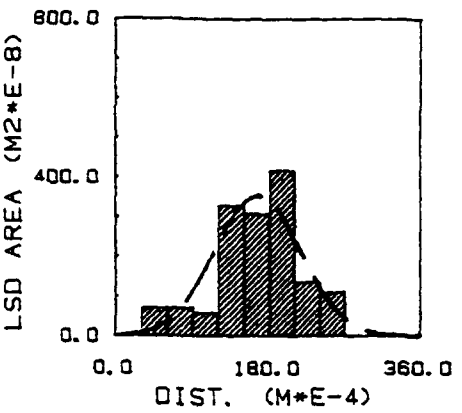
SLIDE 5-6-D



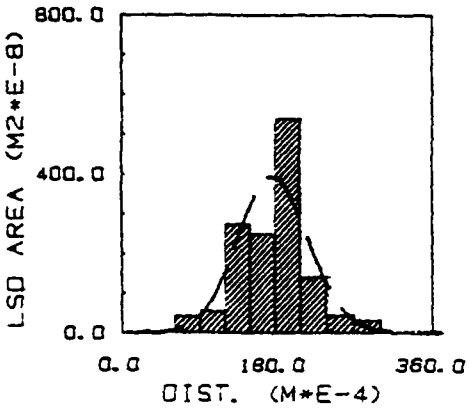
SLIDE 5-6-E



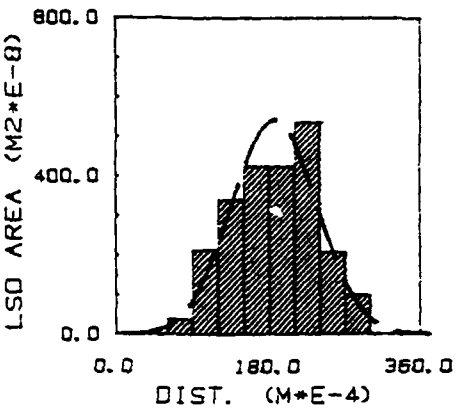
SLIDE 5-6-F



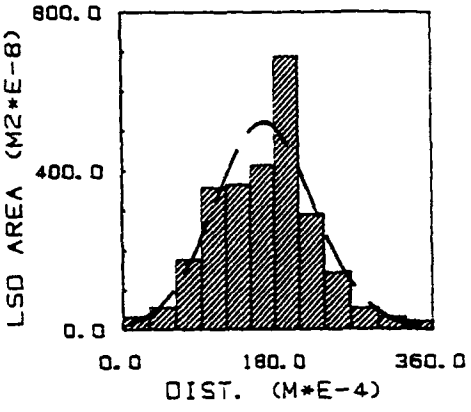
SLIDE 5-6-G



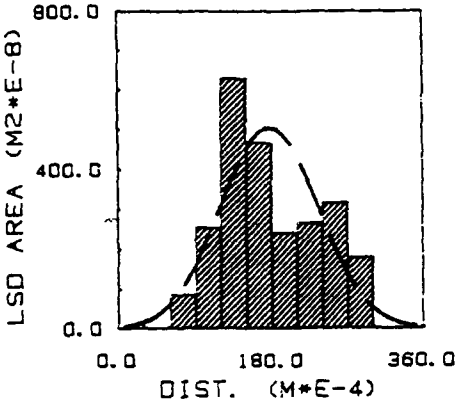
SLIDE 5-6-H



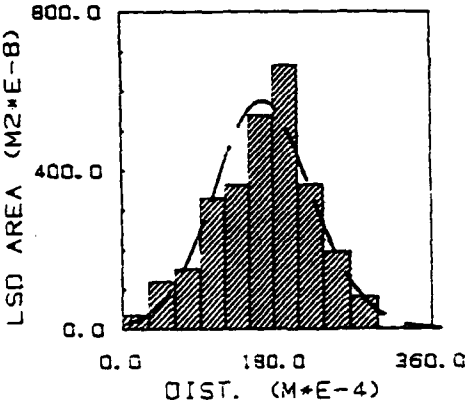
SLIDE 5-6-I



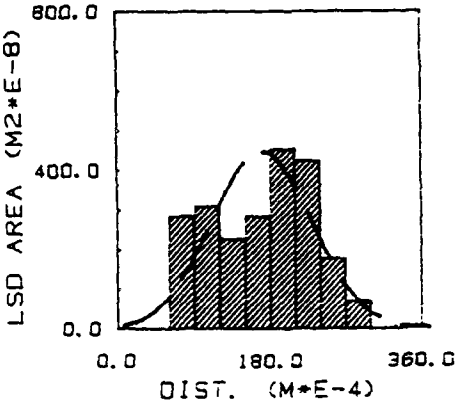
SLIDE 5-6-J



