169
2-4-81
jwp

LA-8428

(2)

MASTER

R-1717

Dr. 2060

Geothermal Energy Enhancement by Thermal Fracture

University of California

LOS ALAMOS SCIENTIFIC LABORATORY

Post Office Box 1663 Los Alamos, New Mexico 87545

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

This work was supported by the US Department of Energy, Division of Geothermal Energy.

Edited by

Karyn Ames

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

LA-8428

UC-66a

Issued: December 1980

Geothermal Energy Enhancement by Thermal Fracture

Ruth B. Demuth
Francis H. Harlow

DISCLAIMER

This book was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Handwritten signature or initials.

CONTENTS

	Page
Abstract	1
I. Introduction	1
II. The Differential Equations	3
III. The Solution Technique	5
IV. Transport of Heat Through the Rock	12
V. Calculation Examples	15
VI. The Computer Program	27
A. General Structure	27
B. The Indexing Notation	29
C. The Input and Derived Data	30
D. The Common Blocks, VAR and ARRAYS	32
E. Output - Prints, Plots	33
F. System Library Subroutine CALLs	35
G. Operating the Terminal	36
VII. Reference	37
Appendix A. REX Computing Code Listing	38

GEOHERMAL ENERGY ENHANCEMENT BY THERMAL FRACTURE

by

Ruth B. Demuth and Francis H. Harlow

ABSTRACT

A large, vertical, circular fracture created deep within hot rock is connected to the surface through two holes. The inlet provides a source of cold water and the outlet extracts heated water. Cooling of the rock produces thermal stresses that fracture the rock adjacent to the primary crack, thereby enhancing the heat extraction rate by means of convective transport. The properties of the thermal fracture network vary with position and time. The REX code for high-speed computer was written and used to study the coupled processes of primary-crack flow and lateral thermal fracture heat transport. Calculations for elapsed times of 100 yr show that thermal fracture enhancement can double the heat extraction rate over the results from conduction alone. Long-term enhancement predictions depend on data from rock-mechanics studies, which the REX code is prepared to accept as they become available.

I. INTRODUCTION

The extraction of heat energy from hot dry rocks depends upon the establishment of a large fracture through which water can circulate. The fracture is created by pressure through a drill hole from the surface, and ideally occurs in a vertically oriented, circular configuration, perhaps a kilometer in radius, but only a fraction of a centimeter in gap width.

A second hole must be drilled to circulate water through this large crack. The inlet to the crack provides a source of relatively cold water, and the outlet from the crack, usually located somewhat above the inlet, removes the heated water. At the surface, the energy is extracted by means of a heat exchanger, and the water is returned in a closed loop to be heated again. Throughout the entire circuit the water pressure is maintained below the boiling level.

As an example, a typical system pumps water at the rate of $10^5 \text{ cm}^3/\text{sec}$ through rock that was initially 300°C . If 100°C water goes down into the crack, then the energy extraction rate is 83 MW, which can be converted to enough electricity to satisfy the needs of a small city. The initial heat extraction rate, however, cannot be maintained and soon drops because cooling forms a thermal skin between the crack and the nearby hot rock. The skin grows in thickness with time, forming a progressively greater insulation to the conduction of heat.

The extent of the power drop depends on various factors, some within the control of the plant engineer and some that depend on the changing state of the rock itself. In this latter category is the potentiality for enhancement of heat extraction rate as a result of thermal stress fractures. Because the cooler rock has contracted, it is expected that a system of lateral fractures will develop, branching from the primary crack and extending into the adjacent rock in such a way as to allow convective exchange of heat. The result can be a significant increase in the power output over the results of conduction alone.

This report is the second in a series of studies that investigate the thermal fracture effects in the enhancement of geothermal energy extraction. The first report discusses the configuration and describes the heat and water transport through a single secondary fracture.¹ The results of that study have been used to characterize the single-fracture exchange effects in such a way as to be useful in the overall investigation of multiple crack effects, which are the subject of this report. The third phase of the study will examine in more detail the transport properties through a connected maze of lateral thermal stress fractures.

The techniques used in each phase of our study are purely theoretical and consist of three essential parts.

(1) The basic natural laws of mass, momentum, and energy conservation are formulated in differential equations appropriate to the circumstances. These are supplemented with the specification of initial and boundary conditions, material properties, exchange rates, and the derivation of transport models.

(2) The differential formulations are replaced by finite-difference approximations suitable for solution by high-speed computer. Numerical analysis techniques are developed in such a way as to ensure numerical stability, efficiency, and a sufficient degree of accuracy for calculations extending over long periods of elapsed time, typically up to 100 yr.

(3) Numerous examples are calculated in order to investigate sensitivity to our assumptions and to exhibit the results that can be expected under a variety of circumstances.

In this study of the overall system as influenced by the presence of lateral thermal stress fractures, the central tool for our investigations is the computer code REX (Rock Energy Extraction), which has demonstrated capability for efficient and accurate long-time calculations through numerous proof tests and applications. The REX code is written to accept whatever data may be provided by the rock-mechanics investigators; and as this report shows, the predictions regarding long-time energy extraction rates depend strongly on the inclusion of accurate descriptions of the rock behavior. Insofar as the results of calculations presented in this report are concerned, any conclusions must be considered tentative pending the refinement by other experts of the models incorporated at this stage. Our emphasis is on the heat and mass transfer; the prediction of fracturing response to cooling is beyond the scope of our current activities.

The plan of this report closely follows that of Ref. 1. We assume that the reader is familiar with that report, although enough of it is outlined here to make this discussion self-contained.

II. THE DIFFERENTIAL EQUATIONS

We assume the Boussinesq approximation for a fluid with density that varies as a result of thermal expansion by a small fraction of its nominal value ρ_0 . Compared to sound speed, the fluid velocity is small enough for the material to be considered otherwise incompressible. Accordingly, the conservation of mass in three space dimensions can be described by a balance of volumetric fluxes. With fluid velocity $\vec{u}$ we have

$$\nabla \cdot \vec{u} = S_{in} + S_{out} + S_c, \quad (1)$$

in which S_{in} and S_{out} are the volumetric inlet source and outlet sink, and S_c is a volumetric sink into the growing lateral cracks. For the momentum equation, we neglect any momentum source or sink at the inlet or outlet. The Boussinesq form is

$$\frac{\partial \vec{u}}{\partial t} + \nabla \cdot (\vec{u}\vec{u}) = -\nabla \bar{\phi} + (\nabla \cdot \nu \nabla) \vec{u} + \vec{g} \frac{\rho}{\rho_0}, \quad (2)$$

in which $\bar{\phi}$ is the ratio of pressure to nominal density, ν is the kinematic viscosity coefficient, and $\vec{g}$ is the acceleration of gravity. Incorporating Eq. (1), we have transformed the usual convective term $\vec{u} \cdot \nabla \vec{u}$ into the divergence of a dyadic in order to produce a conservative form as a basis for the finite difference equations. The ratio of actual to nominal density, ρ/ρ_0 , can be described in terms of water temperature T , a reference temperature T_0 , and volumetric coefficient of expansion β , by the relation

$$\frac{\rho}{\rho_0} = 1 - \beta(T - T_0) \quad . \quad (3)$$

With T in degrees Celsius and ν in units of cm^2/sec , it is sufficiently accurate for our purposes to write

$$\nu = \frac{0.279}{T - 3.8} \quad . \quad (4)$$

As in Ref. 1, we introduce a modified pressure ϕ such that

$$\nabla \phi = \nabla \bar{\phi} - \vec{g}[1 - \beta(\bar{T} - \bar{T}_0)] \quad , \quad (5)$$

in which $\bar{T}$ is the horizontally averaged temperature across the span of the primary fracture, therefore a function only of height and time. Accordingly, Eq. (2) becomes

$$\frac{\partial \vec{u}}{\partial t} + \nabla \cdot (\vec{u} \vec{u}) = - \nabla \phi + (\nabla \cdot \nu \nabla) \vec{u} + \beta \vec{g}(\bar{T} - T) \quad . \quad (6)$$

For the water temperature equation,

$$\frac{\partial T}{\partial t} + \nabla \cdot (\vec{u} T) = S_{\text{in}} T_{\text{in}} + S_{\text{out}} T_{\text{out}} + \text{wall source} \quad . \quad (7)$$

The wall source describes two effects: conduction through the rock and convection through the lateral thermal stress fractures. Our model for these fluxes is discussed in Sec. IV.

III. THE SOLUTION TECHNIQUE

The transformation of the three-dimensional differential equations into two-dimensional finite-difference equations in the plane of the primary crack follows the technique of Ref. 1. W denotes the gap width, which can vary with position and time. Our technique, however, ignores the time derivatives of W because at each stage of the evolution we hold the rock properties constant and seek the corresponding pseudo-steady-state velocity and temperature configuration. The validity of this approach is demonstrated by the existence of two very different time scales: (1) the time required for appreciable changes in the state of the adjacent rock, and (2) the much shorter time required for the water velocity and temperature to adjust to changes in the rock state.

Our coordinate system in the plane of the primary crack is Cartesian, with the origin placed to the lower left at a sufficient distance to allow the entire crack to lie in a rectangular space with positive coordinates (Fig. 1). The horizontal distance from the origin is x and the vertical distance is y .

The region of interest is divided into finite-difference computational cells of width δx and δy in the plane of resolution, and extending a distance $W/2$ in

Fig. 1.

The primary crack with its grid of finite-difference cells. The inlet and outlet cells are labeled I and O, respectively.

both the positive and negative z directions away from the plane of symmetry through the center of the primary crack. Figure 2 shows the locations in the $x-y$ plane to which our subscript notation refers. In particular, cell centers are labeled with integers i and j , with i increasing along the x axis from 2 in the leftmost cell to $\bar{i} + 1$ at the far right. Likewise, j increases in the positive y direction from 2 in the bottom row of cells to $\bar{j} + 1$ at the top. Thus, the cell-number dimensions of the finite-difference mesh resolving the system are $\bar{i}$ by $\bar{j}$.

The x and y components of water velocity, $u_{i+\frac{1}{2}j}$ and $v_{ij+\frac{1}{2}}$, refer to appropriate spatial averages around the edge positions described by the sub-

Fig. 2.

Subscript notation for positions relative to a computational cell in the $x-y$ plane.

scripts. Likewise, ϕ_{ij} , W_{ij} , and T_{ij} refer to cell-centered quantities.

To derive the finite-difference equations for heat and mass transfer in the crack, we integrate the equations over the three-dimensional volume contained in a cell of dimensions δx by δy by W . In this process, we assume that the velocity components vary with z in a Poiseuille profile. Relatively greater buoyancy near the edges of the crack, together with the possible turbulence and the effects of wall irregularity, would induce departures from this idealization. However, because of our lack of precise information about these effects, we write

$$u_{i+\frac{1}{2},j} \left[1 - \left(\frac{2z}{W_{i+\frac{1}{2},j}} \right)^2 \right]$$

and

$$v_{i,j+\frac{1}{2}} \left[1 - \left(\frac{2z}{W_{i,j+\frac{1}{2}}} \right)^2 \right]$$

for the two components of velocity in the localities designated by the subscripts. Working first with Eq. (1), we can arrange its integral to the form

$$D_{ij} \equiv \frac{2}{3} \left[\frac{(W'\tilde{u})_{i+\frac{1}{2},j} - (W'\tilde{u})_{i-\frac{1}{2},j}}{\delta x} + \frac{(W'\tilde{v})_{i,j+\frac{1}{2}} - (W'\tilde{v})_{i,j-\frac{1}{2}}}{\delta y} \right]$$

$$- \frac{1}{\delta x \delta y} (\dot{V}_{ij} - \dot{V}_{cij}) = 0 \quad .$$

(8)

We have put

$$(S_{in})_{ij} \equiv \frac{\dot{V}_{ij}}{W_{ij} \delta x \delta y}$$

for $\dot{V}_{ij} < 0$, and

$$(S_{out})_{ij} \equiv \frac{\dot{V}_{ij}}{W_{ij} \delta x \delta y}$$

for $\dot{V}_{ij} \geq 0$. Thus, $\dot{V}_{ij}$ refers to the total volume of water per unit time entering cell number i, j , with $\dot{V}_{cij}$ denoting the net volume of water per unit time entering the lateral thermal fractures connected to cell number i, j . The prime on W indicates that the magnitude of the gap width is the lesser of the values in the two adjacent cells. The tilde, here and in the following equations, denotes the new time values, whose presence throughout the equations denotes a somewhat higher degree of implicitness than in Ref. 1.

As in Ref. 1, we can derive the appropriate forms for the two components of our momentum equation, which are

$$\begin{aligned} W_{i+\frac{1}{2}j} \frac{\tilde{u}_{i+\frac{1}{2}j} - u_{i+\frac{1}{2}j}^n}{\delta t} = \frac{8}{15} \left[\frac{\langle Wu:\tilde{u} \rangle_{ij} - \langle Wu:\tilde{u} \rangle_{i+1j}}{\delta x} + \frac{\langle W'v:\tilde{u} \rangle_{i+\frac{1}{2}j-\frac{1}{2}} - \langle W'v:\tilde{u} \rangle_{i+\frac{1}{2}j+\frac{1}{2}}}{\delta y} \right] \\ + W_{i+\frac{1}{2}j} \frac{\tilde{\phi}_{ij} - \tilde{\phi}_{i+1j}}{\delta x} - \frac{8\nu}{W'_{i+\frac{1}{2}j}} \tilde{u}_{i+\frac{1}{2}j} \end{aligned} \quad (9)$$

and

$$\begin{aligned} W_{ij+\frac{1}{2}} \frac{\tilde{v}_{ij+\frac{1}{2}} - v_{ij+\frac{1}{2}}^n}{\delta t} = \frac{8}{15} \left[\frac{\langle vW:\tilde{v} \rangle_{ij} - \langle vW:\tilde{v} \rangle_{ij+1}}{\delta y} + \frac{\langle uW':\tilde{v} \rangle_{i-\frac{1}{2}j+\frac{1}{2}} - \langle uW':\tilde{v} \rangle_{i+\frac{1}{2}j+\frac{1}{2}}}{\delta x} \right] \\ + W_{ij+\frac{1}{2}} \frac{\tilde{\phi}_{ij} - \tilde{\phi}_{ij+1}}{\delta y} - \frac{8\nu}{W'_{ij+\frac{1}{2}}} \tilde{v}_{ij+\frac{1}{2}} \\ + W_{ij+\frac{1}{2}} \beta g \left[\frac{T_{ij} + T_{ij+1}}{2} - \bar{T}_{j+\frac{1}{2}} \right]. \end{aligned} \quad (10)$$

As before, the prime on a W means the lesser of two values; in this case the two values are averages, one for the pair of cells upstream of the flow, the other for the downstream pair. We have used donor-cell notation for the convective fluxes. Thus, for any quantities, A and B,

$$\langle Au:B \rangle_{ij} \equiv (Au)_{ij} \cdot \begin{cases} B_{i-\frac{1}{2}j} & \text{if } u_{ij} > 0 \\ B_{i+\frac{1}{2}j} & \text{if } u_{ij} < 0 \end{cases} .$$

In general, the part before the colon is written with the subscripts of the whole term. The part following the colon bears the subscripts denoting a half cell behind the direction of the velocity that occurs in the part before the colon. Except as noted, wherever a quantity like u_{ij} is indexed in reference to a position other than its primary locality, the nomenclature denotes an average of the values at the adjacent true localities.

One section of the REX code solves Eqs. (8) through (10) simultaneously, in which the temperatures are held constant. The procedure is straightforward, but requires extensive algebraic manipulation to derive the appropriate Newton-Raphson iteration expressions. In essence, the goal is to find the zeros of $\tilde{D}_{ij}$ in Eq. (8) subject to variations of $\tilde{\phi}_{ij}$ under the constraints of Eqs. (9) and (10). Each step of the iteration makes use of an equation for the change in $\tilde{D}_{ij}$ that results from a change in $\tilde{\phi}_{ij}$, namely,

$$\delta \tilde{D}_{ij} = \frac{\partial \tilde{D}_{ij}}{\partial \tilde{\phi}_{ij}} \delta \tilde{\phi}_{ij} .$$

With the requirement that $\delta \tilde{D}_{ij}$ describe the change from a current value to zero, we can replace $\delta \tilde{D}_{ij}$ by $-\tilde{D}_{ij}$, solve for $\delta \tilde{\phi}_{ij}$, and thus derive

$$\tilde{\phi}_{ij}(\text{new}) = \tilde{\phi}_{ij}(\text{old}) - \gamma_{ij} \tilde{D}_{ij} , \quad (11)$$

where

$$\frac{1}{\gamma_{ij}} \equiv \frac{\partial \tilde{D}_{ij}}{\partial \tilde{\phi}_{ij}} .$$

The magnitude of γ_{ij} determines the rate of convergence of the iterative process, but otherwise has no effect on the results. To find an expression for γ_{ij} , we differentiate Eq. (8) with respect to the various velocity components, each multiplied by the derivative of that component with respect to $\tilde{\phi}_{ij}$. To find these $\tilde{\phi}_{ij}$ derivatives, we use Eqs. (9) and (10). The resulting lengthy expression for γ_{ij} is then shortened by relocating the variables at surrounding mesh points to the cell center. The result is

$$\gamma_{ij} = \frac{3 \left[W_{ij}^2 + 8v \delta t + \frac{8W_{ij}^2 \delta t}{15} \left(\frac{|u_{ij}|}{\delta x} + \frac{|v_{ij}|}{\delta y} \right) \right]}{4W_{ij}^3 \delta t \left(\frac{1}{\delta x^2} + \frac{1}{\delta y^2} \right)} . \quad (12)$$

With a new value for $\tilde{\phi}_{ij}$ in each iteration, Eqs. (9) and (10) are solved simultaneously for the new values of $\tilde{u}_{i+\frac{1}{2}j}$ and $\tilde{v}_{ij+\frac{1}{2}}$. In the performance of these solutions, we algebraically isolate only those values at the mesh points shown by the subscripts; the values at the other mesh points are those from the previous iteration level. This seemingly dubious simplification has been amply justified by demonstrations of convergence in numerous test cases. Rather than exhibiting those lengthy expressions, we refer the reader to the full listing of the computer code included in this report.

