CR-86.004
December 1985

NCEL
Contract Report

An Investigation Conducted By
University Of California Davis, California
Sponsored By Naval Facilities
Engineering Command

IMPROVED NUMERICAL IMPLEMENTATION OF THE BOUNDING SURFACE PLASTICITY MODEL FOR COHESIVE SOILS

ABSTRACT A substantially more robust numerical algorithm for the evaluation of the bounding surface plasticity model for cohesive soils is developed. The robustness of the new algorithm assures accurate results for reasonable sized solution steps and qualitatively correct predictions, even for exceptionally large steps. The improved predictions are illustrated for three examples.

NAVAL CIVIL ENGINEERING LABORATORY PORT HUENEME, CALIFORNIA 93043

METRIC CONVERSION FACTORS

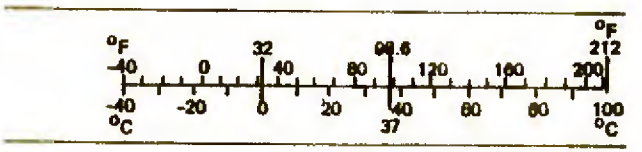
Approximate Conversions to Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	*2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	meters	m
mi	miles	1.6	kilometers	km
		AREA		
in^2	square inches	6.5	square centimeters	cm^2
ft^2	square feet	0.09	square meters	m^2
yd^2	square yards	0.8	square meters	m^2
mi^2	square miles	2.6	square kilometers	km^2
	acres	0.4	hectares	ha
		MASS (weight)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2,000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
Tbsp	tablespoons	15	milliliters	ml
fl oz	fluid ounces	30	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft^3	cubic feet	0.03	cubic meters	m^3
yd^3	cubic yards	0.76	cubic meters	m^3
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

*1 in = 2.54 (exactly). For other exact conversions and more detailed tables, see NBS Misc. Publ. 286, Units of Weights and Measures, Price $2.25, SD Catalog No. C13.10:286.

Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm^2	square centimeters	0.16	square inches	in^2
m^2	square meters	1.2	square yards	yd^2
km^2	square kilometers	0.4	square miles	mi^2
ha	hectares (10,000 m^2)	2.5	acres	
		MASS (weight)		
g	grams	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1,000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m^3	cubic meters	35	cubic feet	ft^3
m^3	cubic meters	1.3	cubic yards	yd^3
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F



Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER CR 86.004	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Improved Numerical Implementation of the Bounding Surface Plasticity Model for Cohesive Soils		5. TYPE OF REPORT & PERIOD COVERED Final Jan 1985 - Sep 1985 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Leonard R. Herrmann, PhD Victor Kaliakin C.K. Shen, PhD		8. CONTRACT OR GRANT NUMBER(s) N62583/85MT176
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of California Davis, CA		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61153N YR023.03.01.006
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Civil Engineering Laboratory Port Hueneme, CA 93043-5003		12. REPORT DATE December 1985 13. NUMBER OF PAGES 64
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Facilities Engineering Command 200 Stovall Street Alexandria, VA 22332-2300		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited.

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

18. SUPPLEMENTARY NOTES

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

finite element, constitutive law, soil plasticity, cohesive soil, numerical implementation

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

A substantially more robust numerical algorithm for the evaluation of the bounding surface plasticity model for cohesive soils is developed. The robustness of the new algorithm assures accurate results for reasonably sized solution steps and qualitatively correct predictions, even for exceptionally large steps. The improved predictions are illustrated for three examples.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

	Page
INTRODUCTION	1
PRELIMINARIES	2
MODIFICATIONS TO "CLAY"	5
IMPLEMENTATION OF CLAY	13
EXAMPLES	24
CONCLUSIONS	25
REFERENCES	31
APPENDIX I	34
APPENDIX II	51
APPENDIX III	55

LIST OF FIGURES

Figure 1.	Schematic illustration of a bounding surface in general stress space	26
Figure 2.	Schematic illustration of a cross-section of the bounding surface for cohesive soils in invariant stress space	27
Figure 3.	Plane strain footing example using SAC2	28

LIST OF TABLES

Table 1.	Triaxial Loading under Drained Conditions	29
Table 2.	Triaxial Loading Under Undrained Conditions	30

INTRODUCTION

The objective of the research reported herein was to improve the "robustness" of the procedure used for the numerical evaluation of the "bounding surface" plasticity model for cohesive soils.

The bounding surface model for soils was developed and extensively evaluated under U.S. Navy and NSF sponsorship (1-9). This development included the numerical evaluation of the model and the writing of a master subroutine (and associated subroutines) CLAY that can be incorporated into existing finite element codes to supply the incremental material properties needed in the analysis of cohesive soil structures (2,5,7). During the original development phase emphasis was placed on demonstrating the capabilities of the model for representing real soil behavior and on the accuracy of the numerical evaluation scheme, with little attention given to the robustness of the numerical evaluation (i.e., the robustness of CLAY). CLAY was incorporated into several new and existing finite element programs and was used in the analysis of a number of Geotechnical engineering problems. During the course of these studies several instances occurred in which the numerical evaluation scheme embedded in CLAY was far from robust.

In all cases the problem involved, for one or more elements, the predicted stress state converging to a point outside of the bounding surface. A simple scaling procedure had been incorporated into CLAY in an attempt to avoid such an occurrance, however, in many cases it proved to be ineffective. Once the stress state had fallen outside of the bounding surface (at the end of a given solution step), in subsequent steps the solution rapidly deteriorated and soon became meaningless and very often convergence could not be achieved (even to an incorrect solution). Using the original version of CLAY the only remedy was to

use smaller global solution increments. In order to achieve an acceptable solution, for many problems, the required solution steps were excessively small and thus the computer costs were excessively large.

The goal of the present research was to understand and to rectify this problem, i.e., to develop a truely robust version of CLAY. Robustness requires that if reasonably sized solution steps are taken, that accurate results be obtained. Of course in general, accurate one step solutions cannot be achieved with a path dependent material properties model. However, the use of very large steps leading to answers which are qualitatively but not quantitatively correct is often useful for a preliminary analysis. Thus, even if ridiculously large solution steps are attempted, a robust routine should produce reasonable (if not entirely accurate) results. Neither of these conditions was met with the original version of CLAY.

PRELIMINARIES

The bounding surface model can be used to supply material properties for all solution schemes applicable to stress analysis problems. The most commonly used method is the finite element procedure; of the three (displacement, force and mixed) finite element formulations available, the displacement method is most commonly used. Thus, for the remainder of this report it is assumed that the bounding surface model is being used in conjunction with a displacement (or a reformulated mixed [10]) finite element analysis. Most of the comments, however, also apply to other methods, e.g., displacement formulation finite difference analyses,etc. In the following paragraphs the finite element code, for which the material properties subroutine (CLAY) is to supply material properties, is called the "parent" program.

The use of a history dependent constitutive model (such as bounding surface plasticity) in a stress analysis program, in general, requires some form of incremental solution procedure. In addition, if the model is nonlinear, then in order to be able to employ reasonably sized solution steps, iteration within each step is usually necessary. The iteration of the total solution by the parent finite element program, will be referred to as "global" iteration.

For a given point in the body under analysis, and for a given iteration K of a given solution step N, the role of a "properties" subroutine, such as CLAY, is to supply to the parent program the relaionship between the stress increment $\{\Delta\sigma\}$ and the strain increment $\{\Delta\varepsilon\}$. The bounding surface model relates "effective" soil stress to strain and, thus, throughout this section $\{\sigma\}$ and $\{\Delta\sigma\}$ will represent effective stresses. The fact that the parent program may be concerned with total stresses will be addressed in a later section. The collection of points (locations) in the body for which the incremental properties are required usually consists of all the element centers or all the element integration points or all the nodes. The following analysis is in no way dependent on which set of points is being used, however, for simplicity of discussion it is assumed that a set of properties is required for each element. The required incremental stress-strain equation is usually written in the form*:

$$\{\Delta\sigma\}_{N,K} = [\bar{D}]_{N,K-1} \{\Delta\varepsilon\}_{N,K} + \{\Delta\sigma_o\}_{N,K-1} \qquad (1)$$

*For a "force" (stress) based analysis, the required relationship is

$$\{\Delta\varepsilon\} = [\bar{C}] \{\Delta\sigma\} + \{\Delta\varepsilon_o\}$$

This expression could be found by inverting eq. 1; however, it would be more economical to re-do the numerical evaluation scheme in CLAY so as to arrive directly at the required expression.

For simplicity, in the remainder of the report, the above expression will be written without explicitly displaying the increment and iteration numbers, i.e.

$$\{\Delta\sigma\} = [\bar{D}]\ \{\Delta\varepsilon\} + \{\Delta\sigma_o\} \tag{2}$$

Thus, the role of the properties subroutine, CLAY, (for a given global iteration of a given solution increment and for a given element) is to provide the quantities $[\bar{D}]$ and $\{\Delta\sigma_o\}$. For a nonlinear model they will be functions of $\{\Delta\sigma\}$ and $\{\Delta\varepsilon\}$, thus the requirement for global iteration. Typically the parent finite element program supplies an estimate (from the previous iteration) of $\{\Delta\varepsilon\}$ and an estimate of $\{\Delta\sigma\}$ is found using eq. 2 and the properties from the previous iteration (or previous increment for the first iteration); these estimates are used by the properties subroutine in the calculations of $[\bar{D}]$ and $\{\Delta\sigma_o\}$. Now until global convergence occurs, these estimates (for the given element) of the stress and strain increments (used in the calculation of the incremental properties) will not in general satisfy the resulting incremental stress-strain equation, (2). Since this inconsistency disappears as global convergence takes place, it is not absolutely necessary to take special steps to rectify it. However, numerical experimentation has demonstrated that overall computational advantages may be realized by resolving it (5,7). For this purpose, "local iteration" can be introduced. Local iteration takes place (for each element) within the material property subroutine (i.e., within CLAY) and involves using eq. 2 and the

original strain estimate to calculate a new estimate of the stress* which in turn is used to find a new set of incremental properties, etc. The local iteration is in addition to the global iteration. Since the global iteration also tends to remove the inconsistency, the convergence criterion for local iteration is typically less restrictive than the global requirement. Obviously the introduction of local iteration increases the computational effort for a given global iteration. However, its use typically decreases the number of global iterations required, resulting in an overall reduction of computational effort.

The calculations involved in finding $[\bar{D}]$ and $\{\Delta\sigma_o\}$ will be discussed in the next section.

MODIFICATIONS TO "CLAY"

The first step was to determine the cause of the lack of robustness of the numerical evaluation scheme used in CLAY. It was found that there were two primary reasons. The first was numerical integration error that occurred for large (and sometimes small) steps. The second was the inadequacy of the scaling procedure used to return a predicted stress state to the bounding surface when it fell outside. When the solution steps were small enough (sometimes exceedingly so) both of these problems disappeared and the original algorithm functioned properly. The question then was how to improve the evaluation scheme.

*An alternative which has not been explored by the author for the bounding surface model is to maintain the initial stress estimate and iteratively modify the strain estimate. The iterative modification of the strain estimate instead of the stress estimate maintains, at the global level, a compatible global displacement field. The rationale for this choice is the displacement continuity requirement of "displacement formulation" finite element procedures. The violation of this condition does not, however, necessarily destroy the convergence characteristics of a displacement formulation; the effect it would have in this case has not been investigated. It should be noted that the present local iteration scheme (which iteratively modifies the stress estimate) would violate the equilibrium requirement of a "force" formulation finite element analysis.