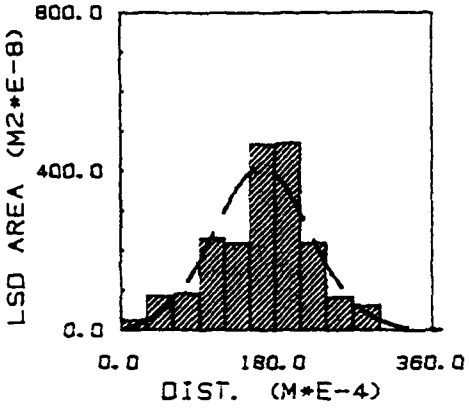
SLIDE 5-6-K



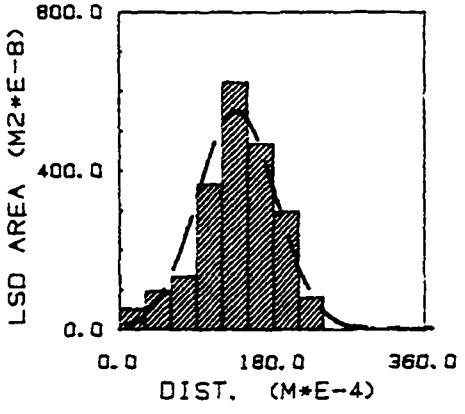
SLIDE 5-6-L



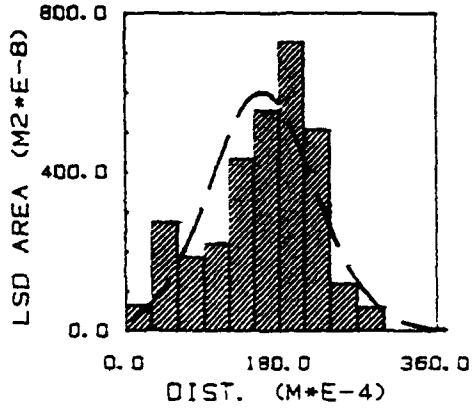
SLIDE 5-6-M



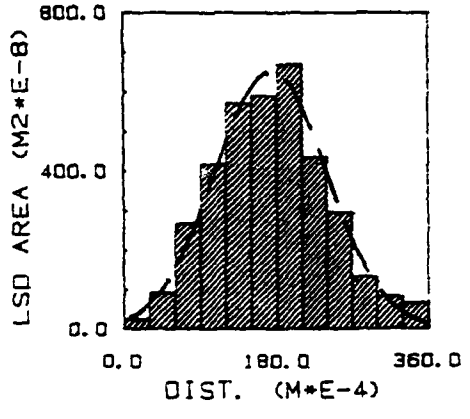
SLIDE 5-6-N



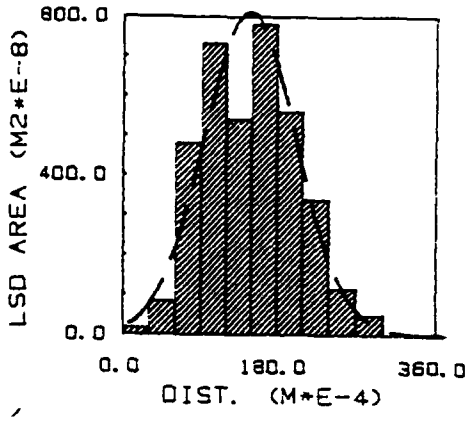
SLIDE 5-6-O



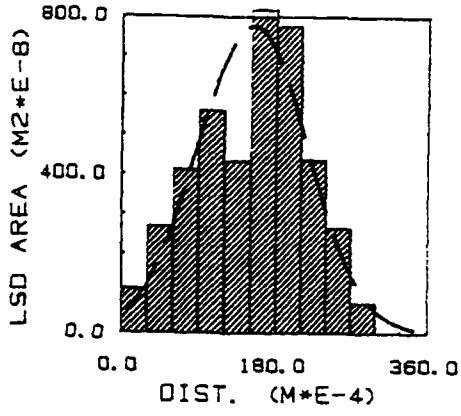
SLIDE 5-6-P



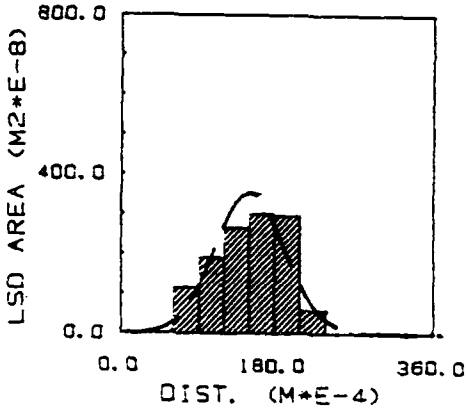
SLIDE 5-6-Q



SLIDE 5-6-R