The other principal calculation in each cycle is for the water temperature. To derive the appropriate equations, we start with Eq. (7) and integrate over a three-dimensional cell δx by δy by W . As in Ref. 1, we also include an extra term, $\tilde{T}_{ij} \tilde{D}_{ij}$, which would vanish if the pressure-velocity iteration were forced to converge perfectly, but which, in the absence of such convergence, has once again been demonstrated to serve an essential purpose for attainment of accurate temperature solutions. After manipulation, Eq. (7) becomes

$$\begin{aligned}
0 = & W_{ij} \frac{\tilde{T}_{ij} - T_{ij}^n}{\delta t} - \tilde{T}_{ij} \tilde{D}_{ij} - \frac{1}{\delta x \delta y} (\dot{V}_{ij} T_e - \dot{V}_{cij} \tilde{T}_{ij}) \\
& - \frac{2}{3} \frac{\langle uW':\tilde{T} \rangle_{i-\frac{1}{2}j} - \langle uW':\tilde{T} \rangle_{i+\frac{1}{2}j}}{\delta x} + \frac{\langle vW':\tilde{T} \rangle_{ij-\frac{1}{2}} - \langle vW':\tilde{T} \rangle_{ij+\frac{1}{2}}}{\delta y} \\
& - \frac{\rho_R b_R \dot{V}_{cij}}{\rho_W b_W \delta x \delta y \beta_R}, \tag{13}
\end{aligned}$$

where $T_e = T_{in}$ if $\dot{V}_{ij} > 0$ and $T_e = T_{ij}$ if $\dot{V}_{ij} < 0$. The last term contains ρ and b , the density and specific heat, subscripted with W and R for water and rock, respectively, together with β_R , the coefficient of thermal expansion of the rock. The essential feature of this form of the term is its expression of the relationship between heat extracted (by both conduction and convection through the thermal stress fractures) and the total volume of rock that has been cooled. This point is discussed in much more detail in Sec. IV.

The procedure for solving Eq. (13) is much like the technique employed for the pressure, Eq. (11). We look for the zeros of Eq. (13) subject to variations of $\tilde{T}_{ij}$, using a Newton-Raphson iteration, for which the equations are given in complete detail in the REX computer listing. As we shall show, the unknown $\tilde{T}_{ij}$ occurs even in the last term of Eq. (13). Its presence there is nonlinear so that the derivative of Eq. (13) with respect to $\tilde{T}_{ij}$ results in a rather lengthy expression.

The overall flow of logic for a computational cycle is shown schematically in Fig. 3, and detailed in the REX computer listing (App. A). In each cycle the iterative solution for pressures and velocities continues to adequate convergence before proceeding to the iterative solution of temperature, which also is forced to adequate convergence before we test for the achievement of a pseudo-steady state. The criterion for pseudo-steady state is the constancy of temperatures between the previous and current subcycles. When this is adequately achieved, the terms $\frac{\partial u}{\partial t}$, $\frac{\partial v}{\partial t}$, and $\frac{\partial T}{\partial t}$ in the momentum and temperature equations all vanish, and the value of δt used in the subcycling becomes irrelevant except that its value must be small enough to avoid numerical instability. Likewise, $\frac{\partial W}{\partial t}$ would vanish if it had been included, but its presence is not required. Indeed, it might be supposed

Fig. 3.

The sequence of calculations for a cycle.

The number of subcycles per cycle is typically 1 to 20, depending principally on the time interval per cycle. In contrast to the small magnitude of δt used in the subcycling, the actual elapsed time per cycle, denoted by DTT in the code, commences with a magnitude 20 sec and increases so that it approaches 7.5% of the elapsed time, accordingly to an interval of 7.5 yr at the end of a 100-yr calculation. It is during that last part of each run that the number of subcycles per cycle rises to as great as 20 or so. If we artificially constrain the number of subcycles to some smaller number than needed for pseudo-steady state, the most apparent manifestation of inaccuracy is visible in the inconsistency between total volume of cooled rock and total amount of energy extracted, for which there must always be the constant of proportionality, $\rho_R b_R / \beta_R$. (In the computer terminal output, the energy extraction rate is converted to megawatts and the total energy to megawatt-seconds, so that this conversion must be considered in any check for consistency.)

When the pseudo-steady state has been attained, the computer calculates the new configuration of the rock, does some bookkeeping, and then either terminates the run or goes on to the next cycle.

that we also could omit the terms $\frac{\partial u}{\partial t}$, $\frac{\partial v}{\partial t}$, and $\frac{\partial T}{\partial t}$ and iterate to convergence at a pseudo-steady state in one subcycle, but this is not feasible for two reasons. First, we need the simultaneous constancy for temperatures and velocities, which, however, are strongly coupled among each other, so that subcycling is the only alternative to an extremely complicated simultaneous solution procedure. Second, the difference formulations for the momentum equations exhibit a seemingly incurable numerical instability in the absence of the velocity-component time derivatives. In any case, subcycling turns out to be a highly efficient procedure for achieving the long elapsed-time computer runs that are necessary for our investigations.

IV. TRANSPORT OF HEAT THROUGH THE ROCK

To derive the last term in Eq. (13), let V_{cij} be the total volume of the lateral thermal stress fractures associated with both sides of the primary crack in cell number i, j , called WC_{ij} in the code. We define a penetration depth R_{ij} such that

$$V_{cij} \equiv 2f\beta_R \delta x \delta y R_{ij} (\xi_\infty - T_{ij}) \quad . \quad (14)$$

The factor of two is for both sides of the crack; f is a rock-temperature profile "shape" factor, referring to the variation of $\xi_\infty - T_{Lij}(z)$ as a function of z , the distance from the primary crack. ξ_∞ is the initial rock temperature, and $T_{Lij}(z)$ is the local water temperature, which equals T_{ij} where $z = 0$. For a linear variation of $\xi_\infty - T_{Lij}(z)$, $f = 0.5$. With conduction alone, the profile would more nearly resemble an error-function integral, but because of the convective heat transfer through the thermal stress fractures, a linear profile may be more appropriate. (This matter will receive more attention in the third phase of our investigations, to be reported later.) Here, we have chosen $f = 0.5$.

The formulation in Eq. (14) ignores the existence of a threshold thermal stress for the initial opening of the lateral cracks. The origin of this threshold arises partly from the pre-existing earth stresses, and partly from the intrinsic cohesiveness of the material. The effects of including such a threshold are discussed in Ref. 1, in which we found that ignoring the threshold resulted in premature openings for which the impedance was, however, so high as to result in a negligible modification to the conclusions. As further rock data becomes available, it may be necessary to include the effects of such a threshold.

Our fundamental postulate states that

$$\frac{dR_{ij}}{dt} \equiv \dot{R}_{ij} = \frac{2\rho_w b}{\rho_R b_R R_{ij}} (\kappa_{oij} + \sigma_{ij}) \quad , \quad (15)$$

in which $\kappa_{oij} = k_R / (\rho_w b_w)$, where k_R is the heat conduction coefficient for the rock except at early times when the water temperature is almost that of the initial rock temperature. At that stage, it is convenient in the code to take $\kappa_{oij} = 0$, and have it thereafter grow to its asymptotic value, given above, according to

$$\kappa_{oij}^{n+1} = \frac{\kappa_{oij}^n + \lambda \frac{k_R}{\rho_w b_w} (\xi_\infty - T_{ij})}{1 + \lambda (\xi_\infty - T_{ij})} ,$$

in which λ is an appropriate relaxation factor.

The crucial quantity in Eq. (15), however, is σ_{ij} , the coefficient for convective transport through the lateral thermal stress fractures. To see the significance of this function, put $f = 0.5$ in Eq. (14) and differentiate with respect to time, neglecting for the moment the time derivative of the temperature.

$$\frac{dV_{cij}}{dt} \equiv \dot{V}_{cij} = \beta_R \delta x \delta y \dot{R}_{ij} (\xi_\infty - T_{ij}) .$$

With this, the last term in Eq. (13) becomes

$$- \frac{\rho_R b_R}{\rho_w b_w} \dot{R}_{ij} (\xi_\infty - T_{ij}) .$$

With Eq. (15) we obtain

$$- 2(\kappa_{oij} + \sigma_{ij}) \frac{\xi_\infty - T_{ij}}{R_{ij}} . \quad (16)$$

The factor of 2 relates to both sides of the primary crack. $\kappa_{oij} + \sigma_{ij}$ is the sum of two thermometric conductivities, one arising from the thermal conductivity of the rock, the other being a convective conductivity. Finally, $(\xi_\infty - T_{ij})/R_{ij}$ estimates the temperature gradient over which the conductivities are operative. Although in the REX code we use the form of the last term shown in Eq. (13), we have also calculated with the form in Eq. (16) with almost identical results, showing that our interpretation of this procedure in terms of energy flux is correct.

The remainder of this section considers the determination of σ_{ij} . Our current formulation is based on heuristic arguments, supplemented by the results from Ref. 1, which are applicable only to the early stages of thermal fracture formation. At first the thermal stress fractures can legitimately be considered isolated plane cracks, vertically oriented. The convective patterns would then be

like those calculated with the GEO code.¹ As the thermal stress fractures grow, however, they are likely to kink, intersect, perhaps terminate or coalesce, and evolve into a maze of interconnecting channels of varying thickness and orientation. At that stage, the GEO calculations must become irrelevant. To describe the heat transport through such a maze will require a detailed investigation of the buoyancy-driven flow through a set of interconnected channels, which will be reported in the next phase of our study.

For now, we approach the problem with the guidance of random-walk theory. The dimensionality of σ_{ij} is the product of a distance by a velocity, and random-walk theory suggests that the distance is the length scale of the crack maze, while the velocity is the circulation speed resulting from a balance between buoyancy and drag. The length scale in the REX code is designated DC; for it we take

$$DC_{ij} = R_{ij} \quad \text{for } R_{ij} \leq DCO$$

and

$$DC_{ij} = DCO \quad \text{for } R_{ij} > DCO ,$$

in which DCO is a specified property of the rock.

For the circulation speed s we balance buoyancy with drag.

$$C_{ij} \beta_w g (\xi_\infty - T_{ij}) \frac{DC_{ij}}{R_{ij}} = \frac{8\omega v}{C_{ij}} s ,$$

in which C_{ij} is the crack width, ω is a friction factor such that $\omega \geq 1.0$, and DC_{ij}/R_{ij} describes the fraction of the entire temperature difference that is operative in producing net buoyancy in the circulation element. The magnitude of C_{ij} can be calculated by accounting for the total open volume produced by rock contraction, which leads to

$$C_{ij} = \frac{1}{2} DC_{ij} \beta_R (\xi_\infty - T_{Lij})$$

Assembling all these elements together with Eq. (4) and inserting a dimensionless factor of order unity σ_o , we obtain

$$\sigma_{ij} = \frac{\sigma_o \beta_R^2 \beta_w g (DC_{ij})^4}{71.424 \omega R_{ij}} (\xi_\infty - T_{ij})^3 (\xi_\infty + T_{ij} - 7.6) . \quad (17)$$

Alternatively, in more universal form,

$$\sigma_{ij} = \frac{\sigma_o \beta_R^2 \beta_w g (DC_{ij})^4}{128 \omega v_{ij} R_{ij}} (\xi_\infty - T_{ij})^3 . \quad (17a)$$

This expresses the current thermal stress fracture heat transport model used in the REX code. The ratio σ_o/ω is expected to lie close to 1.0, and it is seen from the computer results that this value does, indeed, produce reasonable results.

Many of the equations described in this section appear in the REX code, sometimes with Fortran nomenclature at variance with the symbols used here in order to take advantage of multiple storage. However, the meanings of otherwise ambiguous variable names are explained in comment statements, which we hope will be adequate to the readers' needs. Note that the implicit formulation of the temperature equation requires updated quantities that are used also for the rock-status update after the subcycling has been completed.

V. CALCULATION EXAMPLES

Figure 4 shows a typical configuration after an elapsed time of 100 yr. The primary crack is 0.95 km in diameter and 0.1 cm in gap width at the middle. The velocity vectors and the profiles of modified pressure both show the inlet (lower right) and outlet (upper left) for water flowing at the rate of $1.0 \times 10^5 \text{ cm}^3/\text{sec}$. The inlet temperature is 100°C and the initial rock temperature is 300°C . All the relevant conditions for this calculation are given in Table I.

A sequence of temperature contour plots for that same calculation is given in Fig. 5; the earliest is for a time of approximately 1 day and the latest is at the end of 100 yrs.

Fig. 4.

Velocity vectors (left) and modified pressure (right) for a finely resolved calculation, at an elapsed time of 100 yr. The longest velocity vector corresponds to a maximum speed of 101 cm/sec and the contour interval for modified pressure is $1.02 \times 10^6 \text{ dyn/cm}^2$.

Table II describes the input parameters for the same configuration, calculated with a coarsely resolved computing mesh in which δx and δy have been doubled. Velocity vectors and contours of modified pressure, shown in Fig. 6, have the same appearance as those of Fig. 4. The magnitudes of the numbers on the plots differ, however, because of the change in resolution near source and sink. The maximum speed, for example, would become infinite as the cell size around the source approached zero. Nevertheless, the energy extraction rate differs between the two calculations by less than 15% over the entire 100 yr, so that we used the coarse resolution for the rest of the examples described here because of computer cost.

Energy extraction rate is the principal quantity of interest in this investigation. To assess the effects of thermal stress fracture on the enhancement of energy extraction, we performed a set of coarsely resolved calculations that varied from the data in Table II only in the value of σ_0/ω (called SIGOMEG in the table). The results are shown in Fig. 7. The case with conduction alone corresponds to $\sigma_0/\omega = 0.0$. Our best present guess for this parameter is $\sigma_0/\omega = 1.0$, which shows appreciable thermal fracture enhancement to the energy extraction rate. The cumulative energy extraction is shown in Fig. 8, which, as in Fig. 7,

TABLE I

INPUT PARAMETERS FOR EXAMPLE
CALCULATION, FIGS. 4 AND 5

(Values, cgs; temperatures °C)

ALAM	=	0.1
BW	=	1.0
BR	=	0.25
RHOW	=	1.0
RHOR	=	2.65
AKR	=	6.2×10^{-3}
BETA	=	2.07×10^{-4}
BETAR	=	8.3×10^{-6}
DCO	=	25.0
DX	=	5.0×10^3
DY	=	5.0×10^3
EPSILON	=	2.0×10^{-3}
GRAV	=	980.0
IBAR	=	21
IIN	=	14
IOUT	=	8
JBAR	=	21
JIN	=	9
JOUT	=	15
RPC	=	5.0×10^4
SIGOMEG	=	0.02
TFAC	=	0.075
TIN	=	100.0
VDOTIN	=	1.0×10^5
VIN	=	0.0
WO	=	0.1
XII	=	300.0

TABLE II

INPUT PARAMETERS FOR EXAMPLE
CALCULATION, FIG. 6

(Values, cgs; temperatures °C)

ALAM	=	0.1
BW	=	1.0
BR	=	0.25
RHOW	=	1.0
RHOR	=	2.65
AKR	=	6.2×10^{-3}
BETA	=	2.07×10^{-4}
BETAR	=	8.3×10^{-6}
DCO	=	25.0
DX	=	1.0×10^4
DY	=	1.0×10^4
EPSILON	=	2.0×10^{-3}
GRAV	=	980.0
IBAR	=	11
IIN	=	8
IOUT	=	6
JBAR	=	11
JIN	=	5
JOUT	=	8
RPC	=	5.0×10^4
SIGOMEG	=	.02
TFAC	=	0.075
TIN	=	100.0
VDOTIN	=	1.0×10^5
VIN	=	0.0
WO	=	0.1
XII	=	300.0

Fig. 5.

Four water temperature contours at 1.08×10^5 , 1.96×10^6 , 3.53×10^7 , and 3.13×10^9 sec (100 yr), for the same calculation illustrated in Fig. 4. The minimum temperature is 100°C at the inlet, and the maximum is 300° at the outer edges of the primary crack.

Fig. 6.

