

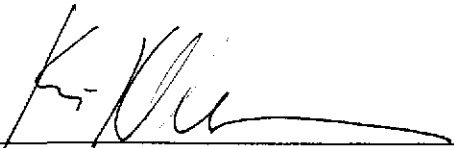
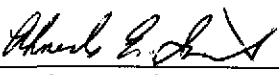
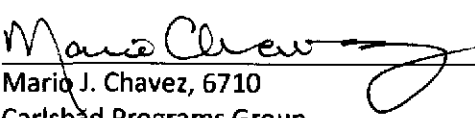
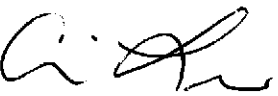
552951

Analysis Report for the CRA-2009 PABC Culebra Flow and Transport Calculations

AP-144

Task Number: 1.2.5

Report Date: 2/3/2010

| | | |
|--------------------|---|-------------------------|
| Author: |  _____ Kristopher L. Kuhlman, 6712 Repository Performance Department | <u>2/3/10</u> Date |
| Technical Review: |  _____ Ahmed E. Ismail, 6711 Performance Assessment Department | <u>2/3/2010</u> Date |
| QA Review: |  _____ Mario J. Chavez, 6710 Carlsbad Programs Group | <u>2/3/2010</u> Date |
| Management Review: |  _____ Christi Leigh, 6712 Manager, Repository Performance Department | <u>2/3/2010</u> Date |

Contents

List of Figures.....4

List of Tables.....7

1 Introduction.....9

2 Background.....10

3 Culebra T-Field Mining Modifications and Flow-field Calculations.....12

 3.1 Overview12

 3.2 Approach13

 3.2.1 Modeling Assumptions.....13

 3.2.2 Potash Mining Parameter Modifications.....13

 3.2.3 Software and Run Control.....14

 3.2.4 Radionuclide Transport Calculations.....15

 3.3 Model Domain and Discretization.....15

 3.3.1 Boundary and Initial Conditions.....17

 3.3.2 Determination of Potential Mining Areas.....19

 3.3.3 Use of Mining Zones in Forward Simulations.....22

 3.3.4 Particle Tracking Simulations22

 3.4 Particle Tracking Results.....23

 3.4.1 Particle Travel Times23

 3.4.2 Flow Directions.....25

 3.4.3 Particle Speeds29

 3.5 Particle Tracking Discussion32

4 Radionuclide Transport Calculations.....34

 4.1 Background and Theoretical Overview34

 4.1.1 Relationship Between Flow and Transport Modeling Domains.....34

 4.1.2 Flow-Field Extraction.....34

 4.1.3 VTRAN2 Output Summary.....37

 4.1.4 Solute Transport Modeling.....39

 4.1.5 Model Parameters.....41

 4.2 Transport Results.....45

 4.2.1 Radionuclide Transport Under Partial-Mining Conditions46

 4.2.2 Radionuclide Transport Under Full-Mining Conditions.....48

4.3 Transport Summary.....51

5 Summary and Conclusions52

6 Bibliography.....54

Appendix 1 Mining Modification Process Narrative.....58

 A1.1 Step 0 – Shapefile Conversion and Mapping to Grid58

 A1.2 Step 1 – MODFLOW File Checkout and Directory Re-structuring64

 A1.3 Step 2 – Enlarge Mining Areas.....66

 A1.4 Step 3 – Perform Mining Modifications69

 A1.4.1 Modification of Boundary Conditions70

 A1.4.2 Modification of Transmissivity Fields.....72

 A1.5 Step 4 – Run MODFLOW and DTRKMF for Each Realization.....77

 A1.6 Step 5 – Post-Process DTRKMF Output for Plotting.....79

 A1.7 Step 6 – Convert MODFLOW Input Data to Transport Mesh.....81

 A1.8 Step 7 – Run MODFLOW Using the Finer Transport Mesh84

 A1.9 Step 8 – Convert Binary MODFLOW Budget Files to ASCII for VTRAN285

 A1.9.1 Binary to ASCII Budget File Conversion85

Appendix 2 SECOTP2D Transport Process Narrative.....87

 A2.1 Step 0 – Sampling of Subjectively Uncertain Parameters88

 A2.2 Step 1 – Mesh Generation and Material Property Assignment89

 A2.3 Step 2 – Uncertain Parameter Assignments89

 A2.4 Step 3 – Evaluation of Oxidation-State-Dependent Parameters92

 A2.5 Step 4 – Tabulation of Mining Factors and Flow-Field Indices.....95

 A2.6 Step 5 – Flow-Field Extraction.....96

 A2.7 Step 6 – Transport Calculations.....97

Appendix 3 VTRAN2 Utility.....100

 A3.1 VTRAN2 Build Info100

 A3.2 VTRAN2 Verification.....106

 A3.2.1 VTRAN2 Verification Test Case 1.....106

 A3.2.2 VTRAN2 Verification Test Case 2.....108

Appendix 4 Script Source Listing..... 111

 A4.1 Python Script convert_shapefile_to_ASCII.py..... 111

 A4.2 MATLAB Script import_polyline_determine_mined_areas.m..... 114

A4.3 Bash Shell Script run_mining_mods.sh..... 115
A4.4 Python Script enlarge_mining_areas.py..... 117
A4.5 Python Script mining_mod.py..... 120
A4.6 Bash Shell Script link_input_run_mf_dtrk.sh 124
A4.7 Python Script combine_dtrkmf_output_for_gnuplot.py..... 126
A4.8 Python Script extract_dtrkmf_lwb_travel_times.py..... 128
A4.9 Python Script 100x100_to_50x50.py 130
A4.10 Bash Shell Script run_50x50_modflow.sh 134
A4.11 Bash Shell Script convert_rename_modflow_50x50_budget.sh..... 136
A4.12 Python Script budget_bin2ascii.py 137

List of Figures

Figure 3-1. Generalized stratigraphy near WIPP14
Figure 3-2. PABC-2004 (black dashed line) and PABC-2009 (solid blue line) MODFLOW domains, SECOTP2D transport domain (red), and WIPP LWB shown for comparison..... 16
Figure 3-3. Initial and boundary heads for non-constant-head portion of domain (left) and entire domain (right) (Hart et al., 2009). Coordinates are cell-based.....17
Figure 3-4. Comparison of minable potash to the flow and transport modeling domains; green hatched area from BLM shapefile (Cranston, 2009) 18
Figure 3-5. Stencil used to expand areas of predicted potash (red cell with M) to model cells affected by mining-related subsidence from 45° angle of draw (blue cells with A) 19
Figure 3-6. Definitions of mining-affected areas in full-mining scenario between current and previous models. Base image is Figure 3.2 from Lowry and Kanney (2005).....20
Figure 3-7. Definitions of partial-mining-affected areas between current and previous applications; base image is Figure 3.3 from Lowry and Kanney (2005).....21
Figure 3-8. Comparison of minable potash distribution inside WIPP LWB for PABC-2004 (dark gray) and PABC-2009 (translucent green). The WIPP repository plan is shown for comparison, from Figure 3.6 of Lowry and Kanney (2005)22
Figure 3-9. CDF of advective particle travel times from the center of the WIPP waste panels to the WIPP LWB for full, partial, and non-mining scenarios.....24
Figure 3-10. Comparison of advective particle travel time CDFs for PABC-2009, PABC-2004, and CCA.25
Figure 3-11. Particle tracks for non-mining scenario; black box is WIPP LWB, green circles are Culebra monitoring well locations.....26
Figure 3-12. Particle tracks for R1. Small magenta squares and black crosses indicate centers of MODFLOW cells located within potash and mining-affected areas, respectively; thin black line is minable potash definition.26
Figure 3-13. Particle tracks for R2. Small magenta squares and black crosses indicate centers of MODFLOW cells located within potash and mining-affected areas, respectively; thin black line is minable potash definition.27

Figure 3-14. Particle tracks for R3. Small magenta squares and black crosses indicate centers of MODFLOW cells located within potash and mining-affected areas, respectively; thin black line is minable potash definition.27

Figure 3-15. Histograms of particle x-coordinates at exit point from LWB; full and partial-mining include all three replicates (note different vertical scales between plots; no mining contains 100 particles while mining scenarios include 300 particles).....28

Figure 3-16. Magnitude of Darcy flux for r440 R2 for no, partial, and full-mining scenarios using cell-based coordinates; LWB (black) and SECOTP model domain (red) shown for reference.....29

Figure 3-17. Particle speeds for non-mining scenario computed from DTRKMF results.....30

Figure 3-18. Particle speeds for R1, computed from DTRKMF results.....31

Figure 3-19. Particle speeds for R2, computed from DTRKMF results.....31

Figure 3-20. Particle speeds for R3, computed from DTRKMF results.....31

Figure 3-21. Mining factor and travel time to WIPP LWB for full-mining scenario (all replicates).....32

Figure 3-22. Mining factor and travel time to WIPP LWB for partial-mining scenario (all replicates)33

Figure 4-1. MODFLOW volumetric flux and Darcy velocity; A_y is perpendicular to the arrow pointing left; A_x is perpendicular to the arrow pointing right35

Figure 4-2. Velocity transfer between MODFLOW and SECOTP2D meshes36

Figure 4-3. Ensemble average and standard deviation of Darcy flow velocity magnitude $|V|$ and direction V_{dir} from VTRAN2 (each figure is constructed from 300 realizations); SECOTP2D cell-based coordinates. Outer black line is the LWB, small black box outlines the WIPP panels, with the yellow star at the release point (C-2737).....38

Figure 4-4. Dual-Porosity Conceptual Model39

Figure 4-5. Histograms showing distribution of cumulative releases greater than 10^{-9} kg, each x-axis is \log_{10} cumulative release (kg), each y-axis is number of vectors per bin (bins are one log cycle wide).....47

Figure 4-6. \log_{10} -distribution of ^{241}Am at 10,000 years for the vector with the largest cumulative release of ^{241}Am to the LWB (dashed line); SECOTP2D cell-based coordinates49

Figure 4-7. \log_{10} -distribution of ^{239}Pu at 10,000 years for the vector with the largest cumulative release of ^{239}Pu to the LWB (dashed line); SECOTP2D cell-based coordinates50

Figure 4-8. \log_{10} -distribution of ^{234}U at 10,000 years for the vector with the largest cumulative release of ^{234}U to the LWB (dashed line); SECOTP2D cell-based coordinates50

Figure 4-9. \log_{10} -distribution of ^{230}Th (daughter product) at 10,000 years for the vector with the largest cumulative release of ^{230}Th to the LWB (dashed line); SECOTP2D cell-based coordinates51

Figure 4-10. \log_{10} -distribution of ^{230}ThA (released ^{230}Th) at 10,000 years for the vector with the largest cumulative release of ^{230}ThA to the LWB (dashed line); SECOTP2D cell-based coordinates.....51

Figure A1-1. 54 polygons defining mining areas, color-coded by their relationship to the large black polygon. Green are outside the large polygon and define mined areas, red are inside the black polygon and define non-mined areas. Blue rectangle indicates PABC-2009 MODFLOW model extents.....63

Figure A1-2. MODFLOW model domain with mined model cells marked with a small blue dot. 34,440 of the total 87,188 elements have mining (some cells in the northwest portion of the domain are inactive – see Figure 3-2)64

Figure A1-3. Directory structure of MiningMod CVS repository; r??? indicates the 100 realizations (not numbered sequentially) selected in Hart et al., (2009) after simplifying the directory structure (see Table A1-6).....65

Figure A1-4. Dependency between scripts used in mining modifications on Pentium 4 PA cluster. In the middle column, the execution order is from top to bottom. The two gray boxes indicate scripts only used for post-processing DTRKMF output.65

Figure A1-5. Matrices of full, partial, and unexpanded mining factors (region corresponding to Table A1-9 marked with dashed line); maroon cells have minable potash in both mining scenarios, purple indicates the halo of cells affected by mining that are not directly mined, orange cells are only mined in the full-mining scenario (with the corresponding halo of affected full-mining only cells in dark gray), and the background of model cells are unaffected by mining.69

Figure A1-6. MODFLOW IBOUND arrays for original, partial-mining, and full-mining cases. Green is inactive or no-flow (zero), brick red is active (one), blue is specified head (negative integers). The only difference between full- and partial-mining scenarios is near row 175, column 150. ...70

Figure A1-7. Overlay of three original, partial-mining, and full-mining IBOUND arrays given separately in Figure A1-6. The only difference between full and partial mining is a small tan sliver near row 175, column 150.....71

Figure A1-8. Comparison of r440 R2 for non-mined, fully mined, and partially mined scenarios (same color scale in all three plots). Coordinates are cell-based (each cell 100 m on a side); heavy black line indicates location of cross section (Figure A1-9), while LWB (black dashed) and SECOTP2D domain (red) are shown for comparison.76

Figure A1-9. Longitudinal cross section through the WIPP LWB (see Figure A1-8 for location) showing all seven T-fields associated with r440. Dashed black line is original non-mined T field, three solid lines are partial-mining scenarios, three dotted lines are full-mining scenarios. Light vertical lines indicate extent of LWB.....77

Figure A1-10. Comparison of initial head arrays for original and expanded grids. First 100 columns of first row (north boundary) in the original (red circles) and first two rows of the expanded (blue dots) model83

Figure A1-11. VTRAN2 output (\log_{10} Darcy velocity in m/s) corresponding to a single MODFLOW realization (r440). Left column is full mining, right column is partial mining; mining factors: R1=663.4, R2=77.07, R3=902.6. Color scale is the same between all figures, relative vector length scaling is not.....86

Figure A2-1. Culebra transport calculation flowchart.....88

Figure A3-1. Mesh for VTRAN2 verification 107

List of Tables

| | |
|--|----|
| Table 3-1. UTM NAD27 Zone 13 coordinates defining extents of model domains..... | 16 |
| Table 3-2. Particle tracking travel time statistics (years); PABC-2004 statistics from Table 3.22 of Lowry and Kanney (2005) | 24 |
| Table 4-1. Deterministic physical transport parameters..... | 42 |
| Table 4-2. Uncertain physical parameters | 43 |
| Table 4-3. Deterministic chemical parameters | 44 |
| Table 4-4. Uncertain chemical parameters | 45 |
| Table 4-5. Number of vectors exceeding the 10^{-9} kg cumulative release threshold during the 10,000 year simulation..... | 46 |
| Table 4-6. Maximum cumulative release (kg) within each group of 100 vectors at 10,000 years | 47 |
| Table 4-7. Vector number corresponding to maximum cumulative release to WIPP LWB over 10,000 years (see Table 4-6) | 48 |
| Table 4-8. Median concentration at 10,000 years across all 100 vectors for each species and mining scenario | 48 |
| Table A1-1. Commercial off-the-shelf software used in mining modification analysis | 58 |
| Table A1-2. Input and output files for Python script <code>convert_shapefile_to_ASCII.py</code> | 59 |
| Table A1-3. Listing of the first five data points in the files <code>polyline_dump_matlab_part0000.dat</code> and <code>polyline_dump_for_gnuplot.dat</code> | 60 |
| Table A1-4. Input and output files for MATLAB script <code>import_polyline_determine_mined_areas.m</code> | 60 |
| Table A1-5. Directory listing of input scripts, input shapefile, intermediate files, and output files for shapefile conversion (step 0). Files are located in a zip archive saved in the CVS repository MiningMod in the package Auxiliary. | 62 |
| Table A1-6. Listing of file keepers (original column) with equivalent short name (short column) after modification by <code>run_mining_mods.sh</code> | 66 |
| Table A1-7. Input and output file for Python script <code>enlarge_mining_areas.py</code> | 67 |
| Table A1-8. Mining indices used in <code>expand_mining_areas.py</code> | 67 |
| Table A1-9. Excerpts from <code>mining_areas_matrix.dat</code> (top), <code>New_Full_09.dat</code> (middle), and <code>New_Part_09.dat</code> (bottom) for rows 101-129 and columns 71-121 (area including NW corner of WIPP LWB). 0 is not mined or affected, 1 is mined or affected by mining (see Figure A1-5 for location of area)..... | 68 |
| Table A1-10. Input and output files for Python script <code>mining_mod.py</code> | 71 |
| Table A1-11. Mining factors applied to each MODFLOW realization. Keepers id refers to the order the realizations are listed in the keepers file (Table A1-6). First five columns are sorted by MODFLOW realization name, last five columns sorted by order in keepers file. | 72 |
| Table A1-12. Mining factors and MODFLOW flow simulation corresponding to each SECOTP2D vector and replicate..... | 74 |
| Table A1-13. Input and output files for MODFLOW flow calculations (100 m × 100 m grid) | 78 |
| Table A1-14. Input and output files for DTRKMF particle tracking calculations (100 m × 100 m grid) | 79 |

| | |
|--|-----|
| Table A1-15. Input and output files for DTRKMF post-processing Python scripts..... | 81 |
| Table A1-16. Input and output files for Python script 100x100_to_50x50.py | 82 |
| Table A1-17. Northwest corner of Culebra top elevation field; top half is original field (4x4), bottom half is expanded field corresponding to same area (8x8). Four and 16 cells are colored-coded to illustrate correspondence. | 82 |
| Table A1-18. Initial head along north edge of model domain (row 1) in original (top) and expanded (bottom) starting head..... | 83 |
| Table A1-19. Section of IBOUND arrays at southeast corner of domain for 100 m grid (top) and 50 m grid (bottom). -1 (in red) indicates specified head; 1 indicates an active cell..... | 84 |
| Table A1-20. Filenames of corresponding MODFLOW input files for 100 m and 50 m grid problems..... | 84 |
| Table A1-21. General structure of a data block in a MODFLOW budget files..... | 85 |
| Table A2-1. Step 1 input and output files..... | 89 |
| Table A2-2. Step 2 input and output files..... | 90 |
| Table A2-3. Sampled values of uncertain SECOTP2D physical parameters for all three replicates..... | 91 |
| Table A2-4. Sampled values of uncertain K_d parameter; OXSTAT indicates whether the U and Pu species are in their high or low oxidation states for each vector and replicate..... | 93 |
| Table A2-5. Step 3 script, executables, input, and output files..... | 95 |
| Table A2-6. Step 4 script, executables, input and output files..... | 96 |
| Table A2-7. Step 5 script, executable, input and output files..... | 97 |
| Table A2-8. Step 6 script, executables, input and output files (general case)..... | 98 |
| Table A2-9. Step 6 modified input runs..... | 99 |
| Table A2-10. Step 6 modified input run file names | 99 |
| Table A3-1. VTRAN2 . F source listing | 100 |
| Table A 3-2. VTRAN2 VMS build and link commands..... | 105 |
| Table A3-3. VTRAN2 build info | 105 |
| Table A3-4. VTRAN2 test files..... | 106 |
| Table A3-5. VTRAN2_TEST_1 .CMD..... | 107 |
| Table A3-6. VTRAN2_TEST_1 .BUD..... | 108 |
| Table A3-7. VTRAN2_TEST_1 .VEL..... | 108 |
| Table A3-8. VTRAN3_TEST_1 .DBG..... | 108 |
| Table A3-9. VTRAN2_TEST_2 .CMD..... | 109 |
| Table A3-10. VTRAN2_TEST_2 .BUD..... | 109 |
| Table A3-11. VTRAN2_TEST_2 .VEL..... | 110 |
| Table A3-12. VTRAN2_TEST_2 .DBG..... | 110 |

1 Introduction

The Waste Isolation Pilot Plant (WIPP), located in southeastern New Mexico, has been developed by the U.S. Department of Energy (DOE) for the geologic (deep underground) disposal of transuranic (TRU) waste. Containment of TRU waste at the WIPP is regulated by the U.S. Environmental Protection Agency (EPA) according to the regulations set forth in Title 40 of the Code of Federal Regulations (CFR), Part 191 (U.S. EPA, 1993). The DOE demonstrates compliance with the containment requirements according to the Certification Criteria in Title 40 CFR Part 194 (U.S. EPA, 1996) by means of performance assessment (PA) calculations performed by Sandia National Laboratories (SNL). WIPP PA calculations estimate the probability and consequence of potential radionuclide releases from the repository to the accessible environment for a regulatory period of 10,000 years after facility closure. The models are maintained and updated with new information as part of a recertification process that occurs at five-year intervals after the first waste is received at the site.

