Department of Energy
Carlsbad Area Office
P. O. Box 3090
Carlsbad, New Mexico 88221

A-93-02
II-I-10

February 26, 1997

Ms. Ramona Trovato, Director
US Environmental Protection Agency
Office of Radiation Programs
401 M. Street SW
Washington, DC 20460

Dear Ms. Trovato:

The Carlsbad Area Office (CAO) is pleased to submit this fifth and final group of responses to requests for supplemental information contained in the EPA letter of December 19, 1996 to Al Alm. We have submitted four previous groups (January 17 and 24, and February 7 and 14) as the material was developed in order to provide EPA with the requested information as early as possible. This final group completes our responses to your request. As in the earlier submittals, we have reproduced the issue verbatim from the December 19, 1996 letter and inserted the CAO response in each case.

We are confident that the EPA will find this supplemental information helpful in your review process. Should you have any questions regarding this information or require anything further, please contact me at (505) 234-7300.

Sincerely,

George E. Dials
Manager

Enclosure

cc:
F. Marcinowski, EPA

97-0458
UFC 1200

Printed on recycled paper

EPA Comment
Enclosure 1, Page 4
194.14(a)(2)

Comment Text

194.14(a)(2)
Part 194 requires a description of the geology, geophysics, hydrogeology, hydrology, and geochemistry of the disposal system and its vicinity and how these conditions are expected to change and interact over time.

The CCA does not include updated information obtained from recent site investigation-related studies. The CCA states that "these recent studies... provide detailed information necessary to construct the conceptual models," but does not summarize what these studies entailed and how they impact the understanding of site characteristics relative to older data. The CCA implies, on page 2-9, Section 2.1, that these data are included in Chapter 6 and associated appendices.

The CCA should include more detailed information pertaining to the more recent studies so that an understanding of the site conditions and linkages of this information with the conceptual model development can be achieved. In addition, the CCA should provide a discussion of newly acquired site-specific information (i.e., information on Culebra and retardation studies presented at the 10/11/96 meeting between DOE and State of New Mexico representatives), and discuss how this information impacts site conceptual model development.

DOE Response

Detailed information pertaining to recent development on the conceptual model for transport in the Culebra that was presented at the October 11, 1996 meeting between DOE and State of New Mexico representatives is contained in SAND97-0194 *Conceptual Model for Transport Processes in the Culebra Dolomite Member, Rustler Formation* by R. M. Holt. This report is in review at Sandia National Laboratories and is not available for distribution at this time. Arrangements can be made through the records center for EPA representatives to see a high-quality, near-final draft of this report. A publication pre-print will be available for release in March. The concepts of Culebra transport contained in this report were used in development of the performance assessment parameter values. These concepts represent a significant increase in confidence regarding the predictability of Culebra transport processes and a significant decrease in associated uncertainties.

The SAND report by Holt focusses on the Culebra field tests and conceptual insight gained from them. It also contains a brief description of the intact-core column testing elution experimental technique; identification of the radioactive tracers used as non-sorbing and sorbing tracers; presentation of typical elution breakthrough curves for Na-22 and U-232; observation of the fact that, of the actinides, only Np and U have eluted (so Am, Pu, and Th have not). In addition, the report contains petrographic descriptions of the core columns and interpretations of the advective and diffusive porosity distributions in the cores.

EPA Comment
Enclosure 1, page 5
194.22(a)(2)(iii)

Comment Text

Models and Computer Codes
Part 194 requires that the CCA include a description of conceptual models and scenario construction used to support the CCA. In addition, Part 194 states that documentation of all models and computer codes must be included.

There is a significant problem with the completeness of the CCA documentation that deals with the CCDF formalism and the codes that implement it. While the current versions of the formalism and codes may be doing exactly what is required of them, and while those intended activities may be what is needed for the PA, it is often difficult and sometimes impossible to determine what it is, exactly, that they *are* doing and to *verify* that this is all happening as intended. The documentation is, in places, too sparse to enable a reviewer to acquire a comprehensive understanding of the current form of the formalism and codes.

DOE needs to provide documentation for the CCDF formalism and for the codes that implement it. Specific examples are provided below.

DOE Response

Because the comment on 194.22(a)(2)(iii), enclosure 1, page 5 and 194.23(a)(3)(ii), enclosure 1, page 7, raise very similar points, the DOE has chosen to address them with a common response.

Because these two comments raise very similar points, the DOE has chosen to address them in a single response.

Many of the questions raised here are addressed in Appendix SA of the CCA, where the construction of CCDFs for cuttings and cavings releases, spallings releases, and direct brine releases are described. For example, Section SA.3, and Table SA-1 specifically, describe the construction of CCDFs for cuttings and cavings, including a description of the use of interpolation. Section SA.5, and Tables SA-2 and SA-3, contain similar information for spallings releases. Section SA.8, and Tables SA-4 and SA-5 provide the analogous information for direct brine releases.

Additional documentation of the construction of CCDFs has been provided to the EPA

EPA Comment
Enclosure 2, Page 3
194.23(a)(3)(iv)

Comment Text

194.23(a)(3)(iv)
Section 194.23(a)(3)(iv) requires documentation that the "...computer codes are free of coding errors and produce stable results."

One feature of the SECOFL2D computer code (SECO User's Manual) that was not tested was that the code implements the transition from a regional grid to a local grid.

The Department needs to devise a test of this key component and document the accuracy of the bilinear interpolation scheme for both boundaries and properties.

DOE Response

PRESECOFL2D, the pre-processor for SECOFL2D, is the code that implements the transition from regional properties to local properties for SECOFL2D. Documentation of the testing of PRESECOFL2D which documents the accuracy of the bilinear interpolation scheme for the properties is found in the QA records for PRESECOFL2D: *PRESECOFL2D, Version 4.02ZO, Version Date 5/20/94, Requirements Document and Verification & Validation Plan* contains the design of the test problem, and the results of the test are found in *PRESECOFL2D, Version 4.02ZO, Version Date 5/20/94, Validation Document* (both documents are attached).

SECOFL2D implements the transition from the solution on the regional grid to the local grid boundary. Documentation of the testing of SECOFL2D which documents the accuracy of the bilinear interpolateion scheme for the boundaries is found in the QA records for SECOFL2D: *SECOFL2D, Version 3.01ZO, Version Date 8/9/93, Requirements Document and Verification & Validation Plan* contains the design of the test problem, and the results of the test are found in *SECOFL2D, Version 3.01ZO, Version Date 8/9/93, Validation Document* (both documents were previously supplied to the EPA).

WIPP PA

REQUIREMENTS DOCUMENT AND VERIFICATION AND VALIDATION PLAN (RD/VVP)

for

PRESECOFL2D

Version 4.02ZO

WPO# 23318

INFORMATION ONLY

TABLE OF CONTENTS

1.0 INTRODUCTION 4
1.1 Software Identifier 4
1.2 Points of Contact 4
1.3 Description 4
2.0 REQUIREMENTS 4
2.1 Functional Requirements 4
2.2 Performance Requirements 5
2.3 Attribute Requirements 5
2.4 External Interface Requirements 5
2.5 Other Requirements 5
3.0 DESIGN OVERVIEW 5
3.1 I/O Description 5
3.2 Context Diagram 6
3.3 Design Constraints 6
4.0 ADDITIONAL FUNCTIONALITY TO BE TESTED 6
5.0 FUNCTIONALITY NOT TESTED 7
6.0 TESTING ENVIRONMENT 7
7.0 STATIC TESTING 8
8.0 COVERAGE TESTING 9
9.0 FUNCTIONAL TESTING 10
9.1 Test Case #1 - Interpolation Test 12
9.1.1 Test Objective 12
9.1.2 Test Procedure 12
9.1.3 Input Files 15
9.1.4 Acceptance Criteria 24
10.0 INSTALLATION AND REGRESSION TESTING 24
11.0 REFERENCES 25
12.0 APPENDICES 26
APPENDIX A - REVIEW COMMENTS
APPENDIX B - PRIMITIVE BASELINE DOCUMENTS

INFORMATION ONLY

LIST OF FIGURES

3.1 Data Flow for PRESECOFL2D 6

9.1 SF2D1_TEST_NODBG_RUN.COM - Command file for Running The Test Case 11

9.1-1 SF2D1_PRESECOFL_TEST.INP - ASCII Input File for Running Test Case 1 16

9.1-2 SF2D1_READ_PRP - Program Used to Convert the Binary Property File to ASCII 17

9.1-3 SF2D1_TEST_READ_1 - Program Used Read Selected Data from the Binary Property File 21

INFORMATION ONLY

1.0 INTRODUCTION

The purpose of this document is to identify the requirements of the PRESECOFL2D code and to describe the testing, including coverage analysis and regression testing, that will be performed. This code was qualified as Level A under QAP 19-1 Rev. F. The QA package for this code, referred to as the primitive baseline, is on file in the SWCF. The verification section of that package demonstrates that the functional requirement R.1 stated in this document is satisfied.

1.1 Software Identifier

PRESECOFL2D Version 4.02ZO WIPP PA Code Prefix: SF2D1

1.2 Points of Contact

Code Sponsor: Rebecca L. Blaine
Code Consultant: Rebecca L. Blaine
Ecodynamics Research Associates, Inc.
P. O. Box 9229
Albuquerque, New Mexico 87119
(505)843-7445

1.3 Description

The purpose of PRESECOFL2D is to create all of the input files required to run the code SECOFL2D. Material properties and grid information are obtained from CAMDAT databases.

2.0 REQUIREMENTS

2.1 Functional Requirements

R.1 - Creates all input files needed to run SECOFL2D. These files are:
1) An ASCII file containing run parameters
2) A binary file containing regional material property information obtained from a CAMDAT database.
3) A binary file containing local material property information.

R.2 - Interpolates material properties from the regional grid to the local grid.

INFORMATION ONLY

2.2 Performance Requirements

This code has no performance requirements.

2.3 Attribute Requirements

This code has no attribute requirements.

2.4 External Interface Requirements

R.3 - This code reads an ASCII input file containing run parameters.

R.4 - This code reads an input regional CAMDAT database with grid information material properties.

R.5 - This code reads an input local CAMDAT database with grid information.

R.6 - This code produces all of the input files necessary to run SECOFL2D.

R.7 - This code must be linked to the libraries CAMCON_LIB, CAMDAT_LIB and CAMSUPES_LIB.

2.5 Other Requirements

There are no other requirements for PRESECOFL2D that need verification.

3.0 DESIGN OVERVIEW

3.1 I/O Description

The following is a description of the input and output files associated with PRESECOFL2D. The first three output files are used as input to SECOFL2D. All file names shown here are for example only. The user chooses the appropriate file names for the particular calculation being done at PRESECOFL2D execution time.

Input Files:

presecofl.inp - ASCII input file containing run parameters.

secofl_loc.cdb - Input local CAMDAT database containing local grid information.

secofl_reg.cdb - Input regional CAMDAT database containing material property and grid information.

INFORMATION ONLY

Output Files:

secofl.inp - ASCII output file created by PRESECOFL2D containing run directives and boundary condition information. Used as an input file by SECOFL2D.

secofl_reg.prp - Binary output file created by PRESECOFL2D containing the grid material property information for the regional grid. Used as an input file by SECOFL2D.

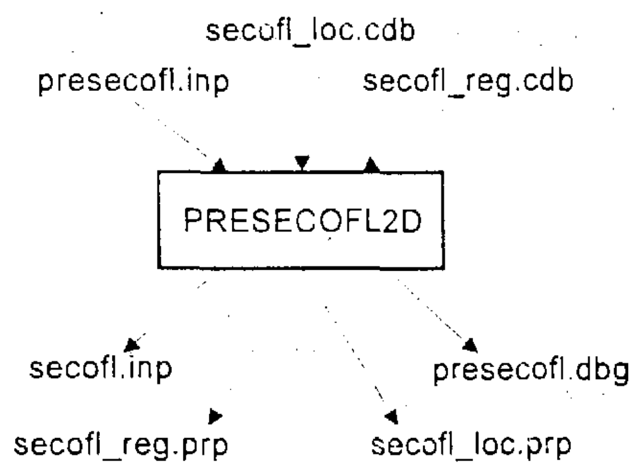
secofl_loc.prp - Binary output file created by PRESECOFL2D containing the grid and material property information for the local grid. Used as an input file by SECOFL2D.

presecofl.dbg - ASCII output debug/log file

3.2 Context Diagram

The data flow design for PRESECOFL2D is presented using the context diagram in **Figure 3.1**.

FIGURE 3.1
Data Flow for PRESECOFL2D



3.3 Design Constraints

This code has already been developed and therefore has no design constraints.

4.0 ADDITIONAL FUNCTIONALITY TO BE TESTED

There is no additional functionality to be tested.

INFORMATION ONLY

5.0 FUNCTIONALITY NOT TESTED

The following features will not be tested because they will not be used as part of the WIPP PA.

1) Input of well parameters - standard and time dependent.
2) Input of lake, river, and precipitation factors.

6.0 TESTING ENVIRONMENT

Hardware Platform:	DEC Alpha
Operating System:	OpenVMS Version 6.1.
Directory:	WP$TESTROOT:[SF2D1]

INFORMATION ONLY

7.0 STATIC TESTING

Static testing will be performed using both the source code analyzer FLINT and the source code analyzer DECset-SCA. FLINT is used to detect non-ANSI standard FORTRAN coding. DECset-SCA is used to reveal any uncalled modules, to reveal any unreferenced variables, and the display the calling tree. Since PRESECOFL2D is a multimodule program, a single file should be created with all of the modules appended together. FLINT should be used on this file. The output from FLINT will be examined by the code sponsor and any findings will be resolved.

DECset-SCA will be invoked during the build process. The build process will be performed using the build script SF2D1_BUILD.COM. This process will be described in more detail in the *WIPP PA Implementation Document for PRESECOFL2D Version 4.02ZO*. As a result of the build process, a debug executable capable of doing PCA (Performance Coverage Analysis) analysis will be created and the SCA code analyzer will create the following files:

SF2D1_SCA_MOD_NOT_REF.TXT
SF2D1_CALLTREE.TXT
SCA$EVENT.DAT

These files will be described in complete detail in the *WIPP PA Validation Document for PRESECOFL2D Version 4.02ZO*. A complete description of the process outlined below to convert the data in SCA$EVENT.DAT to a readable format will also be contained in that document.

Following the build process, SCA will be invoked at the VMS $ prompt and the following commands should be executed on the SCA$EVENT.DAT file in the WP$TESTROOT:[SF2D1.SCA] directory to convert data in the file to ASCII format:

SCA> **set lib []**
SCA> **show module/all/output=sf2d1_modules.out**
SCA> **exit**

The SCA output will be examined by the code sponsor. Unreachable coding will either be changed so that it is reachable, or justified as not being relevant to the performance of the software as it relates to WIPP PA.

INFORMATION ONLY

8.0 COVERAGE TESTING

The coverage analysis for PRESECOFL2D will be performed using DECset PCA. This tool will be used to identify modules that are not exercised by the test set. A unique executable will be created for the purpose of coverage analysis during the build process. This executable will be generated by using the SCMS build script, SF2D1_BUILD.COM. The creation of this executable will be described in more detail in the *WIPP PA Implementation Document for PRESECOFL2D Version 4.02ZO*.

The command file SF2D1_TESTCASE_PCA_CUM.COM will be used to run the PRESECOFL2D test cases with PCA. The PCA output will be examined by the code sponsor and any unexercised modules must be justified.

INFORMATION ONLY

9.0 FUNCTIONAL TESTING

Verification of functional requirement R.1 has been done and is on record in the primitive baseline submitted to the SWCF. The following test case will be used for verifying requirement R.2, for coverage analysis, and for regression testing.

All files needed to test PRESECOFL2D can be fetched from the SCMS. The directory WP$TESTROOT:[SF2D1.TESTCASES] on the DEC Alpha has been created and should be used for running the test case. To verify requirement R.2 and for regression testing, Test Case 1 (described below) is run using the command file SF2D1_TEST_NODBG_RUN.COM. This file is shown in **Figure 9.1**.

INFORMATION ONLY

FIGURE 9.1
SF2D1_TEST_NODBG_RUN.COM - Command file for Running The Test Case

```
$ set noon
$!
$!==========================================================================
$!
$! This file runs all test cases for PRESECOFL2D using the NODEBUG executable
$!
$!==========================================================================
$!
$!
$ TESTDIR_SYM == "wp$testroot:[sf2d1.testcases]"
$ PRESECOFL2D == "$ wp$prodroot:[sf2d.exe]presecofl2d.exe"
$!
$ define /proc testdir wp$testroot:[sf2d1.testcases]
$!
$ show sym testdir_sym
$ IF F$MODE () .EQS. "BATCH" .AND. F$ENVIRONMENT ("DEPTH") .EQ. 0 THEN -
$    SET DEFAULT 'TESTDIR_SYM'
$!
$!
$! Set up the first test case and run PRESECOFL2D
$!
$  WRITE SYS$OUTPUT "STARTING THE RUN FOR THE PRESECOFL2D TEST CASE"
$!
$  PRESECOFL2D SF2D1_REGION_TEST.CDB SF2D1_REGION_TEST.CDB -
     SF2D1_LOCAL_TEST.CDB CANCEL SF2D1_PRESECOFL_TEST.INP SECOFL2D_TEST.INP -
     SF2D1_REGION_TEST.PRP SF2D1_LOCAL_TEST.PRP CANCEL PRESECOFL_TEST.DBG
$!
$  write sys$output "SUCCESSFUL COMPLETION OF TEST EXECUTION"
$!==========================================================================
```

INFORMATION ONLY

9.1 Test Case 1 - Interpolation Test and Regression Test

9.1.1 Test Objective

This test is run to test the interpolation of properties from the regional grid to the local grid. This test case satisfies requirement R.2. This test is also run for regression testing. The purpose of the regression test is to show that PRESECOFL2D produces identical results when writing the regional binary property file as were produced in the verification process documented in the primitive baseline. The entire verification process is not relevant to this regression test and so only the portion of the results in the primitive baseline that relate to writing the binary property file will be used. The regression test will show that data currently written to the regional binary property file is exactly the same as in the baseline verification process. PRESECOFL2D also writes an ASCII input file and a debug file. The ASCII input file was verified by visual inspection and by demonstrating that SECOFL2D read and used the input data correctly. The nature of this verification is not useful for regression testing and is not included here. The debug file does not contain information relevant to the verification process in the baseline and is not documented there.

9.1.2 Test Procedure

The following is a list of all the files that are associated with this test case. All of the input files are input for PRESECOFL2D. The output files have designated in parenthesis which program has produced them. The testing tools have designated, in parenthesis, the test they are used for.

Input files:

WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_REGION_TEST.CDB
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_LOCAL_TEST.CDB
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_PRESECOFL_TEST.CDB

Output Files:

WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_REGION_TEST.PRP (PRESECOFL2D)
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_LOCAL_TEST.PRP (PRESECOFL2D)
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_REGION_TEST.ASC (READ_PRP)
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_LOCAL_TEST.ASC (READ_PRP)
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_TEST_READ.OUT (TEST_READ_1)

Testing Tools:

WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_READ_PRP.FOR (Interpolation)
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_TEST_READ_1.FOR (Regression)

Run the test case with the command file shown in **Figure 9.1**. The procedure to follow for regression testing is stated in Section 10 below. For the interpolation test, following the

INFORMATION ONLY

execution of the command file, the program SF2D1_READ_PRP (compiled and linked from SF2D1_READ_PRP.FOR) should be executed twice, once using the regional data and once using the local data. This program converts the regional binary property file, SF2D1_REGION_TEST.PRP, to an ASCII file, SF2D1_REGION_TEST.ASC, and the local binary property file, SF2D1_LOCAL_TEST.PRP, to an ASCII file, SF2D1_LOCAL_TEST.ASC, so that they can be examined with an editor. All of the material properties used to run SECOFL2D are constants except for hydraulic conductivity. Therefore, checks of porosity (phi) and bulk compressibility (alpha) need only show that they have remained constant. A hand-calculation is done for hydraulic conductivity to show that the values have been correctly interpolated (using bilinear interpolation) from the regional grid to the local grid.

To verify that the porosity and bulk compressibility have remained constant, edit the file SF2D1_REGION_TEST.ASC. First, search for the string "alpha". Examination of this array shows that this is a constant value of 7.57E-01 for each element in the grid. (The value of bulk compressibility is 7.57E-10 on the database, but it has been scaled by 10^9 as required by SECOFL2D.) Next, search for the string "phi". Examination of this array shows that this is a constant value of 1.4542E-01 for each element in the grid. Now repeat this process with the file SF2D1_LOCAL_TEST.ASC. Searching for the string "alpha" shows that this array is a constant value of 7.57E-01, and searching for the string "phi" shows that this array is a constant of 1.4542E-01.

To verify the interpolation for the non-constant values of hydraulic conductivity, four points are checked by hand-calculation. The four element locations in the local grid are cells (12,15), (20,47), (35,25) and (40,40). First, the element locations relative to the regional grid are found. The element locations in meters are converted to locations (in meters) in regional domain. The x and y offsets of the local grid from the origin of the regional grid are 12826.1 and 10665.8 respectively. The angle of rotation is 38° counter clockwise. The coordinates (in meters) of the cell centers of the elements in the local grid are:

Element 1 (12,15):	1437.5, 1812.5
Element 2 (20,47):	2437.5, 5812.5
Element 3 (35,25):	4312.5, 3062.5
Element 4 (40,40):	4937.5, 4937.5

The regional coordinates are calculated by:

Element 1: $12826.1 + 1437.5\cos 38° - 1812.5\sin 38° = 12842.97903$
$10665.8 + 1812.5\cos 38° + 1437.5\sin 38° = 12979.08286$

Element 2: $12826.1 + 2437.5\cos 38° - 5812.5\sin 38° = 11168.34389$
$10665.8 + 5812.5\cos 38° + 2437.5\sin 38° = 16746.78735$

Element 3: $12826.1 + 4312.5\cos 38° - 3062.5\sin 38° = 14338.93311$
$10665.8 + 3062.5\cos 38° + 4312.5\sin 38° = 15734.12305$

INFORMATION ONLY

Element 4: 12826.1 + 4937.5cos38° - 4937.5sin38° = 13677.07456
10665.8 + 4937.5cos38° + 4937.5sin38° = 17596.43163

Next, the cell centers of the adjacent cells are used (the properties reside at the cell center) to calculate the proportional distances used in bilinear interpolation.

Element 1 proportions from coordinates of regional cells (32,10) and (33,11):

xp = (12842.97903 - 12625.0)/(12850.0 - 12625.0) = 0.9687956889

yp = (12979.08286 - 12750.0)/(13125.0 - 12750.0) = 0.6108876267

Element 2 proportions from coordinates of regional cells (24,34) and (25,35):

xp = (11168.34389 - 11075.0)/(11225.0 - 11075.0) = 0.6222926

yp = (16746.78735 - 16725.0)/(16950.0 - 16725.0) = 0.0968326666

Element 3 proportions from coordinates of regional cells (39,26) and (40,27):

xp = (14338.93311 - 14150.0)/(14450.0 - 14150.0) = 0.6297770333

yp = (15734.12305 - 15600.0)/(15800.0 - 15600.0) = 0.67061525

Element 4 proportions from coordinates of regional cells (37,38) and (38,39):

xp = (13677.07456 - 13675.0)/(13887.5 - 13675.0) = 0.009762635294

yp = (17596.43163 - 17475.0)/(17630.0 - 17475.0) = 0.783429871

Then, the bilinear interpolation is done.

Element 1:

a1 = 498.5 + 0.9687956889(1498.0 - 498.5) = 1466.811291

a2 = 670.3 + 0.9687956889(707.4 - 670.3) = 706.2423201

a = 1466.811291 + 0.6108876267(706.2423201 - 1466.811291) = 1002.189117

Element 2:

a1 = 182.5 + 0.6222926(267.1 - 182.5) = 235.145954

a2 = 109.1 + 0.6222926(119.8 - 109.1) = 115.7585308

a = 235.145954 + 0.0968326666(115.7585308 - 235.145954) = 223.5853515

Element 3:

a1 = 495.8 + 0.6297770333(960.2 - 495.8) = 788.2684543

a2 = 816.1 + 0.6297770333(1041.0 - 816.1) = 957.7368548

a = 788.2684543 + 0.67061525(957.7368548 - 788.2684543) = 901.9165481

Element 4:

a1 = 159.6 + 0.009762635294(277.6 - 159.6) = 160.751991

a2 = 117.4 + 0.009762635294(28.47 - 117.4) = 116.5318088

a = 160.751991 + 0.783429871(116.5318088 - 160.751991) = 126.1085794

INFORMATION ONLY

FIGURE 9.1-1
SF2D1_PRESECOFL_TEST.INP - ASCII Input File for Running Test Case 1

```
!------------------------------------------------------------------------------
! Test case for PRESECOFL2D
!------------------------------------------------------------------------------
!
!
*RUN_TYPE
    SET_AQFR, TYPE=DEF, AQFR_ID=1.001
    SET_GREG, TYPE=def, GREG_ID=1.002
    SET_GLOC, TYPE=DEF, GLOC_ID=1.025, X_REL=12826.1, Y_REL=10665.8, THETA=38.0
    FLOW, both=transient, LOC_BOUND=yes
*constant
     screen_io=BRIEF
*INITIAL_CONDITIONS
     CAMDAT, HEAD=HEADat, TYPE=ATTR
     RESET
*ATTR
 HYCND_Y=HYCND_X
*REG_TIME
     SEQ, NUM_STEP=10, START=0.0, FINISH=10000.
     AUTO, OPTION=DEL_TIME
*LOCAL_TIME
     SEQ, NUM_STEP=10, START=0.0, FINISH=10000.
     AUTO, OPTION=DEL_TIME
*BOUNDARY
     REG, BOUNDARY=LOWER, TYPE=HEAD, END=25000., Clim_frac=0.
     REG, BOUNDARY=UPPER, TYPE=HEAD, END=10000., Clim_frac=1.
     Reg, BOUNDARY=UPPER, TYPE=HEAD, END=17300., Clim_frac=0.
     REG, BOUNDARY=UPPER, TYPE=GRAD, END=25000., clim_frac=0.
     REG, BOUNDARY=RIGHT, TYPE=HEAD, END=27240., clim_frac=0.
     REG, BOUNDARY=RIGHT, TYPE=GRAD, END=30000., clim_frac=0.
     REG, BOUNDARY=LEFT,  TYPE=HEAD, END=4000.,  clim_frac=0.
     REG, BOUNDARY=LEFT,  TYPE=GRAD, END=18595., clim_frac=0.
     REG, BOUNDARY=LEFT,  TYPE=HEAD, END=30000., clim_frac=1.
     LOCAL, BOUNDARY=LOWER, TYPE=HEAD, END=5750.
     LOCAL, BOUNDARY=UPPER, TYPE=HEAD, END=5750.
     LOCAL, BOUNDARY=RIGHT, TYPE=GRAD, END=6625.
     LOCAL, BOUNDARY=LEFT,  TYPE=HEAD, END=6625.
 *REG_CLIMATE
     LAKE, AMP=0.0, CYCLES=1.0, START=0.0, FINISH=25000., RATIO=2.0
     RECHARGE, AMP=0, CYCLES=1., START=0.0, FINISH=10000.
     HEAD, AMP=amplitud, CYCLES=cycles, head_fac=climtidx   &
               START=0.0, FINISH=40000.
 *END
```

INFORMATION ONLY

FIGURE 9.1-2

(Page 1 of 4)

SF2D1_READ_PRP - Program Used to Convert the Binary Property File to ASCII

```
PROGRAM READ_PRP
C
C     Reads binary property file (FOR020.DAT) written by SET_GREG2 and
C     translates it into an ASCII output file.
C     Prompts user for the input file name (i.e. FOR020.DAT) and the
C     user's choice of output file names.
C

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C     IMPLICIT NONE ****************** REMOVED

      dimension TMPARR(0:100,0:100), TMPVEC(0:100)
      INTEGER I, J, IDUM1, IDUM2, ISECO2, IL, JL, IDFIL_GREG
C     REAL RDUM1, RDUM2, RDUM3 ********** superceded
      CHARACTER*40 INFILE
      DATA ISECO2/10/
      DATA IDFIL_GREG/20/

C<><><><><><><><><><><><><>
C...BEGIN PROCEDURES...
C<><><><><><><><><><><><><>

      WRITE(6,*) 'Enter the name of the input file:'
      READ(5,'(A)') INFILE
      OPEN(UNIT=ISECO2,FILE=INFILE,STATUS='OLD',READONLY,
     +     FORM='UNFORMATTED')

      WRITE(6,*) 'Enter the name of the output file:'
      READ(5,'(A)') INFILE
      OPEN(UNIT=IDFIL_GREG,FILE=INFILE,STATUS='NEW')

   10 CONTINUE
        WRITE (IDFIL_GREG,9520)
 9520   FORMAT (' OUTPUT from program SET_GREG2/SET_GLOC2.',/
     $          ' Numeric FORMATS are',/
     $          ' 3(2I6, 1PE13.5)',/
     $          ' 3(2I6, I10, 3x)',/
     $          ' 3( I6, 1PE13.5)',/
     $          ' SET_GREG2 run with ilGR,jlGR, GREG_ID,',
     $                          ' ilGR,jlGR, AQFR_ID = ')
        READ(ISECO2) IL, JL, RDUM1, IDUM1, IDUM2, RDUM2
        WRITE (IDFIL_GREG,953) il, jl, RDUM1, il, jl, RDUM2
  953   FORMAT (3(2I6, 1PE13.5))
        READ(ISECO2) RDUM1, RDUM2, RDUM3
        WRITE (IDFIL_GREG,956) RDUM1, RDUM2, RDUM3
  956   FORMAT (' x1_rel, y1_rel, thet1_rel =',/,1P3E13.5)

c 1-D arrays

  955   FORMAT (3(I6, 1PE13.5))

C       Read x-coordinates
        READ(ISECO2) (TMPVEC(I),I=0,IL+1)
        WRITE (IDFIL_GREG,9540)
 9540   FORMAT (' (i,x(i), i=0, ilGR+1) ')
        WRITE (IDFIL_GREG,955)
     $  (i, TMPVEC(I), I=0,il+1)
```

INFORMATION ONLY

FIGURE 9.1-2
(Page 2 of 4)
SF2D1_READ_PRP - Program Used to Convert the Binary Property File to ASCII

```
C       Read y-coordinates
        READ(ISECO2) (TMPVEC(I),I=0,JL+1)
        WRITE (IDFIL_GREG,9541)
 9541   FORMAT (' (j, y(j), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,955)
     $  (j, TMPVEC(J), J=0,jl+1)

C       Read delta x's
        READ(ISECO2) (TMPVEC(I),I=0,IL+1)
        WRITE (IDFIL_GREG,9542)
 9542   FORMAT (' (i, adx(i), i=0, ilGR+1) ')
        WRITE (IDFIL_GREG,955)
     $  (i, TMPVEC(I), I=0,il+1)

C       Read delta y's
        READ(ISECO2) (TMPVEC(I),I=0,JL+1)
        WRITE (IDFIL_GREG,9543)
 9543   FORMAT (' (j, ady(j), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,955)
     $  (j, TMPVEC(J), J=0,jl+1)

c 2-D arrays

C       Read x relative offsets
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9529)
 9529   FORMAT (' ((i,j,x_relRA(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read y relative offsets
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9528)
 9528   FORMAT (' ((i,j,y_relRA(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read x hydraulic conductivies
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9531)
 9531   FORMAT (' ((i,j,aKx(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read y hydraulic conductivies
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9532)
 9532   FORMAT (' ((i,j,aKy(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read fluid compressibility
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,8532)
 8532   FORMAT (' ((i,j,alpha (i,j), i=0, ilGR+1), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read porosity
```

INFORMATION ONLY

FIGURE 9.1-2

(Page 3 of 4)

SF2D1_READ_PRP - Program Used to Convert the Binary Property File to ASCII

```
READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,8533)
 8533 FORMAT (' ((i,j,phi (i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read specific storage
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9533)
 9533 FORMAT (' ((i,j,spec_stor(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read aquifer storitivity
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9534)
 9534 FORMAT (' ((i,j,aq_stor(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read aquifer thickness
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9535)
 9535 FORMAT (' ((i,j,aq_thick(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read aquifer bottom
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9635)
 9635 FORMAT (' ((i,j,aq_bot(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read well/river recharge term
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9536)
 9536 FORMAT (' ((i,j,c10_rch(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read river conductivity
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9537)
 9537 FORMAT (' ((i,j,riv_con(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read river head
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9637)
 9637 FORMAT (' ((i,j,riv_head(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
      WRITE (IDFIL_GREG,953)
     $ ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C     Read river bottom
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9737)
 9737 FORMAT (' ((i,j,riv_bot(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
```

INFORMATION ONLY

FIGURE 9.1-2

(Page 4 of 4)

SF2D1_READ_PRP - Program Used to Convert the Binary Property File to ASCII

```
WRITE (IDFIL_GREG,953)
    $  ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read constant head
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9538)
 9538   FORMAT (' ((i,j,head(i,j), i=0, ilGR+1), j=0, jlGR+1) ')
        WRITE (IDFIL_GREG,953)
    $  ((i,j,TMPARR(I,J), I=0,il+1), J=0,jl+1)

      STOP 'READ_PRP: NORMAL COMPLETION'
      END
```

INFORMATION ONLY

FIGURE 9.1-3

(Page 1 of 3)

SF2D1_TEST_READ_1 - Program Used Read Selected Data from the Binary Property File

```
PROGRAM TEST_READ_1
C
C.....6-9-94 ELLEN S. DOMBROSKI
C.....THIS PROGRAM WAS WRITTEN FOR THE SINGLE PURPOSE OF
C.....VERIFYING BINARY OUTPUT FROM SECOFL2D.
C.....SPECIFIED RANGES OF SPECIFIED ARRAYS ARE PRINTED TO THE
C.....OUTPUT FILE FOR VERIFICATION BY INSPECTION.
C.....
C     Reads binary property file (FOR020.DAT) written by SET_GREG2 and
C     translates it into an ASCII output file.
C     Prompts user for the input file name (i.e. FOR020.DAT) and the
C     user's choice of output file names.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      dimension TMPARR(0:100,0:100)
      CHARACTER*50 INFILE, OUTFILE
      DATA ISECO2/10/
      DATA IDFIL_GREG/20/

C<><><><><><><><><><><><><><><>
C...BEGIN PROCEDURES...
C<><><><><><><><><><><><><><><>

      SCE = 1.D-9

      WRITE(6,*) 'Enter the name of the input file:'
      READ(5,'(A)') INFILE
      OPEN(UNIT=ISECO2,FILE=INFILE,STATUS='OLD',READONLY,
     +     FORM='UNFORMATTED')

      WRITE(6,*) 'Enter the name of the output file:'
      READ(5,'(A)') OUTFILE
      OPEN(UNIT=IDFIL_GREG,FILE=OUTFILE,STATUS='NEW')

      WRITE (IDFIL_GREG,9520) INFILE
 9520 FORMAT (' ASCII OUTPUT FROM FILE ',A/
     $ ' THIS FILE CONTAINS SPECIFIC RANGES THAT WILL BE USED',
     $ ' FOR VERIFICATION.  6-9-94.')

C  953   FORMAT (3(2I6, 1PE13.5))
  953   FORMAT (2I6, 1PE15.5)

C.....READ(ISECO2) IL, JL, RDUM1, IDUM1, IDUM2, RDUM2
      READ(ISECO2) IL, JL
C.....READ(ISECO2) RDUM1, RDUM2, RDUM3
      READ(ISECO2) RDUM1, RDUM2, RDUM3

C.....1-D arrays
C.....Read x-coordinates
      READ(ISECO2)
C.....Read y-coordinates
      READ(ISECO2)
C.....Read delta x's
      READ(ISECO2)
C.....Read delta y's
      READ(ISECO2)
C.....2-D arrays
C.....Read x relative offsets
```

INFORMATION ONLY

FIGURE 9.1-3

(Page 2 of 3)

SF2D1_TEST_READ_1 - Program Used Read Selected Data from the Binary Property File

```
READ(ISECO2)
C.....Read y relative offsets
      READ(ISECO2)
C.....
C.....READ AND ECHO THESE FOR TEST. 6-9-94
C.....Read x hydraulic conductivies
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,100)
  100 FORMAT (' ((i,j,aKx(i,j), i= 1,10), j= 1, 1)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=1,10), J=1,1)
C.....
      WRITE (IDFIL_GREG,102)
  102 FORMAT (' ((i,j,aKx(i,j), i=21, 30), j=28,28)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=21,30), J=28,28)
C.....
      WRITE (IDFIL_GREG,104)
  104 FORMAT (' ((i,j,aKx(i,j), i=41, 50), j=57,57)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=41,50), J=57,57)
C.....
C.....READ AND ECHO THESE FOR TEST. 6-9-94
C.....Read y hydraulic conductivies
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,110)
  110 FORMAT (' ((i,j,aKy(i,j), i= 1,10), j= 1, 1)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=1,10), J=1,1)
C.....
      WRITE (IDFIL_GREG,112)
  112 FORMAT (' ((i,j,aKy(i,j), i=21, 30), j=28,28)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=21,30), J=28,28)
C.....
      WRITE (IDFIL_GREG,114)
  114 FORMAT (' ((i,j,aKy(i,j), i=41, 50), j=57,57)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=41,50), J=57,57)
C.....
C.....READ AND ECHO THESE FOR TEST. 6-9-94
C.....Read fluid compressibility
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,120)
  120 FORMAT (' ((i,j,alpha(i,j), i= 1,10), j= 1, 1)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=1,10), J=1,1)
C.....
      WRITE (IDFIL_GREG,122)
  122 FORMAT (' ((i,j,alpha(i,j), i=21, 30), j=28,28)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=21,30), J=28,28)
C.....
      WRITE (IDFIL_GREG,124)
  124 FORMAT (' ((i,j,alpha(i,j), i=41, 50), j=57,57)')
      WRITE (IDFIL_GREG,953)
```

INFORMATION ONLY

FIGURE 9.1-3

(Page 3 of 3)

SF2D1_TEST_READ_1 - Program Used Read Selected Data from the Binary Property File

```
$((i,j,TMPARR(I,J)*SCF, I=41,50), J=57,57)
C.....
C.....READ AND ECHO THESE FOR TEST. 6-9-94
C.....Read porosity
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,130)
  130 FORMAT (' ((i,j,phi(i,j), i= 1,10), j= 1, 1)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J), I=1,10), J=1,1)
C.....
      WRITE (IDFIL_GREG,132)
  132 FORMAT (' ((i,j,phi(i,j), i=21, 30), j=28,28)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J), I=21,30), J=28,28)
C.....
      WRITE (IDFIL_GREG,134)
  134 FORMAT (' ((i,j,phi(i,j), i=41, 50), j=57,57)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J), I=41,50), J=57,57)
C.....
C.....Read specific storage
C     READ(ISECO2)
C.....Read aquifer storitivity
C     READ(ISECO2)
C.....Read aquifer thickness
C     READ(ISECO2)
C.....Read aquifer bottom
C     READ(ISECO2)
C.....Read well/river recharge term
C     READ(ISECO2)
C.....Read river conductivity
C     READ(ISECO2)
C.....Read river head
C     READ(ISECO2)
C.....Read river bottom
C     READ(ISECO2)
C.....Read constant head
C     READ(ISECO2)
C.....
      STOP 'TEST_READ_1: NORMAL COMPLETION'
      END
```

INFORMATION ONLY

9.1.4 Acceptance Criteria

The acceptance criteria for the regression tesing is stated in Section 10 below.

For requirement R.2: The values for porosity and bulk compressibility remain constant. The values for hydraulic conductivity from LOCAL_TEST.ASC (interpolated from the regional grid) will agree with the hand-calculation to 6 decimal places.

	From file LOCAL_TEST.ASC		From hand-calculation
Element 1	(12,15)	1.00219E+03	1002.19
Element 2	(20,47)	2.23585E+02	233.585
Element 3	(35,25)	9.01917E+02	901.917
Element 4	(40,40)	1.26109E+02	126.109

10.0 INSTALLATION AND REGRESSION TESTING

Test Case 1 is suitable for installation and regression testing. For regression testing against the primitive baseline on file in the SWCF, after the command file to run this test case is executed, run the program SF2D1_TEST_READ_1 (compiled and linked from SF2D1_TEST_READ_1.FOR). The FORTRAN file is shown in **Figure 9.1-3**. The user will be prompted for input and output file names. The input file is SF2D1_REGION_TEST.PRP. The output file name is SF2D1_TEST_READ.OUT. This program was previously used to produce the output on pages 4-6 of the verification section of the primitive baseline using the regional property file created during that verification process. Copies of these pages are included as Appendix B. This program should produce the identical results to the previous results in the primitive baseline when run during this testing exercise. The results from the current execution in the ASCII file SF2D1_TEST_READ.OUT can be printed.