Velocity vectors (left) and modified pressure (right) for the coarsely resolved version of the calculation shown in Fig. 4, at an elapsed time of 100 yr. The longest velocity vector corresponds to a speed of 61 cm/sec, and the contour interval for modified pressure is 7.6×10^5 dyn/cm².

Fig. 7.

Energy extraction rate as a function of time for various values of σ_0/ω .

Fig. 8.

Cumulative energy extraction as a function of time for two values of σ_0/ω .

somewhat higher in the primary crack. The data are given in Table III. With $\sigma_0/\omega = 1.0$ and the volumetric water flow rate of $1.0 \times 10^5 \text{ cm}^3/\text{sec}$, the calculations resemble those described by Figs. 6 through 8, the only differences being in

shows a factor of two enhancement for $\sigma_0/\omega = 5.0$. The effects of σ also can be exhibited by the variations of its ratio for κ_0 (Fig. 9).

A similar pair of calculations, illustrated in Fig. 10, shows the effect of cutting the volumetric water flow rate to $5 \times 10^4 \text{ cm}^3/\text{sec}$, with all other data remaining as in Table II.

The REX code is capable of examining the effects of variations in the primary crack configuration, in whatever manner may be dictated by rock-mechanics investigations. To illustrate this capability, we calculated a pair of examples in which the inlet position was moved

Fig. 9.

Ratio of σ to κ_0 as a function of time.

TABLE III

INPUT PARAMETERS FOR EXAMPLE CALCULATION, FIG. 11

(Values, cgs; temperatures °C)

ALAM	=	0.1
BW	=	1.0
BR	=	0.25
RHOW	=	1.0
RHOR	=	2.65
AKR	=	6.2×10^{-3}
BETA	=	2.07×10^{-4}
BETAR	=	8.3×10^{-6}
DCO	=	25.0
DX	=	1.0×10^4
DY	=	1.0×10^4
EPSILON	=	2.0×10^{-3}
GRAV	=	980.0
IBAR	=	11
IIN	=	8
IOUT	=	6
JBAR	=	11
JIN	=	7
JOUT	=	8
RPC	=	5.0×10^4
SIGOMEG	=	1.0
TFAC	=	0.075
TIN	=	100.0
VDOTIN	=	1.0×10^5
VIN	=	0.0
WO	=	0.1
XII	=	300.0

Fig. 10.

Energy extraction rate as a function of time with a volumetric water flow rate of $5 \times 10^4 \text{ cm}^3/\text{sec}$.

the placement of the inlet for Fig. 11 and in the variable primary crack width for Fig. 12. Figure 11 shows close similarity at 100 yr of elapsed time with the configuration of Fig. 6. Figure 12, in contrast, illustrates the effects of opening the primary crack from an initial maximum gap width of 0.1 cm to 0.5 cm at 1 yr and thereafter. The flow configurations exhibit a profound alteration, changing from bypass flow to buoyancy-dominated flow in which the inlet cold water falls to the bottom of the primary crack before rising to the inlet. The result after 1 yr is a significant enhancement to the energy extraction rate (Fig. 13).

The REX computer program was written to allow for the efficient calculation of buoyancy-dominated flows without numerical instability. Most of our illustrated examples do not take full advantage of this capability, but investigators using the program may find this feature of the code to be very useful.

We have performed many additional calculations to prooftest the code under a wide variety of circumstances. Some representative results are presented here, demonstrating the attributes of the code and showing the magnitudes of thermal fracture energy extraction enhancement that result from a particular model of rock behavior. The stage is set accordingly for two additional investigations:

- (1) Accurate descriptions of rock response to thermal stress should be derived and included in the calculation.
- (2) Detailed studies of heat and water flow in a lateral-fracture maze should be undertaken to support or modify the theory described in this report. We anticipate pursuing this second course of action, with results to be presented later.

Fig. 11.

Velocity vectors and contours of modified pressure, water temperature, and thermal fracture depth at an elapsed time of 100 yr for the constant-width primary crack.

Fig. 12.

Velocity vectors and contours of modified pressure and water temperature at 1.08×10^5 , 3.53×10^7 , 1.50×10^8 , and 3.13×10^9 sec (100 yr), for the variable width primary crack.

Fig. 12. (continued).

Fig. 13.

Energy extraction rate as a function of time for the constant and variable primary crack widths.

VI. THE COMPUTER PROGRAM

The numerical procedure for solving the equations is implemented by REX. The flow of logic for a computational cycle is shown in Fig. 3. A full listing of the computer code for the calculational example shown in Fig. 12 is given in App. A. Coarse-mesh scoping calculations over 100 yr of elapsed time, as shown in Figs. 4, 6, 11, and 12, typically require 1.5 to 3 min of CDC-7600 CPU time, utilizing 111734 octal words of small core memory (SCM) and large core memory (LCM) storage. A finely zoned calculation, as shown in Fig. 5, requires 9 min of CPU time and results in roughly a 20% increase in accuracy. Here, we describe the input and running procedure.

A. General Structure

REX is a computer code written to operate under the Livermore Time-Sharing System (LTSS) on the CDC 7600. The program is designed with one main subroutine to handle the central flow of the logic and several other subroutines, including those required for output in the form of printing and plotting. The steps in a cycle are as follows.

(1) The first executable statement in REX is a library routine UNQCHG, which changes the name of the drop file to a unique name. In the LTSS operating system, the executing program is rolled out from memory to disk many times before it completes execution. The copy on the disk is called the drop file. Ordinarily, the

drop file name would be the same as the name of the original compiled program, and thus the drop file would occupy the same physical disk space as the original program. As a result, the original file would be overwritten continually with the current image of the executing program. This procedure is satisfactory unless one wishes the drop file to be available for repeated execution with different input data and for use with DBCTRL, the debugging program on LTSS. UNQCHG tells LTSS to drop the executing program to a disk file other than the original one before the first rollout of the program, and causes a duplicate of the original file to be written. This file becomes the executing drop file and is given a unique name change from the original file, consisting of a prefix + and a random A to Z alphabetic suffix.

A call to UNDROP follows. This subroutine is a REX routine, which contains a series of LTSS library system-interface routines (described in Sec. VI.F) that create output and film files required by the code. The program now calls OOTIM, a library routine, which obtains the integer microseconds charged for the run, which are then converted to seconds and printed out at the end of the output.

(2) The program then proceeds to the main subroutine, which contains an initial variable list (described in Sec. VI.C), initial conditions, and automatic finish times calculated before the computational cycle begins.

(3) The computational cycle begins with the calculation of W , DT , and $TBAR$, followed by the initialization for velocity iteration.

(4) The pressures and velocities are solved by iteration. The iteration process is a subcycling within the main cycle. The mass equation discrepancy D is calculated; γ and the new value of ϕ are calculated; and tests are made for convergence. If convergence has been obtained, the flow leaves the iteration loop and continues through the cycle. If convergence has not been obtained, the new velocities are then calculated and the flow returns to the beginning of the velocity iteration loop.

(5) Initialize for temperature iteration.

(6) Temperatures are solved by iteration. SIG and $RDOT$ are calculated. If convergence has been obtained, the flow leaves the iteration loop and continues through the cycle. If convergence has not been obtained, the new temperatures are calculated and the flow returns to the beginning of the temperature iteration loop.

(7) When convergence for both the pressure-velocity loop and the temperature loop have been obtained, the flow of the cycle proceeds to a test for pseudo-

steady state. If pseudo-steady state is reached, the code continues through the cycle. If pseudo-steady state is not reached, flow returns to (4) above and the pressures and velocities and the temperatures subcycle through their respective iterative procedures again.

(8) Upon attainment of pseudo-steady state, the rock configuration is advanced, the energy difference and volume are calculated, and the elapsed time is updated.

(9) The REX print (PRINT) and plotting (PLOTPKG) subroutines are now called. These produce output in the form of microfiche plots and printed information on either microfiche or fanfold paper. PLOTPKG in turn calls the modified LTSS library routines PLOJB, CONTRJB, and TRICJB. TERM is then called, which prints information on the terminal regarding energy, volume, and iteration and subcycling history. A test is made to see if the finish time has been reached.

(10) When the printing and plotting are finished for the last cycle, it is time to terminate the program. This is done by calling the routine DROP, which calls several system interface routines to close job files (Sec. VI.F). The library routine OOTIM is called in order to obtain the integer microseconds charged for the run, which are then converted to seconds and printed out at the end of the output. Subroutine EXIT now terminates the run. To leave a drop file in the user file space, CALL EXIT(n) must be used, where n is a nonzero integer. This form is recommended for debugging. The library routines ALLOUT and GIVE may now be used to dump the output as explained in Sec. IX.E.

B. The Indexing Notation

Figure 14 shows that some variables are located at cell centers and some at cell edges, a typical arrangement for many Eulerian codes. FORTRAN representation of these values is complicated by the half-integer indices. Whereas T_{ij} can be represented as $T(I,J)$, $u_{i+\frac{1}{2}j}$ cannot be represented as $U(I + .5,J)$. Our convention is shown in Fig. 14, such, for example, that $u_{i+\frac{1}{2}j}$ becomes in FORTRAN $U(I,J)$ and $v_{ij+\frac{1}{2}}$ becomes $V(I,J)$.

The computer storage grid must be at least $(\bar{i} + 1)$ by $(\bar{j} + 1)$, as the number of cell edges is one greater than the number of cells in each direction. In our FORTRAN index notation, the letter P is used to represent +, and M represents -. For example,

$$IBP1 \equiv IBAR + 1$$

$$JBM1 \equiv JBAR - 1$$

Fig. 14.

Location of some representative variables in a REX cell.

In addition, it is necessary to surround the computing mesh with an extra set of cells on all four sides, which gives an $(\bar{i} + 2)$ by $(\bar{j} + 2)$ mesh (Fig. 15). These cells are needed to handle the boundary conditions.

C. The Input and Derived Data

The data are input directly from initial variable values at the beginning of the code. These values can be changed easily if new problem conditions are desired. The following list gives the names and a brief description of these variables.

Fig. 15.

FORTRAN computing mesh used in REX. Dashed lines indicate fictitious cells.

NAME	DESCRIPTION
ALAM	λ , relaxation rate factor for κ_o
AKAPO	κ_o , ratio of thermal conductivity in rock to the product of the density of water and specific heat coefficient for water
AKAPR	κ_R , thermometric conductivity of rock
AKR	k_R , coefficient of heat conduction in rock
BETA	β_w , volumetric expansion coefficient for water
BETAR	β_R , linear expansion coefficient for rock
BR	b_R , specific heat of rock
BW	b_w , specific heat of water
DCO	Upper limit for the distance between secondary cracks
DX	δx , the cell size in the horizontal direction
DXR	$1/\delta x$
DXSQ	δx^2
DXSQR	$1/\delta x^2$
DY	δy , the cell size in the vertical direction
DYR	$1/\delta y$
DYSQ	δy^2
DYSQR	$1/\delta y^2$
EIGFIF	8/15
EPSILON	ϵ , the convergence criterion test number for the pressure iteration
FORFIF	4/15
GRAV	g , acceleration of gravity
IBAR	$\bar{i}$, the total number of calculational cells in the horizontal direction
IBP1	$\bar{i} + 1$
IBP2	$\bar{i} + 2$
IIN	i for the inlet cell
IOUT	i for the outlet cell
JBAR	$\bar{j}$, the total number of calculational cells in the vertical direction
JBP1	$\bar{j} + 1$
JBP2	$\bar{j} + 2$
JIN	j for the inlet cell
JOUT	j for the outlet cell
NCYF	Total number of cycles in a calculation
NCYI	Number of cycles between prints

NCYP	Print interval counter
NCYTER	Terminal interval counter
NCYTERI	Number of cycles between terminal messages
ONETHI	1/3
RHOR	ρ_R , density of rock
RHOW	ρ_w , density of water
RMESH	Total number of calculational cells
RPC	Radius of primary crack
RPCSR	$1/RPC^2$
SIGOMEG	σ_0/ω , parameter for specifying the magnitude of σ
SIGCO	Coefficient for the calculation of σ
TFAC	Factor for increment of DTT
TI	t, total elapsed time
TIN	Water temperature at inlet
TWOTHI	2/3
VDOTIN	$\dot{V}_{in}$, volume of water per unit coming in
VIN	Directed component of velocity at inlet
WO	W_0 , primary crack width at center
XII	ξ_0 , rock temperature far from the crack

D. The Common Blocks, VAR and ARRAYS

In addition to the variables contained in the input data (Sec. VI.C), which can be set initially, the SCM common block VAR also contains:

DSUM	Measure of total nonconvergence discrepancy
DT	δt , time increment per hydrodynamic cycle
DTR	$1/\delta t$
DTT	δt , true elapsed time per calculational cycle
EDOTDIF	Difference between energy flow rates at inlet and outlet, converted to megawatts
ESUM	Cumulative amount of energy extracted, converted to megawatt-seconds
NCYCLE	Number of cycles in the problem at a given time
NIT	Number of velocity iterations within a given cycle
NITT	Number of temperature iterations within a given cycle
NSUB	Number of subcycles to reach pseudo-steady state
VDOTOUT	$\dot{V}_{out}$, volume of water per unit time going out
VOL	Total volume of primary and lateral cracks

WMAX Largest width within the crack

The common block, ARRAYS, contains the cell storage information:

AKO(I,J)	Coefficient of heat conduction for rock divided by the product of density and specific heat for water
D(I,J)	D, mass equation discrepancy
DC(I,J)	Distance between secondary cracks
GAM(I,J)	γ , Newton-Raphson relaxation coefficient
HC(I,J)	Heat convection term
PHI(I,J)	ϕ , modified pressure/density
R(I,J)	R, depth of thermal cracks
RDOT(I,J)	$\dot{R}$, penetration speed
SIG(I,J)	σ , transport function for heat through the lateral cracks
SVDC(I,J)	Total rate of change of crack volume
T(I,J)	T, water temperature
TBAR(J)	$\bar{T}$, mean temperature for each level
U(I,J)	u, velocity in horizontal direction
UWBAR(I,J)	$\overline{uW}$, initialization quantity for iteration; also used in temperature equation iterative solution
V(I,J)	v, velocity in vertical direction
VDOT(I,J)	$\dot{V}$, volumetric source or sink
VWBAR(I,J)	$\overline{vW}$, initialization quantity for iteration; also used in temperature equation iterative solution
W(I,J)	W, crack width
WC(I,J)	WC, total volume of lateral cracks

E. Output - Prints, Plots

The code REX provides output in three forms. They are microfiche plots, printed output information on either microfiche or fanfold paper, and printed output at the remote terminal. The plot and print times are controlled by setting NCYI (cycle interval) and NCYTERI (cycle terminal interval) in the initial variable list.

The microfiche plots are the most useful form of output from our fluid dynamics codes because they quickly convey information about a flow process that would be more difficult to grasp by examining only numerical listings. The plot package (subroutine PLOTPKG) allows the user to create four contour plots and one vector plot.

Contour plots are given for the arrays W (primary crack width), T (water temperature), PHI (pressure, as modified by the removal of the hydrostatic part and the normalization of net buoyancy force to zero in the primary crack), and R (depth of thermal fracture).

These contour plots are created by connected vector segments, which join points of equal value. The contour increment is linear. The vector plot is useful because it shows at a glance both the direction of flow and the relative magnitude of the velocities. The vector plot is drawn with each vector originating from a cell center, and the vector length and direction are determined by the x and y velocity components. The vectors are normalized such that the maximum magnitude lies within one cell size. The microfiche plots are created by using the library GIVE program, followed by the REX-created FX10512XXX file.

In addition to the plot capability, REX also provides for printed numerical data by means of subroutine PRINT. The long print is a complete numerical listing of the major field variables in each cell. Two lines containing i and j and 14 of the two-dimensional field variables are given for each cell:

$$i, j, u_{i+\frac{1}{2}j}, v_{ij+\frac{1}{2}}, \phi_{ij}, T_{ij}, W_{ij}, \gamma_{ij}, D_{ij}, \dot{V}_{ij}$$

and

$$i, j, WC_{ij}, \sigma_{ij}, \kappa_{oij}, SVDC_{ij}, \dot{R}_{ij}, R_{ij}, \dot{V}_{ij}.$$

Preceding each long print is a two-line short print containing the time, cycle number, and iteration number. It also contains the difference between the energy outflow and inflow rates (EDOTDIF), and two summation values, DSUM and ESUM. DSUM is the sum of all the D_{ij} values in the mesh. DSUM is used as a check on the calculation; its magnitude should always be very small compared with DEPS, which is calculated each cycle as a function of EPSILON. ESUM is the cumulative amount of energy extracted. In addition, all of the initial input variables are printed between the short and long prints at $TI = 0$.

The printed output can be dumped on microfiche by using the ALLOUT FR80 REXOUT option or on fanfold paper with ALLOUT REXOUT. This output is created automatically at a time $TI = 0$. Thereafter, the plot and print times are controlled by setting NCYI (cycle interval) in the initial variable list.

Printed listings of output for the described runs are available from the authors upon request.