The bounding surface plasticity model leads to the following relationship between the stress and strain rates (see eq. 4 of reference 7):

$$\{\dot{\sigma}\} = [D]\{\dot{\varepsilon}\} \tag{3}$$

For a precise definition of the quantity $[D]$, for the bounding surface model, the reader is referred to eq. 5 of reference 7. Because $[D]$ is a function of the history of the stress state over the interval, for any given solution step this equation is nonlinear (and thus its use, in general, will require iteration). One way of proceeding is to integrate over the step $t_{N-1} \rightarrow t_N$ (in performing this operation it is convenient to think in terms of time even though the particular model may be rate independent), i.e.,

$$\int_{t_{N-1}}^{t_N} \{\dot{\sigma}\}\, dt = \int_{t_{N-1}}^{t_N} [D]\{\dot{\varepsilon}\}\, dt \tag{4}$$

If it is assumed, for the given solution step, that all the strain components are proportional (i.e, $\Delta\varepsilon_{ij}/\Delta\varepsilon_{11}$ = constant for $t_{N-1} \leq t \leq t_N$), then for a rate independent model the input history for the interval can be selected such that all components of $\{\dot{\varepsilon}\}$ are constant* and are given by $\{\dot{\varepsilon}\}=\{\Delta\varepsilon\}/\Delta t_N$, hence

$$\{\Delta\sigma\} = \frac{\{\Delta\varepsilon\}}{\Delta t_N} \int_{t_{N-1}}^{t_N} [D]\, dt \tag{5}$$

*If there are pore water pressure changes due to water flow, then even for a rate independent model and proportionate strain increments, this step will involve an approximation. This approximation occurs because the assumption of a constant strain rate over the interval would not necessarily be consistant with the actual history of the water movement.

letting

$$[\bar{D}] = \frac{1}{\Delta t_N} \int_{t_{N-1}}^{t_N} [D]\, dt \tag{6}$$

then gives

$$\{\Delta\sigma\} = [\bar{D}]\{\Delta\varepsilon\} \tag{7}$$

To this point no approximations other than those mentioned above have been made. Thus what is needed is an accurate evaluation of the average value of $[D]$ over the solution increment. Previously, [2,5,7] the simple one step trapezoidal rule was used, i.e.,

$$[\bar{D}] \simeq \frac{1}{2\Delta t_N} \{[D]_{N-1} + [D]_N\} \tag{8}$$

where $[D]_{N-1}$ and $[D]_N$ are the values of $[D]$ that correspond to the stress and strain states at the beginning and end of the step, respectively. It must be noted that in reality $[D]_{N-1}$ is also a function of $\{\sigma\}_N$ (and hence $\{\Delta\sigma\}$) because the latter quantity determines whether or not "loading" or "unloading" occurs over the interval.

The previously cited lack of robustness has made it evident that in some instances this simple numerical integration is not adequate.

A promising "adaptive" two point integration method was tried but failed to converge. Resort was then made to the multi-step (of equal length) trapezoidal rule*.

*While the multi-step trapezoidal rule has proven to be quite effective it is not the most efficient scheme available. Thus, in some future research the use of other integration schemes should be explored. It would seem to be desirable that the end points of the interval be included in the set of integration points (because the corresponding stress states explicitly occur in the incremental formulation). Thus Simpson's rule and the Gaussian quadrature method that includes the end points (Lobatto quadrature) would appear to be the most promising candidates.

The limit to the accuracy achieved with any improved integration scheme is the required assumption (see derivation above) that the strain components vary proportionately and their rates are constant over the solution step.

In the following discussion the solution step prescribed by the parent program (the finite element program that is calling CLAY) will be referred to as the "step". If required, for accurate numerical integration, the algorithm in the new CLAY may further subdivide this step into "sub-steps". These sub-steps, however, will be transparent to the parent program and to the user (except for increased computer cost).

The use of M sub-steps in the trapezoidal rule leads to the following incremental properties

$$[\bar{D}] = \sum_{m=1}^{M} [\bar{D}]_m \tag{9}$$

The incremental properties $[\bar{D}]_m$, over sub-step m, are found from eq. 8 where $[D]_{N-1}$ and $[D]_N$, which are the values of $[D]$ at times t_{N-1} and t_N, are replaced by properties corresponding to times; $(t_{N-1}, t_{N-1} + \Delta t_N/M)$, $(t_{N-1} + \Delta t_N/M, t_{N-1} + 2\Delta t_N/M)$, etc. and Δt_N is replaced by $\Delta t_N/M$. The strain at time $t_{N-1} + m\,\Delta t_N/M$ is assumed to be (see previous paragraphs discussing the assumptions made)

$$\{\varepsilon\}_m \simeq \{\varepsilon\}_{N-1} + \frac{m}{M}\{\Delta\varepsilon_N\} \tag{10}$$

The stress estimate at the corresponding time is initially (in the first local iteration) taken to be:

$$\{\sigma\}_m \simeq \{\sigma\}_{m-1} + \{\Delta\sigma\}_{m-1} = \{\sigma\}_{N-1} + \sum_{i=1}^{m-1} \{\Delta\sigma\}_i + \{\Delta\sigma\}_{m-1} \tag{11}$$

That is, an estimate equal to the value found in the previous sub-step is used for $\{\Delta\sigma\}_m$. When local iteration is used, then this is successively modified by replacing $\{\Delta\sigma\}_{m-1}$ with improved estimates calculated from eq. 7 (using $\{\Delta\varepsilon\}_m$ and $[\bar{D}]_m$ from the previous local iteration).

The determination of the number of sub-steps (for a given element) to be used in each solution step is based upon the following consideration. It was noted previously that a symptom of inaccurate numerical integration is the prediction of a stress state outside of the bounding surface. The factor that measures this phenomenon is β which relates the image stress state $\bar{\sigma}_{ij}$ to the actual stress state σ_{ij} (see Figure 1), i.e.,

$$(\{\bar{\sigma}\} - \{\sigma_c\}) = \beta(\{\sigma\} - \{\sigma_c\}) \tag{12}$$

The image stress state is defined by the intersection of the projection line with the bounding surface. The projection line originates at the projection center (see Figure 1) and passes through the stress state. For the current model the projection center lies on the I axis and is located at cI_o (see Figure 2). The size of the bounding surface is determine by the current value of I_o (see [7]). It is seen from Figure 1 and eq. 12 that a value of β of less than 1.0 indicates a stress state $\{\sigma\}$ which lies outside of the bounding surface.

In a given global iteration, an initial attempt is made to use one step integration. If the value of β, for the calculated stress state at the end of the solution step is greater or only slightly less than 1.0 ($\geq 0.999^*$), then it is assumed that no problem exists and one step tra-

*A number of rather arbitrary limits such as this have been used in the new version of CLAY. Some numerical experimentation was done to show that the values are adequate, however, future work should investigate a range of alternatives in an attempt to optimize the quantities.

pezoidal integration is used as in the past (2,5,7). When this criterion is not satisfied the new CLAY attempts to use two sub-steps in the integration process. If at the end of either the first or second sub-step the β does not satisfy the $\geq$ 0.999 criterion, the sub-step is again halved (i.e. four sub-steps) and the integration is started again (using one quarter of the initial stress estimate for the beginning of the global iteration). This process is continued until either the criterion is met or a limit of a maximum of 32 sub-steps is reached. In the latter case the solution process is then continued as if the limit on β had been satisfied. Denote the number of sub-steps (for the given element) arrived at by this process as M (M = 1 or 2 or 4 ....32).

Once, sub-stepping is used, it is used for all subsequent global iterations for that solution step and for the particular element in question. As global iteration proceeds, the value of M (number of sub-steps) is not permitted to decrease but it may be increased.* However, in the first global iteration of the next solution step (for the given element) once again one step integration is attemped (i.e., M is reset to 1). The number of sub-steps used (M), is remembered by CLAY (the mechanism will be explained later).

Within each sub-step local iteration is used** to determine the value of $\{\Delta\sigma\}_m$. The local iteration is continued until the value of β for the end of the sub-step meets a 1% convergence criterion or a limit

*Initially a rather sophisticated scheme was tried in which the several sub-steps into which the step was divied could be of varying length and the number of sub-steps could vary from one to a set maximum as needed. At the beginning of each global iteration the number of sub-steps was started at one, thus the sub-stepping pattern could (and did) change from global iteration to global iteration; the resulting process had poor convergence characteristics (due to the changing sub-stepping pattern) and was thus abandoned.

** Thus the control over local iteration is no longer left to the User as in the previous version of CLAY.

of a maximum of 5 local iterations is reached. At least two local interactions are always performed (the first calculation of β is counted as iteration number one). Because of the local iteration, the initial estimate for $\{\Delta\sigma\}$ is not overly important and a value of zero could be used if desired.

The need for an adequate procedure for returning a predicted stress state that is outside of the bounding surface to the surface still exists, even though a more accurate integration scheme has been adopted. Such a need persists because of the use of the 0.999 limit on β instead of 1.0, the maximum of 32 placed on the number of sub-steps for a given solution step, and for the intermediate calculations in the local iteration process. The limits on β and M are used to avoid excessive sub-stepping and thus excessive computational cost.

Classical "radial return" (11) has been adopted to bring a point back to the bounding surface (while the previously used scheme in CLAY was based on the radial return concept, it was only approximate). Whenever a stress state (at the beginning of the step, or at the end of the step or one of the sub-steps) is found to be outside of the bound it is scaled back to the bound (along the line connecting it to the projection center). This scaled stress state is used for the purpose of calculating the plastic modulus.

The one instant where the scaled value of the stress is not used is in the updating of the size of the bounding surface as determined by the value of I_o (see eq. (26) of reference 7). The extreme importance of using the unscaled stress for this operation appears to stem from the fact that the size of its bounding surface is really controlled by strain considerations and the strains are not scaled.

The scaling of the invariants of the stress (the invariants are defined in eq. 20 of (2)) yields:

$$I_{scaled} = \beta(I - cI_0) + cI_0$$

$$J_{scaled} = \beta J \qquad (13)$$

$$S_{scaled} = \beta^3 S$$

When the stress state at the beginning of the step $\{\sigma\}_{N-1}$ is outside of the bound, it is of considerably more concern than the corresponding problem for the state at the end of a step or sub-step. The state at the beginning of the step is an accepted solution (to that point in time), whereas at the end of a step it is merely an intermediate prediction in the iteration process. The discrepancy at the beginning of the step represents an error which has crept into the solution, whereas at the end of a step it is only a potential error. In an attempt to prevent error build-up (as incremental stresses are accumulated), if the stress state at the beginning of the step is found to be oustide of the bound, the individual components are then scaled back to the bound, i.e.,:

$$\{\sigma\}_{N-1_{scaled}} = \beta\{\sigma\}_{N-1} + (1 - \beta)\{c\} \qquad (14)$$

and a stress correlation vector $\{\Delta\sigma_0\}$ is calculated

$$\{\Delta\sigma_0\} = -(1-\beta)(\{\sigma\} - \{c\}) \qquad (15)$$

where $\{c\} = \frac{1}{3} cI_0 < 1,1,1,0,0,0 >^T$

This stress correction is incorporated into the global analysis by treating it as a strain independent stress term, i.e., by modifying eq. 7 to read (i.e., the form is that of eq. 2):

$$\{\Delta\sigma\} = [\bar{D}]\ \{\Delta\varepsilon\} + \{\Delta\sigma_0\} \qquad (16)$$

The use of the stress correction vector prevents a gradual "straying" outside of the bounding surface for cases where the state should be moving on the surface. Equation (16) is only used at the global level because in CLAY the correction represented by $\{\Delta\sigma_o\}$ is directly achieved by the scaling of $\{\sigma\}_{N-1}$; i.e., all operations within CLAY use eq. (7).

For the purpose of calculating $[\bar{D}]$, the scaling of the stress states back to the bounding surface is of major importance, however, the incorporation of the stress correction vector $\{\Delta\sigma_o\}$ into the global analysis can be neglected with only relatively minor consequences.

Situations can arise where the one step trapezoidal rule is not adequate, even though it does not result in obviously incorrect values of β. An example of such a problem area is a nearly neutral loading situation where inaccurate integration may predict "loading" when "unloading" should occur or vise-versa. To avoid inaccurate integration, in general, a second sub-stepping criterion is imposed. The ratio $|L_o^n - L_o^m| / L_o^n$ is required to be less than 0.01, where L_o^m and L_o^n are the sums of the absolute values of the calculated stress components at the end of the increment with different numbers of sub-steps (i.e. 1 and 2 or 2 and 4, etc). This criterion will obviously always require at least 2 sub-steps. As noted above, an upper limit of 32 on the number of sub-steps is imposed.

IMPLEMENTATION OF CLAY

Subroutine CLAY, along with its supporting subroutines, (listed in Appendix I) is intended to be incorporated into new and existing finite element (or finite difference) programs in order to supply the incremental material properties for cohesive soils as predicted by the bounding surface plasticity model. First some general comments concerning

its use will be made, followed by specific instructions for its "call", and finally some concluding remarks concerning the parent program and the calculation of pore pressures.

Subroutine CLAY returns to the parent finite element program the matrices $[\bar{D}]$ and $[D_t]$, and the vectors $\{\Delta\sigma\}$ and $\{\Delta\sigma_o\}$. The matrix $[D]$ and the vector $\{\Delta\sigma_o\}$, as described above, relate the incremental stress and strain vectors by means of eq. 2. The matrix $[D_t]$ is the tangent stiffness matrix (12) and is the value of $[D]$, (eq. 5 of reference 7) at the end of the solution step.

If the global nonlinear solution scheme uses successive substitution, only $[\bar{D}]$ (and of course $\{\Delta\sigma_o\}$) is needed (5,12). If the Newton-Raphson method is being used then the Jacobian can be approximated by the expression $(1-a)\ [\bar{D}] + a[D_t]$. When a=0 the procedure reverts to successive substitution and when a=1 it yields the "tangent stiffness" method*. For a quasi-Newton method (13) neither $[\bar{D}]$ nor $[D_t]$ are needed (except for initiation of the approximate Jacobian and for occasional updates, if used) and can be ignored. The incremental stress $\{\Delta\sigma\}$, however, is needed in the update formula and is calculated in CLAY by the previously described integration process and is returned to the parent program.