INFORMATION ONLY

11.0 REFERENCES

1) WIPP-PA - Qualification Guide for Pre-existing Software, Version 2.0, dated 8/24/95.

2) WIPP-PA - Implementation Document for PRESECOFL2D Version 4.02ZO, WPO# 23296

3) WIPP-PA - Validation Document for PRESECOFL2D Version 4.02ZO WPO# 23319

4) Rechard, R.P., ed. 1992, User's Manual for CAMCON: Compliance Assessment Methodology Controller Version 3.0. SAND 90-1983, Albuquerque, NM: Sandia National Laboratories.

5) WIPP-PA - User's Manual for PRESECOFL2D Version 4.02ZO, WPO# 23297

INFORMATION ONLY

12.0 APPENDICES

APPENDIX A - REVIEW COMMENTS

See attached

INFORMATION ONLY

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Requirements Document Reviewer's Form

Form Number: 458 Effective: 7/31/95

Procedure: 19-1 Revision: 1 Page 1 of 1

WPO Number: 23318 Software Classification: SNL-SW

Reviewer Instructions:

Check "Yes for each item reviewed and found acceptable.

Check "No" for each item which requires further work.

Check "N/A" for items which are not applicable.

Check "N/R" for items not reviewed (only if there are muiltiple reviewers).

For multiple reviewers, each reviewer shall complete an RD review form.

Prior to sign-off of the RD, all "No" items shall be appropriately addressed by the code sponsor so that "Yes" or "N/A" may be checked, or a memo from the DM shall be attached to the RD explaining the reason for the nonconformance.

This form shall be included as part of the baseline RD.

	Yes	No	N/A	N/R
1. **Functionality** Are the functions that the software is to perform adequately identified?	☒	☐	☐	☐
2. **Performance** Are the time-related issues of software operations such as speed, recovery time, or response time identified?	☐	☐	☒	☐
3. **Design Constraints** Are elements that will restrict design options identified?	☐	☐	☒	☐
4. **Attributes** Are non-time related issues of software operation such as portability acceptance criteria, access criteria, and maintainablility identified?	☐	☐	☒	☐
5. **External Interfaces** Are required interactions with people, hardware, and other software identified?	☒	☐	☐	☐
6. **Completeness** Are the requirements complete?	☒	☐	☐	☐
7. **Verifiability** Can meeting the requirements be verified?	☒	☐	☐	☐
8. **Consistency** Are the requirements consistent with each other?	☒	☐	☐	☐
9. **Technical Feasibility** Are the requirements technically feasible?	☒	☐	☐	☐

INFORMATION ONLY

C. David Updegraff — Reviewer Name (printed)

C. David Updegraff — Signature

11 Oct 95 — Date

SWCF File Code: 1.1.6.9

WBS#

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Verification and Validation Plan Reviewer's Form

Form Number: 460 Effective: 7/31/95

Procedure: 19-1 Revision: 1 Page 1 of 1

WPO Number: 23318 Software Classification: SNL-SW

Reviewer Instructions:

Check "Yes for each item reviewed and found acceptable.

Check "No" for each item which requires further work.

Check "N/A" for items which are not applicable.

Check "N/R" for items not reviewed (only if there are multiple reviewers).

For multiple reviewers, each reviewer shall complete a VVP Review form.

Prior to sign-off of the VVP, all "No" items shall be appropriately addressed by the code sponsor so that "Yes" or "N/A" may be checked, or a memo from the DM shall be attached to the VVP explaining the reason for the nonconformance.

This form shall be included as part of the baseline VVP.

Item	Yes	No	N/A	N/R
1. **Sufficient Test Cases** Does the VVP identify sufficient test cases and acceptance criteria to ensure the final software end-product satisifies the requirements of the RD? *(Check "N/A" if peer review fulfills the validation requirements)*	☒	☐	☐	☐
2. **Adequacy of Test Cases** Do the test cases demonstrate that the code adequately performs all intended functions and produces valid results for problems encompassing the range of permitted usage as defined by the User's Manual? *(See Section 4.3.3 for User's Manual description)*	☒	☐	☐	☐
3. **Operational Control** If the software is used for operational control, do tests demonstrate required performance over the range of operation of the controlled function or process?	☐	☐	☒	☐
4. **Unintended Functions** Do the test cases show that the code does not perform any unintended function that either by itself or in combination with other functions can degrade the intended outcomes of the software?	☐	☐	☒	☐
5. **Test Case Comparisons** Are the test case results compared to one of the following? - hand calculations; - calculations using comparable provern problems - empirical data and information from confirmed published data - correlation and/or technical literature; or - results from other validated software of similar purpose. *(Enter N/A if peer review is used to fulfill this function)*	☒	☐	☐	☐
6. **Peer Review** If peer review will be required to validate the software, is it identified in the VVP?	~~☒ Yes~~ (CDU)	☐	☒	☐
7. **Installation and Regression Testing** Are test cases which are suitable for installation testing and regression testing identified from the set of verification and validation test cases?	☒	☐	☐	☐

INFORMATION ONLY

C. David Updegraff — Reviewer's Name *(printed)*

C. David Updegraff — Signature

11 Oct 95 — Date

SWCF File Code: 1.1.6.9

WBS#

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Review Comment (Software)

Form Number: 299 — Effective: 7/31/95

Procedure: QAP 19-1 — Revision: 1 — Page 1 of 2

File Code: SWCF- 4 — 1.1.6.9 (WBS #(s)) — PA :SFT (Alpha Code)

Keywords: *(Used for unique identification)*

PRESECOFL2D, SF201

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECOFL2D	4.02	5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

RD/VVP, pg 7, Section 5.0

The RD/VVP for SECOFL2D listed many capabilities for SECOFL2D that were not tested. If PRESECOFL2D has the ability to handle these capabilities and they are not listed in Section 5.0, they should be.

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☒ Agree ☐ Disagree

Most of the functions not tested in SECOFL2D are controlled by switches in the input file. The switch controlls the setting of a variable. The default setting is against the function so they are really transparent to the user

SWCF File Code: 1.1.6.9 (WBS#)

INFORMATION ONLY

WASTE ISOLATION PILOT PLANT
Sandia National Laboratories

Review Comment (Software)

Form Number: 299 Effective: 7/31/95

Procedure: QAP 19-1 Revision: 1 Page 1 of 2

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA :SFT (Alpha Code)

Keywords: *(Used for unique identification)*
PRESECOFL2D, SF2D1

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECOFL2D	4.02	5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

RD/VVP, page 6, line 6
The output file "secofl_loc.prp" line is missing some information. It contains "material property information for the local grid." In addition, it is also used as an input file to SECOFL2D and POSTSECOFL2D

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☒ Agree ☐ Disagree

Text now added

SWCF File Code: 1.1.6.9 (WBS#)

INFORMATION ONLY

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Review Comment (Software)

Form Number: 299 Effective: 7/31/95

Procedure: QAP 19-1 Revision: 1

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA :SFT (Alpha Code)

Keywords: *(Used for unique identification)*
PRESECOFL2D, SF201

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECOFL2D	4.02	5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

RD/VVP, page 5, Section 2.4
Is requirement R.6 needed? It appears to be the same as R.1.

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☐ Agree ☒ Disagree

I believe it belongs in both categories. I don't think the categories are mutually exclusive. The requirement is both functional and an external interface requirement.

INFORMATION ONLY

WASTE ISOLATION PILOT PLANT
Sandia National Laboratories

Review Comment (Software)

Form Number: 299 | Effective: 7/31/95

Procedure: QAP 19-1 | Revision: 1 | Page 1 of 2

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA :SFT (Alpha Code)

Keywords: *(Used for unique identification)*
PRESECOFL2D, SF2D1

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECOFL2D	4.02	5/20/97

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

RD/VVP, page 10, Section 10.0
PRESECOFL2D produces 4 output files. Only part of the output file, SF2D1_REGION_TEST.PRP, is compared to the original ~~file~~ baseline output file. The remaining files, excluding the debug file, should also be compared to make sure they contain expected results.

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☐ Agree ☒ Disagree

The only file used in the baseline verification is the regional property file. Requirement R.2 and its verification, verify that the local property file is correct. The two ascii files were not verified in the baseline in such a way to make them useful for regression testing. Words have been added to make this clearer.

SWCF File Code: 1.1.6.9 WBS#

INFORMATION ONLY

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Review Comment (Software)

Form Number: 299 | Effective: 7/31/95

Procedure: QAP 19-1 | Revision: 1 |

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA :SFT (Alpha Code)

Keywords: (Used for unique identification)

PRESECO FL2D, SP201

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECO FL2D	4.02 ~~ZO~~	5/20/94

2. Type of Comment: (check one) [X] Required [ ] Suggested [ ] Error

Software Type: (check one) [ ] Commercial [X] Developed [ ] System

4. Comment: (check one, attach pages as needed) [ ] Technical [X] Document

RD/VVP, page 13:
The references to Figure #'s on This page should be moved to The last paragraph on page 10.

INFORMATION ONLY

SECTION II

5. Comment Resolution: (check one; attach pages as needed; describe impact to previous uses of code, if applicable) [ ] Agree [X] Disagree

It is better to reference these input files in the Input File section, as that is what it is there for.

APPENDIX B - PRIMITIVE BASELINE DOCUMENTS

The following pages are from the verification section primitive baseline documentation currently stored in SWCF as a part of the prior QA efforts. The results of the current test will be compared to these numbers with the numbers matching exactly.

FigureB-1

(Page 1 of 3)

Pages 4-6 of the Verification Section of the Primitive Baseline


```
ASCII OUTPUT FROM FILE nfs1:: [camcon.presecofl2d.test]regdat_test.inp
THIS FILE- CONTAINS SPECIFIC RANGES THAT WILL BE USED FOR VERIFICATION. 6-9-94.
((i, j, aKx (i, j) , i= 1, 10) , j= 1, 1)
  1     1    1.89500E-04
  2     1    2.04700E-04
  3     1    3.45600E-05
  4     1    2.67800E-05
  5     1    2.38800E-05
  1     1    1.87400E-05
  7     1    2.04300E-05
  8     1    2.96700E-05
  9     1    1.68900E-05
 10     1    7.59600E-06
((i,j,aKx (i,j), i=21, 30), j=28,28)
 21    28    1.50900E-07
 22    28    1.32000E-07
 23    28    1.42400E-07
 24    28    1.63900E-07
 25    28    4.32500E-07
 26    28    1.86500E-06
 27    28    1.87000E-06
 28    28    1.15800E-06
 29    28    1.30800E-06
 30    28    1.49000E-06
((i,j,aKx (i, j), i=41, 50), j=57, 57)
 41    57    4.20400E-10
 42    57    1.85600E-10
 43    57    2.02000E-10
 44    57    2.59200E-10
 45    57    2.66900E-10
 46    57    1.62500E-10
 47    57    2.1580CE-10
 48    57    2.51900E-11
 49    57    1.78000E-10
 50    57    2.50900E-09
((i,j,aKy(i, j) , i=1, 10), j=1, 1)
  1     1    1.89500E-04
  2     1    2.04700E-04
  3     1    3.45600-=-05
  4     1    2.67800F--05
  5     1    2.38800E-05
  6     1    1.87400E-05
  7     1    2.04300E-05
  8     1    2.96700E-05
  9     1    1.68900E-05
 10     1    7.59600E-06
((i,j,aKy (i,j)i=21 , 30) ,j=28, 28)
 21    28    1.50900E-07
 22    28    1.32000E-07
 23    28    1.42400E-07
```

INFORMATION ONLY

FigureB-1

(Page 2 of 3)

Pages 4-6 of the Verification Section of the Primitive Baseline


```
 24     28    1.63900E-07
 25     28    4.32500E-07
 26     28    1.86500E-06
 27     28    1.87000E-06
 28     28    1.15800E-06
 29     28    1.30800E-06
 30     28    1.49000E-06
(i ,jaKy (i, j), i=4 1, 5 0) , j=57, 5 7)
 41     57    4.20400E-10
 42     57    1.85600E-10
 43     57    2.02000E-10
 44     57    2.59200E-10
 45     57    2.66900E-10
 46     57    1.62500E-10
 47     57    2.15800E-10
 48     57    2.51900E-10
 49     57    1.78000E-10
 50     57    2.50900E-10
((i,j alpha (I, j), i=1,10), j=1, 1)
  1      1    7.57000E-10
  2      1    7.57000E-10
  3      1    7.57000E-10
  4      1    7.57000E-10
  5      1    7.57000E-10
  6      1    7.57000E-10
  7      1    7.57000E-10
  8      1    7.17000E-10
  9      1    7.57000E-10
 10      1    7.57000E-10
(i, j, alpha (i, j) , i=21, 30), j=28, 28)
 21     28    7.57000E-10
 22     28    7.57000E-10
 23     28    7.57000E-10
 24     28    7.57000E-10
 25     28    7.57000E-10
 26     28    7.57000E-10
 27     28    7.57000E-10
 28     23    7.57000E-10
 29     28    7.57000E-10
 30     28    7.57000E-10
(i, j, alpha (i, j) , i=41, 50), j=57, 57)
 41     57    7.57000E-10
 42     57    7.57000E-10
 43     57    7.57000E-10
 44     57    7.57000E-10
 45     57    7.57000E-10
 46     57    7.57000E-10
 47     57    7.57000E-10
 48     57    7.57000E-10
 49     57    7.57000E-10
 50     57    7.57000E-10
((i, j, phi (i, j) , i=1, 10), j=1, 1)
  1      1    1.45420E-01
  2      1    1.45420E-01-
  3      1    1.45420E-01
  4      1    1.45420E-01
  5      1    1.45420E-01
  6      1    1.45420E-01
```

INFORMATION ONLY

FigureB-1

(Page 3 of 3)

Pages 4-6 of the Verification Section of the Primitive Baseline


```
  7      1     1.45420E-01
  8      1     1.45420E-01
  9      1     1.45420E-01
 10      1     1.45420E-01
((i, j, phi (i, j) , i=21, 30), j=28, 28)
 21     28     1.45420E-01
 22     28     1.45420E-01
 23     28     1.45420E-01
 24     28     1.45420E-01
 25     28     1.45420E-01
 26     28     1.45420E-01
 27     28     1.45420E-01
 28     28     1.45420E-01
 29     28     1.45420E-01
 30     23     1.45420E-01
((i, j, phi (i, j) , i=41, 50), j=57,57)
 41     57     1.45420E-01
 42     57     1.45420E-01
 43     57     1.45420E-01
 44     57     1.45420E-01
 45     57     1.45420E-01
 46     57     1.45420E-01
 47     57     1.45420E-01
 48     57     1.45420E-01
 49     57     1.45420E-01
 50     57     1.45420E-01
```

INFORMATION ONLY

APPENDIX B - PRIMITIVE BASELINE DOCUMENTS

The following pages are from the verification section primitive baseline documentation currently stored in SWCF as a part of the prior QA efforts. The results of the current test will be compared to these numbers with the numbers matching exactly.

Figure B-1

(Page 1 of 3)

Pages 4-6 of the Verification Section of the Primitive Baseline


```
ASCII OUTPUT FROM FILE nfs1:: [camcon.presecofl2d.test]regdat_test.inp
THIS FILE CONTAINS SPECIFIC RANGES THAT WILL BE USED FOR VERIFICATION. 6-9-94.
((i, j, aKx (i, j) , i= 1, 10) , j= 1, 1)
  1      1    1.89500E-04
  2      1    2.04700E-04
  3      1    3.45600E-05
  4      1    2.67800E-05
  5      1    2.38800E-05
  1      1    1.87400E-05
  7      1    2.04300E-05
  8      1    2.96700E-05
  9      1    1.68900E-05
 10      1    7.59600E-06
((i,j,aKx (i,j), i=21, 30), j=28,28)
 21     28    1.50900E-07
 22     28    1.32000E-07
 23     28    1.42400E-07
 24     28    1.63900E-07
 25     28    4.32500E-07
 26     28    1.86500E-06
 27     28    1.87000E-06
 28     28    1.15800E-06
 29     28    1.30800E-06
 30     28    1.49000E-06
((i,j,aKx (i, j), i=41, 50), j=57, 57)
 41     57    4.20400E-10
 42     57    1.85600E-10
 43     57    2.02000E-10
 44     57    2.59200E-10
 45     57    2.66900E-10
 46     57    1.62500E-10
 47     57    2.15800E-10
 48     57    2.51900E-11
 49     57    1.78000E-10
 50     57    2.50900E-09
((i,j,aKy(i, j) , i=1, 10), j=1, 1)
  1      1    1.89500E-04
  2      1    2.04700E-04
  3      1    3.45600-=-05
  4      1    2.67800F--05
  5      1    2.38800E-05
  6      1    1.87400E-05
  7      1    2.04300E-05
  8      1    2.96700E-05
  9      1    1.68900E-05
 10      1    7.59600E-06
((i,j,aKy (i,j)i=21 , 30) ,j=28, 28)
 21     28    1.50900E-07
 22     28    1.32000E-07
 23     28    1.42400E-07
```

INFORMATION ONLY

Figure B-1

(Page 2 of 3)

Pages 4-6 of the Verification Section of the Primitive Baseline


```
 24     28    1.63900E-07
 25     28    4.32500E-07
 26     28    1.86500E-06
 27     28    1.87000E-06
 28     28    1.15800E-06
 29     28    1.30800E-06
 30     28    1.49000E-06
(i ,jaKy (i, j), i=4 1, 5 0) , j=57, 5 7)
 41     57    4.20400E-10
 42     57    1.85600E-10
 43     57    2.02000E-10
 44     57    2.59200E-10
 45     57    2.66900E-10
 46     57    1.62500E-10
 47     57    2.15800E-10
 48     57    2.51900E-10
 49     57    1.78000E-10
 50     57    2.50900E-10
 ((i,j alpha (I, j), i=1,10), j=1, 1)
  1      1    7.57000E-10
  2      1    7.57000E-10
  3      1    7.57000E-10
  4      1    7.57000E-10
  5      1    7.57000E-10
  6      1    7.57000E-10
  7      1    7.57000E-10
  8      1    7.17000E-10
  9      1    7.57000E-10
 10      1    7.57000E-10
 (i, j, alpha (i, j) , i=21, 30), j=28, 28)
 21     28    7.57000E-10
 22     28    7.57000E-10
 23     28    7.57000E-10
 24     28    7.57000E-10
 25     28    7.57000E-10
 26     28    7.57000E-10
 27     28    7.57000E-10
 28     23    7.57000E-10
 29     28    7.57000E-10
 30     28    7.57000E-10
 (i, j, alpha (i, j) , i=41, 50), j=57, 57)
 41     57    7.57000E-10
 42     57    7.57000E-10
 43     57    7.57000E-10
 44     57    7.57000E-10
 45     57    7.57000E-10
 46     57    7.57000E-10
 47     57    7.57000E-10
 48     57    7.57000E-10
 49     57    7.57000E-10
 50     57    7.57000E-10
 ((i, j, phi (i, j) , i=1, 10), j=1, 1)
  1      1    1.45420E-01
  2      1    1.45420E-01
  3      1    1.45420E-01
  4      1    1.45420E-01
  5      1    1.45420E-01
  6      1    1.45420E-01
```

INFORMATION ONLY

Figure B-1
(Page 3 of 3)
Pages 4-6 of the Verification Section of the Primitive Baseline


```
 7     1    1.45420E-01
 8     1    1.45420E-01
 9     1    1.45420E-01
10     1    1.45420E-01
((i, j, phi (i, j) , i=21, 30), j=28, 28)
21    28    1.45420E-01
22    28    1.45420E-01
23    28    1.45420E-01
24    28    1.45420E-01
25    28    1.45420E-01
26    28    1.45420E-01
27    28    1.45420E-01
28    28    1.45420E-01
29    28    1.45420E-01
30    23    1.45420E-01
((i, j, phi (i, j) , i=41, 50), j=57,57)
41    57    1.45420E-01
42    57    1.45420E-01
43    57    1.45420E-01
44    57    1.45420E-01
45    57    1.45420E-01
46    57    1.45420E-01
47    57    1.45420E-01
48    57    1.45420E-01
49    57    1.45420E-01
50    57    1.45420E-01
```

INFORMATION ONLY

WIPP PA

VALIDATION DOCUMENT

for

PRESECOFL2D

Version 4.02ZO

WPO# 23319

INFORMATION ONLY

TABLE OF CONTENTS

Page

1.0 INTRODUCTION 5
1.1 Software Identifier 5
1.2 Points of Contact 5

2.0 TESTING ENVIRONMENT 5

3.0 TEST TOOLS 5

4.0 STATIC TESTING 6
4.1 Flint Analysis: Procedure and Results 6
4.2 SCA Analysis: Procedure and Results 12

5.0 COVERAGE ANALYSIS 33

6.0 FUNCTIONAL TESTING 53
6.1 Test Case #1: Regression Testing and Coverage Test Case 56
6.1.1 Test Objective 56
6.1.2 Test Procedure 56
6.1.3 Input/Output Files 59
6.1.4 Evaluation 60

7.0 RELATED TESTING DOCUMENTATION 76

8.0 CONCLUSION 77

9.0 REFERENCES 77

10.0 APPENDICES 78

APPENDIX A - REVIEWER'S FORMS

APPENDIX B - PRIMITIVE BASELINE DOCUMENT

INFORMATION ONLY

LIST OF TABLES

Page

3-1 Software Testing Tools 5

4-1 Files Created by the SCA Code Analyzer 12

5-1 Location of Coverage Analysis Output Files 33

7-1 Documentation Related to the Testing of PRESECOFL2D 80

INFORMATION ONLY

LIST OF FIGURES

Page

4.1-1 FLINT Output for Code Modules 7

4.2-1 SF2D1_BUILD.COM Build Script 13

4.2-2 SF2D1.MMS Input File for Build Script 22

4.2-3 SCA Output File of Modules Not Referenced 28

4.2-4 SCA Call Tree Output 29

4.2-5 SCA Output File of All Modules Included in Static Analysis 32

5-1 Script Used to Run Test Cases for PCA Analysis 34

5-2 PCA Log of Coverage Analysis 36

5-3 PCA Coverage Analysis for Regression Testing and Coverage Test Case 38

5-4 Cumulative PCA Coverage Analysis for Test Case 52

5-5 Code Sponsor's Response to Coverage Analysis 56

6-1 Functional Requirements for PRESECOFL2D 54

6-2 Script Used for Running Test Cases for Functional Analysis 55

6.1-1 Listing of READ_PRP.FOR 61

6.1-2 Listing of Sample of Alpha Values from LOCAL_TEST.ASC 65

6.1-3 Listing of Sample of Phi Values from LOCAL_TEST.ASC 66

6.1-4 Listing of Four Rows of AKX Values from LOCAL_TEST.ASC 67

6.1-5 Listing of SF2D1_TEST_READ_1.FOR 69

6.1-6 Listing of SF2D1_TEST_READ.OUT 72

6.1-7 Listing of SF2D1_PRESECOFL_TEST.INP 75

B.1 Pages 4-6 of the Verification Section of the Primitive Baseline

INFORMATION ONLY

1.0 INTRODUCTION

The purpose of this document is to summarize the results of the testing activities prescribed in the RD/VVP and to provide evaluations based on those results. The PRESECOFL2D code has previously undergone a QA process complete with a review and is currently qualified as Level A under QAP-1 Rev. F. Both static testing and coverage testing was carried out in the same manner as with all other codes, but the functional testing consisted primarily of regression testing.

1.1 Software Identifier

PRESECOFL2D version 4.02ZO (WIPP PA Code Prefix: SF2D1)

1.2 Points of Contact

Code Sponsor: Rebecca L.Blaine
Code Consultant: Rebecca L. Blaine

Ecodynamics Research Associates, Inc.
P.O. Box 9229
Albuquerque, NM 87119
(505) 843-7445

2.0 TESTING ENVIRONMENT

Testing for PRESECOFL2D was performed in the following computer environment:

Hardware platform: DEC Alpha
Operating System: OpenVMS version 6.1
Directory: WPSTESTROOT:[SF2D1]

3.0 TEST TOOLS

The testing tools which were used as part of the software validation are described in **Table 3-1.**

Table 3-1
Software Testing Tools

Tool Name	Usage
FORTRAN-Lint (FLINT)	Source Code Analyzer used to identify non ANSI standard coding.
DECset SCA	Source Code analyzer used to identify uncalled modules, to identify unreferenced variables, and to display the calling tree.
DECset PCA	Performance Coverage Analyzer, used to identify any unexercised modules
DEC FORTRAN Compiler version EV6.2-508-274	Compiler, used to create the executables and identify compilation errors and warnings.

INFORMATION ONLY

4.0 STATIC TESTING

For PRESECOFL2D, static analysis was performed using both the source code analyzer FLINT and the source code analyzer DECset-SCA . FLINT was used to detect non-ANSI standard FORTRAN coding. DECset-SCA was used to reveal any uncalled modules, to reveal any unreferenced variables, and to display the calling tree.

The DECset SCA analysis was performed on directory WP$TESTROOT: [SF2D1.BUILD] during the generation of the PCA executable. The FLINT analysis was carried out in the WP$TESTROOT:[SF2D1.SRC] directory.

4.1 Flint Analysis: Procedure And Results

PROCEDURE: Since PRESECOFL2D is a multimodule program, FLINT was used on a file created by appending all the modules together in alphabetical order. The file resulting from running FLINT, SF2D1_FLINT.OUT, is shown in **Figure 4.1-1**.

RESULTS: As a result of running FLINT, two types of issues were noted. First, it was noted that several variables were defined but not referenced. This was as expected by the code sponsor.

Second, it was noted that some lines in the code were over the ANSI standard 72 columns in length. It was found to be a result of padding with blanks and is of no concern with the platform and compiler with which the WIPP PA will be performed.

INFORMATION ONLY

Figure 4.1-1
(Page 1 of 5)
FLINT Output for Code Modules

Comments by Judy Rollstin:

This file, SF2D1_FLINT.OUT, is the output files resulting from using FLINT to analyze a single file created by appending all modules in the PRESECOFL2D computer code. In this way, each module was checked for adherence to ANSI standard FORTRAN.

```
**************************************************************************
FORTRAN-lint    Rev 3.01                    30-Aug-95  11:56:04   Page 1

Default options:  WARNINGS  USAGE  SUPPRESS=[illegible],276,76,261  @TREE  @XREF
Command options:  OUTPUT=sf2d1_flint.out

FORTRAN-lint     Source Analysis                                   Page 2

[illegible] WPSTESTR[illegible]:[illegible]SF2D1.DR

SF2D1.FOR;2

**************************************************************************
   Subroutine ADDMEM                    File SF2D1.FOR           Line 2

SF2D1.FOR:ADDMEM line 50.51  (SF2D1_SECO2R.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:ADDMEM line 48.63  (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

**************************************************************************
   Subroutine DELTAS                    File SF2D1.FOR           Line 314

SF2D1.FOR:DELTAS line 344.63  (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

**************************************************************************
   Subroutine GETEND                    File SF2D1.FOR           Line 445

SF2D1.FOR:GETEND line 488.51  (SF2D1_SECO2R.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

**************************************************************************
Subroutine GETLTS                     File SF2D1.FOR           Line 595

SF2D1.FOR:GETLTS line 642.63  (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.
**************************************************************************
   Subroutine GETRTS                    File SF2D1.FOR           Line 736

SF2D1.FOR:GETRTS line 782.51  (SF2D1_SECO2R.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:GETRTS line 783.63  (SF2D1_LOGIC.INC;2):

SYNTAX WARNING #531- line length exceeds 72 columns.

**************************************************************************
   Subroutine GETRUN                    File SF2D1.FOR           Line 877

SF2D1.FOR:GETRUN line 975.63  (SF2D1_LOGIC.INC;2):
```

INFORMATION ONLY

Figure 4.1-1
(Page 2 of 5)
FLINT Output for Code Modules

```
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:SETRUN line [illegible].51 (SF2D1_SECTOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
   Subroutine [illegible]                    File SF2D1.FOR          Line [illegible]

SF2D1.FOR:[illegible] line [illegible].51 (SF2D1_SECTOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
FORTRAN-lint     Source Analysis                                      Page 3

*****************************************************************************
   Subroutine INISEC                         File SF2D1.FOR          Line 1517

SF2D1.FOR:INISEC line [illegible].63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:INISEC line 1617.51 (SF2D1_SECTOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
   Subroutine PRAQPR                         File SF2D1.FOR          Line [illegible]

SF2D1.FOR:PRAQPR line [illegible].63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
   Subroutine PREPRO                         File SF2D1.FOR          Line [illegible]

SF2D1.FOR:PREPRO line [illegible].63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
     Program PRESECOFL2D                     File SF2D1.FOR          Line [illegible]

SF2D1.FOR:PRESECOFL2D line 2404.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
   Subroutine PROCDB                         File SF2D1.FOR          Line 2488

SF2D1.FOR:PROCDB line 2639.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
   Subroutine PRSECO                         File SF2D1.FOR          Line 3178

SF2D1.FOR:PRSECO line 3343.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
   Subroutine PRSCRC                         File SF2D1.FOR          Line 3536

SF2D1.FOR:PRSCRC line 3566.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

*****************************************************************************
   Subroutine RDAQPR                         File SF2D1.FOR          Line 3615
```

INFORMATION ONLY

Figure 4.1-1

(Page 3 of 5)

FLINT Output for Code Modules

```
SF2D1.FOR:RDATRR line 3665.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
  Subroutine RDBHL[illegible]                File SF2D1.FOR              Line [illegible]

SF2D1.FOR:RDBHL[illegible] line [illegible] (SF2D1_SECOOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
FORTRAN-Lint    Source Analysis                                      Page 4

****************************************************************************
  Subroutine RDCNST                  File SF2D1.FOR              Line 4047

SF2D1.FOR:RDCNST line 4089.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
  Subroutine RDINIT                  File SF2D1.FOR              Line 44[illegible]

SF2D1.FOR:RDINIT line 4457.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:RDINIT line 4458.51 (SF2D1_SECOOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
  Subroutine RDRCUF                  File SF2D1.FOR              Line 49[illegible]

SF2D1.FOR:RDRCUF line 4950.51 (SF2D1_SECOOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
  Subroutine RDRTIM                  File SF2D1.FOR              Line 51[illegible]

SF2D1.FOR:RDRTIM line [illegible].51 (SF2D1_SECOOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
  Subroutine RDSECO                  File SF2D1.FOR              Line 5311

SF2D1.FOR:RDSECO line 5395.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
  Subroutine RDSOLV                  File SF2D1.FOR              Line 5527

SF2D1.FOR:RDSOLV line 5584.51 (SF2D1_SECOOR.INC;2):

SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
  Subroutine RDWELL                  File SF2D1.FOR              Line 5740

SF2D1.FOR:RDWELL line 5792.51 (SF2D1_SECOOR.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:RDWELL line 5793.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

****************************************************************************
```

INFORMATION ONLY

Figure 4.1-1

(Page 4 of 5)

FLINT Output for Code Modules

```
    Subroutine [illegible]                 File SF2D1.FOR                 Line [illegible]

SF2D1.FOR:[illegible] line [illegible] (SF2D1_SECO2R.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:[illegible] line [illegible] (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

********************************************************************************
FORTRAN-lint      Source Analysis                                       Page [illegible]
********************************************************************************
    Subroutine SETCOORD                    File SF2D1.FOR                 Line [illegible]

SF2D1.FOR:SETCOORD line [illegible] (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

********************************************************************************
    Subroutine WREG_[illegible]             File SF2D1.FOR                 Line [illegible]

SF2D1.FOR:WREG_[illegible] line 6732:
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:WREG_[illegible] line 6735:
SYNTAX WARNING #531- line length exceeds 72 columns.

********************************************************************************
Subroutine WRTMSR                        File SF2D1.FOR                 Line 7494

SF2D1.FOR:WRTMSR line 7526.51 (SF2D1_SECO2R.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

********************************************************************************
    Subroutine WRTS2I                      File SF2D1.FOR                 Line [illegible]

SF2D1.FOR:WRTS2I line [illegible].63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:WRTS2I line 7606.51 (SF2D1_SECO2R.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

********************************************************************************
    Subroutine WRTS2L                      File SF2D1.FOR                 Line 7687

SF2D1.FOR:WRTS2L line 7737.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

********************************************************************************
    Subroutine WRTS2R                      File SF2D1.FOR                 Line 7816

SF2D1.FOR:WRTS2R line 7838.51 (SF2D1_SECO2R.INC;2):

SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:WRTS2R line 7881.63 (SF2D1_LOGIC.INC;2):
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:WRTS2R line 7883:
SYNTAX WARNING #531- line length exceeds 72 columns.

SF2D1.FOR:WRTS2R line 7885:
SYNTAX WARNING #531- line length exceeds 72 columns.
```

INFORMATION ONLY

Figure 4.1-1

(Page 5 of 5)

FLINT Output for Code Modules

```
********************************************************************************
   Subroutine WRT[illegible]                    File [illegible].FOR             Line [illegible]

[illegible].FOR:WRT[illegible] line [illegible] [illegible]
SYNTAX WARNING #[illegible] Line length exceeds 72 columns.

FORTRAN-Line        Source Analysis                                           Page

SFC[illegible].FOR:WRTSEC line [illegible] [illegible]
SYNTAX WARNING #[illegible] Line length exceeds 72 columns.

********************************************************************************
   Subroutine WRTWEL                            File SECO[illegible].FOR         Line [illegible]

SECO[illegible].FOR:WRTWEL line [illegible] [illegible]
SYNTAX WARNING #[illegible] Line length exceeds 72 columns.

SECO[illegible].FOR:WRTWEL line [illegible] [illegible]
SYNTAX WARNING #[illegible] Line length exceeds 72 columns.

********************************************************************************
```

INFORMATION ONLY

4.2 SCA Analysis: Procedure and Results

PROCEDURE: DECset-SCA was invoked during the build process. The build process was performed using the build script SF2D1_BUILD.COM developed by Mike Williamson. The build script is shown in **Figure 4.2-1**. The build script is self-contained and is executed in the directory containing the needed input files with the command.

@SF2D1_BUILD

During the build process, the input file SF2D1.MMS is required and is fetched from SCMS. That input file is shown in **Figure 4.2-2.**

As a result of the build process, a debug executable capable of doing PCA analysis was created and the SCA code analyzer created the three output files. These files are listed in Table 4-1 along with a brief description of each file and a reference to where a listing of each file can be found in this document.

Table 4-1
Files Created by the SCA Code Analyzer

Output File	Description	Figure
SF2D1_SCA_MOD_NOT_REF.TXT	A listing of the modules in the code that are not referenced	**Figure 4.2-3**
SF2D1_CALLTREE.TXT	A listing of the calling tree for the code	**Figure 4.2-4**
SCASEVENT.DAT	A binary file containing information about the SCA process	

Following the build process, SCA was invoked at the VMS $ prompt and the following commands were executed on the SCA$EVENT.DAT file in the WP$TESTROOT: [SF2D1.SCA] directory:

SCA> **set lib []**
SCA> **show module/all/output=sf2d1_modules.out**
SCA> **exit**

This procedure converts information in a binary file to an ASCII format file. The resulting ASCII file, SF2D1_MODULES.OUT, is a listing of the modules referenced during the SCA, including such information as the date each module was compiles, the language in which each module is written, etc. associated with each dile used. This listing is shown in **Figure 4.2-5.**

RESULTS: The SCA procedure revealed the only module not being called is the main program itself, as to be expected.