F. System Library Subroutine CALLs

REX uses several system subroutines to handle the microfiche plotting. These routines are entirely system-dependent and will be described here only briefly. They make use of the SC-4020, a microfilm-recording CRT device, which either plots or prints on a 1024 by 1024 grid on the CRT face. These subroutines are accessed by the following FORTRAN calling sequences.

CALL ADV (nf) advances the film by nf frames.

CALL FRAME (IXL, IXR, IYB, IYT) draws a rectangular outline of the computing mesh. Two horizontal axes are drawn through IYT and IYB from IXL to IXR, and two vertical axes are drawn through IXL to IXR from IYT to IYB.

CALL PLT (IX, IY, ch) plots the 4020 character identified by ch at 4020 frame coordinates (IX, IY).

CALL DRV (IX1, IY1, IX2, IY2) draws a straight line vector segment connecting the 4020 point (IX1, IY1) with the 4020 point (IX2, IY2).

CALL CONVRT (Z, IZ, Z1, Z2, IZ1, IZ2) converts the floating point number Z to a 4020 coordinate based on Z1 and Z2.

CALL DLCV (IX, IY, NC, BCD, ISIZE) draw large characters vertical.

CALL DLCH (IX, IY, NC, BDC, ISIZE) draw large characters horizontal.

CALL DGA (IXL, IXR, IYB, IYT, XL, XR, YT, YB) defines the graph area.

In addition, REX subroutine PLOTPKG uses three modified system subroutines for the actual microfiche plotting, which are part of the code (Subsec. A.9). These subroutines are:

CALL CONTRJB produces a contour plot on film with labels by determining minimum and maximum contours and spacing.

CALL TRICJB performs the actual interpolation needed to produce contours as defined in CONTRJB.

CALL PLOJB creates a grid with tic marks and labels axes, writes titles, draws borders, and connects contour points established in TRICJB.

In addition to the CALLs directly related to the microfilm plots described above, REX also calls various other library subroutines for interfacing with the LTSS system.

CALL UNQCHG changes the name of a drop file to a unique name (Subsec. A.1).

CALL FILM80 sets up system linkage to two routines for processing formatted WRITES and job termination, and sets common storage for the film routines. FILM80 has no arguments.

CALL IDENT80 (ifsn, buf, nwbuf, nwdisk, icam, ifilm) identifies an integer file number ifsn as an FR80 film file and supplies other necessary file variables: buf is the name of the LCM buffer for film data storage, nwbuf is the number of words in this buffer, nwdisk is the number of words reserved in the disk buffer, icam is the camera to be used to process the film where the available options are 16 (16 mm), 0 or 35 (35 mm), and 105 (105 mm microfiche), and ifilm is the film type where 0 is black and white and 1RC is color.

CALL HEAD80 (ifsn, nc, char, nbox) puts out header information for the specified film file number ifsn (previously identified by IDENT80), nc is the number of characters (60 maximum), char specifies the characters, and nbox is the film user identification of the form 10HBOXaaaaaaaa.

CALL KEEPFLM (ifilm) indicates that the file ifilm is not to be given to the system. It may be called anytime before the release of film file information. Its purpose in REX is to retain a copy of the file available for scanning under LTSS at a Tektronix terminal.

CALL EXIT terminates the code. The user should be aware that a drop file will remain in user file space only if a CALL EXIT(n) is used to terminate the code where n is a nonzero integer. This latter form is recommended for use in debugging.

G. Operating the Terminal

The REX code is run from the remote terminal using the LTSS operating system. The program is currently resident on CFS and can be accessed upon request. There is a LIX file containing the source code (REXSO), the load module (REXLO), and utility program (UTIL) for ease in compiling and running. The compile file REXLS, binary file (REXBI), map (REXMA), symbol table (REXSY), and executable file (REX) are created at compile and load time and the output (REXOUT) and plot (FX10512xxxx) files are available after running.

After getting the code from CFS with MASS, it is necessary to use LIX to establish the individual files in the working space. Perform a GR ALL under LIX, and the files will be available for individual use. Next, in TRIx, read the utilities (RD!UTIL!). Then, using the CHAT compiler and these utilities, compile in the following sequence

compl(REX)

and after the prompt,

load(REX).

The program may also be run without the utility program by using conventional CHAT compile and load lines.

After leaving TRIX, the program may then be run by simply typing REX. The current time in seconds, the energy extraction rate in megawatts, the total volume of the lateral cracks in the rock in cubic centimeters, and the cumulative amount of the extracted energy in megawatt-seconds will be typed on the terminal for the designated cycles as the program runs. The terminal will also type the number of velocity iterations, the number of temperature iterations, the number of subcycles, and the cycle number. After the program finishes running, the output and plots may be dumped using ALLOUT and GIVE as explained in Sec. IV.E.

The REX program has also been converted to compile and load using the FTN compiler. This can be accomplished by typing the following line.
FTN (I=REX,LOAD,GLIB=CGSFTN). In this case, the files created will be the compile file (LISTFTN), symbol table (SYMFTN), and the executed file (REXX). After REXX is run, the output (TAPE6) and film plot (FR80F) files will be available. Output on fanfold paper can be created by using ALLOUT TAPE6, or on microfiche by using ALLOUT FR80 TAPE6. The plots will become available on microfiche with the command CGSGIVE I=FR80F.

VII. REFERENCE

1. Ruth B. Demuth and Francis H. Harlow, "Thermal Fracture Effects in Geothermal Energy Extraction," Los Alamos Scientific Laboratory report LA-7963 (November 1979).

APPENDIX A

REX COMPUTER CODE LISTING

LASL Identification No. LP-2086

```

1*****
000001 000001 C
000002 000002 C
000003 000003 C
000004 000004 C
000005 000005 C
000006 000006 C
000007 000007 C
000008 000008 C
000009 000009 C
000010 000010 C
000011 000011 C
000012 000012 C
000013 000013 C
000014 000014 C
000015 000015 C
000016 000016 C
000017 000017 C
000018 000018 C
000019 000019 C
000020 000020 C
000021 000021 C
000022 000022 C
000023 000023 C
000024 000024 C
000025 000025 C
000026 000026 C
000027 000027 C
000028 000028 C
000029 000029 C
000030 000030 C
000031 000031 C
000032 000032 C
000033 000033 C
000034 000034 C
000035 000035 C
000036 000036 C
000037 000037 C
000038 000038 C
000039 000039 C
000040 000040 C
000041 000041 C
000042 000042 C
000043 000043 C
000044 000044 C
000045 000045 C
000046 000046 C
000047 000047 C
000048 000048 C
000049 000049 C
000050 000050 C
000051 000051 C
000052 000052 C
000053 000053 C
000054 000054 C
000055 000055 C
000056 000056 C

CHAT 1670 10:02:50 004/04/80

VARIABLE DEFINITIONS

AKAPR - KAPPA ROCK, TERMOMETRIC CONDUCTIVITY OF ROCK
AKAPO - RATIO OF THERMAL CONDUCTIVITY IN ROCK TO THE
        PRODUCT OF THE DENSITY OF WATER AND VOLUMETRIC
        EXPANSION COEFFICIENT FOR WATER
AKR - HEAT CONDUCTIVITY
ALAM - RELAXATION RATE FACTOR FOR AKAPO
BETA - BETA WATER- VOLUMETRIC EXPANSION COEFFICIENT FOR WATER
DC - DISTANCE BETWEEN SECONDARY CRACKS
DT - DELTA T, CHANGE IN THE ELAPSED TIME
DTT - DELTA T, TRUE
DX - CHANGE IN THE DISTANCE IN X DIRECTION
DY - CHANGE IN THE DISTANCE IN Y DIRECTION
GAM - NEWTON RAPHSON RELAXATION COEFFICIENT
GRAV - ACCELERATION OF GRAVITY
HC - HEAT CONVECTION TERMS
IBAR - RIGHT BOUNDARY OF TOTAL MESH IN X DIRECTION
JBAR - TOP BOUNDARY OF MESH IN Y DIRECTION
NCYF - TOTAL CALCULATION TIME, BASED ON CYCLE
NCYP - TIME TO PRINT, BASED ON CYCLE
NCYI - TIME PRINT INTERVAL, BASED ON CYCLE
PHI - PRESSURE/ DENSITY
R - DEPTH OF THERMAL CRACKS
RDOT - PENETRATION SPEED
RPC - RADIUS OF PRIMARY CRACK
SIG - TRANSPORT FUNCTION FOR HEAT THROUGH THE LATERAL CRACKS
SIGMEG - COEFFICIENT FOR THE CALCULATION OF SIG
SVDC - VOLUME CHANGE RATE
T - WATER TEMPERATURE
TFAC - FACTOR FOR INCREMENT OF DTT
TI - RUNNING PROBLEM TIME
TIN - WATER TEMPERATURE AT INLET
U - VELOCITY IN X DIRECTION
V - VELOCITY IN Y DIRECTION
VDDTIN - VOLUME PER UNIT TIME COMING IN
VDDTOUT - VOLUME PER UNIT TIME GOING OUT
VIN - VERTICAL COMPONENT OF VELOCITY AT THE INLET
W - PRIMARY CRACK WIDTH
WC - TOTAL VOLUME OF LATERAL CRACKS
WO - PRIMARY CRACK WIDTH AT CENTER
XII - ROCK TEMPERATURE AT INFINITY

PROGRAM REX(OUTPUT,REXOUT,TAPE6=REXOUT,TAPE63=100B)

PARAMETER (NI=25)
PARAMETER (NJ=25)
COMMON/ARRAYS/ AKO(NI,NJ),D(NI,NJ),DC(NI,NJ),GAM(NI,NJ),
1 HC(NI,NJ),PHI(NI,NJ),R(NI,NJ),RDOT(NI,NJ),
2 SIG(NI,NJ),SVDC(NI,NJ),T(NI,NJ),TBAR(NJ),
3 U(NI,NJ),UWBAR(NI,NJ),V(NI,NJ),VDDT(NI,NJ),

```

```

000057 000057      4      VWBAR(NI,NJ),W(NI,NJ),WC(NI,NJ)
000058 000058      C
000059 000059      C
000060 000060      COMMON/VAR/AKAPO,AKAPR,AKR,ALAM,BR,BW,BETA,BETAR,DCO,DSUM,
000061 000061      1      DT,DTT,DX,DY,DXR,DYR,DXDYR,DXSQ,DYSQ,DXSQR,
000062 000062      2      DYSQR,EIGFIF,EPSILON,GRAV,IBAR,IBP1,IBP2,IIN,
000063 000063      3      IOUT,JBAR,JBP1,JBP2,JIN,JOUT,NCYCLE,NCYF,
000064 000064      4      NCYI,NCYP,NCYTER,NCYTERI,NIT,NITT,RHOR,RHOW,
000065 000065      5      RPC,RPCSQ,SIGCO,SIGOMEG,TFAC,TI,TIN,TWOTH,VDOTIN,
000066 000066      6      VDOTOUT,VIN,VOL,WG,WMAX,XII,NSUB,EDDTDIF,ESUM
000067 000067      C
000068 000068      C
000069 000069      C
000070 000070      CALL UNQCHG
000071 000071      CALL UNDROP
000072 000072      C
000073 000073      C
000074 000074      C TIMING STUDIES
000075 000075      C
000076 000076      CALL DDTIM
000077 000077      ISEC = 0
000078 000078      C
000079 000079      C
000080 000080      C
000081 000081      C VARIABLES
000082 000082      C
000083 000083      ALAM = 0.1
000084 000084      BW = 1.0
000085 000085      BR = 0.25
000086 000086      RHOW = 1.0
000087 000087      RHOR = 2.65
000088 000088      AKR = 6.2E-3
000089 000089      AKAPO = AKR / (RHOW*BW)
000090 000090      AKAPR = AKR / (RHOR*BR)
000091 000091      BETA = 2.07E-4
000092 000092      BETAR = 8.3E-6
000093 000093      DCO = 25.0
000094 000094      DX = 1.E4
000095 000095      DY = 1.E4
000096 000096      DXR = 1./DX
000097 000097      DYR = 1./DY
000098 000098      DXDYR = 1./(DX*DY)
000099 000099      DXSQ = DX**2
000100 000100      DYSQ = DY**2
000101 000101      DXSQR = 1./DXSQ
000102 000102      DYSQR = 1./DYSQ
000103 000103      EIGFIF = 8./15.
000104 000104      FORFIF = 4./15.
000105 000105      EPSILON = 2.E-3
000106 000106      GRAV = 990.0
000107 000107      IBAR = 11
000108 000108      IBP1 = IBAR+1
000109 000109      IBP2 = IBAR+2
000110 000110      IIN = 8
000111 000111      IOUT = 6
000112 000112      JBAR = 11
000113 000113      JBP1 = JBAR+1

```

```

000114 000114 JBP2 = JBAR+2
000115 000115 JIN = 7
000116 000116 JDUT = 8
000117 000117 NCYF = 0
000118 000118 NCYI = 20
000119 000119 NCYP = 0
000120 000120 NCYTER = 0
000121 000121 NCYTERI = 10
000122 000122 RMESH = FLOAT(IBAR)*FLOAT(JBAR)
000123 000123 RPC = 5.E4
000124 000124 RPCSR = 1./RPC**2
000125 000125 SIGOMEG = 1.0
000126 000126 SIGOM = SIGOMEG*BETA*GRAV*BETAR**2 / 71.424
000127 000127 C NOTE- 71.424 = 8 TIMES 2.232 (2.232 IS 8 TIMES THE NUMERATOR IN NU)
000128 000128 TFAC = 0.075
000129 000129 TI = 0.0
000130 000130 TIN = 100.0
000131 000131 CNETHI = 1./3.
000132 000132 TWOTH I = 2./3.
000133 000133 VDOTIN = 1.E5
000134 000134 VIN = 0.0
000135 000135 W0 = 0.1
000136 000136 XII = 300.0
000137 000137 C
000138 000138 C
000139 000139 C INITIAL CONDITIONS
000140 000140 C
000141 000141 DO 199 I = 1,IBP1
000142 000142 DO 198 J = 1,JBP2
000143 000143 U(I,J) = 0.0
000144 000144 198 CONTINUE
000145 000145 199 CONTINUE
000146 000146 C
000147 000147 DO 299 I = 1,IBP2
000148 000148 DO 298 J = 1,JBP1
000149 000149 V(I,J) = 0.0
000150 000150 298 CONTINUE
000151 000151 299 CONTINUE
000152 000152 C
000153 000153 C CALCULATE INDEX FOR THE DOWN SIDE OF MIDDLE
000154 000154 C
000155 000155 HJBAR = FLOAT(JBAR)*.5 + 2.01
000156 000156 JM = HJBAR
000157 000157 C
000158 000158 C
000159 000159 HIBAR = FLOAT(IBAR)*.5 + 2.01
000160 000160 IM = HIBAR
000161 000161 C
000162 000162 C
000163 000163 C CALCULATE W
000164 000164 C
000165 000165 DO 599 I = 1,IBP2
000166 000166 DO 598 J = 1,JBP2
000167 000167 W(I,J) = W0*(1.0-(DXSQ*(FLOAT(I-IM))**2
000168 000168 1 + DYSQ*(FLOAT(J-JM))**2)*RPCSR)
000169 000169 IF(W(I,J).LT.0.0) W(I,J) = 0.0
000170 000170 598 CONTINUE

```

```

000171 000171 599 CONTINUE
000172 000172 C
000173 000173 C
000174 000174 C
000175 000175 DO 359 I = 1,IBP2
000176 000176 DO 359 J = 1,JBP2
000177 000177 T(I,J) = XII
000178 000178 AKO(I,J) = 0.0
000179 000179 R(I,J) = 2.0*DX*DY*AKAPR / (XII-TIN) / VDOTIN
000180 000180 DC(I,J) = R(I,J)
000181 000181 PHI(I,J) = 0.0
000182 000182 WC(I,J) = 0.0
000183 000183 358 CONTINUE
000184 000184 359 CONTINUE
000185 000185 C
000186 000186 C CALCULATE VOLUMETRIC INLET
000187 000187 C
000188 000188 VDOT(IIN,JIN) = VDOTIN
000189 000189 C
000190 000190 C
000191 000191 C INITIALIZE VOLUME OF PRIMARY CRACK
000192 000192 C
000193 000193 VOL = 0.0
000194 000194 DO 379 I = 2,IBP1
000195 000195 DO 378 J = 2,JBP1
000196 000196 VOL = VOL + W(I,J)*DX*DY
000197 000197 378 CONTINUE
000198 000198 379 CONTINUE
000199 000199 C
000200 000200 C
000201 000201 C
000202 000202 C CALCULATE DT AND AUTOMATIC FINISH CYCLE NUMBER
000203 000203 C
000204 000204 UMAX = VDOTIN / (2.0*(DX+DY)*W0)
000205 000205 DT = .1 * (DX + DY) / UMAX
000206 000206 DEPS = W0*UMAX*EPSILON*DXR
000207 000207 DTR = 1./DT
000208 000208 C
000209 000209 DTO = .01*W0*RPC*RPC / VDOTIN
000210 000210 IF(NCYF.NE.0) GO TO 400
000211 000211 C
000212 000212 C NUMBER OF SECONDS IN 100 YEARS IS 3.15576E9
000213 000213 C
000214 000214 ANCYF = ALOG(TFAC*3.15576E9/DTO+1.) / ALOG(1.+TFAC)
000215 000215 NCYF = INT(ANCYF)
000216 000216 400 CONTINUE
000217 000217 C
000218 000218 C
000219 000219 VDOTOUT = -VDOTIN
000220 000220 VDOT(IOUT,JOUT) = -VDOTIN
000221 000221 C
000222 000222 C
000223 000223 C SET CYCLE COUNTERS
000224 000224 C
000225 000225 NCYCLE = 0
000226 000226 C
000227 000227 C*****

