308 - - - Q200312260001

Scientific Notebook No. 491: DECOVALEX III Project Task 2C (11/30/2000 through 03/18/2003)

21
150
R

The Boorum & Pease® Quality Guarantee

The materials and craftsmanship that went into this product are of the finest quality. The pages are thread sewn, meaning they're bound to stay bound. The inks are moisture resistant and will not smear. And the uniform quality of the paper assures consistent rulings, excellent writing surface and erasability. If, at any time during normal use, this product does not perform to your expectations, we will replace it free of charge. Simply write to us:

Boorum & Pease Company
71 Clinton Road, Garden City, NY 11530
Attn: Marketing Services

Any correspondence should include the code number printed at the bottom of this page as well as the book title stamped at the bottom of the spine.

One Good Book Deserves Many Others.

Look for the complete line of Boorum & Pease® Columnar, Journal, and Record books. Custom-designed books also available by special order. For more information about our Customized Book Program, contact your office products dealer. See back cover for other books in this series.

Made in U.S.A.

Initial Entries
Scientific Notebook: #491
Issue Date: November 29, 2001
Project Title: DECOVALEX III project Task 2C
Project staff: Sui-Min Hsiung (210) 522-5209
Project Account No. 20.01402.671

1.1 Background

DECOVALEX (acronym for the **DE**velopment of **CO**upled models and their **VAL**idation against **Ex**periments in nuclear waste isolation) is an international cooperative project to support the development of mathematical models for coupled processes in the geosphere and in thier applications and validation against experiments in the field of nuclear waste isolation. The DECOVALEX project has been designed to increase understanding of coupled thermal-hydrological-mechanical processes as they affect rock-mass responses and radionuclide release and transport from a repository to the biosphere and also to assess how these processes can be described by mathematical models. The DECOVALEX project also attempts to identify contributions of these coupled processes to the overall performance assessment in both near- and far-fields. DECOVALEX includes three phases. DECOVALEX I began in 1991 and was completed in 1995. In this phase, the activities focused on modeling laboratory experiments and benchmark problems. The U.S. Nuclear Regulatory Commission (NRC) was one of the funding members of DECOVALEX I and an active participant. The Center for Nuclear Waste Regulatory Analyses (CNWRA) assisted NRC in performing the analyses. DECOVALEX II started in 1995 and concluded in 1999. This phase focused on using the modeling experience gained during the first phase to simulate experiments conducted in the field. Attention was also given to relate the effects of the thermal-hydrological-mechanical processes to the performance of nuclear waste isolation. The NRC and CNWRA did not participate in this phase of the project. DECOVALEX III began in 1999 and includes four tasks. Task 1 involves modeling the *in situ*, full-scale engineered barriers experiment. Task 2 involves modeling the drift-scale heater test at Yucca Mountain. Task 3 includes three benchmark problems. The first problem presents the implication of thermal-hydrological-mechanical coupling on the near-field performance of a nuclear waste repository. The second problem investigates the effects of upscaling thermal-hydrological-mechanical processes on performance assessment results. The third problem studies the effects of glaciation on rock-mass behavior surrounding a nuclear waste repository. Task 4 attempts to address the issue of applying the effects of thermal-hydrological-mechanical-chemical processes to performance assessment. The NRC, with the assistance of the CNWRA, is actively participating in the DECOVALEX III project. Task 2 is the focus of NRC involvement because this task is most relevant to the high-level waste program in the United States.

11/30/2001

1.2 Objective and Scope

As discussed previously, Task 2 of the DECOVALEX III project models the thermal-hydrological-mechanical-chemical behavior of the drift-scale heater test performed at the Exploratory Studies Facility at Yucca Mountain, Nevada. This task includes four subtasks. Subtask 2A focuses on performing thermal-hydrological modeling analyses of the drift-scale heater test to predict the temperature and saturation distribution in the rock during the heating and cooling phases of the test. The outcome of this predictive analysis forms the basis for comparison with the measured temperatures and saturations to validate the models used to represent thermal-hydrological processes. Subtask 2B is related to modeling of rock-mass deformation at various times of the heating and cooling phases of the test. The predicted and measured displacements will be compared to validate the thermal-hydrological-mechanical models. In this subtask, the predicted temperatures from Subtask 2A will be used for analyses. Subtask 2C also involves thermal-mechanical modeling of the drift-scale heater test. This subtask, however, uses measured temperatures for analysis instead of the predicted ones from Subtask 2A. Subtask 2D includes modeling of thermal-hydrological-chemical processes associated with the drift-scale heater test.

The NRC/CNWRA modeling effort for Task 2 focuses on Subtasks 2A and 2C. Analyses for Subtask 2A have been completed. A report documenting analyses results for the progress on Subtask 2A was submitted to NRC and the Secretariat of the DECOVALEX III project in May 2001 (Green et al., 2001) and is currently being reviewed by the international research teams of the DECOVALEX III project.

On completion of the thermal-hydrological analysis of the drift-scale heater test in Subtask 2A in May 2001, the preliminary analysis for Subtask 2C began late in fiscal year 2001. This report presents the technical approach adopted and a detailed work plan for Subtask 2C. This technical approach has been presented at the DECOVALEX III Annual Workshop in Naantali, Finland, October 22–26, 2001. Specific topics addressed in this report are (i) technical approach, (ii) thermal-mechanical models to be used in the analysis, (iii) work plan, and (iv) schedule. The modeling studies proposed in this report are an important part of the process for assessing and developing confidence on both U.S. Department of Energy models and those models used by NRC to independently evaluate the safety case for the proposed geologic repository at Yucca Mountain.

2 TECHNICAL APPROACH

Two fundamentally different approaches are available for numerically modeling rock mass below ground. The first modeling approach assumes that a rock mass behaves as a continuous material often called the continuum approach. The presence of discontinuities may be accounted for by making various assumptions. It is a common understanding that discontinuities in rock media make rock media softer and weaker. A softer rock deforms more and can be reflected by systematically adjusting rock stiffness parameters. A weaker rock can be modeled by reducing the rock strength parameters. The finite element technique is well-suited for modeling this type of material. The second concept for modeling rock mass is to include discontinuities explicitly into the model, and the discontinuities are modeled by appropriate material and strength properties. This approach is referred to as the discontinuum approach. Discrete element and discontinuous deformation analysis methods are among the techniques currently available and used for direct modeling of discontinuities in the numerical analysis.

In the exercise of Subtask 2C for thermal-mechanical modeling of the drift-scale heater test, both finite element and discontinuous deformation analysis methods will be used. The finite element code to be used for conducting the continuum analysis is ABAQUS. ABAQUS is a commercially available code that has been widely used for modeling geological media. ABAQUS is controlled by the CNWRA software quality assurance procedure (Technical Operating Procedure–018, Development and Control of Scientific and Engineering Software). For conducting discontinuum analysis, a computer code called dda_ct2 will be used.

Sin-Min Hsiung 11/30/2001

Green, R.T, M.E. Hill, and S.L. Painter. "Progress Report for DECOVALEX III Task 2A: Numerical Simulation of the Drift-Scale Heater Test at Yucca Mountain." San Antonio, Texas: CNWRA. 2001. Sin-Min Hsiung 11/30/2001

3 WORK PLAN FOR THERMAL-MECHANICAL MODELING OF DRIFT-SCALE HEATER TEST

The drift-scale heater test facility is located in the Topopah Spring middle nonlithophysal zone (CRWMS M&O, 1997a). The Topopah Spring middle nonlithophysal zone is approximately 30–40 m [98.4–131.2 ft] thick at the location of the drift-scale test area. This zone is overlain by the Topopah Spring upper lithophysal and underlain by the Topopah Spring lower lithophysal zones. Figure 3-1 shows a generalized stratigraphic column including expanded lithologic information from ground surface to below Calico Hills formation for the proposed repository.[1]

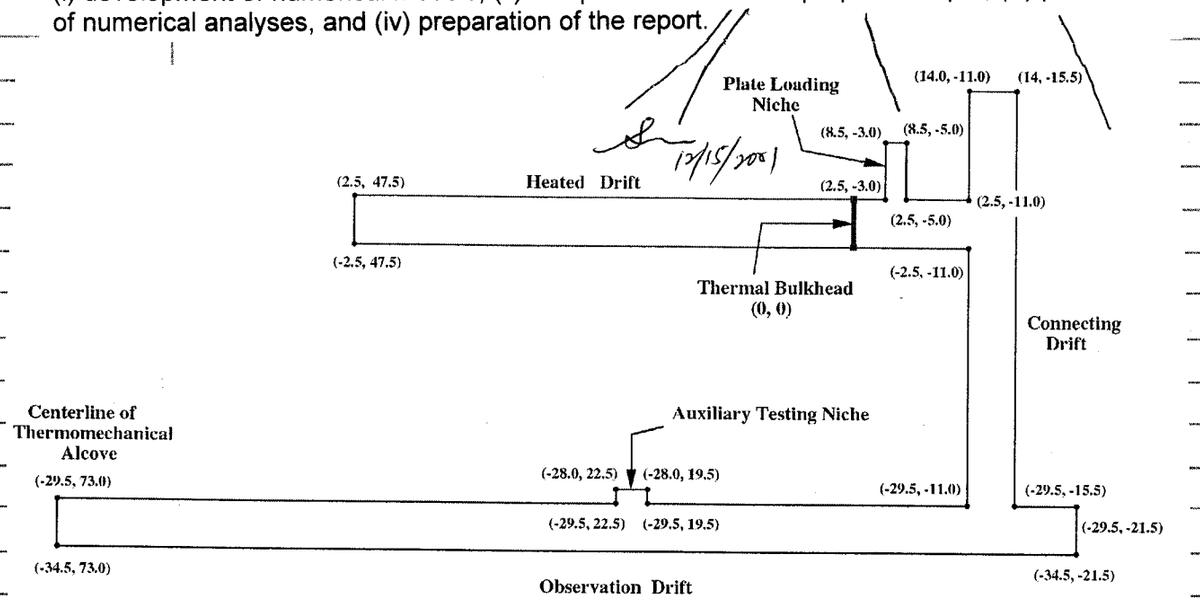
The drift-scale heater test block was characterized prior to the onset of heating. The characterization includes geologic mapping, local geology, rock-mass classification, and some geotechnical data. Figure 3-2 shows a plane view schematic of the drift-scale heater test region and associated access. The heater drift is about 5 m [16.4 ft] in diameter and 47.5-m [155.8-ft] long and is closed at the east end by a thermal bulkhead. Approximately 12 m [139.4 ft] of the heated drift, from the west end, is lined with a cast-in-place concrete liner. A concrete invert was poured along the entire floor of the heated drift. Eight 20-mm [0.79-in] thermal expansion joints were cast into the invert at a nominal spacing of 6 m [19.7 ft]. Thermal sources for the heated drift consist of 9 canister heaters, placed end to end on the concrete inverts of the heated drift and 50 wing heaters (25 on either side) placed in horizontal boreholes drilled into the sidewalls of the heated drift about 0.25 m [0.8 ft] below the springline. Locations of the wing heaters around the heated drift can be found in a report prepared by CRWMS M&O (1997b). The wing heaters are spaced 1.83 m [6 ft] apart. Each wing heater has two segments {5 m long [16.4 ft]} with a larger power output from the outer segment. The inner wing heater segment is separated from the heater drift by 1.5 m [4.9 ft].

[1]This information is provided by the technical monitor research team for Task 2 of the DECOVALEX III project.

Sin-Min Hsiung 12/15/2001

Temperatures are measured at approximately 2,662 locations within the drift-scale heater test block. Temperature measurements in the rock and wing heaters were used in developing the temperature distributions in the rock at intervals of 2 d. The dimensions of the block used for the development of temperature distribution are 70 m [229.7 ft] wide in the *x*-direction, 60 m [196.8 ft] long in the *y*-direction (axis along the heated drift), and 70 m [229.7 ft] high in the *z*-direction. The center of the block is located at the center of the heated drift approximately 25 m [82 ft] from the thermal bulkhead. Ambient temperatures {approximately 24 °C [75 °F]} were applied to the boundaries of the block to develop temperature distribution. The temperature data were generated at intervals of 1 m [3.3 ft] along all three directions. In the process of developing temperature data, if a resulting temperature value was less than 20 °C [68 °F], a value of 20 °C [68 °F] was assigned. This value is slightly lower than the ambient temperature. Figure 3-3 shows the contours of the temperature distribution on a vertical cross section at 23 m [75.5 ft] from the thermal bulkhead of the heated drift after 1,002 d of heating.

The temperature distribution data generated from the measured temperatures are provided to the research teams involved in the modeling effort of Task 2C of the DECOVALEX III project. The NRC/CNWRA research team effort to accomplish Task 2C includes four major activities: (i) development of numerical models, (ii) compilation of material properties input, (iii) production of numerical analyses, and (iv) preparation of the report.



Notes:
1. X and Y coordinates are given in meters.
2. Z coordinates (vertical) are not included.
3. Only key locations are provided.
4. Coordinates are based on design with the following as-builts tolerances:
   Horizontal alignment = - 0 to + 0.5m
   All other = - 0 to + 0.3m

Fig. 3-2

Sin-Min Hsiung 12/15/2001

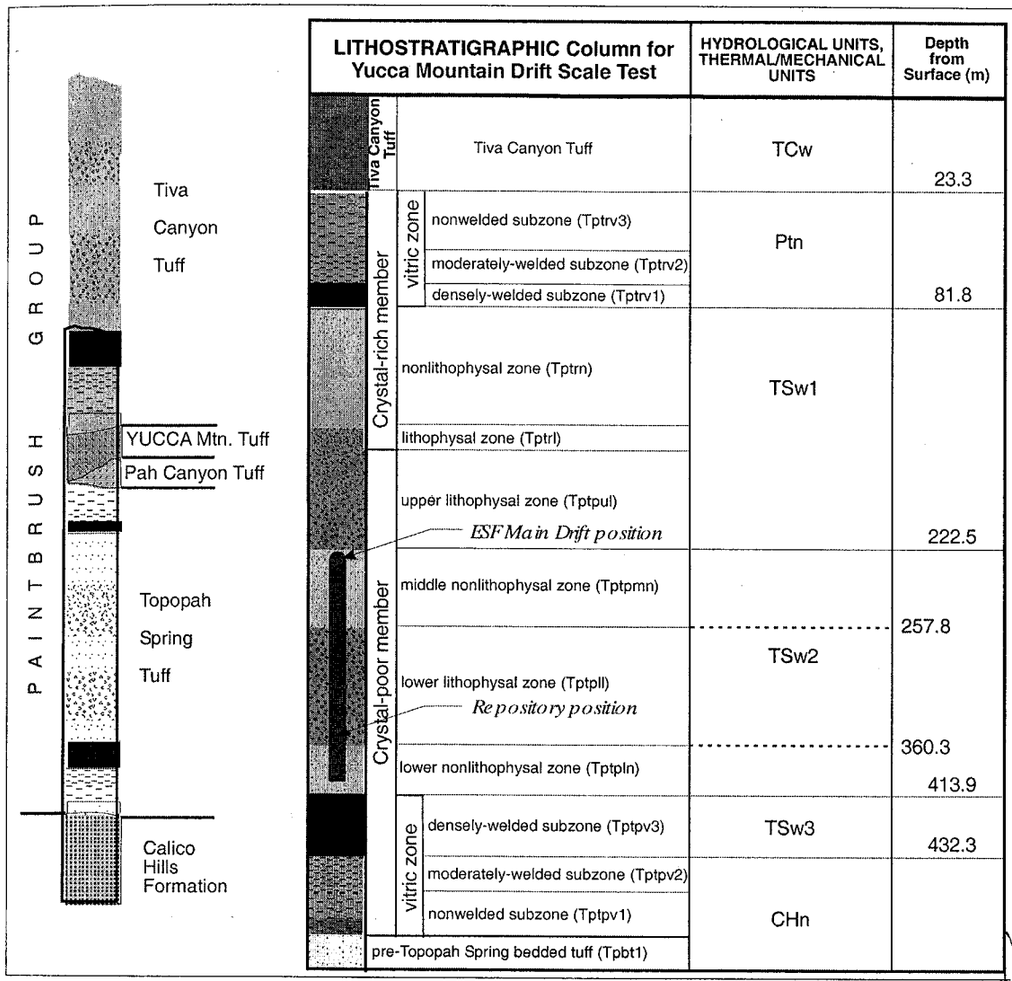


Fig. 3-1

Sin-Min Hsiung 12/15/2001

Color	Range Beg.	Range End
	0.00	40.00
	40.00	60.00
	60.00	80.00
	80.00	100.00
	100.00	120.00
	120.00	140.00
	140.00	160.00
	160.00	180.00
	180.00	200.00
	200.00	220.00
	220.00	240.00
	240.00	260.00
	260.00	280.00
	280.00	300.00
	300.00	320.00
	320.00	400.00

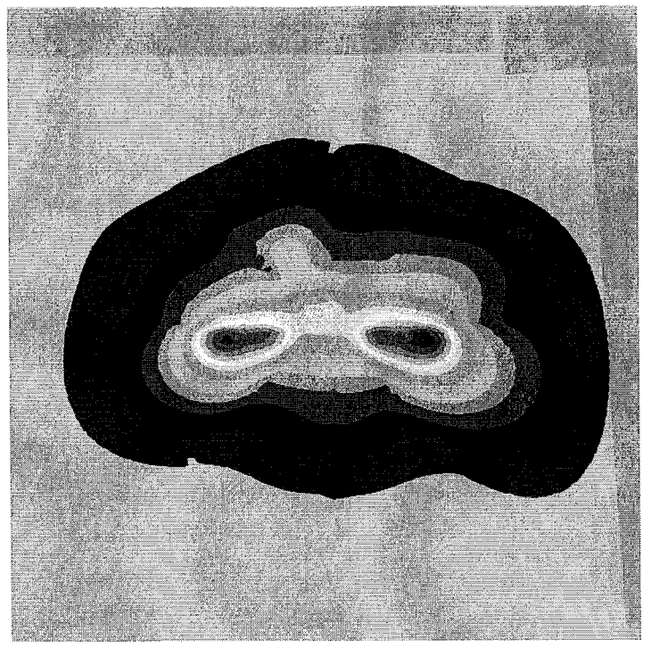


Figure 3-3. Temperature Contour, 23 m [75.5 ft] from the Thermal Bulkhead after 1,002 d of Heating

both continuum and discontinuum analyses will be conducted in this study to predict thermal effects on rock-mass deformation. The two-dimensional numerical models will simulate a plane strain condition. A vertical cross section, 21 m [68.9 ft] from the thermal bulkhead, will be taken to develop the numerical model. This cross section coincides with four multiple position extensometers used for displacement measurements. The models used by both approaches will have the same dimension {200 m [656 ft] wide and tall} with a 5-m [16.4-ft] diameter drift in the middle of the model domain. Fixed horizontal displacement boundaries will be applied to the sides and a fixed vertical displacement boundary to the bottom of the model. A constant stress boundary will be applied to the top of the model to simulate the remaining overburden. Initial stresses consistent with overburden depth will be applied as an initial condition. The thermal load will be calculated based on the temperature distribution at 21 m [68.9 ft] from the thermal bulkhead coinciding with the location of numerical models.

The height of the model domain has been subsequently extended to ground surface based on Fig. 3-1.

Sin Min Hsiung 12/17/2001

The four extensometers at the cross section are MPBX-7, MPBX-8, MPBX-9, and MPBX-10, respectively. The relative position of anchors relative to the tunnel surface is

MPBX location.txt

154	ESF-HD-MPBX-7	MPBX - Rock Mass Displacement				
1.285	21.020	2.186	8.867	21.011	15.099	59.6
155	ESF-HD-MPBX-8	MPBX - Rock Mass Displacement				
-1.317	21.006	2.270	-9.016	20.919	15.268	59.4
156	ESF-HD-MPBX-9	MPBX - Rock Mass Displacement				
-0.013	21.001	2.504	0.028	20.944	17.482	89.7
157	ESF-HD-MPBX-10	MPBX - Rock Mass Displacement				
-0.004	21.037	-1.624	0.031	21.014	-17.820	-89.9

154	ESF-HD-154-MPBX7-ANC-1	Mechanical - MPBX	1.842	21.019	3.135
154	ESF-HD-154-MPBX7-ANC-2	Mechanical - MPBX	2.348	21.019	3.997
154	ESF-HD-154-MPBX7-ANC-3	Mechanical - MPBX	3.336	21.018	5.678
154	ESF-HD-154-MPBX7-ANC-4	Mechanical - MPBX	8.804	21.011	14.992
155	ESF-HD-155-MPBX8-ANC-1	Mechanical - MPBX	-1.867	21.000	3.199
155	ESF-HD-155-MPBX8-ANC-2	Mechanical - MPBX	-2.372	20.994	4.051
155	ESF-HD-155-MPBX8-ANC-3	Mechanical - MPBX	-3.381	20.983	5.755
155	ESF-HD-155-MPBX8-ANC-4	Mechanical - MPBX	-8.757	20.922	14.832
156	ESF-HD-156-MPBX9-ANC-1	Mechanical - MPBX	-0.010	20.997	3.604
156	ESF-HD-156-MPBX9-ANC-2	Mechanical - MPBX	-0.007	20.993	4.604
156	ESF-HD-156-MPBX9-ANC-3	Mechanical - MPBX	-0.002	20.985	6.604
156	ESF-HD-156-MPBX9-ANC-4	Mechanical - MPBX	0.028	20.944	17.354
157	ESF-HD-157-MPBX10-ANC-1	Mechanical - MPBX	0.000	21.034	-3.684
157	ESF-HD-157-MPBX10-ANC-2	Mechanical - MPBX	0.003	21.033	-4.674
157	ESF-HD-157-MPBX10-ANC-3	Mechanical - MPBX	0.007	21.030	-6.674
157	ESF-HD-157-MPBX10-ANC-4	Mechanical - MPBX	0.029	21.015	-16.924

Data extracted from
CRWMS M&O. 1998. Drift Scale Test As-Built Report. BABB000000-01717-5700-00003. Rev. 1. Las Vegas: NV. (Appendix)

These anchors are explicitly modeled in the analysis so that the displacements at these anchors can be compared directly with the measured data.

For these extensometers, two are vertically oriented (one in the crown and one in the invert), and the remaining two are inclined at approximately 30° to the vertical extensometer on either side of the vertical extensometer.

Because of the size of the model domain currently being considered, the numerical models will include three generalized lithologic zones: Topopah Spring upper lithophysal zone on the top, Topopah Spring middle nonlithophysal zone in the middle, and Topopah Spring lower lithophysal zone on the bottom. The development of finite element models for continuum analysis to include these three zones should be relatively straightforward because the model geometry to be developed is relatively simple. The development of finite element models in this activity will focus on the close vicinity of the heated drift with relatively smaller size of elements. The size of the elements will be scaled up gradually because these elements are located farther from the area of interest. Effort may also be made in this activity to ensure the points at which displacement predictions are required coincide with the physical nodes of the finite elements so that displacements for these points can be obtained directly from the modeling results without interpolation.

Development of numerical models for DDA for this study will require joint geometrical information for the three lithologic zones mentioned in the previous paragraph. The joint geometrical information required includes orientation, spacing, and length as a minimum. Because the joint information specific to the heated drift is not readily available, the joint information developed from the geological mapping data collected from the Exploratory Studies Facility will be used in this activity as a basis to generate DDA block models. The joint geometrical information for the three lithologic zones presented in Tables 3-1 through 3-3 is estimated from two reports prepared by CRWMS M&O (2000a,b).

Table 3-1. Joint Information for Topopah Spring Middle Nonlithophysal Zone

Joint Set Number	Dip Angle, Degrees	Dip Direction, Degrees	Mean Length, m [ft]	Mean Bridge Length, m [ft]	Mean Spacing, m [ft]
1	84	221	2.54 [8.33]	0.4 [1.31]	0.60 [1.97]
2	83	299	2.71 [8.89]	0.4 [1.31]	1.92 [6.30]
3	9	59	3.23 [10.60]	0.4 [1.31]	0.56 [1.84]

Table 3-3. Joint Information for Topopah Spring Upper Lithophysal Zone

Joint Set Number	Dip Angle, Degrees	Dip Direction, Degrees	Mean Length, m [ft]	Mean Bridge Length, m [ft]	Mean Spacing, m [ft]
1	82	288	3.29 [10.79]	-1.0 [-3.28]	3.76 [12.34]
2	82	229	2.88 [9.45]	-1.0 [-3.28]	3.76 [12.34]
3	14	40	5.16 [16.93]	-1.0 [-3.28]	3.21 [10.53]

Table 3-2. Joint Information for Topopah Spring Lower Lithophysal Zone

Joint Set Number	Dip Angle, Degrees	Dip Direction, Degrees	Mean Length, m [ft]	Mean Bridge Length, m [ft]	Mean Spacing, m [ft]
1	82	235	4.56 [14.96]	-1.0 [-3.28]	3.47 [11.38]
2	79	270	4.02 [13.19]	-1.0 [-3.28]	4.05 [13.29]
3	5	45	7.36 [24.15]	-1.0 [-3.28]	2.94 [9.65]

Note that Bridge Length in the tables are assumed values since they are not available.

———. "Drift Degradation Analysis." ANL–EBS–MD–000027. Revision 01. Las Vegas, Nevada: CRWMS M&O. 2000a.

———. "Fracture Geometry Analysis for the Stratigraphic Units of the Repository Host Horizon." ANL–EBS–GE–000006. Revision 00. Las Vegas, Nevada: CRWMS M&O. 2000b.

The joint information is scaled up for the analysis to reduce the number of blocks in the model. In this study, joint spacings and joint lengths for the three lithologic zones are scaled. The information used in the analysis are given below.

(next page)

Sui Min Hu 12/20/2001

Units	Joint Set	Dip Angle	Dip Direction	Joint Spacing, m	Joint Length, m
Tptpmn	1	84	221	3.00	10.16
	2	83	299	9.60	10.84
	3	9	59	2.80	12.92
Tptpul	1	82	288	7.52	19.74
	2	82	229	7.52	17.28
	3	14	40	6.42	30.96
Tptpll	1	82	235	6.94	27.36
	2	79	270	8.10	24.12
	3	5	45	5.88	44.16

3.2 Compilation of Material Properties Input

Material properties needed for both continuum and discontinuum analyses to be performed in this study include (i) rock-mass deformation modulus, (ii) Poisson's ratio, and (iii) strength properties. Additional material properties necessary for discontinuum analysis include (i) joint normal and shear stiffnesses and (ii) joint cohesion and friction angle. The objectives of this activity involve collecting and compiling input information for modeling use.

3.2.1 *In-Situ* Deformation Moduli

The *in-situ* deformation modulus of a rock mass is an important parameter in any form of numerical analysis and in the interpretation of monitored deformation around the heated drift. The intact rock Young's moduli, along with Poisson's ratios and bulk densities, for the three lithologic zones are listed in Table 3-7. As discussed in Section 1, the presence of discontinuities in rock media tends to soften and weaken a rock medium. Therefore, the deformation modulus of a rock mass could be substantially different from that of the intact rocks that form the rock mass. The extent of difference in deformation modules depends on the intensity and the surface properties of joints present in the host rock. A reasonable approach to determine rock-mass deformation modulus is to conduct tests in the field. This approach, however, is often difficult to perform and is expensive. Attempts have been made to develop methods for estimating its value, based on rock-mass classifications. The two most widely used rock-mass classifications are the RMR (Bieniawski, 1976, 1989) and the Q (Barton et al., 1974). Both methods rely heavily on joint information.

Table 3-7. Rock Block Material Properties [CRWMS M&O (1997c)]

	Upper Lithophysal Zone	Middle Nonlithophysal Zone	Lower Lithophysal Zone
Bulk Density, kg/m^3 [lb/ft^3]	2,160±80 [134.8±5.0]	2,250±70 [140.5±4.4]	2,250±60 [140.5±3.7]
Young's Modulus, Gpa [10^6 psi]	20.36±6.75 [2.95±0.98]	32.93±5.47 [4.77±0.79]	27.54±7.49 [3.99±1.09]
Poisson's Ratio	0.23±0.07	0.21±0.03	0.21±0.06

It should be noted, however, large uncertainties are associated with the characterization of joint intensity and joint properties. Consequently, determination of rock-mass deformation modulus by taking into consideration the presence of joints will similarly involve large uncertainties. As part of this subactivity, both rock-mass classification methods will be used to derive rock-mass deformation modulus. Attempts will be made to quantify the uncertainties. The uncertainties may be evaluated to estimate their effects on displacement prediction in the Task 2C, activity (iii): Numerical Analyses.

It is recognized that the deformation modulus for rock blocks in a discontinuum analysis may be different from that for elements in a continuum analysis because at least a portion of the joints in the rock mass is modeled explicitly. It is reasonable not to include the effects of these

explicitly modeled joints from the estimation of the rock-mass deformation modulus. Otherwise, these effects tend to be double counted, resulting in an overprediction of displacements. No acceptable procedure is available for this exercise. Consequently, overprediction may be an acceptable alternative (i.e., use the rock-mass deformation modulus in the discontinuum analysis).

3.2.2 Strength Properties

Another important parameter to support numerical analysis of rock-mass behavior is strength properties. These properties are equally if not more difficult to determine than the rock-mass deformation modulus. In this study, the Hoek-Brown failure criterion will be used in the continuum analysis to judge failure of finite elements. The procedure outlined by Hoek and Brown (1997) will be followed to estimate the related strength properties. Previous work conducted by the CNWRA on rock-mass strength properties for the Topopah Spring middle nonlithophysal zone (Ofoegbu, 2000) will be considered in this subactivity.

While applying this failure criterion, particular attention will be made to its applicability to the rock masses being considered. Hoek, et al. (1995) indicate,"the Hoek-Brown failure criterion is only applicable to intact rock or to heavily jointed rock masses which [sic] can be considered homogeneous and isotropic." In other words, the use of this failure criterion is not appropriate for conditions in which an individual joint set has a dominant influence on the behavior of the rock mass even in extreme cases (Hoek, et al., 1995).

For discontinuum analysis, the Mohr-Coulomb failure criterion may be used to assess potential failure of rock blocks and the corresponding strength parameters will be derived, in this subactivity, from the Hoek-Brown failure criterion (Hoek and Brown, 1997). As an alternative, the Hoek-Brown failure criterion could be coded into the dda_ct2 computer program and used as the failure criterion to assess block failure. This option is being considered.

A concern similar to that discussed in the last paragraph of Section 3.2.1 is equally applicable to the determination of strength properties for the blocks of discontinuum analysis. Unless a viable alternative can be found, rock-mass strength properties will be assumed for the blocks.

3.2.3 Joint Stiffness and Strength Properties

The primary objective of this subactivity is to collect and develop, if necessary, joint stiffness and strength properties for the three lithologic zones for discontinuum modeling. Limited test results of joints related to the Topopah Spring rock unit are presented in a report prepared by CRWMS M&O (1997c). These joint properties along with the test results on the Apache Leap tuff joints, obtained from an early CNWRA study sponsored by the NRC Office of Research (Hsiung, et al., 1994), will be analyzed together in this subactivity to develop reasonable joint properties for the study.

3.3 Numerical Modeling

Thermal-mechanical modeling of the drift-scale heater test for Subtask 2C of the DECOVALEX III project will be performed using both the finite element and DDA methods. Planned work related to this activity is discussed in the following subsections.

3.3.1 Continuum Analyses

In this subactivity, the finite element computer code ABAQUS will be used to conduct the continuum analysis. The material and strength properties obtained from Subsections 3.2.1 and 3.2.2, along with the coefficients of thermal expansion and the temperature distribution data, will be used as input for the analysis.

The analysis will be performed at three discrete heating times: 364, 730, and 1,002 d. At each discrete time, the state of stresses for the overall model and the displacements at predetermined locations will be assessed. Failure of finite elements is allowed, and the failed elements are assumed to exhibit perfectly plastic behavior.

If time permits, the effects of uncertainties associated with the rock-mass deformation modulus and the strength properties will be evaluated and quantified in the analysis. The primary focus will be the effects of the property uncertainties related to the Topopah Spring middle nonlithophysal zone.

3.3.2 Discontinuum Analyses

In this subactivity, the computer code dda_ct2 will be used to conduct the discontinuum analysis. The material and strength properties obtained from Subsections 3.2.1, 3.2.2, and 3.2.3, along with the coefficients of thermal expansion and the temperature distribution data, will be used as input for the analysis.

Similar to the continuum analysis, the DDA will be performed at three discrete heating times: 364, 730, and 1,002 d. At each discrete time, the state of stresses for the overall model and the displacements at specific points will be assessed. The dda_ct2 computer code to be used in this study permits rock blocks to break if the failure condition is met. As discussed previously, a Mohr-Coulomb failure criterion is available for judging block failure. When the block is failed, it will be broken into two or four subblocks, depending on the type of failure experienced. For shear failure, the block will break into four pieces and for tensile failure, two pieces .