SLIDE 5-6-S

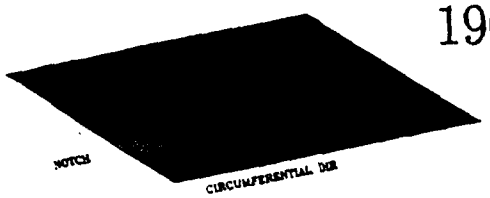


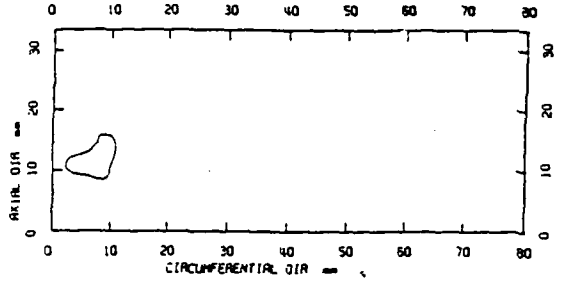
Appendix D

LSD Evolution plot for Cylinder 5–6

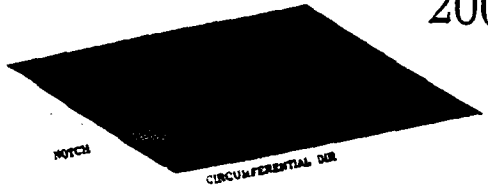
Cycle #

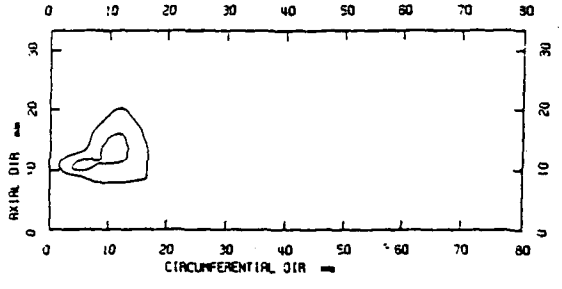
190



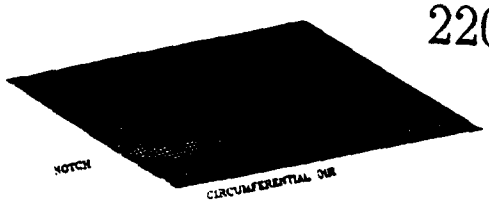


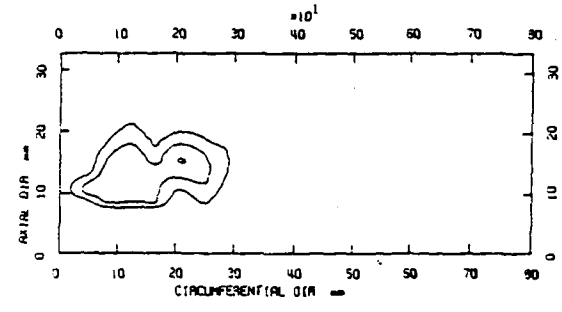
200





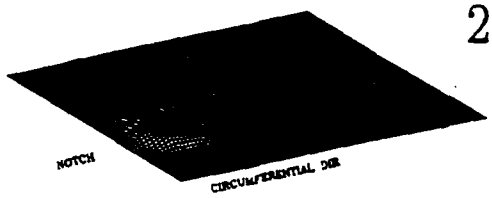
220