The CLAY "subroutine package" consists of the subroutine CLAY and eight supporting subroutines. All the subroutines in this package are written in FORTRAN 77, thus taking advantage of the structuring offered by the language. In keeping with modern programming practice, the

* Contrary to what is suggested in (12), studies performed by the authors for one element problems have found the successive substitution method to show somewhat better convergence characteristics than the tangent stiffness method. The successive substitution method can be significantly improved by using an adaptive convergence factor (7).

design of the package is quite modular and facilitates its incorporation into new and/or existing finite element programs for the analysis of earth structures.

To access the CLAY subroutine package a parent finite element program needs to only call subroutine CLAY. This is done for each iteration of each loading increment and for all points in the idealized earth structure where the incremental properties are required (e.g., at element centers or at quadrature points). The call involves the following argument list:

```
CALL CLAY(INC,ITNO,ITYPE,KIND,LARGE,GAMMA,PROP,STOR,SIG,EP,DSIG,DEP,U,
          DLTAU,DBAR,DTAN,DSIGO)
```

The quantities INC, ITNO, ITYPE, KIND and LARGE are integer variables, U, DLTAU, and GAMMA are real variables, and PROP, STOR, SIG, EP, DSIG, DEP, DBAR, DTAN, and DSIGO are real arrays with the dimensions (21), (6), (6), (6), (6), (6), (6,6), (6,6) and (6), respectively.

The arguments in the call statement are now discussed in detail. For clarity, the order of discussion is changed slightly from that in the subroutine call.

INC: Global solution increment number (the first increment must be numbered 1).

ITNO: Global iteration number (the first iteration in each increment must be numbered 1).

ITYPE: A flag indicating which form of the bounding surface is to be used in the analysis (further details regarding the possible forms of the surface are given in Appendix III).

PROP: A one-dimensional array containing the values of the model parameters. At the point in the idealized earth structure for which the incremental material properties are sought, these

parameters characterize the bounding surface plasticity model for the soil. The parent finite element program must read and store the values of the model parameters for each different type of soil in the structure being analyzed, and then, for each call to CLAY, present the appropriate values for the soil at the point in question.

The model parameters are stored in the PROP array in the following order*: λ, κ, M_c, M_e/M_c, ν or G, Γ, p_ℓ, P_{atm}, R_c, A_c, T, R_e/R_c, A_e/A_c, c, s, m, h_c, h_e/h_c, h_2, a, w. That is, PROP(1) = λ, PROP(21) = w, etc. After the model parameters are read into the parent program, but before the first call to CLAY, the values for M_c and p_ℓ must be converted to associated quantities in invariant stress space. This is achieved by multiplying these parameters by $1/\sqrt{27}$ and 3, respectively. It is suggested that subroutines RPROP and TCHECK, listed in Appendix II, be incorporated into the parent finite element program for the purpose of reading, echo printing and scaling the material parameters. The input instructions for RPROP are given in Appendix III along with a brief discussion of the new single surface option for the bounding surface.

STOR: This one-dimensional array is used to store the values of certain quantities (e.g., internal variables) which describe the current state of the soil and parameters related to the evaluation of the model such as the number of sub-steps M. For a given step in the analysis these values are unique to the point in the structure under consideration (e.g., element centers). At the beginning of the analysis the parent program must initialize, for each point in question, the values of STOR(1) and

*A detailed discussion of the model parameters is given in ref. 6,7.

STOR(5). STOR(1) must contain the initial value of the effective preconsolidation pressure ($I_o = 3p_o$), while the initial value of the total void ratio (e_o) must be stored in STOR(5).* These values are not read by the material parameter input subroutine (RPROP) because they will, in general, vary from point to point within the deposit, even for a homogeneous soil. After each call to CLAY, the values in STOR must be stored by the parent program for each point in the earth structure for which the incremental properties are required. Prior to each call to CLAY the appropriate values for the point in question must be retrieved from storage (i.e., from a two-dimensional array or from a disk file) and then presented to the CLAY subroutine through the CALL.

SIG: This one-dimensional array contains the components of the stress** at the beginning of the increment and must be supplied to CLAY by the parent program. It corresponds to the vector $\{\sigma\}_{N-1}$. Compressive normal stresses are considered to be positive.

EP: The components of the strains at the beginning of the increment are stored in this one-dimensional array and must be supplied to CLAY by the parent program. It corresponds to the vector $\{\epsilon\}_{N-1}$. Compressive normal strains are considered to be positive.

DSIG: This one-dimensional array contains an estimate of the stress** increment. It corresponds to the vector $\{\Delta\sigma\}_N$ and is calculated and returned to the parent program by CLAY.

*Note: When the new version of CLAY is introduced into finite element programs SAC2 and SAC3 (9), the storage of the initial void ratio in STOR(7) must be changed to STOR(5) in Subroutines GEOM and STIFNS.

**Whether these are to be "total" or "effective" stresses is discussed later.

DEP: The estimate of the strain increment is contained in this one-dimensional array. It corresponds to the vector $\{\Delta\varepsilon\}_N$ and is supplied by the parent program.

U: This variable represents the pore pressure at the beginning of the increment N-1. It corresponds to the quantity u_{N-1}*.

DLTAU: The estimate of the pore pressure increment is represented by this variable. It corresponds to the quantity Δu_N*.

GAMMA: This variable is the combined bulk modulus (Γ) for the soil particles and pore fluid (14)*.

KIND: The value assigned to this flag is determined by how saturated conditions are modelled in the parent program.

A value of zero is required when the special (mixed) formulation for incompressible and nearly-incompressible solids (10,15) is used. In such instances, the pore pressure is treated as a primary dependent variable at the global level. Thus, the parent program calculates u_{N-1} and Δu_{N-1} at the global level. Whether or not the parent program is required to send them to CLAY (in u and DLATU) depends on whether it sends total or effective stresses in SIG and DSIG (these possibilities are explored in greater detail later). For such a formulation, the value of Γ (GAMMA) will be required by the parent program and is supplied by CLAY.

When a conventional displacement formulation for compressible solids is used for idealized undrained conditions (14), (or for unsaturated conditions where Γ is equal to zero), KIND is set equal to one. In such cases, the only primary global variables are the displacements; the pore pressure is treated as a

*Whether this quantity is supplied by the parent program or by CLAY is discussed in the explanation of the variable KIND.

secondary dependent variable. The CLAY subroutine calculates the values of u_{N-1} (U) and Δu_N (DLTAU), and supplies them to the main program for printing.

DBAR and DTAN: These arrays contain the estimates of the incremental stress-strain properties $[\bar{D}]$ (see eq. 2) and the "tangent stiffness matrix" $[D_t]$ respectively. For KIND = 1 these matrices have been augmented by adding Γ to the matrix elements in the upper left 3 x 3 corner (more details are given later; see also 7,14).

Subroutine CLAY calculates $[\bar{D}]$ and $[D_t]$ and returns them to the parent program. It is desirable, but not absolutely necessary, that the parent program "remembers" the last $[\bar{D}]$ calculated for the point in question and returns it to CLAY in the next call. This is similar to the requirement for STOR. For the first iteration of the first solution step (and at other instances for which the last calculated $[\bar{D}]$ is not available) it should be set to zero.

DSIGO: This array contains the stress correction vector of eq. 2, i.e. $\{\Delta\sigma_o\}$.

LARGE: The value of this flag depends on the definitions used for the stresses and strains in the analysis. If engineering measures of stresses and strains are used, LARGE is set equal to zero. If true stresses and logarithmic (natural) strains are used, LARGE is set equal to one.

Although the CLAY subroutine package determines three-dimensional incremental stress-strain properties, it can also be used to supply properties for two-dimensional analyses. For a three-dimensional analysis the ordering of the components in the stress (and strain) vector is

$$<\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{xz}, \tau_{yz}>^T$$

In order to use CLAY in conjunction with a two-dimensional analysis the strain and stress vectors in the parent program must be expanded into this form. For example, for an axisymmetric analysis the stress and strain components are ordered in the following manner:

$$<\sigma_r,\sigma_\theta,\sigma_z,\tau_{r\theta}, 0,0,>^T \text{ and } <\varepsilon_r,\varepsilon_\theta,\varepsilon_z,\gamma_{r\theta}, 0, 0>^T$$

The indicated zero values must be supplied by the parent program for the vectors $\{\sigma\}_N$, $\{\varepsilon\}_N$, and $\{\Delta\varepsilon\}_N$. The incremental stress-strain properties of interest are located in the upper left 4x4 corners of the 6x6 DBAR and DTAN arrays returned by the CLAY subroutine, etc.

For plane strain conditions the stress and strain vectors are ordered in the following manner:

$$<\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, 0, 0>^T \text{ and } <\varepsilon_x, \varepsilon_y, 0, \gamma_{xy}, 0, 0>^T$$

The indicated zero values must be supplied by the parent program for the appropriate arrays. It is seen that the parent program must calculate the stress normal to the plane of the body (i.e., σ_z).

As described to this point, subroutine CLAY provides an incremental relationship between the strain increment and the effective stress increment. (Of course, for ideal drained conditions total and effective stresses are identical). If CLAY is to be incorporated into a parent finite element program which only requires this relationship, then no further considerations are needed. For such an application the "call parameters" KIND, U and DLTAU are assigned values of 0, 0.0 and 0.0 respectively, and the material property GAMMA is ignored. The arrays SIG and DSIG are then effective stresses.

Saturated soils under ideal undrained conditions (no movement of pore fluid) behave as nearly incompressible solids and are often approximated as perfectly incompressible solids. For a perfectly incompressible solid a conventional displacement analysis can not be used (10). The most popular means for handling this problem are the use of a "reformulated" analysis (i.e. a "mixed" approach) (10) and the use of a penalty approximation of the incompressibility (16). The reformulated analysis only requires a relationship between effective stress and strain and hence the comments of the previous paragraph apply. The penatly method in reality considers the material to be only nearly incompressible and is dealt with in the next paragraph.

It is often assumed that the slight compressibility of the pore fluid and the soil particles can be modeled by the simple linear relation:

$$\Delta V/V_0 = - \Gamma \Delta u$$

Where $\Delta V/V_0$ is the incremental change in volume, Γ is a constant bulk modulus* for the pore fluid-soil particles and Δu is the change in pore pressure. (The approximation of incompressible behavior by means of a penalty formulation introduces this same equation where Γ is interpreted as the penalty number (16)). Typical Γ will be very large compared to the deviatoric stiffness and thus the soil will behave as a "nearly incompressible solid." The analysis of nearly incompressible solids is notoriously difficult (10,17) and requires some special consideration. The most commonly used schemes are the conventional displacement formulation with "selected-reduced" integration (16) and the previously mentioned reformulated (mixed) analysis (10).

*The possibility of treating Γ as a variable in an attempt to model partially saturated conditions is beyond the scope of this study.

When a reformulated analysis is used for a nearly incompressible material the parent program needs, in addition to the effective stress-strain relationship, the volume change-pore pressure relationship. For this latter purpose CLAY supplies the value of Γ (GAMMA of the call) to the main program. In a reformulated analysis the pore water pressure is directly calculated in the global finite element analysis. If it is desired that the parent program send and receive total stress information (SIG and DSIG) to CLAY then it must also supply (in the call) the appropriate pore pressure values (U and DLTAU) to CLAY. If, instead, effective stresses are used then U and DLTAU are supplied as zero.

For a conventional displacement analysis of idealized undrained conditions where the water-soil particle compressibility is included, the material stiffness is augmented by the volume compressibility due to Γ, i.e., Γ is added by CLAY to each member of the upper left 3 x 3 corner of $[\bar{D}]$ and $[D_t]$ (DBAR and DTAN of the call). For such conditions the call parameter KIND is set equal to 1, and SIG and DSIG are total stress vectors. Subroutine CLAY calculates the resulting pore pressure at the beginning of the interval (U) and the incremental change (DLTAU) and returns them to the main program for printing purposes.

Either of the two methods discussed for undrained conditions are valid for ideal drained conditions if Γ is set equal to zero.

The most important class of problems is when the soil is saturated and there is time for flow of the pore fluid to take palce (due to non-homogenous stress conditions). Such situations require the solution of the coupled flow-stress analysis problem (a number of references are given in (8)). Such an analysis requires the incremental strain-effective stress relationship which can be supplied by CLAY. The parameter KIND is set equal to 0. The stress vectors (SIG and DSIG) are

either effective or total depending on whether U and DLTAU are supplied to CLAY as zeros or as the actual pore water pressure values (as calculated by the parent program from the coupled analysis).

The subroutines making up the CLAY subroutine package are briefly described below. Listing of the subroutines are given in Appendix I.