Furthermore, to account for variations and uncertainties associated with the joint geometrical information, a Monte Carlo technique will be adopted to generate sample joint spacings and joint lengths. Note that each sample generated is an equally likely realization of joints that honor the information used. In generating these realizations, the joint spacing, length, and bridge length will be assumed uniformly distributed and varied ~~±30~~ ± 25 percent around the mean values of the respective parameters. The fracture geometry analysis report prepared by CRWMS M&O (2000b) shows that the joint spacings and trace lengths for the four Topopah Spring lithologic zones are mostly lognormally distributed, and some are exponentially distributed. Depending on the lower and upper limits used to constrain sampling, the assumption of uniform distribution used in this analysis could potentially underestimate the maximum block size but overestimate the number of relatively large blocks available. This potential overestimation or underestimation may not be of significance in that the joint spacings and joint lengths are scaled up already in this study to reduce unnecessary complexity of the problem. By the same token, uncertainties in joint dip angle and dip direction will not be considered in the analyses planned in this study to avoid producing overly complicated DDA block models.

In this subactivity, sufficient DDA models will be run to capture the effects of uncertainties associated with the joint information used. The results for displacement predictions at specific positions for all DDA runs will be statistically treated and represented. A sufficient number of DDA model runs will be made until an acceptable statistic for the predicted displacement distributions is achieved. These displacement distributions will be compared to the measured displacement data.

References cited are given below.

Barton, N.R., R. Lien, and J. Lunde. "Engineering Classification of Rock Masses for the Design of Tunnel Support." *Rock Mechanics*. Vol. 6, No. 4. pp. 189–239. 1974.

Bieniawski, Z.T. "Rock Mass Classification in Rock Engineering." *Exploration for Rock Engineering*. Proceedings of the Symposium. Z.T. Bieniawski, ed. Vol. 1. pp. 97–106. Cape Town, South Africa: Balkema. 1976.

Bieniawski, Z.T. *Engineering Rock Mass Classifications*. New York City, New York: Wiley and Sons. 1989.

CRWMS M&O. "Ambient Characterization of the Drift-Scale Test Block." BADD00000–01717–5705–00001. Revision 01. Las Vegas, Nevada: CRWMS M&O. 1997a.

———. "Drift Scale Test Design and Forecast Results." BAB000000–01717–4600–00007. Revision 1. Las Vegas, Nevada: CRWMS M&O. 1997b.

———. "Yucca Mountain Site Geotechnical Report." B00000000–01717–5705–00043. Vol. 1. Revision 01. Las Vegas, Nevada: CRWMS M&O. 1997c.

———. "Drift Degradation Analysis." ANL–EBS–MD–000027. Revision 01. Las Vegas, Nevada: CRWMS M&O. 2000a.

———. "Fracture Geometry Analysis for the Stratigraphic Units of the Repository Host Horizon." ANL–EBS–GE–000006. Revision 00. Las Vegas, Nevada: CRWMS M&O. 2000b.

Green, R.T., M.E. Hill, and S.L. Painter. "Progress Report for DECOVALEX III Task 2: Numerical Simulation of the Drift-Scale Heater Test at Yucca Mountain." San Antonio, Texas: CNWRA. 2001.

Hoek, E. and E.T. Brown. "Practical Estimated of Rock Mass Strength." *International Journal of Rock Mechanics and Mining Sciences*. Vol. 34, No. 8. pp. 1,165–1,186. 1997.

Hoek, E., P.K. Kaiser, and W.F. Brawden. *Support of Underground Excavations in Hard Rock*. Rotterdam, The Netherlands: Balkema. 1995.

Hsiung, S.M., D.D. Kana, M.P. Ahola, A.H. Chowdhury, and A. Ghosh. NUREG/CR–6178. "Laboratory Characterization of Rock Joints." Washington, DC: NRC. 1994.

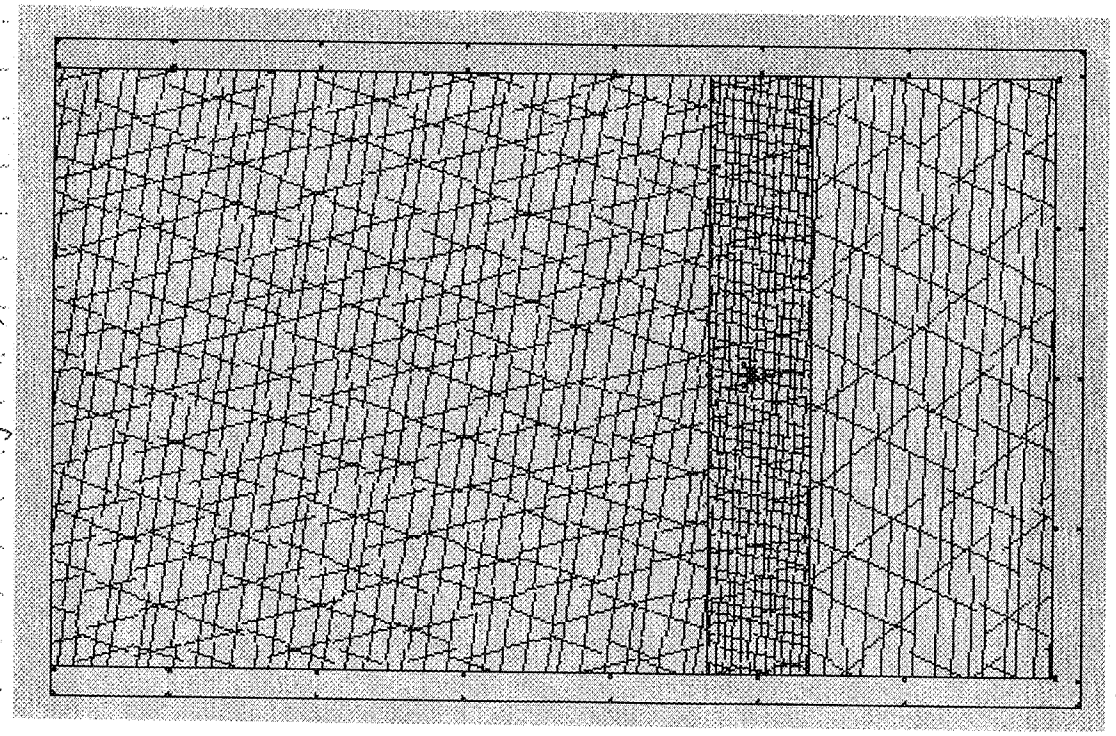
Ofoegbu, G.I. "Thermal-Mechanical Effects on Long-Term Hydrological Properties at the Proposed Yucca Mountain Nuclear Waste Repository." CNWRA 2000–03. San Antonio, Texas: CNWRA. 2000.

Sin-Min Hsiung 12/20/2001

NUMERICAL MODEL (DDA)

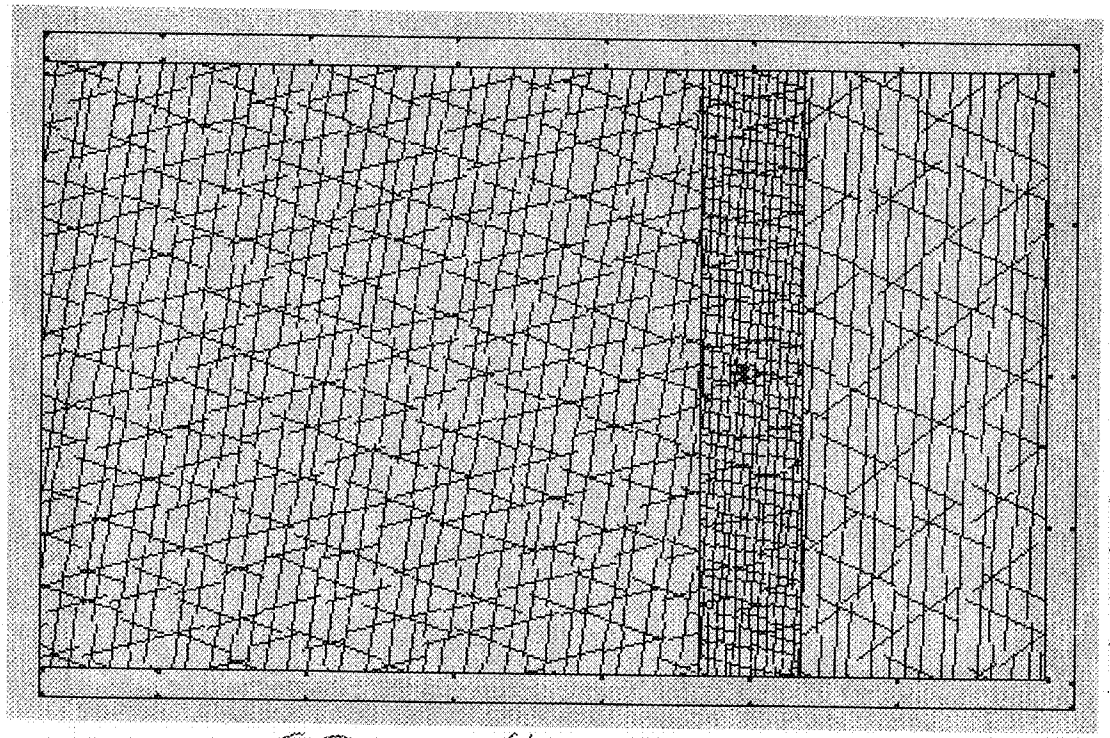
- Two-dimensional discontinuum model using DDA
  - Model dimensions
    - Cross-section Modeled: 21 m from the thermal bulkhead
    - Model top: ground surface
    - Model bottom: 100 m below heater drift horizon
    - Lateral boundaries: 100 m from center of the heater drift
    - Model interfaces between units as joints
  - Boundary Conditions
    - No displacements at bottom and lateral boundaries
    - Model interfaces between units as joints
  - Initial Conditions
    - Temperature consistent with natural geothermal gradient
    - Horizontal stresses determined by Poisson's effect

Sin-Min Hsiung 12/20/2001



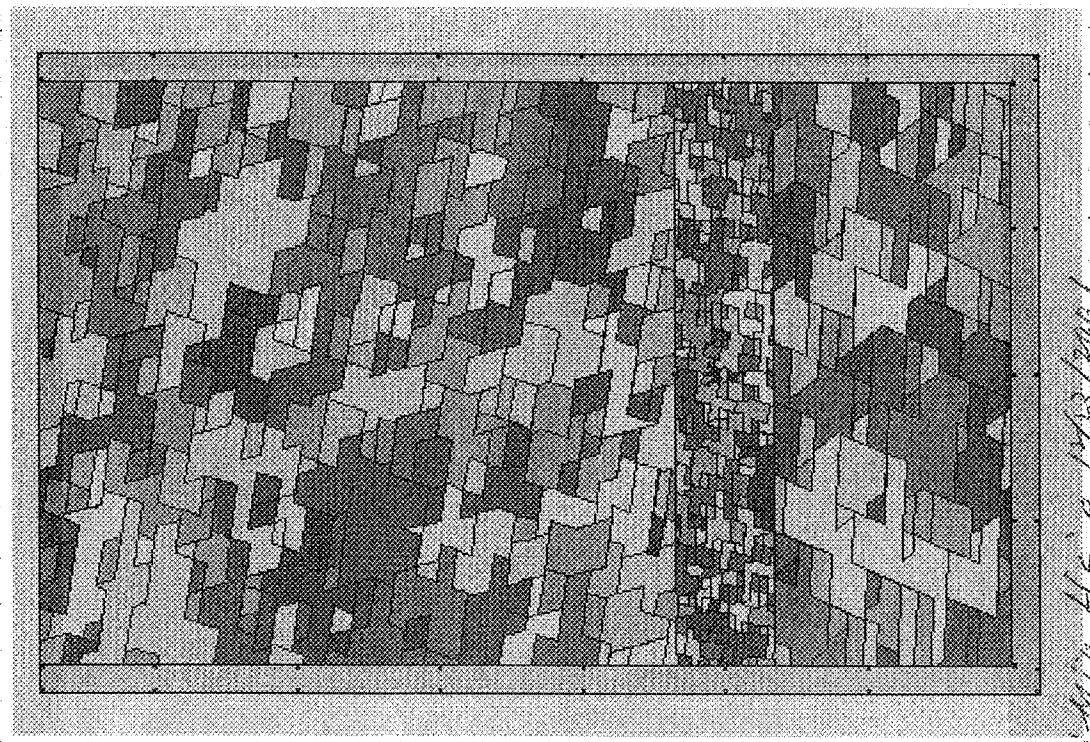
Sui-Min H'g 1/23/2001

Illustrated Tint Use Distribution



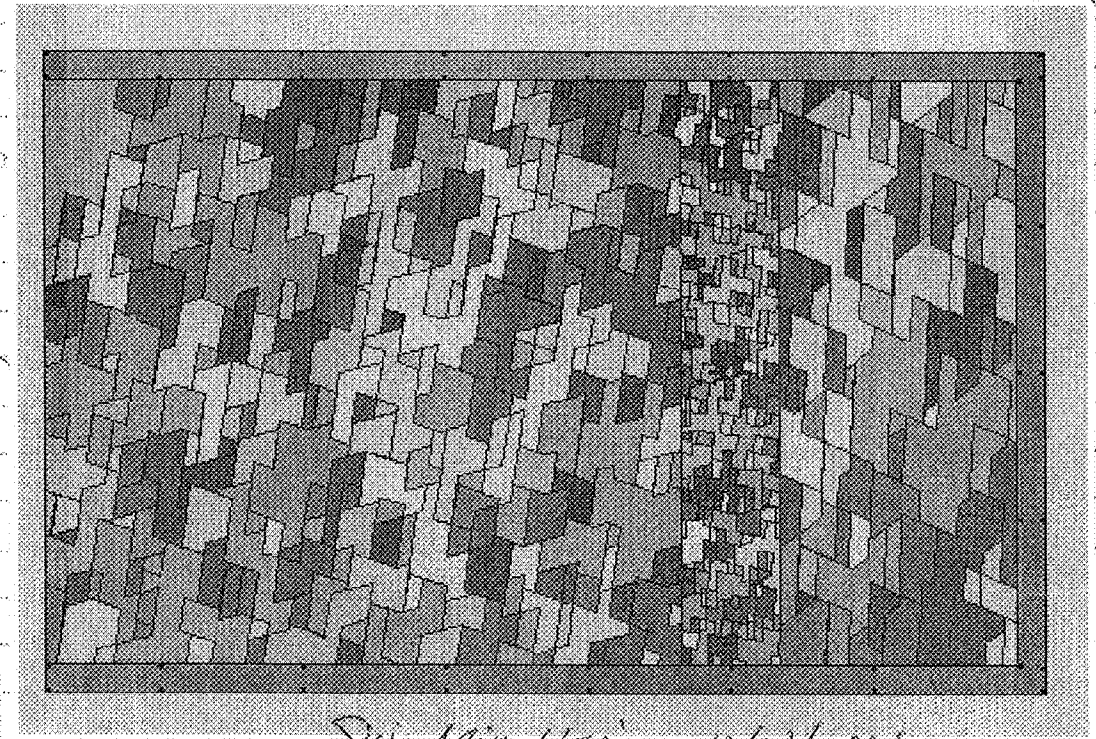
Sui-Min H'g 1/23/2001

Exit Examples of Generated



Sui-Min H'g 12/23/2001

Block formed with Tint Use mode - on p. 16



Sui-Min H'g 12/23/2001

Sui-Min H 12/24/2001 Sui-Min H 12/23/2001

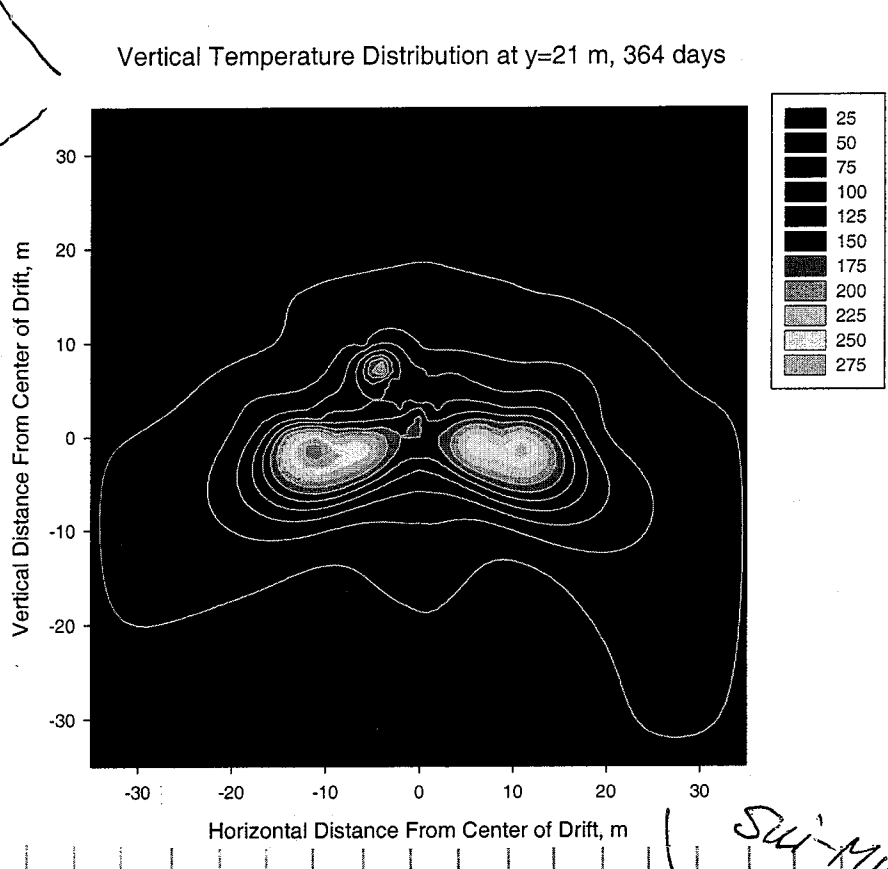
TEMPERATURE INPUT

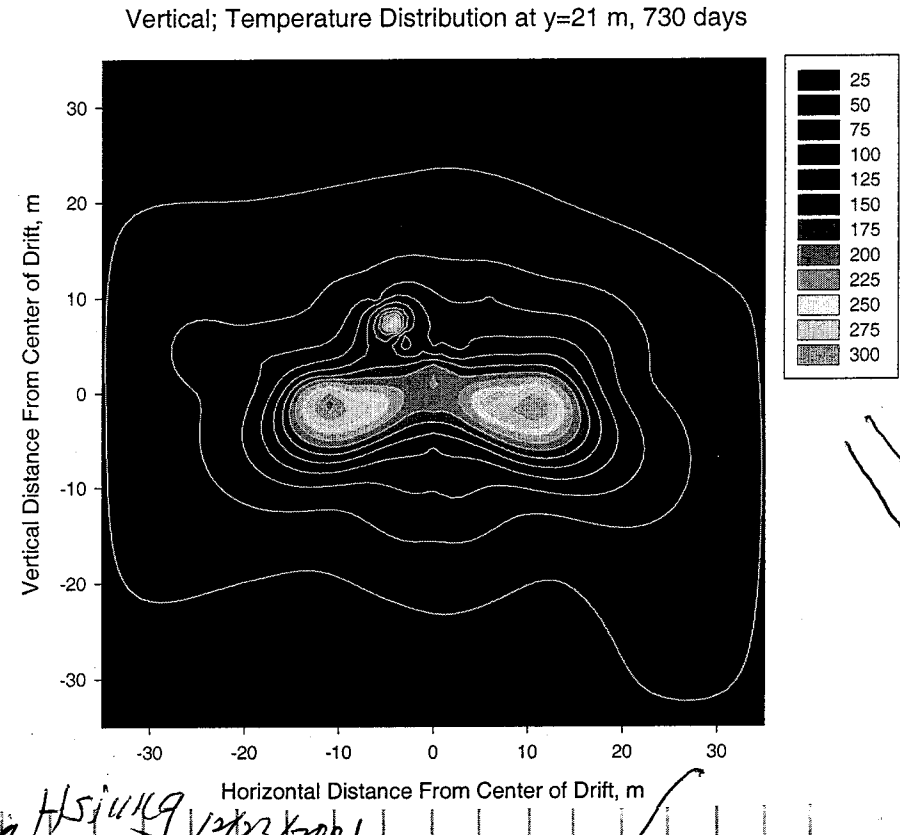
- **Temperature input using the temperature distributions provided by the task lead**
  - **Developed based on measured temperature data from approximately 2,662 locations at a 2-day interval**
- **Temperature distribution data from day 364, 730, and 1,002 at a cross-section, 21 m from the thermal bulkhead, selected for modeling input (total temperature approach)**

Sui-Min Hsiung 12/23/2001

Sui-Min H 12/23/2001

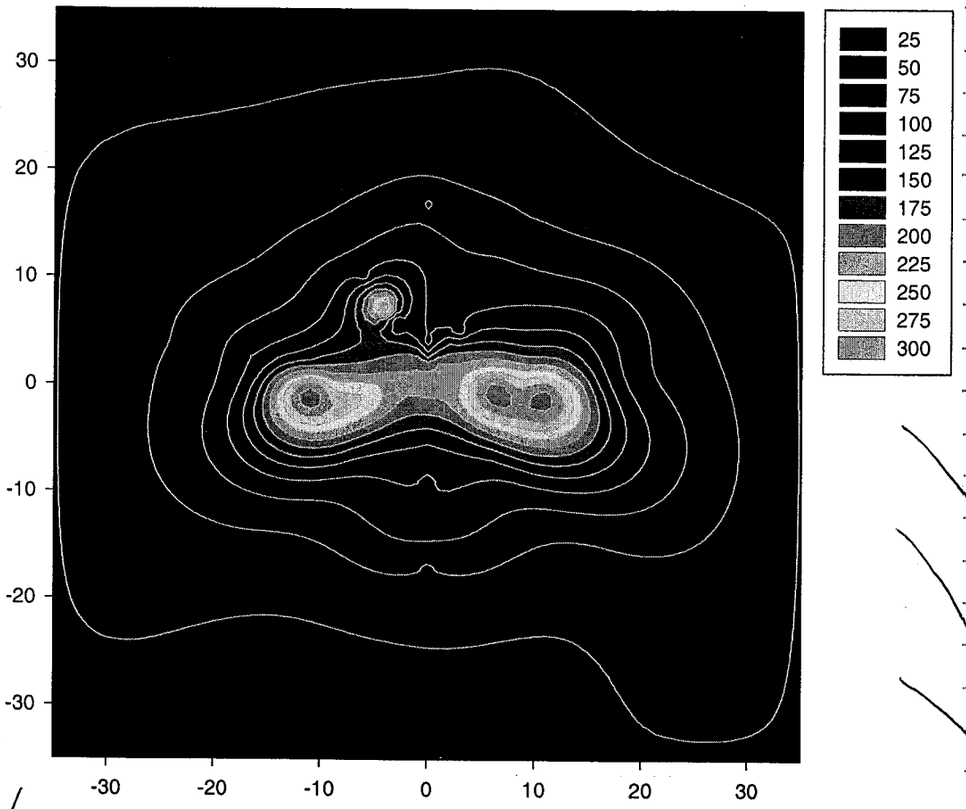
TEMPERATURE DISTRIBUTIONS AT A CROSS-SECTION 21 m FROM THE THERMAL BULKHEAD





Sui-Min Hsiung 12/23/2001

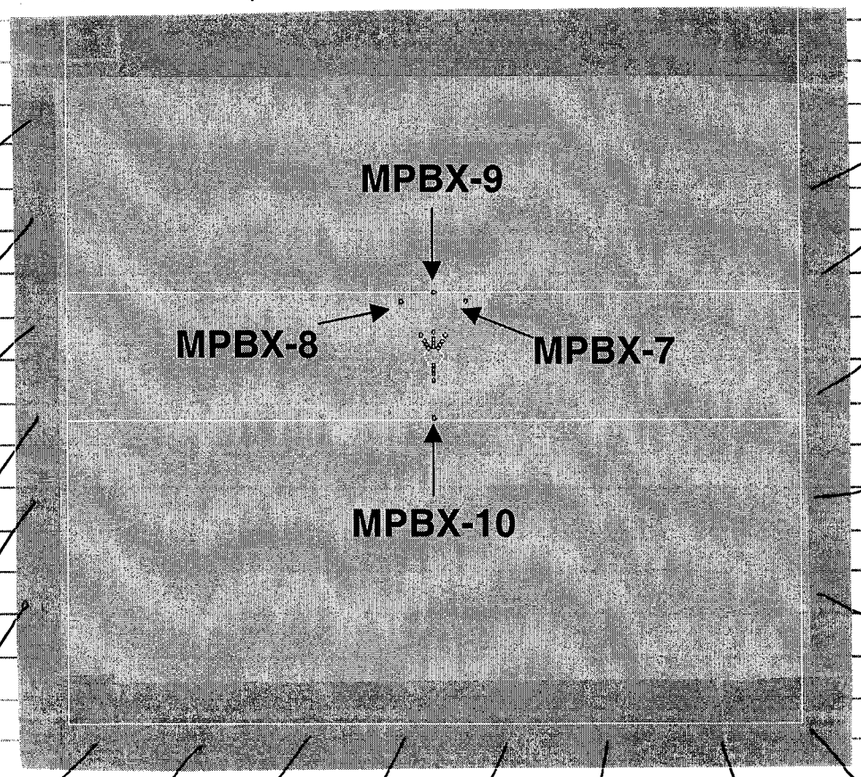
Vertival Temperature Distribution at y=21 m, 1002 days



Sui-Min Hsi
12/23/2001

1/4/2002

Relative positions of extensometer anchors with respect to drift opening



Sui-Min Hsi 1/4/2002

1/15/2002 Sui-Min Hsi

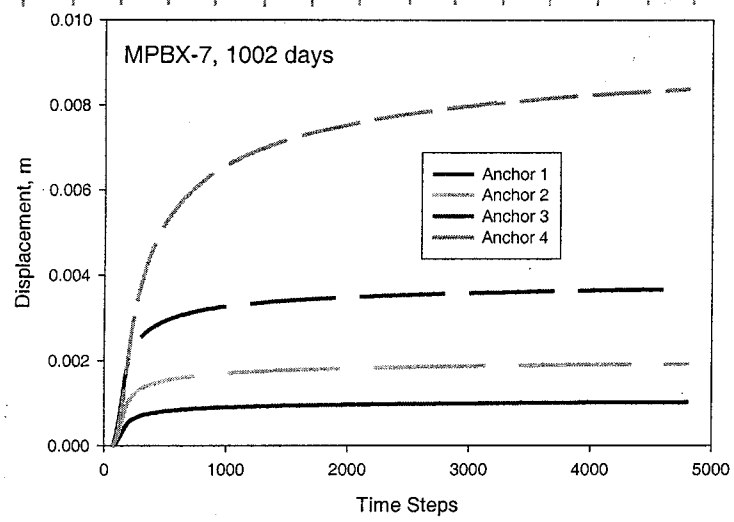
A small program (C++) is written to compute relative displacement between anchor and drift collar (drift wall boundary). This program runs in Windows NT. Compiler used is Borland CBuilder 5.0.

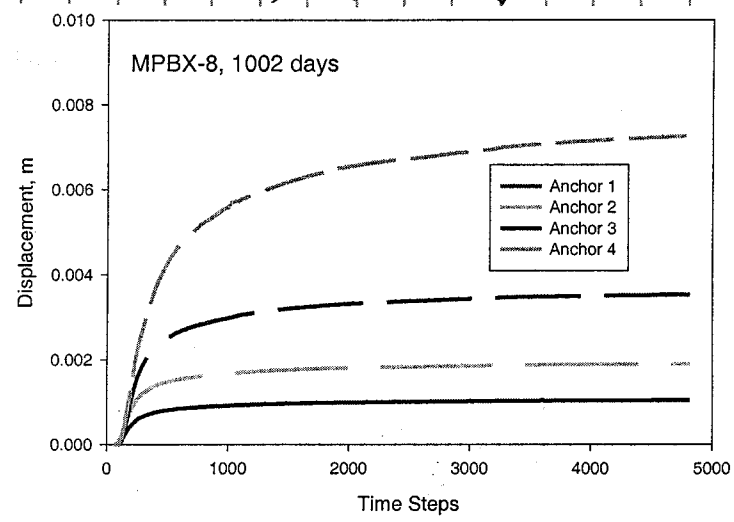
```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "disp.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
      fl0 = fopen("disp.in", "r");
      fl1 = fopen("disp.out", "w");
      int numTimeStep=0,numBolt=0,numAnchor=0,index=0;
      fscanf(fl0,"%d %d %d %d",&numTimeStep,&numBolt,&numAnchor,&index);
      disp.Null(numTimeStep+1,numBolt+1,numAnchor+1,3);
      for(int i=1; i<=numTimeStep; i++)
      {
      for(int j=1; j<=numBolt; j++)
      {
      for(int k=1; k<=numAnchor; k++)
      {
      for(int l=0; l<=2; l++)
      {
      fscanf(fl0,"%Lf",&disp(i,j,k,l));
      } // for(int l=0; l<=2; l++)
      } // for(int k=1; k<=numAnchor; k++)
      } // for(int j=1; j<=numBolt; j++)
      } // for(int i=1; i<=numTimeStep; i++)
      ext.Null(numTimeStep+1,numBolt*(numAnchor-1)+1);
      for(int i=1; i<=numTimeStep; i++)
      {
      ext(i,0)=disp(i,1,1,0);
      for(int j=1; j<=numBolt; j++)
      {
      int jj=(j-1)*(numAnchor-1);
      for(int k=1; k<=numAnchor-1; k++)
      {
      long double x1=disp(i,j,k  ,1);
```

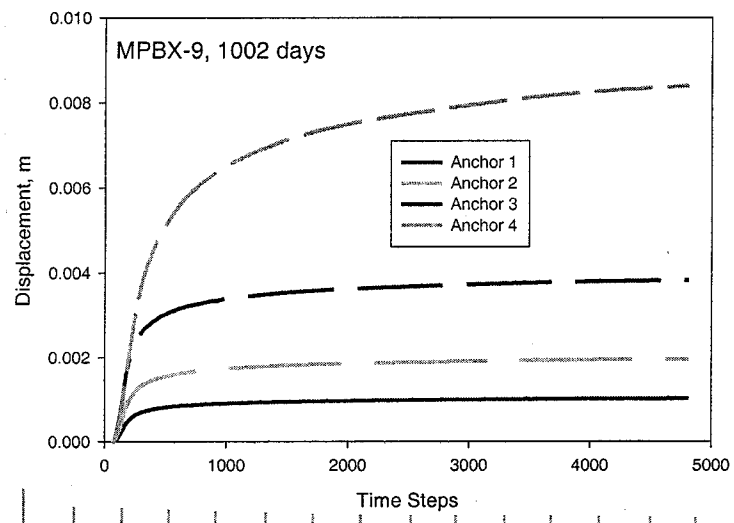


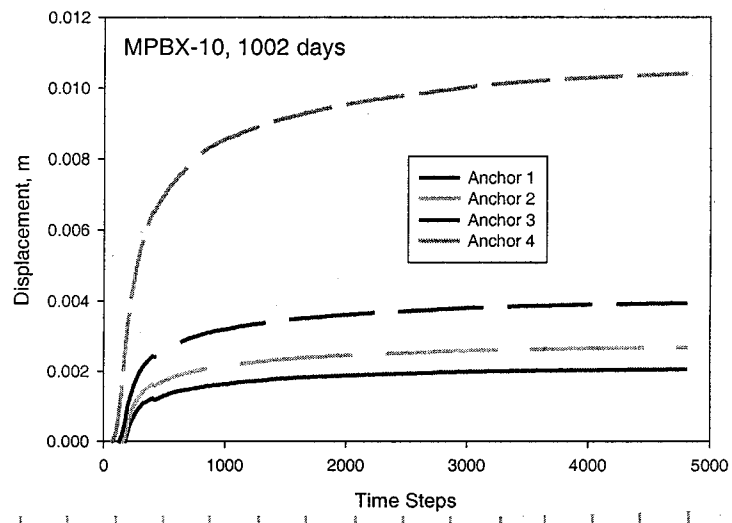
```
      long double y1=disp(i,j,k  ,2);
      long double x2=disp(i,j,k+1,1);
      long double y2=disp(i,j,k+1,2);
      ext(i,jj+k)=sqrtl((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2));
      } // for(int k=1; k<=numAnchor-1; k++)
      } // for(int j=1; j<=numBolt; j++)
      } // for(int i=1; i<=numTimeStep; i++)
      for(int i=index+1; i<=numTimeStep; i++)
      {
      for(int j=1; j<=numBolt*(numAnchor-1); j++)
      {
      ext(i,j)-=ext(index,j);
      } // for(int j=1; j<=numBolt*(numAnchor-1); j++)
      } // for(int i=index+1; i<=numTimeStep; i++)
      for(int j=1; j<=numBolt*(numAnchor-1); j++)
      {
      ext(index,j)=0.0;
      } // for(int j=1; j<=numBolt*(numAnchor-1); j++)
      for(int i=index+1; i<=numTimeStep; i++)
      {
      for(int j=1; j<=numBolt; j++)
      {
       int count=(j-1)*(numAnchor-1);
       for(int k=2; k<=numAnchor-1; k++)
       {
       ext(i,count+k)+=ext(i,count+k-1);
       } // for(int k=1; k<=numAnchor-1; k++)
      } // for(int j=1; j<=numBolt; j++)
      } // for(int i=index+1; i<=numTimeStep; i++)
      for(int i=index; i<=numTimeStep; i++)
      {
      for(int j=0; j<=numBolt*(numAnchor-1); j++)
      {
      if(j==0) fprintf(fl1,"%5d   ",(int)ext(i,j));
      else     fprintf(fl1,"%Lf   ",ext(i,j));
      } // for(int j=1; j<=numBolt*(numAnchor-1); j++)
      fprintf(fl1,"\n");
      } // for(int i=index; i<=numTimeStep; i++)
      int x8=Form1->Width/2,y8=Form1->Height/2;
      char * aa={"Task is Completed!"};
      int len=strlen(aa);
      Image1->Canvas->Brush->Color=clBtnFace;
      Image1->Canvas->Font->Size=24;
//      x8-=len*(Image1->Canvas->Font->Size/2-2); // windows 9&
      x8-=len*(Image1->Canvas->Font->Size/2-4);   // NT
      y8-=(24+Image1->Canvas->Font->Size);
      Image1->Canvas->FillRect(Image1->Canvas->ClipRect);
      Image1->Invalidate();
      Image1->Canvas->TextOut(x8,y8,aa);
//      delete [] aa;
}
//---------------------------------------------------------------------------
```