```

```

000228 000228 C*****BEGIN CYCLE *****
000229 000229 C*****
000230 000230 C
000231 000231 C
000232 000232 C SUB-CYCLING TO PSEUDO STEADY STATE OF VELOCITIES AND
000233 000233 C TEMPERATURES IN EQUILIBRIUM WITH A SUCCESSION OF
000234 000234 C WALL-CONDITION STATES, THESE LATTER PROCEEDING WITH THE
000235 000235 C ACCOUNTING OF TRUE ELAPSED TIME.
000236 000236 C IN REX, THE ADVANTAGE OF IMPLICIT CONVECTION COULD BE
000237 000237 C THE USE OF LARGER VALUES OF OUR SUB-CYCLING DT.
000238 000238 C
000239 000239 900 CONTINUE
000240 000240 NSUB = 0
000241 000241 C
000242 000242 C
000243 000243 C CALCULATE NEW VALUES OF WOV
000244 000244 C
000245 000245 WOV=0.5
000246 000246 IF(TI.GT.3.16E8) GO TO 950
000247 000247 WOV = W0 + 0.4*(TI / 3.16E8)
000248 000248 950 CONTINUE
000249 000249 C
000250 000250 C
000251 000251 C CALCULATE W
000252 000252 C
000253 000253 DO 999 I = 1,IBP2
000254 000254 DO 998 J = 1,JBP2
000255 000255 W(I,J) = WOV*(1.0-(DXSQ*(FLOAT(I-IM))**2
000256 000256 1 + DYSQ*(FLOAT(J-JM))**2)*RPCSR)
000257 000257 IF(W(I,J).LT.0.0) W(I,J) = 0.0
000258 000258 998 CONTINUE
000259 000259 999 CONTINUE
000260 000260 C
000261 000261 C
000262 000262 C
000263 000263 1300 CONTINUE
000264 000264 C
000265 000265 C
000266 000266 C
000267 000267 C TEST VALUE FOR CALCULATING DT
000268 000268 C
000269 000269 JBOT = JIN-1
000270 000270 VELTEST = ABS(V(IIN,JBOT))
000271 000271 VELTOP = ABS(V(IIN,JIN))
000272 000272 VELTEST = AMAX1(VELTOP,VELTEST)
000273 000273 IF(VELTEST.LT.UMAX) GO TO 1400
000274 000274 DT = 0.3*(DX+DY) / VELTEST
000275 000275 DEPS = W(IIN,JIN)*VELTEST*EPSILON*DXR
000276 000276 DTR = 1./DT
000277 000277 GO TO 1500
000278 000278 1400 CONTINUE
000279 000279 DT = 0.1*(DX+DY) / UMAX
000280 000280 DEPS = W(IIN,JIN)*UMAX*EPSILON*DXR
000281 000281 DTR = 1./DT
000282 000282 1500 CONTINUE
000283 000283 C
000284 000284 C

```



```

000285 000285 C CALCULATE TBAR AND INITIALIZE FOR ITERATION ON VELOCITY
000286 000286 C
000287 000287 C
000288 000288 DO 2099 J = 2,JBAR
000289 000289 ANSUM = 0.0
000290 000290 TBARSUM = 0.0
000291 000291 DO 2098 I = 2,IBP1
000292 000292 IF(W(I,J).EQ.0.0) GO TO 2050
000293 000293 IF(W(I,J+1).EQ.0.0) GO TO 2050
000294 000294 TBARSUM = TBARSUM + .5*(T(I,J) + T(I,J+1))
000295 000295 ANSUM = ANSUM + 1.0
000296 000296 2050 CONTINUE
000297 000297 2098 CONTINUE
000298 000298 TBAR(J) = TBARSUM / (ANSUM + 10.E-10)
000299 000299 2099 CONTINUE
000300 000300 C
000301 000301 C
000302 000302 DO 2299 I = 2,IBP1
000303 000303 DO 2298 J = 2,JBAR
000304 000304 VWBAR(I,J) = V(I,J)*.5*(W(I,J)+W(I,J+1))
000305 000305 1 + .5*(W(I,J)+W(I,J+1))*BETA*GRAV*DT
000306 000306 2 *(.5*(T(I,J) + T(I,J+1)) - TBAR(J))
000307 000307 2298 CONTINUE
000308 000308 2299 CONTINUE
000309 000309 C
000310 000310 C
000311 000311 C ***** BEGIN ITERATION OF VELOCITIES *****
000312 000312 C
000313 000313 C
000314 000314 C CALCULATE D (MASS EQUATION)
000315 000315 C
000316 000316 NIT = 0
000317 000317 2500 CONTINUE
000318 000318 DTEST = 0.0
000319 000319 DSUM = 0.0
000320 000320 DO 2599 I = 2,IBP1
000321 000321 DO 2598 J = 2,JBP1
000322 000322 IF(W(I,J).LT.(1.E-2*DEPS)) GO TO 2570
000323 000323 WFACA = AMIN1(W(I,J),W(I+1,J))
000324 000324 WFACB = AMIN1(W(I-1,J),W(I,J))
000325 000325 WFACC = AMIN1(W(I,J),W(I,J+1))
000326 000326 WFACD = AMIN1(W(I,J),W(I,J-1))
000327 000327 C
000328 000328 D(I,J) = TWOTH I*((WFACA*U(I,J)-WFACB*U(I-1,J))
000329 000329 1 *DXR + (WFACC*V(I,J) - WFACD*V(I,J-1))*DYR)
000330 000330 2 - DXDYR*(VDOT(I,J)-SVDC(I,J))
000331 000331 ARSU = .5*ABS(U(I-1,J) + U(I,J))
000332 000332 ARSU = AMIN1(ARSU,UMAX)
000333 000333 ARSV = .5*ABS(V(I,J-1) + V(I,J))
000334 000334 ARSV = AMIN1(ARSV,UMAX)
000335 000335 EIGNU = 2.232 / (T(I,J)-3.8)
000336 000336 GAMDEN = W(I,J)**3*DT*(DXSQR + DYSQR)
000337 000337 GF = W(I,J)*W(I,J)*(1.+EIGFIF*DT*(ARSU*DXR+ARSV*DYR))
000338 000338 1 + EIGNU*DT
000339 000339 GAM(I,J) = 0.2*GF / GAMDEN
000340 000340 DABS = ABS(D(I,J))
000341 000341 DTEST = AMAX1(DABS,DTEST)

```

```

000342      DSUM = DSUM + D(I,J)
000343
000344      C CALCULATE PHI
000345      C
000346      PHI(I,J) = PHI(I,J) - GAM(I,J)*D(I,J)
000347      2570      CONTINUE
000348      2598      CONTINUE
000349      2599      CONTINUE
000350      C
000351      C
000352      C TEST FOR CONVERGENCE OF VELOCITIES
000353      C
000354      IF(NIT.EQ.0) GO TO 2650
000355      IF(DTEST.LT.DEPS) GO TO 3000
000356      IF(NIT.GE.1000) GO TO 3000
000357      2650      CONTINUE
000358      C
000359      C NO CONVERGENCE
000360      C
000361      C
000362      C CALCULATE U AND V
000363      C
000364      DO 2699 I = 2,IBAR
000365      DO 2699 J = 2,JBP1
000366      UC = .5*(U(I-1,J) + U(I,J))
000367      CCC = UC*W(I,J)
000368      IF(UC.GE.0.) CC = -1.0
000369      IF(UC.LT.0.) CC = 1.0
000370      UC = .5*(U(I,J) + U(I+1,J))
000371      DDD = UC*W(I+1,J)
000372      IF(UC.GE.0.) DD = -1.0
000373      IF(UC.LT.0.) DD = 1.0
000374      VC = .5*(V(I,J-1) + V(I+1,J-1))
000375      WW = .5*AMIN1(W(I,J-1)+W(I+1,J-1) , W(I,J)+W(I+1,J))
000376      GGG = WW*VC
000377      IF(VC.GE.0.) GG = -1.0
000378      IF(VC.LT.0.) GG = 1.0
000379      VC = .5*V(I,J) + V(I+1,J)
000380      WW = .5*AMIN1(W(I,J)+W(I+1,J) , W(I,J+1)+W(I+1,J+1))
000381      HHH = VC*WW
000382      IF(VC.GE.0.) HH = -1.0
000383      IF(VC.LT.0.) HH = 1.0
000384      C
000385      UWB = U(I,J)*.5*(W(I,J) + W(I+1,J))
000386      WPRIU = AMIN1(W(I,J),W(I+1,J))
000387      EIGNUU = 2.232*DT / (.5*(T(I,J) + T(I+1,J)) - 3.8)
000388      C
000389      UNUM = WPRIU*(UWB + (WPRIU*DT*DXR)
000390      1      *(PHI(I,J) - PHI(I+1,J)) + (FORFIF*DT*DXR)
000391      2      *(CCC*(1.0-CC)*U(I-1,J) - DDD*(1.0 + DD)*U(I+1,J))
000392      3      + (FORFIF*DT*DYS)*(GGG*(1.0-GG)*U(I,J-1)
000393      4      - HHH*(1.0+HH)*U(I,J+1)))
000394      C
000395      UDEN = WPRIU*(.5*(W(I,J) + W(I+1,J))) + EIGNUU
000396      1      - (WPRIU*FORFIF*DT*DXR)*(CCC*(1.0+CC)
000397      2      - DDD*(1.0-DD)) - (WPRIU*FORFIF*DT*DYS)
000398      3      *(GGG*(1.0+GG) - HHH*(1.0-HH))

```

```

000399 000399 C
000400 000400 C
000401 000401      UWBAR(I,J) = UNUM / UDEN
000402 000402 C
000403 000403 2698 CONTINUE
000404 000404 2699 CONTINUE
000405 000405 C
000406 000406      DO 2799 I = 2,IBP1
000407 000407      DO 2799 J = 2,JBAR
000408 000408          VC = .5*(V(I,J-1) + V(I,J))
000409 000409          AAA = W(I,J)*VC
000410 000410          IF(VC.GE.0.) AA = -1.0
000411 000411          IF(VC.LT.0.) AA = 1.0
000412 000412          VC = .5*(V(I,J) + V(I,J+1))
000413 000413          BBB = W(I,J+1)*VC
000414 000414          IF(VC.GE.0.) BB = -1.0
000415 000415          IF(VC.LT.0.) BB = 1.0
000416 000416          UC = .5*(U(I-1,J) + U(I-1,J+1))
000417 000417          WW = .5*AMIN1(W(I,J)+W(I,J+1) , W(I-1,J)+W(I-1,J+1))
000418 000418          EEE = UC*WW
000419 000419          IF(UC.GE.0.) EE = -1.0
000420 000420          IF(UC.LT.0.) EE = 1.0
000421 000421          UC = .5*(U(I,J) + U(I,J+1))
000422 000422          WW = .5*AMIN1(W(I,J)+W(I,J+1) , W(I+1,J)+W(I+1,J+1))
000423 000423          FFF = WW*UC
000424 000424          IF(UC.GE.0.) FF = -1.0
000425 000425          IF(UC.LT.0.) FF = 1.0
000426 000426 C
000427 000427      WPRIV = AMIN1(W(I,J),W(I,J+1))
000428 000428      EIGNUV = 2.232*DT / (.5*(T(I,J) + T(I,J+1)) - 3.6)
000429 000429 C
000430 000430      VNUM = WPRIV*(VWBAR(I,J) + (WPRIV*DT*DYP)*(PHI(I,J) -
000431 000431      1 PHI(I,J+1)) + (FORFIF*DT*DYP)*(AAA*(1.0-AA)*V(I,J-1)
000432 000432      2 - BBB*(1.0+BB)*V(I,J+1)) + (FORFIF*DT*DIXR)
000433 000433      3 *(EEE*(1.0-EE)*V(I-1,J) - FFF*(1.0+FF)*V(I+1,J)))
000434 000434 C
000435 000435      VDEN = WPRIV*(.5*(W(I,J)+W(I,J+1))) + EIGNUV
000436 000436      1 - (WPRIV*FORFIF*DT*DYP)*(AAA*(1.0+AA) - BBB*(1.0-BB))
000437 000437      2 - (WPRIV*FORFIF*DT*DIXR)*(EEE*(1.0+EE) - FFF*(1.0-FF))
000438 000438 C
000439 000439      HC(I,J) = VNUM / VDEN
000440 000440 2798 CONTINUE
000441 000441 2799 CONTINUE
000442 000442 C
000443 000443 C STORE U AND V INTO PERMANENT MATRICES
000444 000444 C
000445 000445      DO 2899 I = 2,IBAR
000446 000446      DO 2899 J = 2,JBP1
000447 000447          U(I,J) = UWBAR(I,J)
000448 000448 2898 CONTINUE
000449 000449 2899 CONTINUE
000450 000450 C
000451 000451      DO 2999 I = 2,IBP1
000452 000452      DO 2999 J = 2,JBAR
000453 000453          V(I,J) = VC(I,J)
000454 000454 2998 CONTINUE
000455 000455 2999 CONTINUE

```

```

000456 000456 C
000457 000457 C
000458 000458 NIT = NIT + 1
000459 000459 GO TO 2500
000460 000460 C
000461 000461 C ***** END ITERATION OF VELOCITIES *****
000462 000462 C
000463 000463 C
000464 000464 C CONVERGENCE ATTAINED, CONTINUE CYCLE
000465 000465 C
000466 000465 3000 CONTINUE
000467 000467 C
000468 000468 C
000469 000469 C
000470 000470 C NOTE - VWBAR(I,J) IS CURRENT ITERATION TEMPERATURE
000471 000471 C UWBAR(I,J) IS NEWTON-RAPHSON TEST FUNCTION
000472 000472 C HC(I,J) IS NEWTON-RAPHSON ITERATION FACTOR
000473 000473 C
000474 000474 C
000475 000475 C INITIALIZE T-TILDE
000476 000475 C
000477 000477 TTS = 0.0
000478 000478 DO 3099 I = 2,IBP1
000479 000479 DO 3093 J = 2,JBP1
000480 000480 VWBAR(I,J) = T(I,J)
000481 000481 TTS = TTS + T(I,J)*W(I,J) / (W(I,J) + 1.E-6)
000482 000482 3098 CONTINUE
000483 000483 3099 CONTINUE
000484 000484 C
000485 000485 C
000486 000486 DTT = DTD + TFAC*TI
000487 000487 C
000488 000488 NITT = 0
000489 000489 COT = 0.0
000490 000490 4000 CONTINUE
000491 000491 C
000492 000492 C
000493 000493 C ***** BEGIN ITERATION ON TEMPERATURES *****
000494 000494 C
000495 000495 C
000496 000496 TTEST = 0.0
000497 000497 DO 4099 I = 2,IBP1
000498 000498 DO 4093 J = 2,JBP1
000499 000499 AKPD = (AKD(I,J)+AKAPD*ALAM*(XII-VWBAR(I,J)))
000500 000500 / (1.0+ALAM*(XII-VWBAR(I,J)))
000501 000501 SIGS = SIGCD*(DC(I,J))**4 / R(I,J)
000502 000502 SIG(I,J) = SIGS*(VWBAR(I,J)+XII-7.6)*(XII-VWBAR(I,J))**3
000503 000503 RDT(I,J) = RHQW*BW / (RHDR*BR*0.5+R(I,J))*(AKPD+SIG(I,J))
000504 000504 IF(COT.GT.0.5) GO TO 4200
000505 000505 C
000506 000506 C
000507 000507 VDDTT = VDDT(I,J)
000508 000508 IF(VDDTT.GT.0.0) TE = TIN
000509 000509 IF(VDDTT.LE.0.0) TE = VWBAR(I,J)
000510 000510 C
000511 000511 C
000512 000512 WFACA = AMIN1(W(I,J),W(I-1,J))