INFORMATION ONLY

Figure 4.2-1
(Page 1 of 9)
SF2D1_BUILD.COM Build Script

```
$       save_verify = f$verify()          ! This saves the user's setting.
$       Define/proc noLog sys$output ...sf2d1_build.log
$       Set verify
$       ON CONTROL_Y THEN goto TERMINATE
$       ON CONTROL_C THEN goto TERMINATE
$       ON error     THEN goto TERMINATE
$       GOTO START
$!
$!-------------------------------------------------------------------------------
$! Filename:
$!  SF2D1_BUILD.COM
$!
$! Abstract:
$!  Does the build for SF2D1 prefix (PRESECOFL2D) objects.  This file will be
$!  modified or enhanced later for full production builds.
$!  Based on input parameters the build can link in PCA, build from latest
$!  and greatest or a CMS class, and build all from sources.
$!
$!
$!  Parameters:
$!  P1 - Type of build, "T" = test, "P" = Prod, D = developers  local.
$!  p2 - CMS class to build.  Default is latest version, no class.
$!  p3 - FROM_SOURCES build  default is "S", any other is not from_sources"
$!  p4 - Build for PCA  Value PCA expected
$!  p5 - Build and load SCA, and run SCA tests.  Value SCA expected.
$!  p6 - Build debug version.  Expected valued DBG, NODBG
$!
$! Author:
$!  Mike Williamson - Digital Equipment Corporation.
$!
$! Creation Date: 3-Aug-1995
$!--
$!
$START:
$! Mainline code.  Calls subroutines that do the actual work.
$       GOSUB check_params
$       GOSUB initialize
$       GOSUB build_macro
$       GOSUB create_lis
$       GOSUB start_mail
$       GOSUB do_build
$       GOSUB load_sca
$       GOTO terminate
$       EXIT
$!END OF MAINLINE
$!
$!------------------------------------------------------------------------
$! Subroutines, most of the actual work is done in the subroutines.
$!------------------------------------------------------------------------
$BUILD_MACRO:
$! Write the mms macro file to contain requested options.
$! This file needs to be written dynamically as the options can
$! change based on parameters passed to this command procedure.
$!
$       OPEN/write mms_file wp_build:wp_gen_macros.mms
$!
$! Specify the CMS flag, specifically any class requirement.
$       IF ( p2 .nes. "" )
$       THEN
$               WRITE mms_file "cmsflags = /occlude=none/generation=''p2'"
$       ELSE
$               WRITE mms_file "cmsflags = /occlude=none"
```

INFORMATION ONLY

Figure 4.2-1

(Page 2 of 9)

SF2D1_BUILD.COM Build Script

```
$     ENDIF
$!
$! Set the compile and link flags that will be used by the mms description file.
$! For the most part the compile flags are the same as what ever the
$! code owner was using to compile the code. Noopt and debug are used
$! for the test process.
$     IF p1 .eqs. "B"
$     THEN
$        WRITE mms_file "cflags= ''dbg_cflag' ''sca_flag' ''general_cflag'"
$        WRITE mms_file "linkflags = ''dbg_lflag'"
$        WRITE mms_file "exe_name = PRESBITFL2D.exe"
$     ENDIF
$!
$     IF p1 .eqs. "T"
$     THEN
$        WRITE mms_file "cflags= ''dbg_cflag' ''sca_flag' ''general_cflag'"
$        WRITE mms_file "linkflags = ''dbg_lflag'"
$        IF p4 .eqs. "NODBG"
$        THEN
$              WRITE mms_file "exe_name = PRESECOFL2D_test_nodbg.exe"
$        ELSE
$           IF p4 .eqs. "PCA"
$           THEN
$              WRITE mms_file "exe_name = PRESECOFL2D_test_pca.exe"
$              DEFINE nolog lib$debug sys$library:pca$collector.exe
$           ELSE
$              WRITE mms_file "exe_name = PRESECPFL2D_test_dbg.exe"
$              IF F$TRNLNM("LIB$DEBUG") .NES. "" THEN DEASSIGN lib$debug
$           ENDIF
$        ENDIF
$     ENDIF
$!
$     IF p1 .eqs. "D"
$     THEN
$        WRITE mms_file "cflags= ''dbg_cflag' ''sca_flag' ''general_cflag'"
$        WRITE mms_file "linkflags = ''dbg_lflag'"
$        IF p4 .eqs. "NODBG"
$        THEN
$              WRITE mms_file "exe_name = PRESECOFL2D_local_nodbg.exe"
$        ELSE
$           IF p4 .eqs. "PCA"
$           THEN
$              WRITE mms_file "exe_name = PRESECOFL2D_local_pca.exe"
$              DEFINE nolog lib$debug sys$library:pca$collector.exe
$           ELSE
$              WRITE mms_file "exe_name = PRESECOFL2D_local_dbg.exe"
$              IF F$TRNLNM("LIB$DEBUG") .NES. "" THEN DEASSIGN lib$debug
$           ENDIF
$        ENDIF
$     ENDIF
$!
$     CLOSE mms_file
$!
$     RETURN
$!------------------------------------------------------------------------
$ CHECK_PARAMS:
$! Verify reasonable paramater settings, prompt where indicated.
$!
$     mode = f$mode()
$!
$! Uppercase any input params, they are expected to be blank or uppercase
$     IF p1 .nes. "" THEN p1 = f$edit("''p1'", "upcase")
```

INFORMATION ONLY

Figure 4.2-1
(Page 3 of 9)
SF2D1_BUILD.COM Build Script

```
$       IF p2 .nes. "" THEN p2 = f$edit( "''p2'", "upcase" )
$       IF p3 .nes. "" THEN p3 = f$edit( "''p3'", "upcase" )
$       IF p4 .nes. "" THEN p4 = f$edit( "''p4'", "upcase" )
$       IF p5 .nes. "" THEN p5 = f$edit( "''p5'", "upcase" )
$       IF p6 .nes. "" THEN p6 = f$edit( "''p6'", "upcase" )
$!
$! Check build type param
$       IF p1 .eqs. "" .and. mode .eqs. "BATCH"
$       THEN
$               WRITE SYS$OUTPUT "Must specify build type in batch"
$               EXIT
$       ELSE
$               IF p1 .eqs. ""
$               THEN
$                 INQUIRE/nopunct build_type -
$                     "Build type (Ttest, Pprod, Dlocal) [T]: "
$                 build_type = F$EDIT( "''build_type'", "upcase" )
$                 IF build_type .eqs. "" THEN build_type = "T"
$                 p1 = build_type
$               ENDIF
$       ENDIF
$!
$!
$! Check CMS class param
$       IF p2 .eqs. ""
$       THEN
$           IF p1 .eqs. "P" .and. mode .nes. "BATCH"
$           THEN
$             INQUIRE/nopunct cms_class -
$                 "Enter class name or blank for latest and greatest "
$             cms_class = F$EDIT( "''cms_class'", "upcase" )
$             IF cms_class .nes. "" then p2 = cms_class
$           ENDIF
$       ENDIF
$!
$! Are we going to force building from sources?
$       IF p3 .eqs. "" .and. mode .nes. "BATCH"
$       THEN
$           IF p1 .nes. "D"
$           THEN
$             INQUIRE/nopunct build_sources "Build all from sources? (Y/N) [N]: "
$
$             build_sources = F$EDIT( "''build_sources'", "upcase" )
$             IF build_sources .eqs. "Y" THEN p3 = "S"
$           ENDIF
$       ENDIF
$!
$       IF p3 .eqs. "S" .and. p1 .eqs. "D"
$       THEN
$           WRITE sys$output "Development builds are not allowed to be from_sources."
$           p3 = "N"
$       ENDIF
$!
$!
$! If this is a production build, then PCA can not be used as it requires
$! a debug compile and link, and prod is by defination not debug.
$       IF p1 .eqs. "P"
$       THEN
$           IF p4 .nes. ""
$           THEN
$             Write sys$output "Can not have a production build for PCA."
$           ENDIF
```

INFORMATION ONLY

Figure 4.2-1
(Page 4 of 9)
SF2D1_BUILD.COM Build Script

```
$           p4 = ""
$       ELSE
$           IF p4 .eqs. "" .and. mode .nes. "BATCH"
$           THEN
$             INQUIRE /nopunc pca_build "Build for PCA ( Y or N ) [N]: "
$              pca_build = F$EDIT ("''pca_build'", "upcase")
$             IF pca_build .eqs. "Y" then p4 = "PCA"
$           ENDIF
$       ENDIF
$!
$! Need a symbol for the MMS/ call indicating SCA condition.
$       IF p5 .eqs. "" .and. mode .nes. "BATCH"
$       THEN
$           INQUIRE /nopunc build_for_sca "Build for SCA? ( Y or N ) [N]: "
$           build_for_sca = F$EDIT ("''build_for_sca'", "upcase")
$           IF build_for_sca .eqs. "Y" THEN p5 = "SCA"
$       ENDIF
$!
$       IF p5 .eqs. "SCA"
$       THEN
$           mms_sca_flag = "/sca=sca$library"
$       ELSE
$           mms_sca_flag = "/nosca"
$       ENDIF
$!
$! P6 is used to specify debug or no debug build.
$       IF p1 .eqs. "P"                  ! Production is always nodebug.
$       THEN
$           p6 = "NODBG"
$           dbg_cflag = "/opt/nodebug"
$           dbg_lflag = "/nodebug/notraceback"
$       ENDIF
$!
$       IF p4 .eqs. "PCA"             ! PCA builds are always debug.
$       THEN
$           p6 = "DBG"
$           dbg_cflag = "/noopt/debug"
$           dbg_lflag = "/debug/traceback"
$       ENDIF
$!
$       IF p1 .nes. "P"               ! Test and development can be debug.
$       THEN
$           dbg_flag :== ""
$           IF p6 .eqs. "" .and. mode .nes. "BATCH"
$           THEN
$             INQUIRE/nopunc dbg_flag "Build debug ( Y or N ) [Y]: "
$             dbg_flag = f$edit ("''dbg_flag'", "upcase")
$           ENDIF
$!
$           IF dbg_flag .eqs. "N" .or. p6 .eqs. "NODBG"
$           THEN
$             p6 = "NODBG"
$             dbg_cflag = "/opt/nodebug"
$             dbg_lflag = "/nodebug/notraceback"
$           ELSE
$             p6 = "DBG"
$             dbg_cflag = "/noopt/debug"
$             dbg_lflag = "/debug/traceback"
$           ENDIF
$       ENDIF
$!
```

INFORMATION ONLY

Figure 4.2-1
(Page 5 of 9)
SF2D1_BUILD.COM Build Script

```
$       RETURN
$!
$!-----------------------------------------------------------------------
$CREATE_ID:
$! Write the image identification options file dynamicly for build.
$!
$! If no CMS class was specified then this is a "most recent version"
$! build.  If a class was specified then it is a class dependent build.
$! Set the image identifier in the .exe so that we can tell what type of
$! build was run.  You can see the image id using the anal image command.
$       OPEN/write wp_id_file wp_build:wp_id.opt
$       IF p1 .eqs. "P"
$       THEN
$           IF p2 .eqs. ""
$           THEN
$               WRITE wp_id_file "  identification=""PROD -1"""
$           ELSE
$               WRITE wp_id_file "  identification=""PROD ''p2'"""
$           ENDIF
$       ENDIF
$
$       IF p1 .eqs. "T"
$       THEN
$           IF p2 .eqs. ""
$           THEN
$               WRITE wp_id_file "  identification=""TEST -1"""
$           ELSE
$               WRITE wp_id_file "  identification=""TEST ''p2'"""
$           ENDIF
$       ENDIF
$!
$       IF p1 .eqs. "L"
$       THEN
$           WRITE wp_id_file "  identification=""LOCAL -1"""
$       ENDIF
$!
$       CLOSE wp_id_file
$       DEFINE/nolog wp_id wp_build:wp_id.opt
$       RETURN
$!
$!-----------------------------------------------------------------------
$DO_BUILD:
$!
$! If this is a CMS class dependent build we need to fetch the CAMCON
$! libraries from the correct class.  This code should be replaced with
$! libraries references in the MMS file when the CAMCON libraries are
$! in CMS.
$       IF p2 .eqs. ""
$       THEN
$           gen_flag :== ""
$       ELSE
$           gen_flag :== "/gen=''p2'"
$       ENDIF
$!
$! Fetch the CAMCON libraries that are needed.
$       INQUIRE/nopunc camcon_fetch "Fetch CAMCON libs (Y or N) [N]: "
$       camcon_fetch = f$edit ("''camcon_fetch'", "upcase")
$       If camcon_fetch .eqs. "Y"
$       THEN
$           SET def wp_olb
$           CMS fet/occlude=none 'gen_flag camcon_lib.olb   ""
$           CMS fet/occlude=none 'gen_flag camdat_lib.olb   ""
```

INFORMATION ONLY

Figure 4.2-1
(Page 6 of 9)
SF2D1_BUILD.COM Build Script

```
$           CMS fet /occlude=none 'gen_flag' camsupes_lib.olb ""
$     ENDIF
$!
$!
$! Everything is ready, call MMS and run the build.
$     SET def wp_gen
$     MMS -
            /rules = wp_build:wp_mms_default_rules.mms -
            'mms_flag' -
            /description=wp_build:SF2D1.mms -
            /log -
            /out=wp_build:SF2D1_mms.log -
            /macros= wp_build:wp_gen_macros -
            'from_sources' -
            'mms_sca_flag'
$!
$!
$     save_status = $status
$     IF (.not. save_status)
$     THEN
$         WRITE sys$output "The build for SF2D1 had returned an error"
$         WRITE sys$output "Status returned ''save_status'. "
$         WRITE sys$output "Examine SF2D1_build.log and SF2D1_mms.log. "
$         WRITE mail_file "The build for SF2D1 returned an error. "
$         WRITE mail_file "Status returned ''save_status'. "
$         WRITE mail_file "The build is recorded in the two files "
$         WRITE mail_file "SF2D1_BUILD.LOG and SF2D1_MMS.LOG. "
$         mail_switch == 1
$     ENDIF
$!    DELETE/log wp_build:*.mms;*, *.obj;*, SF2D1_*.*;*, *.opt;*
$!
$     RETURN
$!------------------------------------------------------------------------
$INITIALIZE:
$! Set generic build logicals and symbols, initialize variables
$!
$     subsystem      :== "SF2D1_BUILD"
$     location       :== f$environment("DEFAULT")
$     from_sources   :== ""
$     mail_switch    :== 0
$     general_cflag  :== "/list/obj=wp_olb:/show=inc"
$     define/nolog cms$lib  c_SF2D,c_lib,c_wp
$!
$!
$! Check what user wants for from_sources
$     IF p3 .nes. "S"
$     THEN
$         from_sources = "/skip_intermediate"
$     ELSE
$         from_sources = "/from_sources"
$     ENDIF
$!
$! Set SCA param.
$     IF p5 .nes. "SCA"
$     THEN
$         sca_flag = ""
$     ELSE
$         sca_flag = "/ana"
$     ENDIF
$!
$! Define logicals for our working area; if desired they could all point the
$! a single directory.
```

INFORMATION ONLY

Figure 4.2-1

(Page 7 of 9)

SF2D1_BUILD.COM Build Script

```
   IF p1 .eqs. "P"
   THEN
       DEFINE/nolog wp_build wp$prodroot:[sf2d.build]
       DEFINE/nolog wp_exe         wp$prodroot:[sf2d.exe]
       DEFINE/nolog wp_olb         wp$prodroot:[sf2d.olb]
       DEFINE/nolog wp_src         wp$prodroot:[sf2d.src]
       DEFINE/nolog wp_pca         wp$prodroot:[sf2d.pca]
       DEFINE/nolog sca$library    wp$prodroot:[sf2d.sca]
       cms_flag :== "/cms"
   ENDIF

   IF p1 .eqs. "T"
   THEN
       DEFINE/nolog wp_build wp$testroot:[sf2d1.build]
       DEFINE/nolog wp_exe         wp$testroot:[sf2d1.exe]
       DEFINE/nolog wp_olb         wp$testroot:[sf2d1.olb]
       DEFINE/nolog wp_src         wp$testroot:[sf2d1.src]
       DEFINE/nolog wp_pca         wp$testroot:[sf2d1.pca]
       DEFINE/nolog sca$library    wp$testroot:[sf2d1.sca]
       cms_flag :== "/cms"
   ENDIF

   IF p1 .eqs. "L"
   THEN
       def_loc = f$environment("default")
       DEFINE/nolog wp_build 'def_loc'
       DEFINE/nolog wp_olb -
         'def_loc', wp$prodroot:[sf2d.olb], wp$testroot:[sf2d1.olb]
       DEFINE/nolog wp_src -
         'def_loc', wp$prodroot:[sf2d.src], wp$testroot:[sf2d1.src]
       DEFINE/nolog wp_exe         'def_loc'
       DEFINE/nolog wp_pca         'def_loc'
       DEFINE/nolog sca$library    'def_loc'
!  For the local build there are mms files that must be present
       IF f$search("wp_mms$default_rules.mms") .eqs. ""
       THEN
         CMS FETCH wp_mms$default_rules.mms ""
       ENDIF
       IF f$search("SF2D1.mms") .eqs. ""
       THEN
         CMS FETCH SF2D1.mms ""
       ENDIF
!
       IF f$search("PRESECOFL2DLIB.olb") .eqs. ""
       THEN
         IF f$search("wp_olb:presecofl2dlib.olb") .nes. ""
         THEN
            copy wp_olb:PRESECOFL2DLIB.olb 'def_loc'
         ELSE
            lib/cre PRESECOFL2DLIB.olb
         ENDIF
       ENDIF
!
   ENDIF
!
!
! Send the user mail if a error occurs.
    mail_list == f$getjpi( "", "username")
!
! MMS can generate some warnings that we don't need to stop for, disable error
! handling so that we don't stop for trivia.
```

INFORMATION ONLY

Figure 4.2-1

(Page 8 of 9)

SF2D1_BUILD.COM Build Script

```
$       SET NOON
$!
$       RETURN
$!------------------------------------------------------------------------
$ LOAD_SCA:
$! If the user requested a SCA build and load then check for a SCA
$! library, create it if it does not exist, load modules, and run some
$! SCA tests.
$!
$       IF p5 .eqs. "SCA"
$       THEN
$           IF F$SEARCH("sca$library:sca$event.dat") -
              .eqs."" THEN sca create lib sca$library
$!
$           sca
            set library sca$library
            load/replace wp_src:*.ana
            find symbol=function and occurrence=primary -
              AND NOT EXPAND  symbol=function and occurrence=reference -
              /output=wp_build:SF2D1_sca_mod_not_ref.txt
!           find symbol=variable and occurrence=primary -
!             AND NOT EXPAND  symbol=variable and occurrence=reference -
!             /output=wp_build:SF2D1_sca_var_not_ref.txt
            find called_by/end=PRESECOFL2D, depth=all -
              /output=wp_build:SF2D1_calltree.txt
! don't do the rest until the proper headers are in the code.
!             report internals /tar=txt /output=wp_build:SF2D1_sca_internals.txt
!             report help /output=wp_build:SF2D1_sca_help.hlp
              EXIT
$       ENDIF
$!
$       RETURN
$!------------------------------------------------------------------------
$ START_MAIL:
$! A VMS mail message is sent out if the build fails.
$! Provide basic setup for the message.
$!
$       mail_subject == "build for "
$       mailfile     == "wp_build:mail_build.msg"
$       OPEN/write mail_file 'mailfile
$       write mail_file "''F$TIME()' * Beginning ''subsystem' build type ''pt'"
$       RETURN
$!------------------------------------------------------------------------
$TERMINATE:
$!
$! Check for error status, if the mail flag is set send the message.
$! Then cleanup and exit.
$!
$! Try to notify the user if there where any build issues.
$       DEASSIGN /proc sys$output        ! Closes the build log for this file.
$       SET noverify
$!
$! If there was no previously flagged error or problem, search the log files
$! for any fatal, error, or warning messages.
$       IF mail_switch .ne. 1
$       THEN
$           Search wp_build:SF2D1_build.log,wp_build:SF2D1_mms.log -
              "-f-","-e-","-w-"/mat=or
$           IF $STATUS .eq. 1
$           THEN
$           WRITE mail_file  "The build for SF2D1 returned an error or warning. "
$           WRITE mail_file  "Please examine in the build directory: "
```

INFORMATION ONLY

Figure 4.2-1
(Page 9 of 9)
SF2D1_BUILD.COM Build Script

```
$           WRITE mail_file  "SF2D1_BUILD.LOG and SF2D1_MMS.LOG. "
$           mail_switch == 1
$           ENDIF
$       ENDIF
$!
$       CLOSE mail_file
$       IF mail_switch .eq. 1 then -
$           mail /sub="''mail_subject' ''subsystem'" -
$            'mailfile' "''mail_list'"
$       DELETE 'mailfile';*
$       set noon
$!
$       IF location .nes. "" then set default 'location
$       IF F$TRNLNM("wp_build") .NES. "" THEN  DEASSIGN wp_build
$       IF F$TRNLNM("wp_exe") .NES. "" THEN  DEASSIGN wp_exe
$       IF F$TRNLNM("wp_olb") .NES. "" THEN  DEASSIGN wp_olb
$       IF F$TRNLNM("wp_src") .NES. "" THEN  DEASSIGN wp_src
$       IF F$TRNLNM("wp_cca") .NES. "" THEN  DEASSIGN wp_cca
$       IF F$TRNLNM("cms$library") .NES. "" THEN  DEASSIGN cms$library
$       done_verify = F$VERIFY(save_verify)
$       RETURN
$!
$!-----------------------------------------------------------------------
$ EXIT
$!-----------------------------------------------------------------------
$! CMS REPLACEMENT HISTORY, Element SF2D1_BUILD.COM
$! *4     9-AUG-1995 10:58:16 CMWILLI "2c test build in the sf2d1 test area."
$! *3     9-AUG-1995 11:12:17 CMWILLI "IMPROVE THE WAY THE LOCAL BUILD FINDS THE
LOCAL .OLB."
$! *2     8-AUG-1995 16:45:03 CMWILLI "UPDATE TO CURRENT BUILD TEMPLATE, INCLUDES
DEVELOPERS BUILD."
$! *1     8-AUG-1995 11:49:11 CMWILLI "INITIAL LOAD."
$! CMS REPLACEMENT HISTORY, Element SF2D1_BUILD.COM
```

INFORMATION ONLY

Figure 4.2-2
(Page 1 of 6)
SF2D1.MMS Input File for Build Script

```
!macros
LIBRARYMODULES = -
wp_olb:SF2D1_ADDMEM, -
wp_olb:SF2D1_BLOCK, -
wp_olb:SF2D1_DELTAS, -
wp_olb:SF2D1_DYNMEM, -
wp_olb:SF2D1_GETBND, -
wp_olb:SF2D1_GETLDS, -
wp_olb:SF2D1_SETPTS, -
wp_olb:SF2D1_SETPTM, -
wp_olb:SF2D1_GETSLV, -
wp_olb:SF2D1_INIARR, -
wp_olb:SF2D1_ININAM, -
wp_olb:SF2D1_INISEC, -
wp_olb:SF2D1_MEMADJ, -
wp_olb:SF2D1_PRAQPR, -
wp_olb:SF2D1_PREPRO, -
wp_olb:SF2D1_PRESECOFL2D, -
wp_olb:SF2D1_PROCDB, -
wp_olb:SF2D1_PROP_RANGE, -
wp_olb:SF2D1_PRSECD, -
wp_olb:SF2D1_PRSCRC, -
wp_olb:SF2D1_RDAQPR, -
wp_olb:SF2D1_RDBNDC, -
wp_olb:SF2D1_RDCNST, -
wp_olb:SF2D1_RDINIT, -
wp_olb:SF2D1_RDLTIM, -
wp_olb:SF2D1_RDNAME, -
wp_olb:SF2D1_RDRCUF, -
wp_olb:SF2D1_RDPTIM, -
wp_olb:SF2D1_RDSECD, -
wp_olb:SF2D1_RDSOLV, -
wp_olb:SF2D1_RDWELL, -
wp_olb:SF2D1_SCANSI, -
wp_olb:SF2D1_SETCOORD, -
wp_olb:SF2D1_SETSHO, -
wp_olb:SF2D1_SETGHP, -
wp_olb:SF2D1_SETSHA, -
wp_olb:SF2D1_SET_GLOC2, -
wp_olb:SF2D1_SET_GREG2, -
wp_olb:SF2D1_STATUS, -
wp_olb:SF2D1_WRTM2I, -
wp_olb:SF2D1_WRTM2R, -
wp_olb:SF2D1_WRTS2I, -
wp_olb:SF2D1_WRTS2L, -
wp_olb:SF2D1_WRTS2R, -
wp_olb:SF2D1_WRTSEC, -
wp_olb:SF2D1_WRTWEL

LIBRARIES = wp_olb:presecofl2dlib.olb($(LIBRARYMODULES)) -

! dependencies
wp_exe:presecofl2d.exe : $(LIBRARIES)
link $'linkflags' /exe=wp_exe:$(exe_name) -
  wp_olb:presecofl2dlib.olb/lib/inc=presecofl2d,-
  wp_olb:camdat_lib.olb/lib,        -
  wp_olb:camcon_lib.olb/lib,        -
  wp_olb:camsupes_lib.olb/lib,      -
  sys$library:vaxcrtl.olb/lib,      -
  wp_build:wp_id.opt/options

wp_olb:SF2D1_ADDMEM.obj : wp_src:SF2D1_ADDMEM.for, -
```

INFORMATION ONLY

Figure 4.2-2
(Page 2 of 6)
SF2D1.MMS Input File for Build Script

```
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_GEOPAR.inc, -
                              wp_src:SF2D1_INDEX.inc, -
                              wp_src:SF2D1_LENGTH.inc, -
                              wp_src:SF2D1_LPCPAR.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_REGPAR.inc, -
                              wp_src:SF2D1_SEC01L.inc, -
                              wp_src:SF2D1_SEC02R.inc
        fortran $(cflags) wp_src:SF2D1_ACCMEM.for

wp_olb:SF2D1_BLOCK.obj : wp_src:SF2D1_BLOCK.for
        fortran $(cflags) wp_src:SF2D1_BLOCK.for

wp_olb:SF2D1_DELTAS.obj : wp_src:SF2D1_DELTAS.for, -
                              wp_src:SF2D1_LOGIC.inc
        fortran $(cflags) wp_src:SF2D1_DELTAS.for

wp_olb:SF2D1_DYNMEM.obj : wp_src:SF2D1_DYNMEM.for, -
                              wp_src:SF2D1_FILEUN.inc
        fortran $(cflags) wp_src:SF2D1_DYNMEM.for

wp_olb:SF2D1_GETBND.obj : wp_src:SF2D1_GETBND.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_SEC02R.inc, -
                              wp_src:SF2D1_SEC01L.inc
        fortran $(cflags) wp_src:SF2D1_GETBND.for

wp_olb:SF2D1_GETLTS.obj : wp_src:SF2D1_GETLTS.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_SEC01L.inc
        fortran $(cflags) wp_src:SF2D1_GETLTS.for

wp_olb:SF2D1_GETRTS.obj : wp_src:SF2D1_GETRTS.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_SEC02R.inc
        fortran $(cflags) wp_src:SF2D1_GETRTS.for

wp_olb:SF2D1_GETRUN.obj : wp_src:SF2D1_GETRUN.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_INDEX.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
wp_src:SF2D1_SEC01L.inc, -
                              wp_src:SF2D1_SEC02R.inc
        fortran $(cflags) wp_src:SF2D1_GETRUN.for

wp_olb:SF2D1_GETSLV.obj : wp_src:SF2D1_GETSLV.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_SEC01L.inc, -
                              wp_src:SF2D1_SEC02R.inc
        fortran $(cflags) wp_src:SF2D1_GETSLV.for

wp_olb:SF2D1_INIARR.obj : wp_src:SF2D1_INIARR.for
        fortran $(cflags) wp_src:SF2D1_INIARR.for

wp_olb:SF2D1_ININAM.obj : wp_src:SF2D1_ININAM.for, -
                              wp_src:SF2D1_ALTNAM.inc
        fortran $(cflags) wp_src:SF2D1_ININAM.for

wp_olb:SF2D1_INISEC.obj : wp_src:SF2D1_INISEC.for, -
```

INFORMATION ONLY

Figure 4.2-2
(Page 3 of 6)
SF2D1.MMS Input File for Build Script

```
wp_src:SF2D1_LOGIC.inc, -
                                wp_src:SF2D1_SECOOL.inc, -
                                wp_src:SF2D1_SECOOR.inc
         fortran $(cflags) wp_src:SF2D1_INISEC.for

wp_olb:SF2D1_MEMALC.obj : wp_src:SF2D1_MEMALC.for, -
                                wp_src:SF2D1_ARRAY.inc, -
                                wp_src:SF2D1_DYNAME.inc, -
                                wp_src:SF2D1_INDEX.inc, -
                                wp_src:SF2D1_LENGTH.inc, -
                                wp_src:SF2D1_LOCAT.inc
         fortran $(cflags) wp_src:SF2D1_MEMALC.for

wp_olb:SF2D1_PRAIPR.obj : wp_src:SF2D1_PRAIPR.for, -
                                wp_src:SF2D1_ALTNAM.inc, -
                                wp_src:SF2D1_ARRAY.inc, -
                                wp_src:SF2D1_FILEUN.inc, -
                                wp_src:SF2D1_LOGIC.inc
         fortran $(cflags) wp_src:SF2D1_PRAIPR.for

wp_olb:SF2D1_PREPRO.obj : wp_src:SF2D1_PREPRO.for, -
                                wp_src:SF2D1_FILEUN.inc, -
                                wp_src:SF2D1_LOGIC.inc
         fortran $(cflags) wp_src:SF2D1_PREPRO.for

wp_olb:SF2D1_PRESECOFL2D.obj : wp_src:SF2D1_PRESECOFL2D.for, -
                                     wp_src:SF2D1_ARRAY.inc, -
                                     wp_src:SF2D1_ALTNAM.inc, -
                                     wp_src:SF2D1_LOCAT.inc, -
                                     wp_src:SF2D1_LOGIC.inc
         fortran $(cflags) wp_src:SF2D1_PRESECOFL2D.for

wp_olb:SF2D1_PROCDB.obj : wp_src:SF2D1_PROCDB.for, -
                                wp_src:SF2D1_ARRAY.inc, -
                                wp_src:SF2D1_FILEUN.inc, -
wp_src:SF2D1_GEOPAR.inc, -
                                wp_src:SF2D1_LOCAT.inc, -
                                wp_src:SF2D1_LOCPAR.inc, -
                                wp_src:SF2D1_LOGIC.inc, -
                                wp_src:SF2D1_REGPAR.inc
         fortran $(cflags) wp_src:SF2D1_PROCDB.for

wp_olb:SF2D1_PROP_RANGE.obj : wp_src:SF2D1_PROP_RANGE.for
         fortran $(cflags) wp_src:SF2D1_PROP_RANGE.for

wp_olb:SF2D1_PRSECO.obj : wp_src:SF2D1_PRSECO.for, -
                                wp_src:SF2D1_ARRAY.inc, -
                                wp_src:SF2D1_FILEUN.inc, -
                                wp_src:SF2D1_GEOPAR.inc, -
                                wp_src:SF2D1_LOCAT.inc, -
                                wp_src:SF2D1_LOCPAR.inc, -
                                wp_src:SF2D1_LOGIC.inc, -
                                wp_src:SF2D1_REGPAR.inc
         fortran $(cflags) wp_src:SF2D1_PRSECO.for

wp_olb:SF2D1_PRSORC.obj : wp_src:SF2D1_PRSORC.for, -
                                wp_src:SF2D1_ALTNAM.inc, -
                                wp_src:SF2D1_ARRAY.inc, -
                                wp_src:SF2D1_FILEUN.inc, -
                                wp_src:SF2D1_LOGIC.inc
         fortran $(cflags) wp_src:SF2D1_PRSORC.for
```

Figure 4.2-2

(Page 4 of 6)

SF2D1.MMS Input File for Build Script

```
wp_clb:SF2D1_RDAPFR.obj : wp_src:SF2D1_RDAPFR.for, -
wp_src:SF2D1_FFRDAT.inc, -
                          wp_src:SF2D1_FILEUN.inc, -
wp_src:SF2D1_LOCAT.inc
        fortran $(cflags) wp_src:SF2D1_RDAPFR.for

wp_clb:SF2D1_RDBNDC.obj : wp_src:SF2D1_RDBNDC.for, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_RDBNDC.for

wp_clb:SF2D1_RDCNST.obj : wp_src:SF2D1_RDCNST.for, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_LOGIC.inc
        fortran $(cflags) wp_src:SF2D1_RDCNST.for

wp_clb:SF2D1_RDINIT.obj : wp_src:SF2D1_RDINIT.for, -
                          wp_src:SF2D1_ARRAY.inc, -
                          wp_src:SF2D1_ALTNAM.inc, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_LOGIC.inc, -
                          wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_RDINIT.for

wp_clb:SF2D1_RDLTIM.obj : wp_src:SF2D1_RDLTIM.for, -
                          wp_src:SF2D1_FILEUN.inc, -
wp_src:SF2D1_SECO2L.inc
        fortran $(cflags) wp_src:SF2D1_RDLTIM.for

wp_clb:SF2D1_RDNAME.obj : wp_src:SF2D1_RDNAME.for, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_ALTNAM.inc
        fortran $(cflags) wp_src:SF2D1_RDNAME.for

wp_clb:SF2D1_RDRCLP.obj : wp_src:SF2D1_RDRCLP.for, -
                          wp_src:SF2D1_ARRAY.inc, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_RDRCLP.for

wp_clb:SF2D1_RDRTIM.obj : wp_src:SF2D1_RDRTIM.for, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_RDRTIM.for

wp_clb:SF2D1_RDSECO.obj : wp_src:SF2D1_RDSECO.for, -
                          wp_src:SF2D1_ARRAY.inc, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_FFRDAT.inc, -
                          wp_src:SF2D1_LOCAT.inc, -
                          wp_src:SF2D1_LOGIC.inc
        fortran $(cflags) wp_src:SF2D1_RDSECO.for

wp_clb:SF2D1_RDSOLV.obj : wp_src:SF2D1_RDSOLV.for, -
                          wp_src:SF2D1_FILEUN.inc, -
                          wp_src:SF2D1_SECO2L.inc, -
                          wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_RDSOLV.for

wp_clb:SF2D1_RDWELL.obj : wp_src:SF2D1_RDWELL.for, -
                          wp_src:SF2D1_ALTNAM.inc, -
                          wp_src:SF2D1_FILEUN.inc, -
```

INFORMATION ONLY

Figure 4.2-2
(Page 5 of 6)
SF2D1.MMS Input File for Build Script

```
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_SECO1R.inc
fortran $(cflags) wp_src:SF2D1_RDWELL.for

wp_olb:SF2D1_SCANSI.obj : wp_src:SF2D1_SCANSI.for, -
wp_src:SF2D1_EFFDAT.inc, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_SECO1R.inc
        fortran $(cflags) wp_src:SF2D1_SCANSI.for

wp_olb:SF2D1_SETCOORD.obj : wp_src:SF2D1_SETCOORD.for, -
                              wp_src:SF2D1_LOGIC.inc
        fortran $(cflags) wp_src:SF2D1_SETCOORD.for

wp_olb:SF2D1_SETGHC.obj : wp_src:SF2D1_SETGHC.for
        fortran $(cflags) wp_src:SF2D1_SETGHC.for

wp_olb:SF2D1_SETGHR.obj : wp_src:SF2D1_SETGHR.for
        fortran $(cflags) wp_src:SF2D1_SETGHR.for

wp_olb:SF2D1_SETHEA.obj : wp_src:SF2D1_SETHEA.for
        fortran $(cflags) wp_src:SF2D1_SETHEA.for

wp_olb:SF2D1_SET_GLOC1.obj : wp_src:SF2D1_SET_GLOC1.for
        fortran $(cflags) wp_src:SF2D1_SET_GLOC1.for

wp_olb:SF2D1_SET_GREG1.obj : wp_src:SF2D1_SET_GREG1.for
        fortran $(cflags) wp_src:SF2D1_SET_GREG1.for

wp_olb:SF2D1_STATUS.obj : wp_src:SF2D1_STATUS.for, -
                              wp_src:SF2D1_FILEUN.inc
        fortran $(cflags) wp_src:SF2D1_STATUS.for

wp_olb:SF2D1_WRTM2L.obj : wp_src:SF2D1_WRTM2L.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_SECO2L.inc
        fortran $(cflags) wp_src:SF2D1_WRTM2L.for

wp_olb:SF2D1_WRTM2R.obj : wp_src:SF2D1_WRTM2R.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_WRTM2R.for

wp_olb:SF2D1_WRTS2I.obj : wp_src:SF2D1_WRTS2I.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_SECO1L.inc, -
                              wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_WRTS2I.for

wp_olb:SF2D1_WRTS2L.obj : wp_src:SF2D1_WRTS2L.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_SECO2L.inc
        fortran $(cflags) wp_src:SF2D1_WRTS2L.for

wp_olb:SF2D1_WRTS2R.obj : wp_src:SF2D1_WRTS2R.for, -
                              wp_src:SF2D1_FILEUN.inc, -
                              wp_src:SF2D1_LOGIC.inc, -
                              wp_src:SF2D1_SECO2R.inc
        fortran $(cflags) wp_src:SF2D1_WRTS2R.for
```

INFORMATION ONLY

Figure 4.2-2
(Page 6 of 6)
SF2D1.MMS Input File for Build Script

```
wp_olb:SF2D1_WRTSEC.obj  :  wp_src:SF2D1_WRTSEC.for, -
                            wp_src:SF2D1_ARRAY.inc, -
                            wp_src:SF2D1_[illegible].inc, -
                            wp_src:SF2D1_LOGIC.inc, -
                            wp_src:SF2D1_SEC02R.inc
        fortran $(fflags)   wp_src:SF2D1_WRTSEC.for
wp_olb:SF2D1_WRTWEL.obj  :  wp_src:SF2D1_WRTWEL.for, -
wp_src:SF2D1_FILEUN.inc, -
                            wp_src:SF2D1_LOGIC.inc, -
                            wp_src:SF2D1_SEC02R.inc
        fortran $(fflags)   wp_src:SF2D1_WRTWEL.for
```

INFORMATION ONLY

Figure 4.2-3
SCA Output File of Modules Not Referenced

```
PRESECOFL2D procedure
                  PRESECOFL2D 1           SUBROUTINE or PROGRAM declaration
```

INFORMATION ONLY

Figure 4.2-4
(Page 1 of 3)
SCA Call Tree Output

```
PRESECOFL2D procedure calls
   DBIOPEN routine
   PRESETUP routine
   EXTPID routine
   MDINIT routine
   MDINIT routine
   PREPRO routine calls
      FILCMDLIN routine
      FILDFNAM routine
      FILRDNAMS routine
      FILWRNAMS routine
      QAABORT routine
      QABANNER routine
      QAIOEDIS routine
      QAPAGE routine
      QASETUP routine
   PROCDB routine calls
      ADDMEM routine calls
         MAX function
         MAX0 function
         QAABORT routine  (See above)
      ADDMEM2 routine calls
         MAX function  (See above)
         MAX0 function  (See above)
         QAABORT routine  (See above)
      ADDMEM3 routine calls
         MAX function  (See above)
         MAX0 function  (See above)
         QAABORT routine  (See above)
      ADDMEM4 routine calls
         MAX function  (See above)
         MAX0 function  (See above)
         QAABORT routine  (See above)
      DBSIZES routine
      DBINIC routine
      DELTAS routine
      DYNMEM routine calls
         MEMADD routine calls
            MDEROR routine
            MDGET routine
            MDLIST routine
            MDRSRV routine
            MDSTAT routine
      ININAM routine
      INISEC routine
      PRAQPR routine calls
         DBIATTR routine
         DBISTEP routine
         DBIVAR routine
         QAABORT routine  (See above)
      QAABORT routine  (See above)
      SCANSI routine calls
         FFRDFLDS routine
      GETBND routine calls
            FFRDFLDS routine  (See above)
            QAABORT routine  (See above)
         GETLTS routine calls
            FFRDFLDS routine  (See above)
            QAABORT routine  (See above)
         GETRTS routine calls
         FFRDFLDS routine  (See above)
            QAABORT routine  (See above)
```

INFORMATION ONLY

Figure 4.2-4
(Page 2 of 3)
SCA Call Tree Output

```
        GETRIV routine calls
           FFRDFLDS routine  (See above)
           QAABORT routine  (See above)
        GETSLV routine calls
           FFRDFLDS routine  (See above)
           QAABORT routine  (See above)
        QAABORT routine  (See above)
        RECHET routine calls
           FFRDFLDS routine  (See above)
           QAABORT routine  (See above)
        RDNAME routine calls
           FFRDFLDS routine  (See above)
           QAABORT routine  (See above)
     SETCOORD routine
  PRSECO routine calls
     AQFR_GREG routine calls
        COS function
        FIND_ICA routine
        INTF routine
        SIN function
     DBINIT routine  (See above)
     DELTAS routine  (See above)
     GLOC_WRITE routine calls
        DBLE function
        PROP_RANGE routine calls
           MAX function  (See above)
           MIN function
           QAABORT routine  (See above)
     GREG_GLOC routine calls
        COS function  (See above)
        FIND_ICA_L routine
        INTF_L routine
        SIN function  (See above)
     GREG_WRITE routine calls
        DBLE function  (See above)
        PROP_RANGE routine  (See above)
     INIARR routine
     PRAQFR routine  (See above)
     PRSORC routine calls
        DBITIME routine
        DBIVAR routine  (See above)
     QAABORT routine  (See above)
     SETCOORD routine  (See above)
     SETGHC routine
  SETGHR routine
     SETHEA routine
  QACPUS routine
  QAPAGE routine  (See above)
  RDSECO routine calls
     FFRDFLDS routine  (See above)
     QAABORT routine  (See above)
     RDAQFR routine calls
        FFRDFLDS routine  (See above)
        QAABORT routine  (See above)
     RDBNDC routine calls
        FFRDFLDS routine  (See above)
        QAABORT routine  (See above)
     RDINIT routine calls
        DBIATTR routine  (See above)
        DBITIME routine  (See above)
        DBIVAR routine  (See above)
     FFRDFLDS routine  (See above)
```

INFORMATION ONLY

Figure 4.2-4
(Page 3 of 3)
SCA Call Tree Output

```
            QAABORT routine   See above
         RDLTIM routine calls
            FFRDFLDS routine   See above
            QAABORT routine   See above
         RDPROP routine calls
            DBIPROP routine
            FFRDFLDS routine   See above
            QAABORT routine   See above
         RDRTIM routine calls
            FFRDFLDS routine   See above
            QAABORT routine   See above
         RDSOLV routine calls
            FFRDFLDS routine   See above
            QAABORT routine   See above
         RDWELL routine calls
            FFRDFLDS routine   See above
            QAABORT routine   See above
      STATUS routine calls
         LEN function
      WRTBED routine calls
         WRTMCL routine calls
            DBLE function   See above
         WRTMCR routine calls
            DBLE function   See above
         WRTSCI routine
         WRTSCL routine
         WRTSCR routine
         WRTWBL routine calls
                                     DBLE function   See above
```

INFORMATION ONLY

Figure 4.2-5
SCA Output File of All Modules Included in Static Analysis

Module	*	Ident	Language	Compiled
ADDMEM	1	01	FORTRAN	21-Aug-1995 16:24
ADFR_GREG	1	01	FORTRAN	21-Aug-1995 16:25
DELTAS	1	01	FORTRAN	21-Aug-1995 16:24
DYNMEM	1	01	FORTRAN	21-Aug-1995 16:24
[illegible]	1	01	FORTRAN	21-Aug-1995 16:25
[illegible]	1	01	FORTRAN	21-Aug-1995 16:25
GETSWS	1	01	FORTRAN	21-Aug-1995 16:24
GETUTC	1	01	FORTRAN	21-Aug-1995 16:24
GETPTS	1	01	FORTRAN	21-Aug-1995 16:24
GETRUN	1	01	FORTRAN	21-Aug-1995 16:24
GETSLV	1	01	FORTRAN	21-Aug-1995 16:24
GLOC_WRITE	1	01	FORTRAN	21-Aug-1995 16:25
GREG_GLOB	1	01	FORTRAN	21-Aug-1995 16:25
GREG_WRITE	1	01	FORTRAN	21-Aug-1995 16:25
INIARR	1	01	FORTRAN	21-Aug-1995 16:24
INDNAM	1	01	FORTRAN	21-Aug-1995 16:24
INISEP	1	01	FORTRAN	21-Aug-1995 16:24
INTP	1	01	FORTRAN	21-Aug-1995 16:24
INTP_C	1	01	FORTRAN	21-Aug-1995 16:25
MEMACC	1	01	FORTRAN	21-Aug-1995 16:24
PRADPR	1	01	FORTRAN	21-Aug-1995 16:24
PREPRO	1	01	FORTRAN	21-Aug-1995 16:24
PRESECOFL2D	1	01	FORTRAN	21-Aug-1995 16:24
PROCDB	1	01	FORTRAN	21-Aug-1995 16:24
PROP_RANGE	1	01	FORTRAN	21-Aug-1995 16:24
PRSECO	1	01	FORTRAN	21-Aug-1995 16:24
PRSCRD	1	01	FORTRAN	21-Aug-1995 16:25
RDACFR	1	01	FORTRAN	21-Aug-1995 16:25
RDBNDC	1	01	FORTRAN	21-Aug-1995 16:25
RDCNST	1	01	FORTRAN	21-Aug-1995 16:25
RDINIT	1	01	FORTRAN	21-Aug-1995 16:25
RDITIM	1	01	FORTRAN	21-Aug-1995 16:25
RDNAME	1	01	FORTRAN	21-Aug-1995 16:25
RDRCLF	1	01	FORTRAN	21-Aug-1995 16:25
RDRTIM	1	01	FORTRAN	21-Aug-1995 16:25
RDSECO	1	01	FORTRAN	21-Aug-1995 16:25
RDSOLV	1	01	FORTRAN	21-Aug-1995 16:25
RDWELL	1	01	FORTRAN	21-Aug-1995 16:25
SCANSI	1	01	FORTRAN	21-Aug-1995 16:25
SETCOORD	1	01	FORTRAN	21-Aug-1995 16:25
SETGHD	1	01	FORTRAN	21-Aug-1995 16:25
SETGHR	1	01	FORTRAN	21-Aug-1995 16:25
SETHEA	1	01	FORTRAN	21-Aug-1995 16:25
SFIDI_BLOCK	1	01	FORTRAN	21-Aug-1995 16:24
STATUS	1	01	FORTRAN	21-Aug-1995 16:26
WRTM2D	1	01	FORTRAN	21-Aug-1995 16:26
WRTM2R	1	01	FORTRAN	21-Aug-1995 16:26
WRTS2D	1	01	FORTRAN	21-Aug-1995 16:26
WRTS2L	1	01	FORTRAN	21-Aug-1995 16:26
WRTS2R	1	01	FORTRAN	21-Aug-1995 16:26
WRTSEC	1	01	FORTRAN	21-Aug-1995 16:26
WRTWEL	1	02	FORTRAN	21-Aug-1995 16:26

INFORMATION ONLY

5.0 COVERAGE ANALYSIS

The coverage analysis for PRESECOFL2D was performed using DECset PCA. A unique executable, called WP$TESTROOT:[SF2D1.EXE]PRESECOFL2D_TEST_PCA.EXE was created for the purpose of coverage analysis. This executable was generated by using the SCMS build script, SF2D1_BUILD.COM. The build script is shown in **Figure 4.2-1**. The coverage analysis was performed in the directory WP$TESTROOT:[SF2D1.PCA].