Task41_03_1002d.JNB, Anchor displacements at 1002 days of heating



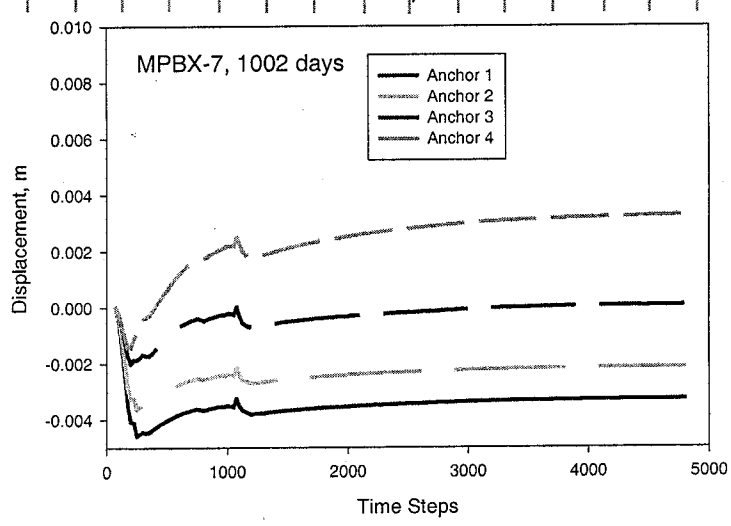


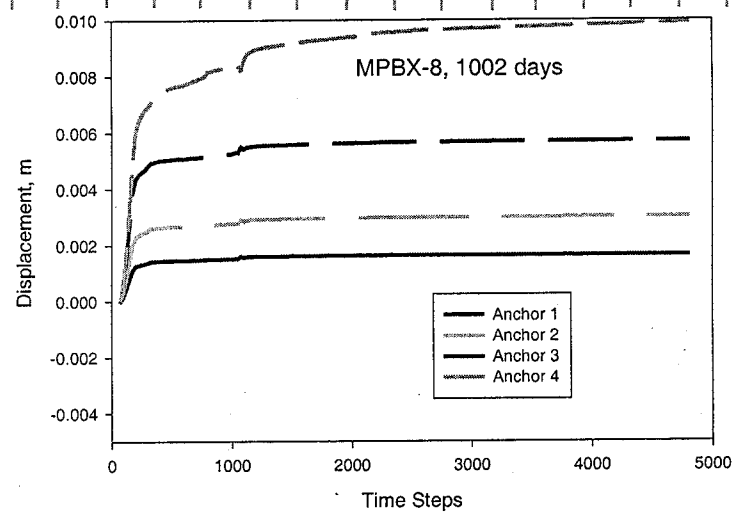


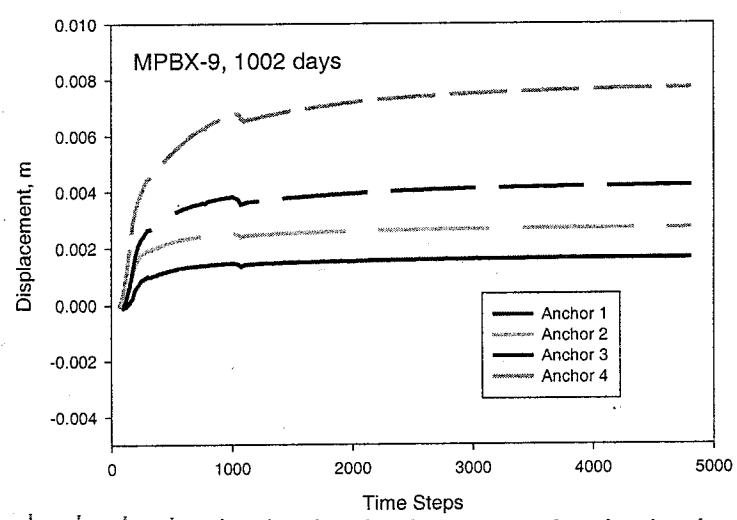


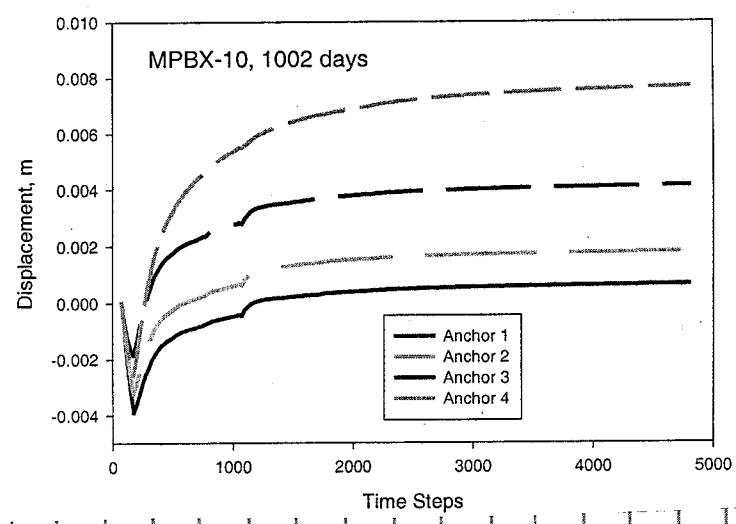
Task41_06_1002d.JNB 2/6/2002

Anchor displacement after 1002 days of heating for model 41_06



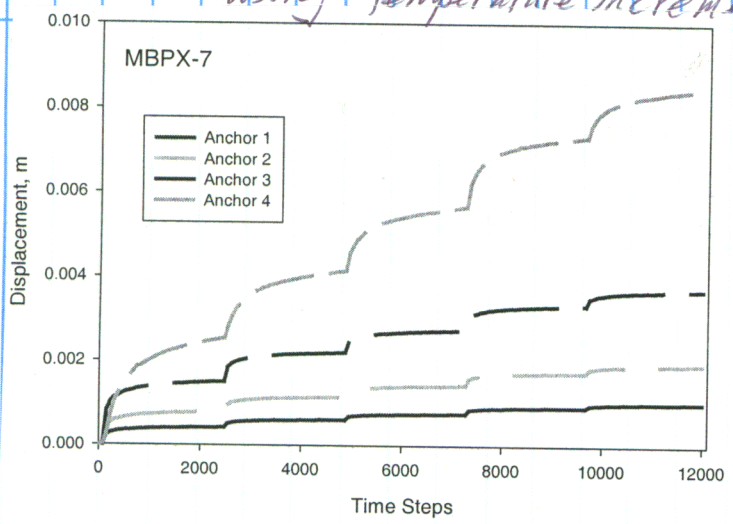


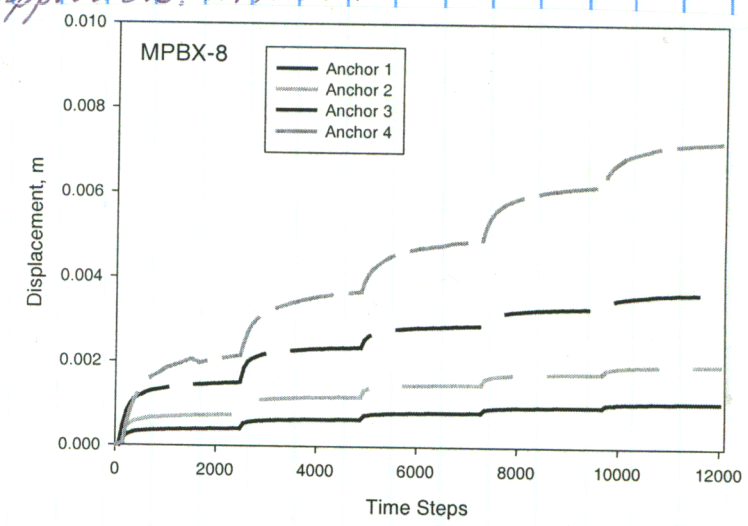


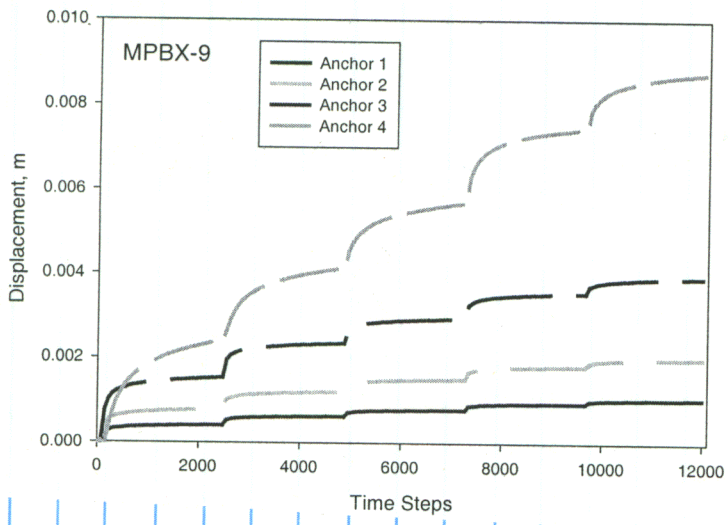


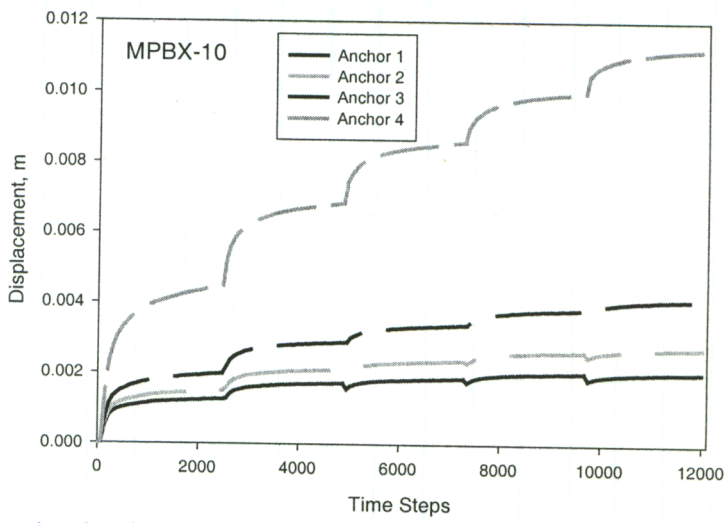
task41LC-03.JNB, displacement prediction after 1002 days of heating using temperature increment approach. DDA model is the same as that on p.24

Displacement prediction is also made using a temperature increment approach. Temperature increases every 100 days

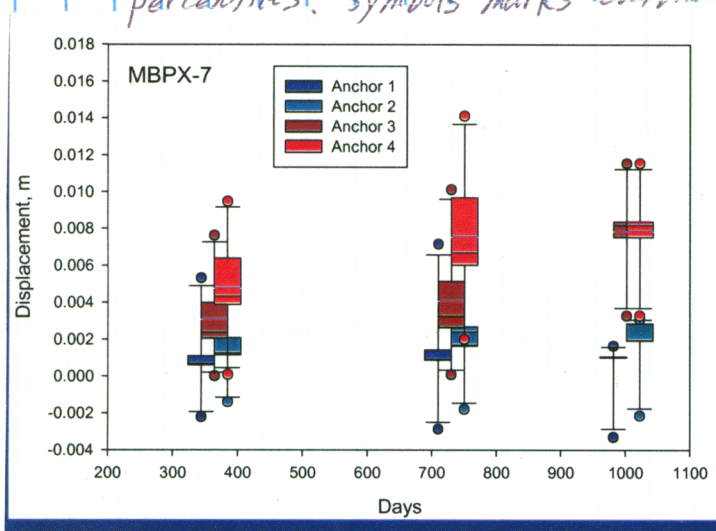


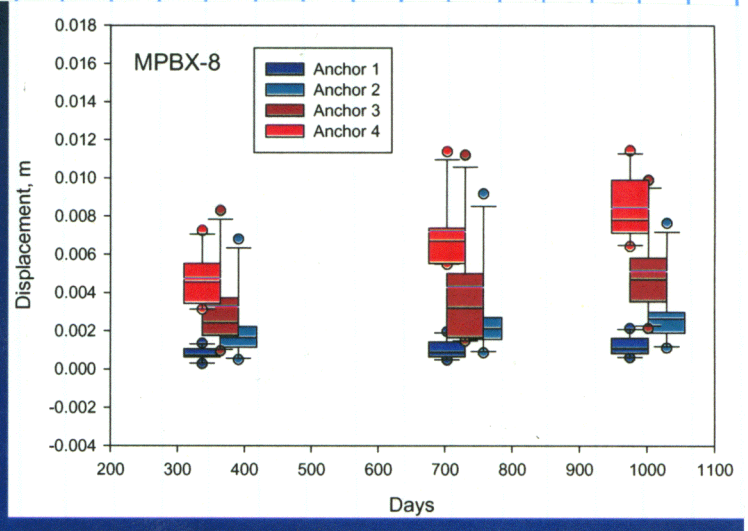


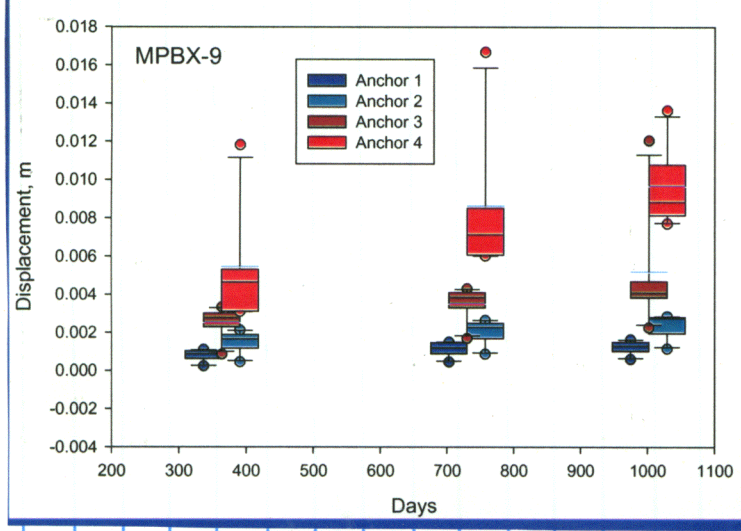


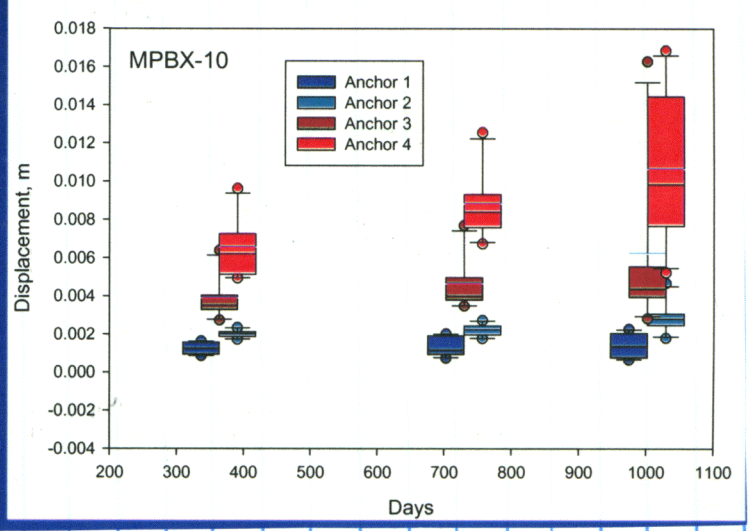


Box plots of anchor displacement prediction on 6 DDA models
Box indicates 25th and 75th percentiles. Capped Bar indicates the 10th and 9th percentiles. Symbols marks outliners. (Black & White version on next page)



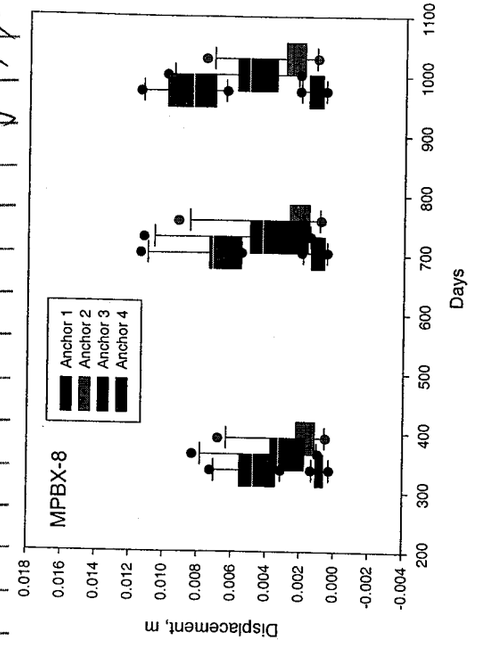


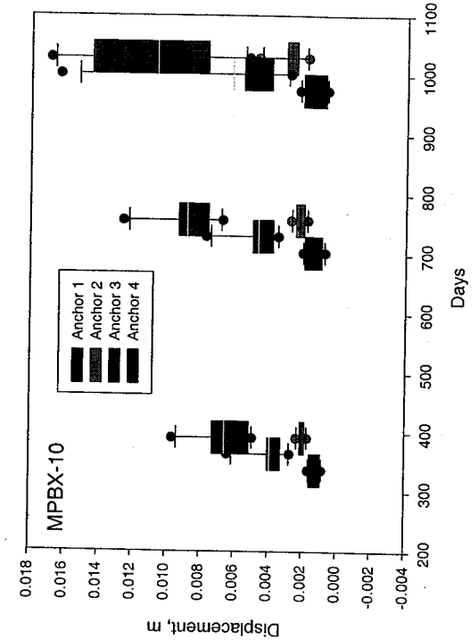




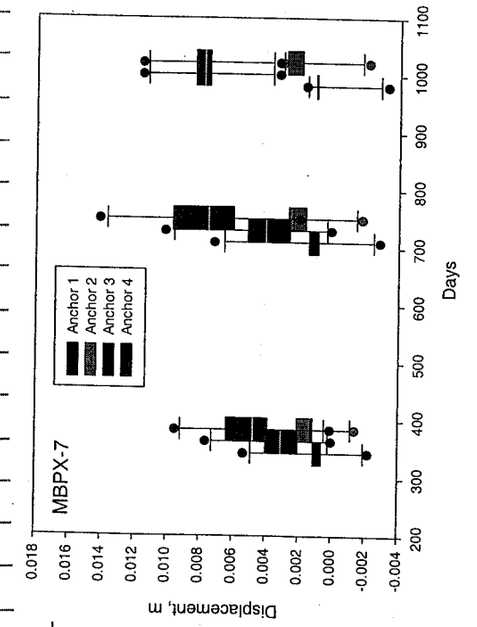
Sui-Min Hsiung 2/5/2002

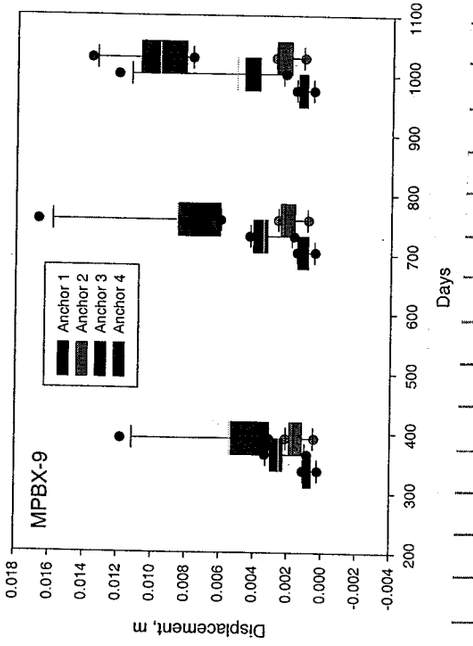
Box Plot on P.27





Black & White version of Box Plot on P.27





The displacement results from DDA analysis were presented at the Task Force meeting in Feb. 2002 in Washington, DC. The Task Lead (DOE research Team) indicated that besides the displacement prediction, the effect of themal-mechanical coupling on permeability changes should be analyzed too. DDA is adequate in displacement prediction. However, to predict TM effects on permeability, it is necessary to modified DDA. It was determined that it may be too much effort for the work at hand. After extensive consideration, we decided to use FLAC computer code to simulate TM-effect on permeability change. FLAC is finite difference code and is currently under CNWRA configuration control. Sui-Min Hsiung 07/10/2002

FLAC is capable of modeling thermal-mechanical (TM) coupling. However, it does not accept temperature change as an input for thermal-mechanical calculation. In our analysis, temperature change is an input. As discussed on page 4, the temperature used in our analysis was calculated based on field measurement. In order to make FLAC to accept input temperature for TM analysis, A FISH function was developed. This FISH function takes the temperature changes to compute thermally-induced stresses in the finite difference zones. and let model to cycle to equilibrium. Through this FISH function, the thermally induced stresses are indirectly generated.

~~The FISH function accounted for thermally induced~~ stresses is called tmeffect.fis.

~~The code for this function is listed on next page~~

Sui-Min Hsiung 7/12/2002

```
;FISH change temperature
;set echo off
;lower lith      thexp=use table 1
;middle nonlith thexp=use table 2
;upper lith      thexp=use table 3
def s_change
  float a1 a2 $ave
  loop i (1, izones)
    loop j (1, jzones)
      $ave=(temp(i,j)+temp(i+1,j)+temp(i+1,j)+temp(i+1,j+1))/4.0
      a1=18.0*bulk_mod(i,j)*shear_mod(i,j)*$ave*thexp(i,j)
      a2=(3.0*bulk_mod(i,j)+4.0*shear_mod(i,j))
      if a2>0 then
        sxx(i,j)=sxx(i,j)-a1/a2
        syy(i,j)=syy(i,j)-a1/a2
        szz(i,j)=szz(i,j)-a1/a2
      endif
    endLoop
  endLoop
end
;s_change
```

A typical FLAC data input is listed in the following.

```
;config ats thermal
title
DECOVALEX III Drift Scale Test TM analysis, intact rock
;
;set up model domain
;Scale for grid number and coords is 1
;grid 401 x 401, Drift center (201,301)
;
grid 400,400
;Mohr-Coulomb model
m ss th_i
;
; density d, friction anlge fri, cohesion coh,c, tentile strnegth ten
; thermal expansion coefficient thexp (intack rock)
; lower lith, Tptpll
;
;prop d=.0022936 fri=47 c=12.65 ten=5.48 thexp 0.00000928 i=1,401 j=1,282
prop d=.0022936 thexp 0.00000928 i=1,401 j=1,282
prop ftab=11 ctab=12 ttab=13 dtab=14 i=1,401 j=1,282
prop fri=47.45 c=12.65 ten=5.48 di=23.73 i=1,401 j=1,282
table 11 0,47.45
table 12 0,12.65
table 13 0,5.48
table 14 0,23.74
;
; middle nonlith, Tptpmn
;
;prop d=.0022936 fr=48 c=38.69 ten=8.91 thexp 0.00000901 i=1,401 j=282,318
prop d=.0022936 thexp 0.00000901 i=1,401 j=282,318
prop ftab=21 ctab=22 ttab=23 dtab=24 i=1,401 j=282,318
prop fri=48.15 c=38.69 ten=8.91 di=24.08 i=1,401 j=282,318
table 21 0,48.15
table 22 0,38.69
table 23 0,8.91
table 24 0,24.08
```



```
;
; upper lith, Tptpul
;
;prop d=.0022 fr=47 c=12.65 ten=5.48 thexp 0.00000898 i=1,401 j=319,401
prop d=.0022 thexp 0.00000898 i=1,401 j=119,201
prop ftab=31 ctab=32 ttab=33 dtab=34 i=1,401 j=319,401
prop fri=47.45 c=12.65 ten=5.48 di=23.73 i=1,401 j=319,401
table 31 0,47.45
table 32 0,12.65
table 33 0,5.48
table 34 0,23.73
ca material.fis
gen circle ~~201~~ 200 ~~301~~ 300 2.5
;
; apply gravity
;
set grav=9.81
;
; set initial stress to the model to reduce consolidtion time
; between 0.0225*540, 0.0225*140, (0.0225*340-0.0225*140)
; 540 m bottom of model from ground surface
; 140 m top    of model from ground surface
;
initial syy=-12.15 var 0,9.0 j 1,401
;
; between 0.0225*540*0.21/(1-0.21), 0.0225*140*0.21/(1-0.21)
; (0.0225*540*0.21/(1-0.21)-0.0225*140*0.21/(1-0.21))
;
initial sxx=-3.23 var 0,2.39 j 1,401
initial szz=-3.23 var 0,2.39 j 1,401
;
; apply contstant stress on top of model (0.0216*140) to simulate
; additional overburden
;
apply syy=-3.024 j=401
;
; displacement boundary conditions
;
fix x i=1
fix x i=401
fix y j=1
set mech on thermal off
;return
solve
save DIII_401x401_intact_pre.sav
;
; set displacements to 0
;

ini xdis=0
ini ydis=0
;
; drift excavation
;
model null region 300 301 300 301
solve
save DIII_401x401_intact_excavate.sav
;return
new
ca DIII_10_401x401_1_intact_ss.dat
```

7/17/2002

The listing for DIII_10_401x401_intact_ss.dat starts on next page.

```
;
;TM effect at cross section 10 m from bulkhead, intact rock
; grid 401 x 401, Drift center (201,301)
;
restore DIII_401x401_Intact_excavate.sav
;
;
; compute TM effects
;
ini xdis=0
ini ydis=0
;
;Define the total temperature array and set them to ambient (24C)
;
ca DefTempArray.fis
;
;define temperature dependency of thermal expansion coefficient
;
ca tableThexp.dat
;
;Read in temperature increments, reset temp(i,j)
;temperature increase at 3 months
;
ca tempInput10z_3m.dat
;
;compute total temperature
;
ca TempArray.fis
Total_temp
;
;initialize temperature dependent thermal expansion cofficients
;
ca expansion.fis
expansion
;
;compute thermally induced stresses due to temperature change
;temperature increase at 3 months
;
call tmeffect.fis
s_change
;
;cycle to equilibrium
;
;return
solve
save DIII_10_401x401_Intact_3m.sav
;
;Read in temperature increments, reset temp(i,j)
;temperature increase between 3m and 6m
;
ca tempInput10z_6m.dat
;
;compute total temperature
;
;ca TempArray.fis
Total_temp
;
;initialize temperature dependent thermal expansion cofficients
;
;ca expansion.fis
expansion
;
;compute thermally induced stresses
;temperature increase at 6 months
;
;call tmeffect.fis
s_change
;
;cycle to equilibrium
;
```

grid
Shw 7/17/2002


```
solve
save DIII_10_401x401_Intact_6m.sav
;
;Read in temperature increments, reset temp(i,j)
;temperature increase between 6m and 9m
;
ca tempInput10z_9m.dat
;
;compute total temperature
;
;ca TempArray.fis
Total_temp
;
;initialize temperature dependent thermal expansion cofficients
;
;ca expansion.fis
expansion
;
;compute thermally induced stresses due to temperature change
;temperature increase at 9 months
;
;call tmeffect.fis
s_change
;
;cycle to equilibrium
;
solve
save DIII_10_401x401_Intact_9m.sav
;
;Read in temperature increments, reset temp(i,j)
;temperature increase between 9m and 1 year
;
ca tempInput10z_1y.dat
;
;compute total temperature
;
;ca TempArray.fis
Total_temp
;
;initialize temperature dependent thermal expansion cofficients
;
;ca expansion.fis
expansion
;
;compute thermally induced stresses due to temperature change
;temperature increase at 1 year
;
;call tmeffect.fis
s_change
;
;cycle to equilibrium
;
;return
solve
save DIII_10_401x401_Intact_1y.sav
;
;Read in temperature increments, reset temp(i,j)
;temperature increase between 1y and 2y
;
ca tempInput10z_2y.dat
;
;compute total temperature
;
;ca TempArray.fis
Total_temp
;
```

```
;initialize temperature dependent thermal expansion cofficients
;
;ca expansion.fis
expansion
;
;compute thermally induced stresses due to temperature change
;temperature increase at 2 years
;
;call tmeffect.fis
s_change
;
;cycle to equilibrium
;
solve
save DIII_10_401x401_Intact_2y.sav
;
;Read in temperature increments, reset temp(i,j)
;temperature increase between 2y and 3y
;
ca tempInput10z_3y.dat
;
;compute total temperature
;
;ca TempArray.fis
Total_temp
;
;initialize temperature dependent thermal expansion cofficients
;
;ca expansion.fis
expansion
;
;compute thermally induced stresses due to temperature change
;temperature increase at 3 year
;
;call tmeffect.fis
s_change
;
;cycle to equilibrium
;
solve
save DIII_10_401x401_Intact_3y.sav
;
;Read in temperature increments, reset temp(i,j)
;temperature increase between 3y and 4y
;
ca tempInput10z_4y.dat
;
;compute total temperature
;
;ca TempArray.fis
Total_temp
;
;initialize temperature dependent thermal expansion cofficients
;
;ca expansion.fis
expansion
;
;compute thermally induced stresses due to temperature change
;temperature increase at 4 years
;
;call tmeffect.fis
s_change
;
;cycle to equilibrium
;
solve
save DIII_10_401x401_Intact_4y.sav
;
return
```


The input data is essentially including several steps. 07/13/2002

The first step is to define the problem domain and failure model by using keywords grid and model.

The second step is to define material properties and strength properties for the three thermal-mechanical units simulated in the model. The material and strength properties are taken from "TBV-332/TBD-325 Resolution Analysis: Geotechnical Rock Properties" B00000000-01717-5705-00134 Rev00, CRWMS M&O, 1999. Las Vegas, Nevada." This report was provided by the Task Lead in March, 2002.

A FISH function "material.fis" was written to calculate Bulk and shear moduli from young's modulus and Poisson's ratio that are the data provided in the above report.

The listing material.fis is listed in the following (next page)

7/14/2002

```
;FISH set bulk and shear moduli, intact rock
;grid 401 x 401
;set echo off
def material
;
;lower lith, Tptpll
;
loop i (1, izones)
  loop j (1, 281)
    bulk_mod(i,j)=33030/(2*(1+0.21))
    shear_mod(i,j)=33030/(3.0*(1.0-2.0*0.21))
  endLoop
endLoop
;
;middle nonlith, Tptpmn
;
loop i (1, izones)
  loop j (282, 318)
    bulk_mod(i,j)=33030.0/(2*(1+0.21))
    shear_mod(i,j)=33030.0/(3.0*(1.0-2.0*0.21))
  endLoop
endLoop
;
;Upper lith, Tptpul
;
loop i (1, izones)
  loop j (319, jzones)
    bulk_mod(i,j)=20360/(2*(1+0.23))
    shear_mod(i,j)=20360/(3.0*(1.0-2.0*0.23))
  endLoop
endLoop
end
material
```

The thermal expansion coefficients for the three rock units in the FLAC analysis are temperature dependent and provided in three tables: table 1 is for Tptpll, table 2 is for Tptpmn, and table 3 is for Tptpul. These data are also from the report listed on page 35.