```

```

000513      000513      UC = U(I-1,J)
000514      000514      IF(UC.GE.0.0) AC = WFACA*UC*VWBAR(I-1,J)
000515      000515      IF(UC.LT.0.0) AC = WFACA*UC*VWBAR(I,J)
000516      000516      WFACB = AMIN1(W(I,J),W(I+1,J))
000517      000517      UC = U(I,J)
000518      000518      IF(UC.GE.0.0) BC = WFACB*UC*VWBAR(I,J)
000519      000519      IF(UC.LT.0.0) BC = WFACB*UC*VWBAR(I+1,J)
000520      000520      WFACC = AMIN1(W(I,J),W(I,J-1))
000521      000521      VC = V(I,J-1)
000522      000522      IF(VC.GE.0.0) CC = WFACC*VC*VWBAR(I,J-1)
000523      000523      IF(VC.LT.0.0) CC = WFACC*VC*VWBAR(I,J)
000524      000524      WFACD = AMIN1(W(I,J),W(I,J+1))
000525      000525      VC = V(I,J)
000526      000526      IF(VC.GE.0.0) DIC = WFACD*VC*VWBAR(I,J)
000527      000527      IF(VC.LT.0.0) DIC = WFACD*VC*VWBAR(I,J+1)
000528      000528      C
000529      000529      C
000530      000530      UWBAR(I,J) = W(I,J)*((VWBAR(I,J) - T(I,J))*DTR)
000531      000531      1      - VWBAR(I,J)*D(I,J) - DXDYR*VDDT(I,J)*TE
000532      000532      2      - TWOTHI*((AC - BC)*DXR + (CC - DIC)*DYR)
000533      000533      3      - RHOX*BR / (RHOX*BW*DTT)*((XII-VWBAR(I,J))
000534      000534      4      *(R(I,J) + RDOT(I,J)*DTT) - WC(I,J)
000535      000535      5      / (RETAR*DX*DY))
000536      000536      6      + DXDYR*VWBAR(I,J)*SVDC(I,J)
000537      000537      C
000538      000538      C
000539      000539      AAC = U(I-1,J)*WFACA
000540      000540      IF(U(I-1,J).GT.0.0) AAC = 0.0
000541      000541      BBC = U(I,J)*WFACB
000542      000542      IF(U(I,J).LT.0.0) BBC = 0.0
000543      000543      CCC = V(I,J-1)*WFACC
000544      000544      IF(V(I,J-1).GT.0.0) CCC = 0.0
000545      000545      DDC = V(I,J)*WFACD
000546      000546      IF(V(I,J).LT.0.0) DDC = 0.0
000547      000547      VTERM = 0.0
000548      000548      IF(VDDT.LT.0.0) VTERM = VDDT(I,J)*DXDYR
000549      000549      C
000550      000550      C
000551      000551      HC(I,J) = 4.0*SIGS*(XII-VWBAR(I,J))*3
000552      000552      1      *(XII+2.0*VWBAR(I,J)-11.4) / R(I,J)
000553      000553      2      + W(I,J)*DTR - D(I,J) - VTERM
000554      000554      3      - TWOTHI*((AAC - BBC)*DXR + (CCC-DDC)*DYR)
000555      000555      4      + RHOX*BR / (RHOX*BW*DTT)*(R(I,J)+RDOT(I,J)*DTT)
000556      000556      5      + DXDYR*SVDC(I,J)
000557      000557      C
000558      000558      C
000559      000559      TTEST = AMAX1(TTEST,ABS(UWBAR(I,J)))
000560      000560      C
000561      000561      4098      CONTINUE
000562      000562      4099      CONTINUE
000563      000563      C
000564      000564      C
000565      000565      C CALCULATE T-TLDE (VWBAR IN CODE)
000566      000566      C
000567      000567      DO 4199 I = 2,IBP1
000568      000568      DO 4198 J = 2,JBP1
000569      000569      IF(HC(I,J).EQ.0.0) GO TO 4150

```

```

000570      VWBAR(I,J) = VWBAR(I,J) - (UWBAR(I,J) / HC(I,J))
000571      4150 = CONTINUE
000572      4198 CONTINUE
000573      4199 CONTINUE
000574      C
000575      C
000576      C TEST FOR CONVERGENCE OF TEMPERATURES
000577      C
000578      NITT = NITT + 1
000579      C
000580      IF(TTEST.LT.(DEPS*XII*0.2)) COT = 1.0
000581      IF(NITT.GE.1000) COT = 1.0
000582      C
000583      C
000584      C NO CONVERGENCE
000585      C
000586      GO TO 4000
000587      C
000588      4200 CONTINUE
000589      C
000590      C
000591      C
000592      C ***** END ITERATION ON TEMPERATURES *****
000593      C
000594      C
000595      C CONVERGENCE ATTAINED, CONTINUE CYCLE
000596      C
000597      C REPLACE T WITH T-TILDES
000598      C
000599      TTS2 = 0.0
000600      DO 4299 I = 2,IBP1
000601      DO 4298 J = 2,JBP1
000602      T(I,J) = VWBAR(I,J)
000603      TTS2 = TTS2 + T(I,J)*W(I,J) / (W(I,J) + 1.E-6)
000604      4298 CONTINUE
000605      4299 CONTINUE
000606      C
000607      C
000608      C TO TEST FOR PSEUDO STEADY STATE OF VELOCITY FIELD
000609      C
000610      NSUB = NSUB + 1
000611      IF(NSUB.GT.100) GO TO 4300
000612      VELLUM = ABS(TTS2 - TTS) / (XII*RMESH)
000613      IF(VELLUM.GT.5.E-5) GO TO 1300
000614      C
000615      C
000616      4300 CONTINUE
000617      C
000618      C CALCULATE SVOC, DC, R, AND VCOTOUT AND VOL
000619      C
000620      SUM = 0.0
000621      C
000622      DO 4499 I = 2,IBP1
000623      DO 4493 J = 2,JBP1
000624      AKO(I,J) = (AKO(I,J) + AKAPO*ALAM*(XII-T(I,J)))
000625      1 / (1.0+ALAM*(XII-T(I,J)))
000626      R(I,J) = R(I,J) + RDOT(I,J)*DTT

```

```

000627      WCOLD = WC(I,J)
000628      WC(I,J) = BETAR*DX*DY*R(I,J)*(XII-T(I,J))
000629      SVDC(I,J) = (WC(I,J) - WCOLD) / DTT
000630      SUM = SUM + SVDC(I,J)
000631      DC(I,J) = R(I,J)
000632      IF(DC(I,J).GT.DCC) DC(I,J) = DCC
000633      4498      CONTINUE
000634      4499      CONTINUE
000635      C
000636      C CALCULATE ENERGY DIFFERENCE
000637      C USE 4.18E-6 TO CONVERT TO MEGAWATTS
000638      C
000639      EDDTIN = RHDW*BW*VDDTIN*TIN
000640      EDDTOUT = RHDW*BW*VDDTOUT*T(IOUT,JOUT)
000641      EDDTDIF = (- EDDTOUT - EDDTIN)*4.18E-6
000642      ESUM = ESUM + EDDTDIF*DTT
000643      C
000644      VDDTOUT = -VDDTIN + SUM
000645      VDDT(IOUT,JOUT) = VDDTOUT
000646      VOL = VOL + SUM*DTT
000647      C
000648      C
000649      C
000650      C CHECK FOR PRINTING INTERVAL
000651      C
000652      IF(NCYCLE.EQ.NCYTER) CALL TERM
000653      4500      CONTINUE
000654      C
000655      IF(NCYCLE.EQ.NCYP) GO TO 5100
000656      C
000657      C
000658      C
000659      5000      CONTINUE
000660      IF(NCYCLE.EQ.NCYF) GO TO 7000
000661      C
000662      C
000663      C
000664      C
000665      C INCREMENT TIME AND CYCLE
000666      C
000667      NCYCLE = NCYCLE + 1
000668      TI = TI + DTT
000669      GO TO 900
000670      C
000671      C *****
000672      C ***** END CYCLE *****
000673      C *****
000674      C
000675      C
000676      5100      CONTINUE
000677      C
000678      C CALL PRINT AND PLOT ROUTINES
000679      C
000680      CALL PRINT
000681      CALL PLOTPKG
000682      C
000683      C CHECK FOR DONE

```

```
000684 000684 C
000685 000685 NCYP = NCYP + NCYI
000686 000686 GO TO 5000
000687 000687 C
000688 000688 C
000689 000689 7000 CONTINUE
000690 000690 CALL PLOTPKG
000691 000691 CALL PRINT
000692 000692 CALL TERM
000693 000693 CALL DRDP
000694 000694 CALL DOTIM(ISEC)
000695 000695 CPTIME = FLOAT(ISEC) / 1.E6
000696 000696 WRITE (6,9100) CPTIME
000697 000697 CALL EXIT(1)
000698 000698 C
000699 000699 C
000700 000700 9100 FORMAT(/,14HSECONDS USED =,1PE14.5)
000701 000701 C
000702 000702 END
```



```

1*****
000001 000703
000002 000704
000003 000705
000004 000706
000005 000707
000006 000708
000007 000709
000008 000710
000009 000711
000010 000712
000011 000713
000012 000714
000013 000715
000014 000716
000015 000717
000016 000718
000017 000719
000018 000720
000019 000721
000020 000722
000021 000723
000022 000724
000023 000725
000024 000726
000025 000727
000026 000728
000027 000729
000028 000730
000029 000731
000030 000732
000031 000733
000032 000734
000033 000735
000034 000736
000035 000737
000036 000738

```

```

CHAT 157D 10:02:50 R04/C4/80

C
C
C
SUBROUTINE TERM
C
C
PARAMETER (NI=25)
PARAMETER (NJ=25)
COMMON/ARRAYS/ AKO(NI,NJ), D(NI,NJ), DC(NI,NJ), GAM(NI,NJ),
1 HC(NI,NJ), PHI(NI,NJ), R(NI,NJ), ROOT(NI,NJ),
2 SIG(NI,NJ), SVDC(NI,NJ), T(NI,NJ), TBAR(NJ),
3 U(NI,NJ), UWBAR(NI,NJ), V(NI,NJ), VDOT(NI,NJ),
4 VWBAR(NI,NJ), W(NI,NJ), WC(NI,NJ)
C
C
COMMON/VAR/ AKAPO, AKAPR, AKR, ALAM, BR, BW, BETA, BETAR, DCO, DSUM,
1 DT, DTR, DTT, DX, DY, OXR, DYR, DXDYR, DXSQ, DYSQ, DXSQR,
2 DYSQR, EIGFIF, EPSILON, GRAV, IBAR, IBP1, IBP2, IIN,
3 IOUT, JBAR, JBP1, JBP2, JIN, JOUT, NCYCLE, NCYF,
4 NCYI, NCYP, NCYTER, NCYTERI, NIT, NITT, RHOR, RHO4,
5 RPC, RPCSR, SIGCO, SIGOMEG, TFAC, TI, TIN, TWOTHI, VDOTIN,
6 VDOTOUT, VIN, VOL, WO, WMAX, XII, NSUB, EDOTOIF, ESUM
C
C
NCYTER = NCYTER + NCYTERI
C
C
C PRINT OUT CURRENT TIME, ENERGY DIFFERENCE, ITERATION NUMBER
C AND CYCLE NUMBER ON TELETYPE WHILE RUNNING
C
PRINT 9110, TI, EDOTOIF, VOL, ESUM, NIT, NITT, NSUB, NCYCLE
RETURN
9110 FORMAT(3HTI=,E9.2,1X,2HE=,E9.2,1X,2HV=,E9.2,1X,
3 3HES=,E9.2,1X,I3,1X,I3,1X,I2,1X,I4)
END

```

```

1*****
000001 000739 C
000002 000740 C
000003 000741 C
000004 000742 SUBROUTINE PRINT
000005 000743 C
000006 000744 C
000007 000745 PARAMETER (NI=25)
000008 000746 PARAMETER (NJ=25)
000009 000747 COMMON/ARRAYS/ AKO(NI,NJ),D(NI,NJ),DC(NI,NJ),GAM(NI,NJ),
000010 000748 1 HC(NI,NJ),PHI(NI,NJ),R(NI,NJ),RDOT(NI,NJ),
000011 000749 2 SIG(NI,NJ),SVDC(NI,NJ),T(NI,NJ),TBAR(NJ),
000012 000750 3 U(NI,NJ),UWBAR(NI,NJ),V(NI,NJ),VDOT(NI,NJ),
000013 000751 4 VWBAR(NI,NJ),W(NI,NJ),WC(NI,NJ)
000014 000752 C
000015 000753 C
000016 000754 COMMON/VAR/ AKAPC,AKAPR,AKR,ALAM,BR,BW,BETA,BETAR,DCO,DSUM,
000017 000755 1 DT,DTR,DTT,DX,DY,DXR,DYR,DXDYR,DXSQ,DYSQ,DXSQR,
000018 000756 2 DYSQR,EIGFIF,EPSILON,GRAV,IBAR,IBP1,IBP2,IIN,
000019 000757 3 IQUT,JBAR,JBP1,JBP2,JIN,JOUT,NCYCLE,NCYF,
000020 000758 4 NCYI,NCYP,NCYTER,NCYTERI,NIT,NITT,RHOR,RHOW,
000021 000759 5 RPC,RPCSR,SIGCO,SIGMEG,TFAC,TI,TIN,TWQTHI,VDOTIN,
000022 000760 6 VDOTQUT,VIN,VOL,W0,WMAX,XII,NSUB,EDOTDIF,ESUM
000023 000761 C
000024 000762 C
000025 000763 C
000026 000764 C PRINT STATEMENTS
000027 000765 C
000028 000766 WRITE(6,9000) TI,NCYCLE,NIT
000029 000767 WRITE(6,9042)
000030 000768 WRITE(6,9005) EDOTDIF,DSUM,ESUM
000031 000769 WRITE(6,9042)
000032 000770 IF(NCYCLE.GT.0.0) GO TO 5310
000033 000771 WRITE(6,9010) AKAPC,AKAPR,AKR,ALAM,BR,BW
000034 000772 WRITE(6,9012) BETA,BETAR,DCO,DSUM,DT,DTT
000035 000773 WRITE(6,9015) DX,DY,EPSILON,GRAV,IBAR
000036 000774 WRITE(6,9017) IIN,IQUT,JBAR,JIN,JOUT,NCYF
000037 000775 WRITE(6,9020) NCYI,NCYP,NCYTER,NCYTERI,RHOR
000038 000776 WRITE(6,9022) RHOW,RPC,SIGMEG,SIGCO,TFAC,TI
000039 000777 WRITE(6,9025) TIN,VDOTIN,VDOTQUT,VOL,VIN
000040 000778 WRITE(6,9027) W0,WMAX,XII,NI,NJ
000041 000779 5310 CONTINUE
000042 000780 C
000043 000781 WRITE(6,9040)
000044 000782 WRITE(6,9035)
000045 000783 WRITE(6,9040)
000046 000784 DO 5399 I = 1,IBP2
000047 000785 DO 5399 J = 1,JBP2
000048 000786 WRITE(6,9044) I,J,U(I,J),V(I,J),PHI(I,J),T(I,J),W(I,J),
000049 000787 1 GAM(I,J),D(I,J),VDOT(I,J)
000050 000788 5398 CONTINUE
000051 000789 WRITE(6,9040)
000052 000790 5399 CONTINUE
000053 000791 C
000054 000792 WRITE(6,9050)
000055 000793 WRITE(6,9055)
000056 000794 WRITE(6,9040)

```

```

000057 000795 DO 5499 I = 1,IBP2
000058 000796 DO 5498 J = 1,J3P2
000059 000797 WRITE(6,9045) I,J,WC(I,J),SIG(I,J),AKO(I,J),SVDC(I,J),
000060 000798 1 RDOT(I,J),R(I,J),VDDOT(I,J)
000061 000799 5498 CONTINUE
000062 000800 WRITE(6,9040)
000063 000801 5499 CONTINUE
000064 000802 C
000065 000803 C
000066 000804 RETURN
000067 000805 C
000068 000806 C
000069 000807 C
000070 000808 C
000071 000809 C FORMATS
000072 000810 C
000073 000811 9000 FORMAT(1H1,5X,6HTIME= ,1PE12.5,15X,8HNCCYCLE= ,I6,10X,5HNIT= ,I6)
000074 000812 9005 FORMAT(7HEDOTDIF,E9.2,5X,5HDSUM=,E9.2,10X,5HESUM=,E9.2)
000075 000813 9010 FORMAT(6HAKAP0=,E9.2,2X,6HAKAPR=,E9.2,2X,4HAKR=,E9.2,5X,
000076 000814 1 5HALAM=,E9.2,3X,3HBR=,E9.2,3X,3HBW=,E9.2)
000077 000815 9012 FORMAT(5HBETA=,E9.2,3X,6HBETAR=,E9.2,2X,4HDCO=,E9.2,5X,
000078 000816 1 5HDSUM=,E9.2,3X,3HDT=,E9.2,3X,4HDTT=,E9.2)
000079 000817 9015 FORMAT(3HDX=,E9.2,5X,3HDY=,E9.2,5X,8HEPSILON=,E9.2,1X,
000080 000818 1 5HGRAV=,E9.2,2X,5HIBAR=,I4)
000081 000819 9017 FORMAT(4HIIN=,I4,9X,5HIOUT=,I4,8X,5HJSAR=,I4,9X,4HJIN=,I4,9X,
000082 000820 1 5HJOUT=,I4,6X,5HNCCYF=,I4)
000083 000821 9020 FORMAT(5HNCCI=,I4,8X,5HNCCYP=,I4,8X,7HNCCYTER=,I4,7X
000084 000822 1 8HNCCYTERI=,I4,5X,4HNIT=,I4)
000085 000823 9022 FORMAT(5HRHOW=,E9.2,3X,4HRPC=,E9.2,4X,8HSIGMEG=,E9.2,1X,
000086 000824 1 6HSIGCO=,E9.2,1X,5HTFAC=,E9.2,3X,3HTI=,E9.2)
000087 000825 9025 FORMAT(4HTIN=,E9.2,4X,7HVDOTIN=,E9.2,1X,8HVDOTOUT=,E9.2,
000088 000826 1 1X,4HVOL=,E9.2,3X,4HVIN=,E9.2)
000089 000827 9027 FORMAT(3HJO=,E9.2,5X,5HWMAX=,E9.2,3X,4HXII=,E9.2,5X,
000090 000828 1 3HNT=,I4,10X,3HNJ=,I4)
000091 000829 9035 FORMAT(/,8HCELL I J,8X,1HU,10X,1HV,10X,3HPHI,11X,1HT,10X,1HW,
000092 000830 1 11X,3HGAM,10X,1HD,10X,4HVDDOT)
000093 000831 9040 FORMAT(/)
000094 000832 9042 FORMAT(/)
000095 000833 9044 FORMAT(2I4,2X,8(1PE10.3,2X))
000096 000834 9045 FORMAT(2I4,2X,8(1PE10.3,2X))
000097 000835 9050 FORMAT(///)
000098 000836 9055 FORMAT(/,8HCELL I J,8X,2HWC,10X,3HSIG,9X,3HAKO,7X,4HSVDC,
000099 000837 1 8X,4HRDOT,8X,1HR,12X,4HVDDOT)
000100 000838 C
000101 000839 END