CLAY: This is the main subroutine of the package. The parent finite element program must call this subroutine in the manner discussed in the previous section. The substepping integration strategy is determined and the local iteration is conducted in the subroutine.

BOUND1: In this subroutine the quantities β, $F,_{\bar{I}}$, $F,_{\bar{J}}$, $F,_{\alpha}$ and $\bar{K}_p$ are computed for the form of the bounding surface consisting of a single ellipse (further details regarding this new form of the surface are given in Appendix III and in (18)); i.e., for ITYPE=1 (refer to the discussion of the call to CLAY).

BOUND3: Values for the quantities listed above are computed for the form of the bounding surface consisting of two ellipses and a hyperbola (7); i.e., for ITYPE=3.

ELASTC: The elastic contribution to the $[D]$ array is computed in this subroutine.

GETH: The value of the hardening function is determined in this subroutine. Further details regarding forms of the hardening function are given in Appendix III.

INVAR: The values of the three stress invariants used in the formulation are computed in this subroutine.

LODFUN: The values of the plastic modulus, K_p, and of the loading index, L (eq. (7) in reference (7)), are computed in this subroutine.

PLASTC: The plastic contribution to the [D] array (eq. (5) in reference (7)) is computed.

SKALE: In this subroutine if the stress state lies outside the bounding surface, then using eqs. (13), it is scaled back to it.

EXAMPLES

A number of examples have been analyzed in order to demonstrate the improvements in "robustness" of CLAY. The first series involved incorporating the new CLAY into program EVAL (7) and solving single element examples. For those cases in which CLAY had previously exhibited poor behavior the improvement was dramatic. In Tables 1 and 2 comparisons are made of the results from two of these analyses for varying numbers of global steps to reach a given final loading condition. While the results obtained with the new CLAY for one very large global step are not entirely accurate, they are not unreasonable as was the case with the old version of CLAY. A careful study of the predicted pore water pressures (u) in Table 2 for the new CLAY reveals, an interesting phenomenon, i.e., the convergence with increasing numbers of global increments is not entirely smooth. This behavior can be observed by comparing the results with 2, 4 and 8 global steps. This phenomenon is caused by the adaptive sub-stepping scheme used in the new CLAY which can result in a greater total number of sub-steps even though the number of global steps is less.

The second series involved using the old and new versions of CLAY in the plane strain finite elementr program SAC2 and analyzing two idealized footing problems. Properties for a typical unsaturated clay were used for the soil. Results from one of these examples are given in Figure 3. The improvement in predictions for large step sizes is clearly evident.

CONCLUSIONS

The numerical scheme used to evaluate the bounding surface model for cohesive soils has been successfully modified in order to substantially improve its robustness. The modifications include the introduction of sub-stepping when necessary and a more accurate radial return algorithm.

It was demonstrated by a number of examples that the new algorithm is quite robust. That is for reasonably sized solution steps it gives accurate results and for very large steps it is convergent and gives reasonable accuracy. The one question that remains to be explored is the impact of these modifications on the computational cost. The cost of the material model evaluation (as compared to other costs such as equation solution) for large finite element analyses of earth structures must be determined.

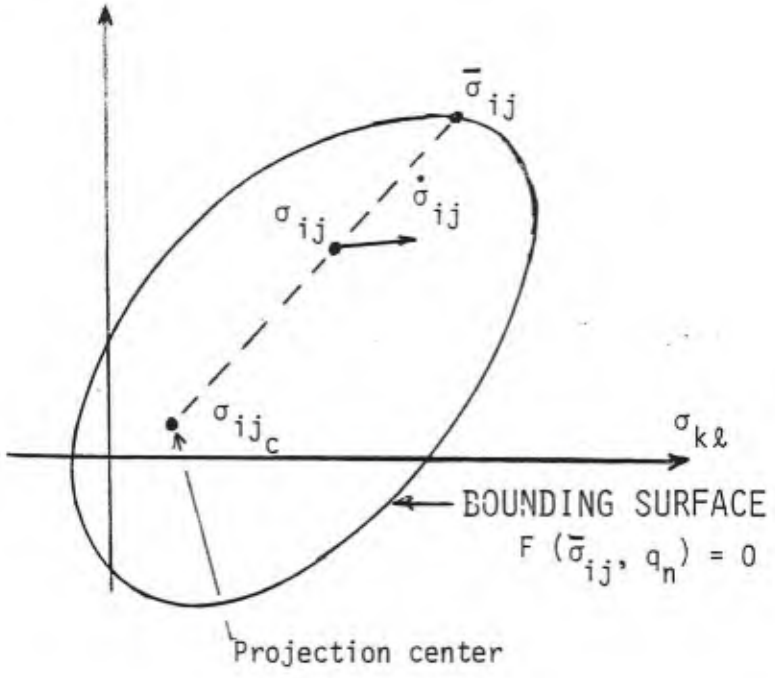


Fig. 1: Schematic illustration of a bounding surface in general stress space

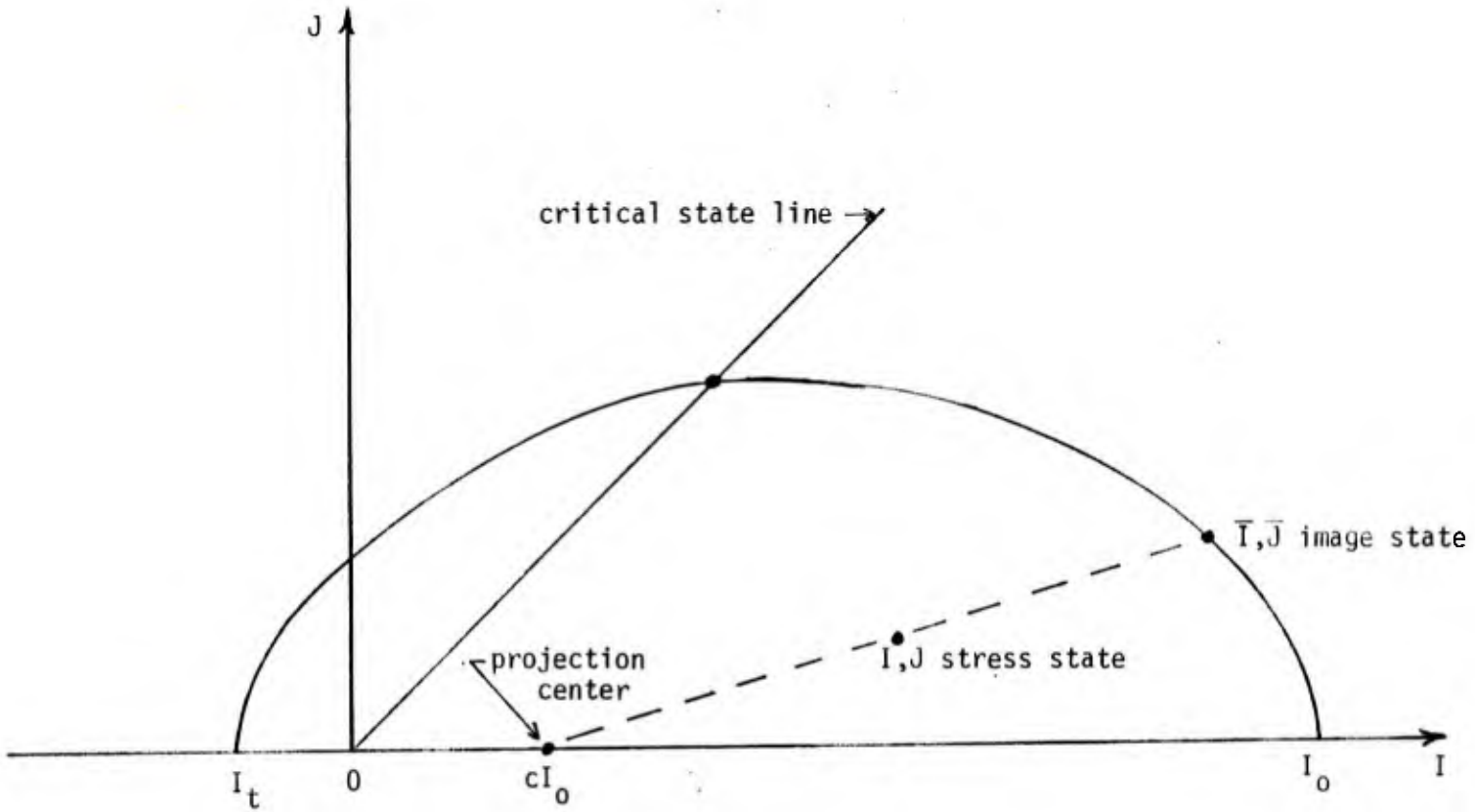


Fig. 2: Schematic illustration of a cross-section of the bounding surface for cohesive soils in invariant stress space

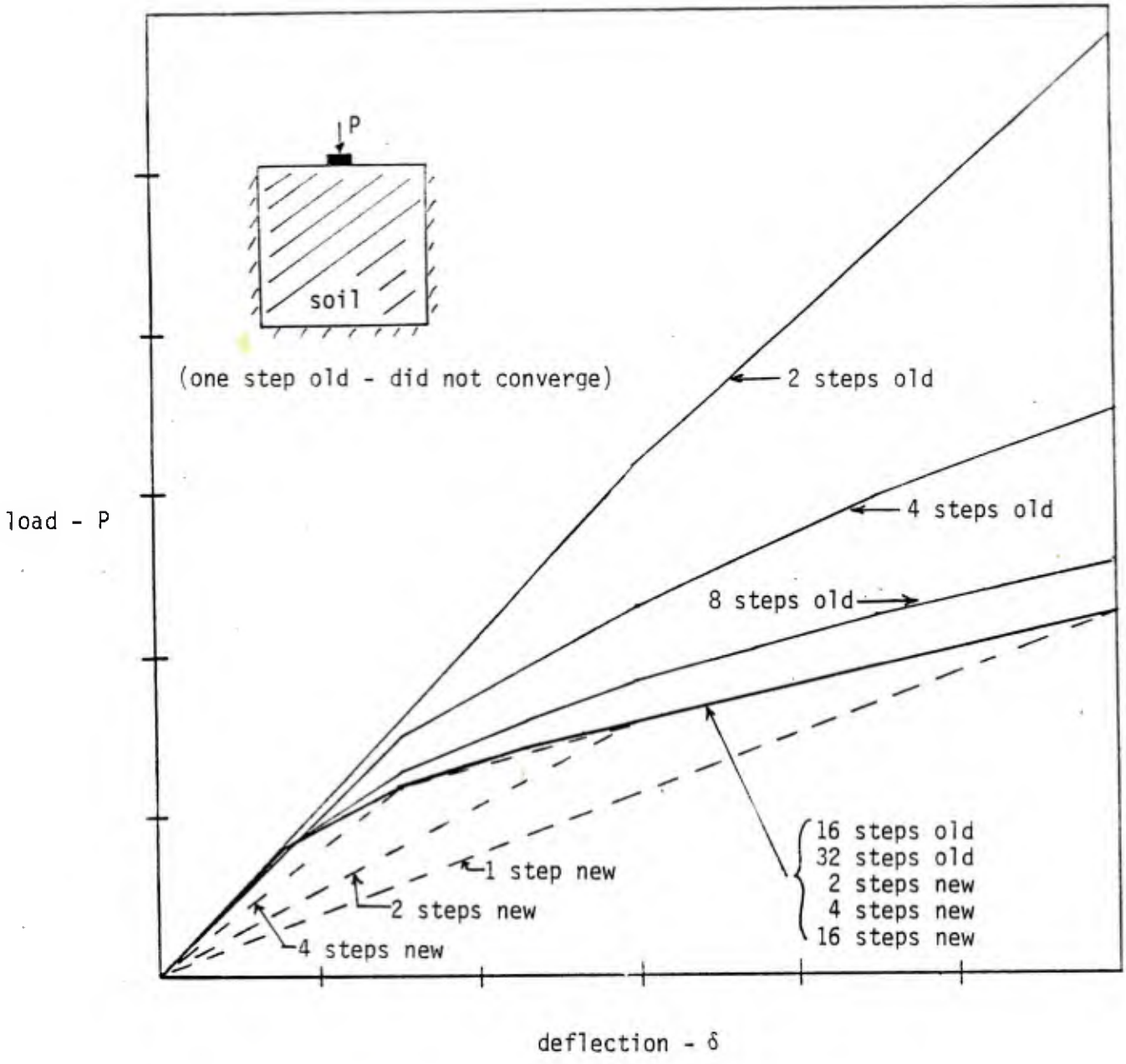
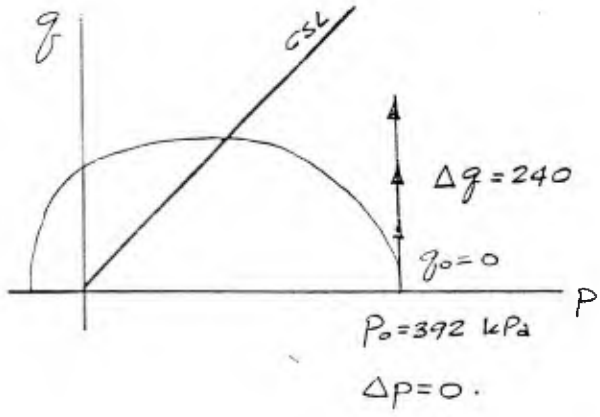