The command file SF2D1_TESTCASE_PCA_CUM.COM, shown in **Figure 5-1**, was used to run the PRESECOFL2D test case with PCA. The PCA output for the test case has been included and can be located by referring to **Table 5-1**. A log of the process was created and can be found in **Figure 5-2**.

Table 5-1
Location of Coverage Analysis Output Files

PCA Output	Figure Number
PCA Coverage Analysis for Test Case	**Figure 5-3**
Cumulative PCA Coverage Analysis Test Cases	**Figure 5-4**

The code sponsor was asked to review the output files from the coverage analysis and to write a summary of that review. The code sponsor's comments can be found in **Figure 5-5**. Nothing unexpected was revealed during the PCA process.

INFORMATION ONLY

Figure 5-1
(Page 1 of 2)
Script Used to Run Test Case for PCA Analysis

```
$ set noon
$!
$!==========================================================================
$!
$!  This file tests PRESECOFL2D.
$!
$!==========================================================================
$!
$!
$ define pca$debug sys$library:pca$collector.exe
$ TESTDIR_SYM == "wp$testroot:[sf2d1.pca]"
$ PRESECOFL2D == "$ wp$testroot:[sf2d1.exe]presecofl2d_test_pca.exe"
$ define/proc testdir wp$testroot:[sf2d1.pca]
$ sh sym testdir_sym
$ IF F$MODE() .EQS. "BATCH" .AND. F$ENVIRONMENT("DEPTH") .EQ. 0 THEN -
    SET DEFAULT 'TESTDIR_SYM'
$!
$!
$! If the cum pca file exists, rename it so we get a new file
$    rename sf2d1_testcase_cum.pca *.pca_old
$!
$!--------------------------------------------------------------------------
$! Set up the first test case and run PRESECOFL2D with the PCA collector
$!    comparison to analytical solution
$!
$!
$ write sys$output "STARTING THE PRESECOFL2D TEST CASE - CUMULATIVE"
$!
$ PRESECOFL2D SF2D1_REGION_TEST.CDB SF2D1_REGION_TEST.CDB -
    SF2D1_LOCAL_TEST.CDB CANCEL SF2D1_PRESECOFL_TEST.INP SECOFL2D_TEST.INP -
    SF2D1_REGION_TEST.PRP SF2D1_LOCAL_TEST.PRP CANCEL PRESECOFL_TEST.DBG
        set data append sf2d1_testcase_cum.pca
                set cover program by routine
        go
$!
$ write sys$output "STARTING THE PRESECOFL2D TEST CASE - SINGLE"
$!
$!
$ PRESECOFL2D SF2D1_REGION_TEST.CDB SF2D1_REGION_TEST.CDB -
    SF2D1_LOCAL_TEST.CDB CANCEL SF2D1_PRESECOFL_TEST.INP SECOFL2D_TEST.INP -
    SF2D1_REGION_TEST.PRP SF2D1_LOCAL_TEST.PRP CANCEL PRESECOFL_TEST.DBG
        set data append sf2d1_testtest.pca
                set cover program by routine
        go
$!
$!
$!--------------------------------------------------------------------------
$!
$! Run the PCA analyzer on the collected data.  Write the results to file
$!
$ write sys$output "STARTING THE PCA FOR THE PRESECOFL2D TEST CASE"
$!
$!
$   pca sf2d1_testtest.pca
        tab/cover program by routine
        file sf2d1_test_cover.txt
        exit
$!
$ write sys$output "STARTING THE CUMULATIVE PCA"
$!
$!
$   pca sf2d1_testcase_cum.pca
```

INFORMATION ONLY

Figure 5-1

(Page 2 of 2)

Script Used to Run Test Case for PCA Analysis

```
        tab/cover program by routine
        file sf2d1_coverage_sum.txt
        tab/nocover program by routine
        file sf2d1_noncoverage_sum.txt
        exit
$!
$  write sys$output "PCA FOR PRESECOFL2D SUCCESSFULLY COMPLETED"
$!
$!----------------------------------------------------------------------
```

INFORMATION ONLY

Figure 5-2
(Page 1 of 2)
PCA Log of Coverage Analysis

```
%DCL-I-SUPERSEDE, previous value of LIBSDEBUG has been superseded
%DCL-I-SUPERSEDE, previous value of TESTDIR has been superseded
  TESTDIR_SYM == "wp$testroot:[sf2d1.pca]"
%RENAME-E-SEARCHFAIL, error searching for
WP$TESTROOT:[000000.SF2D1.PCA]SF2D1_TESTCASE_CUM.PCA;
-RMS-E-FNF, file not found
STARTING THE PRESECOFL2D TEST CASE - CUMULATIVE

         PCA Collector Version V4.3-4

%PCA-I-CREFILE, creating file SF2D1_TESTCASE_CUM.PCA
%PCA-I-BEGINCOL, data collection begins

                              PRESECOFL2D

 PPPPPP  RRRRRR  EEEEEEE  SSSSSS EEEEEEE  CCCCC   OOOOO  FFFFFFF LL          2222
 PP   PP RR   RR EE      SS      EE      CC   CC OO   OO FF      LL         2    2
 PP   PP RR   RR EE      SS      EE      CC      OO   OO FF      LL              2
 PPPPPP  RRRRRR  EEEEE    SSSSS  EEEEE   CC      OO   OO FFFFF   LL             2
 PP      RRRRR   EE           SS EE      CC      OO   OO FF      LL           2
 PP      RR  RR  EE           SS EE      CC   CC OO   OO FF      LL          2
 PP      RR   RR EEEEEEE SSSSSS  EEEEEEE  CCCCC   OOOOO  FF      LLLLLLL   222222

                              PRESECOFL2D

                      PRESECOFL2D Version 4.02ZO
                 Version Date 05/20/94  QA Level A
                      Written by rebecca blaine
                     Sponsored by rebecca blaine

                      Run on 09/08/95 at 10:39:50
                 Run on ALPHA AXP BEATLE OpenVMS V6.1

PRESECOFL2D Normal Completion
%PCA-I-ENDCOL, data collection ends
STARTING THE PRESECOFL2D TEST CASE - SINGLE

         PCA Collector Version V4.3-4

%PCA-I-CREFILE, creating file SF2D1_TESTTEST.PCA
%PCA-I-BEGINCOL, data collection begins

                              PRESECOFL2D

 PPPPPP  RRRRRR  EEEEEEE  SSSSSS EEEEEEE  CCCCC   OOOOO  FFFFFFF LL          2222
 PP   PP RR   RR EE      SS      EE      CC   CC OO   OO FF      LL         2    2
 PP   PP RR   RR EE      SS      EE      CC      OO   OO FF      LL              2
 PPPPPP  RRRRRR  EEEEE    SSSSS  EEEEE   CC      OO   OO FFFFF   LL             2
 PP      RRRRR   EE           SS EE      CC      OO   OO FF      LL           2
 PP      RR  RR  EE           SS EE      CC   CC OO   OO FF      LL          2
 PP      RR   RR EEEEEEE SSSSSS  EEEEEEE  CCCCC   OOOOO  FF      LLLLLLL   222222

                              PRESECOFL2D
PRESECOFL2D Version 4.02ZO
                 Version Date 05/20/94  QA Level A
                      Written by rebecca blaine
Sponsored by rebecca blaine
                      Run on 09/08/95 at 10:40:04
                 Run on ALPHA AXP BEATLE OpenVMS V6.1

PRESECOFL2D Normal Completion
%PCA-I-ENDCOL, data collection ends
STARTING THE PCA FOR THE PRESECOFL2D TEST CASE
```

INFORMATION ONLY

Figure 5-2
(Page 2 of 2)
PCA Log of Coverage Analysis

```
         DEC Performance and Coverage Analyzer Version V4.3-4

                   DEC Performance and Coverage Analyzer                 Page 1

                 (Test Coverage Data (117 data points total) - "*"

                                   Data
  Bucket Name                      Count  Percent
  ADDMEM\
   ADDMEM . . . . . . . . . . . .     1     0.9%
  DBIATTR\
   DBIATTR . . . . . . . . . . . .    1     0.9%
  DBIOPEN\
   DBIOPEN . . . . . . . . . . . .    1     0.9%
  DBIPROP\
   DBIPROP . . . . . . . . . . . .    1     0.9%
  DBISIZES\
   DBISIZES . . . . . . . . . . .     1     0.9%
  DBIXYZ\
   DBIXYZ . . . . . . . . . . . .     1     0.9%
  DBSETUP\
   DBSETUP . . . . . . . . . . . .    1     0.9%
  DELTAS\
   DELTAS . . . . . . . . . . . .     1     0.9%
%PCA-I-CREFILE, creating file WPSTESTROOT:[000000.SF2D1.PCA]SF2D1_TEST_COVER.TXT;1
STARTING THE CUMULATIVE PCA

         DEC Performance and Coverage Analyzer Version V4.3-4

                   DEC Performance and Coverage Analyzer                 Page 1

                 Test Coverage Data (117 data points total) - "*"

                                   Data
  Bucket Name                      Count  Percent
  ADDMEM\
   ADDMEM . . . . . . . . . . . .     1     0.9%
  DBIATTR\
   DBIATTR . . . . . . . . . . . .    1     0.9%
  DBIOPEN\
   DBIOPEN . . . . . . . . . . . .    1     0.9%
DBIPROP\
   DBIPROP . . . . . . . . . . . .    1     0.9%
  DBISIZES\
   DBISIZES . . . . . . . . . . .     1     0.9%
  DBIXYZ\
   DBIXYZ . . . . . . . . . . . .     1     0.9%
  DBSETUP\
   DBSETUP . . . . . . . . . . . .    1     0.9%
  DELTAS\
   DELTAS . . . . . . . . . . . .     1     0.9%
%PCA-I-CREFILE, creating file WPSTESTROOT:[000000.SF2D1.PCA]SF2D1_COVERAGE_SUM.TXT;1
PCA FOR PRESECOFL2D SUCCESSFULLY COMPLETED
```

INFORMATION ONLY

Figure 5-3
(Page 1 or 7)
PCA Coverage Analysis for Regression Testing and Coverage Test Case

```
DEC Performance and Coverage Analyzer                         Page 1
               Test Coverage Data (117 data points total) - "*"
                                       Data
  Bucket Name                          Count  Percent
   ADDMEM\
    ADDMEM                               1     0.9%
   DBIATTR\
    DBIATTR                              1     0.9%
   DBIOPEN\
    DBIOPEN                              1     0.9%
   DBIPROP\
    DBIPROP                              1     0.9%
   DBISIDES\
    DBISIDES                             1     0.9%
   DBIXYZ\
    DBIXYZ                               1     0.9%
   DBSETUP\
    DBSETUP                              1     0.9%
   DELTAS\
    DELTAS                               1     0.9%
   DYNMEM\
    DYNMEM                               1     0.9%
   EXCMDLIN\
    EXCMDLIN                             1     0.9%
   EXCPUS\
    EXCPUS                               1     0.9%
   EXDATE\
    EXDATE                               1     0.9%
   EXFOPEN\
    EXFOPEN                              1     0.9%
   EXMEMX\
    EXMEMX                               1     0.9%
   EXPARM\
    EXPARM                               1     0.9%
   EXTIME\
    EXTIME                               1     0.9%
   FFGLIN\
    FFGLIN                               1     0.9%
   FFGTKN\
    FFGTKN                               1     0.9%
   FFPARSE\
    FFPARSE                              1     0.9%
   FFRDFLDS\
    FFRDFLDS                             1     0.9%
   FFRDLINE\
    FFRDLINE                             1     0.9%
   FILCMDLIN\
    FILCMDLIN                            1     0.9%
   FILOPEN\
    FILOPEN                              1     0.9%
   GETBND\
GETBND                               1     0.9%
   GETLTS\
    GETLTS                               1     0.9%
   GETQALEVEL\
    GETQALEVEL                           1     0.9%
```

INFORMATION ONLY

Figure 5-3
(Page 2 or 7)
PCA Coverage Analysis for Regression Testing and Coverage Test Case

```
                DEC Performance and Coverage Analyzer                    Page 2
          Test Coverage Data (117 data points total) - ***
                               Data
Bucket Name                    Count  Percent
GETRTO\
  GETRTO . . . . . . . . . .     1     0.9%
GETRUN\
  GETRUN . . . . . . . . . .     1     0.9%
GLOC_WRITE\
  FIND_IDA_[illegible] . . .     1     0.9%
  GLOC_WRITE . . . . . . . .     1     0.9%
  GREG_GLOC . . . . . . . .      1     0.9%
  INTF_[illegible] . . . . .     1     0.9%
GREG_WRITE\
  ADRR_GREG . . . . . . . .      1     0.9%
  GREG_WRITE . . . . . . . .     1     0.9%
INIARR\
  INIARR . . . . . . . . . .     1     0.9%
ININAM\
  ININAM . . . . . . . . . .     1     0.9%
INISEC\
  INISEC . . . . . . . . . .     1     0.9%
ISTRLEN\
  ISTRLEN . . . . . . . . .      1     0.9%
IXLCHR\
  IXLCHR . . . . . . . . . .     1     0.9%
IXLNUM\
  IXLNUM . . . . . . . . . .     1     0.9%
MDINIT\
  MDINIT . . . . . . . . . .     1     0.9%
MEMADJ\
  MEMADJ . . . . . . . . . .     1     0.9%
PRADPR\
  PRADPR . . . . . . . . . .     1     0.9%
PREPRC\
  PREPRC . . . . . . . . . .     1     0.9%
PRESECOFL2D\
  PRESECOFL2D . . . . . . .      1     0.9%
PROCDB\
  PROCDB . . . . . . . . . .     1     0.9%
PROP_RANGE\
  PROP_RANGE . . . . . . . .     1     0.9%
PRSECO\
  PRSECO . . . . . . . . . .     1     0.9%
QABANNER\
  QABANNER . . . . . . . . .     1     0.9%
QABIGLET\
  QABIGLET . . . . . . . . .     1     0.9%
QACPUS\
  QACPUS . . . . . . . . . .     1     0.9%
QADOEDIS\
  QADOEDIS . . . . . . . . .     1     0.9%
QAMESSAG\
  QAMESSAG . . . . . . . . .     1     0.9%
QAPAGE\
  QAPAGE . . . . . . . . . .     1     0.9%
```

INFORMATION ONLY

Figure 5-3
(Page 3 or 7)
PCA Coverage Analysis for Regression Testing and Coverage Test Case

```
                 DEC Performance and Coverage Analyzer                 Page 3
         Test Coverage Data (117 data points total) - ***
                              Data
Bucket Name                   Count   Percent
CAPPTSTP\
  CAPPTSTP                        1     0.9%
CASETUP\
  CASETUP                         1     0.9%
RDBNDC\
  RDBNDC                          1     0.9%
RDCNST\
  RDCNST                          1     0.9%
RDINIT\
  RDINIT                          1     0.9%
RDLTIM\
  RDLTIM                          1     0.9%
RDNAME\
  RDNAME                          1     0.9%
RDRCLF\
  RDRCLF                          1     0.9%
RDRTIM\
  RDRTIM                          1     0.9%
RDSECO\
  RDSECO                          1     0.9%
SCANSI\
  SCANSI                          1     0.9%
SETCOORD\
  SETCOORD                        1     0.9%
SETGHO\
  SETGHO                          1     0.9%
SETHEA\
  SETHEA                          1     0.9%
STATUS\
  STATUS                          1     0.9%
STRCMPRS\
  STRCMPRS                        1     0.9%
STRLIMIT\
  STRLIMIT                        1     0.9%
STRPACK\
  STRPACK                         1     0.9%
STRUPCASE\
  STRUPCASE                       1     0.9%
WRTM2L\
  WRTM2L                          1     0.9%
WRTM2R\
  WRTM2R                          1     0.9%
WRTS2I\
  WRTS2I                          1     0.9%
WRTS2L\
  WRTS2L                          1     0.9%
WRTS2R\
  WRTS2R                          1     0.9%
WRTSEC\
  WRTSEC                          1     0.9%
XDBCNTOK\
  XDBCNTOK                        1     0.9%
```

INFORMATION ONLY

Figure 5-3
(Page 5 or 7)
PCA Coverage Analysis for Regression Testing and Coverage Test Case

```
                    DEC Performance and Coverage Analyzer                Page 5
              Test Coverage Data (117 data points total) - "*"
                                    Data
Bucket Name                        Count  Percent
XMELONG\
  XMELONG . . . . . . . . . . . . .   1     0.9%
XMELOCK\
  XMELOCK . . . . . . . . . . . . .   1     0.9%
XMINSRT\
  XMINSRT . . . . . . . . . . . . .   1     0.9%
XMORSRV\
  XMORSRV . . . . . . . . . . . . .   1     0.9%
XMSHFTG\
  XMSHFTG . . . . . . . . . . . . .   1     0.9%
XMSHFTI\
  XMSHFTI . . . . . . . . . . . . .   1     0.9%
XMSRCHC\
  XMSRCHC . . . . . . . . . . . . .   1     0.9%
XMSRCHI\
  XMSRCHI . . . . . . . . . . . . .   1     0.9%
XMTABLE\
  XMTABLE . . . . . . . . . . . . .   1     0.9%
XXRIGHT\
  XXRIGHT . . . . . . . . . . . . .   1     0.9%
XXUPPER\
  XXUPPER . . . . . . . . . . . . .   1     0.9%
DBISTEP\
  DBISTEP . . . . . . . . . . . . .   0     0.0%
DBITIME\
  DBITIME . . . . . . . . . . . . .   0     0.0%
DBIVAR\
  DBIVAR. . . . . . . . . . . . . .   0     0.0%
EXABORT\
  EXABORT . . . . . . . . . . . . .   0     0.0%
EXFLOAT\
  EXFLOAT . . . . . . . . . . . . .   0     0.0%
EXFILFLD\
  EXFILFLD. . . . . . . . . . . . .   0     0.0%
EXLOGICAL\
  EXLOGICAL . . . . . . . . . . . .   0     0.0%
EXNAME\
  EXNAME. . . . . . . . . . . . . .   0     0.0%
EXREAD\
  EXREAD. . . . . . . . . . . . . .   0     0.0%
FILDFNAM\
  FILDFNAM. . . . . . . . . . . . .   0     0.0%
FILPARSE\
  FILPARSE. . . . . . . . . . . . .   0     0.0%
GETSLV\
  GETSLV. . . . . . . . . . . . . .   0     0.0%
GREG_WRITE\
  FIND_IJA. . . . . . . . . . . . .   0     0.0%
  INTP. . . . . . . . . . . . . . .   0     0.0%
IQAERRUNI\
  IQAERRUNI . . . . . . . . . . . .   0     0.0%
```

INFORMATION ONLY

Figure 5-3
(Page 4 or 7)

PCA Coverage Analysis for Regression Testing and Coverage Test Case

```
                  DEC Performance and Coverage Analyzer                    Page 4
             Test Coverage Data ([illegible] data points total) - "**"
                         Data
Bucket Name              Count  %Percent
XDBEKATR
  XDBEKATR                  1      0.9%
XDBEXCNM
  XDBEXCNM                  1  [illegible]
XDBEXRRP
  XDBEXRRP                  1      0.9%
XDBIEVOR
  XDBIEVOR                  1      0.9%
XDBINIINT
  XDBINIINT                 1      0.9%
XDBISCAN
  XDBISCAN                  1      0.9%
XDBIXCUR
  XDBIXMBR                  1  [illegible]
XDBIXDBI
  XDBIXDBI                  1      0.9%
XDBIXNEW
  XDBIXNEW                  1      0.9%
XDBLOKNAM
  XDBLOKNAM                 1      0.9%
XDBMEMBR
  XDBMEMBR                  1      0.9%
XDBMEMRY
  XDBMEMRY                  1      0.9%
XDBMOVHD
  XDBMOVHD                  1      0.9%
XDBNAMEX
  XDBNAMEX                  1      0.9%
XDBRECHD
  XDBRECHD                  1  [illegible]
XDBREWRT
  XDBREWRT                  1      0.9%
XDBUNCNM
  XDBUNCNM                  1      0.9%
XDBXMOVE
  XDBXMOVE                  1      0.9%
XMCDEL
  XMCDEL                    1      0.9%
XMCFIND
  XMCFIND                   1      0.9%
XMCLONG
  XMCLONG                   1      0.9%
XMCNSRTV
  XMCNSRTV                  1      0.9%
XMCRSRV
  XMCRSRV                   1      0.9%
XMDEXEC
  XMDEXEC                   1      0.9%
XMDFIND
  XMDFIND                   1      0.9%
XMDGET
  XMDGET                    1      0.9%
```

INFORMATION ONLY

Figure 5-3
(Page 6 of 7)
PCA Coverage Analysis for Regression Testing and Coverage Test Case

```
                    DEC Performance and Coverage Analyzer                  Page
            Test Coverage Data (117 data points total) - ***
                                Data
Bucket Name                     Count  Percent
ISTRFIND\
 ISTRFIND                          0     0.0%
PRSDRC\
 PRSDRC                            0     0.0%
QAABORT\
 QAABORT                           0     0.0%
QABATCH\
 QABATCH                           0     0.0%
QAMESSAG\QAMESSAG\
 QNCEN                             0     [illegible]
RDAQFR\
 RDAQFR                            0     0.0%
RDSOLV\
 RDSOLV                            0     0.0%
RDWELL\
 RDWELL                            0     0.0%
SETCHR\
 SETCHR                            0     0.0%
SYSTEM$SERVICE\
 SYSTEM$SERVICE                    0     [illegible]
SYSTEM$SPACE\
 SYSTEM$SPACE                      0     [illegible]
WRTWEL\
 WRTWEL                            0     0.0%
XDBCHKHD\
 XDBCHKHD                          0     0.0%
XDBERR\
 XDBERR                            0     0.0%
XDBFNDIX\
 XDBFNDIX                          0     0.0%
XDBINIREA\
 XDBINIREA                         0     0.0%
XDBISTEP\
 XDBISTEP                          0     0.0%
XDBMOVST\
 XDBMOVST                          0     0.0%
XDBRECST\
 XDBRECST                          0     0.0%
XMCCOMP\
 XMCCOMP                           0     0.0%
XMCGET\
 XMCGET                            0     0.0%
XMCMEMY\
 XMCMEMY                           0     0.0%
XMCPRNT\
 XMCPRNT                           0     0.0%
XMDCOMP\
 XMDCOMP                           0     0.0%
XMDDEL\
 XMDDEL                            0     0.0%
XMDEROR\
 XMDEROR                           0     0.0%
```

INFORMATION ONLY

Figure 5-3
(Page 7 or 7)
PCA Coverage Analysis for Regression Testing and Coverage Test Case

```
                    DEC Performance and Coverage Analyzer                 Page [illegible]

                  Test Coverage Data (217 data points total) - "*"

                                   Data
 Bucket Name                       Count  Percent
 XMDGIVE
  XMDGIVE                              0    0.0%
 XMDLIST
  XMDLIST                              2    1.0%
 XMDPRNT
  XMDPRNT                              0    0.0%
 XXGET_XABDAT
  XXGET_XABDAT                         0    0.0%


                    DEC Performance and Coverage Analyzer                 Page [illegible]

                  Test Coverage Data (217 data points total) - "*"

                    PCA Version V4.[illegible]        [illegible]-SEP-1995 [illegible]

 TABULATE Command Summary Information:
 Number of buckets tallied:                                     [illegible]

 Test Coverage Data - "*"
 Number of covered buckets:                                     127   70.0%
 Number of acceptably not covered buckets:                        0    0.0%
 Number of remaining not covered buckets:                        42   25.0%
 Number of buckets with no coverage data:                         3    1.0%

 Data count in largest defined bucket:                            1    0.0%
 Data count in all defined buckets:                             217  100.0%
 Data count not in defined buckets:                               0    0.0%
 Portion of above count in P0 space:                              0    0.0%
 Number of P0 values in P1 space:                                 0    0.0%
 Number of P0 values in system space:                             0    0.0%

 Total number of data values collected:                         217  100.0%

 Command qualifiers and parameters used:
   Qualifiers:
     /COVERAGE /DESCENDING /NOMINIMUM /NOMAXIMUM
     /NOCUMULATIVE /NOSOURCE /ZEROS /NOSCALE /NOCREATOR_PC
     /NOPATHNAME /NOCHAIN_NAME /NOWRAP /NOPARENT_TASK /NOKEEP /NOTREE
     /FILL=("*","o","x","@","%","#","/","+")
     /NOSTACK_DEPTH /NOMAIN_IMAGE
   Node specifications:
     PROGRAM_ADDRESS BY ROUTINE

 No filters are defined
```

INFORMATION ONLY

Figure 5-4

(Page 1 of 7)

Cumulative PCA Coverage for Test Case

```
DEC Performance and Coverage Analyzer                    Page 1
              Test Coverage Data (117 data points total) - "*"
                         Data
 Pointer Name            Count   Percent
 ADDMEM\
  ADDMEM                   1      0.9%
 DBIATTR\
  DBIATTR                  1      0.9%
 DBIOPEN\
  DBIOPEN                  1      0.9%
 DBIPROP\
  DBIPROP                  1      0.9%
 DBISIZES\
  DBISIZES                 1      0.9%
 DBIXXD\
  DBIXXD                   1      0.9%
 DBSETUP\
  DBSETUP                  1      0.9%
 DELTAS\
  DELTAS                   1      0.9%
 DYNMEM\
  DYNMEM                   1      0.9%
 EXCMDLIN\
  EXCMDLIN                 1      0.9%
 EXCPUS\
  EXCPUS                   1      0.9%
 EXDATE\
  EXDATE                   1      0.9%
 EXFOPEN\
  EXFOPEN                  1      0.9%
 EXMEMY\
  EXMEMY                   1      0.9%
 EXPARM\
  EXPARM                   1      0.9%
 EXTIME\
  EXTIME                   1      0.9%
 FFDLIN\
  FFDLIN                   1      0.9%
 FFGTKN\
  FFGTKN                   1      0.9%
 FFPARSE\
  FFPARSE                  1      0.9%
 FFRDPLDS\
  FFRDPLDS                 1      0.9%
 FFRDLINE\
  FFRDLINE                 1      0.9%
 FILCMDLIN\
  FILCMDLIN                1      0.9%
 FILOPEN\
  FILOPEN                  1      0.9%
 GETBND\
  GETBND                   1      0.9%
 GETLTS\
  GETLTS                   1      0.9%
 GETQALEVEL\
  GETQALEVEL               1      0.9%
```

INFORMATION ONLY

Figure 5-4
(Page 2 of 7)
Cumulative PCA Coverage for Test Case

```
              DEC Performance and Coverage Analyzer                  Page 2
       Test Coverage Data (117 data points total) - "*"
                         Data
Bucket Name             Count  Percent
GETRTD\
  GETRTD                    1    0.9%
GETRUN\
  GETRUN                    1    0.9%
HOLD_WRITE\
  FIND_[illegible]          1    0.9%
  HOLD_WRITE                1    0.9%
  GREG_ULOC                 1    0.9%
  INTF_U                    1    0.9%
GREG_WRITE\
  ACER_GREG                 1    0.9%
  GREG_WRITE                1    0.9%
INIARR\
  INIARR                    1    0.9%
ININAM\
  ININAM                    1    0.9%
INIUSE\
  INIUSE                    1    0.9%
ISTRLEN\
  ISTRLEN                   1    0.9%
IXLCHR\
  IXLCHR                    1    0.9%
IXLNUM\
  IXLNUM                    1    0.9%
MDINIT\
  MDINIT                    1    0.9%
MEMADJ\
  MEMADJ                    1    0.9%
PRACRR\
  PRACRR                    1    0.9%
PREPRO\
  PREPRO                    1    0.9%
PRESECOFL2D\
  PRESECOFL2D               1    0.9%
PROCDB\
  PROCDB                    1    0.9%
PROP_RANGE\
  PROP_RANGE                1    0.9%
PRSECO\
  PRSECO                    1    0.9%
QABANNER\
  QABANNER                  1    0.9%
QABIGLET\
  QABIGLET                  1    0.9%
QACPUS\
  QACPUS                    1    0.9%
QADCEDIS\
  QADCEDIS                  1    0.9%
QAMESSAG\
  QAMESSAG                  1    0.9%
QAPAGE\
  QAPAGE                    1    0.9%
```

INFORMATION ONLY

Figure 5-4
(Page 3 of 7)
Cumulative PCA Coverage for Test Case

```
                    DEC Performance and Coverage Analyzer                  Page 3
              Test Coverage Data (117 data points total) - "*"
                               Data
Subpgm Name                    Count   Percent
LAPRTSTR\
  LAPRTSTR                         1     0.9%
CASETUP\
  CASETUP                          1     0.9%
RDBNDC\
  RDBNDC                           1     0.9%
RDCNST\
  RDCNST                           1     0.9%
RDINIT\
  RDINIT                           1     0.9%
RDLTIM\
  RDLTIM                           1     0.9%
RDNAME\
  RDNAME                           1     0.9%
RDRCLF\
  RDRCLF                           1     0.9%
RDRTIM\
  RDRTIM                           1     0.9%
RDSECO\
  RDSECO                           1     0.9%
SCANSI\
  SCANSI                           1     0.9%
SETCOORD\
  SETCOORD                         1     0.9%
SETGHO\
  SETGHO                           1     0.9%
SETHEA\
  SETHEA                           1     0.9%
STATUS\
  STATUS                           1     0.9%
STRCMPRS\
  STRCMPRS                         1     0.9%
STRLIMIT\
  STRLIMIT                         1     0.9%
STRPACK\
  STRPACK                          1     0.9%
STRUPCASE\
  STRUPCASE                        1     0.9%
WRTM2L\
  WRTM2L                           1     0.9%
WRTM2R\
  WRTM2R                           1     0.9%
WRTS2I\
  WRTS2I                           1     0.9%
WRTS2L\
  WRTS2L                           1     0.9%
WRTS2R\
  WRTS2R                           1     0.9%
WRTSEC\
  WRTSEC                           1     0.9%
XDBCNTOK\
  XDBCNTOK                         1     0.9%
```

INFORMATION ONLY

Figure 5-4
(Page 4 of 7)
Cumulative PCA Coverage for Test Case

```
                    DEC Performance and Coverage Analyzer                 Page 4
               Test Coverage Data (117 data points total) - "*"
                                 Data
Bucket Name                      Count   Percent
XDBEXATR\
 XDBEXATR                          1      0.9%
XDBEXDNM\
 XDBEXDNM                          1      0.9%
XDBEXPRP\
 XDBEXPRP                          1      0.9%
XDBIEVCK\
 XDBIEVCK                          1      0.9%
XDBINIINT\
 XDBINIINT                         1      0.9%
XDBISCAN\
 XDBISCAN                          1      0.9%
XDBIXCLR\
 XDBIXCLR                          1      0.9%
XDBIXDBI\
 XDBIXDBI                          1      0.9%
XDBIXNEW\
 XDBIXNEW                          1      0.9%
XDBLOCNAM\
 XDBLOCNAM                         1      0.9%
XDBMEMBR\
 XDBMEMBR                          1      0.9%
XDBMEMRY\
 XDBMEMRY                          1      0.9%
XDBMOVHD\
 XDBMOVHD                          1      0.9%
XDBNAMIX\
 XDBNAMIX                          1      0.9%
XDBRECHD\
 XDBRECHD                          1      0.9%
XDBRENWD\
 XDBRENWD                          1      0.9%
XDBUNINM\
 XDBUNINM                          1      0.9%
XDBXMOVE\
 XDBXMOVE                          1      0.9%
XMCDEL\
 XMCDEL                            1      0.9%
XMCFIND\
 XMCFIND                           1      0.9%
XMCLONG\
 XMCLONG                           1      0.9%
XMCNSRT\
 XMCNSRT                           1      0.9%
XMCRSRV\
 XMCRSRV                           1      0.9%
XMDEXEC\
 XMDEXEC                           1      0.9%
XMDFIND\
 XMDFIND                           1      0.9%
XMDGET\
 XMDGET                            1      0.9%
```

INFORMATION ONLY

Figure 5-4
(Page 5 of 7)
Cumulative PCA Coverage for Test Case

DEC Performance and Coverage Analyzer Page [illegible]
Test Coverage Data (117 data points total) - "**"

Bucket Name	Data Count	Percent
XMDLONG\		
XMDLONG	0	0.0%
XMDLOOK\		
XMDLOOK	0	0.0%
XMDNSRT\		
XMDNSRT	0	0.0%
XMDRSRV\		
XMDRSRV	0	0.0%
XMSHFTC\		
XMSHFTC	0	0.0%
XMSHFTI\		
XMSHFTI	0	0.0%
XMSRCHC\		
XMSRCHC	0	0.0%
XMSRCHI\		
XMSRCHI	0	0.0%
XMVTABLE\		
XMVTABLE	0	0.0%
XXRIGHT\		
XXRIGHT	0	0.0%
XXUPPER\		
XXUPPER	0	0.0%
DBISTEP\		
DBISTEP	0	0.0%
DBITIME\		
DBITIME	0	0.0%
DBIVAR\		
DBIVAR	0	0.0%
EXABORT\		
EXABORT	0	0.0%
ENFILDAT\		
ENFILDAT	0	0.0%
ENFILFLD\		
ENFILFLD	0	0.0%
EXLOGICAL\		
EXLOGICAL	0	0.0%
EXNAME\		
EXNAME	0	0.0%
EXREAD\		
EXREAD	0	0.0%
FILDFNAM\		
FILDFNAM	0	0.0%
FILPARSE\		
FILPARSE	0	0.0%
GETSLV\		
GETSLV	0	0.0%
GREG_WRITE\		
FIND_IJA	0	0.0%
INTF	0	0.0%
IQAERRUNI\		
IQAERRUNI	0	0.0%

INFORMATION ONLY

Figure 5-4
(Page 6 of 7)
Cumulative PCA Coverage for Test Case

DEC Performance and Coverage Analyzer Page 6
Test Coverage Data (117 data points total) - ***

Bucket Name	Data Count	Percent
QSTRBIND\		
QSTRBIND	0	0.0%
PRSORD\		
PRSORD	0	0.0%
QAABORT\		
QAABORT	0	0.0%
QABATCH\		
QABATCH	0	0.0%
QAMESSAG\QAMESSAG\		
MCEN		-
RDAQPR\		
RDAQPR	0	0.0%
RDSCLW\		
RDSCLW	0	0.0%
RDWELL\		
RDWELL	0	0.0%
SETCHR\		
SETCHR	0	0.0%
SYSTEM$SERVICE\		
SYSTEM$SERVICE	0	-
SYSTEM$SPACE\		
SYSTEM$SPACE	0	-
WRTWEL\		
WRTWEL	0	0.0%
XDBCHKHD\		
XDBCHKHD	0	0.0%
XDBERR\		
XDBERR	0	0.0%
XDBFNDIN\		
XDBFNDIN	0	0.0%
XDBINTREA\		
XDBINTREA	0	0.0%
XDBISTEP\		
XDBISTEP	0	0.0%
XDBMOVST\		
XDBMOVST	0	0.0%
XDBRECST\		
XDBRECST	0	0.0%
XMCCOMP\		
XMCCOMP	0	0.0%
XMCGET\		
XMCGET	0	0.0%
XMCMEMY\		
XMCMEMY	0	0.0%
XMCPRNT\		
XMCPRNT	0	0.0%
XMDCOMP\		
XMDCOMP	0	0.0%
XMDDEL\		
XMDDEL	0	0.0%
XMDEROR\		
XMDEROR	0	0.0%

INFORMATION ONLY

Figure 5-4
(Page 7 of 7)
Cumulative PCA Coverage for Test Case

```
                    DEC Performance and Coverage Analyzer                    Page 7
               Test Coverage Data (117 data points total) - "*"
                              Data
  Bucket Name                 Count  Percent
  XMDDIVE
    XMDDIVE                       1    [illegible]%
  XMDLIST
    XMDLIST                       0    [illegible]%
  XMDPRNT
    XMDPRNT                       0    [illegible]%
  XXGET_HASDAT
    XXGET_HASDAT                  [illegible]    [illegible]%
```

```
                    DEC Performance and Coverage Analyzer                    Page [illegible]

               Test Coverage Data (117 data points total) - "*"

                    PCA Version V4.3-4          8-SEP-1995 10:40:04

TABULATE Command Summary Information:
Number of buckets tallied:                                  162

Test Coverage Data - "*"
Number of covered buckets:                                  117    [illegible]%
Number of acceptably not covered buckets:                     0     0.0%
Number of remaining not covered buckets:                     42    25.9%
Number of buckets with no coverage data:                      3     [illegible]%

Data count in largest defined bucket:                         1     [illegible]%
Data count in all defined buckets:                          117   100.0%
Data count not in defined buckets:                            0     0.0%
Portion of above count in P0 space:                           0     0.0%
Number of PC values in P0 space:                              0     0.0%
Number of PC values in system space:                          0     0.0%

Total number of data values collected:                      117   100.0%

Command qualifiers and parameters used:
  Qualifiers:
    /COVERAGE /DESCENDING /NOMINIMUM /NOMAXIMUM
    /NOCUMULATIVE /NOSOURCE /ZEROS /NOSCALE /NOCREATOR_PC
    /NOPATHNAME /NOCHAIN_NAME /NOWRAP /NOPARENT_TASK /NOKEEP /NOTREE
    /FILL=("*","O","x","@","+","#","/","-")
    /NOSTACK_DEPTH /NOMAIN_IMAGE
  Node specifications:
    PROGRAM_ADDRESS BY ROUTINE

No filters are defined
```

INFORMATION ONLY

Figure 5-5
Code Sponsor's Response to Coverage Analysis

Most of the modules that show up in the performance coverage analysis as not being called are modules in the CAMCON libraries that the code is linked to. The following is a list of the non-CAMCON modules in PRESECOFL2D that are not called. The reason why each routine is not called is that feature is not used by the WIPP PA unless otherwise stated. The remaining uncalled modues are CAMCON modules

GETSLV - Prelimary set up of solving parameters. This routine is not called because the default settings are used by the WIPP PA.