PA calculations were included in the 1996 Compliance Certification Application (CCA) (U.S. DOE, 1996), and in a subsequent Performance Assessment Verification Test (PAVT) (MacKinnon and Freeze, 1997a; 1997b; 1997c). Based in part on the CCA and PAVT PA calculations, the EPA certified that the WIPP met the containment criteria in the regulations and was approved for disposal of transuranic waste in May 1998 (U.S. EPA, 1998). PA calculations were also an integral part of the 2004 Compliance Recertification Application (CRA-2004). During their review of the CRA-2004, the EPA requested an additional PA calculation, referred to as the CRA-2004 Performance Assessment Baseline Calculation (PABC) (Leigh et al., 2004), be conducted with modified assumptions and parameter values (Cotsworth, 2005).

Since the CRA-2004 PABC (PABC-2004), additional PA calculations were completed for and documented in the 2009 Compliance Recertification Application (CRA-2009). The CRA-2009 PA resulted from continued review of the PABC-2004, including a number of technical changes and corrections, as well as updates to parameters and improvements to the PA computer codes (Clayton et al., 2008). The EPA has requested that additional information, which was received between the commencement of the CRA-2009 PA (December 2007) and the submittal of the CRA-2009 (March 2009), be included in an additional PA calculation (Cotsworth, 2009), referred to as the CRA-2009 Performance Assessment Baseline Calculation (PABC-2009). The PABC-2009 analysis is guided by analysis plan AP-145 (Clayton, 2009a). This report documents the Culebra flow and radionuclide transport analysis for the PABC-2009 and is compared with the results from the PABC-2004 PA. The EPA requested (Cotsworth, 2009) the PABC-2009 transport calculations utilize the newly conceptualized hydrologic model from AP-114 Task 5 (Hart et al., 2008) and the newly calibrated transmissivity fields from AP-114 Task 7 (Hart et al., 2009). The EPA also requested (Kelly, 2009) the PABC-2009 use revised K_d ranges in the Culebra radionuclide transport calculations.

This analysis report documents the efforts to evaluate the effects of potential future potash mining on the flow and transport in the Culebra Member of the Rustler Formation, in the vicinity of the Waste Isolation Pilot Plant (WIPP), approximately 26 miles southeast of Carlsbad, New Mexico. It is an update of analyses done for the PABC-2004 (Lowry and Kanney, 2005), which updates CCA analyses (Ramsey and Wallace, 1996).

Radionuclide movement through the Culebra is a function of the groundwater flow-field and the transport properties of the radionuclide species being considered. Groundwater flow velocity and direction are highly dependent upon the magnitude and the spatial variability of hydraulic transmissivity (the T-field). WIPP PA considers the potential that future potash mining in the McNutt potash zone of the Salado Formation underlying the Rustler Formation will cause subsidence in the Culebra and hence increase Culebra transmissivity.

The Culebra flow and transport analysis in this report begins with results from Latin Hypercube Sampling (LHS) (Kirchner, 2010) and the calibrated Culebra T-fields (Hart et al., 2009). Cumulative releases of radionuclides through the Culebra to the accessible environment (the Culebra outside the WIPP land withdrawal boundary (LWB)) are inputs to CCDFGF (Camphouse, 2010). Discussion of the data flow between the different elements of PA can be found in the PABC-2009 summary report (Clayton et al., 2010).

The work reported herein was conducted under the analysis plan for the Calculation of Culebra Flow and Transport PABC-2009 (AP-144), (Kuhlman, 2009), which is a subset of the work broadly described by AP-145, the analysis plan for the CRA-2009 Performance Assessment Baseline Calculation (Clayton, 2009a)

This report documents the analysis performed regarding the release mechanism related to groundwater flow and transport through the Culebra. The analysis report associated with AP-114 Task 5 (Hart et al., 2008) documents the development of the 1000 base T-fields (before calibration) for the Culebra and the analysis report associated with AP-114 Task 7 (Hart et al., 2009) documents the subsequent calibration and selection of 100 calibrated realizations from the base T-fields for use in the PA analysis outlined here. Background information on groundwater flow and solute transport within the Culebra is presented in Section 2. The effects of potash mining on the flow-field (Section 3 and Appendix 1) and the transport of radionuclides through these flow-fields (Section 4 and Appendix 2) are both covered in this report. Results of the flow and transport calculations are summarized and conclusions are drawn in Section 5.

2 Background

The WIPP repository is located approximately 26 miles (42 kilometers) southeast of Carlsbad, New Mexico. The disposal horizon of the WIPP is approximately 2,150 feet (655 meters) below the ground surface in the Salado Formation of the Delaware Basin. The Salado is regionally extensive, consisting predominantly of halite, a low permeability evaporite (Powers et al., 1978).

The Rustler Formation is located stratigraphically above the Salado Formation and is of particular importance in estimating the potential for radionuclide releases from the WIPP because it contains the most hydraulically transmissive units above the repository. In the vicinity of the WIPP, the Rustler consists of evaporite units interbedded with carbonates and siliciclastic units (Vine, 1963; Holt and Powers, 1988). The Culebra Dolomite Member has been identified as the most transmissive unit in the Rustler Formation and consequently it is the most likely pathway for subsurface transport of radionuclides from the WIPP panels to the accessible environment outside the LWB.

The Culebra is an approximately 7.75 meter thick fractured dolomite with nonuniform properties in both the horizontal and vertical directions (Holt, 1997). There are multiple scales of porosity and permeability within the Culebra ranging from microfractures to potentially large vuggy zones. Flow occurs through fractures, vugs and, to some extent, through intergranular pores. The large permeability contrast between the different scales of interconnected porosity suggests a dual porosity conceptualization consisting of advective porosity (i.e., fracture porosity) and diffusive porosity (i.e., matrix porosity). The advective porosity is the void space comprised of the highly transmissive portions of the rock such as large open fractures and/or interconnected vugs. The diffusional porosity represents the intragranular porosity potentially including microfractures or vugs. Tracer tests conducted in the Culebra at the WIPP site have shown characteristics of both advective transport and matrix diffusion (Meigs et al., 2000).

Although the Culebra properties vary in the vertical direction, the error introduced by modeling the Culebra in two rather than three dimensions has been determined to be negligible for the objective of groundwater flow and radionuclide transport calculations (Corbet, 1995).

Potash mining in the WIPP area involves resource extraction below the Culebra dolomite in the McNutt Potash zone, a subsection of the larger Salado Formation that underlies the Rustler Formation. It is hypothesized that subsidence of the Culebra due to potash mining leads to fracturing and unconsolidation, resulting in higher Culebra transmissivities. This increase in transmissivity may significantly change the regional groundwater flow pattern in the Culebra and additionally the transport of any nuclides entering the aquifer from the underlying repository. WIPP PA includes mining scenario calculations to estimate the impact potash mining may have on groundwater flow and radionuclide transport.

The key steps in estimating radionuclide transport through the Culebra include:

1. Construction of base geostatistical realizations of Culebra hydraulic transmissivity (T), anisotropy, storativity, and recharge fields (collectively referred to as "T-fields");
2. Calibration of the T-fields to observed heads;
3. Modifying the T-fields to account for potential subsidence due to potash mining beneath the Culebra;
4. Calculating steady-state groundwater flow-fields for each mining-modified T-field; and
5. Calculating radionuclide transport through the Culebra for each flow-field. Culebra transport simulations calculate the cumulative discharge at the WIPP LWB over the 10,000-year regulatory period due to a source located at the center of the waste panel area. The source releases 1 kg for each of several radionuclides over the first 50 years of the simulation.

The work reported herein includes steps 3-5. The construction of base geostatistical realizations of Culebra T-fields (step 1) was covered in AP-114 Task 5 (Hart et al., 2008), and the calibration of these T-fields (step 2) was covered in AP-114 Task 7 (Hart et al., 2009).

3 Culebra T-Field Mining Modifications and Flow-field Calculations

3.1 Overview

The PABC-2009 Culebra T-field mining modifications and flow-field calculations largely follow the procedure used in the PABC-2004 (Lowry and Kanney, 2005), with two exceptions: 1) a new definition of the region containing minable potash is used, and 2) the new T-fields developed and calibrated as part of AP-114 Tasks 5 and 7 are used as inputs. The procedure for this analysis is summarized below:

1. Obtain the sampled values for the random mining modification factor (100 vectors x 3 replicates);
2. Map potential areas of future potash mining onto the groundwater modeling domain for both full- and partial-mining scenarios;
3. Apply the mining modification factor to the 100 stochastically calibrated T-fields from AP-114 Task 7 (Hart et al., 2009), producing 600 mining-modified T-fields (100 vectors x 2 mining scenarios x 3 replicates);
4. Perform steady-state flow simulations for each of the 600 mining-modified T-fields using MODFLOW-2000 (also known as MODFLOW or MF2K);
5. Perform particle tracking using the new mining-affected flow-fields to determine advective travel times to the LWB; and
6. Refine the flow-field to a finer grid size and re-execute MODFLOW to create flow-fields as input for the Culebra radionuclide transport calculations.

This analysis report represents the latest effort in characterizing mining effects in the Culebra and highlights the differences and additions relative to past calculations (Ramsey and Wallace, 1996; Lowry, 2003a; 2003b; Lowry, 2004). The reader is encouraged to review the past documents for further background information.

The PABC-2009 models two categories of mining-impacted transmissivity fields: partial mining with only mining outside the LWB and full mining with regions both inside and outside the LWB mined.

Starting with the 100 stochastically calibrated T-fields from Hart et al., (2009), T-fields are modified to reflect the effects of mining by multiplying the transmissivity value in cells that lie within designated mining zones by a random factor uniformly sampled between 1 and 1000. The range of this factor is set by the EPA in regulation 40 CFR 194.32(b) (U.S. EPA, 1996). The scaling factor for each T-field is provided from Latin Hypercube Sampling (LHS), (Kirchner, 2010).

A forward steady-state flow simulation is run for each new T-field under each mining scenario (full and partial) across three replicates of mining factors, resulting in 600 simulations. Particle tracking is performed on both the 100 original and 600 modified flow-fields to compare the flow path and groundwater travel time from a point above the center of the WIPP disposal panels to the LWB. CDFs are produced for each mining scenario and compared to the undisturbed scenario. The CDFs describe the probability of a conservative tracer (i.e., a "marked" water particle) reaching the LWB at a given

time. In addition to comparing travel times, particle-tracking directions are also examined to determine the effect on the regional flow direction in the WIPP area due to mining.

The parameter fields for the mining scenarios are then copied to a finer grid and the resulting flow-fields are passed to the transport analysis, which performs radionuclide transport modeling in the Culebra.

3.2 Approach

3.2.1 Modeling Assumptions

Besides assumptions inherent in all modeling exercises (e.g., physical processes can be adequately parameterized and estimated on a numerical grid), there are several assumptions that are specific and important to the PABC-2009 analysis. Those assumptions are as follows:

1. It is assumed that the boundary conditions along the model domain edges are known and are not dependent on mining (internal fixed-head cells are modified). The reasoning for this assumption is described in Section 3.3.1.
2. It is assumed that the flow field, over the duration of the particle tracking and transport times, can be represented by steady-state conditions. This is related to the first assumption; the *boundary conditions are also assumed to remain constant over time*. This assumption is necessary since data do not exist that can predict the transient conditions in the Culebra at the site over the time frames involved (>10,000 years).
3. It is assumed that the mining effects can be adequately represented with a single mining factor that increases the transmissivity uniformly across the potential mining zones within the Culebra. This is directed by EPA regulation 40 CFR 194.32(b) (U.S. EPA, 1996).

Further assumptions related to the flow modeling and calibration of the T-fields can be found in AP-114 Task 5 (Hart et al., 2008) and Task 7 (Hart et al., 2009).

3.2.2 Potash Mining Parameter Modifications

Potash mining in the region surrounding WIPP involves underground excavation in the McNutt Potash zone of the upper Salado Formation, which is located stratigraphically above the WIPP repository horizon, but below the Culebra Member of the Rustler Formation (see Figure 3-1). It is hypothesized that subsidence due to collapse of the underground voids created in the McNutt potash zone during mining will lead to increased permeability in the Rustler Formation, due to increased fracturing. The purpose of the mining scenario calculations is to determine the impact of potash mining on groundwater flow directions and transport velocities in the Culebra. This analysis largely represents a re-application of the methods used in PABC-2004 (Lowry and Kanney, 2005), with a few minor exceptions:

1. The definition of the regions where minable potash is believed to exist, obtained from the Bureau of Land Management (BLM) (Cranston, 2009), has been updated.
2. The configuration of the MODFLOW model that mining modifications are being applied to has changed (see Figure 3-2):
 - a. The eastern limit of the model domain has moved 6 km east.

- b. The no-flow boundary condition present along the western edge of the PABC-2004 model has been eliminated in the southern half of the model domain.
 - c. The “halite-sandwiched” region of the Culebra (east of either the M2/H2 or M3/H3 Rustler halite margins) is now constant head.
 3. The way the mining-modified areas interact with internal boundary conditions of the flow model has changed, due to the change in the boundary conditions (there were no internal boundary conditions in the PABC-2004 MODFLOW model).

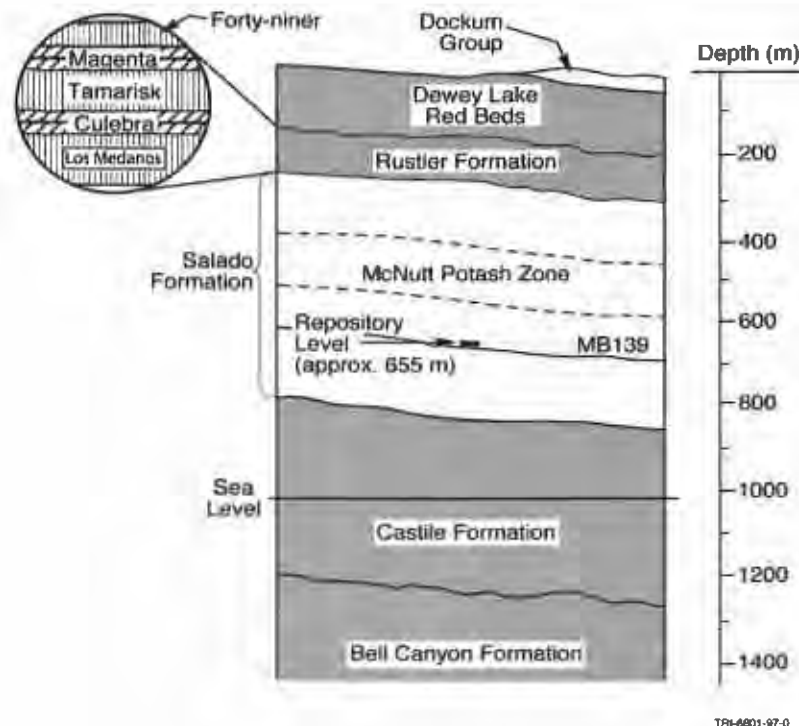


Figure 3-1. Generalized stratigraphy near WIPP

3.2.3 Software and Run Control

The mining-modified Culebra flow-fields and particle tracks were produced on the WIPP PA Pentium 4 Cluster using MODFLOW (Harbaugh et al., 2000), DTRKMF (Rudeen, 2003), and several utility codes (see Appendix 4 for code listings).