Fig. 3: Plane Strain Footing Example Using SAC2

Table 1. Triaxial Loading under Drained Conditions

total stress path:

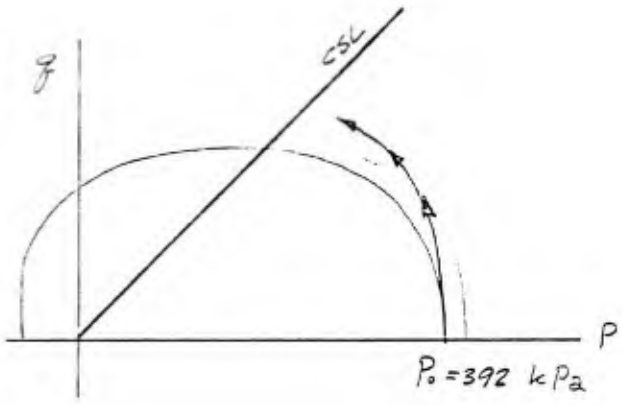


Number of Increments	ε_1 (%)		p (kPa)		ε_v (%)	
	old	new	old	new	old	new
1	1.06	5.18	392	392	0.61	2.64
2	2.34	4.77	-"-	-"-	1.62	2.71
4	3.35	4.38	-"-	-"-	2.27	2.72
8	3.91	4.24	-"-	-"-	2.57	2.72
16	4.11	4.21	-"-	-"-	2.68	2.72
32	4.17	4.19	-"-	-"-	2.71	2.72
64	4.18	4.19	-"-	-"-	2.71	2.72
128	4.19	4.19	-"-	-"-	2.71	2.72

Table 2. Triaxial Loading under Undrained Conditions (reformulated analysis)

undrained stress path:

ε_1 = 10% applied



Number of Increments	q (kPa)		P (kPa)		u (kPa)	
	old	new	old	new	old	new
1	2593.04	249.80	176.06	169.19	1080.28	306.08
2	1459.12	225.01	29.68	180.05	848.69	286.95
4	820.02	225.02	55.80	180.16	609.54	286.84
8	477.68	224.99	96.47	180.12	454.76	286.88
16	312.40	224.97	146.02	180.19	350.11	286.80
32	249.77	224.97	170.66	180.21	304.60	286.78
64	231.07	224.99	179.05	180.23	289.97	286.77
128	226.08	225.00	181.10	180.23	286.26	286.76

REFERENCES

1. Dafalias, Y.F. and L.R. Herrmann, "A Bounding Surface Soil Plasticity Model", Proceedings of the International Symposium of Soils Under Cyclic and Transient Loadings, University of Swansea, United Kingdom, pp. 335-345, January 1980.

2. Herrmann, L.R., Y.F. Dafalias, and J.S. DeNatale, "Bounding Surface Plasticity for Soil Modeling", Final Report to Civil Engineering Laboratory, Naval Construction Battalion Center, Port Hueneme, CA, Order No. USN N62583-80 M R478, October 1980.

3. Herrmann, L.R., C.K. Shen, S. Jafroudi, J.S. DeNatale, and Y.F. Dafalias, "A Verification Study for the Bounding Surface Plasticity Model for Cohesive Soils", Final Report to Civil Engineering Laboratory, Naval Construction Battalion Center, Port Hueneme, California, Order No. USN N62583-81MR320, December 1981.

4. Dafalias, Y.F. and L.R. Herrmann, "Bounding Surface Formulation of Soil Plasticity", Chapter in Soil Mechanics - Transient and Cyclic Loads, John Wiley and Sons, Eds. O.C. Zienkiewicz and G.N. Pande, pp. 253-282, 1982.

5. Herrmann, L.R., Y.F. Dafalias, and J.S. DeNatale, "Numerical Implementation of a Bounding Surface Soil Plasticity Model", Proceedings of the International Symposium on Numerical Models in Geomechanics, Zurich, Switzerland, 1982.

6. DeNatale, J.S., L.R. Herrmann, and Y.F. Dafalias, "Calibration of the Bounding Surface Soil Plasticity Model by Multivariate Optimization", Proceedings of the International Conference on Constitutive Laws for Engineering Materials, Theory and Application, Tucson, Arizona, January 1983.

7. Herrmann, L.R., V.N. Kaliakin, and Y.F. Dafalias, "Computer Implementation of the Bounding Surface Plasticity Model for Cohesive Soils", Report to the Civil Engineering Laboratory, Naval Construction Battalion Center, Port Hueneme, CA, Order No. N62583-83-M-T062, September 1983.

8. Herrmann, L.R. and K.D. Mish, "Finite Element Analysis for Cohesive Soil, Stress and Consolidation Problems Using Bounding Surface Plasticity Theory", Final Report to the Civil Engineering Laboratory, Naval Construction Battalion Center, Port Hueneme, CA, Order No. N62583-83-M-T062, October 1983.

9. Mish, K.D. and L.R. Herrmann, "User's Manual for SAC-3, A Three-Dimensional Nonlinear, Time Dependent Soil Analysis Code Using the Bounding Surface Plasticity Model", Report to Civil Engineering Laboratory, Naval Construction Battalion Center, Port Hueneme, CA, Order No. N62583-83-M-T062, October 1983.

10. Herrmann, L.R., "Elasticity Equations for Incompressible and Nearly Incompressible Materials by a Variational Theory," AIAA J., Vol. 3, No. 10, 1896-1900, 1965.

11. Hughes, J.K., "Numerical Implementation of Constitutive Models: Rate-Independent Devatoric Plasticity", Workshop on the Theoretial Foundation for Large-Scale Computations of Nonlinear Material Behavior", Northwestern University, October 1983.

12. Owen, D.R.J. and E. Hinton, Finite Elements in Plasticity Theory and Practice, Pineridge Press Limited, Swansea, U.K., 1980.

13. Geradin, M., S. Idelsohn, and M. Hogge, "Computational Strategies for the Solution of Large Nonlinear Problems via Quasi-Newton Methods", Computer and Structures, Vol. 13, 1981, pp. 73-81.

14. Sangrey, D.A., D.J. Henkel, and M.I. Espig, "The Effective Stress Response of a Saturated Clay Soil to Repeated Loading", Canadian Geotechnical Journal, 1969, 6(3), 241-252.

15. Taylor, R.L., K.S. Pister, and L.R. Herrmann, "On a Variational Theorem for Incompressible and Nearly Incompressible Orthotropic Elasticity", Int. J. of Solids and Structures, Vol. 4, 875-883, 1968.

16. Zienkiewicz, O.C., The Finite Element Method, McGraw-Hill, London, 1977.

17. Zienkiewicz, O.C., R.L. Taylor and J.M.W. Baynham, "Mixed and Irreducible Formulations in Finite Element Analysis," Chp. 21 in Hybrid and Mixed Finite Element Methods, (eds. S.M. Atluri, R.H. Gallagher and O.Z. Zienkiewicz), John Wiley and Sons, Chichester, 1982.

18. Kaliakin, V.N., "Bounding Surface Elastoplasticity - Viscoplasticity for Clays," dissertation submitted to the University of California, Davis, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