FIND_IJA - Finds the location of a cell in the regional computational grid relative to the problem defining grid. For WIPP PA, these two grids are identical so routine is not called.

INTF - Interoplates the properties from the problem defining grid to the regional computational grid. For WIPP PA, these two grids are identical so this routine is not called.

PRSORC - Used for time dependent wells.

QAABORT - Used for error handling. If no errors occur, this routine will not be called.

QABATCH - Used only if code is run in CAMCON batch mode.

RDAQFR - Used to define precipitation, rivers, lakes and constant head cells. These are not used in WIPP PA.

RDSOLV - Used to set up solving parameters. The defaults are used for WIPP PA.

RDWELL - Sets up wells. Wells are not used in WIPP PA.

SETGHR - Used if databases are predefined with all properties.

WRTWEL - Writes well informtion to SECOFL2D input file. Wells are not used in WIPP PA.

INFORMATION ONLY

6.0 FUNCTIONAL TESTING

A list of requirements for POSTSECOTP2D can be found in **Figure 6-1.** PRESECOFL2D has already undergone QA and is qualified as level A under QAP 19-1 Rev F. Therefore, the test case consisted of running the current code for the purpose of coverage analysis and regression testing.

The test case was run in the directory WP$TESTROOT:[SF2D1.TESTCASES], and all files associated with the functional testing are located in this directory. To run the test case a script was used, and that script can be seen in **Figure 6-2.**

An evaluation of the test case can be found in Sections 6.1

INFORMATION ONLY

Figure 6-1
Requirements of PRESECOFL2D

Functional Requirements:

R.1 Creates all input files needed to run SECOFL2D. These files are:

1) An ASCII file containing run parameters

2) A binary file containing regional material property information obtained from a CAMDAT database.

3) A binary file containing local material property information.

R.2 Interpolates material properties from the regional grid to the local grid.

Performance Requirements:

This code has no performance requirements.

Attribute Requirements:

This code has no attribute requirements.

External Interface Requirements:

R.3 This code reads an ASCII input file containing run parameters.

R.4 This code reads an input regional CAMDAT database with grid information and material properties

R.5 This code reads an input local CAMDAT database with grid information..

R.6 This code produces all of the input files necessary to run SECOFL2D.

Additional Requirements:

There are no additional requirements for PRESECOFL2D that need verification.

INFORMATION ONLY

Figure 6-2
Script Used for Running Test Case for Functional Analysis

```
$ set noon
$!
$!==========================================================================
$!
$! This file runs all test cases for PRESECOFL2D using the NODEBUG executable
$!
$!==========================================================================
$!
$!
$ TESTDIR_SYM == "wp$testroot:[sf2d1.testcases]"
$ PRESECOFL2D == "$ wp$prodroot:[sf2d.exe]presecofl2d.exe"
$!
$ define /proc testdir wp$testroot:[sf2d1.testcases]
$!
$ show sym testdir_sym
$ IF F$MODE() .EQS. "BATCH" .AND. F$ENVIRONMENT("DEPTH") .EQ. 1 THEN -
$    SET DEFAULT 'TESTDIR_SYM'
$!
$!
$! Set up the first test case and run PRESECOFL2D
$!
$  WRITE SYS$OUTPUT "STARTING THE RUN FOR THE PRESECOFL2D TEST CASE"
$!
$  PRESECOFL2D SF2D1_REGION_TEST.CDB SF2D1_REGION_TEST.CDB -
     SF2D1_LOCAL_TEST.CDB CANCEL SF2D1_PRESECOFL_TEST.INP SECOFL2D_TEST.INP -
     SF2D1_REGION_TEST.PRP SF2D1_LOCAL_TEST.PRP CANCEL PRESECOFL_TEST.DBG
$!
$  write sys$output "SUCCESSFUL COMPLETION OF TEST EXECUTION"
$!==========================================================================
```

INFORMATION ONLY

6.1 Test Case #1: Regression Testing And Coverage Test Case

6.1.1 Test Objective

This test case was run to do the following:

1) test the interpolation of properties from the regional grid to the local boundary,
2) regression test the current code in the CMS so that it gets the same results obtained when the code was previously QA'd to the A level, and
3) assure modular coverage.

The issue of modular coverage has been addressed in both the use of DECset SCA and in the PCA Analysis.

6.1.2 Test Procedure

PART 1: (To test the interpolation of properties from the regional grid to the local boundary)

The pre-processor is executed followed by the execution of the program READ_PRP.FOR. This program converts the regional binary property file, REGION_TEST.PRP, to an ASCII file, REGION_TEST.ASC, and the local binary property file, LOCAL_TEST.PRP, to an ASCII file, LOCAL_TEST.ASC, so that they can be examined with an editor. All of the material properties used to run SECOFL2D are constants except for hydraulic conductivity. Therefore, checks of porosity (phi) and bulk compressibility (alpha) need only show that they have remained constant. A hand-calculation is done for hydraulic conductivity to show that the values have been correctly interpolated (using bilinear interpolation) from the regional grid to the local grid.

To verify that the porosity and bulk compressibility have remained constant, edit the file REGION_TEST.ASC. First, search for the string "alpha". Examination of this array shows that this is a constant value of 7.57E-01 for each element in the grid. (The value of bulk compressibility is 7.57E-10 on the database, but it has been scaled by 10^9 as required by SECOFL2D.) Next, search for the string "phi". Examination of this array shows that this is a constant value of 1.4542E-01 for each element in the grid. Now repeat this process with the file LOCAL_TEST.ASC. Searching for the string "alpha" shows that this array is a constant value of 7.57E-01, and searching for the string "phi" shows that this array is a constant of 1.4542E-01. The above mentioned searches were done and the constant values were verified. The first two rows of each 47x54 array are shown for completeness. **Figure 6.1-1** gives a listing of the testing tool READ_PRP.FOR used to convert the CDB files to ASCII files. The first two rows of the "alpha" array can be seen in **Figure 6.1-2** and the first two rows of the "phi" array can be seen in **Figure 6.1-3**. The data in both figures was extracted from the LOCAL_TEST.ASC.

Four points are checked by hand-calculation for the hydraulic conductivity. The four element locations in the local grid are (12,15), (20,47), (35,25) and (40,40). First, the cell locations relative to the regional grid are found. The cell locations in meters are converted to locations (in

INFORMATION ONLY

meters) in the regional domain. The x and y offsets of the local grid from the origin of the regional grid are 12826.1 and 10665.8 respectively. The angle of rotation is 38° counterclockwise. The coordinates (in meters) of the cell centers in the local grid are:

Element 1 (12,15): 1437.5, 1812.5
Element 2 (20,47): 2437.5, 5812.5
Element 3 (35,25): 4312.5, 3062.5
Element 4 (40,40): 4937.5, 4937.5

The regional coordinates are calculated by:

Element 1: $12826.1 + 1437.5\cos 38° - 1812.5\sin 38° = 12842.97903$
$10665.8 + 1812.5\cos 38° + 1437.5\sin 38° = 12979.08286$
Element 2: $12826.1 + 2437.5\cos 38° - 5812.5\sin 38° = 11168.34389$
$10665.8 + 5812.5\cos 38° + 2437.5\sin 38° = 16746.78735$
Element 3: $12826.1 + 4312.5\cos 38° - 3062.5\sin 38° = 14338.93311$
$10665.8 + 3062.5\cos 38° + 4312.5\sin 38° = 15734.12305$
Element 4: $12826.1 + 4937.5\cos 38° - 4937.5\sin 38° = 13677.07456$
$10665.8 + 4937.5\cos 38° + 4937.5\sin 38° = 17596.43163$

Next, the cell centers of the adjacent cells are used (the properties reside at the cell center) to calculate the proportional distances used in bilinear interpolation.

Element 1 proportions from coordinates of regional cells (32,10) and (33,11):
$xp = (12842.97903 - 12625.0)/(12850.0 - 12625.0) = 0.9687956889$
$yp = (12979.08286 - 12750.0)/(13125.0 - 12750.0) = 0.6108876267$
Element 2 proportions from coordinates of regional cells (24,34) and (25,35):
$xp = (11168.34389 - 11075.0)/(11225.0 - 11075.0) = 0.6222926$
$yp = (16746.78735 - 16725.0)/(16950.0 - 16725.0) = 0.0968326666$
Element 3 proportions from coordinates of regional cells (39,26) and (40,27):
$xp = (14338.93311 - 14150.0)/(14450.0 - 14150.0) = 0.6297770333$
$yp = (15734.12305 - 15600.0)/(15800.0 - 15600.0) = 0.67061525$
Element 4 proportions from coordinates of regional cells (37,38) and (38,39):
$xp = (13677.07456 - 13675.0)/(13887.5 - 13675.0) = 0.009762635294$
$yp = (17596.43163 - 17475.0)/(17630.0 - 17475.0) = 0.783429871$

Then, the bilinear interpolation is done.

Element 1:
$a1 = 498.5 + 0.9687956889(1498.0 - 498.5) = 1466.811291$
$a2 = 670.3 + 0.9687956889(707.4 - 670.3) = 706.2423201$
$a = 1466.811291 + 0.6108876267(706.2423201 - 1466.811291) = 1002.189117$

INFORMATION ONLY

Element 2:

a1 = 182.5 + 0.6222926(267.1 - 182.5) = 235.145954

a2 = 109.1 + 0.6222926(119.8 - 109.1) = 115.7585308

a = 235.145954 + 0.0968326666(115.7585308 - 235.145954) = 223.5853515

Element 3:

a1 = 495.8 + 0.6297770333(960.2 - 495.8) = 788.2684543

a2 = 816.1 + 0.6297770333(1041.0 - 816.1) = 957.7368548

a = 788.2684543 + 0.67061525(957.7368548 - 788.2684543) = 901.9165481

Element 4:

a1 = 159.6 + 0.009762635294(277.6 - 159.6) = 160.751991

a2 = 117.4 + 0.009762635294(28.47 - 117.4) = 116.5318088

a = 160.751991 + 0.783429871(116.5318088 - 160.751991) = 126.1085794

Acceptance Criteria :

The values for porosity and bulk compressibility remained constant. The values for hydraulic conductivity from LOCAL_TEST.ASC (interpolated from the regional grid) will agree with the hand-calculation to 6 decimal places.

	From file LOCAL_TEST.ASC		From hand-calculation
Element 1	(12,15)	1.00219E+03	1002.19
Element 2	(20,47)	2.23585E+02	233.585
Element 3	(35,25)	9.01917E+02	901.917
Element 4	(40,40)	1.26109E+02	126.109

Figure 6.1-4 shows rows 15, 25, 40, and 47 of the 47x54 array AKX in the LOCAL_ TEST.ASC file created. Visual inspection verifies that the results are in agreement with the hand calculations performed.

PART 2: (To show that the regression test of the current code in the CMS in fact gets the same results obtained when the code was previously QA'd to the A level)

Test Case 1 is suitable for installation and regression testing. For regression testing against the primitive baseline on file in the SWCF, this test case uses a utility program, TEST_READ_1. This program was used to produce the output on pages 4-6 of the verification section of the primitive baseline. Copies of these pages are included as **Appendix B**. This program should produce the identical results to the results in the primitive baseline when run during this testing exercise.

The input file for TEST_ READ_1.FOR is the SF2D1_REGION_TEST.PRP output file. A listing of TEST_READ_1 .FOR is shown in **Figure 6.1-5**, and the results from the current execution are in the file TEST_ READ.OUT as shown in **Figure 6.1-6.** This Program should

INFORMATION ONLY

produce the identical results to the results in the primitive baseline when run during this testing exercise. The input file for this test case is shown in **Figure 6.1-7.**

6.1.3 Input/Output Files

Since only a single run of the PRESECOFL2D was made to perform the functional analysis, the command file and the input files used is the same for both parts of the analysis. They are:

Command File:
SF2D1_TEST_NODBG_RUN.COM

Input Files:
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_REGION_TEST.CDB
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_LOCAL_TEST.CDB
WP$TESTROOT:[SF2D1.TESTCASES]SF2D1_PRESECOFL_TEST.INP

A listing of the input file SF2D1_PRESECOFL_TEST.INP is shown in **Figure 6.1-7.**

The input files SF2D1_REGION_TEST.CSB and SF2D1_LOCAL_TEST.CDB have not been included in this documents becuase they are not only in binary format but are also very large. The size of ASCII file of the contents would be prohibitive and would lend little information related to the testing. Both files will be submitted and retained in SCMS.

Each part of the functional analysis required the use of different output files which were then examined using short utility codes written for that purpose with resulting output files from that process. These files will be described for each part of the test case.

PART 1:

Output Files:
WP$TESTROOT:[SF2D1_REGION_TEST.PRP]
WP$TESTROOT:[SF2D1_LOCAL_TEST.PRP]

Testing Tool:
WP$TESTROOT:[SF2D1_READ_PRP.FOR]

Output Files from Testing Tool:
WP$TESTROOT:[SF2D1.TESTCASES]REGION_TEST.ASC
WP$TESTROOT:[SF2D.TESTCASES]LOCAL_TEST.ASC

PART 2:

Output Files:
WP$TESTROOT:[SF2D1_REGION_TEST.PRP

INFORMATION ONLY

Testing Tools:
WP$TESTROOT:[SF2D.TESTCASES]TEST_READ_1.FOR

Output Files from Testing Tools:
WP$TESTROOT:[SF2D.TESTCASES]TEST_READ.OUT

6.1.4 Evaluation

PART 1:

The acceptance for this part of the functional analysis was that once the alpha and phi values input were in fact constant, the output for four points on the local grid boundaries would have AKX values consistent with the conductivity values derived with hand calculations. This was in fact the case and the acceptance criteria has been met.

PART 2:

The acceptance for the regression testing of the current code required that the output from the current test case could be examined and the results would show no differences from the output file as noted in the primitive baseline. This was found to be the case and the acceptance criteria has therefore been met. However, only one file of the four generated by PRESECOFL2D was checked.

INFORMATION ONLY

Figure 6.1-1
(Page 1 of 4)
Listing of READ_PRP.FOR

```
PROGRAM READ_PRR
C
C       Reads binary property file (FOR020.DAT) written by SET_GREG2 and
C       translates it into an ASCII output file.
C       Prompts user for the input file name (i.e. FOR020.DAT) and the
C       user's choice of output file name.
C

        IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C       IMPLICIT NONE ******************** REMOVED

        dimension TMPARR(0:100,0:100), TMPVEC(0:100)
        INTEGER I, J, IDUM1, IDUM2, ISECO2, IL, JL, IDFIL_GREG
C       REAL RDUM1, RDUM2, RDUM3 *********** superceded
        CHARACTER*40 INFILE
        DATA ISECO2/10/
        DATA IDFIL_GREG/11/

C
C  BEGIN PROCEDURES...
C

        WRITE(6,*) 'Enter the name of the input file:'
        READ(5,'(A)') INFILE
        OPEN(UNIT=ISECO2,FILE=INFILE,STATUS='OLD',READONLY,
     +       FORM='UNFORMATTED')

        WRITE(6,*) 'Enter the name of the output file:'
        READ(5,'(A)') INFILE
        OPEN(UNIT=IDFIL_GREG,FILE=INFILE,STATUS='NEW')

   10 CONTINUE
          WRITE (IDFIL_GREG,9520)
 9520     FORMAT (' OUTPUT from program SET_GREG2 SET_BLOC2.',
     $            ' Numeric FORMATS are',
     $            ' 3(2I6, 1PE13.5)',
     $            ' 3(2I6, I10)/3x...',
     $            ' 3(I6, 1PE13.5)',
     $            ' SET_GREG2 run with i1GR, j1GR, GREG_ID,',
     $                              ' i1GR,j1GR, AQFR_ID = ')
          READ(ISECO2) IL, JL, RDUM1, IDUM1, IDUM2, RDUM2
          WRITE (IDFIL_GREG,953) i1, j1, RDUM1, i1, j1, RDUM2
  953 FORMAT (3(2I6, 1PE13.5))
          READ(ISECO2) RDUM1, RDUM2, RDUM3
          WRITE (IDFIL_GREG,956) RDUM1, RDUM2, RDUM3
  956 FORMAT (' x1_rel, y1_rel, theti_rel =',/,1P3E13.5)

c 1-D arrays

  955 FORMAT (3(I6, 1PE13.5))

  C       Read x-coordinates
READ(ISECO2) (TMPVEC(I),I=0,IL+1)
          WRITE (IDFIL_GREG,9540)
 9540     FORMAT (' (i,x(i), i=0, i1GR+1) ')
          WRITE (IDFIL_GREG,955)
     $    (i, TMPVEC(I), I=0,i1+1)

  C       Read y-coordinates

READ(ISECO2) (TMPVEC(I),I=0,JL+1)
```

INFORMATION ONLY

Figure 6.1-1
(Page 2 of 4)
Listing of READ_PRP.FOR

```
      WRITE (IDFIL_GREG,9541)
9541  FORMAT (' ((I,Y(I)), I=0, IIGR+1)')
      WRITE (IDFIL_GREG,955)
     $ (I, TMPVEC(I), I=0,IL+1)

C     Read delta x's
      READ(ISECOD) (TMPVEC(I),I=0,IL+1)
      WRITE (IDFIL_GREG,9542)
9542  FORMAT (' ((I, dX(I)), I=0, IIGR+1)')
      WRITE (IDFIL_GREG,955)
     $ (I, TMPVEC(I), I=0,IL+1)

C     Read delta y's
      READ(ISECOD) (TMPVEC(I),I=0,JL+1)
      WRITE (IDFIL_GREG,9543)
9543  FORMAT (' ((I, dY(I)), I=0, JIGR+1)')
      WRITE (IDFIL_GREG,955)
     $ (I, TMPVEC(I), I=0,JL+1)

C 2-D arrays

C     Read x relative offsets
      READ(ISECOD) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9529)
9529  FORMAT (' ((I,J,X_delPA(I,J)), I=0, IIGR+1), J=0, JIGR+1)')
      WRITE (IDFIL_GREG,953)
     $ ((I,J,TMPARR(I,J), I=0,IL+1), J=0,JL+1)

C     Read y relative offsets
      READ(ISECOD) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9529)
9529  FORMAT (' ((I,J,Y_delPA(I,J)), I=0, IIGR+1), J=0, JIGR+1)')
      WRITE (IDFIL_GREG,953)
     $ ((I,J,TMPARR(I,J), I=0,IL+1), J=0,JL+1)

C     Read x hydraulic conductivities
      READ(ISECOD) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,9531)
9531  FORMAT (' ((I,J,aKX(I,J)), I=0, IIGR+1), J=0, JIGR+1)')
      WRITE (IDFIL_GREG,953)
     $ ((I,J,TMPARR(I,J)), I=0,IL+1), J=0,JL+1)

C     Read y hydraulic conductivities
      READ(ISECOD) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1
WRITE (IDFIL_GREG,9532)
9532  FORMAT (' ((I,J,aKY(I,J)), I=0, IIGR+1), J=0, JIGR+1)')
      WRITE (IDFIL_GREG,953)
     $ ((I,J,TMPARR(I,J), I=0,IL+1), J=0,JL+1)

C     Read fluid compressibility
      READ(ISECOD) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,8532)
8532  FORMAT (' ((I,J,alpha(I,J)), I=0, IIGR+1), J=0, JIGR+1)')
      WRITE (IDFIL_GREG,953)
     $ ((I,J,TMPARR(I,J)), I=0,IL+1), J=0,JL+1)

C     Read porosity
      READ(ISECOD) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
      WRITE (IDFIL_GREG,8533)
8533  FORMAT (' ((I,J,phi(I,J)), I=0, IIGR+1), J=0, JIGR+1)')
      WRITE (IDFIL_GREG,953)
     $ ((I,J,TMPARR(I,J)), I=0,IL+1), J=0,JL+1)
```

INFORMATION ONLY

Figure 6.1-1
(Page 3 of 4)
Listing of READ_PRP.FOR

```
C       Read specific storage
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9533)
 9533   FORMAT (' ((i,j),spec_stor(i,j), i=0, iIGR+1), j=0, jIGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read aquifer storativity
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9534)
 9534   FORMAT (' ((i,j),aq_stor(i,j), i=0, iIGR+1), j=0, jIGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read aquifer thickness
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9535)
 9535   FORMAT (' ((i,j),aq_thick(i,j), i=0, iIGR+1), j=0, jIGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read aquifer bottom
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9635)
 9635   FORMAT (' ((i,j),aq_bot(i,j), i=0, iIGR+1), j=0, jIGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read well river recharge term
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9536)
 9536   FORMAT (' ((i,j),wlr_rch(i,j), i=0, iIGR+1), j=0, jIGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read river conductivity
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9537)
 9537   FORMAT (' ((i,j),riv_con(i,j), i=0, iIGR+1), j=0, jIGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read river head
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9637)
 9637   FORMAT (' ((i,j),riv_head(i,j), i=0, iIGR+1), j=0, jIGR+1) ')
        WRITE (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il+1), J=0,jl+1)

C       Read river bottom
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9737)
 9737   FORMAT (' ((i,j),riv_bot(i,j), i=0, iIGR+1), j=0, jIGR-1) ')
        WRITS (IDFIL_GREG,953)
     $  ((i,j),TMPARR(I,J), I=0,il-1), J=0,jl-1)

C       Read constant head
        READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
        WRITE (IDFIL_GREG,9538)
 9538   FORMAT (' ((i,j),head(i,j), i=0, iIGR+1), j=0, jIGR-1) ')
        WRITE (IDFIL_GREG,953)
```

INFORMATION ONLY

Figure 6.1-1
(Page 4 of 4)
Listing of READ_PRP.FOR

```
     2      1,3),TMPARR(I,3),(I=7,II+1),J=3),I=1
STOP 'READ_PRP - NORMAL COMPLETION'
```

INFORMATION ONLY

Figure 6.1-2
Listing of Sample of Alpha Values from LOCAL_TEST.ASC

```
OUTPUT from program SET_GREG2/SET_GLOC2; Numeric FORMATS are
3(2I4, 1PE13.5)
[illegible]
[illegible] 1PE13.5
SET_GREG2 run with [illegible]
[illegible]
[illegible]
[illegible]
[illegible] alpha [illegible]
```

[illegible]	0	7.57000E-01	1	0	7.57000E-01	2	0	7.57000E-01
3	0	7.57000E-01	4	0	7.57000E-01	5	0	7.57000E-01
[illegible]	0	7.57000E-01	7	0	7.57000E-01	8	0	7.57000E-01
[illegible]	0	7.57000E-01	10	0	7.57000E-01	11	0	7.57000E-01
[illegible]	0	7.57000E-01	13	0	7.57000E-01	14	0	7.57000E-01
[illegible]	0	7.57000E-01	[illegible]	0	7.57000E-01	[illegible]	0	7.57000E-01
[illegible]	0	7.57000E-01	[illegible]	0	7.57000E-01	[illegible]	0	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	0	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	0	7.57000E-01	38	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	40	[illegible]	7.57000E-01	41	0	7.57000E-01
[illegible]	[illegible]	7.57000E-01	43	0	7.57000E-01	44	0	7.57000E-01
[illegible]	0	7.57000E-01	46	0	7.57000E-01	47	0	7.57000E-01
[illegible]	1	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	1	7.57000E-01
[illegible]	1	7.57000E-01	4	[illegible]	7.57000E-01	5	1	7.57000E-01
[illegible]	1	7.57000E-01	[illegible]	1	7.57000E-01	8	1	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	20	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	1	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01
[illegible]	[illegible]	7.57000E-01	[illegible]	[illegible]	7.57000E-01	41	[illegible]	7.57000E-01
[illegible]	1	7.57000E-01	43	[illegible]	7.57000E-01	44	[illegible]	7.57000E-01
45	1	7.57000E-01	46	1	7.57000E-01	47	1	7.57000E-01

INFORMATION ONLY

Figure 6.1-2
Listing of Sample of Phi Values from LOCAL_TEST.ASC

```
OUTPUT from program SET_GRESC/SET_GLOC2. Numeric FORMATS are
3(I5, 1PE13.5)
3(I5, I10, 3X)
I5, 1PE13.5

SET_GRESC run with I1GR, J1GR, GRES_ID, I2GR, J2GR, NOFFL_IP = [illegible]
          53      1.02[illegible]E+00      46      53      1.00[illegible]E+0[illegible]
xi_rel, yi_rel, enest_del =
 [illegible]E-04  1.[illegible]E-04  [illegible]E-01

 i, j, phi (i, j), i=0, I1GR-1 , j=0, J1GR-2

    0    0  1.45420E-01     1    0  1.45420E-01     2    0  1.45420E-01
    3    0  1.45420E-01     4    0  1.45420E-01     5    0  1.45420E-01
    6    0  1.45420E-01     7    0  1.45420E-01     8    0  1.45420E-01
    9    0  1.45420E-01    10    0  1.45420E-01    11    0  1.45420E-01
   12    0  1.45420E-01    13    0  1.45420E-01    14    0  1.45420E-01
   15    0  1.45420E-01    16    0  1.45420E-01    17    0  1.45420E-01
   18    0  1.45420E-01    19    0  1.45420E-01    20    0  1.45420E-01
   21    0  1.45420E-01    22    0  1.45420E-01    23    0  1.45420E-01
   24    0  1.45420E-01    25    0  1.45420E-01    26    0  1.45420E-01
   27    0  1.45420E-01    28    0  1.45420E-01    29    0  1.45420E-01
   30    0  1.45420E-01    31    0  1.45420E-01    32    0  1.45420E-01
   33    0  1.45420E-01    34    0  1.45420E-01    35    0  1.45420E-01
   36    0  1.45420E-01    37    0  1.45420E-01    38    0  1.45420E-01
   39    0  1.45420E-01    40    0  1.45420E-01    41    0  1.45420E-01
   42    0  1.45420E-01    43    0  1.45420E-01    44    0  1.45420E-01
   45    0  1.45420E-01    46    0  1.45420E-01    47    0  1.45420E-01
    0    1  1.45420E-01     1    1  1.45420E-01     2    1  1.45420E-01
    3    1  1.45420E-01     4    1  1.45420E-01     5    1  1.45420E-01
    6    1  1.45420E-01     7    1  1.45420E-01     8    1  1.45420E-01
    9    1  1.45420E-01    10    1  1.45420E-01    11    1  1.45420E-01
   12    1  1.45420E-01    13    1  1.45420E-01    14    1  1.45420E-01
   15    1  1.45420E-01    16    1  1.45420E-01    17    1  1.45420E-01
   18    1  1.45420E-01    19    1  1.45420E-01    20    1  1.45420E-01
   21    1  1.45420E-01    22    1  1.45420E-01    23    1  1.45420E-01
   24    1  1.45420E-01    25    1  1.45420E-01    26    1  1.45420E-01
   27    1  1.45420E-01    28    1  1.45420E-01    29    1  1.45420E-01
   30    1  1.45420E-01    31    1  1.45420E-01    32    1  1.45420E-01
   33    1  1.45420E-01    34    1  1.45420E-01    35    1  1.45420E-01
   36    1  1.45420E-01    37    1  1.45420E-01    38    1  1.45420E-01
   39    1  1.45420E-01    40    1  1.45420E-01    41    1  1.45420E-01
   42    1  1.45420E-01    43    1  1.45420E-01    44    1  1.45420E-01
   45    1  1.45420E-01    46    1  1.45420E-01    47    1  1.45420E-01
```

INFORMATION ONLY

Figure 6.1-4

(Page 1 of 2)

Listing of Four Rows of AKX Values from LOCAL_TEST.ASC

```
OUTPUT from program SET_GREG2/SET_GLCC2. Numeric FORMATS are
3(2I6, 1PE13.5)
3(2I6, I10, 3X)
3(2I6, 1PE13.5)

SET_GREG2 run with I1GR, J1GR, GREG_II, I2GR, J2GR, AQFR_ID = 46
     53     1.02500E+03     46     53     1.03200E+03
[illegible]
 [illegible]E+04  1.06658E+04  8.80000E-01

(I,J,AKX(I,J), I=0, I1GR-1), J=0, J1GR-1)
```

0	15	7.88209E+02	1	15	7.88209E+02	2	15	8.22071E+02
3	15	6.97871E+02	4	15	6.65739E+02	5	15	[illegible].82761E+02
6	15	6.86631E+02	7	15	6.76572E+02	8	15	5.85476E+02
9	15	4.89826E+02	10	15	5.08133E+02	11	15	9.71625E+02
12	15	1.00219E+03	13	15	1.21268E+03	14	15	9.81879E+02
15	15	9.17651E+02	16	15	9.43169E+02	17	15	[illegible]
18	15	8.89611E+02	19	15	1.39271E+03	20	15	1.43019E+03
21	15	1.66952E+03	22	15	1.65894E+03	23	15	2.52093E+03
24	15	2.00912E+03	25	15	2.45405E+03	26	15	3.54598E+03
27	15	4.69265E+03	28	15	4.90334E+03	29	15	5.93452E+03
30	15	[illegible].23415E+03	31	15	3.88024E+03	32	15	1.82248E+03
33	15	1.43233E+03	34	15	1.09775E+03	35	15	1.17273E+03
36	15	1.33142E+03	37	15	1.30263E+03	38	15	1.18623E+03
39	15	1.20826E+03	40	15	1.49001E+03	41	15	2.33443E+03
42	15	3.06554E+03	43	15	3.33965E+03	44	15	2.41774E+03
45	15	1.35261E+03	46	15	6.16012E+02	47	15	6.16012E+02
0	25	4.93984E+01	1	25	4.93984E+01	2	25	7.09193E+01
3	25	9.34402E+01	4	25	1.10084E+02	5	25	1.35790E+02
6	25	1.29749E+02	7	25	7.57993E+01	8	25	9.46498E+02
9	25	1.25587E+02	10	25	9.71722E+01	11	25	4.98033E+01
12	25	6.59934E+01	13	25	3.20847E+01	14	25	1.70723E+01
15	25	2.04374E+01	16	25	1.46455E+01	17	25	3.21417E+01
18	25	3.49823E+01	19	25	4.62050E+01	20	25	6.83027E+01
21	25	1.03795E+02	22	25	1.36228E+02	23	25	1.36527E+02
24	25	1.22610E+02	25	25	1.23558E+02	26	25	2.34743E+02
27	25	1.90493E+02	28	25	2.88393E+02	29	25	5.69473E+02
30	25	8.59090E+02	31	25	8.46066E+02	32	25	7.47447E+02
33	25	5.50005E+02	34	25	7.06704E+02	35	25	9.01917E+02
36	25	1.01916E+03	37	25	1.02838E+03	38	25	1.21958E+03
39	25	1.92738E+03	40	25	2.18692E+03	41	25	3.69868E+03
42	25	6.94503E+03	43	25	8.38390E+03	44	25	4.03626E+03
45	25	8.14540E+03	46	25	2.97991E+02	47	25	2.97991E+02
0	40	1.20090E+02	1	40	1.20090E+02	2	40	9.51753E+01
3	40	7.62124E+01	4	40	7.72883E+01	5	40	9.60407E+01
6	40	8.25067E+01	7	40	7.21038E+01	8	40	7.41663E+01
9	40	5.19377E+01	10	40	4.94694E+01	11	40	8.09754E+01
12	40	1.48998E+02	13	40	2.04487E+02	14	40	2.83987E+02
15	40	4.47821E+02	16	40	7.31904E+02	17	40	1.10687E+03
18	40	1.58719E+03	19	40	1.75489E+03	20	40	1.32736E+03
21	40	7.02391E+02	22	40	6.07775E+02	23	40	6.95539E+02
24	40	1.11353E+03	25	40	1.50873E+03	26	40	2.38533E+03
27	40	3.53025E+03	28	40	3.03359E+03	29	40	2.20789E+03
30	40	1.55116E+03	31	40	9.62770E+02	32	40	6.19297E+02
33	40	4.76053E+02	34	40	3.68529E+02	35	40	3.83787E+02
36	40	2.68856E+02	37	40	2.88196E+02	38	40	2.76505E+02
39	40	2.04067E+02	40	40	1.26109E+02	41	40	5.06560E+01
42	40	1.41854E+01	43	40	1.38939E+00	44	40	1.14872E+00
45	40	9.05840E-01	46	40	8.78583E-01	47	40	8.78583E-01

Figure 6.1-4

(Page 2 of 2)

Listing of Four Rows of AKX Values from LOCAL_TEST.ASC

0	47	3.79724E+03	1	47	3.74724E+03	2	47	1.52646E+03
3	47	[illegible].41845E+03	4	47	1.97576E+02	5	47	1.57835E+02
6	47	1.[illegible]E+02	7	47	[illegible].96329E+01	8	47	8.22553E+01
9	47	[illegible]E+01	10	47	9.45110E+01	11	47	[illegible].22032E+01
12	47	[illegible].23348E+01	13	47	5.62578E+01	14	47	5.16998E+01
15	47	[illegible].49328E+0[illegible]	16	47	1.07125E+02	17	47	1.02600E+02
18	47	[illegible]E+02	19	47	[illegible].12022E+02	20	47	2.03585E+02
21	47	[illegible].55630E+02	22	47	3.15973E+02	23	47	[illegible].96429E+02
24	47	1.57234E+02	25	47	2.21443E+02	26	47	[illegible].52229E+02
27	47	9.03528E+01	28	47	7.18963E+01	29	47	5.66314E+01
30	47	8.68542E+01	31	47	9.38033E+01	32	47	1.04812E+02
33	47	1.42465E+02	34	47	1.63751E+02	35	47	1.46651E+02
36	47	[illegible].15297E+02	37	47	1.47466E+02	38	47	1.73553E+02
39	47	1.62115E+02	40	47	1.19609E+02	41	47	4.60231E+01
42	47	[illegible].05721E+00	43	47	2.44703E+00	44	47	7.93126E-01
45	47	6.08122E-01	46	47	3.60993E-01	47	47	3.60999E-01

INFORMATION ONLY

Figure 6.1-5

(Page 1 of 3)

Listing of SF2D1_TEST_READ_1.FOR

```
PROGRAM TEST_READ_1
C.....6-9-94 ELLEN S. DOMBROSKI
C.....THIS PROGRAM WAS WRITTEN FOR THE SINGLE PURPOSE OF
C.....VERIFYING BINARY OUTPUT FROM SECOFL2D.
C.....SPECIFIED RANGES OF SPECIFIED ARRAYS ARE PRINTED TO THE
C.....OUTPUT FILE FOR VERIFICATION BY INSPECTION.
C
C     Reads binary property file (FOR011.DAT) written by SET_PROC and
C     translates it into an ASCII output file.
C     Prompts user for the input file name (i.e. FOR010.DAT) and the
C     user's choice of output file names.
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      dimension TMPARR(0:100,0:100)
      CHARACTER*50 INFILE, OUTFILE
      DATA ISECO2/10/
      DATA IDFIL_GREG/20/

C
C.....BEGIN PROCEDURES...
C
      EPR = 1.D-20

      WRITE(6,*) 'Enter the name of the input file:'
      READ(5,'(A)') INFILE
      OPEN(UNIT=ISECO2,FILE=INFILE,STATUS='OLD',READONLY,
     +     FORM='UNFORMATTED')

      WRITE(6,*) 'Enter the name of the output file:'
      READ(5,'(A)') OUTFILE
      OPEN(UNIT=IDFIL_GREG,FILE=OUTFILE,STATUS='NEW')

      WRITE (IDFIL_GREG,9520) INFILE
 9520 FORMAT (' ASCII OUTPUT FROM FILE ',A,
     &  ' THIS FILE CONTAINS SPECIFIC RANGES THAT WILL BE USED',
     &  ' FOR VERIFICATION  6-9-94')

  950   FORMAT (3(2I5,1PE13.5))
  951   FORMAT (2I6, 1PE15.5)

C.....READ(ISECO2) IL, JL, RDUM1, IDUM1, IDUM2, RDUM2
      READ(ISECO2) IL, JL
C.....READ(ISECO2) RDUM1, RDUM2, RDUM3
      READ(ISECO2) RDUM1, RDUM2, RDUM3

C.....1-D arrays
C.....Read x-coordinates
      READ(ISECO2)
C.....Read y-coordinates
      READ(ISECO2)
C.....Read delta x's
      READ(ISECO2)
C.....Read delta y's
      READ(ISECO2)
C.....2-D arrays
C.....Read x relative offsets
      READ(ISECO2)
C.....Read y relative offsets
      READ(ISECO2)
C
C.....READ AND ECHO THESE FOR TEST. 6-9-94
```

INFORMATION ONLY

Figure 6.1-5
(Page 2 of 3)
Listing of SF2D1_TEST_READ_1.FOR

```
C.....Read x hydraulic conductivities
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,100)
  100 FORMAT (' ((i,j,aKx(i,j)), i= 1,10), j= 1, 1)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=1,10), J=1,1)
C.....
      WRITE (IDFIL_GREG,102)
  102 FORMAT (' ((i,j,aKx(i,j)), i=21, 30), j=28,28)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=21,30), J=28,28)
C.....
      WRITE (IDFIL_GREG,104)
  104 FORMAT (' ((i,j,aKx(i,j)), i=41, 50), j=57,57)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=41,50), J=57,57)
C.....
C.....READ AND ECHO THESE FOR TEST, 6-9-94
C.....Read y hydraulic conductivities
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,110)
  110 FORMAT (' ((i,j,aKy(i,j)), i= 1,10), j= 1, 1)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=1,10), J=1,1)
C.....
      WRITE (IDFIL_GREG,112)
  112 FORMAT (' ((i,j,aKy(i,j)), i=21, 30), j=28,28)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=21,30), J=28,28)
C.....
      WRITE (IDFIL_GREG,114)
  114 FORMAT (' ((i,j,aKy(i,j)), i=41, 50), j=57,57)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=41,50), J=57,57)
C.....
C.....READ AND ECHO THESE FOR TEST, 6-9-94
C.....Read fluid compressibility
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,120)
  120 FORMAT (' ((i,j,alpha(i,j)), i= 1,10), j= 1, 1)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=1,10), J=1,1)
C.....
      WRITE (IDFIL_GREG,122)
  122 FORMAT (' ((i,j,alpha(i,j)), i=21, 30), j=28,28)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=21,30), J=28,28)
C.....
      WRITE (IDFIL_GREG,124)
  124 FORMAT (' ((i,j,alpha(i,j)), i=41, 50), j=57,57)')
      WRITE (IDFIL_GREG,953)
     $((i,j,TMPARR(I,J)*SCF, I=41,50), J=57,57)
C.....
C.....READ AND ECHO THESE FOR TEST, 6-9-94
C.....Read porosity
      READ(ISECO2) ((TMPARR(I,J),I=0,IL+1),J=0,JL+1)
C.....
      WRITE (IDFIL_GREG,130)
  130 FORMAT (' ((i,j,phi(i,j)), i= 1,10), j= 1, 1)')
```

INFORMATION ONLY

Figure 6.1-5

(Page 3 of 3)

Listing of SF2D1_TEST_READ_1.FOR

```
      WRITE (IDFIL_GREG,253)
     $((I,J,TMPARR(I,J)), I=1,10), J=1,1)
C
      WRITE (IDFIL_GREG,132)
  132 FORMAT ([illegible]I2,'),phi [illegible], I=11,30), J=28,28)[illegible]
      WRITE (IDFIL_GREG,253)
     $((I,J,TMPARR(I,J)), I=11,30), J=28,28)

      WRITE (IDFIL_GREG,134)
  134 FORMAT ([illegible]I2,'),phi [illegible], I=41,50), J=57,57)[illegible]
      WRITE (IDFIL_GREG,253)
     $((I,J,TMPARR(I,J)), I=41,50), J=57,57)
C
C.....Read specific storage
C     READ(ISECO1)
C.....Read aquifer storativity
C     READ(ISECO1)
C.....Read aquifer thickness
C     READ(ISECO1)
C.....Read aquifer bottom
C     READ(ISECO1)
C.....Read well river recharge term
C     READ(ISECO1)
C.....Read river conductivity
C     READ(ISECO1)
C.....Read river head
C     READ(ISECO1)
C.....Read river bottom
C     READ(ISECO1)
C.....Read constant head
C     READ(ISECO1)
C
      STOP 'TEST_READ_1: NORMAL COMPLETION'
      END
```