```

```
1*****  
000001      000840      C  
000002      000841      C  
000003      000842  
000004      000843      C  
000005      000844  
000006      000845  
000007      000846  
000008      000847  
000009      000848  
000010      000849  
000011      000850  
000012      000851  
000013      000852  
000014      000853  
000015      000854  
000016      000855  
000017      000856  
000018      000857  
000019      000858  
000020      000859  
000021      000860  
  
CHAT  1670  10:02:50  R04/C4/80  
  
SUBROUTINE UNDROP  
  
LCM (BINJLN),(FLMBUF)  
COMMON /BINJLN/ IBSIZE,IBUFF(5000)  
COMMON /FLMBUF/ FLMBUF(5000)  
DIMENSION IHEAD(6)  
IBSIZE = 5000  
CALL FILM80  
CALL IDENT80 (12,FLMBUF,5000,1100000,105,0)  
IHEAD(1) = 10H T3 RUTH  
IHEAD(2) = 10HREX 219-A  
IHEAD(3) = 10HSIGDM=1.0  
IHEAD(4) = 10HW0 VARIES  
IHEAD(5) = 10HTFAC=.075  
IHEAD(6) = 10HVDOT=1.E5  
CALL HEAD80 (12,60,IHEAD,10HBOX T3RUTH)  
CALL KEEPFLM (12)  
RETURN  
END
```

1*****
000001 000861
000002 000862
000003 000863
000004 000864
000005 000865
000006 000866
000007 000867
000008 000868
000009 000869
000010 000870
000011 000871

C
C
C

CHAT 167D 10:02:50 RC4/04/80

SUBROUTINE DROP

LCM (FLMBUFC)
COMMON /FLMBUFC/ FLMBUF(5000)
CALL FILM80
CALL IDENT80 (12,FLMBUF,5000,1100000,105,0)
CALL KEEPFLM(12)
RETURN
END

```

1*****
000001 000872 C
000002 000873 C
000003 000874 SUBROUTINE PLOTPKG
000004 000875 C
000005 000876 C
000006 000877 PARAMETER (NI=25)
000007 000878 PARAMETER (NJ=25)
000008 000879 COMMON/ARRAYS/ AKG(NI,NJ),D(NI,NJ),DC(NI,NJ),GAM(NI,NJ),
000009 000880 1 HC(NI,NJ),PHI(NI,NJ),R(NI,NJ),RDOT(NI,NJ),
000010 000881 2 SIG(NI,NJ),SVDC(NI,NJ),T(NI,NJ),TBAR(NJ),
000011 000882 3 U(NI,NJ),UWBAR(NI,NJ),V(NI,NJ),VDOT(NI,NJ),
000012 000883 4 VWBAR(NI,NJ),W(NI,NJ),WC(NI,NJ)
000013 000884 C
000014 000885 C
000015 000886 COMMON/VAR/ AKAPO,AKAPR,AKR,ALAM,BR,BW,BETA,BETAR,DCO,DSUM,
000016 000887 1 DT,DTR,DTT,DX,DY,DXR,DYR,DXDYR,DXSQ,DYSQ,DXSQR,
000017 000888 2 DYSQR,EIGFIF,EPSILON,GRAV,IBAR,IBP1,IBP2,IIN,
000018 000889 3 ICUT,JBAR,JBP1,JBP2,JIN,JOUT,NCYCLE,NCYF,
000019 000890 4 NCYI,NCYP,NCYTER,NCYTERI,NIT,NITT,RHOR,RHOW,
000020 000891 5 RPC,RPCSR,SIGCO,SIGOMEG,TFAC,TI,TIN,TWOTHI,VDOTIN,
000021 000892 6 VDOTOUT,VIN,VOL,W0,WMAX,XII,NSUB,EDDTDIF,ESUM
000022 000893 C
000023 000894 C
000024 000895 COMMON/CJF07/ IXL,IXR,IYT,IYB,XMN,XXM,YMX,YMN
000025 000896 DIMENSION CON(25),BCD(2),TIME(6),VAVE(2),X(1),Y(1)
000026 000897 EQUIVALENCE (UWBAR(1,1),X(1)),(VWBAR(1,1),Y(1))
000027 000898 C
000028 000899 C
000029 000900 C CONTOUR PLOTS
000030 000901 C
000031 000902 IPLT=1
000032 000903 C
000033 000904 C SET COORDINATES FOR CELL CENTERED QUANTITIES
000034 000905 C (THE BOUNDARIES ARE UNCHANGED.)
000035 000906 C
000036 000907 95 CONTINUE
000037 000908 X(1)=0.
000038 000909 Y(1)=0.
000039 000910 X(2) = .5*DX
000040 000911 Y(2) = .5*DY
000041 000912 DO 100 I = 2,IBAR
000042 000913 X(I+1) = X(I) + DX
000043 000914 100 CONTINUE
000044 000915 C
000045 000916 DO 120 J = 2,JBAR
000046 000917 Y(J+1) = Y(J) + DY
000047 000918 120 CONTINUE
000048 000919 X(IBP2)=X(IBP1)+.5*DX
000049 000920 Y(IBP2)=Y(IBP1)+.5*DY
000050 000921 C
000051 000922 123 CONTINUE
000052 000923 C
000053 000924 GO TO (125,155,195,650,215),IPLT
000054 000925 C
000055 000926 125 CONTINUE
000056 000927 C

```

```

000057 000928 C LABELS AND NUMBER OF CHARACTERS FOR AXES
000058 000929 BCD(1) = 5HX-AXIS
000059 000930 BCD(2) = 6HY-AXIS
000060 000931 NXN = 6
000061 000932 NYN = 6
000062 000933 C
000063 000934 C NO. OF CHARACTERS FOR FRAME TITLE
000064 000935 MZL = 60
000065 000936 C
000066 000937 C
000067 000938 C
000068 000939 C
000069 000940 C PLOT W - PRIMARY CRACK WIDTH
000070 000941 C
000071 000942 C WRITE LARGE CHARACTERS FOR TIME (REPLACES ENCODE)
000072 000943 C
000073 000944 WRITE(63,900) TI
000074 000945 REWIND 63
000075 000946 READ(63,901) (TIME(I),I=1,6)
000076 000947 901 FORMAT(6A10)
000077 000948 900 FORMAT("PRIMARY CRACK WIDTH AT T= ",E10.3,10X,"W")
000078 000949 C
000079 000950 C STORE CRACK WIDTH IN DUMMY ARRAY HC
000080 000951 JM = .5*JBAR
000081 000952 DO 150 I = 1,I8P2
000082 000953 DO 150 J = 1,J8P2
000083 000954 HC(I,J) = W(I,J)
000084 000955 150 CONTINUE
000085 000956 C
000086 000957 CALL CONTRJB(X,-I8P2,Y,-J8P2,HC(1,1),NI,NJ,10,C.,0.,-1.,
000087 000958 1 CON,DX,DY,1,TIME,MZL,6HX-AXIS,6,6HY-AXIS,6)
000088 000959 IPLT=IPLT+1
000089 000960 GO TO 123
000090 000961 C
000091 000962 C
000092 000963 C
000093 000964 C
000094 000965 C PLOT T - WATER TEMPERATURE
000095 000966 C
000096 000967 155 CONTINUE
000097 000968 REWIND 63
000098 000969 WRITE(63,910) TI
000099 000970 REWIND 63
000100 000971 READ(63,911) (TIME(I),I=1,6)
000101 000972 910 FORMAT("WATER TEMPERATURE AT T= ",E10.3,12X,"T")
000102 000973 C
000103 000974 C STORE WATER TEMPERATURE IN DUMMY ARRAY
000104 000975 DO 180 I=1,I8P2
000105 000976 DO 180 J=1,J8P2
000106 000977 HC(I,J)=T(I,J)
000107 000978 180 CONTINUE
000108 000979 C
000109 000980 C
000110 000981 C
000111 000982 192 CONTINUE
000112 000983 CALL CONTRJB(X,-I8P2,Y,-J8P2,HC(1,1),NI,NJ,10,C.,0.,-1.,
000113 000984 1 CON,DX,DY,1,TIME,MZL,6HX-AXIS,6,6HY-AXIS,6)

```

```

000114 000985 C
000115 000986 C
000116 000987 IPLT=IPLT+1
000117 000988 GO TO 123
000118 000989 C
000119 000990 C
000120 000991 C
000121 000992 C
000122 000993 C PLOT PHI - MODIFIED PRESSURE
000123 000994 C
000124 000995 195 CONTINUE
000125 000996 REWIND 63
000126 000997 WRITE(63,920) TI
000127 000998 REWIND 63
000128 000999 READ(63,901) (TIME(I),I=1,6)
000129 001000 920 FORMAT("MODIFIED PRESSURE AT T= ",E10.3,11X,"PHI")
000130 001001 C
000131 001002 C
000132 001003 C USE HC ARRAY TO PLOT PHIS.
000133 001004 C
000134 001005 DO 200 I = 1,JBP2
000135 001006 DO 200 J = 1,JBP2
000136 001007 HC(I,J) = PHI(I,J)
000137 001008 200 CONTINUE
000138 001009 C
000139 001010 CALL CONTRJB(X,-IBP2,Y,-JBP2,HC(1,1),NI,NJ,10,0.,0.,-1.,
000140 001011 1 CON,DX,DY,1,TIME,4ZL,6HX-AXIS,6,6HY-AXIS,6)
000141 001012 IPLT=IPLT+1
000142 001013 GO TO 123
000143 001014 C
000144 001015 C
000145 001016 C
000146 001017 C
000147 001018 C PLOT R - DEPTH OF THERMAL CRACK
000148 001019 C
000149 001020 650 CONTINUE
000150 001021 REWIND 63
000151 001022 WRITE(63,995) TI
000152 001023 REWIND 63
000153 001024 READ(63,901) (TIME(I),I=1,6)
000154 001025 995 FORMAT("DEPTH OF THERMAL CRACK AT T= ",E10.3,7X,"R")
000155 001026 C
000156 001027 C USE HC ARRAY TO PLOT R
000157 001028 C
000158 001029 DO 679 I = 1,IBP2
000159 001030 DO 679 J = 1,JBP2
000160 001031 HC(I,J) = R(I,J)
000161 001032 679 CONTINUE
000162 001033 679 CONTINUE
000163 001034 C
000164 001035 CALL CONTRJB(X,-IBP2,Y,-JBP2,HC(1,1),NI,NJ,10,0.,0.,-1.,
000165 001036 1 CON,DX,DY,1,TIME,MZL,6HX-AXIS,6,6HY-AXIS,6)
000166 001037 IPLT=IPLT+1
000167 001038 GO TO 123
000168 001039 C
000169 001040 C
000170 001041 C

```



```

000171 001042 C
000172 001043 C
000173 001044 C
000174 001045 C VECTOR PLOTS
000175 001046 C
000176 001047 215 CONTINUE
000177 001048 C
000178 001049 IF(NCYCLE.EQ.0.0) GO TO 997
000179 001050 C
000180 001051 C NORMALIZATION
000181 001052 C
000182 001053 VELF = 0.0
000183 001054 DO 300 I = 2,IBP1
000184 001055 DO 301 J = 2,JBP1
000185 001056 STEST = SQRT((.5*(U(I+1,J)+U(I,J)))*2
000186 001057 1 +(.5*(V(I,J+1)+V(I,J)))*2)
000187 001058 VELF = AMAX1(STEST,VELF)
000188 001059 301 CONTINUE
000189 001060 300 CONTINUE
000190 001061 C
000191 001062 C VELF IS MAXIMUM VELOCITY COMPONENT IN SYSTEM
000192 001063 C PICK OUT THE MAXIMUM VELOCITY COMPONENT IN SYSTEM
000193 001064 C
000194 001065 DF = DX
000195 001066 IF(DX.GT.DY) DF = DY
000196 001067 VNORM = .50
000197 001068 VELF = VNORM*DF/VELF
000198 001069 C
000199 001070 C SET UP FRAME AND LABEL
000200 001071 C
000201 001072 C
000202 001073 C WRITE LARGE CHARACTERS FOR TIME
000203 001074 C
000204 001075 REWIND 63
000205 001076 WRITE(63,930) TI
000206 001077 REWIND 63
000207 001078 READ(63,931) (TIME(I),I=1,6)
000208 001079 930 FORMAT("VELOCITY AT T= ",E10.3)
000209 001080 C
000210 001081 CALL ADV(1)
000211 001082 CALL FRAME(IXL,IXR,IYT,IYB)
000212 001083 CALL DGA(IXL,IXR,IYT,IYB,XMN,XXM,YMN,YYM)
000213 001084 C
000214 001085 C CENTER THE TITLE AND THE LABELS FOR THE VECTOR PLOTS
000215 001086 C IX=MAX0(48,IXL+(IXR-IXL-12*MZL)/2)
000216 001087 IX=IXL
000217 001088 LZL=MAX0(MZL-(1024-IX)/12,0)+1
000218 001089 IF(LZL.GT.1)LZL=(LZL+83)/85+1
000219 001090 CALL DLCH(IX,MAX0(10,IYT-24-16*LZL),MZL,TIME(1),2)
000220 001091 CALL DLCH(MAX0(54,IXL+(IXR-IXL-12*NXN)/2),IYB+26,NXN,BCD(1),2)
000221 001092 CALL DLCH(78,MIN0(IYB+52,IYB-(IYB-IYT-12*NYN)/2),NYN,BCD(2),2)
000222 001093 C
000223 001094 C CALCULATE VELOCITY VECTORS
000224 001095 C
000225 001096 DO 320 J = 2,JBP1
000226 001097 Y1 = Y(J)
000227 001098 DO 320 I = 2,IBP1

```

```

000228 001099      X1 = X(I)
000229 001100      X2 = X1+VELF*(U(I-1,J)+U(I,J))
000230 001101      Y2 = Y1+VELF*(V(I,J-1)+V(I,J))
000231 001102      CALL CONVRT(X1,IX1,XMN,VMX,IXL,IXR)
000232 001103      CALL CONVRT(Y1,IY1,YMN,VMX,IYB,IYT)
000233 001104      CALL CONVRT(X2,IX2,XMN,VMX,IXL,IXR)
000234 001105      CALL CONVRT(Y2,IY2,YMN,VMX,IYB,IYT)
000235 001106      CALL DRV(IX1,IY1,IX2,IY2)
000236 001107      320 CONTINUE
000237 001108      C
000238 001109      C DETERMINE MAX SPEED AND ZONE
000239 001110      VMX = 0.
000240 001111      IMX = I
000241 001112      JMX = J
000242 001113      DO 342 I = 2,IBP1
000243 001114      DO 342 J = 2,JBP1
000244 001115      VAVE(1) = .5*(U(I-1,J) + U(I,J))
000245 001116      VAVE(2) = .5*(V(I,J-1) + V(I,J))
000246 001117      VAVE(1)=VAVE(1)*VAVE(1)
000247 001118      VAVE(2) = VAVE(2)*VAVE(2)
000248 001119      VTEST = SQRT(VAVE(1)+VAVE(2))
000249 001120      IF(VMX.GT.VTEST) GO TO 342
000250 001121      VMX = VTEST
000251 001122      IMX = I
000252 001123      JMX = J
000253 001124      342 CONTINUE
000254 001125      C
000255 001126      C
000256 001127      C IDENTIFY MAX SPEED AND ZONE
000257 001128      REWIND 63
000258 001129      WRITE(63,940) VMX,IMX,JMX
000259 001130      REWIND 63
000260 001131      READ(63,941) (TIME(I),I=1,6)
000261 001132      CALL DLCH(116,1000,60,TIME(1),2)
000262 001133      940 FORMAT(1X,19HMAXIMUM SPEED,I,J: ,1PE12.5,2I4)
000263 001134      941 FORMAT(1X,6A10)
000264 001135      C
000265 001136      C
000266 001137      997 CONTINUE
000267 001138      RETURN
000268 001139      END