Sui-Man Hsu 7/14/2002

The listing for thermal expansion coefficients is in the following.

```
;FISH set temperature depenent thermal expansion coefficient
;set echo off
;
;three table for temerpature dependent thermal expansion cofficients
;table 1: Tptpll;
;
table 1 insert  0, 0.00000641
table 1 insert  50,0.00000641
table 1 insert  51,0.00000815
table 1 insert  75,0.00000815
table 1 insert  76,0.00000877
table 1 insert 100,0.00000877
table 1 insert 101,0.00000912
table 1 insert 125,0.00000912
table 1 insert 126,0.00000987
table 1 insert 150,0.00000987
table 1 insert 151,0.00001075
table 1 insert 175,0.00001075
table 1 insert 176,0.00001255
table 1 insert 200,0.00001255
table 1 insert 201,0.00001514
table 1 insert 225,0.00001514
table 1 insert 226,0.00002519
table 1 insert 250,0.00002519
table 1 insert 251,0.00002615
table 1 insert 275,0.00002615
table 1 insert 276,0.00003340
table 1 insert 500,0.00003340
;
;table 2: Tptpmn;
;
table 2 insert  0, 0.00000689
table 2 insert  50,0.00000689
table 2 insert  51,0.00000845
table 2 insert  75,0.00000845
table 2 insert  76,0.00000895
table 2 insert 100,0.00000895
table 2 insert 101,0.00000950
table 2 insert 125,0.00000950
table 2 insert 126,0.00001012
table 2 insert 150,0.00001012
table 2 insert 151,0.00001095
table 2 insert 175,0.00001095
table 2 insert 176,0.00001209
table 2 insert 200,0.00001209
table 2 insert 201,0.00001457
table 2 insert 225,0.00001457
table 2 insert 226,0.00001945
table 2 insert 250,0.00001945
table 2 insert 251,0.00002724
table 2 insert 275,0.00002724
table 2 insert 276,0.00004156
table 2 insert 500,0.00004156
```

Sui-Man Hsu 7/14/2002

```
;
;table 3: Tptpul;
;
table 3 insert  0, 0.00000741
table 3 insert  50,0.00000741
table 3 insert  51,0.00000843
table 3 insert  75,0.00000843
table 3 insert  76,0.00000889
table 3 insert 100,0.00000889
table 3 insert 101,0.00000952
table 3 insert 125,0.00000952
table 3 insert 126,0.00001086
table 3 insert 150,0.00001086
table 3 insert 151,0.00001351
table 3 insert 175,0.00001351
table 3 insert 176,0.00001938
table 3 insert 200,0.00001938
table 3 insert 201,0.00002934
table 3 insert 225,0.00002934
table 3 insert 226,0.00003235
table 3 insert 250,0.00003235
table 3 insert 251,0.00004016
table 3 insert 275,0.00004016
table 3 insert 276,0.00004883
table 3 insert 500,0.00004883
```

Sui Min Hu 7/14/2002

In the FLAC modeling the initial ambient temperature is set to 24°C using the following FISH function:

```
;FISH define an array to store temperature (not temperature differnece)
; set the initial temperature to 24 as default
;set echo off
;
def Total_temp1
  float TotalTemp
  array TotalTemp(401,401)
  loop i (1, igp)
    loop j (1, jgp)
      TotalTemp(i,j)=24
    endLoop
  endLoop
end
Total_temp1
```

The drift modeled in 6 m in diameter. The vertical boundaries are 300 m away from the center of the drift for the data listing provided on page 30. The bottom of the model is 300 m below the center of the drift and the top of the model is 100 m above,

The rest of overburden about 140 m is accounted for using applied stress on top of the model. Vertical boundaries are defined to prevent horizontal displacement at the boundaries. Bottom boundary prevents vertical displacement at the boundary. The model is first cyled to reach static state. Then the drift is constructed and cycled to static state. Temperature increments take 8 thermal step at 3 month, 6 month, 9 month, 1 year, 2 year, 3 year, and 4 year as specified by the Task Force. Sui Min Hu 7/14/2002 At each time increment, the temperature difference is read in as initial temperature. This difference is added to the temperature to determine the total (absolute) temperature at the time increment Sui Min Hu 7/14/2002 using the FISH function provided in the following

```
;FISH define an array to store temperature (not temperature differnece)
;set echo off
;
def Total_temp
  loop i (1, igp)
    loop j (1, jgp)
      TotalTemp(i,j)=TotalTemp(i,j)+temp(i,j)
      ex_1(i,j)=TotalTemp(i,j)
    endLoop
  endLoop
end
;Total_temp
```

This absolute temperature is then used to determine temperature-dependent thermal expansion coefficient using the following FISH function.

```
;FISH temperature dependent thermal expansion
;grid 401 x 401
;set echo off
def expansion
  float $ave1 $ave
;
;lower lith
;
  loop i (1, izones)
    loop j (1, 281)
      $ave1=TotalTemp(i,j)+TotalTemp(i+1,j)+TotalTemp(i+1,j)
      $ave =($ave1+TotalTemp(i+1,j+1))/4.0
      thexp(i,j)=table(1,$ave)
    endLoop
  endLoop
;
;middle nonlith
;
  loop i (1, izones)
    loop j (282, 318)
      $ave1=TotalTemp(i,j)+TotalTemp(i+1,j)+TotalTemp(i+1,j)
      $ave =($ave1+TotalTemp(i+1,j+1))/4.0
      thexp(i,j)=table(2,$ave)
    endLoop
  endLoop
;
;upper lith
;
  loop i (1, izone)
    loop j (319, jzones)
      $ave1=TotalTemp(i,j)+TotalTemp(i+1,j)+TotalTemp(i+1,j)
      $ave =($ave1+TotalTemp(i+1,j+1))/4.0
      thexp(i,j)=table(3,$ave)
    endLoop
  endLoop
end
;expansion
```

Once thermal expasion coefficients at that temperature state are determined, FISH function on page 30 is used to compute stresses and cycled to static state before the next temperature increase (or change). The above listings are for intact rock properties.

For rock mass properties, the following FISH function is used to compute shear and bulk moduli.

```
;FISH set bulk and shear moduli for rock mass
;set echo off
def material
loop i (1, izones)
  loop j (1, 281)
    bulk_mod(i,j)=12020/(2*(1+0.21))
    shear_mod(i,j)=12020/(3.0*(1.0-2.0*0.21))
  endLoop
endLoop
loop i (1, izones)
  loop j (282, 318)
    bulk_mod(i,j)=12020.0/(2*(1+0.21))
    shear_mod(i,j)=12020.0/(3.0*(1.0-2.0*0.21))
  endLoop
endLoop
loop i (1, izones)
  loop j (319, 400)
    bulk_mod(i,j)=14280/(2*(1+0.23))
    shear_mod(i,j)=14280/(3.0*(1.0-2.0*0.23))
  endLoop
endLoop
end
material
```

Smith HCJ 7/14/2002

William Clay Flannigan
7/26/02

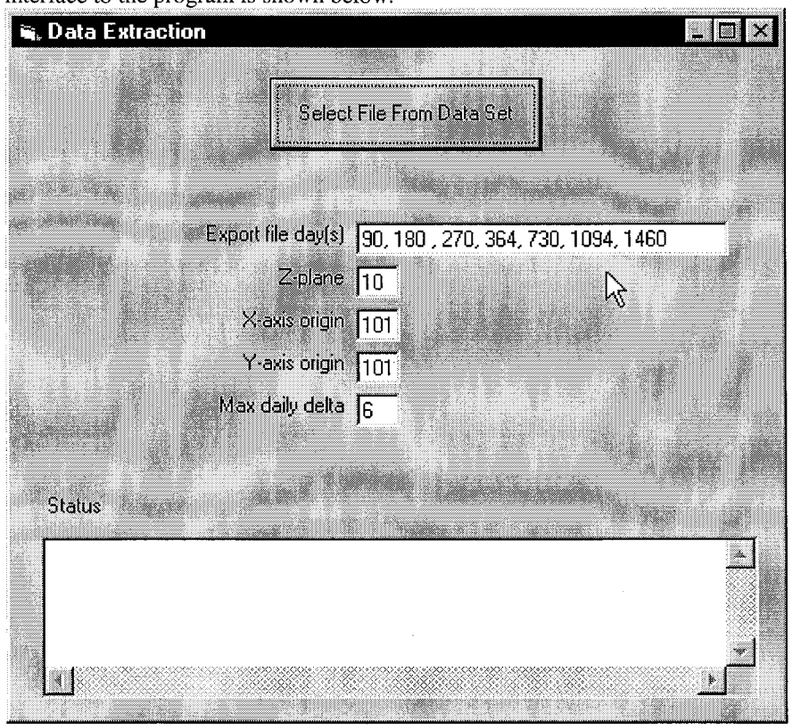
William Clay Flannigan 7/26/02

Temperature Input File Formatting 7/26/02

The raw temperature data files require reformatting for the FLAC code to:

- Select only a single plane of data transverse to the drift for the 2-D analysis
- Shift the coordinate system
- Extract only the two coordinate values and one temperature value
- Generate absolute temperature and temperature differences between the time increments in question
- Compensate for temperature variations due to the heat-stack effect

A simple program compiled in Microsoft Visual Basic 6.0 was created to generate the data sets. The interface to the program is shown below.



The code for the program is listed below.

William Clay Flannigan
7/26/02


```
Option Explicit

Private Type fileType
    temp(-35 To 35, -35 To 35)  As Single
End Type

Private Type dataType
    directory       As String
    fileString      As String
    fileExtension   As String
    exportDays      As Collection
    files           As Collection
End Type

Dim data                                As dataType
Dim curFile(-35 To 35, -35 To 35)   As Single


Private Sub selectFirstFile_Click()

    Dim p1, p2, p3              As Integer
    Dim i                       As Integer
    Dim fileName                As String
    Dim initFileNum             As Integer
    Dim prevPos, pos            As Integer
    Dim fileNum                 As Integer
    Dim prevDay                 As Integer


    On Error GoTo done

    ' Get the first file name
    With OpenDialog
        .InitDir = App.Path
        .Flags = cdlOFNFileMustExist Or cdlOFNPathMustExist Or cdlOFNHideReadOnly
        .Filter = "pdat (*.pdat)|*.pdat"
        .ShowOpen
        .CancelError = True
    End With

    On Error GoTo failure

    txtStatus = ""

    Main.Refresh

    Screen.MousePointer = vbHourglass

    ' Extract name file data
    p1 = InStrRev(OpenDialog.fileName, ".", Len(OpenDialog.fileName))
    data.fileExtension = Mid(OpenDialog.fileName, p1)
    p2 = InStrRev(OpenDialog.fileName, "\", Len(OpenDialog.fileName))
    data.directory = Mid(OpenDialog.fileName, 1, p2)
    p3 = InStr(p2, OpenDialog.fileName, "0")
    data.fileString = Mid(OpenDialog.fileName, p2 + 1, p3 - p2 - 1)
    initFileNum = Mid(OpenDialog.fileName, p3, p1 - p3)

    ' Initialize the data structure
    Set data.files = New Collection
    Set data.exportDays = New Collection

    ' Loop over all the files
    prevPos = 0
    prevDay = -1
    Do
        ' Get the next file name from text string
        pos = InStr(prevPos + 1, txtExport, ",")
        If pos <> 0 Then
            fileNum = Mid(txtExport, prevPos + 1, pos - prevPos - 1) / 2
        Else
            fileNum = Mid(txtExport, prevPos + 1) / 2
        End If
        data.exportDays.Add (fileNum * 2)

        ' Load file
        fileName = data.directory & data.fileString & _
            Format(fileNum, "0000") & data.fileExtension
        loadDataFile fileName, curFile
        Call data.files.Add(curFile, Str(fileNum * 2))

        ' Save and format the file
        saveFile fileNum * 2, prevDay, False

        ' Remove deviations in the data
```

```
            averageData (fileNum * 2)

            ' Save and format the file
            saveFile fileNum * 2, prevDay, True

            prevPos = pos
            prevDay = fileNum * 2
        Loop While pos <> 0

        ' Save data over time to a single file
        saveTimeData ("timeData.dat")

        txtStatus = "done" & vbCrLf & txtStatus
        Screen.MousePointer = vbDefault

        Exit Sub

failure:
    MsgBox Err.Description, vbCritical, "Error"
done:
    Screen.MousePointer = vbDefault
End Sub

Private Sub loadDataFile(fileName As String, curFile() As Single)

    Dim line                    As String
    Dim originalPos(3)          As Integer
    Dim temp                    As Single
    Dim pos                     As Integer
    Dim i                       As Integer
    Dim fileSystemObject, file  As Object

    On Error GoTo fileError

    Set fileSystemObject = CreateObject("Scripting.FileSystemObject")
    Set file = fileSystemObject.openTextFile(fileName)

    txtStatus = "Importing file " & fileName & vbCrLf & txtStatus
    Main.Refresh

    Do Until file.atEndOfStream

        ' Parse a line of data
        line = file.readLine
        If Mid(line, 1, 1) <> "#" And line <> "" Then
            For i = 1 To 3
               pos = InStr(1, line, " ")
               originalPos(i) = Trim(Mid(line, 1, pos))
               line = Trim(Mid(line, pos))
            Next i
            pos = InStr(1, line, " ")
            temp = Val(Trim(Mid(line, 1, pos)))

            ' Record the data if it is in the proper cut plane
            If originalPos(2) = txtZPlane Then
                curFile(originalPos(1), originalPos(3)) = temp
            End If
        End If
    Loop

    file.Close

Exit Sub
fileError:
    MsgBox "Unable to load file " & fileName, vbCritical, "Error"
    file.Close
End Sub

Private Sub saveFile(curDay As Integer, prevDay As Integer, modData As Boolean)

    Dim i, j                As Integer
    Dim fileName            As String
    Dim curTemp, prevTemp   As Single

    On Error GoTo fileError

    ' Actual temperature data
    If modData Then
        fileName = data.directory & "raw_modified_" & "day=" & curDay & "_z=" _
            & txtZPlane & "_center=" & txtXOrigin & "," & txtYOrigin & ".dat"
    Else
        fileName = data.directory & "raw_unmodified_" & "day=" & curDay & "_z=" _
            & txtZPlane & "_center=" & txtXOrigin & "," & txtYOrigin & ".dat"
    End If
```



```
    Open fileName For Output As #1

    ' Delta temperature data
    If modData Then
        fileName = data.directory & "delta_modified_" & "day=" & curDay & "_z=" _
            & txtZPlane & "_center=" & txtXOrigin & "," & txtYOrigin & ".dat"
    Else
        fileName = data.directory & "delta_unmodified_" & "day=" & curDay & "_z=" _
            & txtZPlane & "_center=" & txtXOrigin & "," & txtYOrigin & ".dat"
    End If
    Open fileName For Output As #2

    txtStatus = "Saving file " & fileName & vbCrLf & txtStatus
    Main.Refresh

    For j = 1 To txtYOrigin * 2 - 1
        For i = 1 To txtXOrigin * 2 - 1
            If i < txtXOrigin - 35 Or i > txtXOrigin + 35 Or j < txtYOrigin - 35 Or j > txtYOrigin + 35 Then
'                Print #1, "initial temp 00.0000 i = " & Format(i, "000") & " j = "; Format(j, "000")
'                Print #2, "initial temp 00.0000 i = " & Format(i, "000") & " j = "; Format(j, "000")
            Else
                curTemp = data.files(Str(curDay))(i - txtXOrigin, j - txtYOrigin)
                If prevDay = -1 Then
                    prevTemp = 24
                Else
                    prevTemp = data.files(Str(prevDay))(i - txtXOrigin, j - txtYOrigin)
                End If
                Print #1, "initial temp " & Format(curTemp, "00.0000") _
                    & " i = " & Format(i, "000") & " j = "; Format(j, "000")
                Print #2, "initial temp " & Format(curTemp - prevTemp, "00.0000") _
                    & " i = " & Format(i, "000") & " j = "; Format(j, "000")
            End If
        Next i
    Next j

    Close #1
    Close #2

    Exit Sub
fileError:
    MsgBox "Unable to save file " & fileName, vbCritical, "Error"
    Close #1
    Close #2
End Sub

Private Sub saveTimeData(fileName As String)

    Const numX = 5
    Const numY = 5

    Dim i, j, k                 As Integer
    Dim fileSystemObject, file  As Object
    Dim pointsX(numX)           As Integer
    Dim pointsY(numY)           As Integer

    On Error GoTo fileError

    Set fileSystemObject = CreateObject("Scripting.FileSystemObject")
    Set file = fileSystemObject.createTextFile(data.directory & fileName, True)

    file.write "day"
    For i = 1 To numX
        pointsX(i) = Int((i - 0.5) * 71 \ numX - 35)
        For j = 1 To numY
            pointsY(j) = Int((j - 0.5) * 71 \ numY - 35)
            file.write vbTab & "x=" & pointsX(i) & "y=" & pointsY(j)
        Next j
    Next i
    file.writeline

    For k = 1 To data.exportDays.Count
        file.write data.exportDays(k) & vbTab
        For i = 1 To numX
            For j = 1 To numY
                file.write data.files(Str(data.exportDays(k)))(pointsX(i), pointsY(j)) & vbTab
            Next j
        Next i
        file.writeline
    Next k

    file.Close

    txtStatus = "File " & data.directory & fileName & " saved" & vbCrLf & txtStatus
    Main.Refresh
```

```
    Exit Sub
fileError:
    MsgBox "Unable to write file " & data.directory & fileName, vbCritical, "Error"
    file.Close

End Sub
Private Sub averageData(day As Integer)

    Dim i, j                                    As Integer
    Dim curDay                                  As Integer
    Dim done                                    As Boolean
    Dim goodLowerPoint(-35 To 35, -35 To 35)    As Integer
    Dim goodUpperPoint(-35 To 35, -35 To 35)    As Integer
    Dim fileName                                As String
    Dim newVal                                  As Single

    On Error GoTo failed

    For i = -35 To 35
        For j = -35 To 35
            goodLowerPoint(i, j) = -1
            goodUpperPoint(i, j) = -1
        Next j
    Next i

    ' Look backwards in the files
    curDay = day
    done = False
    While curDay > 0 And Not done

        ' Load the previous file
        curDay = curDay - 2
        fileName = data.directory & data.fileString & _
            Format(curDay / 2, "0000") & data.fileExtension
        loadDataFile fileName, curFile
        Call data.files.Add(curFile, Str(curDay), Str(curDay + 2))

        ' Find the previous good point
        done = True
        For i = -35 To 35
            For j = -35 To 35
                If goodLowerPoint(i, j) = -1 Then
                    If Abs(data.files(Str(curDay))(i, j) - data.files(Str(curDay + 2))(i, j)) > Val(txtDelta) Then
                        done = False
                    Else
                        goodLowerPoint(i, j) = curDay + 2
                    End If
                End If
            Next j
        Next i
    Wend

    ' Look forwards in the files
    curDay = day
    done = False
    While curDay < 1480 And Not done

        ' Load the next file
        curDay = curDay + 2
        fileName = data.directory & data.fileString & _
            Format(curDay / 2, "0000") & data.fileExtension
        loadDataFile fileName, curFile
        Call data.files.Add(curFile, Str(curDay), , Str(curDay - 2))

        ' Find the next good point
        done = True
        For i = -35 To 35
            For j = -35 To 35
                If goodUpperPoint(i, j) = -1 Then
                    If Abs(data.files(Str(curDay))(i, j) - data.files(Str(curDay - 2))(i, j)) > Val(txtDelta) Then
                        done = False
                    Else
                        goodUpperPoint(i, j) = curDay - 2
                    End If
                End If
            Next j
        Next i
    Wend

    ' Interpolate between the good points
    For i = -35 To 35
        For j = -35 To 35
```

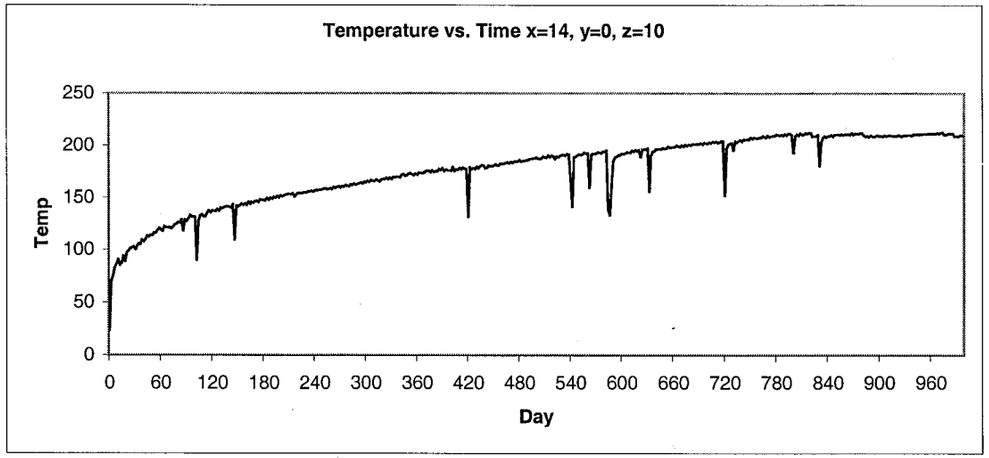


```
                curFile(i, j) = data.files(Str(day))(i, j)
                If goodLowerPoint(i, j) <> goodUpperPoint(i, j) Then
                    newVal = (data.files(Str(goodUpperPoint(i, j)))(i, j) - data.files(Str(goodLowerPoint(i, j)))(i, j)) _
                        * (day - goodLowerPoint(i, j)) / (goodUpperPoint(i, j) - goodLowerPoint(i, j)) _
                        + data.files(Str(goodLowerPoint(i, j)))(i, j)
                    curFile(i, j) = newVal
                End If
            Next j
        Next i
        Call data.files.Remove(Str(day))
        Call data.files.Add(curFile, Str(day), , Str(day - 2))

        txtStatus = "Data interpolation done for day " & day & vbCrLf & txtStatus
        Main.Refresh

        Exit Sub
failed:
    MsgBox "Unable to interpolate point x=" & i & " y=" & j & " on day " & day, vbCritical, "Error"

End Sub
```

The code performs the following actions:

- Extract the user interface inputs
- Read in the input file data
- Save the raw temperature data to output file, shifting the coordinate system
- Save the delta temperature data to output file by taking the difference between the current data set and the previous set
- Interpolate the data to remove deviations from the heat-stack effect
- Save both the raw and delta interpolated temperature data to output file

The only manipulation of the data occurs during the interpolation step in an attempt to eliminate the heat-stack effect. The effect can be seen in the following graph as short term spikes in the temperature data.



The interpolation occurs as follows. For each data point, the temperature value is compared with the previous data set (2 day increments). If the difference is greater than a given tolerance, then the algorithm will continue to look backwards in the data sets until a the temperature difference between two consecutive data sets is within tolerance. The same routine occurs looking forward into the data sets. The result are two valid temperature values that bound the point in question. Linear interpolation is then used to aquire a temperature value for the given day. A delta temperature tolerance of 6 was used.

An example of the calculation is shown for the data graphed above and tabulated below

Day	Temperature
414	178.4941
416	177.8432
418	177.3402
420	131.7695
422	178.7308
424	177.4312
426	178.1216

If the day in question is 420, looking backwards, the difference between days 420 and 418 is 45.5707 – out of tolerance. The difference between 418 and 416 is 0.5030 – within tolerance. Similarly, looking forward, the difference between days 422 and 424 are within tolerance. Linear interpolation is then used between days 422 and 418 to calculate a temperature value for day 420:

$$(178.7308 - 177.3402) * \frac{(420 - 418)}{(422 - 418)} + 177.3402 = 178.0355$$

Siu-Ming Ho 8/1/2002

Another small program (C++) using Borland C++Builder version 6 is written to calculate temperature data input for FLAC modeling. This program runs in MS Windows. The listing is provided below

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "ReadASCII.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TReadTextFile *ReadTextFile;
//---------------------------------------------------------------------------
__fastcall TReadTextFile::TReadTextFile(TComponent* Owner)
    : TForm(Owner),Index(0)
{
  //cross section where temperature profile is extracted
  CrossSection.push_back(10);
  CrossSection.push_back(23);
  CrossSection.push_back(30);
//  CrossSection=30;
//  long CrossSection(21);
  // shift temperature grid to match Flac model grid
  // ModelSize--0:201x201; 1:301x301; 2:401x301
  // corresponding offset
  // 1: xOffSet=101, yOffSet=101
  // 2: xOffSet=151, yOffSet=201
  // 3: xOffSet=201, yOffSet=301
//  long ModelSize(1);
```



```
  ModelIndex[0]="x_";
  ModelIndex[1]="y_";
  ModelIndex[2]="z_";
  xyOffSet.push_back(TPoint(101,101));
  xyOffSet.push_back(TPoint(151,201));
  xyOffSet.push_back(TPoint(201,303));
//  xyOffSet.push_back(TPoint(201,301));
  numCount=-1;
  CS=1; // Cross Section Index
}
//---------------------------------------------------------------------------

void __fastcall TReadTextFile::DoneClick(TObject *Sender)
{
  Close();
}
//---------------------------------------------------------------------------

void __fastcall TReadTextFile::BitBtn1Click(TObject *Sender)
{
  OpenDialog1->InitialDir = ExtractFilePath(ParamStr(0));
  OpenDialog1 = new TOpenDialog(this);
  OpenDialog1->Filter = "Data files (*.dat)|*.dat";
  if(OpenDialog1->Execute())
  {
    if(Index==0)
    {
      readfile(OpenDialog1->FileName);
      processdata();
    }
    if(Index==1)
    {
      Editor->Text="";
      Editor->Lines->LoadFromFile(OpenDialog1->FileName);
    }
  } // if(OpenDialog->Execute())
}
//---------------------------------------------------------------------------

void __fastcall TReadTextFile::BitBtn2Click(TObject *Sender)
{
  Index=1;
  BitBtn1Click(Sender);
  processdata();
}
//---------------------------------------------------------------------------

void __fastcall TReadTextFile::BitBtn3Click(TObject *Sender)
{
  // User Define Reading Approach
  // User Define Reading Approach
//  long numX,numY,StartNo=401;
  StartNo=0;
//  number=StartNo+3;
  number=StartNo+7;
  int number1=730;
  temp1.Null(number+1,3);
  temp2.Null(number+1,5042,3);
```

```
for(fNum=StartNo; fNum<=number1; fNum++)
{
  if(fNum!=0&&fNum!=45&&fNum!=90&&fNum!=135&&fNum!=182&&fNum!=365&&
     fNum!=547&&fNum!=730) continue;
  numCount++;
  numX=0; numY=0;
  AnsiString as;
  if(fNum<=501)              as="dstout_";
  else                       as="dst_";
  if(fNum<10)                as+="000";
  if(fNum>=10&&fNum<=100)    as+="00";
  if(fNum>=100&&fNum<1000)   as+="0";
  if(fNum<=501)              as+=IntToStr(fNum)+".pdat";
  else                       as+=IntToStr(fNum)+"out.pdat";
  RichEdit1->Text="Read and Process "+as;
  if(Index==0)
  {
    inputVector.clear();
    readfile(as);
  }
  else
  {
    Editor->Text="";
    Editor->Lines->LoadFromFile(as);
  }
  processdata();
} // for(fNum=StartNo; fNum<=number; fNum++)
if(Index==0)
{
  buffer="";
  delete [] buffer;
}
else
{
  Editor->Text="";
  delete Editor;
}
processdata2();
RichEdit1->Text="Task Completed";
}
//---------------------------------------------------------------------------

void __fastcall TReadTextFile::BitBtn4Click(TObject *Sender)
{
  Index=1;
  BitBtn3Click(Sender);
}
//---------------------------------------------------------------------------

string TReadTextFile::readline(istream& is)
{
  string line;
  getline(is,line);
  return line;
}
//---------------------------------------------------------------------------

void TReadTextFile::readfile(AnsiString inputfileName)
```



```
{
  ifstream ifs(inputfileName.c_str());
  while(ifs)
  {
    str=readline(ifs);
    inputVector.push_back(str);
  }
}
//---------------------------------------------------------------------------

void TReadTextFile::processdata()
{
  // Only thing that needs to be done is to change codes
  // in this function
  long double A,B,C,D;
  long E,F,G;
  int vec_size;
  bool StopX(false);
  if(Index==0) vec_size=inputVector.size();
  else         vec_size=Editor->Lines->Count;
  numX=0; numY=0;
  i1=0;
  for(int I(24); I<vec_size; I++)
  {
  // variable buffer has to be used here to match what is used
  // in function vssf
    if(Index==0) buffer=inputVector[I].c_str();
    else         buffer=Editor->Lines->Strings[I].c_str();

    vssf("%Lf %Lf %Lf %Lf %d %d %d",&A,&B,&C,&D,&E,&F,&G);
    if(B!=(long double)CrossSection[CS]) continue;  // cross section
    i1++;
/*    temp2(fNum,i1,1)=A;
    temp2(fNum,i1,2)=C;
    temp2(fNum,i1,0)=D; */
    temp2(numCount,i1,1)=A;
    temp2(numCount,i1,2)=C;
    temp2(numCount,i1,0)=D;
    if(D>500)
    {
      int aaaaaa;
      aaaaaa=0;
    }
//    if(temp2(fNum,i1,0)<24.2) temp2(fNum,i1,0)=0.0;
//    else temp2(fNum,i1,0)=temp2(fNum,i1,0)-24.2;
    if(!numX&&!numY) { numX++; numY++; }
    else
    {
      if(temp2(numCount,i1,2)!=temp2(numCount,i1-1,2))
        { numY++; if(numX) StopX=true; }
      if(temp2(numCount,i1,2)==temp2(numCount,i1-1,2)&&!StopX) numX++;
    } // if(!numX&&!numY)
  } // for(int I(24); I<vec_size; I++)
  temp1(numCount,0)=i1;
  temp1(numCount,1)=numX;
  temp1(numCount,2)=numY;
} // void TReadTextFile::processdata()
//---------------------------------------------------------------------------
```

```
void TReadTextFile::processdata2()
{
  i1=temp1(1,0);
  long double tdegree(0);
  AnsiString str1;

  for(int i=StartNo+1; i<=number; i++)
  {
  // output temperature change between time 0 and the 3rd month
      for(int I(0); I<=2; I++)
      {
        str1="tempInput"+IntToStr(CrossSection[CS])+ModelIndex[I];
        if(i==1) str1+="3m.dat";
        if(i==2) str1+="6m.dat";
        if(i==3) str1+="9m.dat";
        if(i==4) str1+="1y.dat";
        if(i==5) str1+="2y.dat";
        if(i==6) str1+="3y.dat";
        if(i==7) str1+="4y.dat";
        f12=fopen(str1.c_str(),"w");
        for(int j=1; j<=i1; j++)
        {
          tdegree=temp2(i,j,0)-temp2(i-1 ,j,0);
          fprintf(f12,"initial temp %Lf i=%4d j=%4d\n",
                tdegree,(long)temp2(i,j,1)+xyOffSet[I].x,
                        (long)temp2(i,j,2)+xyOffSet[I].y);
      } // for(int j=1; j<=i1; j++)
      fclose(f12);
      } // for(int I(0); I<=2; I++)
  } // for(int i=StartNo+1; i<=number; i++)
  f12=NULL;
  delete [] f12;
} // void TReadTextFile::processdata2()
//---------------------------------------------------------------------

int TReadTextFile::vssf(char *fmt, ...)
{
   va_list  argptr;
   int cnt;

   va_start(argptr, fmt);
   cnt = vsscanf(buffer, fmt, argptr);
   va_end(argptr);

   return(cnt);
}
//---------------------------------------------------------------------
```

This small program calculates temperature difference between two temperature profiles of interest.

VERIFICATION OF THE PROGRAM


```
First Three Data at Cross Section 23 m from bulk head
from dstout_0045.pdat (temperature data at 3 months)

-35 23 -35 24.0203342437744 1 29 1
-34 23 -35 24.0206813812256 2 29 1
-33 23 -35 24.0208721160889 3 29 1

First Three Data at Cross Section 23 m from bulk head
from dstout_0000.pdat (temperature data before heating started)

-35 23 -35 24.0200901031494 1 29 1
-34 23 -35 24.0201396942139 2 29 1
-33 23 -35 24.0201988220215 3 29 1

Computed temperature difference between two files at fifferent locations
First three results from file temperature23z_3m.dat

initial temp 0.000244 i= 166 j= 268
initial temp 0.000542 i= 167 j= 268
initial temp 0.000673 i= 168 j= 268

Hand calulated results:

24.0203342437744-24.0200901031494=0.00024414

24.0206813812256-24.0201396942139=0.00054169

24.0208721160889-24.0201988220215=0.00067329

The hand calculation results are the same as the program calculated results
when rounding is considered.
```

Sin-Min Hsiung 8/1/2002

FLAC modeling

Two base cases were established for Task 2C modeling activities. For Base Case 1, the three litho-stratigraphic units were treated as intact rocks in the analysis. The material and strength properties used for these three units were the mean values presented in Tables 3-1 and 3-3. For Base Case 2, the three litho-stratigraphic units were treated as rock masses. The material and strength properties for Rock-Mass Quality Category 2 rock listed in Tables 3-2 and 3-4 were used for Base Case 2. The reason for selecting Rock-Mass Quality Category 2 as a base case is because limited *in-situ* rock-mass modulus data available for the Topopah Spring Welded Tuff Thermal-Mechanical Unit 2 indicated that these data fell into the <20% data range presented in Table 3-4 for the Topopah Spring Welded Tuff Thermal-Mechanical Unit 2 (CRWMS M&O, 1999). Selecting Rock-Mass Quality Category 2 appeared to represent the upper bound values for at least the Middle Nonlithophysal litho-stratigraphic unit.

Sin-Min Hsiung 8/2/2002

The *in-situ* deformation modulus of a rock mass is an important parameter in any form of numerical analysis and in the interpretation of monitored deformation around the heated drift. The intact rock Young's moduli, along with Poisson's ratios and bulk densities, for the three lithologic zones are listed in Table 3-1 (CRWMS M&O, 1999; CRWMS M&O, 1997c).

Table 3-1. Intact Rock Material Properties (CRWMS M&O, 1999; CRWMS M&O, 1997c)

	Upper Lithophysal Zone	Middle Nonlithophysal Zone	Lower Lithophysal Zone
Bulk Density, kg/m^3 [lb/ft^3]	2,160±80 [134.8±5.0]	2,250±70 [140.5±4.4]	2,250±60 [140.5±3.7]
Young's Modulus, GPa [10^6 psi]	20.36±6.75 [2.95±0.98]	33.03±5.94 [4.79±0.86]	33.03±5.94 [4.79±0.86]
Poisson's Ratio	0.23±0.07	0.21±0.04	0.21±0.04

Reference CRWMS M&O, 1999 can be found on p. 35
CRWMS M&O, 1997c can be found on p. 14

Table 3-3. Intact Rock Strength Properties (CRWMS M&O, 1999)

Thermal-Mechanical Unit	Cohesion, MPa [psi]	Uniaxial Compressive Strength, MPa [psi]	Tensile Strength, MPa [psi]	Friction Angle, degree	Dilation Angle, degree
Topopah Spring Welded Tuff Unit 1	12.65 [1,834.3]	58.22±30.69 [8,441.9±4,450.1]	5.48±2.32 [794.6±336.4]	47.45	23.73
Topopah Spring Welded Tuff Unit 2	38.69 [5,610.1]	167.90±65.53 [24,345.5±9,501.9]	8.91±3.39 [1.292±491.6]	48.15	24.08

The FLAC computer code accepts bulk and shear moduli (K and G) as material properties input. A FISH function was prepared to calculate K and G using the following equations.

$$K = \frac{E}{2(1+\nu)} \qquad (3\text{-}1)$$

and

$$G = \frac{E}{3(1-2\nu)} \qquad (3\text{-}2)$$

where E and ν are Young's modulus and Poisson's ratio, respectively.

A FISH function developed for this conversion purpose is provided on p. 36 and 41.

Upper Lithophysal litho-stratigraphic unit is part of the Topopah Spring Welded Tuff Thermal-Mechanical Unit 1 and the Middle Nonlithophysal and Lower Lithophysal litho-stratigraphic units are part of the Topopah Spring Welded Tuff Thermal-Mechanical Unit 2. In this study, the corresponding rock-mass Young's moduli were used for modeling accordingly.

Similar to the selection of Young's modulus values for FLAC modeling, the strength properties for the Upper Lithophysal litho-stratigraphic unit was assumed to be the same as the Topopah Spring Welded Tuff Thermal-Mechanical Unit 1 and the Middle Nonlithophysal litho-stratigraphic unit was the same as the Topopah Spring Welded Tuff Thermal-Mechanical Unit 2. However, the Lower Lithophysal litho-stratigraphic unit might not be as strong as the Middle Nonlithophysal litho-stratigraphic unit even though these two units were grouped in the same thermal-mechanical unit (Topopah Spring Welded Tuff Thermal-Mechanical Unit 2). The Lower Lithophysal litho-stratigraphic unit contains voids of various sizes as opposed to the Middle Nonlithophysal litho-stratigraphic unit which does not contain large voids. It is, therefore, reasonable to assume that the former is relatively weaker than the latter. Consequently, using the same strength properties as the Middle Nonlithophysal unit for the Lower Lithophysal unit may not be appropriate. No strength data for the Lower Lithophysal litho-stratigraphic unit is readily available; therefore, the strength properties for the Topopah Spring Welded Tuff Thermal-Mechanical Unit 1 were used for the purpose of this study.

Table 3-2. Rock-Mass Young's Modulus (CRWMS M&O, 1999)

Thermal-Mechanical Unit	Rock-Mass Quality Categotry	Cumulative Frequency of Occurrence	Rock-Mass Young's Modulus, GPa [10^6 psi]
Topopah Spring Welded Tuff Unit 1	1 (Sensitivity Case 5)	5%	9.03 [1.31]
	2 (Base Case 2)	20%	14.28 [2.07]
	3	40%	19.40 [2.81]
	4	70%	20.36 [2.95]
	5 (Sensitivity Case 6)	90%	20.36 [2.95]
Topopah Spring Welded Tuff Unit 2	1 (Sensitivity Case 5)	5%	8.98 [1.30]
	2 (Base Case 2)	20%	12.02[1.74]
	3	40%	14.77 [2.14]
	4	70%	18.92 [2.74]
	5 (Sensitivity Case 6)	90%	24.71 [5.58]

Table 3-4. Rock-Mass Strength Properties (CRWMS M&O, 1999)

Thermal-Mechanical Unit	Rock-Mass Quality Category	Cumulative Frequency of Occurrence	Cohesion, MPa [psi]	Tensile Strength, MPa [psi]	Friction Angle, degree	Dilation Angle, degree
Topopah Spring Welded Tuff Unit 1	1 (Sensitivity Case 5)	5%	1.1 [159.5]	0.90 [130.5]	44	22
	2 (Base Case 2)	20%	1.4 [203.0]	1.13 [163.9]	46	23
	3	40%	1.7 [246.5]	1.35 [195.8]	46	23
	4	70%	2.1 [304.5]	1.69 [245.1]	47	24
	5 (Sensitivity Case 6)	90%	2.9 [420.5]	2.26 [327.7]	47	24
Topopah Spring Welded Tuff Unit 2	1 (Sensitivity Case 5)	5%	1.9 [275.5]	1.16 [168.2]	56	28
	2 (Base Case 2)	20%	2.3 [333.5]	1.36 [197.2]	57	29
	3	40%	2.6 [377.0]	1.54 [223.3]	57	29
	4	70%	3.2 [464.0]	1.82 [263.9]	58	29
	5 (Sensitivity Case 6)	90%	3.9 [565.5]	2.22 [321.9]	58	29

Besides the two base cases, several sensitivity analyses were also performed. The sensitivity analyses for Base Case 1 focused on investigating the effects of intact rock Young's modulus and strength properties (mainly cohesion and tensile strength) on displacements and permeability changes at predetermined locations.

Swi-Min Hsiung 8/2/2002

Table 3-6 shows the sensitivity analysis cases studied along with the parameters varied from Base Case 1. The parameter values shown in the table were varied from the mean values used in Base Case 1 either one standard deviation greater or smaller to examine the potential effects of uncertainties associated with the determination of these parameters. As can be observed in Table 3-3, standard deviation is not available for intact rock cohesion, C_o. In this case, uniaxial compressive strength, σ_c, provided in Table 3-4 was used to estimate cohesion, C_o, for the three litho-stratigraphic units modeled using the following equation

$$C_o = \frac{\sigma_c(1-\sin\phi)}{2\cos\phi} \qquad (3\text{-}3)$$

where ϕ is intact rock friction angle.

For Base Case 2, sensitivity of rock-mass quality to the displacement and permeability under heated condition were analyzed by considering Rock-Mass Quality Categories 1 and 5 (Sensitivity Cases 5 and 6), respectively. The material and strength properties for these two rock-mass quality categories were listed in Table 3-2 and 3-4.

Table 3-6. Cases of Sensitivity Analyses With Respect To Base Case 1

Case	Rock Unit	Young's Modulus, GPa [10^{-6} psi]	Cohesion, MPa [psi]	Tensile Strength, MPa [psi]
Sensitivity Case 1	Upper Lithophysal	13.61 [1.97]	Same As Base Case 1	Same As Base Case 1
	Middle Nonlithophysal	27.09 [3.93]	Same As Base Case 1	Same As Base Case 1
	Lower Lithophysal	27.09 [3.93]	Same As Base Case 1	Same As Base Case 1
Sensitivity Case 2	Upper Lithophysal	27.11 [3.93]	Same As Base Case 1	Same As Base Case 1
	Middle Nonlithophysal	38.97 [5.65]	Same As Base Case 1	Same As Base Case 1
	Lower Lithophysal	38.97 [5.65]	Same As Base Case 1	Same As Base Case 1

Sensitivity Cases 3 and 4 are listed on next page

Sensitivity Case 3	Lower Lithophysal	13.61 [1.97]	5.36 [777.20]	3.16 [458.20]
	Middle Nonlithophysal	27.09 [3.93]	19.57 [2,837.65]	5.52 [800.40]
	Lower Lithophysal	27.09 [3.93]	5.36 [777.2]	3.16 [458.2]
Sensitivity Case 4	Lower Lithophysal	27.11 [3.93]	17.33 [2,512.85]	5.48 [794.60]
	Middle Nonlithophysal	38.97 [5.65]	44.63 [6,471.35]	12.30 [1,783.50]
	Lower Lithophysal	38.97 [5.65]	17.33 [2,512.85]	5.48 [794.60]

Comparison of Tables 3-3 and 3-4 indicates that the rock-mass friction angles for the Topopah Spring Welded Tuff Thermal-Mechanical Unit 2 appear to be larger than the intact rock friction angle. It is difficult to imagine that the rock-mass friction angle can be larger than the intact one. Investigation in this area is not abundantly available. In this study, if the rock-mass friction angle to be used for analysis was larger than the intact rock friction angle, the intact rock friction angle was used to represent the rock-mass friction angle. In a similar fashion, the rock-mass dilation angle was also adjusted to represent the intact rock dilation angle to be consistent.

Thermal Expansion Coefficient

The thermal expansion coefficients for the three litho-stratigraphic units modeled in this study are provided in Table 3-5 (CRWMS M&O, 1997c). These data were determined based on laboratory experiments. It can be observed that the thermal expansion coefficient for each of the litho-stratigraphic units are temperature dependent. The thermal expansion coefficient varies more than a factor of 5 when the temperature is increased from 25 to 300 °C [77 to 572 °F].

The temperature-dependent nature of the thermal expansion coefficients listed in Table 3-5 were used in the FLAC modeling. Since the thermal expansion coefficient was expressed as a step function, a 1-°C gap between steps was provided as FLAC input. Within this gap, the thermal expansion coefficient was linear interpolated using the step values at the two ends. Also, in this study, the thermal expansion coefficient for both intact rock and rock-mass were assumed the same. The thermal expansion coefficient corresponding to temperature higher than 300 °C [572 °F] is not available; the value associated with 275–300 °C [527–572 °F] was used in the analysis.

The corresponding FISH function using thermal expansion coefficients listed in Table 3-5 on p.59 is provided on p.37 & 38.

Table 3-5 Thermal Expansion Coefficient Data (10^{-5}/°C) (CRWMS M&O, 1999)

Litho-Stratigraphic Unit	25–50 °C	50–75 °C	75–100 °C	100–125 °C	125–150 °C	150–175 °C
Upper Lithophysal	7.41	8.43	8.89	9.52	10.86	13.51
Middle Nonlithophysal	6.89	8.45	8.95	9.50	10.12	10.95
Lower Lithophysal	6.41	8.15	8.77	9.12	9.87	10.75
Litho-Stratigraphic Unit	**175–200 °C**	**200–225 °C**	**225–250 °C**	**250–275 °C**	**275–300 °C**	
Upper Lithophysal	19.38	29.34	32.35	40.16	48.83	
Middle Nonlithophysal	12.09	14.57	19.45	27.24	41.56	
Lower Lithophysal	12.55	15.14	25.19	26.15	33.40	

In the FLAC analyses, temperature distribution at cross section 23 m from the thermal bulkhead was used.

Figure 3-6 shows the contours of the temperature distribution on a vertical cross section at 23 m [75.5 ft] from the thermal bulkhead of the heated drift after 4 years of heating. The two high temperature concentration zones (with temperatures higher than 250 °C [482 °F] in Figure 3-6) appeared to be at locations near the outer wing heaters on both side of the heater drift. However, it is interesting to note that these two zones were located somewhat beneath the outer wing heaters which were 0.25 m [0.82 ft] below the springline. The zones with the highest temperatures were approximately 2 m [6.56 ft] below the springline. Intuitively, these zones should be at or very close to the wing heaters. It is not clear why the temperature shift took place. Figure 3-7 shows the same contours as in Figure 3-6 except with a shift of 2 m [6.56 ft] upwards. Notice that the zones of highest temperature (> 300 °C [572 °F]) were located approximately at the springline. The temperature distribution pattern shown in figure 3-7 appears to be consistent with those predicted temperature distribution results presented in Task 2A[2]. In the interest of this study, temperature distributions shown in both figures were used for the study and the possible effects these two distributions analyzed. To facilitate discussions on modeling results in the latter section, the use of temperature distribution without shifting (Figure 3-6) was named Temperature Option 1 and the use of temperature distribution with location shifting (Figure 3-7) was called Temperature Option 2.

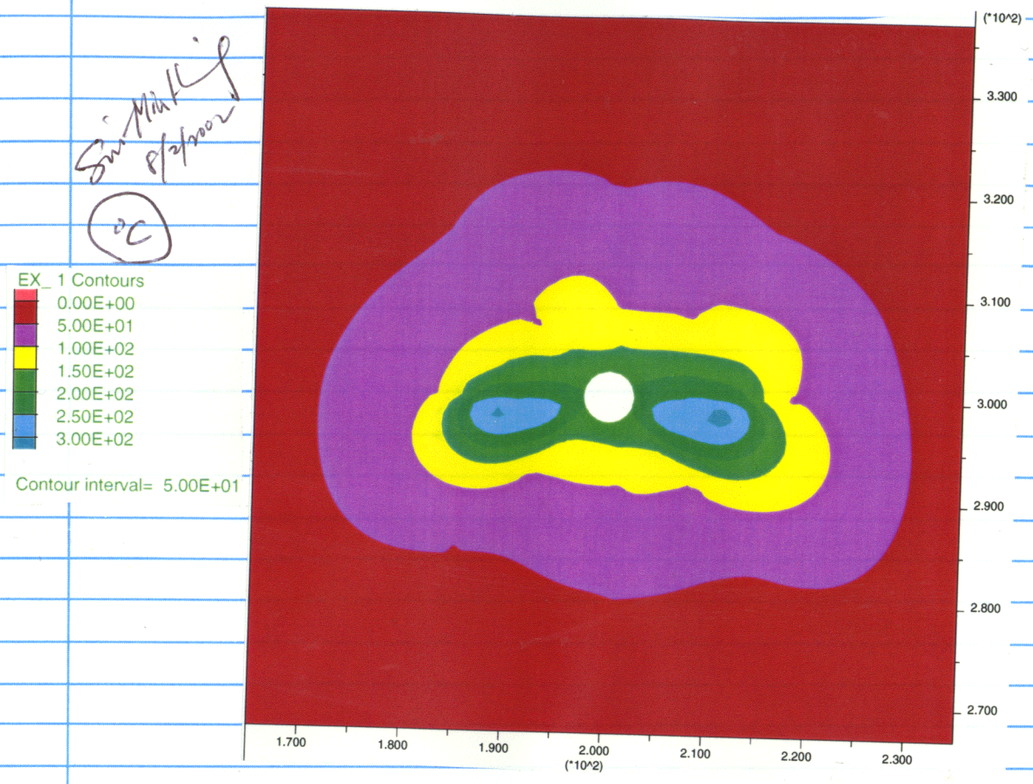


Figure 3-6. Temperature contour, 23m from the thermal bulkhead after 4 years of heating

Sin-Min Hsiung 8/2/2002

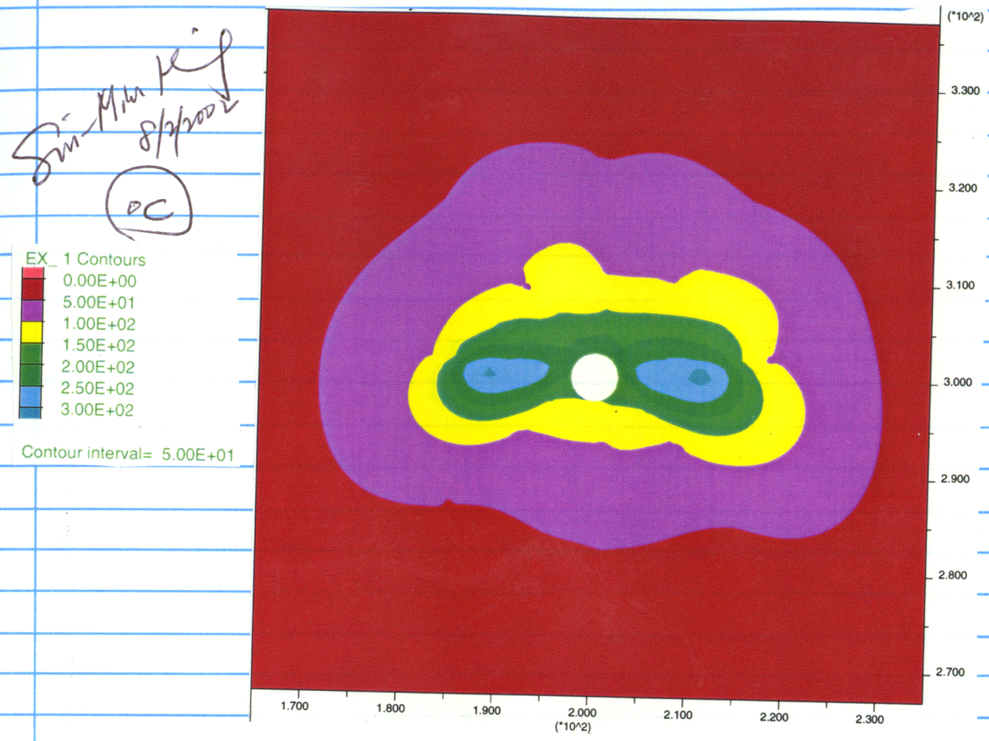


Figure 3-7. Temperature contour, 23 m from the thermal bulkhead after 4 years of heating (temperature distribution was shifted upward by 2m). The ambient temperature is assumed to be 24°C for the analysis

Sin-Min Hsiung 8/2/2002

NUMERICAL PREDICTION OBJECTIVE

numerical predictions are made on (i) rock-mass displacements at locations coincide with the anchors of four multiple-position extensometers located at the cross-section approximately 21 m [68.9 ft] from the thermal bulkhead and (ii) permeabilities variations at locations coincide with the locations of the pressure sensors for hydrologic boreholes, two at the cross-section approximately 10 m [32.1 ft] and the other at the cross-section approximately 30 m [98.4 ft] from the thermal bulkhead.

Sin-Min Hsiung 8/2/2002

Figures 3-3 and 3-4 illustrate the relative locations of hydrologic boreholes and multiple-position extensometers of interest to the heated drift. Also shown in the figures are locations of pressure sensors and anchors in each of the boreholes. The anchors for the all multiple-position extensometers were numbered from 1 to 4 with the anchors with smaller numbers closer to the drift wall than those with large numbers. A similar numbering system was used also for the pressure sensors in the each hydrologic boreholes. Contrary to the multiple-position extensometers, the pressure sensors with larger numbers were located closer to the drift than the sensors with smaller numbers.

Note that hydrologic boreholes 57 and 59 shown in Figure 3-3 were located 10 m [32.1 ft] and hydrologic boreholes 74 and 76 were location 30 m [98.4 ft] from the thermal bulkhead. For the four extensometers shown in Figure 3-4, two were vertically oriented (one in the crown and one in the invert), and the remaining two were inclined at approximately 30° to the vertical extensometer on either side of the vertical extensometer.

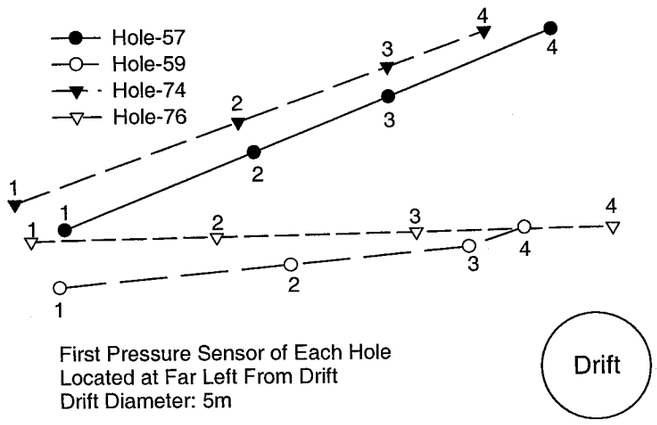


Figure 3-3. Relative location of hydrological measurement holes with respect to drift

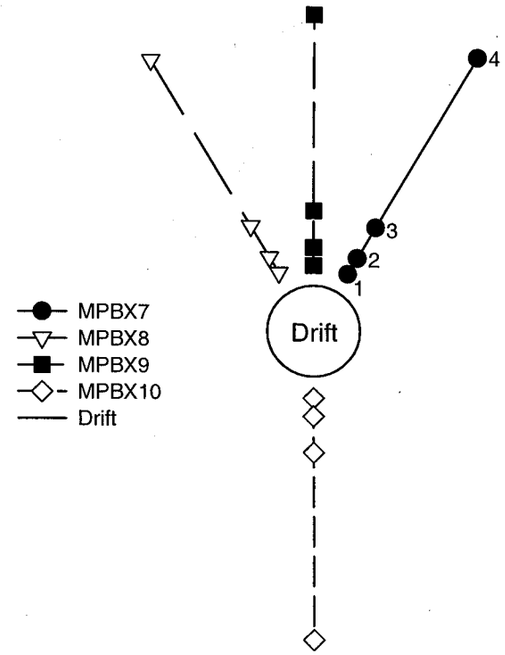
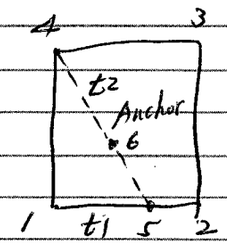


Figure 3-4 Relative positions of multiple-position extensometer anchors with respect to drift

The coordinates of the anchors for the four multiple-position extensometers are shown on p.7. The FLAC models used in the analyses have a regular grid structure (1-m in space between grids) except near the excavation where grid structure is irregular for the purpose of describing the shape of the excavation better. As a result, the anchors for the extensometers are not located on the grids. In order to assess the displacements of these anchors due to heating, a FISH function was developed to tracking the relative position of an anchor

point to the grids near by. There are two possibilities on the anchor located; (i) anchor located inside a zone and (ii) anchor located on edge of a zone. — L 8/3/2002

For the case that an anchor is located inside a zone, anchor location can be determined using the following approach

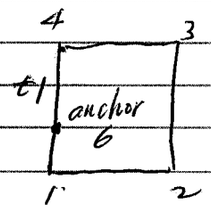


$t1 = \frac{T5}{T2}$ & $t2 = \frac{T6}{T5}$

By tracking — L 8/3/2002

Assuming that t1 and t2 are constant after deformation. Anchor 6 after deformation can be determined.

For the case that an anchor is on an edge of a zone.



$t1 = \frac{46}{41}$

Anchor 6 location can be determined for this case by assuming that t1 remains the same after deformation.

The assumption used for the two cases discussed on p.64 is reasonable if deformation is small.

A computer program was developed to calculate t1 & t2. The listing of this program is provided below.

```
//---------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "ReadASCII.h"
#include "ReadingFile.h"
//---------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TReadTextFile *ReadTextFile;
//---------------------------------------------------------------------
__fastcall TReadTextFile::TReadTextFile(TComponent* Owner)
    : TForm(Owner),Index(0)
{
}
//---------------------------------------------------------------------

void __fastcall TReadTextFile::DoneClick(TObject *Sender)
{
  Close();
}
//---------------------------------------------------------------------

void __fastcall TReadTextFile::BitBtn1Click(TObject *Sender)
{
  ThreadRead=new threadRead(false);
}
//---------------------------------------------------------------------

void TReadTextFile::Loc()
{
  anchor.clear();
  k0.clear();
  d.clear();
  grid.clear();
  rate1.clear(); // zone edge
  rate2.clear(); // line includes anchor
  // MPBX-7, anchor coords
  anchor.push_back(LDPoint(201.271,302.164));
  anchor.push_back(LDPoint(201.842,303.135));
  anchor.push_back(LDPoint(202.348,303.997));
  anchor.push_back(LDPoint(203.336,305.678));
  anchor.push_back(LDPoint(208.804,314.992));
```

```
// MPBX-8
anchor.push_back(LDPoint(198.721,302.160));
anchor.push_back(LDPoint(198.133,303.199));
anchor.push_back(LDPoint(197.628,304.051));
anchor.push_back(LDPoint(196.619,305.755));
anchor.push_back(LDPoint(191.243,314.832));
// MPBX-9
anchor.push_back(LDPoint(200.0,302.510));
anchor.push_back(LDPoint(200.0,303.604));
anchor.push_back(LDPoint(200.0,304.604));
anchor.push_back(LDPoint(200.0,306.604));
anchor.push_back(LDPoint(200.0,317.354));

// MPBX-10
anchor.push_back(LDPoint(200.0,297.490));
anchor.push_back(LDPoint(200.0,296.316));
anchor.push_back(LDPoint(200.0,295.326));
anchor.push_back(LDPoint(200.0,293.326));
anchor.push_back(LDPoint(200.0,283.076));
int J1(0),J3(4),J2(0);
for(int I(0); I<20; I++)
//  for(int I(0); I<10; I++)
 {
   J2=J3*(I+1)-1;
   k0.push_back(TPoint(J1,J2));
   J1=J2+1;
 } // for(int I(0); I<20; I++)

// MPBX-7, grid
preparedata(201,302);
preparedata(201,303);
preparedata(202,303);
preparedata(203,305);
preparedata(208,314);

// MPBX-8
preparedata(198,302);
preparedata(198,303);
preparedata(197,304);
preparedata(196,305);
preparedata(191,314);

// MPBX-9
preparedata(200,302);
preparedata(200,303);
preparedata(200,304);
preparedata(200,306);
preparedata(200,317);

// MPBX-10
preparedata(200,296);
preparedata(200,296);
preparedata(200,295);
preparedata(200,293);
preparedata(200,283);
```



```
//  grid coords, Intact Rock, Low E
d.push_back(LDPoint(2.0111809E+02,3.0223697E+02));
d.push_back(LDPoint(2.0176785E+02,3.0176880E+02));
d.push_back(LDPoint(2.0200000E+02,3.0300112E+02));
d.push_back(LDPoint(2.0100000E+02,3.0300097E+02));
d.push_back(LDPoint(2.0100000E+02,3.0300097E+02));
d.push_back(LDPoint(2.0200000E+02,3.0300112E+02));
d.push_back(LDPoint(2.0199998E+02,3.0400115E+02));
d.push_back(LDPoint(2.0099999E+02,3.0400106E+02));
d.push_back(LDPoint(2.0200000E+02,3.0300112E+02));
d.push_back(LDPoint(2.0300002E+02,3.0300125E+02));
d.push_back(LDPoint(2.0299999E+02,3.0400124E+02));
d.push_back(LDPoint(2.0199998E+02,3.0400115E+02));
d.push_back(LDPoint(2.0299998E+02,3.0500125E+02));
d.push_back(LDPoint(2.0399999E+02,3.0500131E+02));
d.push_back(LDPoint(2.0399998E+02,3.0600131E+02));
d.push_back(LDPoint(2.0299998E+02,3.0600127E+02));
d.push_back(LDPoint(2.0799998E+02,3.1400145E+02));
d.push_back(LDPoint(2.0899998E+02,3.1400146E+02));
d.push_back(LDPoint(2.0899998E+02,3.1500146E+02));
d.push_back(LDPoint(2.0799998E+02,3.1500146E+02));
d.push_back(LDPoint(1.9823216E+02,3.0176880E+02));
d.push_back(LDPoint(1.9888192E+02,3.0223697E+02));
d.push_back(LDPoint(1.9900000E+02,3.0300097E+02));
d.push_back(LDPoint(1.9800000E+02,3.0300111E+02));
d.push_back(LDPoint(1.9800000E+02,3.0300111E+02));
d.push_back(LDPoint(1.9900000E+02,3.0300097E+02));
d.push_back(LDPoint(1.9900002E+02,3.0400106E+02));
d.push_back(LDPoint(1.9800002E+02,3.0400115E+02));
d.push_back(LDPoint(1.9700001E+02,3.0400124E+02));
d.push_back(LDPoint(1.9800002E+02,3.0400115E+02));
d.push_back(LDPoint(1.9800002E+02,3.0500119E+02));
d.push_back(LDPoint(1.9700002E+02,3.0500125E+02));
d.push_back(LDPoint(1.9600001E+02,3.0500131E+02));
d.push_back(LDPoint(1.9700002E+02,3.0500125E+02));
d.push_back(LDPoint(1.9700002E+02,3.0600127E+02));
d.push_back(LDPoint(1.9600002E+02,3.0600131E+02));
d.push_back(LDPoint(1.9100002E+02,3.1400146E+02));
d.push_back(LDPoint(1.9200002E+02,3.1400145E+02));
d.push_back(LDPoint(1.9200002E+02,3.1500146E+02));
d.push_back(LDPoint(1.9100002E+02,3.1500147E+02));
d.push_back(LDPoint(2.0000000E+02,3.0250082E+02));
d.push_back(LDPoint(2.0111809E+02,3.0223697E+02));
d.push_back(LDPoint(2.0100000E+02,3.0300097E+02));
d.push_back(LDPoint(2.0000000E+02,3.0300090E+02));
d.push_back(LDPoint(2.0000000E+02,3.0300090E+02));
d.push_back(LDPoint(2.0100000E+02,3.0300097E+02));
d.push_back(LDPoint(2.0099999E+02,3.0400106E+02));
d.push_back(LDPoint(2.0000000E+02,3.0400103E+02));
d.push_back(LDPoint(2.0000000E+02,3.0400103E+02));
d.push_back(LDPoint(2.0099999E+02,3.0400106E+02));
d.push_back(LDPoint(2.0099999E+02,3.0500114E+02));
d.push_back(LDPoint(2.0000000E+02,3.0500111E+02));
d.push_back(LDPoint(2.0000000E+02,3.0600118E+02));
d.push_back(LDPoint(2.0099999E+02,3.0600119E+02));
d.push_back(LDPoint(2.0099999E+02,3.0700123E+02));
d.push_back(LDPoint(2.0000000E+02,3.0700122E+02));
d.push_back(LDPoint(2.0000000E+02,3.1700143E+02));
```

This position should be changed for different FLAC models, especially when the extensometers will be installed after excavation.