INFORMATION ONLY

Figure 6.1-6
(Page 1 of 3)
Listing of SF2D1_TEST_READ.OUT

```
ASCII OUTPUT FROM FILE SF2D1_REGION_TEST.PRP
THIS FILE CONTAINS SPECIFIC RANGES THAT WILL BE USED FOR VERIFICATION. [illegible]
((i,j,aKx(i,j), i=1, 10), j=1,1)
     1     1    1.89500E-04
     2     1    1.04700E-04
     3     1    1.45600E-05
     4     1    2.67800E-05
     5     1    2.38800E-05
     6     1    1.87400E-05
     7     1    2.04300E-05
     8     1    2.96700E-05
     9     1    1.68900E-05
    10     1    7.59600E-06
((i,j,aKx(i,j), i=21, 30), j=28,28)
    21    28    1.50900E-07
    22    28    1.32000E-07
    23    28    1.42400E-07
    24    28    1.63900E-07
    25    28    4.32500E-07
    26    28    1.86500E-06
    27    28    1.87000E-06
    28    28    1.15800E-06
    29    28    1.30800E-06
    30    28    1.49000E-06
((i,j,aKx(i,j), i=41, 50), j=57,57)
    41    57    4.20400E-10
    42    57    1.85600E-10
    43    57    2.02000E-10
    44    57    2.59200E-10
    45    57    2.68900E-10
    46    57    1.62500E-10
    47    57    2.15900E-10
    48    57    2.51900E-10
    49    57    1.78000E-10
    50    57    1.50900E-09
((i,j,aKy(i,j), i=1, 10), j=1,1)
     1     1    1.89500E-04
     2     1    1.04700E-04
     3     1    1.45600E-05
     4     1    2.67800E-05
     5     1    2.38800E-05
     6     1    1.87400E-05
     7     1    2.04300E-05
     8     1    2.96700E-05
     9     1    1.68900E-05
    10     1    7.59600E-06
((i,j,aKy(i,j), i=21, 30), j=28,28)
    21    28    1.50900E-07
    22    28    1.32000E-07
    23    28    1.42400E-07
    24    28    1.63900E-07
    25    28    4.32500E-07
    26    28    1.86500E-06
    27    28    1.87000E-06
    28    28    1.15800E-06
    29    28    1.30800E-06
    30    28    1.49000E-06
((i,j,aKy(i,j), i=41, 50), j=57,57)
    41    57    4.20400E-10
    42    57    1.85600E-10
    43    57    2.02000E-10
    44    57    2.59200E-10
```

INFORMATION ONLY

Figure 6.1-6

(Page 2 of 3)

Listing of SF2D1_TEST_READ.OUT

```
    45     57    [illegible]
    46     57    [illegible]
    47     57    [illegible]
    48     57    [illegible]
    49     57    [illegible]
    50     57    [illegible]
 ((i,j,alpha(i,j), i= 1,10), j= 1, 1)
     1      1    7.57000E-10
     2      1    7.57000E-10
     3      1    7.57000E-10
     4      1    7.57000E-10
     5      1    7.57000E-10
     6      1    7.57000E-10
     7      1    7.57000E-10
     8      1    7.57000E-10
     9      1    7.57000E-10
    10      1    7.57000E-10
 ((i,j,alpha(i,j), i=21, 30), j=28,28)
    21     28    7.57000E-10
    22     28    7.57000E-10
    23     28    7.57000E-10
    24     28    7.57000E-10
    25     28    7.57000E-10
    26     28    7.57000E-10
    27     28    7.57000E-10
    28     28    7.57000E-10
    29     28    7.57000E-10
    30     28    7.57000E-10
 ((i,j,alpha(i,j), i=41, 50), j=57,57)
    41     57    7.57000E-10
    42     57    7.57000E-10
    43     57    7.57000E-10
    44     57    7.57000E-10
    45     57    7.57000E-10
    46     57    7.57000E-10
    47     57    7.57000E-10
    48     57    7.57000E-10
    49     57    7.57000E-10
    50     57    7.57000E-10
 ((i,j,phi(i,j), i= 1,10), j= 1, 1)
     1      1    1.45420E-01
     2      1    1.45420E-01
     3      1    1.45420E-01
     4      1    1.45420E-01
     5      1    1.45420E-01
     6      1    1.45420E-01
     7      1    1.45420E-01
     8      1    1.45420E-01
     9      1    1.45420E-01
    10      1    1.45420E-01
 ((i,j,phi(i,j), i=21, 30), j=28,28)
    21     28    1.45420E-01
    22     28    1.45420E-01
    23     28    1.45420E-01
    24     28    1.45420E-01
    25     28    1.45420E-01
    26     28    1.45420E-01
    27     28    1.45420E-01
    28     28    1.45420E-01
    29     28    1.45420E-01
    30     28    1.45420E-01
 ((i,j,phi(i,j), i=41, 50), j=57,57)
```

INFORMATION ONLY

Figure 6.1-6
(Page 3 of 3)
Listing of SF2D1_TEST_READ.OUT

```
441    57    1.45403E-[illegible]
442    57    1.45403E-[illegible]
443    57    1.45422E-[illegible]
444    57    1.45422E-[illegible]
445    57    1.45422E-[illegible]
446    57    1.45422E-[illegible]
447    57    1.45422E-[illegible]
448    57    1.45422E-[illegible]
449    57    1.45422E-[illegible]
450    57    1.45422E-[illegible]
```

INFORMATION ONLY

Figure 6.1-7

SF2D1_PRESECOFL_TEST.INP

```
!------------------------------------------------------------------------------
! File - NFS1:[CAMTON.PRESECOFL2D.TEST_PROBLEM]PRESECOFL2D_TEST.INP
! Test case for SECOFL2D
!------------------------------------------------------------------------------
!
*RUN_TYPE
    GET_ACFR, TYPE=DEF, ACFR_ID=1.001
    GET_GREG, TYPE=DEF, GREG_ID=1.001
    GET_GLOC, TYPE=DEF, GLOC_ID=1.005, X_REL=12826.2, Y_REL=12665.8, THETA=38.0
    FLOW, mode=transient, LOC_BOUND=yes
*constant
    screen_io=BRIEF
*INITIAL_CONDITIONS
    CAMDAT, HEAD=HEADat, TYPE=ATTR
    RESET
*ATTR
 HYCND_Y=HYCND_X
 POROSITY=POROSITY
 DEPTH=THICK
*REG_TIME
    SEC, NUM_STEP=10, START=0.0, FINISH=40000.
    AUTO, OPTION=DEL_TIME
*LOCAL_TIME
    SEC, NUM_STEP=10, START=0.0, FINISH=10000.
    AUTO, OPTION=DEL_TIME
*BOUNDARY
    REG, BOUNDARY=LOWER, TYPE=HEAD, END=25000., clim_frac=0.
    REG, BOUNDARY=UPPER, TYPE=HEAD, END=10000., clim_frac=1.
    Reg, BOUNDARY=UPPER, TYPE=HEAD, END=17500., clim_frac=0.
    REG, BOUNDARY=UPPER, TYPE=GRAD, END=25000., clim_frac=0.
    REG, BOUNDARY=RIGHT, TYPE=HEAD, END=27240., clim_frac=0.
    REG, BOUNDARY=RIGHT, TYPE=GRAD, END=30000., clim_frac=0.
    REG, BOUNDARY=LEFT, TYPE=HEAD, END=40000., clim_frac=0.
    REG, BOUNDARY=LEFT, TYPE=GRAD, END=18595., clim_frac=0.
    REG, BOUNDARY=LEFT, TYPE=HEAD, END=30000., clim_frac=1.
    LOCAL, BOUNDARY=LOWER, TYPE=HEAD, END=5750.
    LOCAL, BOUNDARY=UPPER, TYPE=HEAD, END=5750.
    LOCAL, BOUNDARY=RIGHT, TYPE=GRAD, END=6625.
    LOCAL, BOUNDARY=LEFT, TYPE=HEAD, END=6625.
*REG_CLIMATE
    LAKE, AMP=0.0, CYCLES=1.0, START=0.0, FINISH=25000., RATIO=1.0
    RECHARGE, AMP=0, CYCLES=1., START=0.0, FINISH=10000.
    HEAD, AMP=amplitud, CYCLES=cycles, head_fac=climtidx, &
            START=0.0, FINISH=40000.
*END
```

INFORMATION ONLY

7.0 RELATED TESTING DOCUMENTATION

Table 7-1
Documentation Related to the Testing of PRESECOFL2D

Test Item	Description	SCMS Filename
Test Log	Issues and events that arose during the testing of PRESECOFL2D version 4.02ZO	SF2D1_TESTLOG.TXT
FLINT Output	Output generated as a result of running the source code analyzer, FORTRAN-Lint	SF2D1_FLINT.OUT
Scripts used	Command files used to build the executables, and the run the PCA analysis and test cases	SF2D1_BUILD.COM SF2D1.MMS SF2D1_BUILD.LOG SF2D1_TESTCASE_PCA_CUM.COM SF2D1_TEST_NODBG_RUN.COM
SCA Output	Output generated as a result of running the source code analyzer, DECset-SCA	SF2D1_SCA_MOD_NOT_REF.TXT SF2D1_CALLTREE.TXT SF2D1_MODULES.OUT
PCA Output	Output generated as a result of running the program coverage analyzer, DECset-PCA	SF2D1_PCA.LOG SF2D1_TEST_COVER.TXT SF2D1_COVERAGE_CUM.TXT SF2D1_COVERAGE.TXT
Input Files	Input files required to perform both the PCA analysis and the functional testing.	SF2D1_PRESECOFL.INP SF2D1_REGION_TEST.CDB SF2D1_LOCAL_TEST.CDB
Testing Tools	Source codes of utility programs used during testing	SF2D1_READ_PRP.FOR SF2D1_TEST_READ_1.FOR
Input Files for Testing Tools	Input files on which testing tools operate	SF2D1_REGION_TEST.PRP SF2D1_LOCAL_TEST.PRP
Output Files from Testing Tools	Output files resulting from running the testing tools.	SF2D1_REGION_TEST.ASC SF2D1_LOCAL_TEST.ASC SF2D1_TEST_READ.OUT
Requirements Document	A list of the requirements for PRESECOFL2D	N/A
Verification and Validation Plan	A description of the testing strategy for PRESECOFL2D	N/A
Implementation Document	Source code listing and a description of the executable generation	N/A
User's Guide	Instruction for using PRESECOFL2D	N/A

INFORMATION ONLY

8.0 CONCLUSION

The testing for PRESECOFL2D as prescribed in the Verification and Validation Plan has been completed and all acceptance criteria have been satisfied. The test case provided in Section 6 provide complete coverage of all functional requirements as described in Section 6.

9.0 REFERENCES

1) WIPP-PA - Qualification Guide for Pre-existing Software, Version 2.0, dated 8/24/95.

2) WIPP-PA - Implementation Document for PRESECOFL2D Version 4.02ZO, WPO# 23296

3) WIPP-PA - Requirements Document for PRESECOFL2D Version 4.02ZO WPO# 23318

4) Rechard, R.P., ed. 1992, User's Manual for CAMCON: Compliance Assessment Methodology Controller Version 3.0. SAND 90-1983, Albuquerque, NM: Sandia National Laboratories.

6) WIPP-PA - User's Manual for PRESECOFL2D Version 4.02ZO, WPO# 23297

INFORMATION ONLY

10.0 APPENDICES

APPENDICES A - Reviewer's Forms

See attached

INFORMATION ONLY

WASTE ISOLATION PILOT PLANT Sandia National Laboratories	**Validation Document Reviewer's Form**
	Form Number: 452 Effective: 7/31/95
	Procedure: 19-1 Revision: 1 Page 1 of 1

WPO Number: 23319   Software Classification: SNL-SW

Reviewer Instructions:

Check the "Yes" box for each item reviewed and found acceptable.
Check the "No" box for each item which requires further work.
Check the "N/A" box for items that are not applicable.
Check the "N/R" box for items not reviewed (only if there are multiple reviewers).
For multiple reviewers, each reviewer shall complete a Validation Document Reviewer's Form.
Prior to sign-off of the Validation Document, all "No" items shall be appropriately addressed by the code sponsor so that "Yes" or "N/A" may be checked, or a memo from the DM shall be attached to the Validation Document explaining the reason for the non-conformance.
This form shall be included as part of the baseline Validation Document.

INFORMATION ONLY

Item	Yes	No	N/A	N/R
1. **Test Documentation Completeness** Are all tests identified in the VVP documented, including significant inputs and outputs, and the hardware and software configurations used to run the test cases?	☒	☐	☐	☐
2. **Test Result Validation** Are the test results compared to at least one of the following? - hand calculations, - calculations using comparable proven problems, - empirical data from published sources and/or technical literature, or - other validated software of similar purpose.	☒	☐	☐	☐
3. **Peer Review** If test cases did not provide sufficient validation, was a documented peer review performed?	☐	☐	☒	☐
4. **Test Documentation Acceptability** Do the tests meet the acceptance criteria identified in the VVP?	☒	☐	☐	☐
5. **Test Documentation Repeatability** Are the tests documented in sufficient detail such that they can be repeated?	☒	☐	☐	☐
6. **Computer File Documentation** Are the test case input and output files included in the Validation Document?	☒	☐	☐	☐
7. **Understandability of Documentation** Are the validation methods, test data, results, and conclusions documented in a form that can be understood by an independent, technically competent individual?	☒	☐	☐	☐

C. David Updegraff	C. David Updegraff	11 Oct 95
Reviewer Name (printed)	Signature	Date

SWCF File Code: 1.1.6.9

WASTE ISOLATION PILOT PLANT Sandia National Laboratories	# Review Comment (Software) Form Number: 299 Effective: 7/31/95
	Procedure: QAP 19-1 Revision: 1 Page 1 of 2

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA :SFT (Alpha Code)

Keywords: *(Used for unique identification)*
PRESECOFL2D , SF2D1 ,

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECOFL2D	4.02	5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☐ Technical ☒ Document

Validation Document, pg 5 Section 1.2
The "code consultant" information needs to be supplied.

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☒ Agree ☐ Disagree

Done

INFORMATION ONLY

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Review Comment (Software)

Form Number: 299

Effective: 7/31/95

Procedure: QAP 19-1 Revision: 1 Page 1 of 2

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA :SFT (Alpha Code)

Keywords: *(Used for unique identification)* PRESECOFL2D SF2D1

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECOFL2D	4.02	5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

Validation Document, page 58, PART 2, 1st paragraph Figure 6.1-7 referred to in the paragraph is actually a PRESECOFL2D input file. Copies of the comparison pages are ~~atall~~ actually in Appendix B. Figure 6.1-7 is never referenced properly. (Note: The top of page 59 repeats correctly the bottom of page 58).

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☒ Agree ☐ Disagree

This has been fixed.

INFORMATION ONLY

SWCF File Code: 1.1.6.9 (WBS#)

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Review Comment (Software)

Form Number: 299 Effective: 7/31/95

Procedure: QAP 19-1 Revision: 1 Page 1 of 2

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA :SFT (Alpha Code)

Keywords: *(Used for unique identification)*

PRESECOFL2D , SF201

INFORMATION ONLY

SECTION I

1. Software Name: PRESECOFL2D	Version ID: 4.02	Version Date: 5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

Validation Document, pg 60, Section 6.1.4, Part 2
PRESECOFL2D generates ~~only~~ 4 output files. However only part of 1 was checked against the ~~baseline~~ corresponding baseline file. Explain the reasoning for checking only part of one output file.

SECTION II

5. Comment Resolution: *(check one; attach pages as needed: describe impact to previous uses of code, if applicable)* ☐ Agree ☐ Disagree

The only file used in the baseline was the regional property file. Since then, the R.2 requirement was established to verify the local property file. Only [illegible] of the file is used as it is a large file and only spot checking is necessary.

SWCF File Code: 1.1.6.9 (WBS#)

WASTE ISOLATION PILOT PLANT

Sandia National Laboratories

Review Comment (Software)

Form Number: 299

Effective: 7/31/95

Procedure: QAP 19-1 Revision: 1 Page 1 of 2

File Code: SWCF- A 1.1.6.9 WBS #(s) PA :SET Alpha Code

Keywords: *(Used for unique identification)*

PRESECOFL2D, SF2D1

INFORMATION ONLY

SECTION I

1. Software Name:	Version ID:	Version Date:
PRESECOFL2D	4.02	5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

Validation Document, pg 12, Table 4.1
The SCA code analyzer should generate a file called SF2DI-VAR-NOT-REF.TXT. Explain why the file does not show in Table 4.1.

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☒ Agree ☐ Disagree

~~The~~ An older version of the SCA process produced this file. The current version does not. The RD/VP did not reflect the current version. That has been changed.

SWCF File Code: 1.1.6.9 WBS#

WASTE ISOLATION PILOT PLANT Sandia National Laboratories	**Review Comment (Software)** Form Number: 299 Effective: 7/31/95

Procedure: QAP 19-1 Revision: 1 Page 1 of 2

File Code: SWCF- A 1.1.6.9 (WBS #(s)) PA SFT (Alpha Code)

Keywords: *(Used for unique identification)*
PRESECOFL20, SF201

SECTION I

1. Software Name: PRESECOFL20	Version ID: 4.02	Version Date: 5/20/94

2. Type of Comment: *(check one)* ☒ Required ☐ Suggested ☐ Error

Software Type: *(check one)* ☐ Commercial ☒ Developed ☐ System

4. Comment: *(check one, attach pages as needed)* ☒ Technical ☐ Document

Validation Document
Figure 6-1

Requirement R.6 should be eliminated. It is the same as R.1.

SECTION II

5. Comment Resolution: *(check one; attach pages as needed; describe impact to previous uses of code, if applicable)* ☐ Agree ☒ Disagree

I believe this requirement belongs in both categories. I feel there is nothing ~~[illegible]~~ mutually exclusive about the categories. It is a functional requirement and an external interface requirement, both.

INFORMATION ONLY

SWCF File Code: 1.1.6.9 (WBS#)

APPENDIX B: PRIMITIVE BASELINE DOCUMENT

The following pages from the verification section primitive baseline documentation currently stored in SWCF as a part of the prior QA efforts. The basis of the current regression testing is to duplicated the results from prior testing. This was done by reproducing the results found in the baseline documentation. This reproduction is found in **Figure 6.1-2.**

FigureB-1
(Page 1 of 3)
Pages 4-6 of the Verification Section of the Primitive Baseline


```
ASCII OUTPUT FROM FILE dfsl:: [damcon.presecofl2d.test]regdat_test.inp
THIS FILE- CONTAINS SPECIFIC RANGES THAT WILL BE USED FOR VERIFICATION, 5-9-94.
((i, j, aKx (i, j) , i= 1, 10) , j= 1, 1)
  1    1   1.89500E-04
  2    1   2.04700E-04
  3    1   3.45600E-05
  4    1   2.67800E-05
  5    1   2.38800E-05
  6    1   1.87400E-05
  7    1   2.04300E-05
  8    1   2.96700E-05
  9    1   1.68900E-05
 10    1   7.59600E-06
((i,j,aKx (i,j), i=21, 30), j=28,28)
 21   28   1.50900E-07
 22   28   1.32000E-07
 23   28   1.42400E-07
 24   28   1.63900E-07
 25   28   4.32500E-07
 26   28   1.86500E-06
 27   28   1.87000E-06
 28   28   1.15800E-06
 29   28   1.30800E-06
 30   28   1.49000E-06
((i,j,aKx (i, j), i=41, 50 , j=57, 57)
 41   57   4.20400E-10
 42   57   1.85600E-10
 43   57   2.02000E-10
 44   57   2.59200E-10
 45   57   2.66900E-10
 46   57   1.62500E-10
 47   57   2.15800E-10
 48   57   1.51900E-11
 49   57   1.78000E-10
 50   57   2.50900E-09
((i,j,aKy(i, j) , i=1, 10), j=1, 1)
  1    1   1.89500E-04
  2    1   2.04700E-04
  3    1   3.45600-=-05
  4    1   2.67800F--05
  5    1   2.38800E-05
  6    1   1.87400E-05
  7    1   2.04300E-05
  8    1   2.96700E-05
  9    1   1.68900E-05
 10    1   7.59600E-06
((i,j,aKy (i,j)i=21 , 30) ,j=28, 28)
 21   28   1.50900E-07
 22   28   1.32000E-07
 23   28   1.42400E-07
```

INFORMATION ONLY

FigureB-1

(Page 2 of 3)

Pages 4-6 of the Verification Section of the Primitive Baseline


```
 24    28   1.63900E-07
 25    28   4.32500E-07
 26    28   1.86510E-06
 27    28   1.87110E-06
 28    28   1.15800E-06
 29    28   1.31000E-06
 30    28   1.47100E-06
((i, j, Ky (i, j), i=41, 50), j=57, 57)
 41    57   4.01400E-10
 42    57   1.85600E-10
 43    57   2.02000E-10
 44    57   2.59200E-10
 45    57   2.66900E-10
 46    57   1.62500E-10
 47    57   2.15600E-10
 48    57   2.51900E-10
 49    57   1.78000E-10
 50    57   2.51000E-10
((i, j, alpha (i, j), i=1, 10), j=1, 1)
  1     1   7.57000E-10
  2     1   7.57000E-10
  3     1   7.57000E-10
  4     1   7.57000E-10
  5     1   7.57000E-10
  6     1   7.57000E-10
  7     1   7.57000E-10
  8     1   7.57000E-10
  9     1   7.57000E-10
 10     1   7.57000E-10
((i, j, alpha (i, j), i=21, 30), j=28, 28)
 21    28   7.57000E-10
 22    28   7.57000E-10
 23    28   7.57000E-10
 24    28   7.57000E-10
 25    28   7.57000E-10
 26    28   7.57000E-10
 27    28   7.57000E-10
 28    28   7.57000E-10
 29    28   7.57000E-10
 30    28   7.57000E-10
((i, j, alpha (i, j), i=41, 50), j=57, 57)
 41    57   7.57000E-10
 42    57   7.57000E-10
 43    57   7.57000E-10
 44    57   7.57000E-10
 45    57   7.57000E-10
 46    57   7.57000E-10
 47    57   7.57000E-10
 48    57   7.57000E-10
 49    57   7.57000E-10
 50    57   7.57000E-10
((i, j, phi (i, j), i=1, 10), j=1, 1)
  1     1   1.45420E-01
  2     1   1.45420E-01
  3     1   1.45420E-01
  4     1   1.45420E-01
  5     1   1.45420E-01
  6     1   1.45420E-01
  7     1   1.45420E-01
  8     1   1.45420E-01
  9     1   1.45420E-01
 10     1   1.45420E-01
```

INFORMATION ONLY

FigureB-1
(Page 3 of 3)

Pages 4-6 of the Verification Section of the Primitive Baseline


```
 1, 7, phi  1, 3  , i=21, 30 , j=28, 28
21    28   1.45420E-01
22    28   1.45420E-01
23    28   1.45420E-01
24    28   1.45420E-01
25    28   1.45420E-01
26    28   1.45420E-01
27    28   1.45420E-01
28    28   1.45420E-01
29    28   1.45420E-01
30    28   1.45420E-01
 1, 7, phi  1, 5  , i=41, 50 , j=57,57
41    57   1.45420E-01
42    57   1.45420E-01
43    57   1.45420E-01
44    57   1.45420E-01
45    57   1.45420E-01
46    57   1.45420E-01
47    57   1.45420E-01
48    57   1.45420E-01
49    57   1.45420E-01
50    57   1.45420E-01
```

INFORMATION ONLY

EPA Comment
Enclosure 2, Pages 3-4
194.23(a)(3)(iv)

Comment Text

Section 194.23(a)(3)(iv) requires "computer models accurately implement the numerical models" and are free of coding errors and produce stable results.

Appendix PAR identifies the assigned values for both longitudinal and transverse dispersivity in the Culebra as 0.0. Although this value would appear to lead to conservative results by reducing the amount of surface area available for matrix diffusion, there is insufficient evidence presented in the CCA that the SECOTP code will provide stable solutions at such low dispersivities. In fact, in a letter from James McCord to James Ramsey (Sandia National Lab), provided as an attachment to the Parameter Record Package for non-Salado longitudinal dispersivity, Dr. McCord states "Assuming that the numerical codes used correctly solve the governing partial differential equations, simulations using local dispersivities less than or equal to 2 m will yield results consistent with field scale dispersive spreading observations as reported by Gelhar et al. (1992)."

The Department needs to provide evidence that the numerical solver method implemented in the SECOTP code correctly solves the partial differential equations at the dispersivities of 0.0 over the range of Courant numbers used in the CCA.

DOE Response

Numerical approximations of advection-dominate transport problems may result in spatial oscillations near regions where the concentration gradients are steep. These oscillations (commonly referred to as undershoot and overshoot) are caused by the inability of second order numerical approximations to accurately propagate short wavelength harmonics. The phenomenon becomes more apparent when the local Peclet number exceeds a value of 10 because the short wavelength harmonics become increasingly important (Huyakorn and Pinder, 1983, p. 206). The Peclet number, which is inversely proportional to the dispersion coefficient, is quite large when the dispersivities are set equal to zero.

In most transport simulators this phenomenon is addressed by enhancing spatial discretization and/or introducing numerical dispersion by reducing the order of the numerical approximation (upwind or upstream weighting discretization schemes).

Fully-weighted upwind schemes do not have undershoot and overshoot problems, but numerical dispersion is often quite large and the resulting solution may not be accurate. To avoid oscillatory behavior limit numerical dispersion, a total variation diminishing (TVD) flux limiter scheme is invoked in SECOTP2D. The underlying concept is the application of upwinding techniques where needed to the degree necessary to prevent non-physical oscillations. A discussion of TVD including references is found on page 7 of the *WIPP PA User's Manual for SECOTP2D, Version 1.30* (Appendix SECOTP2D of the CCA).

In the CCA, zero dispersivity coefficients were used in both the longitudinal and transverse directions. The dispersion coefficients are extremely small, but non-zero, due to contributions from the molecular diffusion component. Because the SECOTP2D results did not exhibit oscillatory behavior, the CCA runs are in and of themselves a test of the stability of SECOTP2D and demonstrate that zero dispersivities are not a problem for SECOTP2D.

Reference:

Huyakorn, P.S., and G.F. Pinder, 1983. *Computation Methods in Subsurface Flow*, Academic Press, Inc., New York, NY. (Textbook)

EPA Commen
Enclosure 2, I
194.23(a)(3)(i

55

Comment Tex

194.23(a)(3)(iv)
Section 194.23(a)(3)(iv) requires "computer models accurately implement the numerical models."

In regard to the BRAGFLO computer code, Appendix MASS states "Approximating convergent and divergent flow around the intrusion borehole and the shaft creates two narrow necks in the otherwise fairly uniform width grid in the region representing the repository. In the undisturbed performance scenario and under certain conditions in other scenarios, flow in the repository may pass laterally through these necks. In reality, these necks do not exist. Their presence in the model is expected to have a negligible or conservative impact on model predictions compared to predictions that would result from use of a more realistic model geometry." The text further states that "The time scale involved and the permeability contrast between the repository and surrounding rock are sufficient that lateral flow that may occur in the repository is restricted by the rate at which liquid gets into or out of the repository, rather than the rate at which it flows through the repository." To support this contention, a grid study comparing a two-dimensional and three-dimensional model was performed and included as MASS Attachment 4-1. The results of this analysis indicate that under undisturbed performance the grids would provide similar answers. However, the models were parameterized such that, in both cases, brine did not flow up the borehole following an intrusion and therefore, the adequacy of the grid under disturbed conditions cannot be evaluated.

The Department needs to provide a similar analysis that is representative of an intrusion scenario in which brine reaches the Culebra. That is, the pressures in the repository have to be high enough so brine from the repository reaches the Culebra.

DOE Response:

The modeling study documented in Appendix MASS, Attachment MASS 4-1, demonstrates that a 2D grid set up with radial flaring is adequate in approximating the behavior of the 3D system. The critical performance measure is not what in particular happened (e.g., brine flow reaching the Culebra), but that all representative measures of fluxes be comparable in both models. As demonstrated in the plots at the back of MASS Attachment 4-1, this is the case. Note that in addition to examining the effects of a 2D representation of the 3D system, the study documented in MASS

Attachment 4-1 also favorably benchmarked the BRAGFLO code against the TOUGH2 code, which uses quite different numerical approaches in solving similar multiphase flow equations and in defining WIPP-specific processes.

The calculations performed were appropriate for the determination because the geometries are comparable to those used in the CCA and the parameter values used in the codes are similar to the median values used in the CCA performance assessment (at the time the simulation was executed, the values used were median values in the SNL WIPP performance assessment database). To select a case in which brine flow occurred up the borehole, more extreme values than the medians would have to be selected for some variables. As indicated in the DOE response to the EPA comment on grid geometry, page 8 of enclosure 1, 194.23(a)(3)(iv), part (2), the use of median values for model geometry comparison studies has been recommended as appropriate by EPA representatives.

EPA Comment
Enclosure 2, page 4
194.23(c)(2)

Comment Text

194.23(c)(2)
Section 194.23(c)(2) requires, among other things, "...reports on code verification, benchmarking, validation, and quality assurance procedures."

The Requirements Document and the Verification and Validation Plan for the NUTS computer code establishes the criterion that "the integrated sum of releases passing any point of interest should be less than the integrated release from the repository." However, this does not prove that mass is being conserved, nor is evidence of mass balance provided elsewhere in the documentation.

The Department needs to perform a mass balance analysis on the NUTS computer code.

DOE Response

Inspection of the material balance variables associated with the ASCII outputs provided in the Requirements Document and the Verification and Validation Plan (RD/VVP) for all test cases, except Test case #5, show that material is conserved. Results of Test case #5 were reported in Computational Data Base (CDB) files rather than ASCII files, and therefore are not readily available for inspection of material balance. Results of the four other test cases, as reported in the RD/VVP, are sufficient to demonstrate mass balance.

All the ASCII output files reported in the RD/VVP for the NUTS computer code contain the Material Balance Error in terms of the variables mentioned in the response to Item (5) in the comment in Enclosure 1, page 9. However, these variables are not reported in the CDB files.

EPA Comment
Enclosure 2, Pages 5-6
194.23(c)(2)

Comment Text

194.23(c)(2)
Section 194.23(c)(2) requires, among other things, "...reports on code verification, benchmarking, validation, and quality assurance procedures."

The GRASP_INV computer code user's manual describes a number of test problem computer runs. However, none of the test runs is similar to the way in which the code is implemented in the performance assessment. It is also never stated in the documentation that the GRASP-INV code has been tested in a manner in which it will be implemented in the performance assessment.

The Department needs to provide evidence that the GRASP-INV code was tested in a manner in which it will be implemented in the performance assessment, and provide a sample computer run that corresponds to the CCA results.

DOE Response:

The test runs for GRASP-INV were conducted in a manner similar to that used for CCA PA code implementation. The documentation which describes test problem 16 for the *GRASP-INV, Version 2.01, Version Date 8/21/95, Requirements Document and Verification & Validation Plan* and *GRASP-INV, Version 2.01, Version Date 8/21/95, Validation Document* describe the specifications and results of a test problem which is functionally similar to the CCA PA application of GRASP-INV. Test problem 16 contains the 1992 Culebra model grid but uses a two categorical transmissivity classification. The test problem includes steady-state calibration to the 1992-selected steady-state heads as well as a demonstration of the functionality of the transient-calibration process. Thus, only minor differences exist between Test problem 16 and the CCA application of GRASP-INV. Specifically these differences are:

1) the CCA PA model used different boundary conditions,
2) the CCA PA model used different categorical and continuous variable variograms,
3) the CCA PA model used a finer finite-difference grid,
4) the CCA PA model used qualified data, and

5) the CCA PA model assigned weights to the observed steady-state head data.

With the exception of number 5 above, the functionality of the code tested in Test Problem 16 was identical to the application of the code during the CCA PA. In response to number 5, the end-to-end test problem being executed in response to the request in Enclosure 2, page 5, first 194.23(c)(2) citation, addresses this concern.

EPA Comment
Enclosure 2, page 9
194.34(c)

Comment Text

194.34(c)
Section 194.34(c) requires documentation of computational techniques used in generating complementary, cumulative distribution functions.

Although the general approach to sampling of parameters is described briefly in Chapter 6, the User's Manual for Latin Hypercube Sampling (LHS), and Appendix PAR, no detailed discussion of the LHS procedure is included. The User's Manual contains a brief discussion of the advantages of this approach, but it does not clearly describe the implementation of the method.

The Department needs to provide a detailed discussion of the LHS procedure and its implementation.

DOE Response

As the comment notes, Latin Hypercube sampling is described in Chapter 6, the User's Manual for Latin Hypercube Sampling, and Appendix PAR. Additional information about the theory and application of LHS is described in references cited in those portions of the CCA, as appropriate.

The attached text provides additional background information on the use of LHS. Details of the application of LHS in the CCA follow.

For the purposes of creating samples for the CCA performance assessment analysis, the LHS code and its preprocessor, PRELHS, were run in the production mode by the CCA_MASTERs using their execution scripts. PRELHS was run to extract distribution information from the INGRES database and write it out to a file that served as the input file for LHS. At that point, the official execution scripts called the LHS code in order to create the actual samples. Three official samples were created, one for each of the three sample replicates described in the CCA documentation. At that point, the sampled information was available for use by client codes, either in its existing form or after postprocessing. The LHS sample files are not saved into the CMS after creation and are deleted. This is done because the sample files are easy to recreate if needed from PRELHS input files and INGRES database views, which were saved in the electronic archives.

Description of the steps LHS goes through to create a sample

Following is a listing of the steps performed by the LHS code as it creates a sample. It is meant to show the order of how things are accomplished in the LHS process.

1. First, coding is invoked to open and name the input file, output file, and various scratch files used by LHS.

2. Then, a subroutine is called to read the LHS input file containing the information extracted from the INGRES database for each distribution to be sampled. Further information is read defining execution flags and parameters for the upcoming run, along with any specified correlation among distributions.

3. Next, subroutines are called to write a banner to the output file followed by an echoed table of the distribution information and execution flags specified on the input file.

4. Following that, if the user has specified a correlation structure, then the correlation matrix is echoed and checked to make sure that it is positive definite. The Cholesky factorization is computed to be used later as part of the process for inducing the desired correlation structure.

5. At this point, subroutines are called to sample the distributions identified in the input file. For the case of the 57 distributions of interest in the LHS sample created for the CCA performance assessment, each distribution was fully evaluated in the order it appears in the input file, independent of any of the other distributions.

6. With the sampled values in hand, the next task is to call subroutines designed to arrange the sampled values to match the desired correlation structure as specified in the input file. For cases where no correlation structure was defined in the input file, LHS will arrange the sampled values in a way that minimizes correlations between distribution pairs. Correlations are implemented based on the ranks of the sampled distributions rather than on raw distribution values.

7. The final task is to call subroutines designed to write the sample values for each of the distributions to output files. Furthermore, depending on the execution keywords specified in the input file, other output tables may be

created containing sample ranks on a vector by vector basis. Additional output may be produced containing correlation tables based on both rank and raw data and histogram plots for each distribution sampled.

Sampling Techniques used in the WIPP Compliance Certification Application

February 24, 1997 Memorandum

Jon C. Helton
Sandia National Laboratories

1 Sampling Procedures

Extensive use is made of sampling procedures in the CCA performance assessment (PA). In particular, random sampling is used in the generation of individual CCDFs (i.e., for integration over the probability space $(\mathcal{S}_{st}, \mathcal{J}_{st}, p_{st})$ for stochastic uncertainty) and Latin hypercube sampling is used for the assessment of the effects of imprecisely-known analysis inputs (i.e., for integration over the probability space $(\mathcal{S}_{su}, \mathcal{J}_{su}, p_{su})$ for subjective uncertainty). Due to the importance of sampling procedures in this PA, brief descriptions are given for random sampling, importance sampling and Latin hypercube sampling, which are probably the most widely used sampling techniques. For notational convenience, assume that the variable under consideration is represented by

$$\mathbf{x} = [x_1, x_2, ..., x_{nV}] \tag{1.1}$$

and that the corresponding probability space is $(\mathcal{S}, \mathcal{J}, p)$.

In random sampling, sometimes also called simple random sampling, the observations

$$\mathbf{x}_k = [x_{k1}, x_{k2}, ..., x_{k,nV}],\ k = 1, 2, ..., nR, \tag{1.2}$$

where nR is the sample size, are selected according to the joint probability distribution for the elements of $\mathbf{x}$ as defined by $(\mathcal{S}, \mathcal{J}, p)$. In practice, $(\mathcal{S}, \mathcal{J}, p)$ is defined by specifying a distribution D_j for each element x_j of $\mathbf{x}$. Points from different regions of the sample space $\mathcal{S}$ occur in direct relationship to the probability of occurrence of these regions. Further, each sample element is selected independently of all other sample elements. As illustrated in Fig. 1.1 for $x_1 = U$, $x_2 = V$, $nV = 2$ and $nR = 5$, the numbers $RU(1)$, $RU(2)$, ..., $RU(5)$ are sampled from a uniform distribution on [0, 1] and in turn lead to a sample $U(1)$, $U(2)$, ..., $U(5)$ from U based on the cumulative distribution function for U. Similarly, the numbers $RV(1)$, $RV(2)$, ..., $RV(5)$ lead to a sample $V(1)$, $V(2)$, ..., $V(5)$ from V. The pairs

$$\mathbf{x}_k = [U(k), V(k)],\ k = 1, 2, ..., nR = 5, \tag{1.3}$$

then constitute a random sample from $\mathbf{x} = [U, V]$, where U has a normal distribution on [−1, 1] and V has a triangular distribution on [0, 4].

Random samples are generated in an analogous manner when $\mathbf{x}$ has a dimensionality greater than 2 (e.g., nV = 100). Specifically, if the elements of $\mathbf{x}$ are represented by U, V, ..., W and a random sample of size nR is to be generated, then random numbers RU(1), RU(2), ..., RU(nR) are sampled uniformly from [0,1] and used to obtain corresponding values U(1), U(2), ..., U(nR) for U, random numbers RV(1), RV(2), ..., RV(nR) are sampled

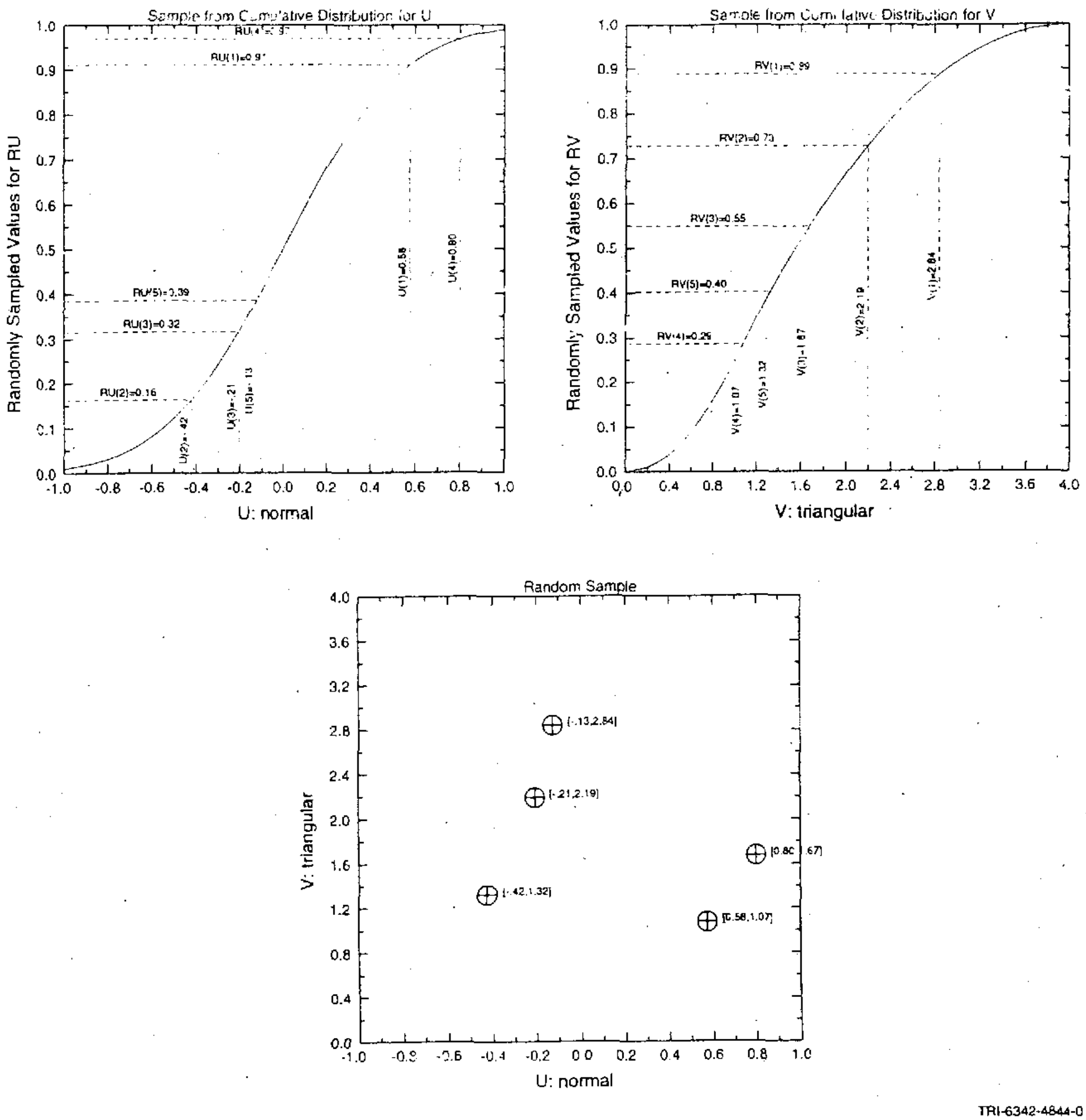


Fig. 1.1. Example of random sampling to generate a sample of size $nR = 5$ from $\mathbf{x} = [U, V]$ with U normal on $[-1, 1]$ (mean = 0, 0.01 quantile = −1, 0.99 quantile = 1) and V triangular on $[0, 4]$ (mode = 1).

uniformly from [0,1] and used to obtain corresponding values V(1), V(2), ..., V(nR) for V, and so on, with the process continuing through all elements of $\mathbf{x}$ and ending with the selection of random numbers RW(1), RW(2), ..., RW(nR) from [0,1] and the generation of the corresponding values W(1), W(2), ..., W(nR) for V. The vectors

$$\mathbf{x}_k = [U(k), V(k)], ..., W(k)], k = 1, 2, ..., nR, \tag{1.3a}$$

then constitute a random sample from $\mathbf{x}$ = [U,V,...,W].