With the exception of the conversion of the potash definition shapefile to ASCII, the calculations were performed under formal run control (Long, 2010). The run control scripts extracted executables and input files from access-controlled repositories, performed the calculations in access-controlled directories, and stored the output files in access-controlled repositories. The Run Control Coordinator performed the calculations using a reserved account (`run_mast`) on the cluster. Input and output files were archived using the CVS version control system.

The mining modification simulations were run using an input file (`mining_mods.inp`) that drives the run control Python program `ReadScript.py` (Kirchner, 2008). The run control program creates the required directory structure, checks out the required input files, executes the main Bash shell script

(`run_mining_mods.sh`), and checks the results back into CVS. This driving Bash shell script performs five major functions (see Appendix 1 for a detailed narrative of the process):

1. Convert potash definition shapefile to ASCII files indicating mining-affected MODFLOW model cells;
2. Modify MODFLOW model properties in response to mining;
3. Run MODFLOW and DTRKMF for each of 700 scenarios (600 mined + 100 original) using original 100 m grid;
4. Convert MODFLOW input data to 50 m transport grid; and
5. Run MODFLOW for each of 600 mined scenarios using 50 m transport grid, converting output to format required for VTRAN2.

3.2.4 Radionuclide Transport Calculations

The steady-state flow-fields are computed using MODFLOW after adjustments to account for potential effects of potash mining. These flow results are used as inputs to the SECOTP2D solute transport model, which predicts the fate of radionuclides released from a point located directly over the center of the waste disposal panels. This portion of the analysis is largely unmodified from the analysis done for PABC-2004 (Lowry and Kanney, 2005), aside from the changes in K_d ranges and the MODFLOW velocity fields used as inputs.

3.3 Model Domain and Discretization

The eastern limit of the MODFLOW model domain used in the PABC-2009 analysis (Hart et al., 2008) is extended eastward, compared to the MODFLOW domain used in the PABC-2004 analysis (see both boundaries in Figure 3-2). This change was made in order to locate the boundary in an area where halite is present in all of the non-dolomite members of the Rustler Formation, simplifying the specification of the eastern model boundary condition. The new extent of the model domain is 601700 m to 630000 m x and 3566500 m to 3597100 m y (see Table 3-1 for these and other relevant UTM NAD27 Zone 13 coordinates). The MODFLOW flow model domain is aligned with the primary compass directions and is aerially discretized into 100 m square cells, yielding a model that is 284 cells or 28.4 km wide (east-west) by 307 cells or 30.7 km tall (north-south). The Culebra is modeled as a single horizontal layer of uniform 7.75 m vertical thickness.

Table 3-1. UTM NAD27 Zone 13 coordinates defining extents of model domains

| Domain Corner | UTM NAD27 X [m] | UTM NAD27 Y [m] |
|--|-----------------|-----------------|
| <i>MODFLOW Flow Model Domain</i> | | |
| Northeast | 630,000 | 3,597,200 |
| Northwest | 601,700 | 3,597,200 |
| Southwest | 601,700 | 3,566,500 |
| Southeast | 630,000 | 3,566,500 |
| <i>WIPP Land Withdrawal Boundary</i> | | |
| Northeast | 616,941 | 3,585,109 |
| Northwest | 610,495 | 3,585,068 |
| Southwest | 610,567 | 3,578,623 |
| Southeast | 617,015 | 3,578,681 |
| <i>SECOTP2D Transport Model Domain</i> | | |
| Northeast | 617,750 | 3,583,000 |
| Northwest | 610,250 | 3,583,000 |
| Southwest | 610,250 | 3,577,600 |
| Southeast | 617,750 | 3,577,600 |

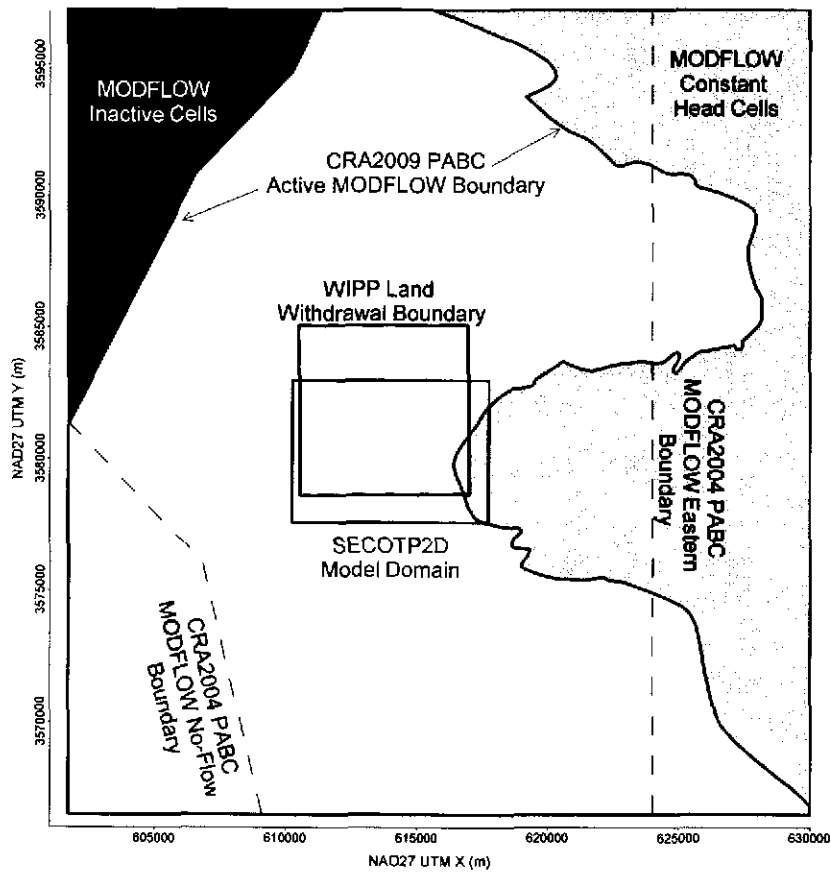


Figure 3-2. PABC-2004 (black dashed line) and PABC-2009 (solid blue line) MODFLOW domains, SECOTP2D transport domain (red), and WIPP LWB shown for comparison.

3.3.1 Boundary and Initial Conditions

Like the model domain and discretization, the boundary and initial conditions used in the PABC-2009 are described fully in AP-114 Task 7 (Hart et al., 2009). Freshwater head data were used from May 2007, consisting of 44 head measurements across the modeling domain; see AP-114 Task 6 (Johnson, 2009) for details regarding the calculation of freshwater head values. A multi-parameter parametric surface is used to fill initial head values everywhere except in the constant-head zone east of the halite margin (Hart et al., 2009).

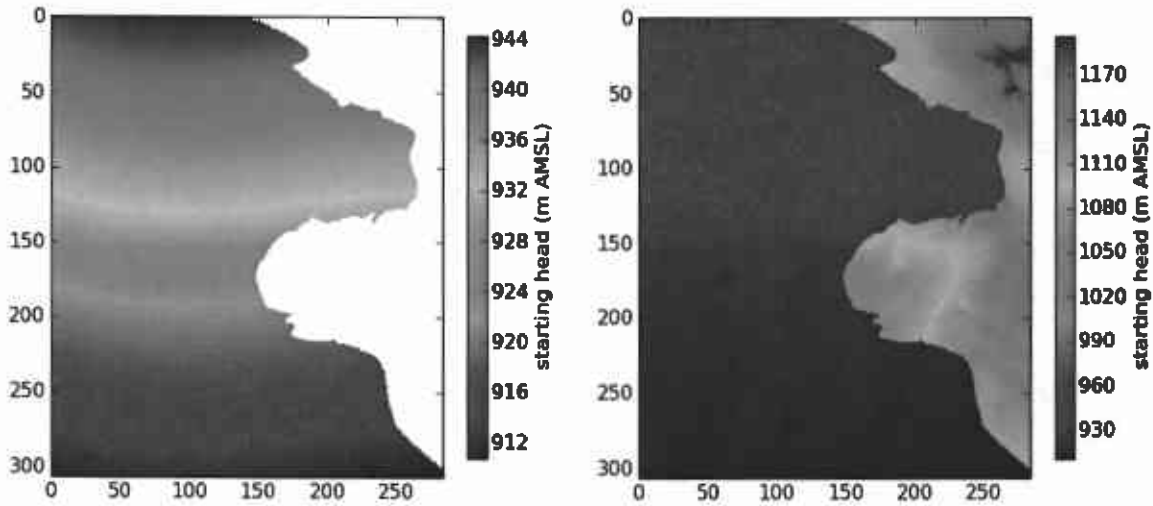


Figure 3-3. Initial and boundary heads for non-constant-head portion of domain (left) and entire domain (right) (Hart et al., 2009). Coordinates are cell-based.

The model boundary conditions along straight-line portions of the north, west, and south edges of the domain are specified head. The parametric function values determining the initial heads are assigned to constant-head cells and kept fixed throughout the simulation. Since simulations are steady state, determination of the initial heads is important only in relation to setting the fixed boundary conditions. The constant-head region associated with the halite-sandwiched portion of the Culebra (light gray region in Figure 3-2) is specified head, set to the land surface elevation (see right plot in Figure 3-3). The piecewise-linear boundary in the northwest corner is a no-flow boundary and is parallel to the Nash Draw groundwater divide (implemented using no-flow cells indicated with dark gray in Figure 3-2). The axis of Nash Draw is interpreted as a regional groundwater divide, draining the Rustler units to the east (and also by symmetry to the west). The initial head contours across the non-constant-head portion of the modeling domain are shown in more detail (only the values specified by the parametric equation) in the left plot of Figure 3-3. The effects of potential potash mining on the boundary conditions must be considered because the extent of minable potash ore extends beyond the modeling domain (see Figure 3-4). Regional flow rates within the flow model are controlled by the boundary conditions and the hydraulic conductivity distribution. The regional gradient across the domain is approximately

$$(943.9 \text{ m} - 911.6 \text{ m})/30.7 \text{ km} = 0.00105 \quad (1)$$

For the current grid, we average the constant heads along the non-halite-sandwiched portion of the northern boundary (columns 1-140, 943.9 m), subtract the average heads along the entire southern boundary (911.6 m), and then divide by the north-south model domain distance (30.7 km). It is assumed that mining impacts would not significantly change this regional gradient and thus the specified initial conditions for the mining scenarios are identical to those in AP-114 Task 7 (Hart et al., 2009). In addition, the CCA, CRA-2004, and PABC-2004 all used this same conceptualization (keeping the outer boundary conditions fixed between the mining and non-mining scenarios); the same conceptualization is maintained to allow for comparisons between the different models.

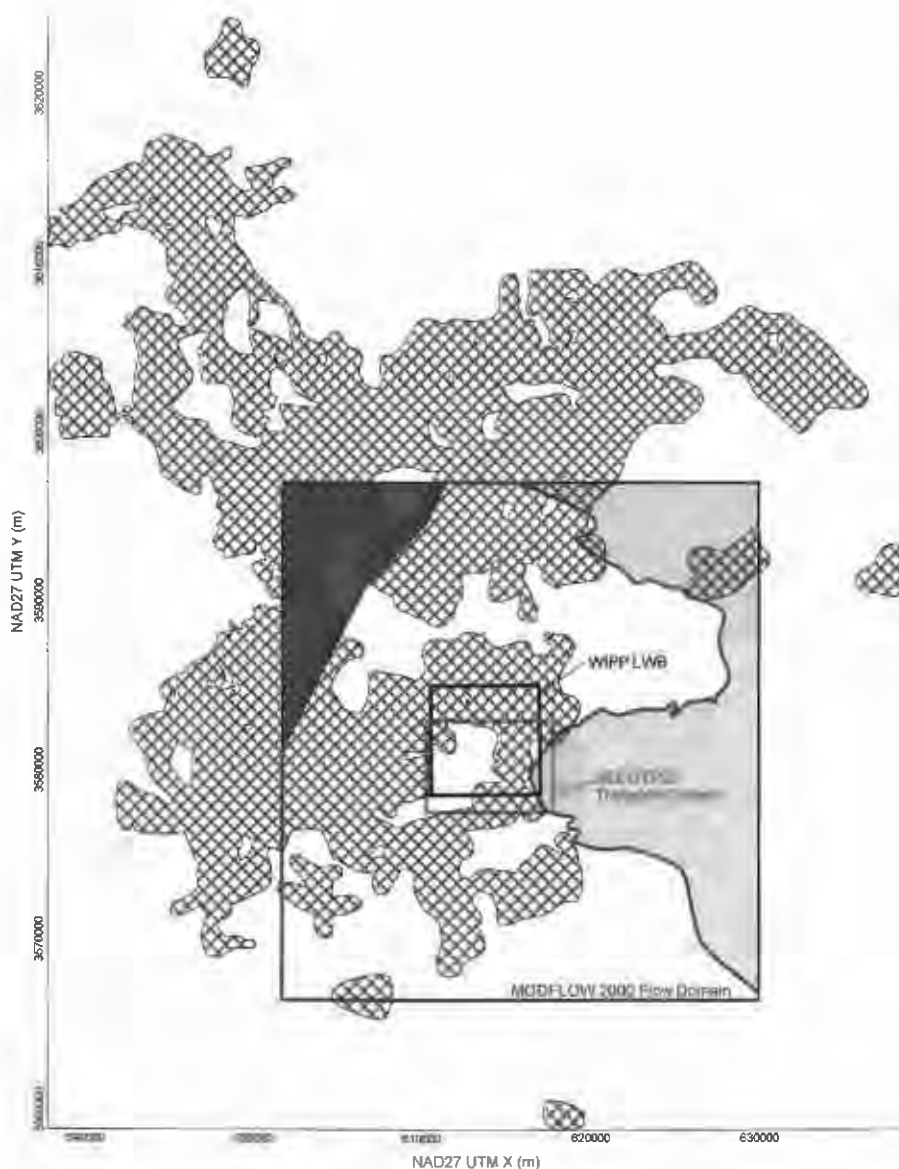


Figure 3-4. Comparison of minable potash to the flow and transport modeling domains; green hatched area from BLM shapefile (Cranston, 2009)

3.3.2 Determination of Potential Mining Areas

The 2009 version of the Bureau of Land Management (BLM) map indicating the distribution of minable potash ore was obtained from BLM as an ESRI shapefile (Cranston, 2009). The conversion of this shapefile to an integer matrix corresponding to Culebra MODFLOW model cells (indicating whether a model cell was affected by mining or not) is explained in Section A1.1.

Since the potash-mining horizon is located in the Salado Formation, below the Culebra, the areas disturbed by mining activities in the Culebra are larger than mined areas in the Salado due to angle-of-draw effects; the subsidence effects do not propagate up vertically, but instead they propagate up and out at 45° angles between horizontal and vertical. The final conclusion is that a 253-m-wide collar was to be added to the mining-impacted areas (Ramsey and Wallace, 1996; Bertram, 1995). This is considered a conservative estimate of the angle-of-draw effects. To accommodate the angle of draw, the mining zone boundaries, as overlaid on the current model grid, were extended outward three cells (300 m) in the *x*- and *y*-directions, and two cells (283 m) in the diagonal directions (see Figure 3-5 for an illustration of the mining-expansion stencil). The PABC-2009 modeling domain and mining zones for the full-mining case are shown in comparison to the 1996 CCA and the CRA-2004 delineations in Figure 3-6. The comparison of the current and previous partial-mining cases is shown in Figure 3-7. A close-up of the WIPP site and the distribution of minable potash is shown in Figure 3-8; it illustrates how the definition inside the WIPP LWB has changed significantly since PABC-2004. For the PABC-2004, the closest minable potash was approximately 1,230 m from the center of the WIPP panels in the southeast direction; for PABC-2009, this distance has reduced to approximately 670 m (in a more easterly direction).

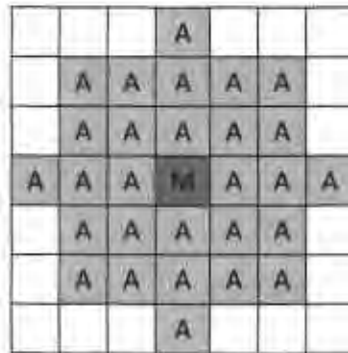


Figure 3-5. Stencil used to expand areas of predicted potash (red cell with M) to model cells affected by mining-related subsidence from 45° angle of draw (blue cells with A)

The output of this mining-area delineation is a file that contains one value for each cell in the grid. A value of 1 means the cell lies within a potential mining-affected zone, and a value of 0 means that it is outside a potential mining-affected zone. See section A1.3 for a detailed narrative description of the calculation of the mining-modified areas.