```
d.push_back(LDPoint(2.0100000E+02,3.1700143E+02));
d.push_back(LDPoint(2.0100000E+02,3.1800144E+02));
d.push_back(LDPoint(2.0000000E+02,3.1800144E+02));
d.push_back(LDPoint(2.0000000E+02,2.9600190E+02));
d.push_back(LDPoint(2.0099998E+02,2.9600186E+02));
d.push_back(LDPoint(2.0057440E+02,2.9756895E+02));
d.push_back(LDPoint(2.0000000E+02,2.9750209E+02));
d.push_back(LDPoint(2.0000000E+02,2.9600190E+02));
d.push_back(LDPoint(2.0099998E+02,2.9600186E+02));
d.push_back(LDPoint(2.0057440E+02,2.9756895E+02));
d.push_back(LDPoint(2.0000000E+02,2.9750209E+02));
d.push_back(LDPoint(2.0000000E+02,2.9500181E+02));
d.push_back(LDPoint(2.0099998E+02,2.9500179E+02));
d.push_back(LDPoint(2.0099998E+02,2.9600186E+02));
d.push_back(LDPoint(2.0000000E+02,2.9600190E+02));
d.push_back(LDPoint(2.0000000E+02,2.9300170E+02));
d.push_back(LDPoint(2.0099999E+02,2.9300169E+02));
d.push_back(LDPoint(2.0099999E+02,2.9400173E+02));
d.push_back(LDPoint(2.0000000E+02,2.9400175E+02));
d.push_back(LDPoint(2.0000000E+02,2.8300148E+02));
d.push_back(LDPoint(2.0100000E+02,2.8300148E+02));
d.push_back(LDPoint(2.0100000E+02,2.8400149E+02));
d.push_back(LDPoint(2.0000000E+02,2.8400149E+02));

int ISize(10);
int i1,i2;
long double x,y;
RichEdit1->Clear();
// anchors in zone
for(int I(0); I<ISize; I++)
{
  i1=k0[I].x;
  i2=k0[I].y;
  for(int J(i1); J<i2-1; J++)
  {
    x2x1(d[i2].x,d[i2].y,anchor[I].x,anchor[I].y,
         d[J].x,d[J].y,d[J+1].x,d[J+1].y);
    IntersectionPoint();
    if(k7==0) continue;
    rate1.push_back(LDLine(grid[J],grid[J+1],t2));
    x=d[J].x+(d[J+1].x-d[J].x)*t2;
    y=d[J].y+(d[J+1].y-d[J].y)*t2;
    long double d1,d2;
    d1=sqrt1((d[i2].x-x)*(d[i2].x-x)+(d[i2].y-y)*(d[i2].y-y));
    d2=sqrt1((d[i2].x-anchor[I].x)*(d[i2].x-anchor[I].x)+
             (d[i2].y-anchor[I].y)*(d[i2].y-anchor[I].y));
    t1=d2/d1;
    rate2.push_back(LDIPoint(grid[i2].x,grid[i2].y,t1));
  } // for(int J(i1); J<i2-1; J++)
  RichEdit1->Lines->Add(FloatToStrF(rate1[I].P1.x,0,7,6)+" "+
    FloatToStrF(rate1[I].P1.y,0,7,6)+" "+FloatToStrF(rate1[I].P2.x,0,7,6)+" "+
    FloatToStrF(rate1[I].P2.y,0,7,6)+" "+FloatToStrF(rate1[I].I,0,7,6));
  RichEdit1->Lines->Add(FloatToStrF(rate2[I].I,0,7,6)+" "+
    FloatToStrF(rate2[I].x,0,7,6)+" "+FloatToStrF(rate2[I].y,0,7,6));
} // for(int I(0); I<ISize; I++)
// prepare FISH for extensometer anchors in finite difference zones
Editor->Lines->Add(";");
Editor->Lines->Add(";First of the two FISH functions");
Editor->Lines->Add(";extensometer anchors in finite difference zone");
```



```
Editor->Lines->Add(";MPBX7 and MPBX8");
Editor->Lines->Add(";(mpbx7_8e(1,1),mpbx9_10e(1,2)):first  grid");
Editor->Lines->Add(";(mpbx7_8e(1,3),mpbx9_10e(1,4)):second grid");
Editor->Lines->Add("; mpbx7_8e(1,5):intersection point");
Editor->Lines->Add(";");
Editor->Lines->Add("def MPBX7_8");
Editor->Lines->Add("  array mpbx7_8e(10,5)");
Editor->Lines->Add("  array mpbx7_8z(10,3)");
for(int I(0); I<ISize; I++)
{
  Editor->Lines->Add("  mpbx7_8e("+IntToStr(I+1)+",1)="+
    FloatToStrF(rate1[I].P1.x,0,7,6));
  Editor->Lines->Add("  mpbx7_8e("+IntToStr(I+1)+",2)="+
    FloatToStrF(rate1[I].P1.y,0,7,6));
  Editor->Lines->Add("  mpbx7_8e("+IntToStr(I+1)+",3)="+
    FloatToStrF(rate1[I].P2.x,0,7,6));
  Editor->Lines->Add("  mpbx7_8e("+IntToStr(I+1)+",4)="+
    FloatToStrF(rate1[I].P2.y,0,7,6));
  Editor->Lines->Add("  mpbx7_8e("+IntToStr(I+1)+",5)="+
    FloatToStrF(rate1[I].I   ,0,7,6));
//  Editor->Lines->Add(FloatToStrF(rate1[I].P1.x,0,7,6)+" "+
//    FloatToStrF(rate1[I].P1.y,0,7,6)+" "+FloatToStrF(rate1[I].P2.x,0,7,6)+" "+
//    FloatToStrF(rate1[I].P2.y,0,7,6)+" "+FloatToStrF(rate1[I].I,0,7,6));
} // for(int I(0); I<ISize; I++)
Editor->PlainText = true;
Editor->Lines->SaveToFile("AnchorInZone.fis");
Editor->Clear();
Editor->Lines->Add(";");
Editor->Lines->Add(";(mpbx7_8z(1,1),mpbx7_8z(1,2)):grid");
Editor->Lines->Add("; mpbx7_8z(1,3):relative location of anchor");
Editor->Lines->Add(";");
for(int I(0); I<ISize; I++)
{
  Editor->Lines->Add("  mpbx7_8z("+IntToStr(I+1)+",1)="+
    FloatToStrF(rate2[I].I,0,7,6));
  Editor->Lines->Add("  mpbx7_8z("+IntToStr(I+1)+",2)="+
    FloatToStrF(rate2[I].x,0,7,6));
  Editor->Lines->Add("  mpbx7_8z("+IntToStr(I+1)+",3)="+
    FloatToStrF(rate2[I].y,0,7,6));
//  Editor->Lines->Add(FloatToStrF(rate2[I].I,0,7,6)+" "+
//    FloatToStrF(rate2[I].x,0,7,6)+" "+FloatToStrF(rate2[I].y,0,7,6));
} // for(int I(0); I<ISize; I++)
Editor->Lines->Add("end");
Editor->Lines->Add("MPBX7_8");
Editor->Lines->SaveToFile("AnchorInZone1.fis");
Editor->Clear();
ISize=k0.size();
rate1_1.clear(); // zone edge
int IC(-1);
```

Sui-Min Hsiung 8/4/2002

```
// anchors on zone edge
for(int I(10); I<ISize; I++)
{
  i1=k0[I].x;
  i2=k0[I].y;
  long double d1,d2;
  d1=sqrtl((d[i2].x-d[i1].x)*(d[i2].x-d[i1].x)+
           (d[i2].y-d[i1].y)*(d[i2].y-d[i1].y));
  d2=sqrtl((d[i2].x-anchor[I].x)*(d[i2].x-anchor[I].x)+
           (d[i2].y-anchor[I].y)*(d[i2].y-anchor[I].y));
  t1=d2/d1;
  rate1_1.push_back(LDLine(grid[i2],grid[i1],t1));
  IC++;
  RichEdit1->Lines->Add(FloatToStrF(rate1_1[IC].P1.x,0,7,6)+" "+
    FloatToStrF(rate1_1[IC].P1.y,0,7,6)+" "+FloatToStrF(rate1_1[IC].P2.x,0,7,6)+
    " "+FloatToStrF(rate1_1[IC].P2.y,0,7,6)+
    " "+FloatToStrF(rate1_1[IC].I,0,7,6));
} // for(int I(0); I<ISize; I++)
// prepare FISH for extensometer anchors on zone edge
Editor->Lines->Add(";");
Editor->Lines->Add(";second of the two FISH functions");
Editor->Lines->Add(";extensometer anchors on zone edge");
Editor->Lines->Add(";MPBX9 and MPBX10");
Editor->Lines->Add(";(mpbx9_10e(1,1),mpbx9_10e(1,2)):first  grid");
Editor->Lines->Add(";(mpbx9_10e(1,3),mpbx9_10e(1,4)):second grid");
Editor->Lines->Add(";mpbx9_10e(1,5):relative location of anchor");
Editor->Lines->Add(";");
Editor->Lines->Add("def MPBX9_10");
Editor->Lines->Add("  array mpbx9_10e(10,5)");
for(int I(0); I<=IC; I++)
{
  Editor->Lines->Add("  mpbx9_10e("+IntToStr(I+1)+",1)="+
    FloatToStrF(rate1_1[I].P1.x,0,7,6));
  Editor->Lines->Add("  mpbx9_10e("+IntToStr(I+1)+",2)="+
    FloatToStrF(rate1_1[I].P1.y,0,7,6));
  Editor->Lines->Add("  mpbx9_10e("+IntToStr(I+1)+",3)="+
    FloatToStrF(rate1_1[I].P2.x,0,7,6));
  Editor->Lines->Add("  mpbx9_10e("+IntToStr(I+1)+",4)="+
    FloatToStrF(rate1_1[I].P2.y,0,7,6));
  Editor->Lines->Add("  mpbx9_10e("+IntToStr(I+1)+",5)="+
    FloatToStrF(rate1_1[I].I   ,0,7,6));
//  Editor->Lines->Add(FloatToStrF(rate1_1[I].P1.x,0,7,6)+" "+
//    FloatToStrF(rate1_1[I].P1.y,0,7,6)+" "+FloatToStrF(rate1_1[I].P2.x,0,7,6)+
//    " "+FloatToStrF(rate1_1[I].P2.y,0,7,6)+
//    " "+FloatToStrF(rate1_1[I].I,0,7,6));
} // for(int I(0); I<=IC; I++)
Editor->Lines->Add("end");
Editor->Lines->Add("MPBX9_10");
Editor->Lines->SaveToFile("AnchorOnEdge.fis");
delete Editor;
// Editor->Clear();
RichEdit1->Lines->Add("Task Completed");
}
```

Sim Mk Hip 8/4/2002


```
//---------------------------------------------------------------

void TReadTextFile::preparedata(int a,int b)
{
/*  d.push_back(LDPoint(a   ,b   ));
  d.push_back(LDPoint(a+1,b   ));
  d.push_back(LDPoint(a+1,b+1));
  d.push_back(LDPoint(a   ,b+1)); */
  grid.push_back(LDPoint(a+1,b+1));
  grid.push_back(LDPoint(a+2,b+1));
  grid.push_back(LDPoint(a+2,b+2));
  grid.push_back(LDPoint(a+1,b+2));
}
//---------------------------------------------------------------

      /*************************************************/
      /* IntersectionPoint: intersection point of two  */
      /*                    lines                      */
      /*************************************************/
      /* x21 y21 x31 y31 x34 y34 x41 y41               */
      /*  k3  t1  t2              d0  d1  d2  a1  b1   */
      void TReadTextFile::IntersectionPoint()
      {
      /*-----------------------------------------------*/
      /* k3=0 no intersection  k3=1 output t1 t2 par   */
      k7=0;
      long double d0, d1, d2, c0;
      long double a1, b1;
      t1=100.0; t2=100.0;
      d0=x21*y34-x34*y21;             /* equation | |   */
      d1=x31*y34-x34*y31;             /* | | for sol t1 */
      d2=x21*y31-x31*y21;             /* | | for sol t2 */
      /*-----------------------------------------------*/
      /* p1 p2 p3 p4  4 points co-line                 */
      if ( fabsl(d0)<0.0000001 &&
           fabsl(d1)<0.0000001    )  goto ln02;
      /*-----------------------------------------------*/
      /* p1 p2   p3 p4  are two parallel lines         */
      if ( fabsl(d0)<0.0000001    )  goto ln01;
      goto ln03;
      /*-----------------------------------------------*/
      /* p1 p2 p3 p4  4 points in same line            */
ln02:;
      a1=x21;
      b1=x31;
      if ( fabsl(x21) > fabsl(y21) ) goto ln05;
      a1=y21;
      b1=y31;
ln05:;
      /*-----------------------------------------------*/
      /* (x3 y3) lies in p1 p2 segment                 */
      t1=b1/a1;
      if ( (t1<-0.000001) || (t1>1.000001) ) goto ln04;
      t2=0;
      k7=1;
      /*************************************************/
```

```
ln04:;
     a1=x21;
     b1=x41;
     if ( fabsl(x21) > fabsl(y21) ) goto ln06;
     a1=y21;
     b1=y41;
     /*-----------------------------------------------*/
     /* (x4 y4) lies in p1 p2 segment                 */
     /* both p3 p4   in p1 p2 segment no intersection */
ln06:;
     c0=b1/a1;
     if ( (c0<-0.000001) || (c0>1.000001) ) goto ln01;
     t1=b1/a1;
     t2=1;
     k7 += 1;
     if ( k7 == 2 )  k7=0;
     goto ln01;
     /*-----------------------------------------------*/
     /* normal intersection                           */
ln03:;
     t1=d1/d0;
//     if ( (t1<-0.000001) || (t1>1.000001) ) goto ln01;
     t2=d2/d0;
     if ( (t2<-0.000001) || (t2>1.000001) ) goto ln01;
     k7=1;
ln01:;
     } // void TReadTextFile::IntersectionPoint()

     /*************************************************/
     /* x2x1: prepare for compute intersection        */
     /*************************************************/
     void TReadTextFile::x2x1(long double x1, long double yi, long double x2,
                              long double y2, long double x333, long double y333,
                              long double x444, long double y444)
     {
     x21=x2-x1;
     y21=y2-yi;
     x31=x333-x1;
     y31=y333-yi;
     x34=x333-x444;
     y34=y333-y444;
     x41=x444-x1;
     y41=y444-yi;
     } //  void TReadTextFile::x2x1()
```

```
//---------------------------------------------

#ifndef ReadASCIIH
#define ReadASCIIH
//---------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Buttons.hpp>
#include <ExtCtrls.hpp>
#include <ComCtrls.hpp>
```



```
#include <Dialogs.hpp>
#include <iostream>
#include <string>
#include <istream>
#include <fstream>
#include <vector>
#include <stdio.h>
#include <stdarg.h>
#include "ddamatrix.h"
#include "ReadingFile.h"
#include "TPoints.h"

using namespace std;
//---------------------------------------------------------------------------
class TReadTextFile : public TForm
{
__published:    // IDE-managed Components
   TPanel *Panel1;
   TBitBtn *BitBtn1;
   TBitBtn *Done;
   TRichEdit *RichEdit1;
   TOpenDialog *OpenDialog1;
   TRichEdit *Editor;
   void __fastcall DoneClick(TObject *Sender);
   void __fastcall BitBtn1Click(TObject *Sender);
private:    // User declarations
   bool Index;
public:     // User declarations
   __fastcall TReadTextFile(TComponent* Owner);
   threadRead* ThreadRead;
   void preparedata(int a,int b);
   void Loc();
   void x2x1(long double x1   , long double yi   , long double x2,
             long double y2   , long double x333, long double y333,
             long double x444, long double y444);
   void IntersectionPoint();
   long double t1,t2;
   long double x21,x34,x31,x41;
   long double y21,y34,y31,y41;
   int k7;
   vector<TPoint> k0;
   vector<LDPoint> grid;
   vector<LDPoint> d;
   vector<LDPoint> anchor;
   vector<LDIPoint> rate2;
   vector<LDLine> rate1;
   vector<LDLine> rate1_1;
};
//---------------------------------------------------------------------------
extern PACKAGE TReadTextFile *ReadTextFile;
//---------------------------------------------------------------------------
#endif
```

Verification of this program is demonstrated later on p.79

Two FISH functions were developed to calculate anchor coorinates and distance between an anchor and the extensometer assembly head after deformation. The listing of these two functions are as follows.

These two functions use t1 & t2 obtained from the program listed on p. 65 as an input.

```
;
;First of the two FISH functions for anchor displacements
;extensometer anchors in finite difference zone
; intact rock
;MPBX7 and MPBX8
;
def MPBX7_8
  float $a1 $a3
  float $b1 $b3
  array mpbx7_8e(10,5)
  array mpbx7_8z(10,3)
   mpbx7_8e(1,1)=202
   mpbx7_8e(1,2)=303
   mpbx7_8e(1,3)=203
   mpbx7_8e(1,4)=303
   mpbx7_8e(1,5)=0.2602904
   mpbx7_8e(2,1)=202
   mpbx7_8e(2,2)=304
   mpbx7_8e(2,3)=203
   mpbx7_8e(2,4)=304
   mpbx7_8e(2,5)=0.9734104
   mpbx7_8e(3,1)=204
   mpbx7_8e(3,2)=304
   mpbx7_8e(3,3)=204
   mpbx7_8e(3,4)=305
   mpbx7_8e(3,5)=0.9913793
   mpbx7_8e(4,1)=205
   mpbx7_8e(4,2)=306
   mpbx7_8e(4,3)=205
   mpbx7_8e(4,4)=307
   mpbx7_8e(4,5)=0.04166667
   mpbx7_8e(5,1)=210
   mpbx7_8e(5,2)=315
   mpbx7_8e(5,3)=210
   mpbx7_8e(5,4)=316
   mpbx7_8e(5,5)=0.9900498
   mpbx7_8e(6,1)=199
   mpbx7_8e(6,2)=303
   mpbx7_8e(6,3)=200
   mpbx7_8e(6,4)=303
   mpbx7_8e(6,5)=0.7848588
   mpbx7_8e(7,1)=199
   mpbx7_8e(7,2)=304
   mpbx7_8e(7,3)=200
   mpbx7_8e(7,4)=304
   mpbx7_8e(7,5)=0.1660424
   mpbx7_8e(8,1)=198
   mpbx7_8e(8,2)=305
   mpbx7_8e(8,3)=199
   mpbx7_8e(8,4)=305
   mpbx7_8e(8,5)=0.6617492
```



```
   mpbx7_8e(9,1)=198
   mpbx7_8e(9,2)=306
   mpbx7_8e(9,3)=198
   mpbx7_8e(9,4)=307
   mpbx7_8e(9,5)=0.6042003
   mpbx7_8e(10,1)=193
   mpbx7_8e(10,2)=315
   mpbx7_8e(10,3)=193
   mpbx7_8e(10,4)=316
   mpbx7_8e(10,5)=0.308642
;
;(mpbx7_8z(1,1),mpbx7_8z(1,2)):grid
; mpbx7_8z(1,3):relative location of anchor
;
   mpbx7_8z(1,1)=202
   mpbx7_8z(1,2)=304
   mpbx7_8z(1,3)=0.9437541
   mpbx7_8z(2,1)=202
   mpbx7_8z(2,2)=305
   mpbx7_8z(2,3)=0.865
   mpbx7_8z(3,1)=203
   mpbx7_8z(3,2)=305
   mpbx7_8z(3,3)=0.348
   mpbx7_8z(4,1)=204
   mpbx7_8z(4,2)=307
   mpbx7_8z(4,3)=0.336
   mpbx7_8z(5,1)=209
   mpbx7_8z(5,2)=316
   mpbx7_8z(5,3)=0.804
   mpbx7_8z(6,1)=199
   mpbx7_8z(6,2)=304
   mpbx7_8z(6,3)=0.971457
   mpbx7_8z(7,1)=199
   mpbx7_8z(7,2)=305
   mpbx7_8z(7,3)=0.801
   mpbx7_8z(8,1)=198
   mpbx7_8z(8,2)=306
   mpbx7_8z(8,3)=0.949
   mpbx7_8z(9,1)=197
   mpbx7_8z(9,2)=307
   mpbx7_8z(9,3)=0.619
   mpbx7_8z(10,1)=192
   mpbx7_8z(10,2)=316
   mpbx7_8z(10,3)=0.243
end
;
;index point coords on zone edge
;
def interPoint7_8
  array intersect(10,2)
  loop i (1,10)
    $a1=x(mpbx7_8e(i,3),mpbx7_8e(i,4))-x(mpbx7_8e(i,1),mpbx7_8e(i,2))
    intersect(i,1)=x(mpbx7_8e(i,1),mpbx7_8e(i,2))+$a1*mpbx7_8e(i,5)
    $a1=y(mpbx7_8e(i,3),mpbx7_8e(i,4))-y(mpbx7_8e(i,1),mpbx7_8e(i,2))
    intersect(i,2)=y(mpbx7_8e(i,1),mpbx7_8e(i,2))+$a1*mpbx7_8e(i,5)
  endLoop
end
def coords7_8
```

Sui-Min Hsiung 8/4/2002

```
;
;compute anchor coords
;
  array coords1(10,2)
  loop i (1,10)
    if i=1 then
      oo=out('Coordinates of Anchors For MPBX-7')
    end_if
    if i=6 then
      oo=out('Coordinates of Anchors For MPBX-8')
    end_if
    $a1=intersect(i,1)-x(mpbx7_8z(i,1),mpbx7_8z(i,2))
    coords1(i,1)=x(mpbx7_8z(i,1),mpbx7_8z(i,2))+$a1*mpbx7_8z(i,3)
    $a1=intersect(i,2)-y(mpbx7_8z(i,1),mpbx7_8z(i,2))
    coords1(i,2)=y(mpbx7_8z(i,1),mpbx7_8z(i,2))+$a1*mpbx7_8z(i,3)
    tt1=fstring(coords1(i,1),9)
    tt2=fstring(coords1(i,2),9)
    oo=out(tt1+' '+tt2)
  endLoop
end
;
;compute relative distances between two adjacent anchors
;
def distance7_8
  array dist7_8(8)
  JJ=0
  loop i (1,2)
    II=(i-1)*5
    loop j (2,5)
      JJ=JJ+1
      KK=II+j
      $a1=(coords1(KK,1)-coords1(KK-1,1))*(coords1(KK,1)-coords1(KK-1,1))
      $a3=(coords1(KK,2)-coords1(KK-1,2))*(coords1(KK,2)-coords1(KK-1,2))
      dist7_8(JJ)=sqrt($a1+$a3)
    endLoop
  endLoop
;
;compute relative distances between anchors and the first anchor
;
  loop i (1,2)
    II=(i-1)*4
    loop j (2,4)
      KK=II+j
      dist7_8(KK)=dist7_8(KK)+dist7_8(KK-1)
    endLoop
  endLoop
;
;print out
;
  loop i (1,8)
    if i=1 then
      oo=out('Distance to the First Anchor For MPBX-7')
    end_if
    if i=5 then
      oo=out('Distance to the First Anchor For MPBX-8')
    end_if
    ttt=fstring(dist7_8(i),9)
    oo=out(ttt)
  endLoop
end
;
MPBX7_8
interPoint7_8
coords7_8
distance7_8
```



```
;
;second of the two FISH functions for anchor displacements
; intact rock
;extensometer anchors on zone edge
;MPBX9 and MPBX10
;(mpbx9_10e(1,1),mpbx9_10e(1,2)):first  grid
;(mpbx9_10e(1,3),mpbx9_10e(1,4)):second grid
; mpbx9_10e(1,5):relative location of anchor
;
def MPBX9_10
  float $a2 $a4
  array mpbx9_10e(10,5)
   mpbx9_10e(1,1)=201
   mpbx9_10e(1,2)=304
   mpbx9_10e(1,3)=201
   mpbx9_10e(1,4)=303
   mpbx9_10e(1,5)=0.98
   mpbx9_10e(2,1)=201
   mpbx9_10e(2,2)=305
   mpbx9_10e(2,3)=201
   mpbx9_10e(2,4)=304
   mpbx9_10e(2,5)=0.396
   mpbx9_10e(3,1)=201
   mpbx9_10e(3,2)=306
   mpbx9_10e(3,3)=201
   mpbx9_10e(3,4)=305
   mpbx9_10e(3,5)=0.396
   mpbx9_10e(4,1)=201
   mpbx9_10e(4,2)=308
   mpbx9_10e(4,3)=201
   mpbx9_10e(4,4)=307
   mpbx9_10e(4,5)=0.396
   mpbx9_10e(5,1)=201
   mpbx9_10e(5,2)=319
   mpbx9_10e(5,3)=201
   mpbx9_10e(5,4)=318
   mpbx9_10e(5,5)=0.646
   mpbx9_10e(6,1)=201
   mpbx9_10e(6,2)=298
   mpbx9_10e(6,3)=201
   mpbx9_10e(6,4)=297
   mpbx9_10e(6,5)=0.006666667
   mpbx9_10e(7,1)=201
   mpbx9_10e(7,2)=298
   mpbx9_10e(7,3)=201
   mpbx9_10e(7,4)=297
   mpbx9_10e(7,5)=0.7893333
   mpbx9_10e(8,1)=201
   mpbx9_10e(8,2)=297
   mpbx9_10e(8,3)=201
   mpbx9_10e(8,4)=296
   mpbx9_10e(8,5)=0.674
   mpbx9_10e(9,1)=201
   mpbx9_10e(9,2)=295
   mpbx9_10e(9,3)=201
   mpbx9_10e(9,4)=294
   mpbx9_10e(9,5)=0.674
   mpbx9_10e(10,1)=201
   mpbx9_10e(10,2)=285
   mpbx9_10e(10,3)=201
   mpbx9_10e(10,4)=284
   mpbx9_10e(10,5)=0.924
end
def coords9_10
```

Continue on next page

```
;
;anchor coords
;
  array coords2(10,2)
  loop i (1,10)
    if i=1 then
      oo=out('Coordinates of Anchors For MPBX-9')
    end_if
    if i=6 then
      oo=out('Coordinates of Anchors For MPBX-10')
    end_if
    $a1=x(mpbx9_10e(i,3),mpbx9_10e(i,4))-x(mpbx9_10e(i,1),mpbx9_10e(i,2))
    tt1=fstring($a1,9)
    oo=out(tt1)
    coords2(i,1)=x(mpbx9_10e(i,1),mpbx9_10e(i,2))+$a1*mpbx9_10e(i,5)
    $a1=y(mpbx9_10e(i,3),mpbx9_10e(i,4))-y(mpbx9_10e(i,1),mpbx9_10e(i,2))
    tt1=fstring($a1,9)
    oo=out(tt1)
    coords2(i,2)=y(mpbx9_10e(i,1),mpbx9_10e(i,2))+$a1*mpbx9_10e(i,5)
    tt1=fstring(coords2(i,1),9)
    tt2=fstring(coords2(i,2),9)
    oo=out(tt1+' '+tt2)
  endLoop
end
;
;compute relative distances between two adjacent anchors
;
def distance9_10
  array dist9_10(8)
  JJ=0
  loop i (1,2)
    II=(i-1)*5
    loop j (2,5)
      JJ=JJ+1
      KK=II+j
      $a2=(coords2(KK,1)-coords2(KK-1,1))*(coords2(KK,1)-coords2(KK-1,1))
      $a4=(coords2(KK,2)-coords2(KK-1,2))*(coords2(KK,2)-coords2(KK-1,2))
      dist9_10(JJ)=sqrt($a2+$a4)
    endLoop
  endLoop
;
;compute relative distances between anchors and the first anchor
;
  loop i (1,2)
    II=(i-1)*4
    loop j (2,4)
      KK=II+j
      dist9_10(KK)=dist9_10(KK)+dist9_10(KK-1)
    endLoop
  endLoop
;
;print out
;
  loop i (1,8)
    if i=1 then
      oo=out('Distance to the First Anchor For MPBX-9')
    end_if
    if i=5 then
      oo=out('Distance to the First Anchor For MPBX-10')
    end_if
    ttt=fstring(dist9_10(i),9)
    oo=out(ttt)
  endLoop
end
;
MPBX9_10
coords9_10
distance9_10
```


VERIFICATION OF PROGRAM CALCULATING t_1 & t_2 (listing starts on p.65) AND THE FISH FUNCTIONS

```
Anchor Coordinates with respect to drift center
MPBX7-head       1.271       2.164
MPBX7-1          1.842       3.135
MPBX7-2          2.348       3.997
MPBX7-3          3.336       5.678
MPBX7-4          8.804      14.992
MPBX8-head      -1.279       2.160
MPBX8-1         -1.867       3.199
MPBX8-2         -2.372       4.051
MPBX8-3         -3.381       5.755
MPBX8-4         -8.757      14.832
MPBX9-head      -0.010       2.510
MPBX9-1         -0.010       3.604
MPBX9-2         -0.007       4.604
MPBX9-3         -0.002       6.604
MPBX9-4          0.028      17.354
MPBX10-head      0.000      -2.510
MPBX10-1         0.000      -3.684
MPBX10-2         0.003      -4.674
MPBX10-3         0.007      -6.674
MPBX10-4         0.029     -16.924
For simplicity, actual corrdinates used in FLAC analyses
MPBX7-head       1.271       2.164
MPBX7-1          1.842       3.135
MPBX7-2          2.348       3.997
MPBX7-3          3.336       5.678
MPBX7-4          8.804      14.992
MPBX8-head      -1.279       2.160
MPBX8-1         -1.867       3.199
MPBX8-2         -2.372       4.051
MPBX8-3         -3.381       5.755
MPBX8-4         -8.757      14.832
MPBX9-head      -0.000       2.510
MPBX9-1         -0.000       3.604
MPBX9-2         -0.000       4.604
MPBX9-3         -0.000       6.604
MPBX9-4          0.000      17.354
MPBX10-head      0.000      -2.510
MPBX10-1         0.000      -3.684
MPBX10-2         0.000      -4.674
MPBX10-3         0.000      -6.674
MPBX10-4         0.000     -16.924
Hand Calculation
 Distance to the First Anchor For MPBX-7
 Anchor 1:  sqrt[(1.842-1.271)^2+(3.135-2.164)^2]=1.126446625
 Anchor 2:  sqrt[(2.348-1.271)^2+(3.997-2.164)^2]=2.125986359
 Anchor 3:  sqrt[(3.336-1.271)^2+(5.678-2.164)^2]=4.07583378
 Anchor 4:  sqrt[(8.804-1.271)^2+(14.992-2.164)^2]=14.87627887
 Distance to the First Anchor For MPBX-10
 Anchor 1:  3.684-2.51= 1.174
 Anchor 2:  4.674-2.51= 2.164
 Anchor 3:  6.674-2.51= 4.164
 Anchor 4: 16.924-2.51=14.414
As discussed earlier, FLAC grid coordinates were used to calculate
t1 and t2 for anchors located in zones and t1 for anchors located
on zone edge. These t1 and t2 were then used in the FISH function
to calculate anchor coordinates and distance between anchors. If
anchor coordinates after intallation is used as datum, anchor
displacement relative to the assembly head can be determined.
Also, the anchor coordinates determined using the FISH functions
before heating started should be approximately the same as those
measured immediately after installation of the extensometers. In
other word, the distance of between an anchor in an extensometer
to the assembly head should be the same.
```

```
The distance calculated using FISH function for MPBX7 and MPBX10
are shown as follows:

 >MPBX7_8
 >interPoint7_8
 >coords7_8
 Coordinates of Anchors For MPBX-7
 2.0127100E+02 3.0216400E+02
 2.0184200E+02 3.0313500E+02
 2.0234800E+02 3.0399700E+02
 2.0333600E+02 3.0567800E+02
 2.0880400E+02 3.1499200E+02
 >distance7_8
 Distance to the First Anchor For MPBX-7
 1.1264475E+00
 2.1259874E+00
 4.0758348E+00
 1.4876280E+01
 Coordinates of Anchors For MPBX-10
 2.0000000E+02 2.9749000E+02
 2.0000000E+02 2.9631600E+02
 2.0000000E+02 2.9532600E+02
 2.0000000E+02 2.9332600E+02
 2.0000000E+02 2.8307600E+02
 >distance9_10
 Distance to the First Anchor For MPBX-10
 1.1739999E+00
 2.1640000E+00
 4.1640000E+00
 1.4414000E+01
```

The difference between the hand-calculated results and the results obtained using the FLAC FISH functions are due to precision and rounding errors. The difference is small and acceptable.

The roof-to-floor and wall-to-wall convergences are determined using the following FISH function. This FISH Function ~~computer~~ compute drift height and width at a ~~date~~ predetermined condition, e.g., after excavation and after certain period of heating. The difference between two conditions is the convergence.

```
;
;First of the two FISH functions
;Convergence
;
def convergence
  a111=x(203,301)-x(198,301)
  a112=y(203,301)-y(198,301)
  ra=sqrt(a111*a111+a112*a112)*1000
  oo=out('Drift Wall Width')
  tt1=fstring(ra,9)
  oo=out(tt1)
  a111=x(201,303)-x(201,298)
  a112=y(201,303)-y(201,298)
  ra=sqrt(a111*a111+a112*a112)*1000
  oo=out('Roof Height')
  tt1=fstring(ra,9)
  oo=out(tt1)
end
;
convergence
```


For a 5-m diameter drift, the FLAC result using the FISH Function on p.80 is shown below (verification). The unit ~~the~~ for the FLAC result is in mm.