APPENDIX I

```
      SUBROUTINE CLAY (INC,ITNO,ITYPE,KIND,LARGE,GAMMA,PROP,STOR,SIGTM,
     $                 EPM,DSIGM,DEPM,UB,DLTAU,DBAR,DTAN,DSIGO)
*********************************************************************
*         Subroutine to evaluate Yannis Dafalias' bounding          *
*             surface plasticity model for clay soils.              *
*     Fortran 77 version, Prepared by L.R. Herrmann and V.Kaliakin  *
*            at the University of California, Davis Campus.         *
*                                                                   *
*   "Robust" version using radial return and sub-incrementing - 11/85  *
*********************************************************************
      INTEGER I,J,K,IT,INC,ITNO,ITYPE,KIND,LARGE,LOIT,LOITMX,II(6)
      REAL PROP(1),STOR(6),SIGTM(6),EPM(6),DSIGM(6),DEPM(6),DBAR(6,6),
     $     DTAN(6,6),SIGB(6),EPB(6),DSIG(6),DEP(6),DEPT(3,3),SB(3,3),
     $     SE(3,3),DLTA(3,3),DB(6,6),DSIGO(6),SIGEM(6),SIGBP(6),
     $     UB,DLTAU,GAMMA,SMALL,DIL,DDIL,VOIDB,VOIDE,XIB,XIE,XJB,XJE,
     $     SCUBEB,SCUBEE,SIN3AB,SIN3AE,COS3AB,COS3AE,BULKB,BULKE,GB,GE,
     $     XIOB,XIOE,XIL,XIBSV,XIESV,BETA,ER,REF,CONV,PRT,RTS,
     $     TEMP1,TEMP2,TEMP3,TEMP4
C
      DATA II/11,22,33,12,13,23/, DLTA/1.0,3*0.0,1.0,3*0.0,1.0/
C -----------------------------------------------------------------
C            A number of arbitrary limits are defined
C
      SMALL=0.0001*PROP(8) ! small value for stress invariant
      BETALM=0.999         ! limit on "beta" (how far can be out of bound)
      STPMIN=1./32. + .01  ! limit of 32 sub-steps
      LOITMX=5               ! limit on number of local iterations
      CONV=0.01              ! convergence limit for local iterations
C                                  and sub-stepping
C -----------------------------------------------------------------
C             Initialize history for first increment
C
      IF(INC .EQ. 1 .AND. ITNO .EQ. 1)  THEN
          STOR(2)=STOR(1)  !     size of bounding surface Io
          STOR(3)=UB       ! pore water pressure(for non-reformulated)
          STOR(6)=1.0      !     size of sub-steps
      END IF
C
C        Update history for new increment
C
      IF(INC .GT. 1 .AND. ITNO .EQ. 1)  THEN
          STOR(1)=STOR(2)   !     size of bounding surface Io
          STOR(3)=STOR(4)   ! pore water pressure(for non-reformulated)
          STOR(6)=1.0       !     size of sub-steps
      END IF
C
C     Convert from total stress formulation to effective stress
C
      GAMMA=PROP(6)            ! bulk modulus for water & soil particles
```

```
      IF(KIND .EQ. 1)  THEN     ! unreformulated analysis
          UB=STOR(3)
          DLTAU=GAMMA*(DEPM(1)+DEPM(2)+DEPM(3))  !change in pore pressure
          STOR(4)=UB + DLTAU  ! store pore press. (unreformulated analys)
C
          DO 5 I=1,3          ! remove stiffness due to gamma
              DO 4 J=1,3
                  DBAR(I,J)=DBAR(I,J) - GAMMA
    4         CONTINUE
    5     CONTINUE
      ENDIF
      DO 10 I=1,3                          ! convert to effective stresses
          SIGEM(I)=SIGTM(I) - UB
          SIGBP(I)=SIGEM(I)           ! initial guess of stress for end
          SIGEM(I+3)=SIGTM(I+3)       !       of increment
          SIGBP(I+3)=SIGEM(I+3)
   10 CONTINUE
C
C     Calculate the initial estimate for the incremental stress
C
      DO 30 I=1,6
          TEMP=0.0
          DO 20 J=1,6
             TEMP=TEMP + DBAR(I,J)*DEPM(J)
   20     CONTINUE
          DSIGM(I)=TEMP
   30 CONTINUE
C
      RTS=STOR(6)  ! recall sub-step size used in previous iteration
C
   40 CONTINUE      ! return point when change sub-step size
C
C     Transfer stress ,strain and pore water pressure to local arrays so
C     as not to disturb the values brought in from the parent program
C
      DO 100 I=1,6
          SIGB(I)=SIGEM(I)
          DSIG(I)=DSIGM(I)*RTS
          EPB(I)=EPM(I)
          DEP(I)=DEPM(I)*RTS
          DSIG0(I)=0.0
          DO 50 J=1,6
             DBAR(I,J)=0.0
             DTAN(I,J)=0.0
   50     CONTINUE
  100 CONTINUE
C
C       Determine 3-dimensional incremental properties.
C
      FACTOR=0.0           ! Tangent properties at start of increment
      XIOB=STOR(1)
      CALL INVAR (FACTOR,STOR,SIGB,DSIG,EPB,DEP,DEPT,VOIDB,LARGE,
```

```
     $           SMALL,XIB,XJB,DIL,DDIL,SB,SCUBEB,SIN3AB,COS3AB,
     $           DLTA,II)
      XIBSV=XIB    !save unscaled value for finding change in Io
      CALL ELASTC (PROP,VOIDB,XIB,DB,BULKB,GB)
      CALL PLASTC (ITYPE,PROP,DEPT,VOIDB,XIB,XJB,XIOB,DDIL,SB,SCUBEB,
     $            SIN3AB,COS3AB,DB,BULKB,GB,DLTA,II,INC,ITNO,BETA)
C
C     If stress at start of increment outside of bound move it back
C                and calculate stress correction vector
C
      IF(BETA .LT. 1.0)  THEN
          DO 200 I=1,3
              DSIG0(I)=-(1.0 - BETA)*(SIGB(I) - XIB*PROP(14)/3.0)
              SIGB(I) =SIGB(I) + DSIG0(I)
              DSIG0(I+3)=-(1.0 - BETA)*SIGB(I+3)
              SIGB(I+3) =SIGB(I+3) + DSIG0(I+3)
  200     CONTINUE
      END IF
C
C     Sub-incrementing (if needed) integration loop over solution step
C
      PRT=0.0
C
  300 CONTINUE                                    ! Sub-incrementing loop
C
      BETAL=0.0                ! initialize memory for beta
      DO 325 LOIT=1,LOITMX     ! Local iteration on stress sub-increment
C
         FACTOR=1.0          ! Tangent properties at end of sub-increment
         CALL INVAR (FACTOR,STOR,SIGB,DSIG,EPB,DEP,DEPT,VOIDE,LARGE,
     $             SMALL,XIE,XJE,DIL,DDIL,SE,SCUBEE,SIN3AE,COS3AE,
     $             DLTA,II)
         XIESV=XIE        ! save unscaled value for finding change in Io
C
         CALL ELASTC (PROP,VOIDE,XIE,DTAN,BULKE,GE)
C
         XIL=PROP(7)              ! calculate size of bounding surface
         TEMP1=1.0/(PROP(1) - PROP(2))
         TEMP2=(XIESV - XIBSV)/3.0
         IF(XIOB .GE. XIL)  THEN
            XIOE=XIOB*EXP(TEMP1*0.5*((VOIDB + VOIDE)*DDIL
     $           - (VOIDB/BULKB + VOIDE/BULKE)*TEMP2))
         ELSE
            XIOE=XIOB + TEMP1*XIL*0.5*((VOIDE+VOIDB)*DDIL
     $              - (VOIDB/BULKB + VOIDE/BULKE)*TEMP2)
         END IF
         CALL PLASTC (ITYPE,PROP,DEPT,VOIDE,XIE,XJE,XIOE,DDIL,SE,SCUBEE,
     $               SIN3AE,COS3AE,DTAN,BULKE,GE,DLTA,II,INC,ITNO,BETA)
C
         DO 320 I=1,6      ! estimate of stress change over sub-increment
             TEMP=0.0
             DO 310 J=1,6
```

```
                TEMP=TEMP + 0.5*(DB(I,J) + DTAN(I,J))*DEP(J)
  310         CONTINUE
              DSIG(I)=TEMP
  320     CONTINUE                                    ! check convergence
          IF(ABS(BETA-BETAL)/MAX(BETAL,BETA) .LT. CONV)  GOTO 326
          BETAL=BETA                                  ! update memory of beta
  325 CONTINUE
  326 CONTINUE
C
C      Check to see if sub-incrementing is sufficiently fine
C
      IF(BETA .LT. BETALM  .AND.  RTS .GT. STPMIN) THEN
           RTS=RTS*0.5                          ! halve the interval
C
           GO TO 40       ! start over on the step with smaller sub-steps
C
      END IF
      PRT=PRT + RTS       ! update at end of sub-step
      BULKB=BULKE
      XIBSV=XIESV
      XIB=XIE
      VOIDB=VOIDE
      XIOB=XIOE
C
      DO 400 I=1,6
          SIGB(I)=SIGB(I) + DSIG(I)
          EPB(I) =EPB(I) + DEP(I)
          DO 390 J=1,6
              DBAR(I,J)=DBAR(I,J) + 0.5*(DB(I,J) + DTAN(I,J))*RTS
              DB(I,J)=DTAN(I,J)
  390     CONTINUE
  400 CONTINUE
C
      IF(PRT .LT. 0.99)  GOTO 300    ! solution increment not yet complete
C
C       Check to see if sub-stepping is sufficiently fine to give
C                    accurate stress predictions
C
      ER=0.0
      REF=0.0
      DO 450 I=1,6
          ER=ER + ABS(SIGB(I) - SIGBP(I))
          REF=REF + ABS(SIGB(I))
          SIGBP(I)=SIGB(I)               ! remember stress with previous
  450 CONTINUE                           !          step size
      IF(ER/REF .GT. CONV .AND. RTS .GT. STPMIN)  THEN
          RTS=RTS*0.5
          GOTO 40              ! start over on the step, only with
      END IF                   !     smaller sub-steps
      IF(BETA .LT. 1.0)WRITE(4,*)'ITNO=',ITNO,'BETA=',BETA
C
      STOR(2)=XIOE                     ! store size of bounding surface
```

```
      STOR(6)=RTS                          ! store sub-step size
C
C     Calculate incremental stress and return it to the main program
C
      DO 500 I=1,6                                ! stress increment
          DSIGM(I)=SIGB(I) - SIGEM(I)
  500 CONTINUE
C
C         Conversion from effective stress to total stress
C
      IF(KIND .EQ. 1)  THEN                ! for non-reformulated analysis
          DO 560 I=1,3                     ! add in stiffness due to gamma
              DO 555 J=1,3
                  DBAR(I,J)=DBAR(I,J) + GAMMA
                  DTAN(I,J)=DTAN(I,J) + GAMMA
  555         CONTINUE
  560     CONTINUE
      END IF
      DO 600 I=1,3                 ! add pore pressure to effective stress
          DSIGM(I)=DSIGM(I)+DLTAU
  600 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INVAR (FACTOR,STOR,SIG,DSIG,EPB,DEP,DEPT,VOID,LARGE,
     $                  SMALL,XI,XJ,DIL,DDIL,S,SCUBE,SIN3A,COS3A,
     $                  DLTA,II)
C
C            Subroutine to compute values of invariants
C
      INTEGER I,J,K,N,IK,LARGE,II(1)
      REAL STOR(6),SIG(6),DSIG(6),EPB(6),DEP(6),DEPT(3,3),
     $     S(3,3),DLTA(3,3),VOID,SMALL,XI,XJ,DIL,DDIL,SCUBE,SIN3A,
     $     COS3A,ARB,FACTOR,TEMP1
      PARAMETER (yes=1)
      DATA ARB/1000.0/
C
      DIL=0.0
      DDIL=0.0
      DO 20 I=1,3
          DIL =DIL  +  EPB(I)
          DDIL=DDIL +  DEP(I)
   20 CONTINUE
      XI=0.0
      XJ=0.0
      SCUBE=0.0
      SIN3A=0.0
C
C        Calculate first stress invariant
C
      DO 100 I=1,3
          XI=XI + SIG(I) + FACTOR*DSIG(I)
  100 CONTINUE
C
      VOID=1.0 + STOR(5)
      IF(LARGE .EQ. yes)
     $        VOID=VOID*EXP(-DIL - FACTOR*DDIL)
C
C      Change matrix components to tensor components;
C             calculate deviatoric stresses.
C
      DO 200 N=1,6
         I=II(N)/10
         J=MOD(II(N),10)
         S(I,J)=SIG(N) + FACTOR*DSIG(N) - XI*DLTA(I,J)/3.0
         S(J,I)=S(I,J)
         DEPT(I,J)=DEP(N)*(1.0 + DLTA(I,J))*0.5
         DEPT(J,I)=DEPT(I,J)
  200 CONTINUE
C
C      Avoid near zero value of the first stress invariant
C
      IF(ABS(XI) .LE. SMALL) THEN
          TEMP1=SIGN(1.0,XI)
          XI=SMALL*TEMP1
      END IF
```

```
C
C    Compute the square root of the second deviatoric stress invariant
C            as well as the third deviatoric stress invariant
C
      DO 300 I=1,3
          DO 300 J=1,3
              XJ=XJ + S(I,J)*S(I,J)
              DO 300 K=1,3
                  SCUBE=SCUBE + S(I,J)*S(J,K)*S(K,I)
  300 CONTINUE
      SCUBE=SCUBE/3.0
C
C      Arbitrary check to avoid excessively small values of J
C
      XJ=SQRT(0.5*XJ)
      IF(XJ*ARB .LT. XI)  XJ=0.0
C
C      Compute the sine and cosine of three times the "Lode" angle
C
      IF(XJ .GT. SMALL)     SIN3A=1.5*SQRT(3.0)*SCUBE/(XJ*XJ*XJ)
      IF(SIN3A .GT.  1.0)  SIN3A= 1.0
      IF(SIN3A .LT. -1.0)  SIN3A=-1.0
      COS3A=SQRT(1.0 - SIN3A*SIN3A)
C
      RETURN
      END
```

```
      SUBROUTINE ELASTC (PROP,VOID,XI,D,BULK,G)
C
      INTEGER I,J
      REAL PROP(1),D(6,6),VOID,XI,XIL,BULK,G,CONST1,CONST2,TEMP1
C
C        Calculate the elastic bulk and (if necessary) shear moduli
C
      XIL=PROP(7)
      BULK=VOID/(3.0*PROP(2))*(MAX(XI,XIL))
      IF(PROP(5) .GT. 0.5)  THEN                  ! shear modulus is input
          G=PROP(5)
      ELSE
          TEMP1=1.5*(1.0 - 2.0*PROP(5))/(1.0 + PROP(5))  ! Poisson's ratio
          G=TEMP1*BULK                                   !    is input
      END IF
C
C        Calculate elastic portion of the incremental properties
C
      CONST1=BULK + G/0.75
      CONST2=BULK - G/1.50
C
      DO 100 I=1,6                                ! initialize the D array
          DO 100 J=I,6
              D(I,J)=0.0
              D(J,I)=0.0
  100 CONTINUE
      DO 200 I=1,3                                ! load up the diagonal
          D(I,I)=CONST1                           !      elements
          D(I+3,I+3)=G
  200 CONTINUE
      D(1,2)=CONST2                               !  load up the nonzero
      D(1,3)=CONST2                               ! off-diagonal elements
      D(2,3)=CONST2
      D(2,1)=CONST2
      D(3,1)=CONST2
      D(3,2)=CONST2
C
      RETURN
      END
```

```
      SUBROUTINE PLASTC (ITYPE,PROP,DEPT,VOID,XI,XJ,XIO,DDIL,S,SCUBE,
     $                   SIN3A,COS3A,D,BULK,G,DLTA,II,INC,ITNO,BETAS)
C
      INTEGER I,ITYPE,J,K,L,M,N,LL,LFLAG,II(1)
      REAL ALFUN,CV,RT,SINV
      REAL PROP(1),DEPT(3,3),S(3,3),D(6,6),DLTA(3,3),VOID,XI,
     $     XJ,XIO,DDIL,SCUBE,SIN3A,COS3A,BULK,G,XKP,XKPBAR,BETA,
     $     DBETA,DFI,DFJ,DFAL,DFJJ,XN,R,A,H1,TEMP,TEMP1,TEMP2,
     $     TEMP3,TEMP4,TEMP5,TEMP6,TEMP7,BETAS
C
      ALFUN(CV,RT,SINV)=2.0*RT*CV/(1.0 + RT - (1.0 - RT)*SINV)
C
      XN=ALFUN(PROP(3), PROP(4),SIN3A)    ! compute model parameters that
      H1=ALFUN(PROP(17),PROP(18),SIN3A)   !  are function of Lode angle
C
C              Calculate bounding surface parameters
C
      IF(ITYPE .EQ. 3)  THEN
          R=ALFUN(PROP(9), PROP(12),SIN3A)
          A=ALFUN(PROP(10),PROP(13),SIN3A)
          CALL BOUND3 (PROP,S,XN,R,A,VOID,XI,XJ,XIO,SCUBE,XKPBAR,DFI,
     $                 DFJ,DFJJ,DFAL,BETA,BETAS)
      ELSE
          CALL BOUND1 (PROP,S,XN,VOID,XI,XJ,XIO,SCUBE,XKPBAR,DFI,DFJ,
     $                 DFJJ,DFAL,BETA,BETAS)
      END IF
C
      IF(BETAS .LT. 1.0)  CALL ELASTC (PROP,VOID,XI,D,BULK,G)
      DBETA=BETA - 1.0
      IF(DBETA .LT. 0.0)  DBETA=0.0
C
C             Check for elastic zone
C
      TEMP1=BETA - DBETA*PROP(15)
      IF(TEMP1 .GT. 0.0)  THEN
          LFLAG=1
C
C         Calculate the plastic modulus and loading function
C
          CALL LODFUN (ITYPE,PROP,DEPT,XN,H1,XI,XJ,XIO,DDIL,S,SCUBE,
     $                 COS3A,BULK,G,XKP,XKPBAR,VOID,DBETA,TEMP1,DFI,
     $                 DFJ,DFJJ,DFAL,LFLAG)
      ELSE
          LFLAG=0
      END IF
C
C       Calculate plastic portion of the incremental properties
C
      IF(LFLAG .NE. 0)   THEN
          TEMP3=3.0*BULK*DFI
          TEMP4=G*DFJJ
          TEMP5=SQRT(3.0)*G*DFAL
```

```
      TEMP6=XKP + 9.0*BULK*DFI*DFI + G*DFJ*DFJ
     $            + G*(DFAL*COS3A)*(DFAL*COS3A)
      TEMP7=XJ*XJ
      DO 200 M=1,6
          I=II(M)/10
          J=MOD(II(M),10)
          DO 200 N=M,6
              K=II(N)/10
              L=MOD(II(N),10)
C
              TEMP=0.0
              TEMP1=0.0
              TEMP2=0.0
              IF(TEMP7*TEMP7 .NE. 0.0)  THEN
                  DO 100 LL=1,3
                      TEMP1=TEMP1 + S(I,LL)*S(LL,J)
                      TEMP2=TEMP2 + S(K,LL)*S(LL,L)
  100             CONTINUE
                  TEMP1=TEMP5*((TEMP1-1.5*SCUBE*S(I,J)/TEMP7)/TEMP7
     $                -DLTA(I,J)/1.5)
                  TEMP2=TEMP5*((TEMP2-1.5*SCUBE*S(K,L)/TEMP7)/TEMP7
     $                -DLTA(K,L)/1.5)
              END IF
              TEMP=(TEMP3*DLTA(I,J) + TEMP4*S(I,J) + TEMP1)
     $            *(TEMP3*DLTA(K,L) + TEMP4*S(K,L) + TEMP2)/TEMP6
              D(M,N)=D(M,N) - TEMP
              D(N,M)=D(M,N)
  200     CONTINUE
      END IF
      RETURN
      END
```

```
      SUBROUTINE BOUND1 (PROP,S,XN,VOID,XI,XJ,XIO,SCUBE,XKPBAR,DFI,DFJ,
     $            DFJJ,DFAL,BETA,BETAS)
C
C     Subroutine to evaluate relationship of current stress state
C                       to the bounding surface
C           (the latter consisting of a single ellipse)
C
      REAL DFUN,RT,FUN,FUNC
      REAL PROP(1),VOID,XI,XJ,XIO,XIC,XIL,XIBAR,XJBAR,XKPBAR,BETA,
     $     GAM,THETA,DFI,DFJ,DFJJ,DFAL,XN,DNAL,R,C,ARB,BIG,SMALL,
     $     TEMP,TEMP1,TEMP2,TEMP3,TEMP4,TEMP5,SCUBE,BETAS,S(3,3)
C
      DATA ARB/0.001/,BIG/1.0E+20/,SMALL/1.0E-20/
      DFUN(FUN,RT,FUNC)=FUN*FUN*(1.0 - RT)/(2.0*RT*FUNC)
C
      DNAL=DFUN(XN,PROP(4),PROP(3))
      R=PROP(9)
      C=PROP(14)
C
      XIC=XIO*C
      TEMP1=XI - XIC
      IF(ABS(TEMP1) .LT. ARB) TEMP1=ARB
      TEMP2=C - 1.0/R
      TEMP3=TEMP1*TEMP2
      TEMP4=C*(C - 2.0/R)
      TEMP5=TEMP1*TEMP1 + (R - 1.0)*XJ/XN*(R - 1.0)*XJ/XN
C
      BETA=XIO*(-TEMP3+SQRT(TEMP3*TEMP3-TEMP5*(TEMP4+(2.0-R)/R)))
     $      /TEMP5
C
C     Check if stress point is outside bound and if so scale back to it
C
      CALL SKALE (PROP,S,XI,XJ,XIO,SCUBE,BETA,BETAS)
C
C      Compute derivatives of the bounding surface w.r.t. invariants
C           and the value of the "bounding" plastic modulus
C
      XIBAR=BETA*(XI - XIC) + XIC
      IF(XIBAR .EQ. 0.0)  XIBAR=SMALL
      XJBAR=BETA*XJ
      GAM=XIBAR/XIO
      THETA=XJBAR/XIBAR
      XIL=PROP(7)
      TEMP=12.0*VOID/(PROP(1) - PROP(2))*(MAX(XIO,XIL))*XIO*XIO
C
      DFI =2.0*XIO*(GAM - 1.0/R)
      DFJJ=2.0*BETA*(R - 1.0)/XN*(R - 1.0)/XN
      DFJ =DFJJ*XJ
      DFAL=-6.0*XJBAR*(R - 1.0)*(R - 1.0)*DNAL/(XN*XN*XN)
      XKPBAR=TEMP*(GAM - 1.0/R)*(GAM + R - 2.0)/R
      RETURN
      END
```

```
      SUBROUTINE BOUND3 (PROP,S,XN,R,A,VOID,XI,XJ,XIO,SCUBE,
     $                   XKPBAR,DFI,DFJ,DFJJ,DFAL,BETA,BETAS)
C
C     Subroutine to evaluate relationship of current stress state
C                    to the bounding surface
C       (the latter consisting of two ellipses and a hyperbola)
C
      INTEGER IZONE
      REAL DFUN,FUN,FUNC
      REAL PROP(1),VOID,X,XI,XJ,XIO,XIL,XKPBAR,BETA,BETAS,DFI,DFJ,DFJJ,
     $    DFAL,XIC,XN,DNAL,R,DRAL,A,DAAL,Y,C,ARB,BIG,SMALL,T,Q,QC,QO,
     $    FOP,XJO,BT,RHO,XIBAR,THETA,PSI,GAM,DYSAL,DFOPAL,DJOAL,DBTAL,
     $    DRHOAL,TEMP,TEMP1,TEMP2,TEMP3,TEMP4,TEMP5,TEMP6,TEMP7,TEMP8
C
      DATA ARB/0.001/, BIG/1.0E+20/, SMALL/1.0E-20/
      DFUN(FUN,RT,FUNC)=FUN*FUN*(1.0 - RT)/(2.0*RT*FUNC)
C
      DNAL=DFUN(XN,PROP(4),PROP(3))
      DRAL=DFUN(R,PROP(12),PROP(9))
      DAAL=DFUN(A,PROP(13),PROP(10))
      Y=R*A/XN
      C=PROP(14)
      XIC=XIO*C
C
C           Shift projection point
C
      TEMP1=XI - XIC
      IF(ABS(TEMP1) .LT. ARB) TEMP1=ARB
      TEMP2=C - 1.0/R
      TEMP3=TEMP1*TEMP2
      TEMP4=C*(C - 2.0/R)
      Q =XJ/TEMP1
      QC=XN/(1.0 - R*C)
      IF(C .NE. 0.0)   THEN
          QO=XN*(SQRT(1.0 + Y*Y) - (1.0 + Y))/R/C
      ELSE
          QO=-BIG
      END IF
C
      IF(XJ .EQ. 0.0)  THEN
          IF(TEMP1 .GT. 0.0)  THEN
              IZONE=1
          ELSE
              IZONE=3
          END IF
      ELSE IF(C .GE. 1.0/R)  THEN
          IF(Q .GE. 0.0 .OR. Q .LE. QC)  THEN
              IZONE=1
          ELSE IF(Q .GE. QO)  THEN
              IZONE=3
          ELSE
              IZONE=2
```

```
          END IF
        ELSE
          IF(Q .GE. QC)  THEN
            IZONE=2
          ELSE IF(Q .GE. 0.0)  THEN
            IZONE=1
          ELSE IF(Q .LE. QO)   THEN
            IZONE=2
          ELSE
            IZONE=3
          END IF
        END IF
        IF(IZONE .EQ. 1)  THEN
C
C           Projection on ellipse 1
C
        TEMP5=TEMP1*TEMP1 + ((R - 1.0)*XJ/XN)**2
        BETA=XIO*(-TEMP3+SQRT(TEMP3*TEMP3-TEMP5*(TEMP4+(2.0-R)/R)))
     $       /TEMP5
        ELSE IF(IZONE .EQ. 2)  THEN
C
C           Projection on hyperbola
C
          TEMP5=TEMP4 - 2.0*A/R/XN
          TEMP6=XJ*(1.0/R + A/XN)/XN
          TEMP7=TEMP3 + TEMP6
          TEMP8=TEMP1*TEMP1 - (XJ/XN)*(XJ/XN)
          BETA=-0.5*XIO*TEMP5/TEMP7
          IF(TEMP8 .NE. 0.0)
     $         BETA=XIO*(-TEMP7 + SQRT(TEMP7*TEMP7 - TEMP8*TEMP5))/TEMP8
        ELSE
C
C           Projecton on ellipse 2
C
          T=PROP(11)
          TEMP5=SQRT(1.0 + Y*Y)
          FOP=XN/TEMP5
          XJO=A*(1.0 + Y - TEMP5)/Y
          BT=T*(XJO - T*FOP)/(XJO - 2.0*T*FOP)
          RHO=(BT - T)/FOP/XJO
          TEMP6=T - BT + C
          TEMP7=TEMP1*TEMP6
          TEMP8=TEMP1*TEMP1 + RHO*XJ*XJ
          BETA=XIO*(-TEMP7+SQRT(TEMP7*TEMP7
     $            -TEMP8*(TEMP6*TEMP6 - BT*BT)))/TEMP8
        END IF
C
C     Check to see if state point is on outside the surface, and if so,
C                       scale it back to the surface.
C
        CALL SKALE (PROP,S,XI,XJ,XIO,SCUBE,BETA,BETAS)
C
```

```
C      Compute derivatives of the bounding surface w.r.t. invariants
C        and the value of the "bounding" plastic modulus for the
C                        appropriate zone.
C
      XIBAR=BETA*(XI - XIC) + XIC
      IF(XIBAR .EQ. 0.0)  XIBAR=SMALL
      GAM=XIBAR/XIO
      THETA=BETA*XJ/XIBAR
      X=THETA/XN
      XIL=PROP(7)
      TEMP=12.0*VOID/(PROP(1) - PROP(2))*XIO*XIO*(MAX(XIO,XIL))
C
      IF(IZONE .EQ. 1)  THEN
C
C          Normal consolidation zone (ellipse 1)
C
          PSI=Y/((R - 1.0)*(R - 1.0))
          TEMP5=R*(1.0 + X*X + R*(R - 2.0)*X*X)
          DFI=2.0*XIO*(GAM - 1.0/R)*PSI
          DFJJ=2.0*XIO*GAM*((R - 1.0)/XN)**2*PSI*BETA/XIBAR
          DFJ=DFJJ*XJ
          XKPBAR=TEMP*(GAM - 1.0/R)*(GAM + R - 2.0)*PSI*PSI/R
          DFAL=PSI*6.0*(R-1.0)*THETA*GAM*XIO*(((R-1.0)/(R*R*
     $         (2.0/R-GAM-1.0)) + 1.0)*DRAL - (R-1.0)*DNAL/XN)/(XN*XN)
          RETURN
       ELSE IF(IZONE .EQ. 2)  THEN
C
C          Overconsolidated zone (hyperbola)
C
          TEMP5=1.0 - X*(1.0 + Y)
          DFI=2.0*XIO*(GAM - 1.0/R)
          DFJ=2.0*XIO*((1.0 + Y)/R - X*GAM)/XN
          DFJJ=DFJ/XJ
          XKPBAR=TEMP*(GAM - 1.0/R)*(TEMP5*GAM + 2.0*A/XN)/R
          DFAL=6.0*XIO*(DNAL*(THETA*GAM/XN-1.0/R+A/(R*THETA*GAM)
     $        -2.0*A/XN)/(XN*XN)+DRAL*(1.0/THETA-1.0/XN+A/(XN*THETA*GAM))
     $        /(R*R) + DAAL*(1.0/XN - 1.0/(THETA*GAM*R))/XN)
          RETURN
       ELSE
C
C          Tension zone (ellipse 2)
C
          PSI=1.0/(R*(BT - T))
          DFI=2.0*PSI*XIO*(GAM + T - BT)
          DFJJ=2.0*PSI*XIO*GAM*RHO*BETA/XIBAR
          DFJ=DFJJ*XJ
          XKPBAR=TEMP*PSI*PSI*(GAM+T-BT)*(GAM*(BT-T) + T*(2.0*BT-T))
          DYSAL=Y*(DRAL/R + DAAL/A - DNAL/XN)
          DFOPAL=FOP*(DNAL/XN - Y*DYSAL/(1.0 + Y*Y))
          DJOAL=XJO*(DAAL/A - DYSAL/Y) + A*(1.0/Y - FOP/XN)*DYSAL
          DBTAL=((T-BT)*DJOAL - (T-2.0*BT)*T*DFOPAL)/(XJO-2.0*T*FOP)
          DRHOAL=DBTAL/FOP/XJO - RHO*(DFOPAL/FOP + DJOAL/XJO)
```

```
      DFAL=3.0*PSI*XIO*THETA*GAM*(DRHOAL + 2.0*RHO*DBTAL
     $          /(T+GAM-2.0*BT))
      RETURN
      END IF
      END


      SUBROUTINE LODFUN (ITYPE,PROP,DEPT,XN,H1,XI,XJ,XIO,DDIL,S,SCUBE,
     $                   COS3A,BULK,G,XKP,XKPBAR,VOID,DBETA,DENOM,DFI,
     $                   DFJ,DFJJ,DFAL,LFLAG)
C
C      Subroutine to calculate the plastic modulus and loading function
C
      INTEGER I,ITYPE,J,K,LFLAG
      REAL PROP(1),DEPT(3,3),S(3,3),XJ,XIO,DDIL,SCUBE,COS3A,
     $     VOID,BULK,G,XKP,XKPBAR,DBETA,DENOM,DFI,DFJ,DFAL,DFJJ,XN,
     $     H1,H,XLF,SUM1,SUM2,TEMP1,TEMP2,TEMP3,TEMP4,TEMP5,TEMP6
C
C     Get value of hardening function and compute the plastic modulus
C
      CALL GETH (ITYPE,PROP,XN,H1,H,XI,XJ,XIO,DFI,DFJ,VOID)
      XKP=XKPBAR + H*DBETA/DENOM
C
      SUM2=0.0                          ! compute the value of the loading
function
      TEMP1=0.0
      TEMP2=3.0*BULK*DFI
      TEMP3=G*DFJJ
      TEMP4=SQRT(3.0)*G*DFAL
      TEMP5=XKP + 9.0*BULK*DFI*DFI + G*DFJ*DFJ
     $             + G*(DFAL*COS3A)*(DFAL*COS3A)
      TEMP6=XJ*XJ
      IF(TEMP6 .NE. 0.0)  THEN
          DO 200 I=1,3
              DO 200 J=1,3
                  SUM1=0.0
                  DO 100 K=1,3
                      SUM1=SUM1 + S(I,K)*S(K,J)
  100             CONTINUE
                  TEMP1=TEMP1 + (SUM1 - 1.5*SCUBE*S(I,J)/TEMP6)
     $                 *DEPT(I,J)/TEMP6
                  SUM2=SUM2 + S(I,J)*DEPT(I,J)
  200     CONTINUE
          TEMP1=TEMP1 - DDIL/1.5
      END IF
      XLF=(TEMP2*DDIL + TEMP3*SUM2 + TEMP4*TEMP1)/TEMP5
C
C           Check for unloading or neutral loading
C
      IF(XLF .LE. 0.0)  LFLAG=0
      RETURN
      END
```

```
      SUBROUTINE GETH (ITYPE,PROP,XN,H1,H,XI,XJ,XIO,DFI,DFJ,VOID)
C
C        Subroutine to compute the hardening function H
C
      INTEGER ITYPE
      REAL PROP(1),XN,H,H1,H2,XI,XJ,XIO,DFI,DFJ,UNTNOR,R,SM,Z,
     $     VOID,PATM,VAL1,VAL2,TEMP1,TEMP2,TEMP3,TEMP4
C
      PATM=PROP(8)
      R =PROP(9)
      SM=PROP(16)
      H2=PROP(19)
      VAL1=PROP(20)                          ! first experimental const.
      VAL2=PROP(21)                          ! second experimental const.
C
      Z=XJ*R/(XN*XIO)
      TEMP1=Z**SM
      TEMP2=9.0*DFI*DFI + DFJ*DFJ/3.0
      UNTNOR=3.0*DFI/SQRT(TEMP2)
      TEMP3=SIGN(1.0,UNTNOR)
      IF(VAL1 .LE. 0.0 .OR. ITYPE .EQ. 3)  THEN
          TEMP4=1.0                              ! constant h desired
      ELSE
          TEMP4=0.5*(VAL1 + TEMP3*((ABS(UNTNOR))**(1.0/VAL2)))
      END IF
C
      H=VOID/(PROP(1) - PROP(2))*PATM*TEMP2*TEMP4*
     $          (TEMP1*H1 + (1.0 - TEMP1)*H2)
      RETURN
      END
```

```
      SUBROUTINE SKALE (PROP,S,XI,XJ,XIO,SCUBE,BETA,BETAS)
C
C     Subroutine to check if the current stress state lies outside the
C       bounding surface, and if so, scale it back to the surface.
C
      REAL PROP(1),S(3,3),BETA,BETAS,XI,XJ,SCUBE,C,XIC,XIO
C
      C=PROP(14)
      XIC=C*XIO
      BETAS=BETA
      IF(BETA .LT. 1.0)  THEN
         XJ=XJ*BETA
         SCUBE=SCUBE*BETA*BETA*BETA
         XI=BETA*(XI - XIC)+XIC
         DO 200 I=1,3
             DO 100 J=1,3
                 S(I,J)=BETA*S(I,J)
  100        CONTINUE
  200    CONTINUE
         BETA=1.0
      ENDIF
      RETURN
      END
```