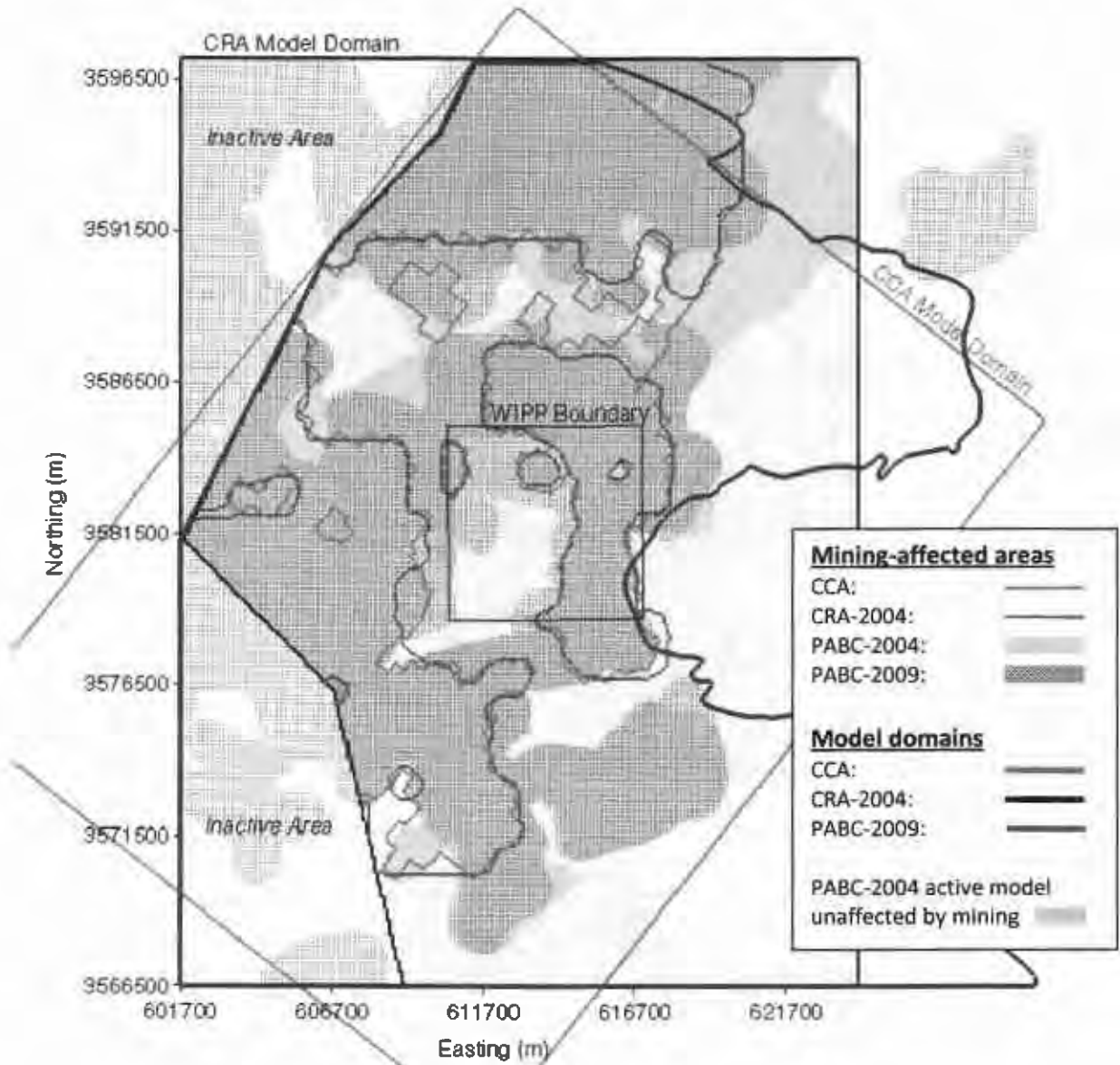


Figure 3-6. Definitions of mining-affected areas in full-mining scenario between current and previous models. Base image is Figure 3.2 from Lowry and Kanney (2005).

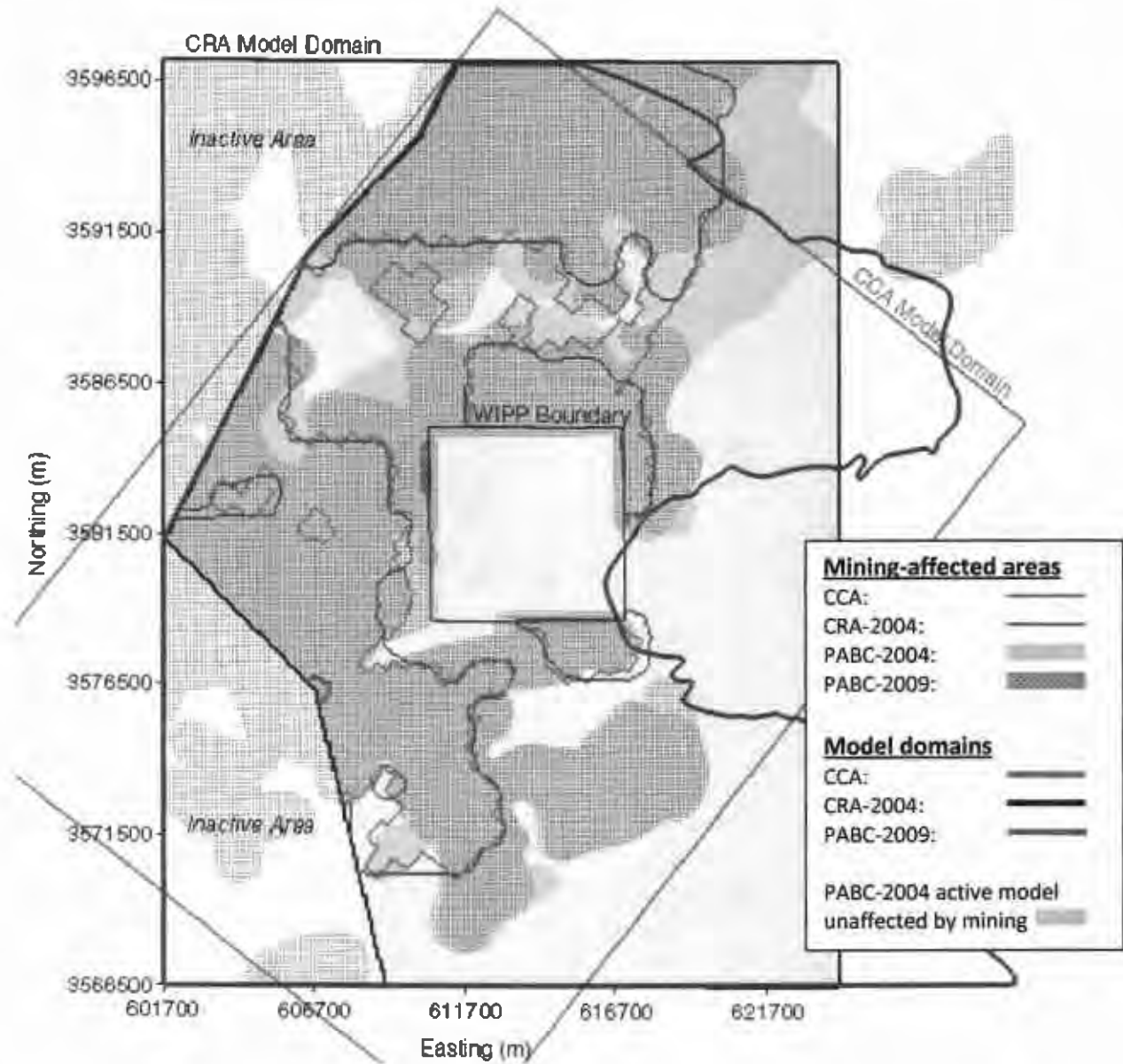


Figure 3-7. Definitions of partial-mining-affected areas between current and previous applications; base image is Figure 3.3 from Lowry and Kanney (2005).

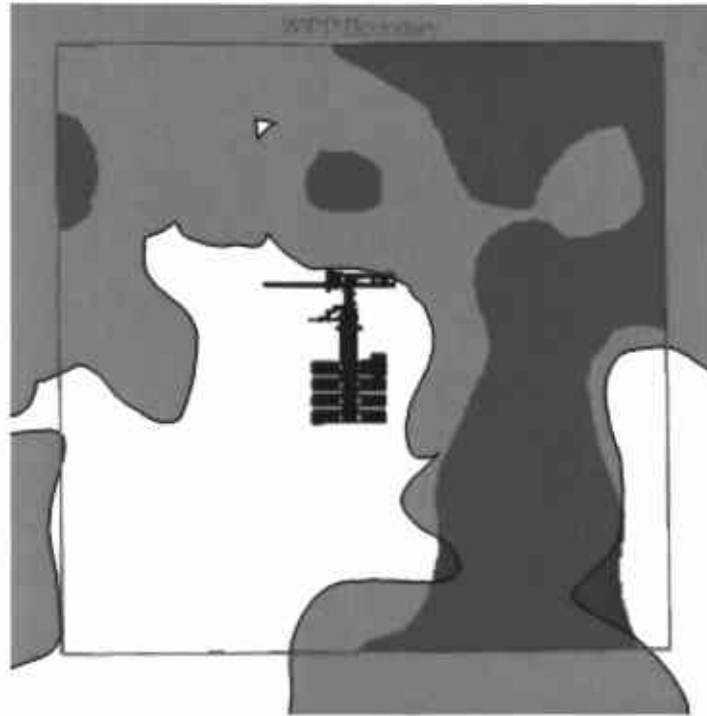


Figure 3-8. Comparison of minable potash distribution inside WIPP LWB for PABC-2004 (dark gray) and PABC-2009 (translucent green). The WIPP repository plan is shown for comparison, from Figure 3.6 of Lowry and Kanney (2005)

3.3.3 Use of Mining Zones in Forward Simulations

The calibration process in Hart et al., (2009) produces 100 sets of transmissivity, horizontal anisotropy, storativity, and areal recharge fields (often referred to simply as “the T-fields” because in previous versions only transmissivity was calibrated) that each minimize the error between observed and model-calculated head distributions. The hydrologic conceptual model produces an ensemble of 1000 uncalibrated parameter fields (Hart et al., 2008). Two hundred of these were randomly sampled from the ensemble and calibrated; the best 100 were selected as the final calibrated set based on their ability to recreate observed head data (Hart et al., 2009). To simulate the effects of mining, each selected T-field is multiplied by its own unique mining scaling factor in areas of potential mining, and MODFLOW is run with these mining-modified T-fields to produce the mining-affected head and flow distributions. The cell-by-cell flow budget files are used in particle tracking and radionuclide transport calculations. To demonstrate stability in mean results, three different sets of mining factors are used, each set forming a replicate (given here as R1, R2, and R3). Thus, for each mining scenario (full and partial), three sets of 100 mining-altered T-fields and related cell-by-cell flow budgets are produced.

3.3.4 Particle Tracking Simulations

In each realization, a single conservative particle is tracked from the UTM NAD27 coordinate $x = 6135975$ m, $y = 35813852$ m (i.e., the location of monitoring well C-2737, directly above the center of the WIPP waste panels) to the LWB (Table 3-1) for each combination of T-field, replicate, and mining scenario using the DTRKMF code. Two main outputs are generated from the suite of particle tracks. First, plots are constructed showing the individual tracks for all 100 T-fields in each scenario for each

replicate (six plots total). This allows visual comparison of the prevailing flow directions for the full- and partial-mining scenarios and the qualitative comparison of the variability of the tracking direction. Secondly, CDFs are constructed for each replicate and scenario, which describe the probability that a water particle will cross the LWB in a given amount of time. The six plots and the CDFs are presented in Section 3.4.

3.4 Particle Tracking Results

Particle tracks were computed using DTRKMF (Rudeen, 2003), which uses the binary cell-by-cell flow budget files produced by MODFLOW-2000. In flow calculations, the full 7.75 m thickness of the Culebra is used, while for transport and particle tracking purposes the thickness is reduced to 4.0 m to focus all flow through the lower, more permeable portion of the Culebra (Holt, 1997). An average value of 16% porosity is used for the particle tracking calculations, as was used in AP-114 Task 7 (Hart et al., 2009). Porosity directly affects transport, but is not needed for the calibration of the flow model.

Particle tracking is performed to allow comparison between the two mining scenarios and the non-mining scenario, which was not used in the SECOTP2D radionuclide transport calculations. The particle tracking results illustrate the advective pathway taken by a marked water particle and do not take into consideration retardation, dispersion, or molecular diffusion. The particle tracks also allow easier comparison of the 600 results (each a 1D trace) in a single plot, in contrast showing 600 sets of concentrations (each a 2D field) produced from SECOTP2D.

3.4.1 Particle Travel Times

Compared to the non-mining scenario (results already given in AP-114 Task 7 (Hart et al., 2009)), the travel times for the partial-mining scenarios are longer, while travel times for the full-mining scenarios are shorter. The median travel time across all three replicates for the full-mining scenario is approximately 0.689 times the median travel time of the non-mined scenario (see Table 3-2, Figure 3-9 and Figure 3-10 for summary statistics and comparison to PABC-2004 results). All advective particle travel times are plotted, but it should be noted that the regulatory limit for radionuclide transport modeling is 10,000 years, taking into consideration retardation, diffusion, and dispersion (which don't apply to particle track modeling). The median travel time across all three replicates for the partial-mining scenario is 3.034 times greater than for the non-mining scenario. For PABC-2004, travel times in both the full- and partial-mining scenarios were slower (longer) than for the non-mining scenario. The CDFs for the full-, partial-, and non-mining scenarios are shown in Figure 3-9.

Table 3-2. Particle tracking travel time statistics (years); PABC-2004 statistics from Table 3.22 of Lowry and Kanney (2005)

| Replicate | Statistic | PABC-2009 | | | PABC-2004 | | |
|-----------|-----------|-----------|---------|-----------|-----------|-----------|-----------|
| | | Full | Partial | No Mining | Full | Partial | No Mining |
| R1 | Median | 5,138 | 22,581 | N/A | 64,026 | 117,815 | N/A |
| | Max | 200,260 | 91,119 | | 2,175,165 | 2,727,191 | |
| | Min | 1,591 | 5,042 | | 2,130 | 5,185 | |
| R2 | Median | 4,956 | 21,999 | | 80,801 | 148,489 | |
| | Max | 94,852 | 84,929 | | 2,059,263 | 1,667,084 | |
| | Min | 1,421 | 5,037 | | 2,463 | 4,855 | |
| R3 | Median | 5,560 | 22,537 | | 74,315 | 118,919 | |
| | Max | 93,172 | 86,758 | | 1,779,512 | 3,128,693 | |
| | Min | 1,421 | 4,505 | | 2,507 | 3,314 | |
| Global | Median | 5,084 | 22,376 | 7,374 | 70,170 | 131,705 | 18,289 |
| | Max | 200,260 | 91,119 | 73,912 | 2,175,165 | 3,128,693 | 101,205 |
| | Min | 1,421 | 4,505 | 2,618 | 2,130 | 3,314 | 3,111 |

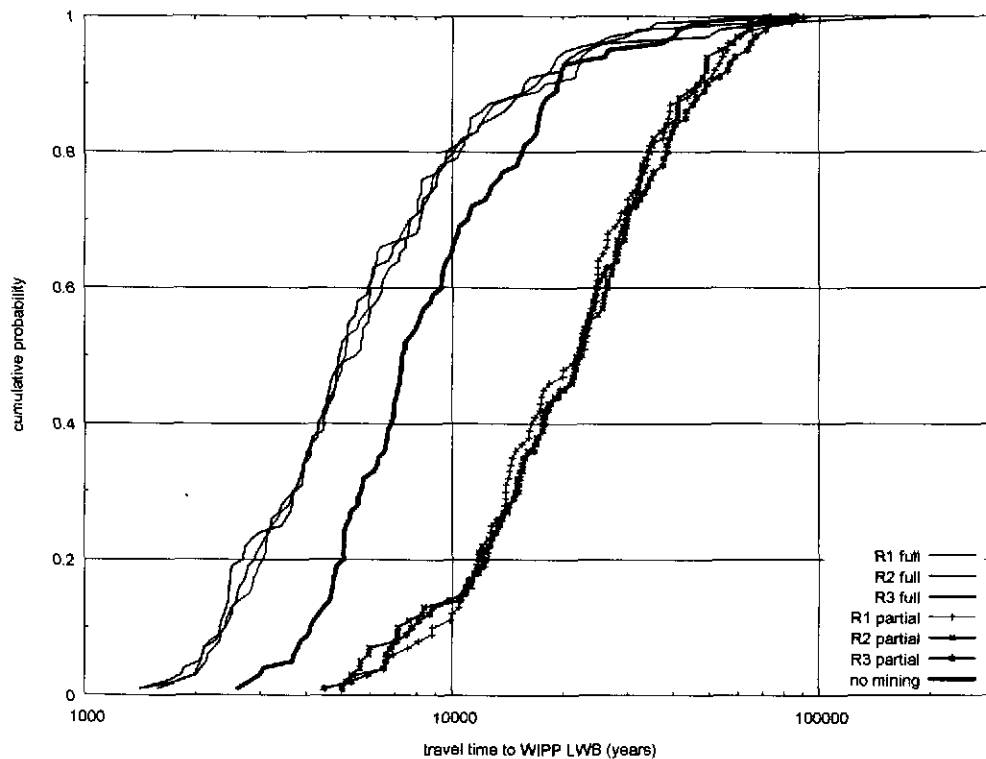


Figure 3-9. CDF of advective particle travel times from the center of the WIPP waste panels to the WIPP LWB for full, partial, and non-mining scenarios

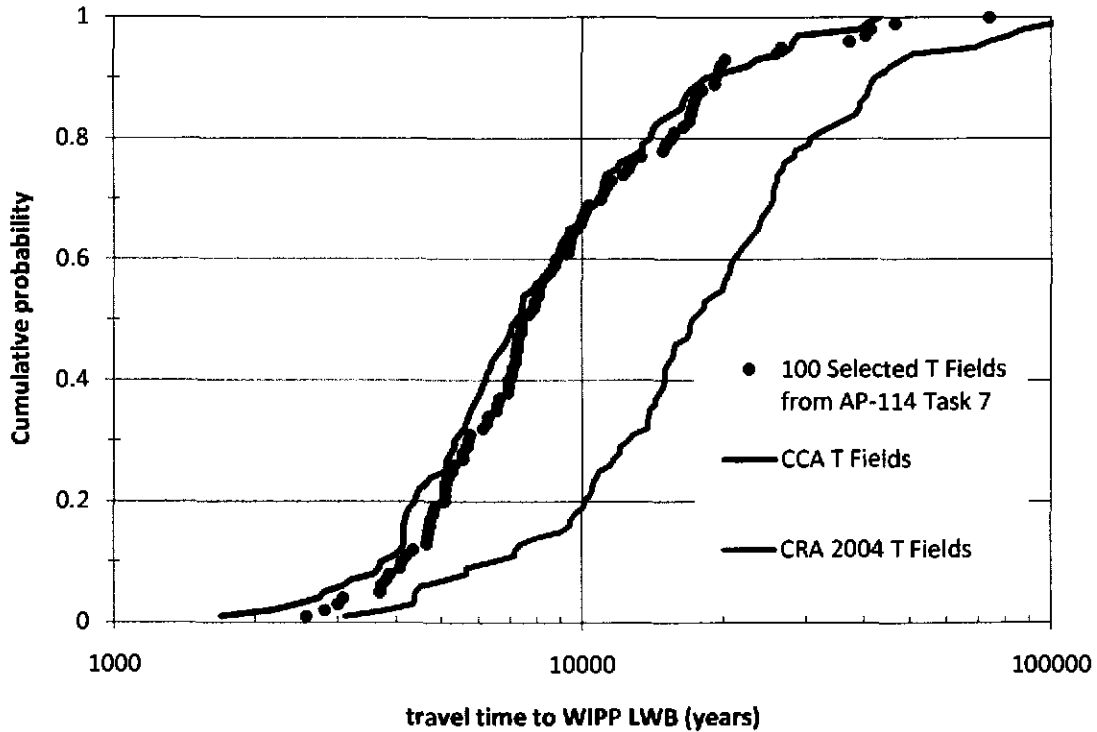


Figure 3-10. Comparison of advective particle travel time CDFs for PABC-2009, PABC-2004, and CCA.