In random sampling, there is no assurance that points will be sampled from any given sub-region of the sample space $\mathcal{S}$. Also, it is possible for an inefficient sampling of $\mathcal{S}$ to occur due to several sampled values falling very close together. The preceding problems can be partially ameliorated by using importance sampling. With this technique, $\mathcal{S}$ is exhaustively divided into a number of nonoverlapping subregions (i.e., strata) $\mathcal{S}_i$, $i = 1, 2, ..., nS$. Then, nS_i values for $\mathbf{x}$ are randomly sampled from $\mathcal{S}_i$, with the random sampling carried out in consistency with the definition of $(\mathcal{S}, \mathcal{J}, p)$ and the restriction of $\mathbf{x}$ to $\mathcal{S}_i$. The resultant vectors

$$\mathbf{x}_k = [x_{k1}, x_{k2}, ..., x_{k,nV}], \; k = 1, 2, ..., \sum_{i=1}^{nS} nS_i \,, \tag{1.4}$$

then constitute an importance-based sample from $\mathcal{S}$ (i.e., a sample obtained by importance sampling). Typically, only one value is sampled from each $\mathcal{S}_i$, with the result that the sample has the form

$$\mathbf{x}_k = [x_{k1}, x_{k2}, ..., x_{k,nV}], \; k = 1, 2, ..., nS, \tag{1.5}$$

The name importance sampling derives from the fact that the $\mathcal{S}_i$ are in part defined on the basis of how important the $\mathbf{x}$'s contained in each set are to the final outcome of the analysis. Often, importance sampling is used to assure the inclusion in an analysis of $\mathbf{x}$'s that have high consequences but low probabilities (i.e., the probabilities $p(\mathcal{S}_i)$ are small for the $\mathcal{S}_i$ that contain such $\mathbf{x}$'s). When importance sampling is used, the probabilities $p(\mathcal{S}_i)$ and number of observations nS_i taken from each $\mathcal{S}_i$ must be folded back into the analysis before results can be meaningfully presented.

Several examples of importance sampling for $\mathbf{x} = [U, V]$ are given in Fig. 1.2. The two top frames are for strata of equal probability (i.e., all $p(\mathcal{S}_i)$ are equal). For two uniform distributions, this results in all strata having the same area (upper left frame). For two nonuniform distributions, different strata can have different areas even though they have the same probability (upper right frame). The two lower frames are for strata of unequal probability. In this case, the variable distributions and the strata probabilities interact to determine the area of the strata. However, it is important to recognize that specifying variable distributions, number of strata and strata probabilities does not uniquely define an importance sampling procedure; rather, there are many ways in which the strata $\mathcal{S}_i$ can be defined

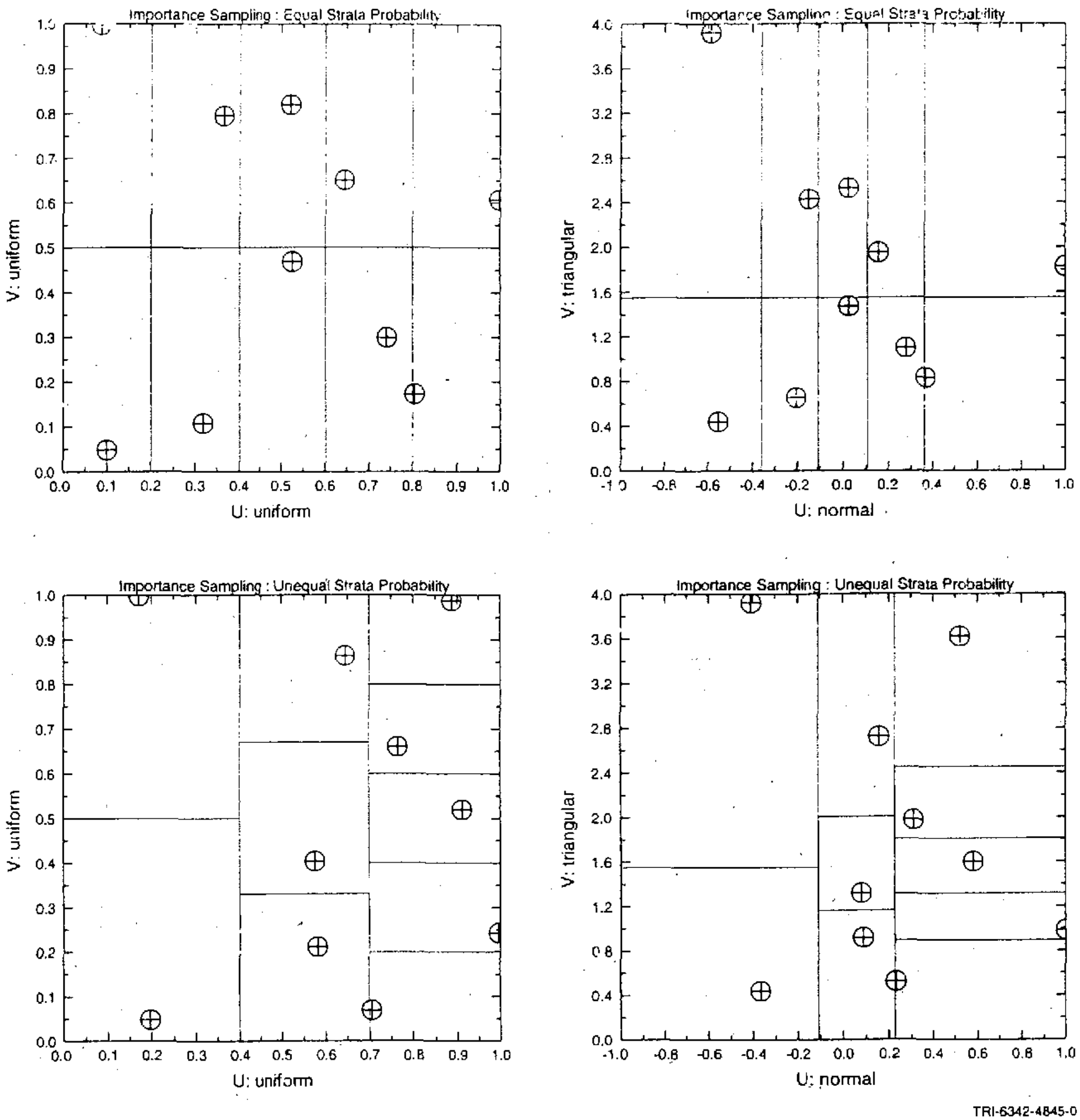


Fig. 1.2. Examples of importance sampling with ten strata (i.e., $nS = 10$), one random sample per strata (i.e., $nS_i = 1$), equal strata probability (i.e., $p(S_i) = 1/10$, upper frames), unequal strata probability (i.e., $p(S_i) = 0.2, 0.2, 0.1, 0.1, 0.1, 0.06, 0.06, 0.06, 0.06$, lower frames), U and V uniform on [0, 1] (left frames) and U normal on [−1, 1] (mean = 0, 0.01 quantile = −1, 0.99 quantile = 1) and V triangular on [0, 4] (mode = 1) (right frames)

that are consistent for the preceding constraints. In particular, appropriate definition of strata will depend on specific properties of individual analyses. Similar ideas also hold for more than two variables, in which case the strata become enclosed volumes in a space with the same dimension as $\mathbf{x}$.

Importance sampling operates to ensure the full coverage of specified regions in the sample space. This idea is carried farther in Latin hypercube sampling to ensure the full coverage of the range of each variable. Specifically, the range of each variable (i.e., the x_j) is divided into *nLHS* intervals of equal probability and one value is selected at random from each interval. The *nLHS* values thus obtained for x_1 are paired at random without replacement with the *nLHS* values obtained for x_2. These *nLHS* pairs are combined in a random manner without replacement with the *nLHS* values of x_3 to form *nLHS* triples. This process is continued until a set of *nLHS nV*-tuples is formed. These *nV*-tuples are of the form

$$\mathbf{x}_k = [x_{k1}, x_{k2}, \ldots, x_{k,nV}],\ k = 1, \ldots, nLHS, \tag{1.6}$$

and constitute the LHS. The individual x_j must be independent for the preceding construction procedure to work; a method for generating Latin hypercube and random samples from correlated variables has been developed by Iman and Conover (1982) and will be discussed briefly. Latin hypercube sampling is an extension of quota sampling (Steinberg 1963) and can be viewed as an *n*-dimensional randomized generalization of Latin square sampling (Raj 1968).

The generation of an *LHS* of size *nLHS* = 5 from $\mathbf{x} = [U, V]$ is illustrated in Fig. 1.3. Initially, the ranges of U and V are subdivided into five intervals of equal probability, with this subdivision represented by the lines that originate at 0.2, 0.4, 0.6 and 0.8 on the ordinates of the two upper frames in Fig. 1.3, extend horizontally to the cumulative distribution functions, and then drop vertically to the abscissas to produce the 5 indicated intervals. Random values $U(1)$, $U(2)$, ..., $U(5)$ and $V(1)$, $V(2)$, ..., $V(5)$ are then sampled from these intervals. The sampling of these random values is implemented by (1) sampling $RU(1)$ and $RV(1)$ from a uniform distribution on [0, 0.2], $RU(2)$ and $RV(2)$ from a uniform distribution on [0.2, 0.4], and so on and (2) then using the cumulative distribution functions to identify (i.e., sample) the corresponding U and V values, with this identification represented by the dashed lines that originate on the ordinates of the two upper frames in Fig. 1.3, extend horizontally to the cumulative distribution functions, and then drop vertically to the abscissas to produce $U(1)$, $U(2)$, ..., $U(5)$ and $V(1)$, $V(2)$, ..., $V(5)$. The generation of the *LHS* is then completed by randomly pairing (without replacement) the resulting values for U and V. As this pairing is not unique, many possible *LHS*s can result. Two such *LHS*s are shown in the lower two frames in Fig. 1.3, with one *LHS* resulting from the pairings [$U(1)$, $V(2)$], [$U(2)$, $V(5)$], [$U(3)$, $V(3)$], [$U(4)$, $V(1)$], [$U(5)$, $V(3)$] (lower left frame) and the other LHS resulting from the pairings [$U($ $)$, $V($ $)$], [$U($ $)$, $V($ $)$], [$U($ $)$, $V($ $)$], [$U($ $)$, $V($ $)$], [$U($ $)$, $V($ $)$].

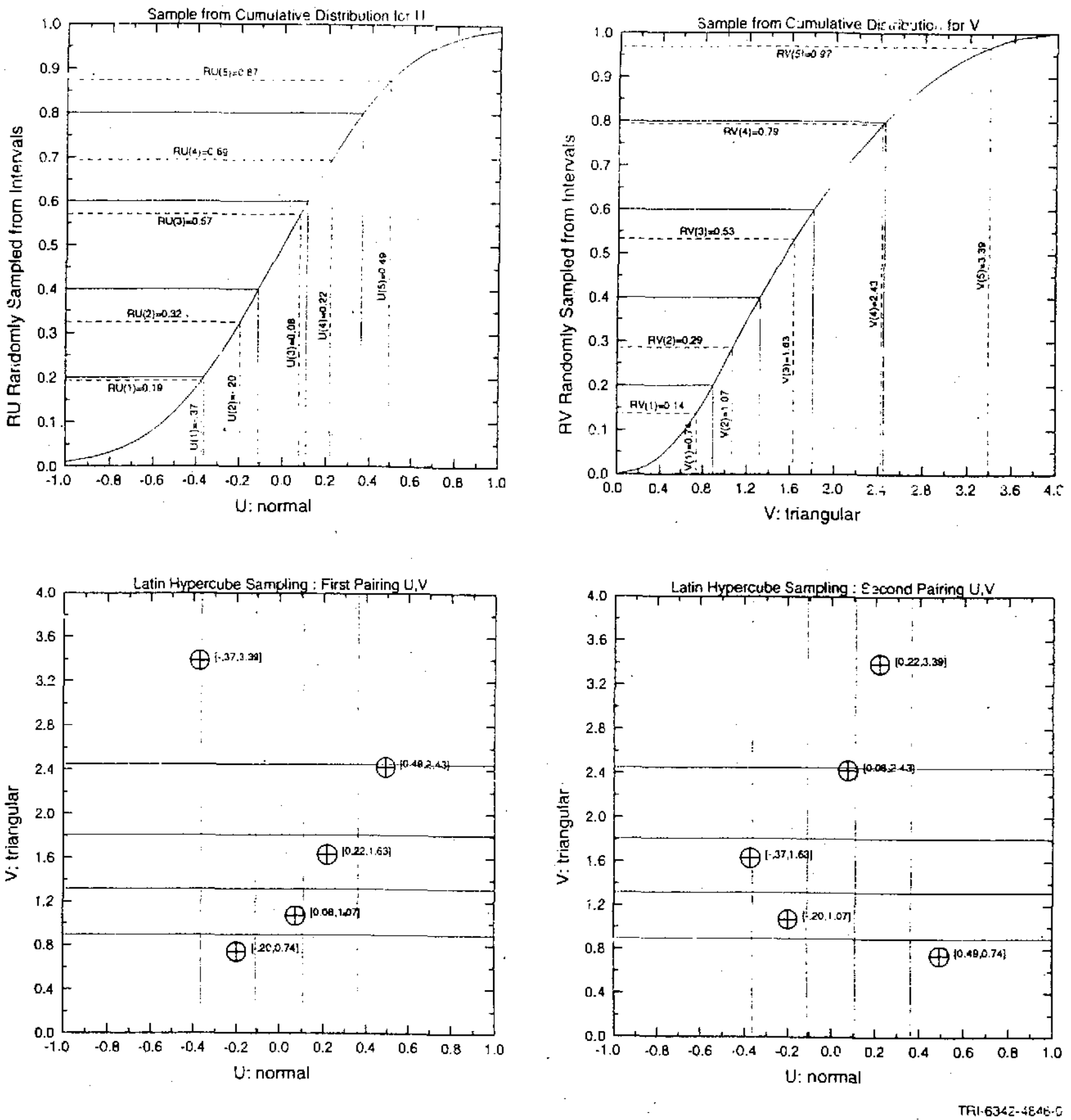


Fig. 1.3. Example of Latin hypercube sampling to generate a sample of size *nLHS* = 5 from **x** = [*U*, *V*] with *U* normal on [−1, 1] (mean = 0, 0.01 quantile = −1, 0.99 quantile = 1) and *V* triangular on [1, 4] (mode = 1).

The generation of an *LHS* for $nV > 2$ proceeds in a manner similar to that shown in Fig. 1.3 for $nV = 2$. The sampling of the individual variables for $nV > 2$ takes place in exactly the same manner as shown in Fig. 1.3. However, the nV variables define an nV-dimensional solid rather than a 2-dimensional rectangle in the plane. Thus, the two lower frames in Fig. 1.3 would involve a partitioning of an nV-dimensional solid rather than a rectangle.

Random sampling is the preferred technique when sufficiently large samples are possible because it is easy to implement, easy to explain, and provides unbiased estimates for means, variances and distribution functions. The possible problems with random sampling derive from the rather vague phrase "sufficiently large" in the preceding sentence. When the underlying models are expensive to evaluate (e.g., many hours of CPU time per evaluation) or estimates of extreme quantiles are needed (e.g., the 0.999999 quantile), the required sample size to achieve a specific purpose may be too large to be computationally practicable. In the CCA PA, random sampling is used for the estimation of CCDFs (i.e., integration over $(\mathcal{S}_{st}, \mathcal{J}_{st}, p_{st})$) because it was possible to develop a computational strategy that allowed the use of a sample of size $nS = 10000$ to estimate an exceedance probability of 0.001 (i.e., the 0.999 quantile of the distribution of normalized releases to the accessible environment).

When random sampling is not computationally feasible for the estimation of extreme quantiles, importance sampling is often employed. However, the use of importance sampling on nontrivial problems is not easy due to the difficulty of defining the necessary strata and also of calculating the probabilities of these strata. For example, the fault and event tree techniques used in probabilistic risk assessments for nuclear power stations and other complex engineered facilities can be viewed as algorithms for defining importance sampling procedures. The bottom line is that the definition and implementation of an importance sampling procedure is not easy. Further, without extensive *a priori* knowledge, the strata may end up being defined much more finely than is necessary, with the result that the importance sampling procedure ends up requiring more calculations than the use of random sampling to calculate the same outcomes. For example, the numbers of strata in the importance sampling procedure used to estimate CCDFs in the 1991 and 1992 preliminary WIPP PAs (Helton and Iuzzolino 1993) greatly exceeds the size of the random samples used in the CCA PA to estimate CCDFs. The unequal strata probabilities also make the outcomes of analyses based on importance sampling inconvenient for use in sensitivity analyses (e.g., how does one interpret a scatterplot or a regression analysis derived from results obtained from an importance sampling procedure). For the preceding reasons, importance sampling was not used in the PA for the CCA.

Latin hypercube sampling is used when large samples are not computationally practicable and the estimation of very high quantiles is not required. The preceding is typically the case in uncertainty and sensitivity studies to assess the effects of subjective uncertainty. First, the models under consideration are often computationally demanding, with the result that the number of calculations that can be performed to support the analysis is necessarily limited. For example, the totality of the model calculations (i.e., BRAGFLO, NUTS, PANEL, GRASP_INV, SECOFL2D, SECOTP2D, CUTTINGS_S, BRAGFLO_DBR) in the CCA PA is too extensive to permit the generation of

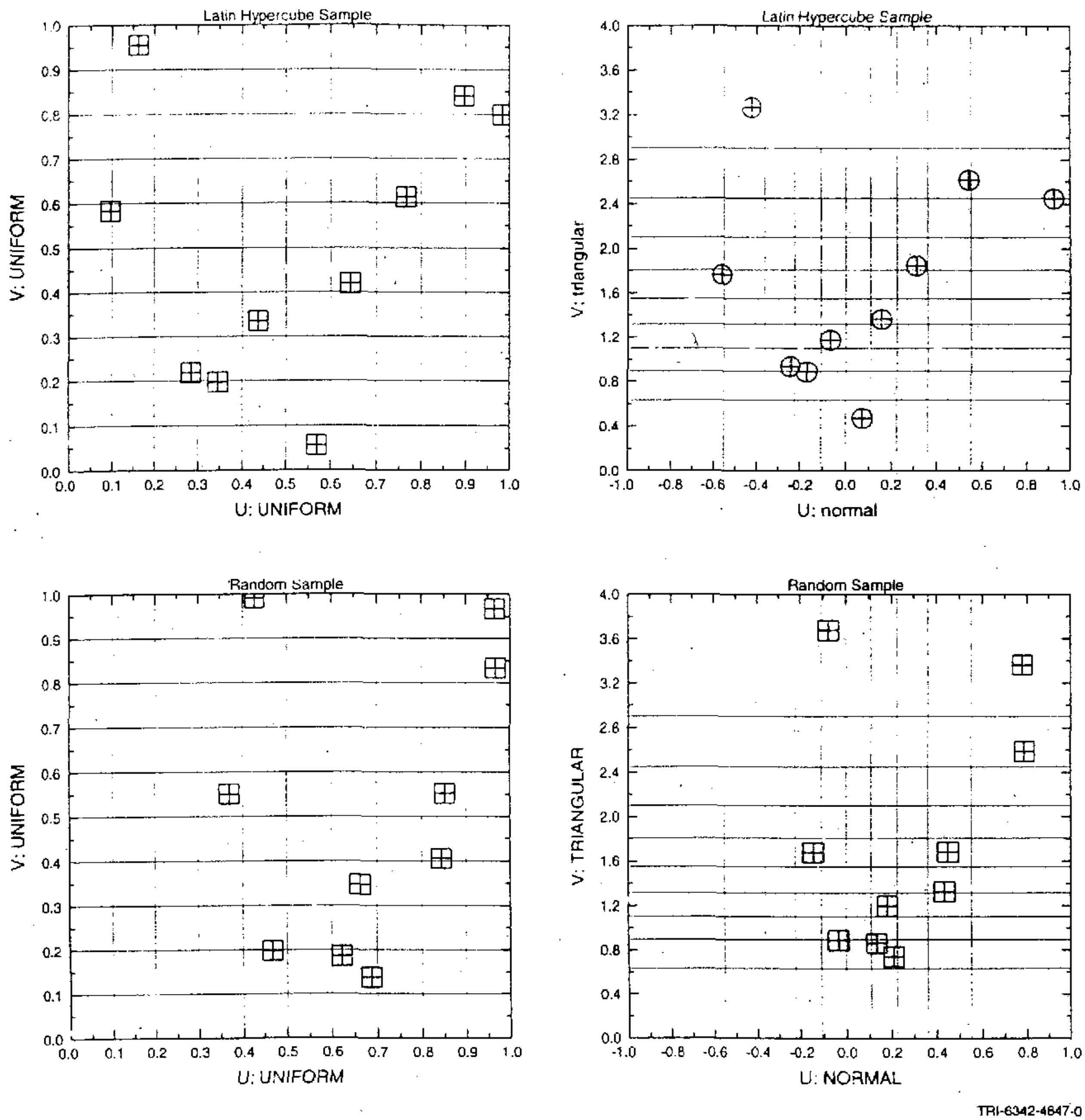


Fig. 1.4. Examples of Latin hypercube and random sampling to generate a sample of size 10 from variables *U* and *V* with (1) *U* and *V* uniform on [−1, 1], and (2) *U* normal on [−1, 1] (mean = 0, 0.01 quantile = −1, 0.99 quantile = 1) and *V* triangular on [0, 4] (mode = 1).

thousands of CCDFs in an uncertainty/sensitivity study to assess the effects of subjective uncertainty. Second, the estimation of very high quantiles is generally not required in an analysis to assess the effects of subjective uncertainty. Typically, a 0.90 or 0.95 quantile is adequate to establish where the available information indicates a particular analysis outcome is likely to be located; in particular, a 0.99, 0.999 or 0.9999 quantile is probably not needed in assessing the effects of subjective uncertainty.

Desirable features of Latin hypercube sampling include unbiased estimates for means and distribution functions and dense stratification across the range of each sampled variable (McKay et al. 1979). In particular, uncertainty and sensitivity analysis results obtained with Latin hypercube sampling have been observed to be quite robust even when relatively small samples (i.e., $nLHS = 50$ to 200) are used (Iman and Helton 1988, 1991; Helton et al. 1995).

For perspective, Latin hypercube and random sampling are illustrated in Fig. 1.4 for two different distribution pairs. To facilitate comparisons, the grid that underlies the *LHS*s is also shown for the random samples, although it plays no role in the actual generation of these samples. The desirability of Latin hypercube sampling derives from the full coverage of the range of the sampled variables; specifically, each equal probability interval for U and also each probability interval for V has exactly one value sampled from it. In contrast, the random sampling makes less efficient use of the sampled points, with the possibility existing that significant parts of a variables range will be omitted and that other parts will be overemphasized. The enforced stratification in Latin hypercube sampling prevents such inefficient samplings while still providing unbiased estimates for means and distribution functions.

The outcome of the enforced stratification associated with Latin hypercube sampling is that estimates of means and distribution functions tend to be more stable when generated by Latin hypercube sampling than by random sampling. Here, stability refers to the amount of variation between results obtained with different samples generated by the particular sampling technique under consideration. This stability can be illustrated by comparison of estimates of the CDF for the simple function

$$f(U, V) = U + V + UV \tag{1.6}$$

obtained with Latin hypercube and random sampling under the assumption that U and V are uniformly distributed on [0, 2]. In particular, each sampling technique is used to generate 100 samples of size 10 and also 100 samples of size 100 from U and V. Each sample gives rise to an estimated CDF for f (Fig. 1.5). The goal is to compare the variability between the estimates obtained with Latin hypercube and random sampling. Presenting plots similar to those in Fig. 1.5 for 100 CDFs at a time is not very informative because the CDFs tend to turn into a solid black mass. A more informative presentation is to summarize the distributions of CDFs with mean and percentile curves. The location of the percentile curves then provides an indication of how stable the estimates of the CDFs are. In particular, limited separation between low and high percentiles (e.g., the 10th and 90th) indicates that the sampling

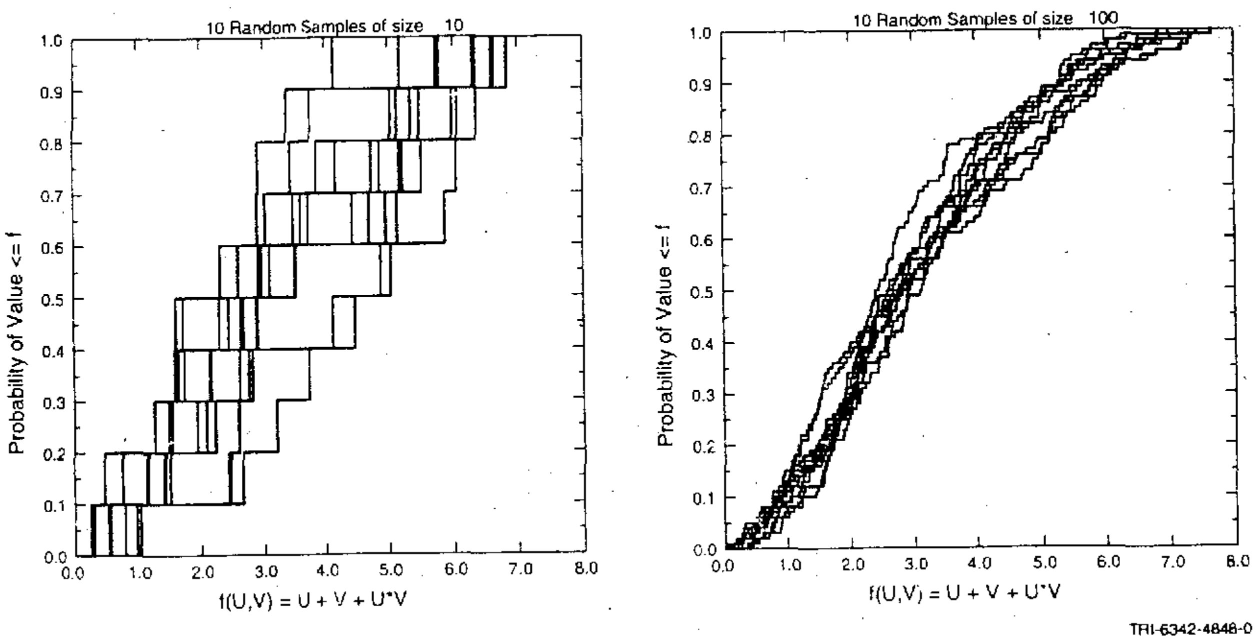


Fig. 1.5. Example CDFs for $f(U, V) = U + V + UV$ estimated with random samples of size 10 and 100 under the assumption that U and V are uniformly distributed on $[0, 2]$.

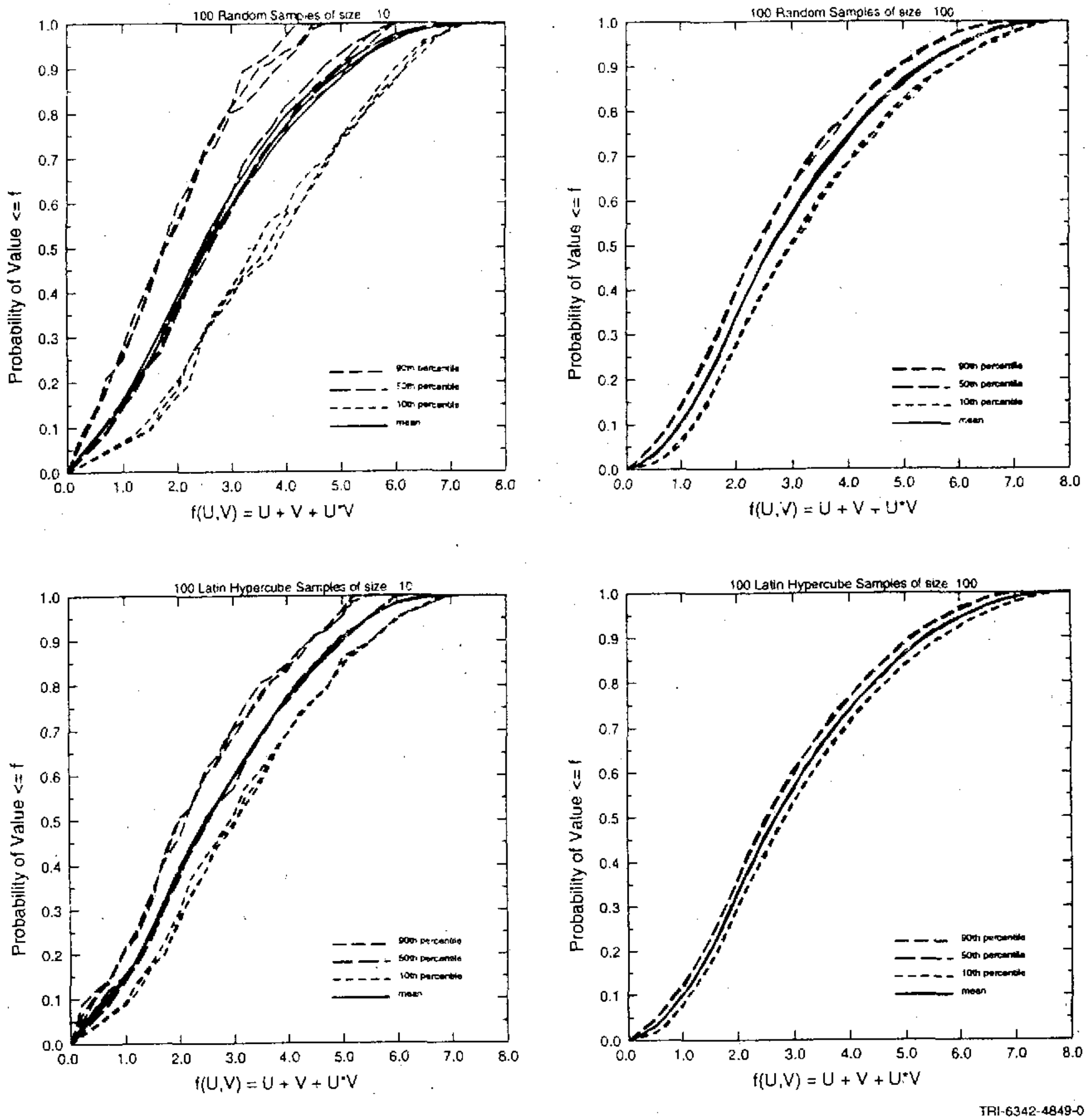


Fig. 1.6. Summary of distribution of CDFs for $f(U, V) = U + V + UV$ estimated with 3 replications of 100 Latin hypercube and random samples of size 10 and 100 under the assumption that U and V are uniformly distributed on [0, 2].

procedure is providing stable estimates of the CDF (i.e., there is little variability in the estimated CDF from one sample to the next); in contrast, a large spread between low and high percentiles indicates that the sampling procedure is not providing stable estimates of the CDF (i.e., there is substantial variability in the estimated CCDF from one sample to the next). The previously indicated 100 samples of size 10 and 100 are summarized in this manner in Fig. 1.6. Further, the analysis was replicated three times to give three estimates of the 10th percentile, three estimates of the 50th percentile, and so on. As examination of Fig. 1.6 shows, Latin hypercube sampling is producing CDF estimates that are more stable than those produced by random sampling (i.e., the spread between the 10th and 90th percentile curves is tighter for Latin hypercube sampling than for random sampling). The stability of the mean and percentile estimates across the three replicates indicates that the observed stability is real rather than a chance occurrence associated with a particular set of 100 Latin hypercube or random samples.

From the perspective of uncertainty and sensitivity analysis, the full stratification over the range of each sampled variable is a particularly desirable property of Latin hypercube sampling. In a large analysis such as the CCA PA, there are potentially hundreds of predicted variables that will be examined at some point in the uncertainty and sensitivity analysis. Further, it is likely that almost every sampled variable will be important with respect to at least one of these predicted variables. With Latin hypercube sampling, every variable gets equal treatment (i.e., full stratification) within the sample; should a variable be important with respect to a particular output variable, it has been sampled in a way that will permit this importance to be identified. In contrast, it is very difficult to design an importance sampling procedure that provides acceptable results for a large number of sampled and predicted variables. In one sense, Latin hypercube sampling can be viewed as a compromise importance sampling procedure when *a priori* knowledge of the relationships between the sampled and predicted variables is not available. When random sampling is used with a small sample size in an analysis that involves a large number of sampled and predicted variables, the possibility exists that the chance structure of the sample will result in a poor representation of the relationships between some of the sampled and predicted variables. Such poor relationships can also occur for Latin hypercube sampling when several sampled variables affect a given predicted variable, but are less likely to do so than is the case with random sampling.

For reasons just outlined, the CCA PA uses Latin hypercube sampling to determine the effects of subjective uncertainty (i.e., to integrate over ($\mathcal{S}_{su}$, $\mathcal{A}_{su}$, p_{su})). In particular, Latin hypercube sampling is felt to be the most appropriate procedure to use to meet the requirement in 40 CFR 194.34(b) that "Computational techniques, which draw random samples from across the entire range of the probability distributions developed pursuant to paragraph (b) of this section, shall be used in generating CCDFs and shall be documented in any compliance application."

2 Correlation Control (Adapted from Sect. 3.2 of Helton 1993)

Control of correlation within a sample can be very important. If two or more variables are correlated, then it is necessary that the appropriate correlation structure be incorporated into the sample if meaningful results are to be obtained in subsequent uncertainty/sensitivity studies. On the other hand, it is equally important that variables do not appear to be correlated when they are really independent.

It is often difficult to induce a desired correlation structure on a sample. Indeed, most multivariate distributions are incompatible with the majority of correlation patterns that might be proposed for them. Thus, it is fairly common to encounter analysis situations where the proposed variable distributions and the suggested correlations between the variables are inconsistent; that is, it is not possible to have both the desired variable distributions and the requested correlations between the variables.

In response to this situation, Iman and Conover (1982) have proposed a method of controlling the correlation structure in random and Latin hypercube samples that is based on rank correlation (i.e., on rank-transformed variables) rather than sample correlation (i.e., on the original untransformed data). With their technique, it is possible to induce any desired rank-correlation structure onto the sample. This technique has a number of desirable properties: (1) It is distribution free. That is, it may be used with equal facility on all types of distribution functions. (2) It is simple. No unusual mathematical techniques are required to implement the method. (3) It can be applied to any sampling scheme for which correlated input variables can logically be considered, while preserving the intent of the sampling scheme. That is, the same numbers originally selected as input values are retained; only their pairing is affected to achieve the desired rank correlations. This means that in Latin hypercube sampling the integrity of the intervals is maintained. If some other structure is used for selection of values, that same structure is retained. (4) The marginal distributions remain intact.

For many, if not most, uncertainty/sensitivity analysis problems, rank-correlation is probably a more natural measure of congruent variable behavior than is the more traditional sample correlation. What is known in most situations is some idea of the extent to which variables tend to move up or down together; more detailed assessments of variable linkage are usually not available. It is precisely this level of knowledge that rank correlation captures.

The following discussion provides an overview of the Iman/Conover procedure for inducing a desired rank correlation structure on either a random or a Latin hypercube sample. A more detailed discussion of the procedure is given in the original article. The procedure begins with a sample of size m from the n input variables under consideration. This sample can be represented by the $m \times n$ matrix

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} \tag{2.1}$$

where x_{ij} is the value for variable j in sample element i. Thus, the rows of $\mathbf{X}$ correspond to sample elements, and the columns of $\mathbf{X}$ contain the sampled values for individual variables.

The procedure is based on rearranging the values in the individual columns of **X** so that a desired rank correlation structure results between the individual variables. For convenience, let the desired correlation structure be represented by the $n \times n$ matrix

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix} \tag{2.2}$$

where c_{kl} is the desired rank correlation between variables x_k and x_l.

Although the procedure is based on rearranging the values in the individual columns of **X** to obtain a new matrix $\mathbf{X}^*$ that has a rank correlation structure close to that described by **C**, it is not possible to work directly with **X**. Rather, it is necessary to define a new matrix.

$$\mathbf{S} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1n} \\ s_{21} & s_{22} & \cdots & s_{2n} \\ \vdots & \vdots & & \vdots \\ s_{m1} & s_{m2} & \cdots & s_{mn} \end{bmatrix} \tag{2.3}$$

that has the same dimensions as **X**, but is otherwise independent of **X**. Each column of **S** contains a random permutation of the m van der Waerden scores (Conover 1980) $\Phi^{-1}(i/m + 1)$, $i = 1, 2, \ldots, m$, where Φ^{-1} is the inverse of the standard normal distribution. The matrix **S** is then rearranged to obtain the correlation structure defined by **C**. This rearrangement is based on the Cholesky factorization (Golub and van Loan 1983) of **C**. That is, a lower triangular matrix P is constructed such that

$$\mathbf{C} = \mathbf{P}\mathbf{P}^T \tag{2.4}$$

This construction is possible because **C** is a symmetric, positive-definite matrix (Golub and van Loan 1983, p. 88).

If the correlation matrix associated with **S** is the $n \times n$ identity matrix (i.e., if the correlations between the values in different columns of **S** are zero), then the correlation matrix for

$$\mathbf{S}^* = \mathbf{S}\mathbf{P}^T \tag{2.5}$$

is **C** (Anderson 1984, p. 25). At this point, the success of the procedure depends on the following two conditions: (1) that the correlation matrix associated with **S** be close to the $n \times n$ identity matrix; and (2) that the correlation matrix for $\mathbf{S}^*$ be approximately equal to the rank correlation matrix for $\mathbf{S}^*$. If these two conditions hold, then the desired matrix $\mathbf{X}^*$ can be obtained by simply rearranging the values in the individual columns of **X** in the same rank order as the values in the individual columns of $\mathbf{S}^*$. This is the first time that the variable values contained in $\mathbf{X}$ enter into the correlation process. When $\mathbf{X}^*$ is constructed in this manner, it will have the same rank correlation matrix as $\mathbf{S}^*$. Thus, the rank correlation matrix for $\mathbf{X}^*$ will approximate $\mathbf{C}$ to the same extent that the rank correlation matrix for $\mathbf{S}^*$ does.