```
>;
>convergence
Drift Wall Width
5.0000000E+03
Roof Height
5.0000000E+03
```

2.3 Continuum Model for Fracture Deformation

2.3.1 Matrix Permeability of Fractured Rocks

~~As indicated in Section 1,~~ one of the objectives of this study is to predict the thermal-mechanical effects on rock-mass permeability. Permeability is a measure of the ability of a material to transmit fluid under a hydraulic gradient. Permeability is the most important rock parameter pertinent to fluid flow; it is an intrinsic property of rock and relates to the presence of interconnected voids (pores) and fractures. In general, permeability of a rock-mass can be divided into two parts: matrix and fracture permeabilities.

For a porous medium, the matrix permeability, k_m, may be determined using the Carman-Kozeny equation (Panda and Lake, 1994)

$$k_m = \frac{D_p^2 \varphi_m^3}{72\tau_m (1-\varphi_m)^2} \tag{2-7}$$

where D_p is the particle diameter, φ_m is the porosity of the medium, and τ_m is tortuosity of the medium. Assuming that D_p and τ_m are constant when they are subjected to stress changes, the matrix permeability at a state of stresses can be expressed as

$$k_m = k_{mo} \frac{\varphi_m^3}{(1-\varphi_m)^2} \frac{(1-\varphi_{mo})^2}{\varphi_{mo}^3} \tag{2-8}$$

where φ_{mo} is reference matrix porosity and k_{mo} is the intrinsic permeability for matrix porosity φ_{mo}.

2.3.2 Fracture Permeability of Fractured Rocks

Permeability in fractures depends on size, density, connectivity, orientation, and smoothness of fractures. The permeability, k_f, for individual fractures can be represented by

$$k_f = \frac{b^2}{12} \tag{2-9}$$

where b is fracture aperture. This expression may be modified to account for fracture roughness. The fracture-set permeability for a set of fractures characterized by joint density, f_d, on the other hand, can be expressed by (Ofoegbu, 2000; Elsworth and Mase, 1993; Elsworth, 1989)

$$k_f = f_d \frac{b^3}{12} \tag{2-10}$$

The fracture permeability in the fracture-parallel direction for a two-orthogonal-fracture-set system is give by (Ofoegbu, 2000; Elsworth and Mase, 1993; Elsworth, 1989)

$$k_f = f_d \frac{b^3}{6} \tag{2-11}$$

2.3.3 Continuum Representation for Mechanical Effect on Fractured Rock Permeability

In a continuum approach, direct representation of fracture permeability is not possible. A mathematical derivation of fractured rock-mass permeability for the purpose of this study, based on several assumptions, is presented in this section.

When a fractured rock medium is subjected to stress changes resulting from either excavation or temperature variation, deformation of the rock-mass affected will be induced. As a result, the volume of the fractured rock medium changes accordingly. The extent of volume change depends on the magnitude of stress changes.

This volume change may be related to change in matrix and fracture porosities of the fractured rock medium through volumetric strain. For a deformed rock block, its volumetric strain, e_v, can be determined using

$$e_v = \frac{\Delta V}{V} \tag{2-12}$$

where is ΔV is the volume change and V is the rock volume before deformation took place. As discussed earlier, e_v may be directly related to the changes in both matrix and fracture porosities, $\Delta\varphi_m$ and $\Delta\varphi_f$.

$$e_v = \Delta\varphi_m + \Delta\varphi_f \tag{2-13}$$

In deformable fractured rocks, changes in matrix porosity are small. Furthermore, for the fractured rocks of interest in this study, their matrix permeabilities are more than four-order of

magnitude smaller than the fracture permeabilities. Consequently, the influence of volume-change induced matrix-permeability change to the overall permeability of a fractured rock-mass is likely very small. With this understanding in mind, Eq. (2-13) can be approximated by

$$e_v = \Delta\varphi_f \tag{2-14}$$

With the assumptions that a fractured rock-mass consists of three mutually perpendicular fracture sets, each with fracture density f_d and aperture b, Ofoegbu (2000) suggested that the linear fracture density normal to a given fracture set is f_d and the volumetric fracture density is $3f_d$. The fracture aperture of the rock-mass can then be related to fracture porosity by

$$b = \frac{\varphi_f}{3f_d} \tag{2-15}$$

The fractured rock-mass permeability is related to fracture porosity by combining Eqs. (2-11) and (2-15)

$$k_f = \frac{\varphi_f^3}{27 f_d^2} \tag{2-16}$$

162 10/09/2002

Eq. (2-16) can be further expressed using the fracture porosity before change, φ_{fo}, and fracture porosity change $\Delta\varphi_f$

$$k_f = \frac{\left(\varphi_{fo} + \Delta\varphi_f\right)^3}{27 f_d^2} \tag{2-17}$$

162 10/09/2002

The fractured rock-mass permeability after stress change can be directly related to the permeability before stress change, k_{fo}

$$k_f = k_{fo}\left(1 + \frac{\Delta\varphi_f}{\varphi_{fo}}\right)^3 = k_{fo}\left(1 + \frac{e_v}{\varphi_{fo}}\right)^3 \tag{2-18}$$

Note that e_v in Eq. (2-18) includes both elastic e_{ve} and inelastic volumetric strain e_{vi}. The elastic volumetric strain e_{ve} is caused by normal compression or extension of fractures and the shear deformation of fracture before dilation while the inelastic volumetric strain e_{vi} could result from fracture shear dilation or inelastic extension. A plasticity-based yielding or failure criterion can be used to accumulate inelastic volumetric strain. While the inelastic volumetric strain is irreversible, elastic volumetric strain is recoverable. Eq. (2-18) offers a means of analyzing permeability change caused by elastic deformation of rock-mass and an opportunity of assessing the effect of cooling planned for the Drift-Scale Heater Test on rock-mass permeability. This equation suggests that permeability is fully recoverable if the rock-mass is in the elastic range throughout the loading path. Otherwise, after complete unloading, inelastic deformation induced permeability change remains.

Laboratory experiments have suggested that the compressibility of fracture decreases as applied normal stress increases. A fracture can no longer be compressed when a limiting aperture is reached (Witherspoon et al., 1980; Bandis et al., 1983; Barton et al., 1985; Schrauf and Evans, 1986; Hsiung et al., 1994). This limiting aperture is believed to be fracture roughness and fracture wall strength dependent. To account for this behavior, a limiting value for fracture porosity can be established for Eq. (2-18). For example, the limiting value φ_L can be set as

$$\varphi_L = e_{veL} = R_L \varphi_{fo} \tag{2-19}$$

where R_L is a fraction value and e_{veL} is the limiting elastic strain. The resulting fracture porosity because of deformation should not be smaller than φ_L. A complete description of continuum representation of the deformation-permeability model is obtained by combining Eqs. (2-18) and (2-19).

$$k_f = k_{fo}\left(1+\frac{e_{ve}+e_{vi}}{\varphi_{fo}}\right)^3, \qquad e_{ve} \geq -R_L\varphi_{fo}$$
$$k_f = k_{fo}\left(1+\frac{-R_L\varphi_{fo}+e_{vi}}{\varphi_{fo}}\right)^3, \quad e_{ve} < -R_L\varphi_{fo} \tag{2-20}$$

Implementation of Eq(2-20) is provided in the FISH functions listed as follows.

① A wraper function:

```
;FISH to calculate permeability for large fracture porosity
;permeabilityL.fis
;set echo off
set log on
ca FracturePorosityChangeL.fis
ca permeability.fis
ca PermeabilityInHoleL.fis
set log off
```

Sui-Min Hu 8/10/2002

② Calculate changes in fracture porosity for FLAC models

```
;FISH to calculate change in fracture porosity
;FracturePorosityChange.fis
;set echo off
;
def FracPorosityChange
  float $a1 $a2 $a5 fraction residPoro
  array initFP(3)
  array residPoro(3)
  array initPerm(3)
  array Area2(400,400)
;  initFP(1)=0.000329 ; define initial fractrure porosity, Tptpll
;  initFP(2)=0.000263 ; define initial fractrure porosity, Tptpmn
;  initFP(3)=0.000171 ; define initial fractrure porosity, Tptpul
;initFP: initial fracture porosity
;Data from from MultiscaleThermoHydrologicModelAMR, 2001 Rev00 ICN02
  initFP(1)=0.011
  initFP(2)=0.01
  initFP(3)=0.0066
;initPerm: initial permeability
;Data from from MultiscaleThermoHydrologicModelAMR, 2001 Rev00 ICN02
  initPerm(1)=1.29e-12
  initPerm(2)=2.76e-13
  initPerm(3)=5.50e-13
;Data from from Task 2A draft report
;  initFP(1)=0.000329
;  initFP(2)=0.000263
;  initFP(3)=0.000171
  ; fraction: factor used to define residual fracture porosity
  ; residPoro: residual porosity beyond which fracture cannot be
  ;            compressed further
  fraction=0.05
  loop i (1,3)
    residPoro(i)=initFP(i)*fraction
  endLoop
  loop i (1,izones)
    loop j (1,jzones)
;      if shear_mod(i,j)>0 then
;        if bulk_mod(i,j)>0 then
;          ex_3(i,j)=Area1(i,j)
;        end_if
;      end_if
      ;elastic strain
      $a1=Area1(i,j)/Area0(i,j)-abs(e_plastic(i,j))
      if j<=281 then
        $a5=-residPoro(1)
      else
        if j<=318 then
          $a5=-residPoro(2)
        else
          $a5=-residPoro(3)
        end_if
      end_if
      if $a1<$a5 then
        $a1=$a5
      end_if
      ; adjust to effctive volumetric strain
      Area2(i,j)=$a1+abs(e_plastic(i,j))
    endLoop
  endLoop
end
computeareachange
FracPorosityChange
```

③ Calculate permeability for FLAC models

```
;FISH to calculate permeability
;Permeability.fis
;ex_2: array of permeability
;ex_3: ratio of fracture porosity to initial fracture porosity
;ex_4: ratio of current permeability to initial permeability
;set echo off
;
def permability
  loop i (1,izones)
    loop j (1,jzones)
      ; skip drift opening
      if shear_mod(i,j)>0 then
        if bulk_mod(i,j)>0 then
          ;find appropriate initail fracture porosity and
          ;permeability
          if j<=281 then
            $a5=initFP(1)
            $a1=initPerm(1)
          else
            if j<=318 then
              $a5=initFP(2)
              $a1=initPerm(2)
            else
              $a1=initPerm(3)
              $a5=initFP(3)
            end_if
          end_if
          ;compute permeability
          $a2=1+Area2(i,j)/$a5
          ex_3(i,j)=$a2
          ex_4(i,j)=$a2^3
          ex_2(i,j)=$a1*$a2^3
        end_if
      end_if
    endLoop
  endLoop
end
permability
```

④

```
;FISH to output permeability in specific locations in boreholes
;approximated using permeability for zones in which the packers
;are located
;PermeabilityInHole.fis
;Locations are relative to the center of the drift
;Source of Data: As Built report
;Borehole 57; (-24,329,6.603)
;Borehole 57; (-15.589,10.197)
;Borehole 57; (-9.386,12.748)
;Borehole 57; (-1.878,15.836)
;Borehole 59; (-24.595,3.883)
;Borehole 59; (-13.973,4.902)
;Borehole 59; (-5.779,5.689)
;Borehole 59; (-3.212,6.552)
;Borehole 74; (-26.59,7.836)
;Borehole 74; (-16.287,11.62)
;Borehole 74; (-9.418,14.141)
;Borehole 74; (-4.947,15.784)
;Borehole 76; (-25.866,6.065)
;Borehole 76; (-17.331,6.222)
;Borehole 76; (-8.186,6.389)
;Borehole 76; ( 0.844,6.555)
```

```
;Relative positions with respect to FLAC model
;cneter of the drift in FLAC model (200,300), (0,0) is at the
;lower left corner of the model
;Borehole 57; (175.671,306.603)
;Borehole 57; (184.411,310.197)
;Borehole 57; (190.614,312.748)
;Borehole 57; (198.122,315.836)
;Borehole 59; (175.405,303.883)
;Borehole 59; (186.027,304.902)
;Borehole 59; (194.221,305.689)
;Borehole 59; (196.788,306.552)
;Borehole 74; (173.410,307.836)
;Borehole 74; (183.713,311.620)
;Borehole 74; (190.582,314.141)
;Borehole 74; (195.053,315.784)
;Borehole 76; (174.134,306.065)
;Borehole 76; (182.669,306.222)
;Borehole 76; (191.814,306.389)
;Borehole 76; (200.844,306.555)
;set echo off
;
def permeability_hole
;  initFP(1)=0.011
;  initFP(2)=0.01
;  initFP(3)=0.0066
  ttt = fstring(ex_2(175,306),7)
  oo  = out('57-1 Permeability '+ttt)
  ttt = fstring(ex_2(184,310),7)
  oo  = out('57-2 Permeability '+ttt)
  ttt = fstring(ex_2(190,312),7)
  oo  = out('57-3 Permeability '+ttt)
  ttt = fstring(ex_2(198,315),7)
  oo  = out('57-4 Permeability '+ttt)
  ttt = fstring(ex_2(175,303),7)
  oo  = out('59-1 Permeability '+ttt)
  ttt = fstring(ex_2(186,304),7)
  oo  = out('59-2 Permeability '+ttt)
  ttt = fstring(ex_2(194,305),7)
  oo  = out('59-3 Permeability '+ttt)
  ttt = fstring(ex_2(196,306),7)
  oo  = out('59-4 Permeability '+ttt)
  ttt = fstring(ex_2(173,307),7)
  oo  = out('74-1 Permeability '+ttt)
  ttt = fstring(ex_2(183,311),7)
  oo  = out('74-2 Permeability '+ttt)
  ttt = fstring(ex_2(190,314),7)
  oo  = out('74-3 Permeability '+ttt)
  ttt = fstring(ex_2(195,315),7)
  oo  = out('74-4 Permeability '+ttt)
  ttt = fstring(ex_2(174,306),7)
  oo  = out('76-1 Permeability '+ttt)
  ttt = fstring(ex_2(182,306),7)
  oo  = out('76-2 Permeability '+ttt)
  ttt = fstring(ex_2(191,306),7)
  oo  = out('76-3 Permeability '+ttt)
  ttt = fstring(ex_2(200,306),7)
  oo  = out('76-4 Permeability '+ttt)
  ttt = fstring(ex_4(175,306),7)
  oo  = out('57-1 Permeability Ratio '+ttt)
  ttt = fstring(ex_4(184,310),7)
  oo  = out('57-2 Permeability Ratio '+ttt)
  ttt = fstring(ex_4(190,312),7)
  oo  = out('57-3 Permeability Ratio '+ttt)
  ttt = fstring(ex_4(198,315),7)
  oo  = out('57-4 Permeability Ratio '+ttt)
```

```
ttt = fstring(ex_4(175,303),7)
oo  = out('59-1 Permeability Ratio '+ttt)
ttt = fstring(ex_4(186,304),7)
oo  = out('59-2 Permeability Ratio '+ttt)
ttt = fstring(ex_4(194,305),7)
oo  = out('59-3 Permeability Ratio '+ttt)
ttt = fstring(ex_4(196,306),7)
oo  = out('59-4 Permeability Ratio '+ttt)
ttt = fstring(ex_4(173,307),7)
oo  = out('74-1 Permeability Ratio '+ttt)
ttt = fstring(ex_4(183,311),7)
oo  = out('74-2 Permeability Ratio '+ttt)
ttt = fstring(ex_4(190,314),7)
oo  = out('74-3 Permeability Ratio '+ttt)
ttt = fstring(ex_4(195,315),7)
oo  = out('74-4 Permeability Ratio '+ttt)
ttt = fstring(ex_4(174,306),7)
oo  = out('76-1 Permeability Ratio '+ttt)
ttt = fstring(ex_4(182,306),7)
oo  = out('76-2 Permeability Ratio '+ttt)
ttt = fstring(ex_4(191,306),7)
oo  = out('76-3 Permeability Ratio '+ttt)
ttt = fstring(ex_4(200,306),7)
oo  = out('76-4 Permeability Ratio '+ttt)

ttt = fstring(ex_3(175,306),7)
oo  = out('57-1 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(184,310),7)
oo  = out('57-2 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(190,312),7)
oo  = out('57-3 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(198,315),7)
oo  = out('57-4 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(175,303),7)
oo  = out('59-1 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(186,304),7)
oo  = out('59-2 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(194,305),7)
oo  = out('59-3 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(196,306),7)
oo  = out('59-4 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(173,307),7)
oo  = out('74-1 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(183,311),7)
oo  = out('74-2 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(190,314),7)
oo  = out('74-3 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(195,315),7)
oo  = out('74-4 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(174,306),7)
oo  = out('76-1 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(182,306),7)
oo  = out('76-2 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(191,306),7)
oo  = out('76-3 Fracture Porosity Ratio '+ttt)
ttt = fstring(ex_3(200,306),7)
oo  = out('76-4 Fracture Porosity Ratio '+ttt)
```



```
oo  = out('Permeability Presented in row')
tt1 = fstring(ex_2(175,306),7)
tt2 = fstring(ex_2(184,310),7)
tt3 = fstring(ex_2(190,312),7)
tt4 = fstring(ex_2(198,315),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_2(175,303),7)
tt2 = fstring(ex_2(186,304),7)
tt3 = fstring(ex_2(194,305),7)
tt4 = fstring(ex_2(196,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_2(173,307),7)
tt2 = fstring(ex_2(183,311),7)
tt3 = fstring(ex_2(190,314),7)
tt4 = fstring(ex_2(195,315),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_2(174,306),7)
tt2 = fstring(ex_2(182,306),7)
tt3 = fstring(ex_2(191,306),7)
tt4 = fstring(ex_2(200,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)

oo  = out('Permeability Presented in column')
tt1 = fstring(ex_2(175,306),7)
tt2 = fstring(ex_2(175,303),7)
tt3 = fstring(ex_2(173,307),7)
tt4 = fstring(ex_2(174,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_2(184,310),7)
tt2 = fstring(ex_2(186,304),7)
tt3 = fstring(ex_2(183,311),7)
tt4 = fstring(ex_2(182,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_2(190,312),7)
tt2 = fstring(ex_2(194,305),7)
tt3 = fstring(ex_2(190,314),7)
tt4 = fstring(ex_2(191,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_2(198,315),7)
tt2 = fstring(ex_2(196,306),7)
tt3 = fstring(ex_2(195,315),7)
tt4 = fstring(ex_2(200,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)

oo  = out('Permeability Ratio Presented in row')
tt1 = fstring(ex_4(175,306),7)
tt2 = fstring(ex_4(184,310),7)
tt3 = fstring(ex_4(190,312),7)
tt4 = fstring(ex_4(198,315),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_4(175,303),7)
tt2 = fstring(ex_4(186,304),7)
tt3 = fstring(ex_4(194,305),7)
tt4 = fstring(ex_4(196,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_4(173,307),7)
tt2 = fstring(ex_4(183,311),7)
tt3 = fstring(ex_4(190,314),7)
tt4 = fstring(ex_4(195,315),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
tt1 = fstring(ex_4(174,306),7)
tt2 = fstring(ex_4(182,306),7)
tt3 = fstring(ex_4(191,306),7)
tt4 = fstring(ex_4(200,306),7)
oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
```