3.4.2 Flow Directions

The particle track directions for the non-, full-, and partial-mining scenarios for the PABC-2009 are illustrated in Figure 3-11 to Figure 3-14. Like past mining scenario calculations (i.e., PABC-2004), there is a strong similarity between the three replicates (R1, R2, and R3) for each scenario (full or partial mining), although the travel directions for the PABC-2009 are different than for the PABC-2004 (Lowry and Kanney, 2005). A larger amount of minable ore inside the WIPP LWB, especially the ore immediately to the east of the particle release point, leads to different effects of full mining on travel times, compared to PABC-2004. Nearly all particles immediately go east to this boundary and then move south along it towards to the edge of the LWB at approximately $x = 612.75$ km (see Figure 3-12 and Figure 3-15). This is in contrast to the partial-mining scenario where the tracking directions are more similar to the non-mining scenario, but more evenly distributed spatially along the southern boundary. In the non-mining scenario, most of the particles exit near the high-transmissivity zone at approximately $x = 615$ km.

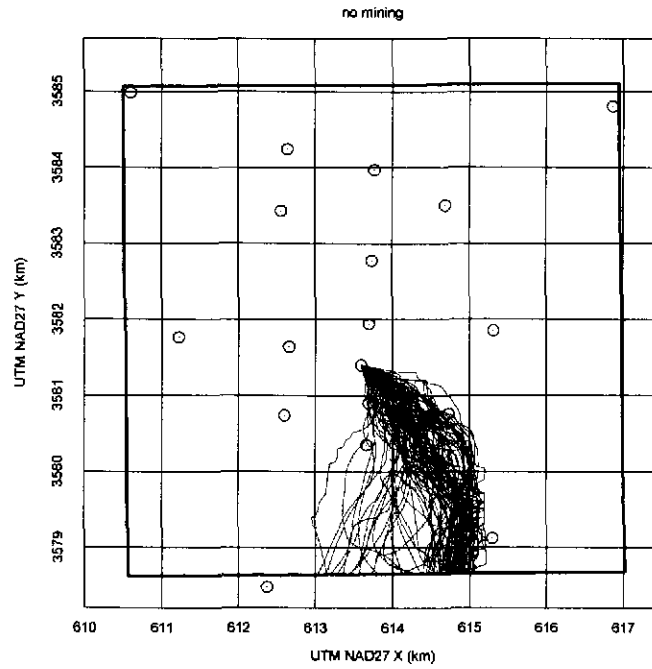


Figure 3-11. Particle tracks for non-mining scenario; black box is WIPP LWB, green circles are Culebra monitoring well locations.

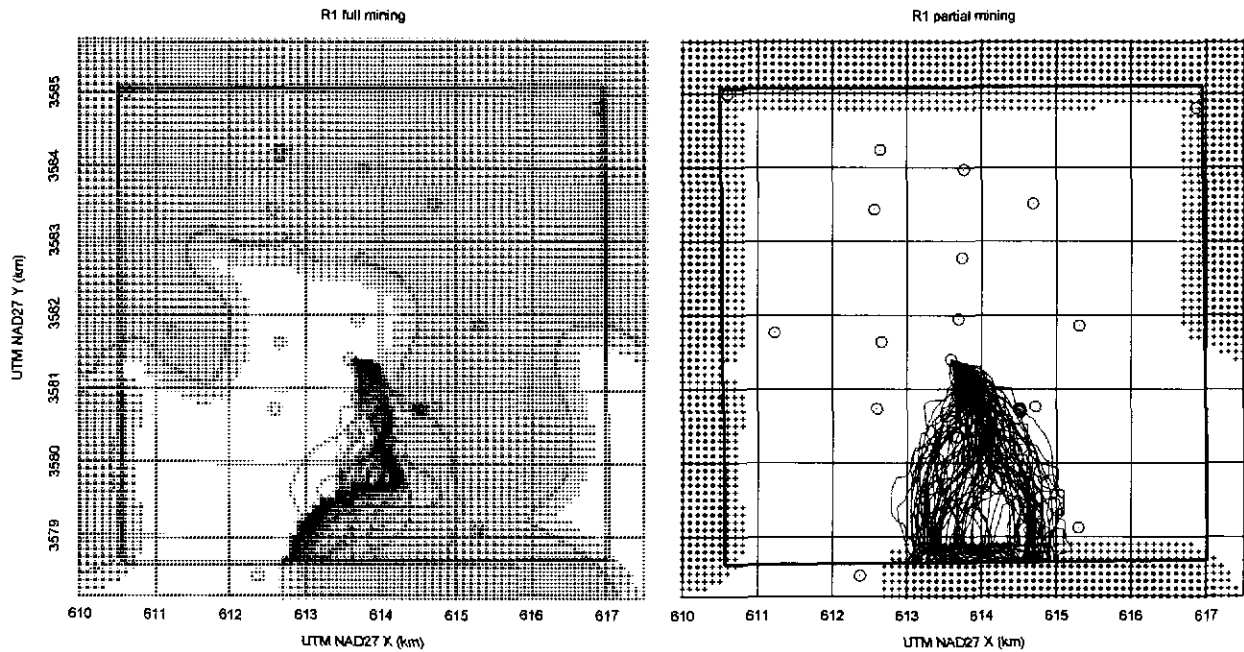


Figure 3-12. Particle tracks for R1. Small magenta squares and black crosses indicate centers of MODFLOW cells located within potash and mining-affected areas, respectively; thin black line is minable potash definition.

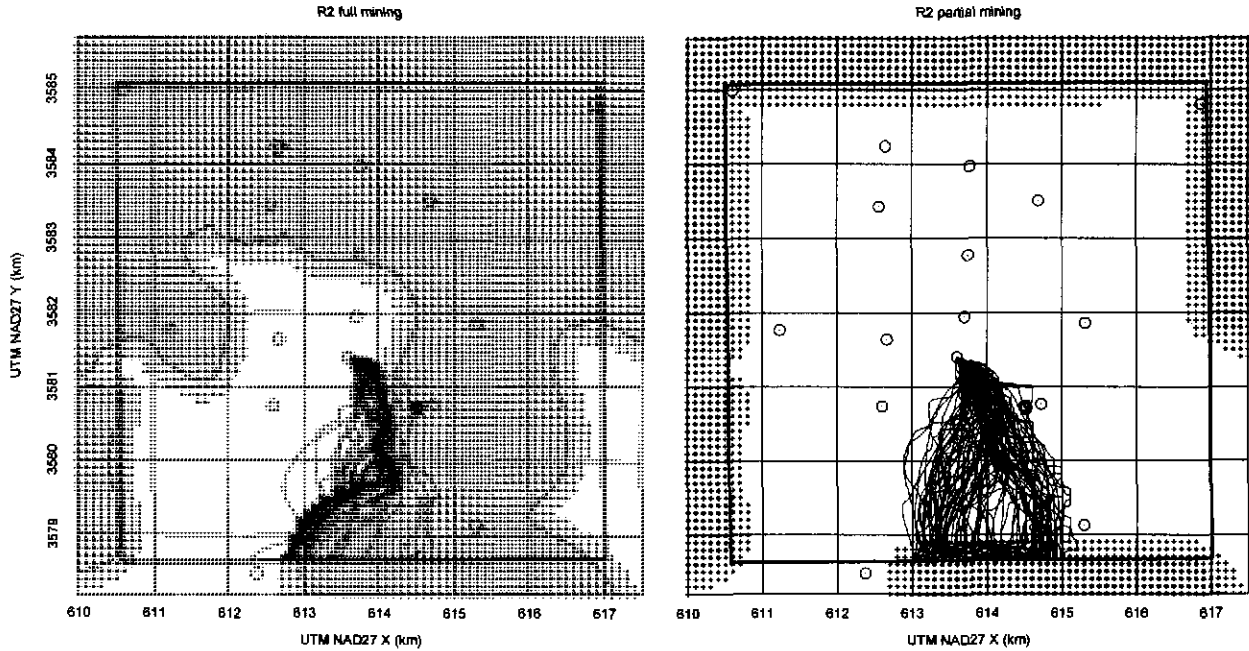


Figure 3-13. Particle tracks for R2. Small magenta squares and black crosses indicate centers of MODFLOW cells located within potash and mining-affected areas, respectively; thin black line is minable potash definition.

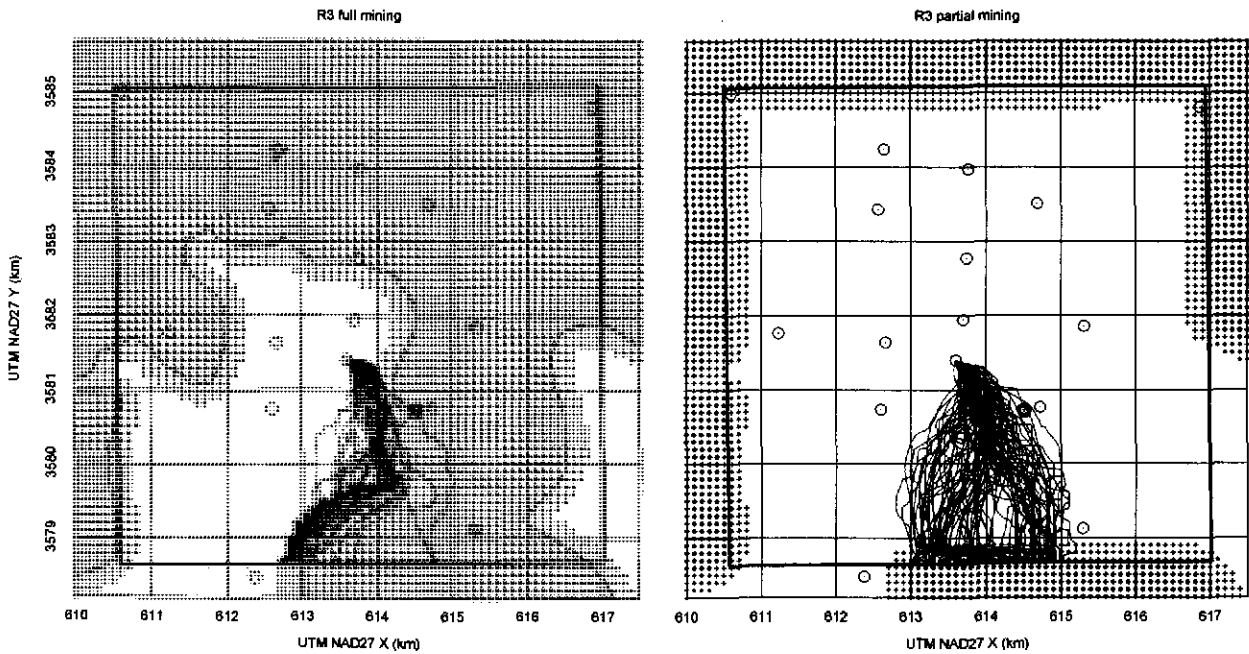


Figure 3-14. Particle tracks for R3. Small magenta squares and black crosses indicate centers of MODFLOW cells located within potash and mining-affected areas, respectively; thin black line is minable potash definition.

High-transmissivity areas corresponding to the mining-affected zones create preferential pathways through the system (see oranges and yellows in Figure 3-16). These preferential pathways result in higher velocities and flow rates through the mining zone and therefore relatively slower velocities in the non-mined areas. In the partial-mining scenario, where there is no mining inside the WIPP LWB, the preferential pathway goes “around” the LWB, rather than through it (similar to behavior seen in both

mining scenarios for PABC-2004). In the full-mining scenario, the mined regions are closer to the release point than in PABC-2004 (see Figure 3-8 for comparison), giving the particles a high-transmissivity pathway from the release point to the LWB, resulting in shorter travel times than the non-mined scenario (this behavior is different from that predicted using the PABC-2004 model). A comparison of the median, maximum, and minimum travel times for each scenario is presented in Table 3-2.

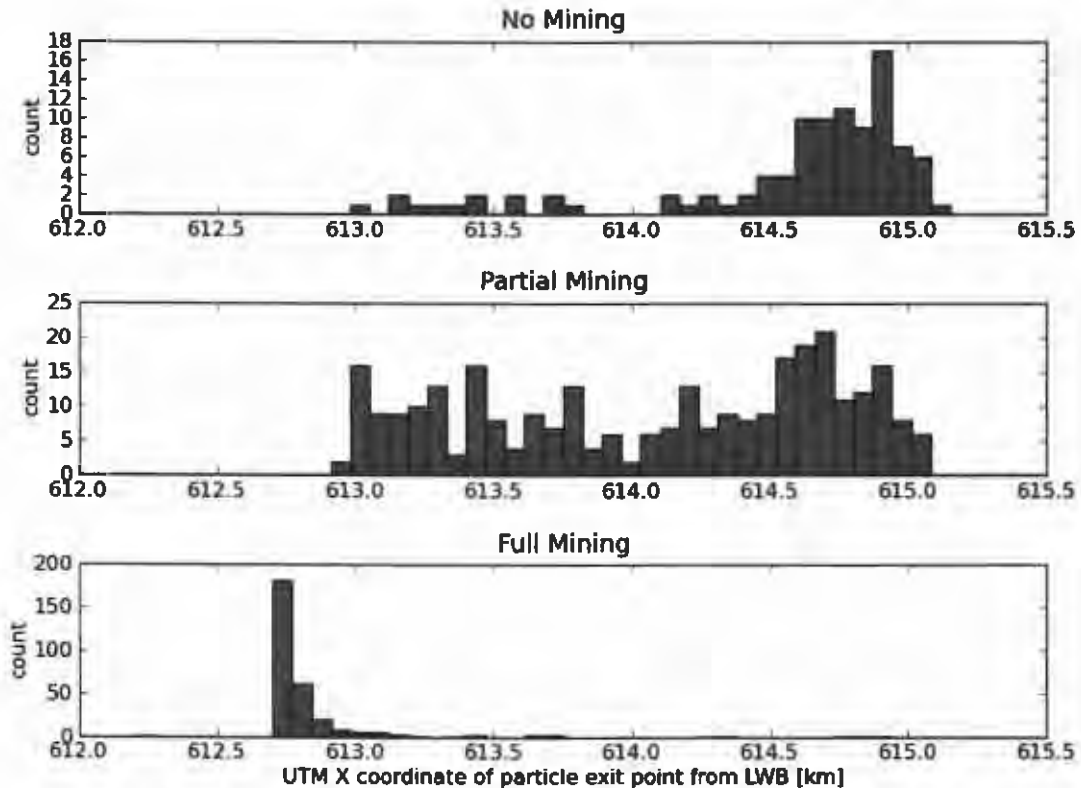


Figure 3-15. Histograms of particle x-coordinates at exit point from LWB; full and partial-mining include all three replicates (note different vertical scales between plots; no mining contains 100 particles while mining scenarios include 300 particles)