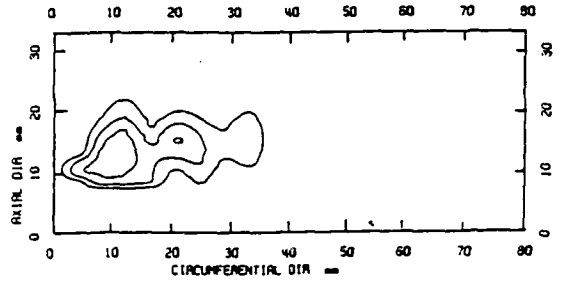




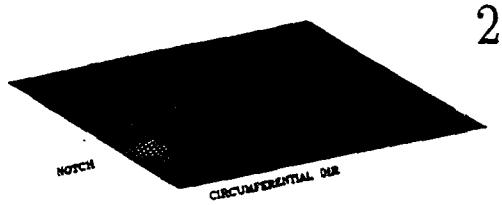
Cycle #

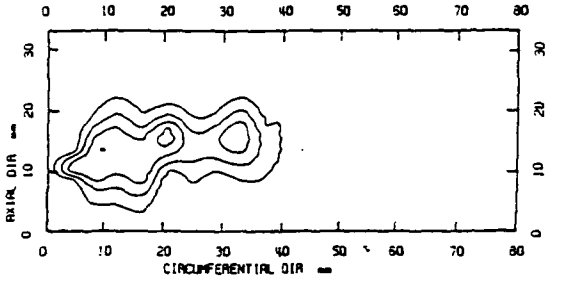
230





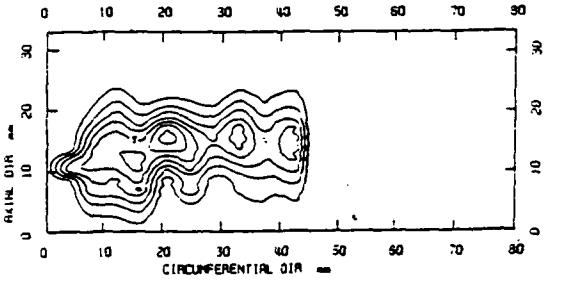
235

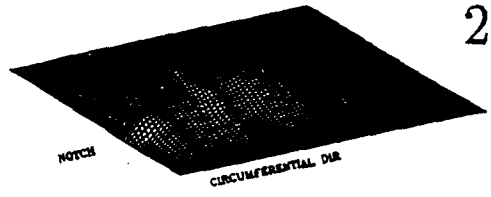




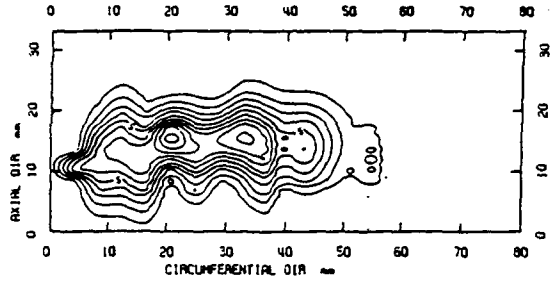
240

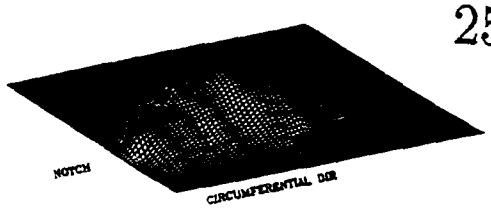




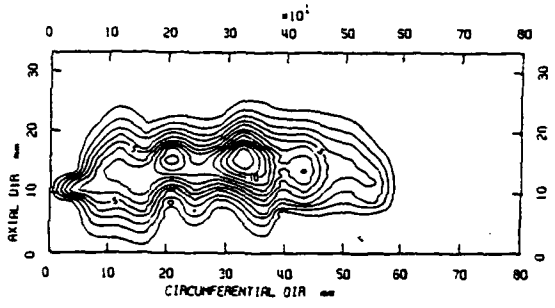

NOTCH
CIRCUMFERENTIAL DIR

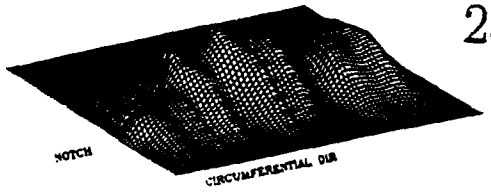