APPENDIX II

```
      SUBROUTINE RPROP (ITYPE,PROP)
C
C     This subroutine reads in and modifies the parameters required
C      by the bounding surface plasticity model for cohesive soils.
C
      INTEGER I,ITYPE
      REAL PROP(1)
      READ(5,*) ITYPE
      IF(ITYPE .EQ. 1)  THEN
          READ(5,*) (PROP(I),I=1,9),(PROP(I),I=14,21)
      ELSE                                              ! ITYPE = 3
          READ(5,*) (PROP(I),I=1,19)
      END IF
C
      WRITE(6,900) PROP(1),PROP(3),PROP(2),PROP(4)
      IF(PROP(5) .LT. 0.5)  THEN
          WRITE(6,902) PROP(5)
      ELSE
          WRITE(6,904) PROP(5)
      END IF
      WRITE(6,906) PROP(7),PROP(8)
      IF(ITYPE .EQ. 1)  THEN
          WRITE(6,908) PROP(9),PROP(14),PROP(15)
      ELSE
          WRITE(6,909) PROP(9),PROP(12),PROP(10),PROP(13)
          CALL TCHECK (PROP)
          WRITE(6,911) PROP(11),PROP(14),PROP(15)
      END IF
C
      WRITE(6,910) PROP(16),PROP(19),PROP(17),PROP(18)
      IF(PROP(20) .GT. 0.0)  THEN
          WRITE(6,912) PROP(20),PROP(21)
      ELSE
          WRITE(6,914)
      END IF
C
C        Convert parameters from triaxial to invariant stress space
C
      PROP(3)=PROP(3)/SQRT(27.0)
      PROP(7)=PROP(7)*3.0
      RETURN
C
  900 FORMAT(13X,'TRADITIONAL CLAY MATERIAL PARAMETERS:',/,13X,37('-'),
     $     /,10X,'Lambda =',F7.3,17X,'Mc =',F7.3,/,
     $       10X,'Kappa  =',F7.3,14X,'Me/Mc =',F7.3,/)
  902 FORMAT(19X,'Poisson''s ratio =',F7.3)
  904 FORMAT(18X,'Shear modulus, G =',1PE10.3)
  906 FORMAT(11X,'Transitional stress, P1 =',1PE10.3,/,
     $       14X,'Atmospheric pressure =',1PE10.3,//)
  908 FORMAT( 5X,'Bounding surface shape parameter, R =',F7.3,/,
     $       10X,'Projection center parameter, C =',F7.3,/,
     $       12X,'Elastic nucleus parameter, S =',F7.3,/)
  909 FORMAT(15X,'BOUNDING SURFACE SHAPE PARAMETERS:',/,15X,34('-'),
     $     /,14X,'Rc =',F7.3,14X,'Re/Rc =',F7.3,/,
```

```
   $       14X,'Ac =',F7.3,14X,'Ae/Ac =',F7.3)
910 FORMAT(21X,'HARDENING PARAMETERS:',/,21X,21('-'),/,
   $       15X,'m =',F7.3,17X,'H2 =',F7.3,/,
   $       14X,'Hc =',F7.3,14X,'He/Hc =',F7.3)
911 FORMAT(15X,'T =',F7.3,//,
   $       14X,'Projection center parameter, C =',F7.3,/,
   $       14X,'  Elastic nucleus parameter, S =',F7.3,/)
912 FORMAT(15X,'a =',F7.3,18X,'w =',F7.3)
914 FORMAT(/,10X,3('*'),' The shape hardening function is constant ',
   $            3('*'),/)
    END
```

```
      SUBROUTINE TCHECK (PROP)
C
C        This subroutine checks the value of the bounding surface shape
C   parameter "T" and adjusts this value if it exceeds the theoretical max.
C              Original version written by J.S. De Natale.
C
      REAL TEMP1,TEMP2,TEMPR,PROP(1)
      YFUN(TT)=(1.0 + TT)*SQRT(1.0 + TT*TT) - (1.0 + TT*TT)
C
C        Check against theoretical limit in compression
C
      TEMP1=PROP(11)
      TEMP2=PROP(9)*PROP(10)*SQRT(27.0)/PROP(3)
      TEMP2=YFUN(TEMP2)
      TEMP2=TEMP2/2.0/PROP(9)
      IF(PROP(11) .GT. TEMP2)  PROP(11)=TEMP2
C
C        Check against theoretical limit in extension
C
      TEMPR=PROP(9)*PROP(12)
      TEMP2=TEMPR*PROP(10)*PROP(13)*SQRT(27.0)/PROP(3)/PROP(4)
      TEMP2=YFUN(TEMP2)
      TEMP2=TEMP2/2.0/TEMPR
      IF(PROP(11) .GT. TEMP2)  PROP(11)=TEMP2
      IF(PROP(11) .NE. TEMP1)  WRITE(6,900)
  900 FORMAT(/,7X,'>>>  THE USER-SPECIFIED VALUE OF T EXCEEDS THE MAX',
     $    ' <<<',/,12X,'   PERMISSIBLE VALUE AND HAS BEEN RESET TO:',/)
C
      RETURN
      END
```