The condition that the correlation matrix associated with $\mathbf{S}$ be close to the identity matrix is now considered. For convenience, the correlation matrix for $\mathbf{S}$ will be represented by $\mathbf{E}$. Unfortunately, $\mathbf{E}$ will not always be the identity matrix. However, it is possible to make a correction for this. The starting point for this correction is the Cholesky factorization for $\mathbf{E}$:

$$\mathbf{E} = \mathbf{Q}\mathbf{Q}^T \tag{2.6}$$

This factorization exists because $\mathbf{E}$ is a symmetric, positive-definite matrix. The matrix S^* defined by

$$\mathbf{S}^* = \mathbf{S}(\mathbf{Q}^{-1})^T\mathbf{P}^T \tag{2.7}$$

has $\mathbf{C}$ as its correlation matrix. In essence, multiplication of $\mathbf{S}$ by $(\mathbf{Q}^{-1})^T$ transforms $\mathbf{S}$ into a matrix whose associated correlation matrix is the $n \times n$ identity matrix; then, multiplication by $\mathbf{P}^T$ produces a matrix whose associated correlation matrix is $\mathbf{C}$. As it is not possible to be sure that $\mathbf{E}$ will be an identity matrix, the matrix S^* used in the procedure to produce correlated input should be defined in the corrected form shown in (2.7) rather than in the uncorrected form shown in (2.5).

The condition that the correlation matrix for $\mathbf{S}^*$ be approximately equal to the rank correlation matrix for $\mathbf{S}^*$ depends on the choice of the scores used in the definition of $\mathbf{S}$. On the basis of empirical investigations, Iman and Conover (1982) found that van der Waerden scores provided an effective means of defining $\mathbf{S}$, and these scores are incorporated into the rank correlation procedure in the widely used LHS program (Iman and Shortencarier 1984). Other possibilities for defining these scores exist, but have not been extensively investigated. The user should examine the rank correlation matrix associated with $\mathbf{S}^*$ to ensure that it is close to the target correlation matrix $\mathbf{C}$. If this is not the case, the construction procedure used to obtain $\mathbf{S}^*$ can be repeated until a suitable approximation to $\mathbf{C}$ is obtained. Results given in Iman and Conover (1982) indicate that the use of van der Waerden scores leads to rank correlation matrices for $\mathbf{S}^*$ that are close to the target matrix $\mathbf{C}$.

Additional information on the Iman/Conover (i.e., restricted pairing) technique to induce a desired rank-correlation structure is given in the original article. The results of various rank-correlation assumptions are illustrated in Iman and Davenport (1982). The LHS (Latin Hypercube Sample) program generates both random and Latin hypercube samples with user-specified rank correlations between variables.

3 Sample Size for Incorporation of Subjective Uncertainty

The guidance in 40 CFR 194.34(d) states that "The number of CCDFs generated shall be large enough such that, at cumulative releases of 1 and 10, the maximum CCDF generated exceeds the 99th percentile of the population of CCDFs with at least a 0.95 probability." For a Latin hypercube or random sample of size n, the preceding guidance is equivalent to the inequality

$$1 - 0.99^n > 0.95, \tag{3.1}$$

which results in a minimum value of 298 for n. In consistency with the preceding result, the 1996 CCA PA uses an LHS of size 300 to integrate over the probability space $(\mathcal{S}_{su}, \mathcal{A}_{su}, p_{su})$ for subjective uncertainty. Actually, as discussed in the next section, three replicated LHSs of size 100 each are used, which results in a total sample size of 300.

4 Statistical Confidence on Mean CCDF

The guidance in 40 CFR 194.34(f) states that "Any compliance assessment shall provide information which demonstrates that there is at least a 95 percent level of statistical confidence that the mean of the population of CCDFs meets the containment requirements of § 191.13 of this chapter." Given that Latin hypercube sampling is to be used, the confidence intervals required in 194.34(f) can be obtained with a replicated sampling technique proposed by R.L. Iman (1982). In this technique, the LHS in Eq. (1.6) is repeatedly generated with different random seeds. These samples lead to a sequence $\overline{P}_r(R)$, $r = 1, 2, \ldots, nR$, of estimated mean exceedance probabilities, where $\overline{P}_r(R)$ defines the mean CCDF obtained for sample r (i.e., $\overline{P}_r(R)$ is the mean probability that a normalized release of size R will be exceeded) and nR is the number of independent LHSs generated with different random seeds. Then,

$$\overline{P}(R) = \sum_{r=1}^{nR} \overline{P}_r(R) / nR \tag{4.1}$$

and

$$SE(R) = \left\{ \sum_{r=1}^{nR} \left[\overline{P}_r(R) - \overline{P}(R) \right]^2 / nR(nR-1) \right\}^{1/2} \tag{4.2}$$

provide an additional estimate of the mean CCDF and an estimate of the standard error associated with the mean exceedance probabilities. The t-distribution with $nR-1$ degrees of freedom can be used to place confidence intervals around the mean exceedance probabilities for individual R values (i.e., around $\overline{P}(R)$). Specifically, the $1-\alpha$ confidence interval is given by $\overline{P}(R) \pm t_{1-\alpha/2}\, SE(R)$, where $t_{1-\alpha/2}$ is the $1-\alpha/2$ quantile of the t-distribution with

$nR-1$ degrees of freedom (e.g., $t_{1-\alpha/2} = 4.303$ for $\alpha = 0.05$ and $nR = 3$). The same procedure can also be used to place pointwise confidence intervals around percentile curves.

5 Generation of LHSs

The LHS program (Iman and Shortencarier 1984) was used to produce three independently generated LHSs of size $nLHS = 100$ each, for a total of 300 sample elements. Each individual replicate is an LHS of the form

$$\mathbf{x}_{su,k}=[x_{k1},x_{k2},...,x_{k,nV}],\ k=1,2,...,nLHS=100. \tag{5.1}$$

In the context of the replicated sampling procedure described in Sect. 4, $nR = 3$ replicates are being used with each replicate of size 100. For notational convenience, the replicates are designated by R1, R2 and R3 for replicates 1, 2 and 3, respectively.

At the beginning of the analysis, only the 31 variables that are used as input to BRAGFLO had been fully specified (i.e., their distributions D_j had been unambiguously defined); the remaining variables were still under development. To allow the calculations with BRAGFLO to proceed, the previously indicated LHSs were generated from $nV = 75$ variables, with the first 31 variables being the then specified inputs to BRAGFLO and the remaining 44 variables being assigned uniform distributions on [0, 1]. Later, when the additional variables were fully specified, the uniformly distributed variables were used to generate sampled values from them consistent with their assigned distributions. This procedure allowed the analysis to go forward while maintaining the integrity of Latin hypercube sampling procedure for the overall analysis.

With $nV = 75$ in the LHSs and 31 variables already assigned, 44 additional variables were available for incorporation into the analysis. To assure that the number of available positions in the LHSs was not exceeded, each group of investigators developing characterizations of variable uncertainty was assigned a maximum number of variables that they could elect to have incorporated into the analysis, with the sum of these maximums being less than 44. Ultimately, 26 additional variables were selected for incorporation into the analysis, which produced the 57 variables sampled in the CCA PA.

The restricted pairing technique described in Sect. 3 was used to induce requested correlations and also to assure that uncorrelated variables had correlations close to zero. Due to the sequential manner in which the variables were developed, it was actually only the first 31 variables used as input to BRAGFLO that could have specified non-zero correlations. The correlations for the remaining variables were controlled in the sense that they were forced to be close to zero.

The variable pairs (*ANHCOMP*, *ANHPRM*), (*HALCOMP*, *HALPRM*) and (*BPCOMP*, *BPPRM*) were assigned rank correlations of –0.99, –0.99 and –0.75, respectively. Further, all other variable pairs were assigned rank correlations of zero. The restricted pairing technique was quite successful in producing these correlations. Specifically, the correlated variables have correlations that are close to their specified values and uncorrelated variables have correlations that are close to zero.

14. REFERENCES

1. Anderson, T.W. 1984. *An Introduction to Multivariate Statistical Analysis.* 2nd Ed. New York, NY: Wiley.

2. Conover, W. J. 1980. *Practical Nonparametric Statistics* (2nd ed.). New York: Wiley.

3. Golub, G., and C.F. van Loan. 1983. *Matrix Computations.* Baltimore, MD: John Hopkins University Press.

4. Helton, J.C. and H.J. Iuzzolino. Construction of complementary cumulative distribution functions for comparison with the EPA release limits for radioactive waste disposal, Reliabl. Eng. Syst. Safety 40 (1993d) 277-293.

5. Helton, J.C. 1993. "Uncertainty and Sensitivity Analysis Techniques for Use in Performance Assessment for Radioactive Waste Disposal," *Reliability Engineering & System Safety.* Vol. 42, no. 2-3, 327-367.

6. Helton, J.C., D.R. Anderson, B.L. Baker, J.E. Bean, J.W. Berglund, W. Beyeler, J.W. Garner, H.J. Iuzzolino, M.G. Marietta, R.P. Rechard, P.J. Roache, D.K. Rudeen, J.D. Schreiber, P.N. Swift, M.S. Tierney and P. Vaughn. 1994. Effect of alternative conceptual models in a preliminary performance assessment for the waste isolation pilot plant. Reprinted from Nuclear Engineering and Design 154 (1995) 251-344.

7. Iman, R.L. 1982. "Statistical Methods for Including Uncertainties Associated with Geologic Isolation of Radioactive Waste Which Allow for a Comparison with Licensing Criteria," in D. C. Kocher (ed.). *Proceedings of the Symposium on Uncertainties Associated with the Regulation of the Geologic Disposal of High-Level Radioactive Waste: March 9-13, 1981, Gatlinburg., TN.* NUREG/CP-0022, CONF-810372. Oak Ridge, TN: Oak Ridge National Laboratory. 145-157.

8. Iman, R.L. and W.J. Conover. A distribution-free approach to inducing rank correlation among input variables, Commu. Stat. B11 (1982a) 311-334.

9. Iman, R.L. and J.M. Davenport. Rank correlation plots for use with correlated input variables. Commun. Stat. B11 (1982b) 335-360.

10. Iman, R.L. and J.C. Helton. An investigation of uncertainty and sensitivity analysis techniques for computer models, Risk Anal. 8 (1988) 71-90.

11. Iman, R.L. and J.C. Helton. The repeatability of uncertainty and sensitivity analyses for complex probabilistic risk assessments, Risk Anal. 11 (1991) 591-606.

12. Iman, R.L., and M.J. Shortencarier. 1984. *A Fortran 77 Program and User's Guide for the Generation of Latin Hypercube and Random Samples for Use with Computer Models.* SAND83-2365. Albuquerque, NM: Sandia National Laboratories. Sandia WIPP Central Files WPO # 41121.

13. McKay, M.D., R.J. Beckman, and W.J. Conover. 1979. "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics.* Vol. 21, no. 2, 239-245. Sandia WIPP Central Files WPO # 41318.

14. Raj, D. 1968. *Sampling Theory.* New York, NY: McGraw, Hill. 206-209.

15. Steinberg, H.A. 1963. "Generalized Quota Sampling." *Nuclear Science and Engineering* 15: 142-145.

EPA Comment
Enclosure 2, Page 11
194.44

Comment Text

Engineered Barriers

194.44
Section 194.44 requires that the disposal system incorporate engineered barriers designed to prevent or substantially delay the movement of radionuclides towards the accessible environment.

While the inclusion [of] magnesium oxide (MgO) as a backfill material will improve repository performance, the Department must provide an engineered design which supports the assumptions about the performance of MgO. The evidence must support the assumptions used in PA.

The Department must provide an engineering design which provides the method of placement and quantity emplaced such that the MgO will be distributed as assumed in the conceptual models to support the reaction of MgO to be as predicted in the expected WIPP repository environment. The Department must also provide information which demonstrates that the excess volume proposed to be emplaced can actually be accommodated and whether it covers the uncertainties in the actual geochemical process.

DOE Response

The DOE will respond to this comment in two parts: Part I will address the physical placement of MgO within the repository and address the stated concern regarding the distribution of MgO being consistent with the conceptual models assumed. Part II demonstrates that the excess volume can be emplaced and that the uncertainties in possible reactions has been considered in determining the quantity to be emplaced.

Uncertainty in the geochemical reactions that may occur is not covered as part of the response to this EPA comment. Rather, uncertainty in geochemical reactions is being addressed in response to a different EPA comment through direct experimental demonstration of the efficacy of MgO. It is important to remember that the MgO backfill is an assurance measure intended to satisfy, in part, the requirement to include an engineered barrier in the disposal system.

Part I- Engineering Design for MgO placement

The method of MgO placement and quantity are described in Section 3.3.3 of the CCA. Preliminary placement test have been performed using minisacks and supersacks in the repository setting and demonstrate that backfill can be emplaced as described in Section 3.3.3 without significant impact to waste handling operations.

WIPP waste handling procedures WH-1011, Revision 2, dated October 1, 1996, describes the emplacement procedure for mini sacks of MgO in the void spaces between the waste drums in the 7-pack configuration. Current waste handling procedures for emplacement of waste in rooms are being modified to specify MgO placement with standard waste boxes (SWBs), MgO placement along the rib (space between the waste containers and the wall of the room), and super sack placement on top of waste containers. These procedures will provide those specific actions that must be performed such that the emplacement design specified in Section 3.3.3 will be achieved.

Part II- Consistency of emplacement method with conceptual models and consideration of uncertainty in determination of quantities of MgO to be emplaced.

The emplacement methodology described in Section 3.3.3 of the CCA was developed with specific consideration of the performance criteria required by the conceptual models relevant to MgO backfill. The assumptions in the conceptual models regarding MgO are covered in Sections 6.4.3.3 (Gas Generation), 6.4.3.3 (Chemical Conditions), 6.4.3.5 (Dissolved Actinide Source Term), and 6.4.3.6 (Colloidal Actinide Source Term), as well as Appendix SOTERM and Appendix BACK of the CCA. Additional discussions of MgO backfill can be found in Chapter 9, Section 9.3.1.2.10.2 and Appendix PEER, Section 1. The essential performance criteria relevant to placement of MgO are (1) that MgO be distributed in such a manner that the assumption of uniformity of chemical conditions through disposal rooms is valid, and (2) enough MgO is emplaced such that the consumption of all CO_2 produced is assured.

The MgO will be emplaced consistent with the design specified in Section 3.3.3. This design provides for MgO to be distributed throughout waste disposal rooms, including between waste drums and standard waste boxes. This distribution is sufficient to assure accessibility of MgO to chemical reactions taking place over the regulatory time scale. The Conceptual Model Peer Review Panel considered this in their examination of the MgO issue and found the emplacement design to be consistent with the conceptual model assumptions (Final Report, July 1996, page 3-154, Assumption (1)).

The quantity of MgO required to be emplaced to assure removal of CO_2 from the gas phase is based on calculations that consider all processes that might contribute to CO_2 production. These calculations are very conservative in that they utilize a maximum estimate of CO_2 production, a quantity that is unlikely to be attained. Based on the BIR and memoranda in the records center, the total number of moles of MgO required to react with the maximum possible amount of CO_2 generated is 9.85×10^8 moles. Using the appropriate conversion factors (40.3 gm/mole, 0.001 kg/gm, 2.202 kg/lb, 0.0005 lb/ton), at total of 43,700 tons of MgO are required to react with this maximum estimate of carbon dioxide production. Section 3.3.3 of the CCA documents that approximately 85,600 tons of backfill will be emplaced in the repository. Therefore, by dividing the mass of backfill to be emplaced (85,600 tons) by the maximum mass of MgO required to react with the maximum possible carbon dioxide production (43,700 tons), a 1.95 factor of safety results. In other words, 95% more MgO will be emplaced than is required to react with a conservative estimate of the maximum quantity of CO_2 production.

EPA Comment
Enclosure 2, Page 11
194.51

<u>*Consideration of Protected Individual*</u>

Section 194.51 requires, among other things, that exposure from all sources of radionuclide release from the disposal system to the accessible environment be examined.

Chapter 8 of the CCA provides a bounding analysis to demonstrate compliance with 40 CFR 191.15. However, the analysis only assumes exposure via consumption of potable water. It does not explicitly include the analysis of doses posed by other potential exposure pathways such as stock consumption or irrigation.

The Department needs to provide documentation which discusses why pathways other than consumption of potable water are not considered.

DOE Response

The DOE has analyzed pathways other than the consumption of potable water as reported in the attached report, "*Analysis Report for Estimating Dose from Cattle, Vegetable Consumption and Inhalation Pathways Utilizing Contaminated Water from the Top of the Salado, Culebra, and Selected Marker Beds for an Undisturbed Case Supporting Review Compliance Certification Application.*" The report shows that doses received from these pathways are indeed negligible and are bounded by the analysis conducted, described, and reported in Chapter 8 of the CCA. It is important to remember that these pathways are more unrealistic than the bounding case evaluated in the CCA.

Analysis Report for Estimating Dose from Cattle , Vegetable Consumption and Inhalation Pathways Utilizing Contaminated Water from the Top of the Salado, Culebra, and Selected Marker Beds for an Undisturbed Case Supporting Review Compliance Certification Application

Document Version 1.01

WBS # 1.2.01.5.3.1

Leo J. Rahal ____ 6849 ____ 2/26/97
Leo J. Rahal Signature Organization Date
Principal Investigator

Hong-Nian Jow ____ 6848 ____ 2/26/97
Hong-Nian Jow Signature Organization Date
Department Manager

Hong-Nian Jow for Leo Gomez ____ 6849 ____ 2/26/97
Leo S. Gomez Signature Organization Date
Technical Reviewer

Deborah Coffey ____ 6811 ____ 2/26/97
Deborah Coffey Signature Organization Date

Wendell D. Weart ____ 6000 ____ 2/26/97
Wendell Weart Signature Organization Date
Managerial Reviewer

Margaret Chu ____ 6801 ____ 2/26/97
Margaret Chu Signature Organization Date
Deputy Program Manager Wipp

TABLE OF CONTENTS

List of Tables 3

1.0 Introduction 4

1.1 Purpose of this Analysis and Background Information 4

1.2 Bounding Analysis 7

1.3 Dose Calculation Results 9

2.0 Summary of Compliance with the Individual Protection Standard 10

3.0 *Software Used for Analysis* 11

3.1 Point of Contact 11

4.0 Calculational Procedure 12

5.0 References 13

Appendix A: Listing of Input and Output Data Files for Dose Calculation: Marker Bed 139 (Table A-1) 14

Appendix B: Graph of Mass Loading Data Taken from 1991 to 1996 for Lea County 15

Appendix C: Procedure for Extraction of NUTS data used in this analysis 16

List of Tables

Table		Page
1-1	Maximum Concentrations of Radionuclides (Undiluted) Within the Salado Interbeds at the Disposal System Boundary occuring at 10,000 yrs. after Closure. (Modified from Table 8-1, CCA)	6
1-2	GENII-A Input Parameters for Farm Family Scenario	8
1-3	GENII-A Input Parameters for Cattle Rancher Scenario	8
1-4	GENII-A Input Parameter for Farm Family Inhalation Pathway	8
1-5	Calculated Annual Committed Effective Doses at 10,000 yrs. after closure	9
A-1	Listing of Input and Output Data Files for Dose Calculations: Marker Bed 139	14

1.0 Introduction

This analysis report summarizes the background, analysis procedure and results for the Waste Isolation Pilot Plant (WIPP) Repository dose calculation resulting from (1) consumption of beef cattle drinking water from a stockpond utilizing a contaminated ground-water source by a rancher residing at the location of the well; (2) consumption of crops irrigated from a contaminated ground-water source by a farm family residing at the location of the well; and (3) inhalation of resuspended irrigated soil. The requirements and standards which form the guidance for this calculation are set forth in the following paragraphs. This analysis represents a continuation of the drinking water pathway calculation, reported in Chapter 8 of the CCA.

1.1 Purpose of this Analysis and Background Information

The purpose of this analysis is to provide quantitative analysis of pathways for human exposure to radionuclides which potentially may be released through MB 139 (and Mbs 138, a & b) at the site boundary during 10,000 years of undisturbed performance. The CCA contained results of a conservative and bounding analysis of the hypothetical doses from consumption of contaminated drinking water, (Chapter 8, CCA). This analysis uses the same conservative and bounding assumptions used in the CCA to examine alternative exposure pathways. Specifically, this analysis presents dose calculations for ingestion pathways for beef and irrigated crops, and for the inhalation of resuspended particles of irrigated soil. These analyses were performed to address comments raised by the EPA in their review of the CCA **[EPA-1]**.

The undiluted sources for this analysis appear in Table 1-1. A dilution factor of 32.4 was used for the purpose of achieving a potable water supply. This was based on a recommendation of 10,000 ppm (mg/L) TDS for potential drinking water sources contained in 40 CFR 191 (1993) . However, in the NALCO Water Handbook **[NA-1]** a value of 500 ppm (mg/L) TDS is indicated as the level for potable water.

Three hundred realizations of the modeling system were generated during the CCA analyses for the containment requirements (Chapter 6, CCA). These same realizations were also used for individual and groundwater protection requirements (Chapter 8, CCA). These 300 realizations are comprised of three sets (or replicates) of one hundred realizations each, generated using the Latin Hypercube sampling technique. Of the 300 realizations, none show any radionuclides reaching the top of the Salado through the sealed shafts.

Nine of the 300 realizations show concentrations of radionuclides greater than zero reaching the accessible environment through the anhydrite interbeds. All of the remaining 291 realizations show that no radionuclides reach the accessible environment during the regulatory time frame of 10,000 years after repository closure through the anhydrite interbeds. A receptor in the accessible environment could not come in contact with the anhydrite interbeds located at a depth greater than 2000 feet. Table 1-1 shows the maximum concentrations of radionuclides calculated by the modeling evaluation as reaching the accessible environment in the nine non-zero realizations. The full range of estimated values for radionuclide concentrations is from zero to the values shown in Table 1-1. The maximum concentration values shown in Table 1-1 occur 10,000 years after closure. These are the same values used in Chapter 8, CCA.

For the purposes of this analysis, the maximum concentration set, Replicate 3 Vector 64, was used to determine doses for the cattle, vegetable consumption, and inhalation pathways since this represents the largest concentration and gave the largest dose from drinking water as reported in Chapter 8, Table 8-2, CCA.

40 CFR Part 194.51 states that doses must be estimated for an individual who resides at the location in the accessible environment where that individual would be expected to receive the highest exposure from radionuclide releases from the disposal system. All potential pathways for exposure associated with the undisturbed performance of the repository must be assessed (40 CFR § 194.52).

Table 1-1. Maximum Concentrations of Radionuclides (Undiluted) Within the Salado Interbeds at the Disposal System Boundary Occuring at 10,000 yrs. after Closure(Modified from Table 8-1, CCA)

Realization No.	Vector No.[1]	Concentration (Curies/liter)				
		^{241}Am	^{239}Pu	^{238}Pu	^{234}U	^{230}Th
1	**Replicate 1 Vector 46**	**1.36×10^{-17}**	**4.33×10^{-12}**	**N [2]**	**5.82×10^{-13}**	**2.10×10^{-14}**
2	**Replicate 2 Vector 16**	**N**	**5.13×10^{-14}**	**N**	**6.77×10^{-15}**	**1.89×10^{-17}**
3	**Replicate 2 Vector 25**	**N**	**1.35×10^{-15}**	**N**	**1.65×10^{-16}**	**7.00×10^{-18}**
4	**Replicate 2 Vector 33**	**1.32×10^{-17}**	**7.18×10^{-14}**	**N**	**9.76×10^{-15}**	**9.36×10^{-16}**
5	**Replicate 2 Vector 81**	**N**	**6.23×10^{-18}**	**N**	**N**	**N**
6	**Replicate 2 Vector 90**	**N**	**5.20×10^{-16}**	**N**	**7.40×10^{-17}**	**N**
7	**Replicate 3 Vector 3**	**3.50×10^{-18}**	**3.08×10^{-13}**	**N**	**4.32×10^{-14}**	**1.07×10^{-16}**
8	**Replicate 3 Vector 60**	**5.98×10^{-17}**	**7.41×10^{-14}**	**N**	**9.09×10^{-15}**	**2.30×10^{-15}**
9	**Replicate 3 Vector 64**	**5.42×10^{-17}**	**5.85×10^{-12}**	**N**	**7.61×10^{-13}**	**4.68×10^{-15}**
10-300	**-**	**N**	**N**	**N**	**N**	**N**

1. The procedure used to extract these values from the NUTS data is described in Appendix C.

2. Values less than 10^{-18} curies per liter are considered to be negligible (N) relative to the other values and are not reported.

1.2 Bounding Analysis

Uncertainty in the calculation of radionuclide concentrations in the anhydrite interbeds is described in Section 6.1.2 ([**CCA-1**]) . Additional uncertainty is involved in the calculation of doses resulting from the specified exposure pathways. Given this uncertainty, the DOE has elected to perform a bounding analysis using assumptions that do not represent reality, but that would result instead in a bounding estimate that is much greater than any reasonably expected dose to a receptor. If this unrealistic bounding analysis results in calculated doses to the receptor that are below the regulatory limit, compliance with the standard can be demonstrated.

The bounding analysis used for this assessment is based on the following factors and assumptions:

1. No specific transport mechanism is postulated. Instead, all of the contaminants reaching the accessible environment within the anhydrite interbeds during the year of maximum releases (10,000 years after closure) within the 10,000 year period, are assumed to be available to a receptor.

2. Brine derived from the anhydrite interbeds has total dissolved solids (TDS) concentrations of about 324,000 parts per million [**BR-1**]; this represents a concentration that is too high to be consumed by humans. For the bounding analysis, the calculation includes the dilution of this brine by a factor of 32.4 to a TDS concentration of 10,000 parts per million.

3. The resulting annual committed effective dose is calculated based on a 50-year dose commitment. Calculations were performed using the GENII-A dose code (Appendix GENII, CCA). A 50-year dose commitment is selected because this period is specified in Appendix B of 40 CFR Part 191.

4. The parameters associated with the individual receptors for each scenario appear in Tables 1-2, 1-3, and 1-4. Data related to food pathways, irrigation and inhalation were selected as representative values typical of the associated activities [**NRC-1**], [**DOE-1**], except where noted.

Table 1-2 GENII-A Input Parameters for Farm Family Scenario-Terrestrial Food Consumption Utilizing Irrigation from Ground Water Source.

Food Type	Grow Time Days	Irrigation Rate (cm/yr)	Time months	Yield kg/m2	Consumption Holdup (days)	Consumption Rate (kg/yr)
Leaf	90	100	6	1.5	14	15
Root	90	100	6	4.0	14	140
Fruit	90	100	6	2.0	14	64
Grain	90	100	6	0.8	180	72

Table 1-3 GENII-A Input Parameters for Cattle Rancher Scenario.

Food Type	Consumption Rate (kg/yr)	Consumption Holdup (days)	Drinking Water Contamination Fraction	Diet Fraction	Grow Time days	Stored Feed Irrigation Rate (cm/yr)	Stored Feed Time (months)	Stored Feed Yield (kg/m3)	Stored Feed Storage Time (days)
Beef	70	34	1	1	90	100	6	0.8	180

Table 1-4 GENII-A Input Parameters for Farm Family Inhalation Pathway

Breathing Rate	270 cm^3/sec. (Chronic)
Inhalation Period	8760 hours/yr.
Mass Loading Factor	1.0E-04 gm/m^3

The mass loading factor is based on data representative of regional resuspension data for the 1991 to 1996 time period , **Appendix-B** and [**AIRS-1**]. Section 194.51 states that DOE shall assume that an individual resides at the single geographic point where that individual would receive the highest dose. With the bounding analysis, the DOE complies with the intent of this criterion, but the specific location of the receptor is not identified because all of the contaminants reaching the accessible environment within the anhydrite interbeds during the year of maximum releases are assumed to be directly available to the receptor, regardless of the location of the receptor. The well from which the receptor drinks is assumed to be located such that the contaminants reaching the anhydrite interbeds are delivered directly to the well. This well is the source of the stockpond from which the cattle drink and from which irrigation for feed and vegetable crops is obtained. Additionally, an inhalation calculation for the farm family represents a pathway by which dried irrigated soil is resuspended above the farm area and inhaled by the farm inhabitants. The data used in this analysis appear in Tables 1-2, 1-3, and 1-4. Data related to food pathways, irrigation, and inhalation were selected as representative values typical of the associated activities **[NRC-1]**, **[DOE-1]**, except where noted .

The bounding analysis dose calculation was performed using the GENII-A code, Version 2.10 **[GEN-1]**. This program runs on the DEC Alpha System. Appendix GENII of the CCA **[CCA-1]** describes the modeling method. GENII-A incorporates dose-calculation guidance provided in Appendix B of 40 CFR Part 191.

1.3 Dose Calculation Results

The maximum doses calculated to result from the releases listed in Table 1-1 after applying the factors and assumptions listed above, are shown in Table 1-5. Because of the conservative and unrealistic assumptions underlying the analysis, the bounding doses are greater than any realistic doses that could be delivered to a receptor. The calculated bounding doses are well below the regulatory standard, which is an annual committed effective dose of 15 millirem. The full range of estimated radiation doses is from zero to some value less than the bounding values shown in Table 1-5

Table 1-5 Calculated Annual Committed Effective Doses at 10,000 yrs. after Closure

Scenario .	Annual Committed Effective Dose (millirem)
Farm Family Inhalation	**3.1×10^{-4}**
Farm Family Ingestion	**4.6×10^{-1}**
Cattle Rancher	**3.3×10^{-8}**

For comparison, the maximum dose reported in the CCA for the drinking water pathway is 4.7×10^{-1} millirem/yr, (Table 8-2 of the CCA).

2.0 Summary of Compliance with the Individual Protection Standard

In performing the compliance assessment, the DOE applied a bounding-analysis approach using unrealistic assumptions that result in the over-estimation of potential doses and contaminant concentrations. This conservative approach assumes that all contaminants reaching the accessible environment are directly available to a receptor. Using this very conservative approach, the calculated maximum potential dose to an individual would be about one-thirtieth of the individual protection standard.

3.0 Software Used for Analysis

NUTS **[NU-1]**

GENII-A , **Version 2.10 [GEN-1], [GEN-2], [GEN-3]**

This program runs on the DEC Alpha System under VMS Operating system. This analysis was performed by Leo J. Rahal, the code sponsor.

3.1 Point of Contact

Code Sponsor: **GENII-A** Leo J. Rahal , Org. 6849, Geo-Centers Inc. (505) 766-9629 The GENII-A code was run on the DEC Alpha system using VMS. The calculations were performed by Leo J. Rahal , the code sponsor.

Radionuclide source data were obtained from the **NUTS [NU-1]** output through the **Compliance Assessment Methodology Controller** (CAMCON) library data access process. All calculations were performed within the **Configuration Management System** (CMS) environment to ensure QA procedures are followed.

4.0 Calculational Procedure

The following information describes the calculation procedure used to evaluate doses at selected times and locations. These input and output files are listed in Appendix A, Table B-1. The Description column indicates the location, vector number and time for the input data obtained from the NUTS code. The output files are generated by running these input files according to the command given in Appendix A, i.e.@run_gi2 100 where 100 is the number associated with the data file gi2_calc100.inp for example. For the purpose of this analysis the time selected was 10,000 years after closure. 10,000 years is the cut-off time for required calculations. The 10,000 year period represents the time at which maximum values occur within the required 10,000 year period. The GENII-A **[GEN-1]** dose code determined the dose corresponding to this selected time.

5.0 References

[AIRS-1] U.S. Environmental Protection Agency. 1992. National Air Data Branch. **Aerometric Information Retrieval System.** Total suspended particulate information for LEA County.

[BR-1] Data taken from BRAGFLO output file relating to brine inventory at 10,000 years after closure for Replicate 1, Scenario 1 (Undisturbed Case), Vector 46. The file from which these data were taken is as follows: **DISK$TINA_CCA, 19963:[BF.JDMILLE.CCA, 1996.POSTALG.R1S1]OSTALG_CCA, 1996_R1_S1_V046.CDB**

[CCA-1] Compliance Certification Application for the Waste Isolation Pilot Plant, United States Department of Energy Waste Isolation Pilot Plant, Carlsbad Area Office, Carlsbad, New Mexico, October 29, 1996.

[DOE-1] Performance Assessment Task Team Progress Report . Radioactive Waste Technical Support Program, May 1994. DOE/LLW-157 Revision 1.

[EPA-1] Letter from EPA , Mary D. Nichols to Alvin Alm, Department of Energy. December 19, 1996

[GEN-1] WIPP PA User's Manual for GENII-A, Version 2.10. Document Version 1.00, WPO # 27751. November 13, 1995.

[GEN-2] Napier, B.A., R.A. Peloquin, D.L. Strenge and J.V. Ramsdell. 1988. GENII- *The Hanford Environmental Radiation Dosimetry Software System. Vol. 1: Conceptual Representation.* PNL-6584, Vol. 1. Richland, WA: Pacific Northwest Laboratory.

[GEN-3] Napier, B.A., R.A. Peloquin, D.L. Strenge and J.V. Ramsdell. 1988. GENII- *The Hanford Environmental Radiation Dosimetry Software System Vol. 2: User's Manual. PNL-6584, Vol. 1. Richland, WA: Pacific Northwest Laboratory.*

[NA-1] The NALCO Water Handbook, Frank N. Kemmer, Nalco Chemical Company. McGraw-Hill Book Company. 1987. Pg. 35.2 Table 35.1A.

[NRC-1] U.S. Nuclear Regulatory Commission, "Calculation of Annual Doses to Man from Routine Releases of Reactor Effluents for the Purpose of Evaluating Compliance with 10 CFR Part 50, Appendix I, " Regulatory Guide 1.109 (1997).

[NU-1] WIPP PA User's Manual for NUTS, Version 2.02. Document Version 1.00, WPO # 37927. May 28, 1996.

6.0 Appendices

Appendix A: Listing of Input and Output Data Files for Dose Calculation: Marker Bed 139 (Table A-1)

Table A-1 Input and Output Files for Dose Calculations (1)

INPUT FILE	OUTPUT FILE	DESCRIPTION
1. gi2_calc311.inp	gi2_calc311_trn.out	r3s1v064 10,000 yrs. MB139s (Farm Family- Food Ingestion)
2 gi2_calc312.inp	gi2_calc312_trn.out	r3s1v064 10,000 yrs. MB139s (Cattle Rancher)
3 gi2_calc812.inp	gi2_calc812_trn.out	r3s1v064 10,000 yrs. MB139s (Farm Family-Inhalation)

Footnote 1:
To run these data files type the following command: RUN_GI2 100 where 100 is the number associated with the data file GI2_CALC100.INP, for example.
An output file GI2_CALC100_TRN.OUT, will be generated.
The edit command EDT filename was used to open and edit input and output files.
Legend:
r3 replicate 3 of NUTS OUTPUT
s1 undisturbed case
v vector number

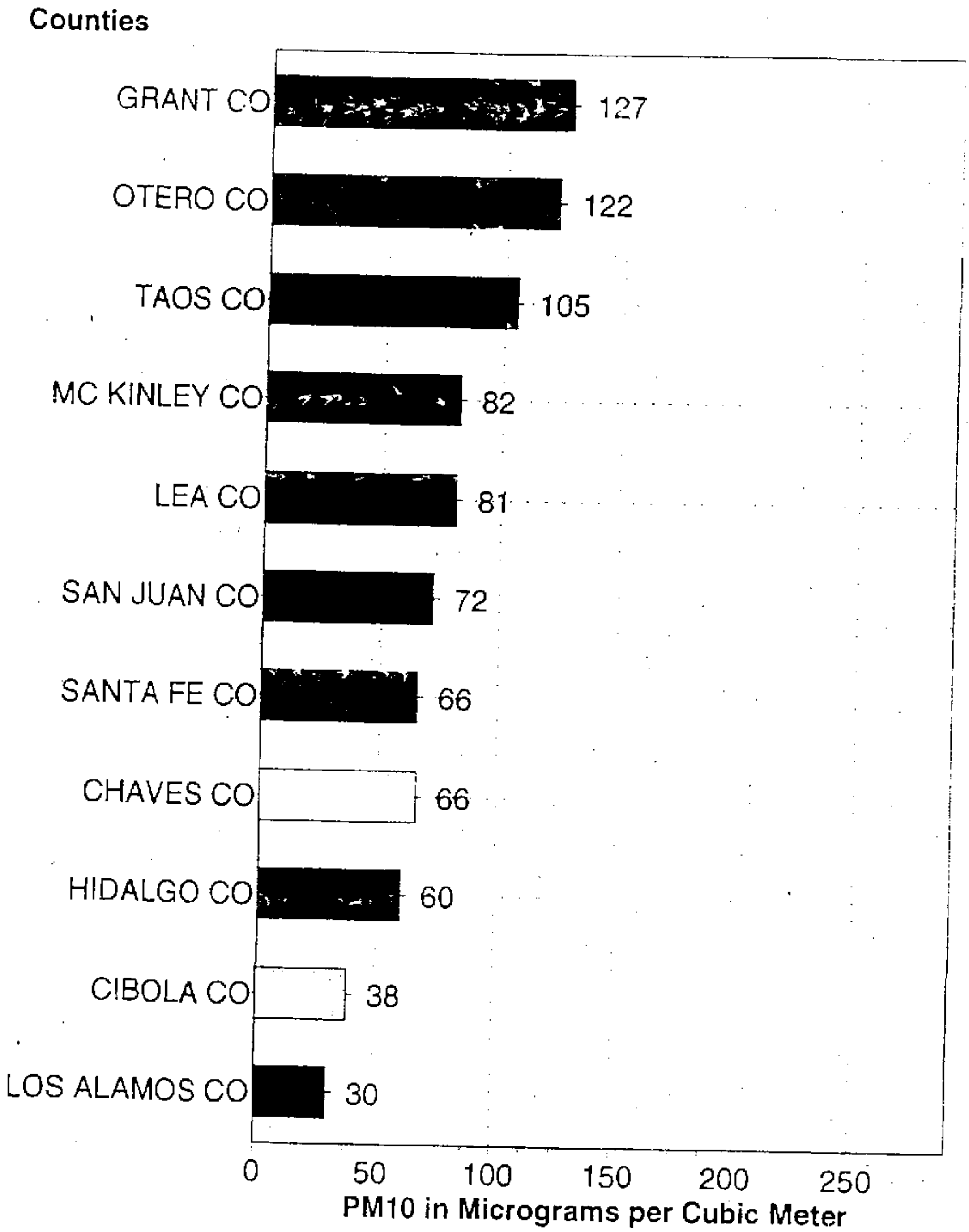
+

Appendix B: Graph of Mass Loading Data Token from 1991 to 1996 for Lea County.

Max Value Chart for NEW MEXICO

PM10 Air Quality Monitors (All Years)

Data Extrac



National Standard = 150.0000

Appendix C: Procedure for Extraction of NUTS data used in this analysis.

The directory U1:[JDMILLE.TEST] contains the following files used in the extraction of data from NUTS:

1. PA_NUTS_ISO_S1_CONC.INP , the equation file , plus the NUTS.CDB ouput file are used to extract data from NUTS.

2. PA_NUTS_ISO_CONC.COM is the command file for ALGEBRACDB **[ALG-1]** used to process these data.

3. The resulting CDB files are :

PA_NUTS_ISO_R1S1_CONC_V046.CDB
PA_NUTS_ISO_R2S1_CONC_V016.CDB
PA_NUTS_ISO_R2S1_CONC_V025.CDB
PA_NUTS_ISO_R2S1_CONC_V033.CDB
PA_NUTS_ISO_R2S1_CONC_V081.CDB
PA_NUTS_ISO_R2S1_CONC_V090.CDB
PA_NUTS_ISO_R3S1_CONC_V003.CDB
PA_NUTS_ISO_R3S1_CONC_V060.CDB
PA_NUTS_ISO_R3S1_CONC_V064.CDB

4. Selected times are obtained through processing by SUMMARIZE **[SUM-1]** , using the input file

PA_NUTS_ISO_S1_CONC.SMZ

5. This output is transferred to an EXCEL spreadsheet form through multiple files, one for each vector. The EXCEL input file, containing all vectors is PA_NUTS_ISO_S1_CONC.TBL.

6. All files have been transferred to the CMS system.

[ALG-1] **ALGEBRACDB** , WPO# 21247, 1995

[SUM-1] **SUMMARIZE** , WPO# 21781, 1995