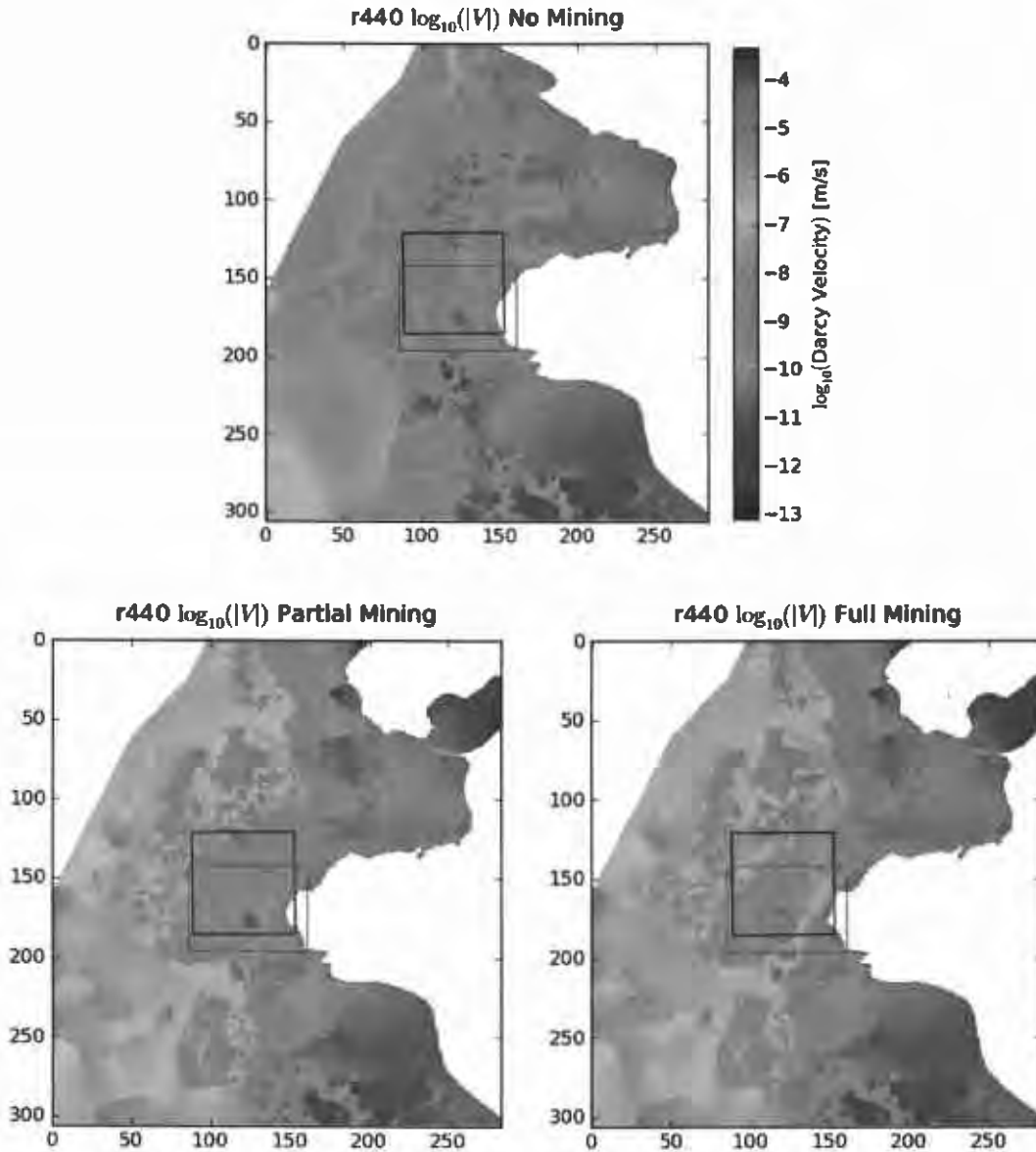


Figure 3-16. Magnitude of Darcy flux for r440 R2 for no, partial, and full-mining scenarios using cell-based coordinates; LWB (black) and SECOTP model domain (red) shown for reference.

3.4.3 Particle Speeds

Instantaneous speeds (the magnitude of particle velocities) were calculated from the DTRKMF particle locations and times using backwards finite differences,

$$v(t_{i+1}) = \frac{\sqrt{[x(t_i) - x(t_{i+1})]^2 + [y(t_i) - y(t_{i+1})]^2}}{t_{i+1} - t_i} \quad (2)$$

where a subscript i indicates the previous time step (a record or line in the DTRKMF output file) and a subscript $i+1$ is the current time step. This approach assumes a straight line connects the locations at the beginning and ends of the step, so it is potentially underestimating speeds, but step sizes are small

and error should be minimal; these values should be used for qualitative comparisons between realizations and scenarios, rather than quantitative estimates of true particle velocities.

In Figure 3-17 through Figure 3-20, the color of the diamond indicates the particle velocity; the dots are located at the midpoint of the step, e.g., $x_{\text{midpt}} = \frac{1}{2}[x(t_i) + x(t_{i+1})]$, $y_{\text{midpt}} = \frac{1}{2}[y(t_i) + y(t_{i+1})]$. In the no-mining case (Figure 3-17), the highest particle velocities are in the southeastern portion of the particle swarm, corresponding to the high-transmissivity pathway (Hart et al., 2009) exiting the LWB at approximately $X=614,750$ m (Figure 3-15). The effects of the full-mining scenario are clearly evident in the left portions of Figure 3-18 through Figure 3-20; high particle velocities (yellows and oranges) are found along the margin of the mining-affected areas, where particles enter the increased-transmissivity region. For comparison, in the partial-mining scenario the particles are relatively slowed down and more evenly distributed in the region between the release point and the southern WIPP LWB.

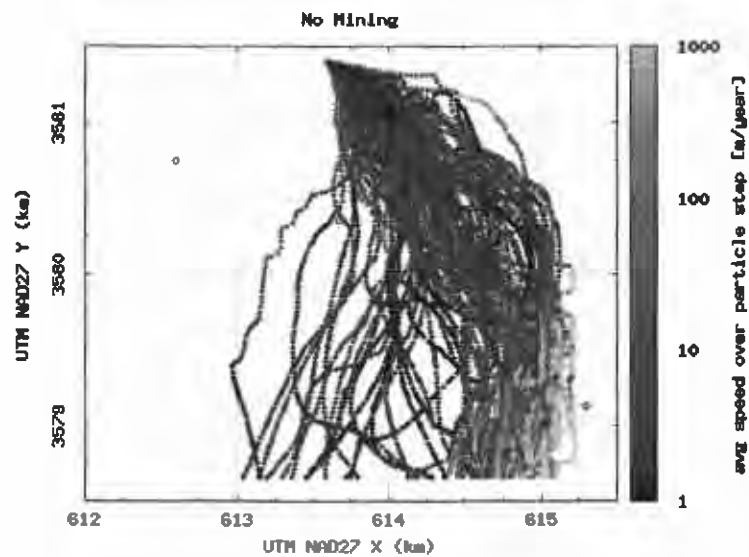


Figure 3-17. Particle speeds for non-mining scenario computed from DTRKMF results

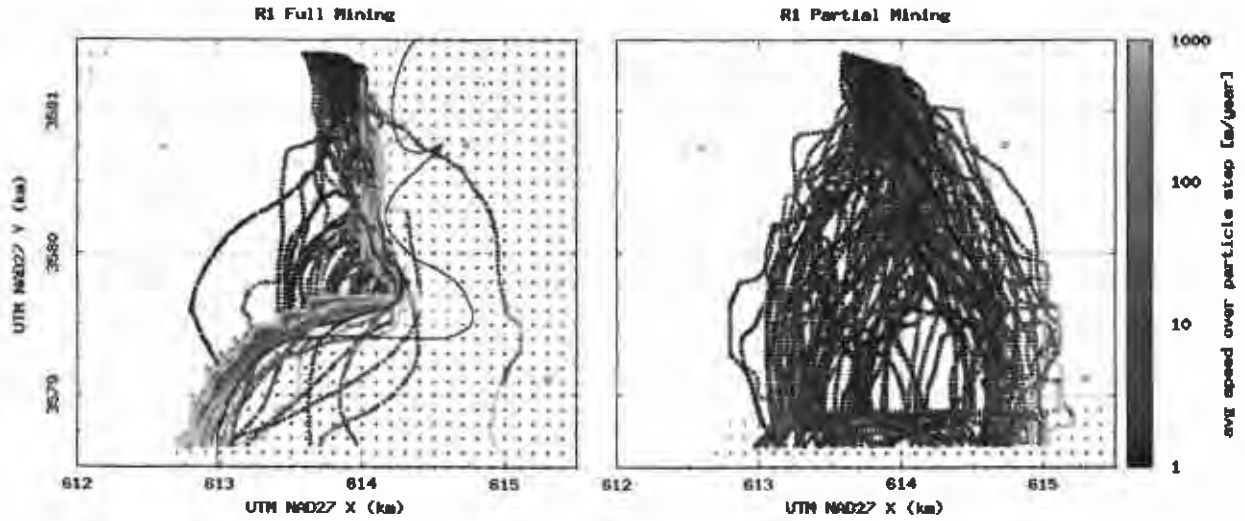


Figure 3-18. Particle speeds for R1, computed from DTRKMF results

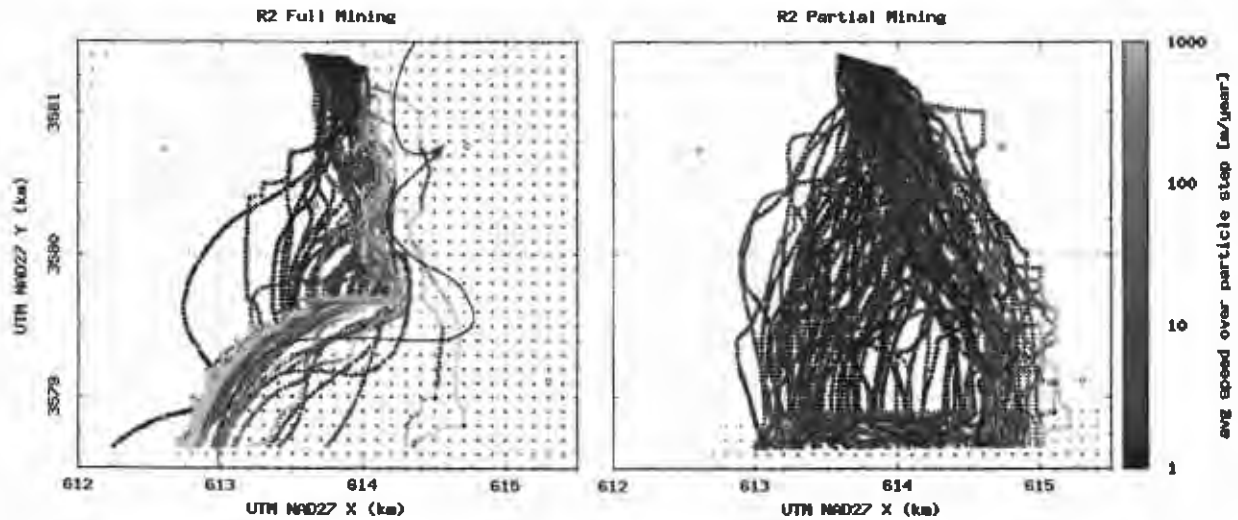


Figure 3-19. Particle speeds for R2, computed from DTRKMF results

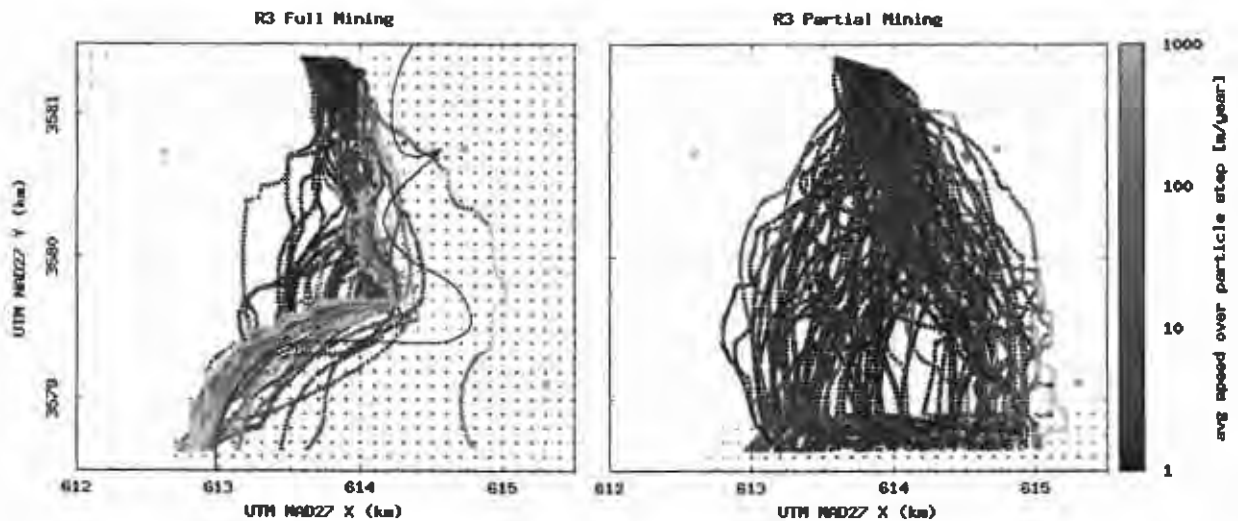


Figure 3-20. Particle speeds for R3, computed from DTRKMF results

3.5 Particle Tracking Discussion

Correlation analysis for the PABC-2009 particle tracking calculations shows that travel time and the random mining factor are weakly correlated for the partial- (Figure 3-21) or full-mining (Figure 3-22) scenarios. This is similar to what was observed for the PABC-2004 (Lowry and Kanney, 2005). The high scatter in Figure 3-21 and Figure 3-22 indicates that the transmissivity spatial distribution plays the more significant role in determining the travel time than the mining factor does. See Appendix 1 (Figure A1-9) for a cross-sectional comparison of transmissivity for each mining type, showing that the variability in the transmissivity due to calibration is on the same order as that due to mining for a single realization. The mining factor plays a weak but slightly larger role in explaining the observed variance for the partial-mining realizations (Figure 3-22) than the full-mining realizations (Figure 3-21), based on the larger (but still very small) R^2 value for the straight-line fit of \log_{10} travel times to mining factors.

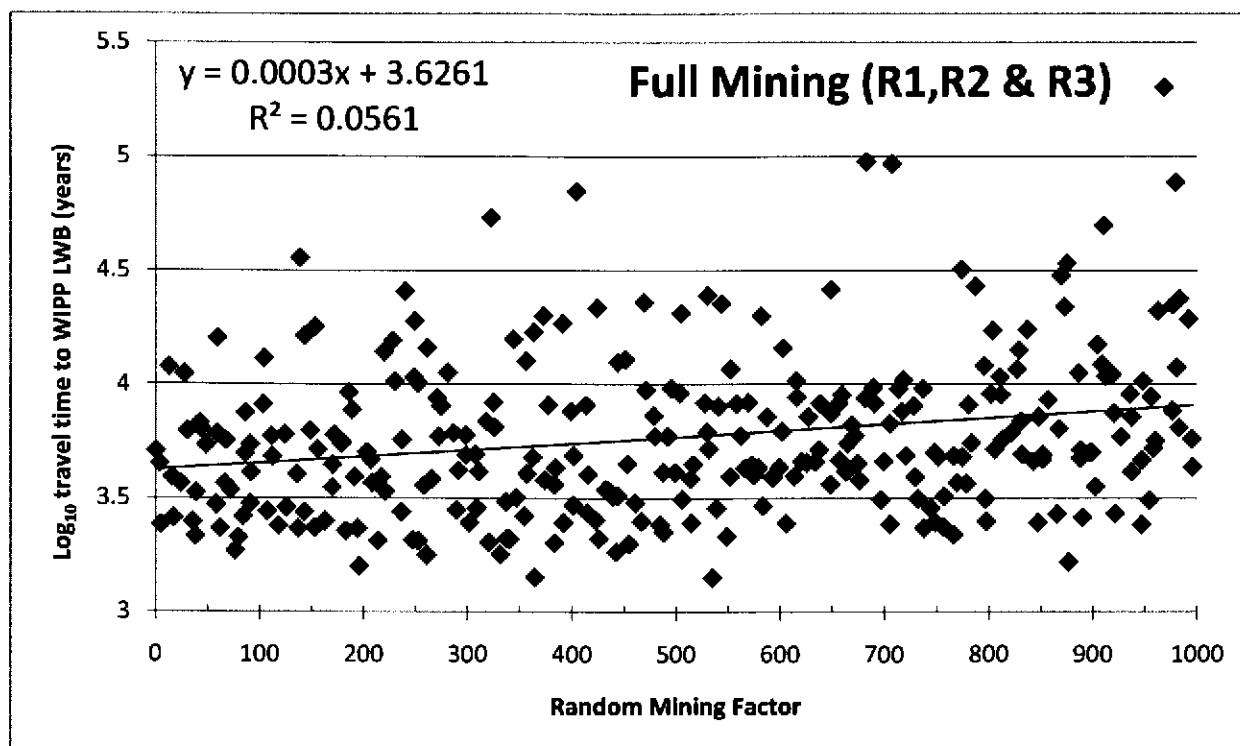


Figure 3-21. Mining factor and travel time to WIPP LWB for full-mining scenario (all replicates)

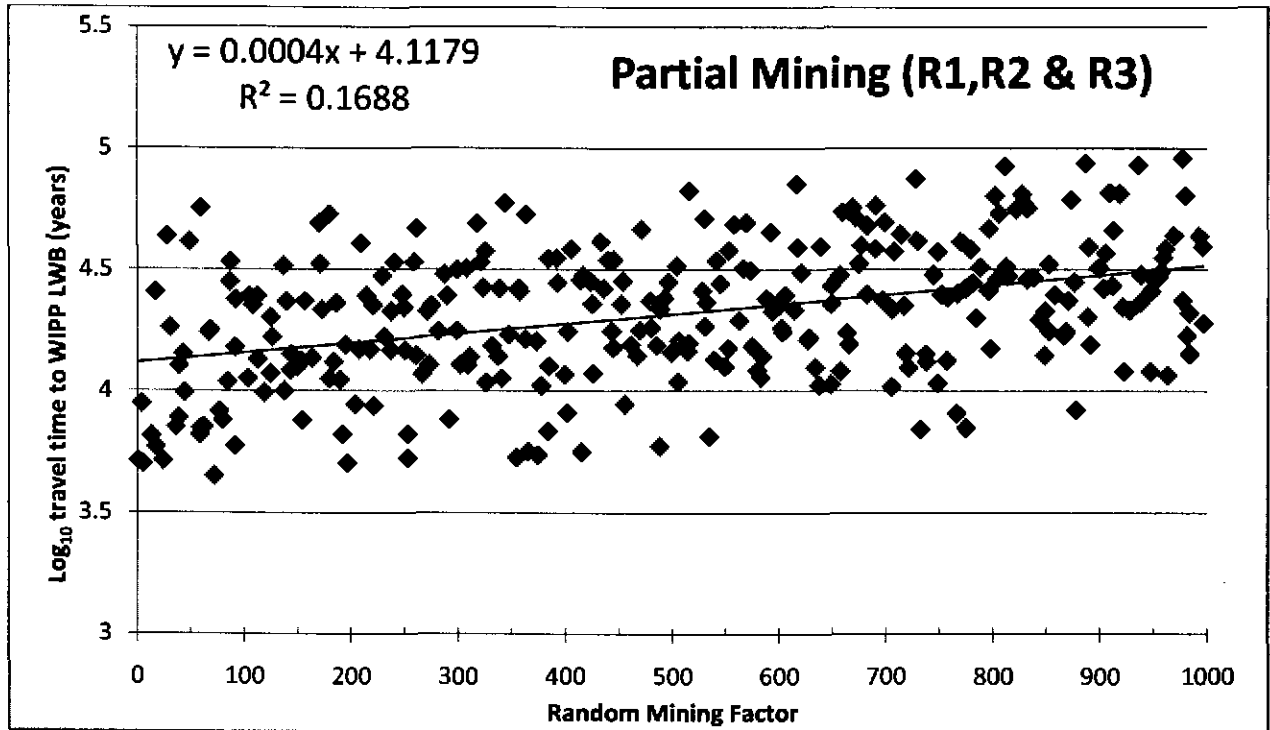


Figure 3-22. Mining factor and travel time to WIPP LWB for partial-mining scenario (all replicates)

4 Radionuclide Transport Calculations

4.1 Background and Theoretical Overview

The Culebra radionuclide transport calculations are performed using SECOTP2D, a two-dimensional, dual-porosity solute transport code developed to simulate radionuclide transport through fractured porous media (Salari and Blaine, 1996; Ramsey, 1997). This section describes the process for incorporating the MODFLOW flow-fields into the Culebra transport model, an overview of the theory underlying the SECOTP2D code, and a description of the parameters used in the Culebra transport calculations.