```
    oo  = out('Permeability Ratio Presented in column')
    tt1 = fstring(ex_4(175,306),7)
    tt2 = fstring(ex_4(175,303),7)
    tt3 = fstring(ex_4(173,307),7)
    tt4 = fstring(ex_4(174,306),7)
    oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
    tt1 = fstring(ex_4(184,310),7)
    tt2 = fstring(ex_4(186,304),7)
    tt3 = fstring(ex_4(183,311),7)
    tt4 = fstring(ex_4(182,306),7)
    oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
    tt1 = fstring(ex_4(190,312),7)
    tt2 = fstring(ex_4(194,305),7)
    tt3 = fstring(ex_4(190,314),7)
    tt4 = fstring(ex_4(191,306),7)
    oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
    tt1 = fstring(ex_4(198,315),7)
    tt2 = fstring(ex_4(196,306),7)
    tt3 = fstring(ex_4(195,315),7)
    tt4 = fstring(ex_4(200,306),7)
    oo  = out(tt1+' '+tt2+' '+tt3+' '+tt4)
end
permeability_hole
```


Bandis, S.C., A.C. Lumsden, and N.R. Barton. "Fundamentals of rock joint deformation." *International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts.* 20(6). 249–368. 1983.

Barton, N.R., S.C. Bandis, and K. Bakhtar. "Strength, deformation, and conductivity coupling of rock joints." *International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts.* 22(3). 121–140. 1985.

Elsworth, D. "Thermal Permeability Enhancement of Blocky Rocks: One-Dimensional Flow." *International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts.* 26(3/4). 329–339. 1989.

Elsworth, D., and C.R. Mase. "Chapter 8: Groundwater in rock Engineering." *Comprehensive Rock Engineering* 1. J.A. Hudson, ed. New York, Pergamon Press. 201–226. 1993.

Hsiung, S.M., D.D. Kana, M.P. Ahola, A.H. Chowdhury, and A. Ghosh. "Laboratory Characterization of Rock Joints." NUREG/CR–6178. Washington, DC: Nuclear Regulatory Commission. 1994.

Ofoegbu, G.I. "Variations of drift at the proposed Yucca Mountain repository." Rock Mechanics for Industry. Proceedings of the 37th U.S. Rock Mechanics Symposium. B. Amadei, R.L. Kranz, G.A. Scott, and P. Smeallie, eds. Rotteram, The Netherlands: A.A. Balkema. 767–773. 1999.

Ofoegbu, G.I. "Thermal-Mechanical Effects on Long-Term Hydrological Properties at the Proposed Yucca Mountain Nuclear Waste Repository." CNWRA 2000–03. San Antonio, Texas: CNWRA. 2000.

Panda, M.N., and L.W. Lake. "Estimation of single-phase permeability from the parameters of a particle-size distribution. AAPG Bulletin. 78(7). 1028–1039. 1994.

Schrauf, T.W. and D.D. Evans. "Laboratory Studies of Gas Flow Through a Single Natural Fracture." *Water Resources Research.* 22(7). 1038–1050. 1986.

Witherspoon, P.A., J.S.Y. Wang, K. Iwai, and J.E. Gale. "Validity of Cubic Law for Fluid Flow in a Deformation Rock Fracture." *Water Resources Research.* 16(6). 1016–1024. 1980.

Sun-Mi Hi 8/10/2002

A revised temperature distribution time history was provided by the Task 2 Technical monitoring research team in 01/03. The Task 2C research teams were requested to redo the Task 2C activities.

The new objectives are:

to conduct thermal-mechanical analyses of the drift-scale heater test using the revised temperature distribution data as provided by the Task 2 technical monitoring research team. Compared to the earlier thermal-mechanical analyses (Hsiung and Chowdhury, 2002), these analyses focuses on investigating effects of rock-mass quality, different rock-mass failure criteria, and treatment of thermal expansion coefficient on thermally induced deformation and thermal-mechanical effects on rock-mass permeability.

The analyses used temperature distributions at 7 thermal time steps: 3 months, 6 months, 9 months, 1 year, 2 years, 3 years, and 4 years. The temperature contours for these 7 thermal time steps were taken on a vertical cross section at 23m from the thermal bulkhead of the heated drift.

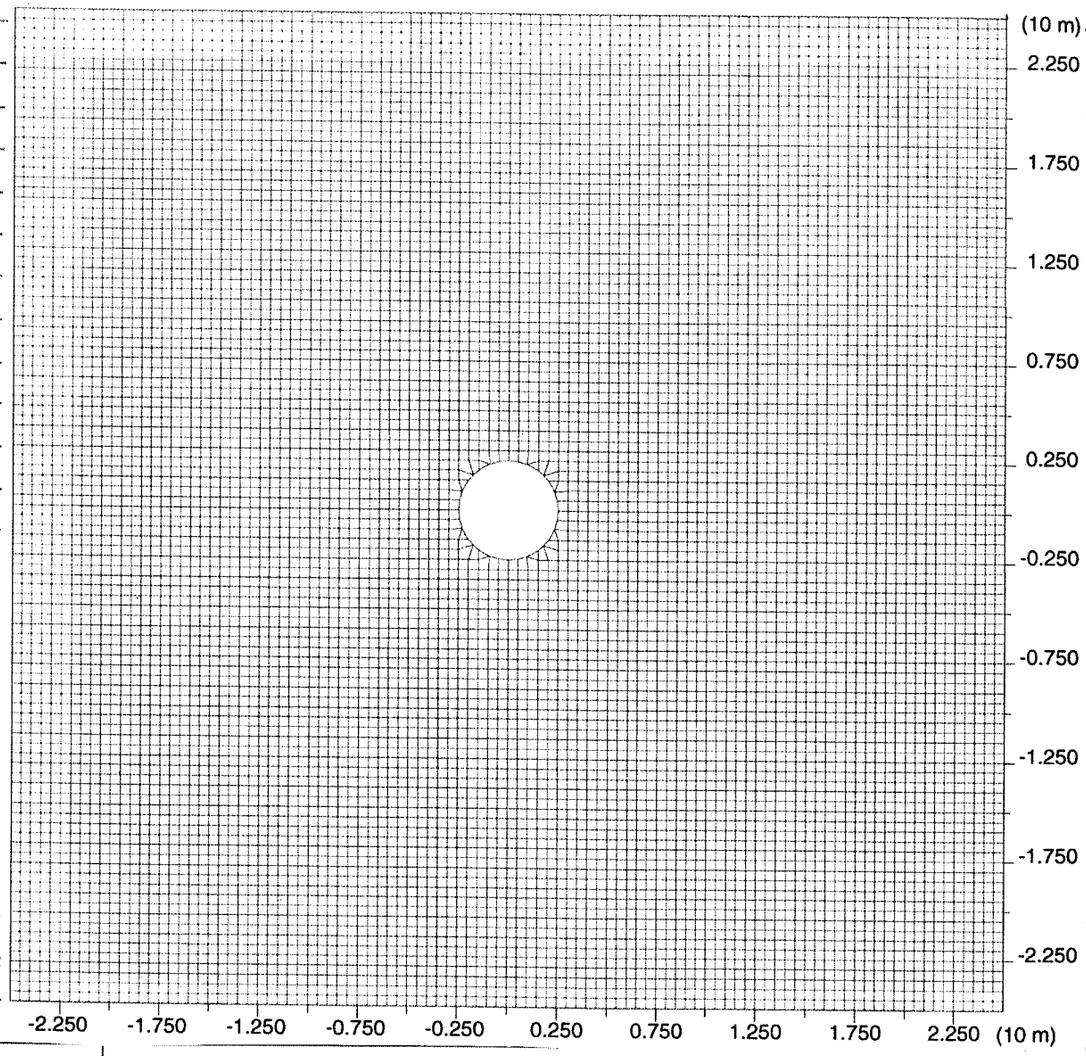
Drift-Scale Heater Test Model Domain

A vertically oriented two-dimensional cross section was configured for numerical analysis using FLAC computer code. This cross section intersected the axis of the heated drift at middistance between the thermal bulkhead and the terminus of the heated drift. The two-dimensional numerical models developed, based on this cross section, simulated a plane strain condition. The FLAC models developed had a dimension of 1,000 m [3,281 ft] in width and 740 m [2,428 ft] in height with the origin of the coordinates located at the center of the heated drift. A 5-m [16.4-ft] diameter circular drift was constructed with its center located 500 m [1,640 ft] from the left boundary and 500 m [1,640 ft] from the bottom of the model domain.

Sun-Mi Hi 01/06/2003

The FLAC models developed consisted of 400 × 400 grids. The finite difference zone size in a rectangular region 35 m [115 ft] from the center of the heated drift was 0.5 m × 0.5 m [1.64 ft × 1.64 ft] except for grids near the heated drift. The grid distance around the heated drift was adjusted to better describe the shape of the drift. The finite difference zone size was increased gradually from the boundaries of the rectangular region to the model boundaries.

A closeup of the FLAC model with grids in the vicinity of the heated drift is shown below



Base and Sensitivity Analysis Cases

Several case studies were performed for Task 2C modeling activities. The objectives of the study were to investigate the effects of rock-mass mechanical and strength properties, thermal expansion coefficients, and failure criteria used on displacements and permeability changes at predetermined locations. Among all the cases studied, the one using the material and strength properties for Rock-Mass Quality Category 2 rock listed in Tables 3-2 and 3-4 was assigned as the basecase. Rock-Mass Quality Category 2 was selected as a basecase because CRWMS M&O (1999) reported that limited rock-mass modulus obtained from field tests for the Topopah Spring Welded Tuff Thermal-Mechanical Unit 2 was smaller than the rock-mass Young's modulus designated for the Topopah Spring Welded Tuff Thermal-Mechanical Unit 2 (CRWMS M&O, 1999). Toward this end, Rock-Mass Quality Category 2 might be reasonable to represent rock-mass quality for at least the Middle Nonlithophysal litho-stratigraphic unit.

The basecase used the ubiquitous fracture failure criterion to assess failure condition of the rock mass. The fracture properties needed for the ubiquitous fracture failure criterion are listed in Table 3-6. The fracture tensile strength for all litho-stratigraphic units was assumed to be 0. Sensitivity of failure criteria to the displacement and permeability under heated conditions were evaluated by replacing the ubiquitous fracture failure criterion with the commonly used Mohr-Coulomb failure criterion.

Table 3-6. Fracture Properties Used For Ubiquitous Fracture Failure Criterion (CRWMS M&O, 2000)

Litho-Stratigraphic Unit	Fracture Orientation (counterclockwise from x axis), degree	Cohesion, MPa [psi]	Friction Angle, degree	Dilation Angle, degree
Upper Lithophysal	82.0	0.1 [14.5]	41	20.5
Middle Nonlithophysal	83.5	0.1 [14.5]	41	20.5
Lower Lithophysal	80.5	0.1 [14.5]	41	20.5

The thermal expansion coefficients used for the basecase are those listed in Table 3-5 on p.59. At each thermal time step, the thermal expansion coefficient was determined using the temperature at that thermal time.

01/07/2003 Jin-Ming Hong

Sensitivity of the thermal expansion coefficient was assessed considering a constant thermal expansion coefficient for each litho-stratigraphic units. In this case, the thermal expansion coefficient for 125–150 °C [257–302 °F] were chosen for study. For the convenience of discussion, the first thermal expansion coefficient used was designated as medium and the second one was designed as high thermal expansion coefficient.

Sensitivity of rock-mass quality to the displacement and permeability under heated conditions were analyzed by considering Rock-Mass Quality Categories 1 and 5. The material and strength properties for these two rock-mass quality categories were listed in Tables 3-2 and 3-4. Mohr-Coulomb failure criterion was used for both cases.

Tables 3-2 and 3-4 can be found on p. 55 and p. 56.

Table 3-7. Study Cases and Parameters Used

Case No.	Rock-Mass Quality Category	Failure Criterion Used	Thermal Expansion Coefficient
1	2	Mohr-Coulomb	Temperature-Dependent
2	1	Mohr-Coulomb	Temperature-Dependent
3	85 01/12/2003	Mohr-Coulomb	Temperature-Dependent
4 (basecase)	2	Ubiquitous	Temperature-Dependent
5	2	Mohr-Coulomb	Constant, Medium
6	1	Ubiquitous	Temperature-Dependent
7	2	Ubiquitous	Constant, Medium
8	3	Ubiquitous	Temperature-Dependent

01/08/2003 Sin-Min Hsiung

A new deformation-permability relationship was introduced for this analysis. 02/21/2003

Continuum Model of Deformation – ~~Perma~~ Permeability Model

Permeability is a measure of the ability of a material to transmit fluid under a hydraulic gradient. Permeability is the most important rock parameter pertinent to fluid flow; it is an intrinsic property of rock and relates to the presence of interconnected voids (pores) and fractures. In general, permeability of a rock mass can be divided into two parts, matrix and fracture permeabilities.

In deformable fractured rocks, changes in matrix permeabilities are likely to be small. Furthermore, for the fractured rocks of interest in this study, their matrix permeabilities are more than four orders of magnitude smaller than the fracture permeabilities (CRWMS M&O, 2001). Consequently, the contribution of the rock-matrix-deformation-induced matrix-permeability change to the overall permeability of a fractured-rock mass is likely very small. With this understanding, only the effects of mechanically induced fracture deformation on the rock-mass permeability are considered in the deformation-permeability model developed in the following sections.

The ability of a fractured rock mass to transport fluid is controlled by, among other things, the geometry of the fracture system and the magnitude and orientation of the *in situ* stress field (Raven and Gale, 1985). The existing stress field may be modified by, for example, underground excavations, geological processes, and, in the cases of geological disposal of high-level nuclear wastes, stress changes due to thermal-decay of the disposed high-level wastes. Such perturbations on the existing stress field may change the normal and shear stresses acting across fracture planes of the fracture sets in rock-mass media, which, in turn, affecting the permeabilities related to these fracture sets. In general, mechanically induced change in fracture permeability comes from two sources: (i) change in fracture normal stress before and (ii) fracture dilation due to fracture shear or tensile failure. The first case causes a change in fracture aperture that is reversible while the second case is assumed to change fracture aperture permanently.

In a continuum approach, direct representation of fracture permeability is not possible. A mathematical derivation of fractured-rock-mass permeability for this study, based on several assumptions, is presented in the following sections. This representation takes advantage of knowledge on relationship between fracture normal stress and deformation and potential rock-mass plastic deformation.

Effect of Fracture Normal Stress

The change in fracture aperture because of the change in fracture normal stress before tensile failure may be estimated by a fracture normal closure model proposed by Barton and Bandis

(1982). This model describes the relationship of fracture normal stress, σ_n, and the associated closure, u_n, using the following equation.

$$\sigma_n = \frac{u_n}{a - cu_n} \tag{2-7}$$

where a and c are constants and σ_n should be in compression and assumed to be positive in value. An example of this relationship is shown as the curve in red if Figure 2-1. Essentially, this curve indicates that a fracture can be compressed more at a relatively smaller fracture normal stress level. The fracture becomes more difficult to compress as the normal stress goes higher and there appears to exist a maximum fracture deformation level beyond which the fracture can no longer be compressed. This maximum deformation level is called maximum possible closure.

Assuming that fracture tensile strength exists, then some amount of fracture tensile normal stress (negative in value) is possible. Equation (2-7) is not adequate to account for a possible tensile normal stress condition. To address this problem, Eq. (2-7) needs to be extended by including appropriate fracture tensile strength. In this study, σ_n is replaced by the term $\sigma_n + \sigma_t$. σ_t is the fracture tensile strength or can be assumed to be the tensile strength of the rock mass. Fracture dilation due to tension occurs when the tensile σ_n is greater than the fracture tensile strength.

Equation (2-7) can be expressed in terms of fracture closure, u_n, as follows

$$u_n = \frac{a\sigma_n}{c\sigma_n + 1} \tag{2-8}$$

The constant a and c can be estimated based on laboratory tests or field measurements. The latter is more preferable. In this study, neither laboratory tests nor field measurements are not available to empirically determine constants a and c. Consequently, the numerical values of these two constants will have to be estimated.

Bandis et al. (1983) suggest that the constant a in Eqs. (2-7) and (2-8) can be approximated with the initial stiffness, K_{ni}, of the fracture

$$a = \frac{1}{K_{ni}} \tag{2-9}$$

and the maximum possible closure of the fracture, $u_{n\,max}$, may be defined as

$$u_{n\,max} = \frac{a}{c} \tag{2-10}$$

Laboratory experiments have suggested that the compressibility of fracture decreases as applied normal stress increases. A fracture can no longer be compressed when a limiting aperture or after the maximum possible closure is reached (Witherspoon et al., 1980; Bandis et al., 1983; Barton et al., 1985; Schrauf and Evans, 1986; Hsiung et al., 1994). This limiting aperture is believed to be fracture roughness and fracture wall-strength dependent. This limiting aperture is called residual fracture aperture, b_{res} (Figure 2-1). In the convenience of this study, this residual fracture aperture is assumed to be small and can be neglected. With this assumption, $u_{n\,max}$ can be related to the fracture aperture b_0 at which time the fracture normal stress, σ_n, is 0.

$$b_0 = u_{n\,max} = \frac{a}{c} \tag{2-11}$$

With the relationship in Eq. (2-11) established, the reference fracture aperture, b_r, for the reference fracture permeability and porosity at the reference fracture normal stress, σ_{nr}, can be determined using the following equation

$$b_r = b_0 - u_{nr} \tag{2-12}$$

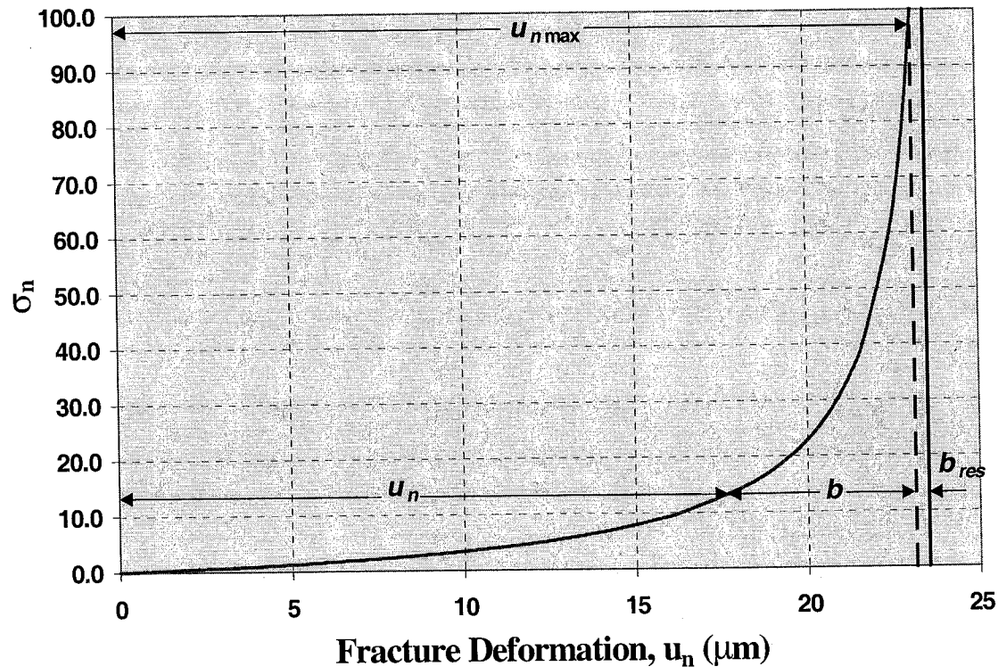


Figure 2-1. Fracture normal stress and displacement curve

where u_{nr} is the fracture closure at σ_{nr}. By incorporating Eqs. (2-8) and (2-11), Eq. (2-12) can be rewritten as

$$b_r = \frac{a}{c} - \frac{a\sigma_{nr}}{c\sigma_{nr}+1} = \frac{a}{c(c\sigma_{nr}+1)} \tag{2-13}$$

Eq. (2-13) may be manipulated to become a quadratic equation

$$\sigma_{nr}c^2 + c - \frac{a}{b_r} = 0 \tag{2-14}$$

This quadratic equation can then be used to determine constant c.

$$c = \frac{-1 \pm \sqrt{1 + \frac{4\sigma_{nr}a}{b_r}}}{2\sigma_{nr}} \tag{2-15}$$

Although constant c can be determined using Eq. (2-15), the reference fracture aperture, b_r, is still a difficult quantity to obtain. It is, therefore, necessary to replace the reference fracture aperture, b_r, with some known measurements. It should be noted that the fracture aperture, b, of a fracture set may be related to its corresponding fracture porosity, ϕ_f, and the fracture density, f_d, for any fracture normal stress, σ_n, using the following equation

$$b = \frac{\phi_f}{f_d} \tag{2-16}$$

and the fracture-set permeability, k_f, for a set of fractures can be characterized by b and f_d using the following equation (Ofoegbu, 2000; Elworth and Mase, 1993; Elsworth, 1989)

$$k_f = f_d \frac{b^3}{12} \tag{2-17}$$

Replacing f_d in Eq. (2-16) with that in Eq. (2-17), b can be expressed in terms of k_f and ϕ_f

$$b = \sqrt{\frac{12k_f}{\phi_f}} \tag{2-18}$$

For a predetermined reference fracture normal stress level, the corresponding reference fracture aperture, b_r, can be shown as follows

$$b_r = \sqrt{\frac{12k_{fr}}{\phi_{fr}}} \tag{2-19}$$

where k_{fr} and ϕ_{fr} are the reference fracture permeability and porosity, respectively of a fracture set. Replacing b_r in Eq. (2-15) with that in Eq. (2-19), c can now be estimated with known variables

$$c = \frac{-1 \pm \sqrt{1 + 4\sigma_{nr}a\sqrt{\frac{\phi_{fr}}{12k_{fr}}}}}{2\sigma_{nr}} \tag{2-20}$$

With a and c known, the fracture aperture at any given fracture normal stress level can be determined.

Fracture Dilation of Fractured Rocks

The change in fracture permeability due to plastic deformation of a fracture set may be estimated by assuming that

$$\Delta\phi_{fp} = e_{ftp} + e_{fsp}\tan\psi_f \tag{2-21}$$

where $\Delta\phi_{fp}$ is the change in fracture porosity, e_{ftp} is the plastic strain due to tensile failure of the fracture set, e_{fsp} is the plastic shear strain of the fracture set, and ψ_f is the dilation angle of the fracture set.

In this study, both Mohr-Coulomb failure criterion and ubiquitous fracture failure criterion were used to evaluate the mechanical effect on rock-mass permeability. For the cases using Mohr-Coulomb failure criterion, e_{ftp}, e_{fsp}, and ψ_f were replaced by those of the rock mass while, for the cases using a ubiquitous fracture failure criterion, e_{ftp}, e_{fsp}, and ψ_f were those of the fracture set.

Continuum Representation for Mechanical Effect on Fractured Rock Permeability

Knowing the fracture normal stress, σ_n, and fracture plastic strains, e_{ftp} and e_{fsp}, the effective fracture aperture, at that fracture normal stress level, is the sum of reversible aperture as a function of fracture normal stress and the fracture plastic deformation and can be determined by

$$b = b_0 - u_n + \Delta b_p \tag{2-22}$$

where Δb_p is the aperture change because of the fracture plastic deformation and can be shown as

$$\Delta b_p = \frac{\Delta \phi_{fp}}{f_d} \tag{2-23}$$

Variables in right hand side of Eq. (2-22) can be replaced by those shown in Eqs. (2-8), (2-11), and (2-23) to obtain the following equation

$$b = \frac{a}{c} - \frac{a\sigma_n}{c\sigma_n + 1} + \frac{\Delta \phi_{fp}}{f_d} = \frac{a}{c(c\sigma_n + 1)} + \frac{\Delta \phi_{fp}}{f_d} \tag{2-24}$$

Combining Eq. (2-20) with Eq. (2-24), Eq. (2-24) can be rewritten as

$$b = \frac{a}{c(c\sigma_n + 1)} + \frac{e_{ftp} + e_{fsp} \tan \psi_f}{f_d} \tag{2-25}$$

By normalizing Eq. (2-17) with the reference fracture permeability, k_{fr}, and applying Eq. (2-25), a general mathematical form representing the mechanical effect on fracture permeability can be obtained

3/17/03

$$\frac{k_f}{k_{fr}} = \frac{f_d \frac{b^3}{12}}{f_d \frac{b_r^3}{12}} = \left[\frac{a}{c(c\sigma_n + 1)} \frac{1}{b_r} + \frac{e_{ftp} + e_{fsp} \tan \psi_f}{f_d b_r} \right]^3 \tag{2-26}$$

Notice that $f_d b_r$ is equal to ϕ_{fr} [Eq. (2-16)] and b_r can be expressed as shown in Eq. (2-19); consequently, Eq. (2-26) can be rewritten as

$$k_f = k_{fr} \left[\frac{a}{c(c\sigma_n + 1)} \sqrt{\frac{\phi_{fr}}{12 k_{fr}}} + \frac{e_{ftp} + e_{fsp} \tan \psi_f}{\phi_{fr}} \right]^3 \tag{2-27}$$

02/23/2003 Sai-Ming Hsiung

4.2.2 Parameters Needed for Permeability Calculation

4.2.2.1 Initial Fracture Stiffness

The initial stiffness for rock fractures is difficult to determine. Hsiung et al. (1994) conducted a series of laboratory experiments on natural Apache Leap welded tuff fractures with an attempt to establish the normal stress versus fracture closure relationship. These experiments were conducted with five repeated normal load cycles with the maximum normal stress of 8.0 MPa [1.16×10^3 psi] applied to the samples tested. The normal stress versus fracture closure curve was used to calculate the initial stiffness. The test results (Hsiung et al., 1994) indicated that the initial fracture stiffness might be ranging from 7 GPa/m to 51 GPa/m [2.58×10^4 psi/in to 1.88×10^5 psi/in]. This range appeared to be at the lower bound because the maximum normal stress used for the cyclic tests might not be sufficient high to reach the level of compression necessary to establish a reasonable condition for determining fracture initial stiffness.

For the three Topopah Spring Lith-Stratigraphic Units, initial fracture stiffness was assumed according to rock-mass quality (CRWMS M&O, 1999). The assumed initial stiffness ranges from 5.10×10^4 MPa/m [1.88×10^5 psi/in] for Roc-Mass Quality Category 1 to 9.00×10^5 MPa/m [3.31×10^6 psi/in] for Rock-Mass Quality Category 5. It appeared that higher bound value determined from laboratory by Hsiung et al. (1994) matched the lower bound value assumed for the Topopah Spring Lith-Stratigraphic Units.

In this study, initial fracture stiffness of 2.01×10^5 MPa/m [7.40×10^5 psi/in] was used to evaluate rock-mass permeability variations resulting from heating. This value represented assumed initial stiffness for fractures in rock mass for Rock-Mass Quality Category 2. Further study may be necessary to assess this selection.

4.2.2.2 Sensitivity of Constant *a* on Permeability Change

02/23/2003 2-27

The continuum representation of the deformation-permeability relationship of Eq. ~~(4-1)~~ is sensitive to constant *a* which, in turn, is sensitive to initial stiffness of the fracture set. Figure 4-20 illustrates the effects of constant *a* on rock-mass permeability. The vertical axis represents the ratio of the rock-mass permeability at the time of interest to that at a reference point. The numbers listed in the captures are values used for constant *a*. Note that the inverse of constant *a* is the initial fracture stiffness. This figure suggested that the larger the value for constant *a* is the less variability on rock-mass permeability due to stress variation will be. In other words, other things being equal, fractures with high stiffness is less sensitive to fracture normal stress change.

4.2.2.3 Fracture Properties

To account for the possible effects of the presence of fractures, permeability of rock-mass along two pseudo-fracture sets for each litho-stratigraphic units at various thermal times was calculated. One fracture set was oriented subvertically and the other subhorizontally. The dip and dilation angles of the fracture sets in three litho-stratigraphic units are listed in Table 4-1.

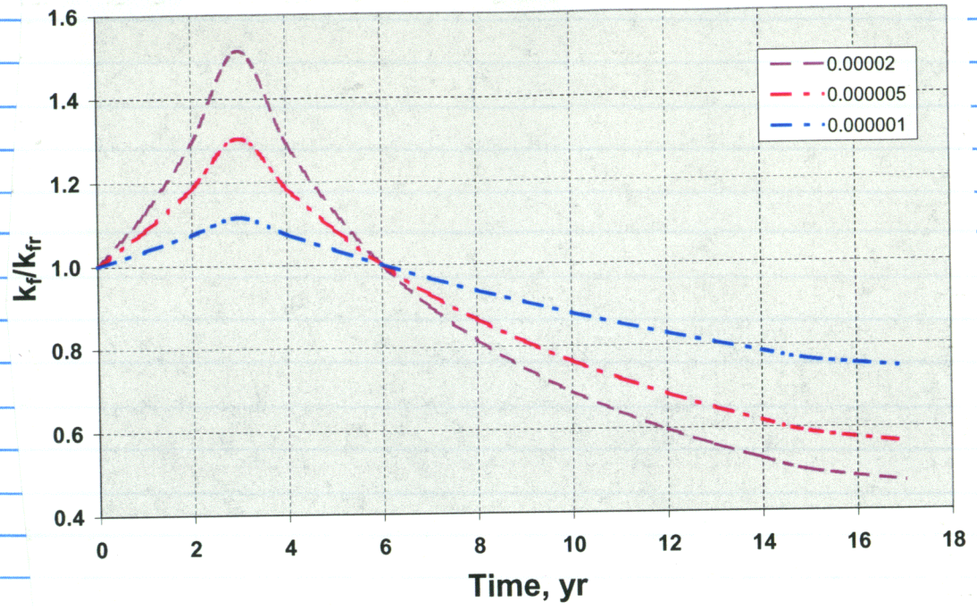


Figure 4-20. Effect of constant *a* on rock-mass permeability

02/23/2003
Sui-Min Hsiung

Table 4-1. Fracture Dip and Dilation Angles (CRWMS M&O, 2000)

Litho-Stratigraphic Unit	Fracture Dip Angle, degree (Counterclockwise from x axis)		Fracture Dilation angle, degree		Rock-Mass Dilation angle, degree
	Sub-vertical	Sub-horizontal	Sub-vertical	Sub-horizontal	
Upper Lithophysal	82.0	14	20.5	20.5	23.00
Middle Nonlithophysal	83.5	9	20.5	20.5	24.08
Lower Lithophysal	80.5	5	20.5	20.5	23.00

Notice that the dip and dilation angles for the subvertical fracture sets used in permeability calculation were the same as those for cases 4 and 6 where a ubiquitous fracture failure criterion was used for analysis to be consistent. For the cases where a Mohr-Coulomb failure criterion was used to assess rock-mass failure, dilation angles of rock-mass were used for permeability calculation.

4.2.2.4 Reference Fracture Porosity and Permeability

For this study, k_{fr} and φ_{fr} were assumed to be the initial values before initiation of heating. The reference fracture porosities and permeabilities for the three litho-stratigraphic units used in this study are listed in Table 4-2. The fracture porosity and permeability for the subvertical and subhorizontal fracture sets were assumed to be the same. This assumption might not be a valid one. However, because of a lack of information, this might the only alternative.

Table 4-2. Reference Fracture Porosities and Fracture Permeabilities (CRWMS M&O, 2001)

Litho-Stratigraphic Unit	Porosity	Permeability, m^2 [ft^2]
Upper Lithophysal	0.0066	5.50×10^{-13} [5.92×10^{-12}]
Middle Nonlithophysal	0.01	2.76×10^{-13} [2.97×10^{-12}]
Lower Lithophysal	0.011	1.29×10^{-12} [1.39×10^{-11}]

02/23/2003
Sui-Min Hsiung

Calculate constant c: FISH function

```
;
; Compute constant c in un=a*sig_n/(c*sig_n+1)
; un    : fracture closure
; sig_n: fracture normal stress (should be compression)
; for UBIQUITOUS MODEL
;
def compute_c
  array frFP(3)
  array frPerm(3)
  array b_r(3)
  array vsig_n(400,400) ; fracture normal stress prior excavation, set 1
  array hsig_n(400,400) ; fracture normal stress prior excavation, set 2
  array const_cv(400,400)
  array const_ch(400,400)
  array m_janglev(3)
  array m_jangleh(3)

  float $s22p $theta $cs $sn $cla $clb $csn $clc $clab
  float $theta1 $theta2

;
; --- Define angles of fracture sets
;
  m_janglev(1)=80.5 * degrad
  m_janglev(2)=83.5 * degrad
  m_janglev(3)=82.0 * degrad
  m_jangleh(1)= 5.0 * degrad
  m_jangleh(2)= 9.0 * degrad
  m_jangleh(3)=14.0 * degrad

; --- frFP(1); define reference fractrure porosity, Tptpll
; --- frFP(2); define reference fractrure porosity, Tptpmn
; --- frFP(3); define reference fractrure porosity, Tptpul
; --- Data from from MultiscaleThermoHydrologicModelAMR, 2001 Rev00 ICN02
;
  frFP(1)=0.011
  frFP(2)=0.01
  frFP(3)=0.0066
; --- frPerm: reference permeability
; --- Data from from MultiscaleThermoHydrologicModelAMR, 2001 Rev00 ICN02
;
  frPerm(1)=1.29e-12
  frPerm(2)=2.76e-13
  frPerm(3)=5.50e-13

;  const_a=0.00002 ; Const_A=1/K_ini, K_ini=50000MPa/m

  const_a=0.000004975 ; Const_A=1/K_ini, K_ini=201,000MPa/m
;
; --- Compute b_r
; --- b_r=sqrt(12*k_fr/phi_fr)
;
; --- b_r    : reference fracture aperture (stored as 1/b_r)
; --- k_fr   : reference fracture permeability (before excavation)
; --- phi_fr: refernece fracture porosity
;
  loop i (1,3)
    b_r(i)=sqrt(frFP(i)/12/frPerm(i))
  endLoop
;
; --- computer fracture normal stress (vsig_n, hsig_n)
;

  loop i (1,izones)
    loop j (1,jzones)
      loop jk (1,2)
        if j <= 229 then
          $theta1 = m_janglev(1)
          $theta2 = m_jangleh(1)
        else
          if j <= 301 then
            $theta1 = m_janglev(2)
            $theta2 = m_jangleh(2)
          else
            $theta1 = m_janglev(3)
            $theta2 = m_jangleh(3)
          end_if
        end_if
        if jk = 1 then
          $theta = $theta1
        else
          $theta = $theta2
        end_if
        $cs    = cos ($theta)
        $sn    = sin ($theta)
        $cla   = $cs  * $cs
        $clb   = $sn  * $sn
        $csn   = $cs  * $sn
        $clc   = 2.0  * $csn
; --- Find normal stress referred to the shear plane ---
        $s22p = sxx(i,j) * $clb + syy(i,j) * $cla - sxy(i,j) * $clc
        if jk = 1 then
          vsig_n(i,j) = $s22p
        else
          hsig_n(i,j) = $s22p
        end_if
      endLoop
    endLoop
  endLoop
;
; --- compute constant c
;
  loop i (1,izones)
    loop j (1,jzones)
      if j <= 229 then
        $a5=b_r(1)          ; Tptpll
      else
        if j <= 301 then
          $a5=b_r(2)        ; Tptpmn
        else
          $a5=b_r(3)        ; Tptpul
        end_if
      end_if
      $vsig=-vsig_n(i,j)+jtension(i,j) ; including fracture tensile strength
      $hsig=-hsig_n(i,j)+tension(i,j)
      if vsig_n(i,j) <= jtension(i,j) then
        if vsig_n(i,j) # 0 then
          const_cv(i,j)=(-1+sqrt(1+4*$vsig*const_a*$a5))/2/$vsig
        end_if
      end_if
```

```
      if hsig_n(i,j) <= tension(i,j) then
        if hsig_n(i,j) # 0 then
          const_ch(i,j)=(-1+sqrt(1+4*$hsig*const_a*$a5))/2/$hsig
        end_if
      end_if
    endLoop
  endLoop
end
def print_cv
;
; --- print const_cv into a file for later use
;
  loop i (1,izones)
    loop j (1,jzones)
      ttt = fstring(const_cv(i,j),7)
      $i  = fstring(i,0)
      $j  = fstring(j,0)
      oo  = out('init ex_2 ' + ttt + ' i= ' + $i + ' j= ' + $j)
    endLoop
  endLoop
end
def print_ch
;
; --- print const_ch into a file for later use
;
  loop i (1,izones)
    loop j (1,jzones)
      ttt = fstring(const_ch(i,j),7)
      $i  = fstring(i,0)
      $j  = fstring(j,0)
      oo  = out('init ex_3 ' + ttt + ' i= ' + $i + ' j= ' + $j)
    endLoop
  endLoop
end
compute_c
;set log const_cv.dat
;print_cv
;set log off
set log const_ch.dat
print_ch
set log off
```

Computepermeability for FLAC models (permeability_bb.fis)

```
;
; Compute fracture permeability using
; k_f=k_fr*{a/[c(c*sig_n+1)*b_r]+e_fp/phi_fr}^3
; k_f    : current fracture permeability
; k_fr   : reference fracture permeability
; phi_fr: reference fracture porosity
; a      : constant 1/(init fracture stiffness)
; c:     : constant (read in from either const_cv or const_ch.dat
; b_r    : fracture aperture
; sig_n : fracture normal stress
; e_fp   : fracture plastic strain
;
; for UBIQUITOUS MODEL
;
```



```
def permeability
  array frFP(3)
  array frPerm(3)
  array b_r(3)
  array vsig_n(400,400) ; fracture normal stress, set 1
  array hsig_n(400,400) ; fracture normal stress, set 2
;  array const_cv(400,400)
;  array const_ch(400,400)
  array m_janglev(3)
  array m_jangleh(3)
  array adil(3)          ; filation angle

  float $s22p $theta $cs $sn $cla $clb $csn $clc $clab
  float $theta1 $theta2 $model_idx

;
; --- model_idx=1: strain-hardening/softening model
; --- model_idx=2: strain-hardening/softening ubiquitous joint model
;
  model_idx = 2
  if model_idx = 1 then
    adil(1)=23    * degrad
    adil(2)=24.08 * degrad
    adil(3)=23    * degrad
  else
    $ddil=20.5 * degrad
    adil(1)=$ddil
    adil(2)=$ddil
    adil(3)=$ddil
  end_if

;
; --- Define angles of fracture sets
;
  m_janglev(1)=80.5 * degrad
  m_janglev(2)=83.5 * degrad
  m_janglev(3)=82.0 * degrad
  m_jangleh(1)= 5.0 * degrad
  m_jangleh(2)= 9.0 * degrad
  m_jangleh(3)=14.0 * degrad

;
; --- frFP(1); define reference fractrure porosity, Tptpll
; --- frFP(2); define reference fractrure porosity, Tptpmn
; --- frFP(3); define reference fractrure porosity, Tptpul
; --- Data from from MultiscaleThermoHydrologicModelAMR, 2001 Rev00 ICN02
;
  frFP(1)=0.011
  frFP(2)=0.01
  frFP(3)=0.0066
;
; --- frPerm: reference permeability
; --- Data from from MultiscaleThermoHydrologicModelAMR, 2001 Rev00 ICN02
;
  frPerm(1)=1.29e-12
  frPerm(2)=2.76e-13
  frPerm(3)=5.50e-13

;  const_a=0.00002 ; const_a=1/K_ini, K_ini=50000MPa/m (value assumed)

  const_a=0.000004975 ; Const_A=1/K_ini, K_ini=201,000MPa/m
;
; --- Compute b_r
; --- b_r=sqrt(12*k_fr/phi_fr)
;
; --- b_r    : reference fracture aperture (STORED AS 1/b_r)
; --- k_fr   : reference fracture permeability (before excavation)
; --- phi_fr: refernece fracture porosity
;
```

```
  loop i (1,3)
    b_r(i)=sqrt(frFP(i)/12/frPerm(i))
  endLoop

;
; --- computer fracture normal stress (vsig_n, hsig_n)
;

  loop i (1,izones)
    loop j (1,jzones)
      loop jk (1,2)
        if j <= 229 then
          $theta1 = m_janglev(1)
          $theta2 = m_jangleh(1)
        else
          if j <= 301 then
            $theta1 = m_janglev(2)
            $theta2 = m_jangleh(2)
          else
            $theta1 = m_janglev(3)
            $theta2 = m_jangleh(3)
          end_if
        end_if
        if jk = 1 then
          $theta = $theta1
        else
          $theta = $theta2
        end_if
        $cs    = cos ($theta)
        $sn    = sin ($theta)
        $cla   = $cs  * $cs
        $clb   = $sn  * $sn
        $csn   = $cs  * $sn
        $clc   = 2.0  * $csn
;
; --- Find normal stress referred to the shear plane ---
;
        $s22p = sxx(i,j) * $clb + syy(i,j) * $cla - sxy(i,j) * $clc
        if $s22p > 0 then
          $s22p = 0;
        end_if
        if jk = 1 then
          vsig_n(i,j) = $s22p
        else
          hsig_n(i,j) = $s22p
        end_if
      endLoop
    endLoop
  endLoop
;
; --- compute permeability
;

  loop i (1,izones)
    loop j (1,jzones)
      if j <= 229 then
        $a5=b_r(1)         ; Tptpll
        $a4=frFP(1)
        $a3=frPerm(1)
        $a2=adil(1)
        $a1=23     * degrad
      else
        if j <= 301 then
          $a5=b_r(2)       ; Tptpmn
          $a4=frFP(2)
          $a3=frPerm(2)
          $a2=adil(2)
          $a1=24.08 * degrad
        else
          $a5=b_r(3)       ; Tptpul
          $a4=frFP(3)
          $a3=frPerm(3)
          $a2=adil(3)
          $a1=23     * degrad
        end_if
      end_if
      $cv    = ex_2(i,j)
      $ch    = ex_3(i,j)
;      $u_nv = const_a/($cv*($cv*(abs(vsig_n(i,j))+tension(i,j))+1))*$a5
;      $u_nh = const_a/($ch*($ch*(abs(hsig_n(i,j))+tension(i,j))+1))*$a5
      $u_nv1= -vsig_n(i,j)+jtension(i,j) ; fracture tensile strength
      $u_nh1= -hsig_n(i,j)+ tension(i,j)
      if $u_nv1 < 0 then
        $u_nv1 = 0
      end_if
      if $u_nh1 < 0 then
        $u_nh1 = 0
      end_if
      if $cv # 0 then
        $u_nv = const_a/($cv*($cv*$u_nv1+1))*$a5
      end_if
      if $ch # 0 then
        $u_nh = const_a/($ch*($ch*$u_nh1+1))*$a5
      end_if
      if model_idx = 1 then
;
; --- for strain-hardening/softening model, fracture set 1
;
        ex_4(i,j)=($u_nv+(e_plastic(i,j)*tan($a2)+et_plastic(i,j))/$a4)^3 ; ratio
;
; --- for strain-hardening/softening model, fracture set 2
;
        ex_5(i,j)=($u_nh+(e_plastic(i,j)*tan($a2)+et_plastic(i,j))/$a4)^3 ; ratio
      else
;
; --- for strain-hardening/softening ubiquitous joint model, fracture set 1
;
        ex_4(i,j)=($u_nv+(ej_plastic(i,j)*tan($a2)+etj_plastic(i,j))/$a4)^3 ; ratio
;
; --- for strain-hardening/softening ubiquitous joint model, fracture set 2
; --- no information is available on ej_plastic nor etj_plastic for set 2
; --- becasue fracture set 2 was not modeled
;
;        ex_5(i,j)=($u_nh+(ej_plastic(i,j)*tan($a2)+etj_plastic(i,j))/$a4)^3 ; ratio
        ex_5(i,j)=($u_nh+(e_plastic(i,j)*tan($a1)+et_plastic(i,j))/$a4)^3 ; ratio
      end_if
      ex_6(i,j)=ex_4(i,j)*$a3 ; permeability, fracture set 1
      ex_7(i,j)=ex_5(i,j)*$a3 ; permeability, fracture set 2
    endLoop
  endLoop
end
;permeability
```

Verification of the FISH functions for constant c and permeability/permeability ratio calculation.

If a FLAC model result is used to calculate Constant c and the same model result is

then again used to calculate permeability/ permeability ratio, the permeability ratio should come out as unity. The following FLAC run verifies that. This is a proof that the two FISH functions work as they suppose to do.

```
                        Verification_Perm_eTAve.log
* FLAC log-file opened  2-Mar-03  15:00
DECOVALEX III DST TM analysis, RMQ2, Ubiquitous Model

From File : diii_400x400vu_rmq2_excavate.sav
>read_c

>permeability

Permeability Ratio Presented in row, Set 1

1.0000000E+00 1.0000001E+00 1.0000000E+00 9.9999995E-01

1.0000001E+00 9.9999993E-01 9.9999996E-01 9.9999997E-01

1.0000000E+00 9.9999997E-01 1.0000000E+00 9.9999993E-01

9.9999993E-01 9.9999993E-01 1.0000000E+00 9.9999997E-01

>permeability_hole

; a = 4.97E-06

57-1 Permeability, Set 1 2.7600000E-13

57-1 Permeability, Set 1 2.7599999E-13

57-1 Permeability, Set 1 2.7600001E-13

57-1 Permeability, Set 1 2.7599999E-13
```

03/02/2003

Sui-Min Hsiung

Data file used to calculate permeability

```
;FISH to calculate permeability for fracture porosity
;permeability.dat
;set echo off
set log off
  ca const_cv.dat
  ca const_ch.dat
ca permeability_bb.fis
;ca PermeabilityInHole_bb.fis
;set log Flac_Perm.log
permeability
;permeability_hole
;set log off
```

03/18/2003

Sui-Min Hsiung

I have reviewed this scientific notebook and find it in compliance with QAP-001. There is sufficient information regarding procedures used for conducting tests, acquiring and analyzing data so that another qualified individual could repeat the activity

9-29-03

ADDITIONAL INFORMATION FOR SCIENTIFIC NOTEBOOK #: 491

Document Date:	11/30/2001
Availability:	Southwest Research Institute® Center for Nuclear Waste Regulatory Analyses 6220 Culebra Road San Antonio, Texas 78228
Contact:	Southwest Research Institute® Center for Nuclear Waste Regulatory Analyses 6220 Culebra Road San Antonio, TX 78228-5166 Attn.: Director of Administration 210.522.5054
Data Sensitivity:	■"Non-Sensitive" □ Sensitive □"Non-Sensitive - Copyright" □ Sensitive - Copyright
Date Generated:	04/08/2002
Operating System: (including version number)	Windows
Application Used: (including version number)	Various
Media Type: (CDs, 3 1/2, 5 1/4 disks, etc.)	1 zip drive
File Types: (.exe, .bat, .zip, etc.)	Various
Remarks: (computer runs, etc.)	Media contains: Data input and results from DDA modeling for DECOVALEX III Task 2—thermal mechanical modeling of the drift scale heater test.