APPENDIX III

Input to the RPROP Subroutine

The following quantities are read by the RPROP subroutine: (Note: the input is free-form).

Line 1: (integer)

ITYPE = { 1 — Bounding surface consists of a single ellipse (see discussion below)

ITYPE = { 3 — Bounding surface consists of a combination of two ellipses and a hyperbola (7)

Subsequent line(s): (all parameters are real)

λ

κ

M_c

M_e/M_c

ν or G

Γ

p

P_{atm}

if ITYPE = 1:	if ITYPE = 3:
R	R_c
c	A_c
s	T
m	R_e/R_c

h_c	A_e/A_c
h_e/h_c	c
h_2	s
a	m
w	h_c
	h_e/h_c
	h_2

The Single Ellipse Model

Recently a form of the bounding surface model was developed in which the surface consists of a single ellipse (further details regarding all aspects of this new model are given in reference (18)). Although the adoption of a single ellipse simplifies the explicit definition of the bounding surface, (in previous applications the surface consisted of a combination of two ellipses and a hyperbola (7)), it requires the modiciation of the shape hardening function h (refer to "modification 5" in referenced (7)). More precisely, if the single ellipse is used instead of a "flatter" (with respect to the critical state line) hyperbola in the region to the left of the critical state line undesirably high levels of $\bar{J}$ will be attained at large overconsolidation ratios. The following hardening function is therefore used:

$$\hat{h} = [z^m h_1(\alpha) + (1 - z^m)h_2]\left[\frac{1}{2}\left(a + \text{sign}(n_p)\sqrt[w]{|n_p|}\right)\right]$$

In the above expression n_p represents the component in the p-direction of the unit normal in triaxial space; the parameters a and w control the decrease of $\hat{h}$ (further details are given in reference (18)). The inclusion of the terms in the second set of brackets renders $\hat{h}$ a

function of the ratio $\eta = q/p$. Noting that n_p varies in magnitude from +1 (along the p-axis in a positive direction) to -1 (along the p-axis in a negative direction), the function $\hat{h}$ is seen to decrease abruptly in value when the critical state line is crossed (i.e. when n_p passes through zero).

If, on the other hand, a bounding surface consisting of two ellipses and a hyperbola is specified (i.e., ITYPE = 3), the hardening function used is (7):

$$\hat{h} = [z^m h_1(\alpha) + (1 - z^m)h_2]$$