4.1.1 Relationship Between Flow and Transport Modeling Domains

The spatial domain used for the transport calculations, in relation to the groundwater modeling domain and the LWB, is shown in Figure 3-2. The UTM coordinates of the extent of the transport domain are given in Table 3-1.

The domain used in the transport calculations is a subregion of that used for the groundwater flow calculations. This subregion is approximately 7.5 km by 5.4 km, aligned with the principal compass directions and the principal directions of the groundwater flow domain. The transport domain extends beyond the boundaries of the WIPP LWB in the east-west direction (approximately 250 m to the west and approximately 750 m to the east). Since the undisturbed groundwater flow direction is generally north to south, the transport domain is shifted so that it extends from a point midway between the center of the waste panels and the northern LWB to approximately 1 km beyond the southern edge of the LWB. The transport calculations use a uniform computational grid composed of square 50 m cells, created by subdividing MODFLOW flow model cells into four equal-sized cells.

4.1.2 Flow-Field Extraction

Several issues need to be addressed when using the results of the MODFLOW groundwater flow calculations as input to the SECOTP2D Culebra transport calculations:

1. The MODFLOW code outputs the volumetric flux [m^3/s] across the face of each cell in the computational mesh, while the SECOTP2D code suite expects the flow-field in terms of the Darcy or specific discharge [m/s] at cell faces.
2. The computational domain used in the transport calculations is a sub-region of that used in the groundwater flow calculations.
3. The origin of the two-dimensional MODFLOW computational mesh is the northwest corner of the groundwater flow modeling domain (i.e., spreadsheet convention), while the origin of the SECOTP2D computational mesh is the southwest corner of the transport model domain (i.e., Cartesian coordinate convention).
4. The sense of coordinate system used by MODFLOW also differs from that used by SECOTP2D; the positive y -direction in MODFLOW is opposite (increasing to the south) that of SECOTP2D (increasing to the north) and thus the flux in the y -direction has different sense in the two systems.

5. MODFLOW defines the x -direction flux for a given cell index as the flux through that cell's right face while SECOTP2D defines it as the flux through the neighboring cell's left face.

The specific discharge or Darcy velocity [m/s] across the cell face is computed by dividing the volumetric flux across the cell face by its area (see Figure 4-1);

$$u = \frac{Q_x}{A_x} = \frac{Q_x}{\Delta y \cdot \Delta z} \quad (3)$$

$$v = \frac{Q_y}{A_y} = \frac{Q_y}{\Delta x \cdot \Delta z} \quad (4)$$

where u, v are the specific discharge values (Darcy velocity) across the cell face in the x - and y -directions, Q_x, Q_y are the volumetric fluxes [m^3/s] across the cell faces, and A_x, A_y are areas of the cell faces perpendicular to the x - and y -directions [m^2], respectively.

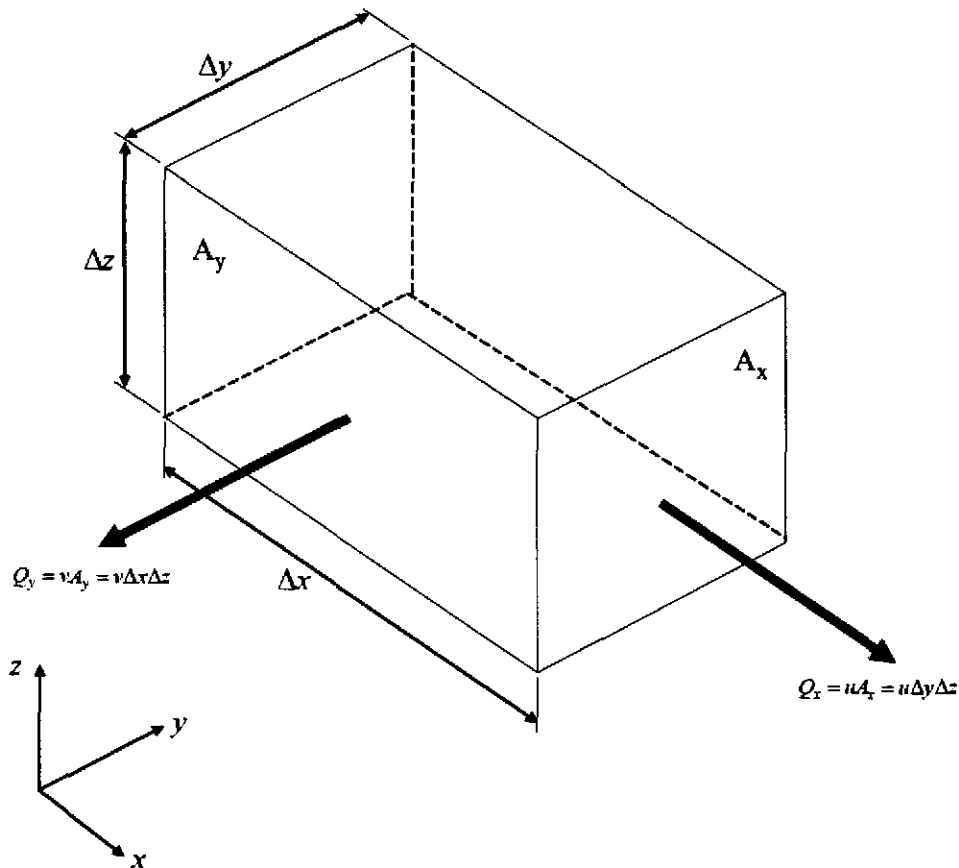


Figure 4-1. MODFLOW volumetric flux and Darcy velocity; A_y is perpendicular to the arrow pointing left; A_x is perpendicular to the arrow pointing right

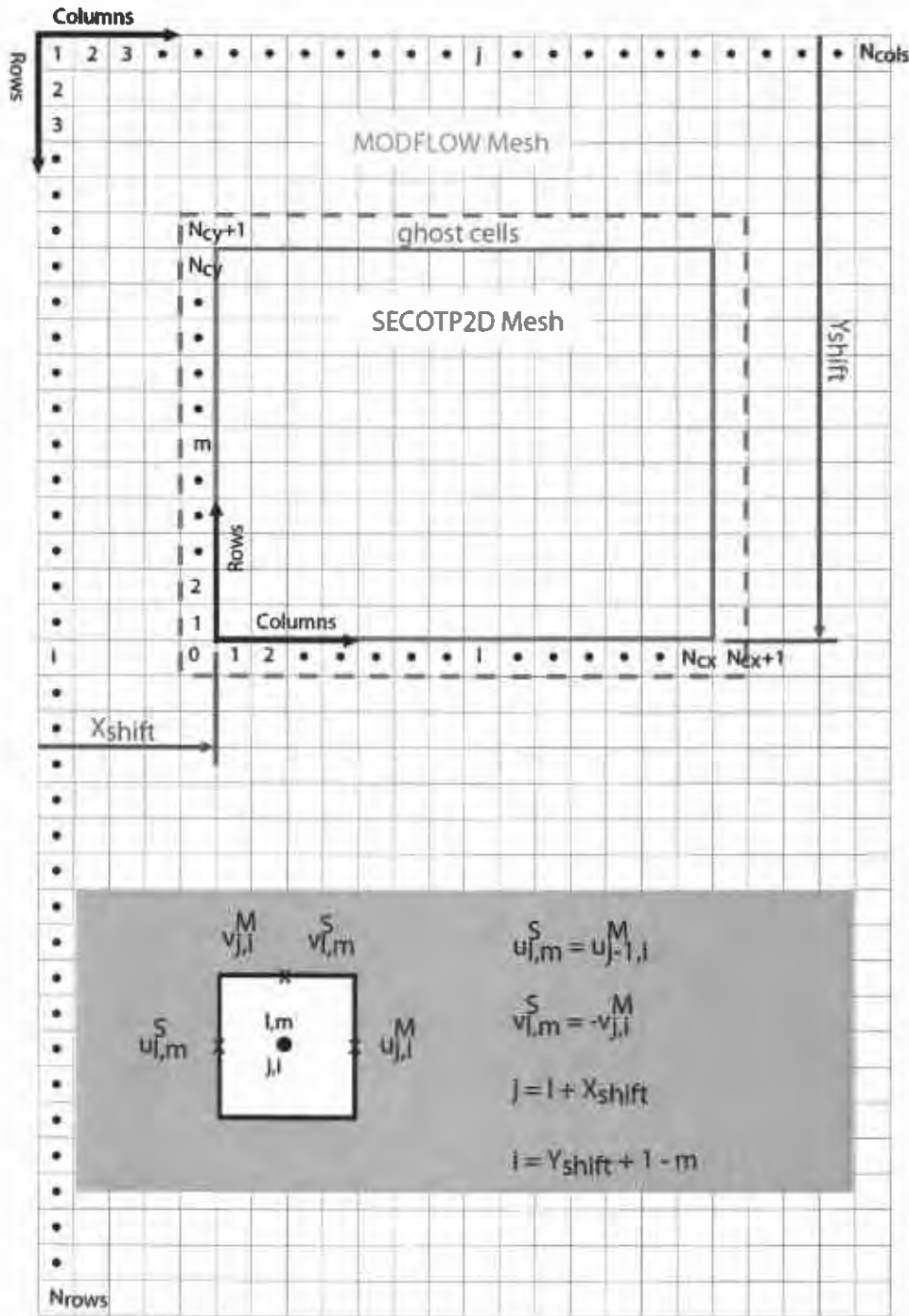


Figure 4-2. Velocity transfer between MODFLOW and SECOTP2D meshes

The remaining issues can be dealt with as outlined below and illustrated in Figure 4-2. Let (j, i) be the x -direction and y -direction indices of a cell in the MODFLOW mesh with $j = 1, \dots, N_{cols}$ and $i = 1, \dots, N_{rows}$. Let (l, m) be the indices of the same cell referenced to the SECOTP2D mesh with $l = 0, \dots, N_{cx} + 1$ and $m = 0, \dots, N_{cy} + 1$. Note that the SECOTP2D mesh uses a border of ghost cells

that extend one cell beyond the boundaries of the transport domain in order to implement boundary conditions.

Since the positive y -direction of the two meshes are opposite in sense, we must have

$$v_{l,m}^S = -v_{i,j}^M \quad (5)$$

where the superscript S denotes the SECOTP2D velocity and M denotes the MODFLOW velocity. The difference in conventions regarding which face to associate with a given cell can be written as

$$u_{l,m}^S = u_{j-1,i}^M \quad (6)$$

Let X_{shift} be the x -direction distance (in number of cells) between the origin of the MODFLOW mesh and the SECOTP2D. Let Y_{shift} represent the corresponding distance in the y -direction. The MODFLOW cell indices corresponding to the cell (l, m) in the SECOTP2D mesh are then given by

$$j = l + X_{\text{shift}} \quad (7)$$

$$i = Y_{\text{shift}} + 1 - m \quad (8)$$

The rules specified by equations 5 and 8 may be summarized by the following pseudo-code algorithm:

```

for  $l = 1$  to  $N_{cx}$  do
  for  $m = 1$  to  $N_{cy}$  do
     $j \leftarrow l + X_{\text{shift}}$ 
     $i \leftarrow Y_{\text{shift}} + 1 - m$ 
     $u_{l,m}^S \leftarrow u_{j-1,i}^M$ 
     $v_{l,m}^S \leftarrow -v_{i,j}^M$ 
  end for
end for

```

4.1.3 VTRAN2 Output Summary

Figure 4-3 shows ensemble average and standard deviation distributions for Darcy velocity (from VTRAN output). Each figure is either an average or standard deviation across 300 realizations (3 replicates \times 100 realizations). The WIPP LWB, panel outline, and release locations are indicated on each of the six plots.

The top two plots in Figure 4-3 show the average magnitude of \log_{10} Darcy velocity (color plot) and scaled vectors (only one of every five is plotted for clarity) indicating direction. Although color scales are consistent in each row, arrow sizes are not. The high-transmissivity pathway observed during the SNL-14 pumping test (the north-south feature located in southern half of the domain approximately following column 95) is also apparent in both the full- and partial-mining scenarios, as it was in the unmined MODFLOW scenarios (Hart et al., 2009). The edges of the areas affected by potash mining can be seen in the color plots. In the full-mining case this appears as the blue irregular shape occupying the west half of the transport model domain and in the partial-mining case the the edge of the affects of

mining can be seen just inside the WIPP LWB (see Section 3.3.2 for discussion and illustration of mining-affected areas).

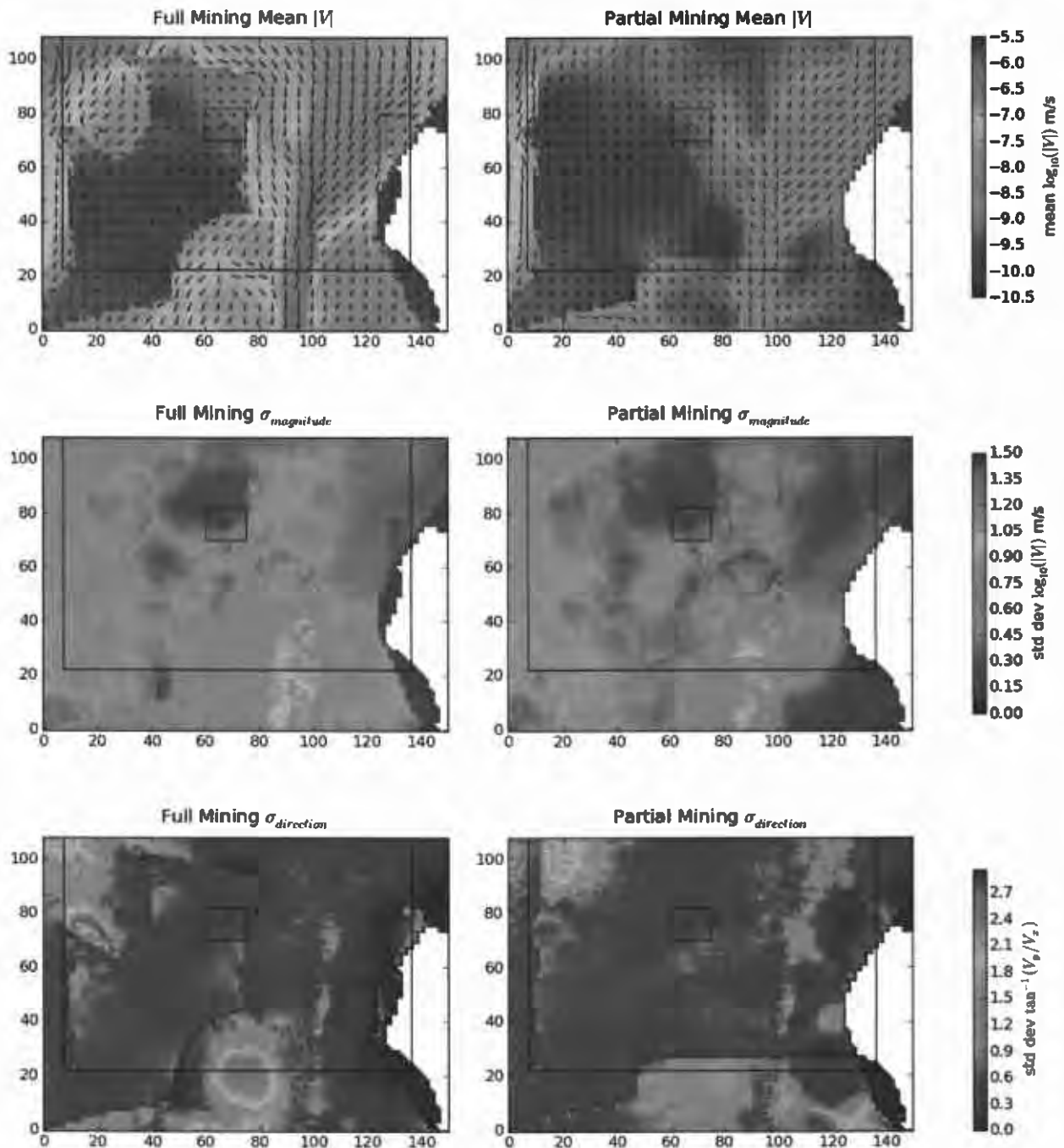


Figure 4-3. Ensemble average and standard deviation of Darcy flow velocity magnitude $|V|$ and direction V_{dir} from VTRAN2 (each figure is constructed from 300 realizations); SECOTP2D cell-based coordinates. Outer black line is the LWB, small black box outlines the WIPP panels, with the yellow star at the release point (C-2737).