```

```

1*****
000001 001140 C CHAT 1570 10:02:50 R04/04/80
000002 001141 C
000003 001142 SUBROUTINE PLOJB(X,Y,NPTS,INC,LNN,NSYM,C,XAA,YAA,LABELZ,NZL,LABELX
000004 001143 1,NXL,LABELY,NYL)
000005 001144 C
000006 001145 C LABEL AXES, WRITE TITLES, DRAW BORDERS AND CONNECT CONTOUR POINTS
000007 001146 C ESTABLISHED IN TRICJB
000008 001147 COMMON /CIE07/ IXL,IXR,IYT,IYB,XMX,XXA,YMX,YMA
000009 001148 DIMENSION X(1), Y(1)
000010 001149 INTEGER GRIDF
000011 001150 R=AMAX1(AMAX1(C,0.)*(LNN+1),0.)
000012 001151 LIN=LNN
000013 001152 KSYM=IABS(NSYM)
000014 001153 KINC=MAX0(IABS(INC),1)
000015 001154 C
000016 001155 C ESTABLISH NO. CHARACTERS AND NO. POINTS
000017 001156 MPTS=IABS(NPTS)
000018 001157 NZL=IABS(NZL)
000019 001158 XXA=ABS(XAA)
000020 001159 YYA=ABS(YAA)
000021 001160 NXN=IABS(NXL)
000022 001161 NYN=IABS(NYL)
000023 001162 GRIDF=AMAX1(1.,ABS(C))
000024 001163 IF (NSYM.LT.0) GO TO 130
000025 001164 C
000026 001165 C ADVANCE FRAME AND DETERMINE MAX AND MIN FOR AXES
000027 001166 CALL ADV (1)
000028 001167 CALL MAXV (X,KINC,MPTS,ISUB,XMX)
000029 001168 CALL MAXV (Y,KINC,MPTS,ISUB,YMX)
000030 001169 CALL MINV (X,KINC,MPTS,ISUB,XMN)
000031 001170 CALL MINV (Y,KINC,MPTS,ISUB,YMN)
000032 001171 C
000033 001172 C IF XXA AND/OR YYA ARE NON-ZERO THE LENGTHS
000034 001173 C WILL BE CONSIDERED AS RATIOS WHERE THE LONGEST
000035 001174 C SIDE IS FITTED ON A 860 POINT LINE.
000036 001175 C DETERMINE ASPECT RATIO OF BORDERS
000037 001176 IF (XXA.EQ.0) XXA=6.
000038 001177 IF (YYA.EQ.0) YYA=10.
000039 001178 DXM=.001*ABS(XMX)
000040 001179 IF (DXM.EQ.0) DXM=.0001
000041 001180 XMN=XMN-DXM
000042 001181 XMX=XMX+DXM
000043 001182 60 IF (ISIGN(1,NYL).LT.0.AND.INC.GT.0) YYA=(YMX-YMN)/YYA
000044 001183 IF (ISIGN(1,NXL).LT.0.AND.NPTS.GT.0) XXA=(XMX-XMN)/XXA
000045 001184 C
000046 001185 C DETERMINE 4020 COORDINATES OF BORDER
000047 001186 IXL=120
000048 001187 IYB=910
000049 001188 FACT=860./AMAX1(XXA,YYA)
000050 001189 IXR=120+IFIX(FACT*XXA)
000051 001190 IYT=910-IFIX(FACT*YYA)
000052 001191 CALL FRAME (IXL,IXR,IYT,IYB)
000053 001192 IF (SIGN(1.,XAA).GT.0) GO TO 70
000054 001193 SWAP=XMN
000055 001194 XMN=XMX
000056 001195 XMX=SWAP

```

```

000057 001196 70 IF (SIGN(1.,YAA).GT.0) GO TO 80
000058 001197 SWAP=YMN
000059 001198 YMN=YMN
000060 001199 YMX=SWAP
000061 001200 80 CALL DGA (IXL,IXR,IYT,IYB,XMN,XXM,YMX,YMN)
000062 001201 C
000063 001202 C CENTER TITLE AND LABEL AXES
000064 001203 IF (NXN.NE.0) CALL DLCH (MAX0(54,IXL+(IXR-IXL-12*NXN)/2),IYB+25
000065 001204 1,NXN,LABELX,2)
000066 001205 INCX=78
000067 001206 IF (NYN.GT.85) INCX=18
000068 001207 IF (NYN.NE.0) CALL DLCH (INCX,MIN0(IYB+52,IYB-(IYB-IYT-12*NYN)/2)
000069 001208 1,NYN,LABELY,2)
000070 001209 C
000071 001210 IX=MAX0(49,IXL+(IXR-IXL-12*MZL)/2)
000072 001211 IX=IXL
000073 001212 LZL=MAX0(MZL-(1024-IX)/12,0)+1
000074 001213 IF (LZL.GT.1) LZL=(LZL+83)/85+1
000075 001214 IF (N7L.NE.0) CALL DLCH (IX,MAX0(10,IYT-24-16*LZL),MZL,LABELZ,2)
000076 001215 C
000077 001216 C INCREASE EXPOSURE
000078 001217 CALL EXH
000079 001218 IF (N7L.LT.0) GO TO 220
000080 001219 C
000081 001220 C PLOT POINTS AND/OR LINE
000082 001221 130 MPTS=MPTS*KINC
000083 001222 DO 210 NXP=1,MPTS,KINC
000084 001223 XTWO=X(NXP)
000085 001224 YTWO=Y(NXP)
000086 001225 IF (NPTS.LT.0) XTWO=ALOG10(XTWO)
000087 001226 IF (INC.LT.0) YTWO=ALOG10(YTWO)
000088 001227 CALL CONVRT (XTWO,NXTWO,XMN,XXM,IXL,IXR)
000089 001228 CALL CONVRT (YTWO,NYTWO,YMN,YMX,IYB,IYT)
000090 001229 IF (NXP.EQ.1) GO TO 190
000091 001230 IF (LIN.GE.0) GO TO 180
000092 001231 140 IF (MOD(((NXP-1)/KINC),IABS(LIN)).NE.0) GO TO 150
000093 001232 C
000094 001233 C LIGHTEN EXPOSURE
000095 001234 CALL EXL
000096 001235 C
000097 001236 C REMOVES CONTOUR LABELS
000098 001237 C
000099 001238 CALL DLCH (NXTWO,NYTWO,0,KSYM,1)
000100 001239 CALL EXH
000101 001240 GO TO 200
000102 001241 150 IF (B.EQ.0.) GO TO 200
000103 001242 160 DO 170 IB=1,4
000104 001243 170 CALL PLT (NXTWO,NYTWO,42)
000105 001244 GO TO 200
000106 001245 180 IF (B.EQ.0.) CALL DRV (NXONE,NYONE,NXTWO,NYTWO)
000107 001246 190 IF (LIN.NE.0) GO TO 140
000108 001247 IF (B.NE.0.) GO TO 160
000109 001248 200 NYONE=NYTWO
000110 001249 NXONE=NXTWO
000111 001250 210 CONTINUE
220 RETURN
END

```

```

000057 001307 C
000058 001308 C STORE THE ARRAY BEING CONTOURED IN ZT AND CONTOUR ONE CELL AT A TIME
000059 001309 50 DO 90 NY=2,N0Y
000060 001310 IX=MOD(NY,2)
000061 001311 DY=Y(NY)-Y(NY-1)
000062 001312 DO 80 INX=2,N0X
000063 001313 NX=INX
000064 001314 IF (IX.NE.0) NX=N0X-INX+2
000065 001315 ZT(1)=Z(NX-1,NY-1)
000066 001316 ZT(2)=Z(NX,NY-1)
000067 001317 ZT(3)=Z(NX,NY)
000068 001318 ZT(4)=Z(NX-1,NY)
000069 001319 DX=X(NX)-X(NX-1)
000070 001320 IF (ABS(ZT(3)-ZT(1))-ABS(ZT(4)-ZT(2))) 70,60,60
000071 001321 60 CALL TRICJB (X(NX),Y(NY),-DX,-DY,N0C,ZPLAN,ZT(4),ZT(3),ZT(2))
000072 001322 CALL TRICJB (X(NX-1),Y(NY-1),DX,DY,N0C,ZPLAN,ZT(2),ZT(1),ZT(4))
000073 001323 GO TO 80
000074 001324 70 CALL TRICJB (X(NX-1),Y(NY),DX,-DY,N0C,ZPLAN,ZT(3),ZT(4),ZT(1))
000075 001325 CALL TRICJB (X(NX),Y(NY-1),-DX,DY,N0C,ZPLAN,ZT(1),ZT(2),ZT(3))
000076 001326 80 CONTINUE
000077 001327 90 CONTINUE
000078 001328 C
000079 001329 C
000080 001330 C IDENTIFY THE MAX, MIN, AND SPACING FOR CONTOURS
000081 001331 REWIND 63
000082 001332 WRITE(63,900) ZPLAN(1),ZPLAN(N0C),DELZ
000083 001333 REWIND 63
000084 001334 READ(63,901) (CON(I),I=1,6)
000085 001335 CALL DLCH(48,1000,60,CON(1),2)
000086 001336 900 FORMAT(1X,15HMIN,MAX,DELTA: ,1P3E12.4)
000087 001337 901 FORMAT(1X,6A10)
000088 001338 RETURN
000089 001339 C
000090 001340 END

```

```

1*****
000001 001251 C
000002 001252 C
000003 001253 SUBROUTINE CONTRJB(X,NNX,Y,NNY,Z,NZX,NZY,NC,ZMN,ZMX,DLZ,ZPLAN,DMPX
000004 001254 1,DMPY,IGRD,ITITLE,NTITLE,XLABEL,NXLBL,YLABEL,NYLBL)
000005 001255 C
000006 001256 C
000007 001257 C ESTABLISH THE CONTOUR SCHEME TO BE USED BY TRICJB
000008 001258 COMMON /CNTRCOM/ ISYM(50)
000009 001259 DIMENSION XSCALE(2), YSCALE(2)
000010 001260 EQUIVALENCE (XMIN,XSCALE(1)), (XMAX,XSCALE(2))
000011 001261 EQUIVALENCE (YMIN,YSCALE(1)), (YMAX,YSCALE(2))
000012 001262 DIMENSION X(1), Y(1), Z(NZX,1), ZPLAN(1)
000013 001263 DIMENSION ZT(4)
000014 001264 DIMENSION CON(6)
000015 001265 NDC=MINO(IABS(NC),50)
000016 001266 ZMIN=ZMN
000017 001267 ZMAX=ZMX
000018 001268 DELZ=DLZ
000019 001269 DMAPX=DMPX
000020 001270 DMAPY=DMPY
000021 001271 NOX=IABS(NNX)
000022 001272 NOY=IABS(NNY)
000023 001273 DO 10 I=1,50
000024 001274 10 ISYM(I)=0
000025 001275 C
000026 001276 C ESTABLISH SCALES
000027 001277 CALL MINV (X,1,NOX,I,XMIN)
000028 001278 CALL MAXV (Y,1,NOY,I,YMAX)
000029 001279 CALL MAXV (X,1,NOX,I,XMAX)
000030 001280 CALL MINV (Y,1,NOY,I,YMIN)
000031 001281 FGRD=0.
000032 001282 IF (IGRD.GT.0) FGRD=-IGRD
000033 001283 CALL PLOJB (XSCALE,YSCALE,2,1,1,1,FGRD,DMAPX,DMAPY,ITITLE,-NTITLE
000034 001284 1,XLABEL,NXLBL,YLABEL,NYLBL)
000035 001285 IF (NC.LT.0) GO TO 50
000036 001286 C
000037 001287 C DETERMINE MAX-MIN CONTOURS AND SPACING
000038 001288 C STORE IN ARRAY ZPLAN
000039 001289 IF (NNX.LE.0) CALL MINM (Z,NZX,NOX,NOY,I,J,ZMIN)
000040 001290 IF (NNY.LE.0) CALL MAXM (Z,NZX,NOX,NOY,I,J,ZMAX)
000041 001291 IF (DELZ.GT.0) GO TO 20
000042 001292 C
000043 001293 C ADD FUDGE FACTOR TO KEEP MIN AND MAX CONTOUR
000044 001294 C LINES FROM BEING PLOTTED RANDOMLY
000045 001295 C
000046 001296 FUDGE = 1.E-10*(ZMAX-ZMIN)
000047 001297 ZMIN = ZMIN + FUDGE
000048 001298 ZMAX = ZMAX - FUDGE
000049 001299 DELZ=(ZMAX-ZMIN)/(NDC-1.)
000050 001300 20 IF (NZY.GT.0) GO TO 30
000051 001301 ZMAX=ZMAX-AMOD(ZMAX,DELZ)
000052 001302 ZMIN=ZMIN-AMOD(ZMIN,DELZ)
000053 001303 NDC=MINO(NDC,IFIX((ZMAX-ZMIN)/DELZ+1.01))
000054 001304 30 ZPLAN(1)=ZMIN
000055 001305 DO 40 I=2,NDC
000056 001306 40 ZPLAN(I)=ZPLAN(I-1)+DELZ

```

1*****

CHAT 157D 10:02:50 R04/04/80

```

000001 001341 C
000002 001342 C
000003 001343 SUBROUTINE TRICJB(X,Y,DX,DY,NOC,ZPLAN,ZX,ZV,ZY)
000004 001344 C
000005 001345 C
000006 001346 C PERFORM ACTUAL INTERPOLATIONS NEEDED TO PRODUCE CONTOURS
000007 001347 C DEFINED IN CONTRJB. PLOJB USED TO CONNECT POINTS.
000008 001348 C
000009 001349 COMMON /CNTRCOM/ ISYM(50)
000010 001350 DIMENSION XP(2,50), YP(2,50), ZT(4), ZPLAN(1)
000011 001351 ZT(1)=ZX
000012 001352 ZT(2)=ZY
000013 001353 ZT(3)=ZY
000014 001354 ZT(4)=ZX
000015 001355 ZTMIN=AMIN1(ZT(1),ZT(2),ZT(3))
000016 001356 ZTMAX=AMAX1(ZT(1),ZT(2),ZT(3))
000017 001357 IMIN=NOC+1
000018 001358 IMAX=0
000019 001359 DO 10 K=1,NOC
000020 001360 J=NOC-K+1
000021 001361 IF (ZPLAN(J).GE.ZTMIN) IMIN=J
000022 001362 IF (ZPLAN(K).LE.ZTMAX) IMAX=K
000023 001363 10 CONTINUE
000024 001364 INT=IMAX-IMIN
000025 001365 IF (INT.LT.0.OR.ZTMIN.EQ.ZTMAX) GO TO 130
000026 001366 I2=1
000027 001367 DO 110 K=1,3
000028 001368 ZTMAX=AMAX1(ZT(K),ZT(K+1))
000029 001369 ZPMIN=AMIN1(ZT(K),ZT(K+1))
000030 001370 MIN=NOC+1
000031 001371 MAX=0
000032 001372 DO 20 J=1,NOC
000033 001373 INZ=NOC-J+1
000034 001374 IF (ZPLAN(INZ).GT.ZPMIN.OR.(ZPLAN(INZ).EQ.ZPMIN.AND.ZTMIN.EQ.ZPMIN
000035 001375 1)) MIN=INZ
000036 001376 IF (ZPLAN(J).LE.ZTMAX) MAX=J
000037 001377 20 CONTINUE
000038 001378 INZ=MAX-MIN
000039 001379 IF (INZ.LT.0.OR.ZTMAX.EQ.ZPMIN) GO TO 110
000040 001380 IF (INZ-INT) 40,30,40
000041 001381 30 GO TO (50,40), I2
000042 001382 40 I2=1
000043 001383 GO TO 60
000044 001384 50 I2=2
000045 001385 DO 100 J=MIN,MAX
000046 001386 GO TO (70,80,90), K
000047 001387 70 XP(I2,J)=X+DX*(ZPLAN(J)-ZT(2))/(ZT(1)-ZT(2))
000048 001388 YP(I2,J)=Y
000049 001389 GO TO 100
000050 001390 80 XP(I2,J)=X
000051 001391 YP(I2,J)=Y+DY*(ZPLAN(J)-ZT(2))/(ZT(3)-ZT(2))
000052 001392 GO TO 100
000053 001393 90 XP(I2,J)=X+DX*(ZPLAN(J)-ZT(3))/(ZT(1)-ZT(3))
000054 001394 YP(I2,J)=Y+DY*(ZPLAN(J)-ZT(1))/(ZT(3)-ZT(1))
000055 001395 100 CONTINUE
000056 001396 110 CONTINUE

```

000057	001397	DO 120 J=IMIN,IMAX
000058	001398	ISYM(J)=ISYM(J)+1
000059	001399	L=3
000060	001400	IF (MOD(ISYM(J),10).NE.1) L=0
000061	001401	CALL PLQJR (XP(1,J),YP(1,J),2,1,L,-J,0.,0.,0.,0.,0.,0.,0.)
000062	001402	120 CONTINUE
000063	001403	C
000064	001404	130 RETURN
000065	001405	END