245


AXIAL DIA
CIRCUMFERENTIAL DIA

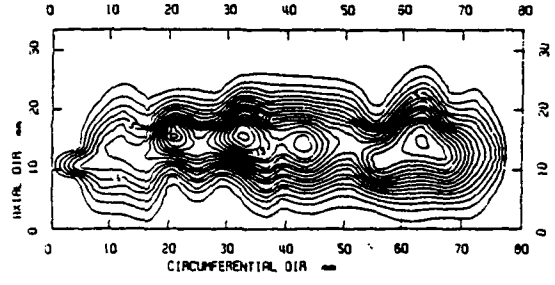

NOTCH
CIRCUMFERENTIAL DIR

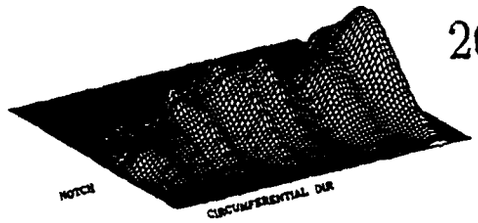
250


AXIAL DIA
CIRCUMFERENTIAL DIA

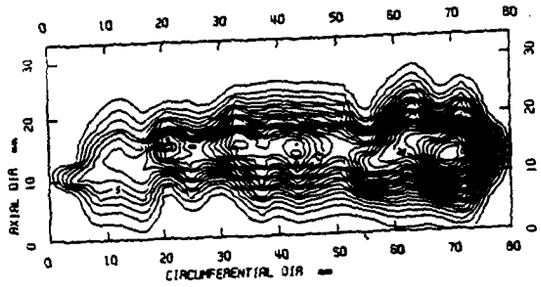

NOTCH
CIRCUMFERENTIAL DIR

256


AXIAL DIA
CIRCUMFERENTIAL DIA


NOTCH
CIRCUMFERENTIAL DIR

262


0
10
20
30
40
50
60
70
80
AXIAL DIR mm
CIRCUMFERENTIAL DIR mm