The middle two plots in Figure 4-3 show the variability in the magnitude of the \log_{10} Darcy velocity. The highest variability (red to yellow) is associated with the southern part of the high-T pathway, at or south

of where the pathway crosses the WIPP LWB. The high-T pathway is the estimated distribution of T associated with the response observed due to pumping SNL-14 (Hart et al., 2009). The bottom two plots show variability in the direction of the Darcy velocity. In the full-mining case, there is high variability (red) in the direction at the WIPP LWB west of the high-T pathway and along the northwest boundary of the non-mining area (west and north of the release point). Both of these regions of high direction variance are in areas that are affected by potash mining located immediately next to non-mining areas. Low variability (blues) in both the magnitude and direction of the Darcy velocity across all the realizations is found inside the non-mining areas.

Example VTRAN2 output for a single MODFLOW realization (r440) is shown in the six plots in Appendix 1 (Figure A1-11). R1 and R3 have relatively high mining factors (663.4 and 902.6, respectively), while R2 has a moderate mining factor (77.07).

4.1.4 Solute Transport Modeling

The SECOTP2D code assumes parallel-plate fracturing where fluid flow is restricted to the advective continuum (fractures) and mass is transferred between the advective and diffusive (matrix) continua via molecular diffusion. The dual-porosity conceptualization is illustrated in Figure 4-4. Retardation is permitted in both the advective and diffusive domains assuming linear equilibrium isotherms. Radioactive decay is accounted for through the use of multiple straight decay chains.

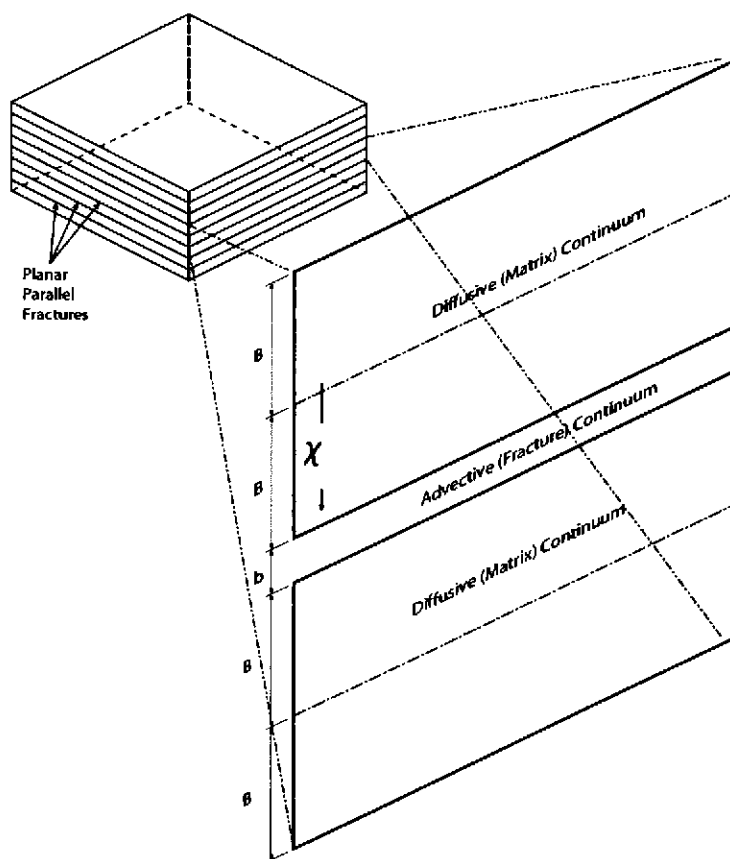


Figure 4-4. Dual-Porosity Conceptual Model

SECOTP2D solves the following partial differential equation (PDE) for radionuclide transport in the advective (i.e., fracture) continuum:

$$\phi R_\ell \frac{\partial C_\ell}{\partial t} = \nabla \cdot (\phi \mathbf{D}_\ell \nabla C_\ell - \mathbf{V} C_\ell) - \phi R_\ell \lambda_\ell C_\ell + \phi R_{\ell-1} \lambda_{\ell-1} C_{\ell-1} + Q_\ell + \Gamma_\ell \quad (9)$$

where $\ell = 1, \dots, N_s$ is a species index (N_s is the number of species), C_ℓ is the unknown concentration of the ℓ^{th} radionuclide in the advective continuum [kg/m^3], \mathbf{V} is the specific discharge vector [m/s], \mathbf{D}_ℓ is the second-rank hydrodynamic dispersion tensor [m^2/s], ϕ is the dimensionless advective porosity, R_ℓ is the dimensionless retardation coefficient, λ_ℓ is the radioactive species decay constant [$1/\text{s}$], Q_ℓ is the specific injection rate [$\text{kg}/(\text{m}^3 \cdot \text{s})$], and Γ_ℓ is the mass transfer coefficient between the advective and diffusive continua per unit volume [$\text{kg}/(\text{m}^3 \cdot \text{s})$]. Only the dependent concentration variables (C_ℓ) and the mass transfer coefficient (Γ_ℓ) are both spatially and temporally variable. Flow velocities (the vector \mathbf{V} and its components u and v) are spatially variable, but steady state. All other parameters in the equations are constant in space and time for each realization.

The concentration C_ℓ is defined as the mass of the ℓ^{th} radionuclide per unit volume of pore fluid. The advective porosity ϕ is defined as the ratio of the advective pore volume to the total volume. Similarly, the specific injection rate Q_ℓ is defined as the rate of mass injected per unit bulk volume per time. Terms in equation 9 involving $\ell - 1$ are omitted for species that are parents of their respective decay chains ($\ell = 1$). The advective transport equation is linear in C_ℓ and is solved simultaneously for all N_s species in a given decay chain.

The vector flow-field \mathbf{V} is assumed to be independent of the solute concentration. The flow-field is obtained from MODFLOW and is scaled by a random factor between 1 and 2.25 to account for the potential impact of climate change (Corbet and Swift, 1996).

The product $\phi \mathbf{D}_\ell$ is defined as

$$\phi \mathbf{D}_\ell = \frac{1}{|\mathbf{V}|} \begin{bmatrix} u & -v \\ v & u \end{bmatrix} \begin{bmatrix} \alpha_L & 0 \\ 0 & \alpha_T \end{bmatrix} \begin{bmatrix} u & v \\ v & u \end{bmatrix} + \phi \tau D_\ell^* \quad (10)$$

where α_L and α_T are the longitudinal and transverse dispersivities of the advective continuum [m], u and v are the scalar x - and y -components of the specific discharge vector [m/s], D_ℓ^* is the free water molecular diffusion coefficient [m^2/s], and τ is the dimensionless advective tortuosity defined as the ratio of the fluid particle flow path length to the length of the porous medium. Note that WIPP PA takes no direct credit for hydrodynamic dispersion in Culebra transport calculations; only mixing due to molecular diffusion is allowed (i.e., $\alpha_L = \alpha_T = 0$ in equation 10 and therefore $\phi \mathbf{D}_\ell = \phi \tau D_\ell^*$). WIPP PA treats the free liquid molecular diffusion coefficient as a function of both radionuclide species and oxidation state.

The retardation coefficient R_ℓ is defined by:

$$R_\ell = 1 + \rho_s K_d^\ell \frac{1 - \phi}{\phi} \quad (11)$$

where ρ_s is the Culebra dolomite grain density [kg/m^3] and K_d^ℓ is the distribution coefficient of the ℓ^{th} radionuclide [m^3/kg], equation 11 completely describes the linear isotherm for radionuclide sorption to the Culebra.

SECOTP2D solves the following one-dimensional PDE for diffusional radionuclide transport in the matrix continuum:

$$\phi' R_\ell' \frac{\partial C_\ell'}{\partial t} - \frac{\partial}{\partial \chi} \left(\phi' D_\ell' \frac{\partial C_\ell'}{\partial \chi} \right) = -\phi' R_\ell' \lambda_\ell C_\ell' + \phi' R_{\ell-1}' \lambda_{\ell-1} C_{\ell-1}' \quad (12)$$

where C_ℓ' is the unknown concentration of the ℓ^{th} radionuclide in the diffusive continuum [kg/m^3], χ is the spatial coordinate originating from the symmetry line of a matrix block (as shown in Figure 4-4), and D_ℓ' is the matrix diffusion coefficient. The matrix diffusion coefficient is defined as:

$$D_\ell' = \tau' D_\ell^* \quad (13)$$

where τ' is the matrix tortuosity. All other symbols in equation 12 for diffusive transport have analogous meaning as those in equation 9 for advective transport except that the prime denotes diffusive-continuum properties.

The governing equations for the advective (9) and diffusive (12) continua are coupled through the mass transfer term, Γ_ℓ . Applying Fick's law at the interface between the two continua results in the mass-transfer equation:

$$\Gamma_\ell = -\frac{2\phi}{b} \left(\phi' D_\ell' \frac{\partial C_\ell'}{\partial \chi} \Big|_{\chi=B} \right) \quad (14)$$

where B is the matrix half-block length [m], b is the fracture aperture [m] (see Figure 4-4), the term in parentheses represents the mass flux per unit area of contact between the advective and diffusive continua. The term $2\phi/b$ represents the specific surface area (ratio of surface area to bulk volume) of the coupled system. In the parallel-plate formulation used in SECOTP2D, the fracture aperture is defined in terms of advective porosity and matrix half-block length by:

$$b = \frac{\phi B}{1 - \phi} \quad (15)$$

4.1.5 Model Parameters

The transport equations for the advective and diffusive continua described in Section 4.1.4 contain many parameters, which must be obtained from measurements, theoretical considerations, or expert judgment. In general, WIPP PA treats some process model parameters as deterministic and others as uncertain. Uncertain parameters are represented by probability distributions, and sampled values of uncertain parameters are used in a probabilistic (Monte Carlo) modeling approach to predict repository performance.

The parameters can be divided into physical parameters and chemical parameters. The physical parameters are generally properties of the porous Culebra material. The chemical parameters are

generally properties of the radionuclide species transported. Some chemical parameters are functions of the oxidation state as well as species.

WIPP PA treats the following Culebra physical transport parameters as deterministic:

1. advective continuum longitudinal and transverse dispersivity (α_L, α_T);
2. fracture tortuosity (τ);
3. matrix tortuosity (τ');
4. skin resistance factor; and
5. Culebra material grain density (ρ_s).

The WIPP PA Parameter Database (PAPDB) designations for these parameters, along with the values used in this analysis, are shown in Table 4-1; these parameter values have not changed since the CCA. Table 4-1 shows that no credit is taken for dispersion in the transport of radionuclides in the Culebra ($\alpha_L = \alpha_T = 0$).

Table 4-1. Deterministic physical transport parameters

| Model Parameter | MATERIAL:PROPERTY | Value | Units |
|---------------------------|-------------------|-------|-------------------|
| Longitudinal dispersivity | CULEBRA:DISP_L | 0 | m |
| Transverse dispersivity | CULEBRA:DISPT_L | 0 | m |
| Fracture tortuosity | CULEBRA:FTORT | 1.0 | - |
| Matrix tortuosity | CULEBRA:DTORT | 0.11 | - |
| Skin resistance | CULEBRA:SKIN_RES | 0 | - |
| Material grain density | CULEBRA:DNSGRAIN | 2820 | kg/m ³ |

WIPP PA treats the following Culebra physical transport parameters as subjectively uncertain:

1. advective (fracture) porosity (ϕ);
2. diffusive (matrix) porosity (ϕ');
3. matrix half-block length (B);
4. climate index;
5. flow-field index; and
6. mining factor.

The PAPDB designations for these parameters, along with the probability distributions used in this analysis are shown in Table 4-2. Note that these parameter distributions have not changed since the CCA. The sampled values for any given analysis will depend upon the random seed used in the sampling algorithm.

Potential climatic changes are simulated as leading to an increase in hydraulic gradients in the Culebra. This is implemented as a uniform increase in the steady-state velocity fields by the multiplier CLIMTIDX at the beginning of the simulation (Corbet and Swift, 1996). Calibrated MODFLOW realizations and mining factors are mapped from one to the other (a different mapping for each replicate) using the integer index TRANS_IDX.

See Appendix 2 (Table A2-3) for sampled values of APOROS, DPOROS, HMBLKLT, and CLIMTIDX used in analysis, and see Appendix 1 (Table A1-12) for values of MINP_FAC and TRANS_IDX used in the analysis.

Table 4-2. Uncertain physical parameters

| Model Parameter | MATERIAL:PROPERTY | Units | Distribution | Range | Median |
|--------------------------|-------------------|-------|--------------|--------------------|----------|
| Advective porosity | CULEBRA:APOROS | - | log-uniform | [1.00E-4, 1.00E-2] | 1.000E-3 |
| Diffusive porosity | CULEBRA:DPOROS | - | cumulative | [1.00E-1, 2.50E-1] | 1.600E-1 |
| Matrix half-block length | CULEBRA:HMBLKLT | m | uniform | [5.00E-2, 5.00E-1] | 2.750E-1 |
| Climate index | GLOBAL:CLIMTIDX | - | cumulative | [1.00E+0, 2.25E+0] | 1.170E+0 |
| Flow-field index | GLOBAL:TRANSIDX | - | uniform | [0.00E+0, 1.00E+0] | 5.000E-1 |
| Mining factor | CULEBRA:MINP_FAC | - | uniform | [1.00E+0, 1.00E+3] | 5.005E+2 |

WIPP PA treats the following radionuclide chemical parameters as deterministic:

1. radionuclide atomic weight;
2. radionuclide half-life; and
3. radionuclide free liquid molecular diffusion coefficient (D_f^*).

The PAPDB designations for these parameters, along with the values used in this analysis, are shown in Table 4-3. Note that these parameter values have not changed since the CCA.

Table 4-3. Deterministic chemical parameters

| Model Parameter | MATERIAL:PROPERTY | Value | Units |
|--|-------------------|-------------|-------------------|
| Atomic Weight¹ | | | |
| ²⁴¹ Am | AM241:ATWEIGHT | 2.41057E-01 | kg/mole |
| ²³⁹ Pu | PU239:ATWEIGHT | 2.39052E-01 | kg/mole |
| ²³⁰ Th | TH230:ATWEIGHT | 2.30033E-01 | kg/mole |
| ²³⁴ U | U234:ATWEIGHT | 2.34041E-01 | kg/mole |
| Half-life (t_{1/2})^{1,2} | | | |
| ²⁴¹ Am | AM241:HALFLIFE | 1.36400E+10 | s |
| ²³⁹ Pu | PU239:HALFLIFE | 7.59400E+11 | s |
| ²³⁰ Th | TH230:HALFLIFE | 2.43000E+12 | s |
| ²³⁴ U | U234:HALFLIFE | 7.71600E+12 | s |
| Molecular diffusion coefficient³ | | | |
| Am(III) | AM+3:MD0 | 3.00000E-10 | m ² /s |
| Pu(III) | PU+3:MD0 | 3.00000E-10 | m ² /s |
| Pu(IV) | PU+4:MD0 | 1.53000E-10 | m ² /s |
| Th(IV) | TH+4:MD0 | 1.53000E-10 | m ² /s |
| U(IV) | U+4:MD0 | 1.53000E-10 | m ² /s |
| U(VI) | U+6:MD0 | 4.26000E-10 | m ² /s |

1. Atomic weight and half-life used to calculate specific activity
2. $\lambda_d = (\ln 2)/t_{1/2}$
3. Free-liquid diffusion coefficient is a function of species and oxidation state

WIPP PA treats the following radionuclide chemical parameters as subjectively uncertain:

1. oxidation state; and
2. matrix distribution coefficient (K_d^e).

The PAPDB designations for these parameters, along with the probability distributions used in this analysis, are shown in Table 4-4. Note that the lower bounds of the K_d ranges for constituents modeled (except U(VI)) were lowered (Clayton, 2009b), as requested by the EPA in the third 2009 completeness letter (Kelly, 2009). The oxidation states and upper limits on the K_d parameter distributions have not changed since the CCA. The sampled values for any given analysis will depend upon the random seed used in the sampling algorithm.

Table 4-4. Uncertain chemical parameters

| Model Parameter | MATERIAL:PROPERTY | Units | Distribution | Range | Median |
|--|-------------------|--------------------|--------------|--------------------|---------|
| Oxidation state index | GLOBAL:OXSTAT | - | uniform | [0.00E+0, 1.00E+0] | 5.00E-1 |
| Matrix distribution coefficient | | | | | |
| Am(III) | AM+3:MKD_AM | m ³ /kg | log-uniform | [5.00E-3, 4.00E-1] | 4.50E-2 |
| Pu(III) | PU+3:MKD_PU | m ³ /kg | log-uniform | [5.00E-3, 4.00E-1] | 4.50E-2 |
| Pu(IV) | PU+4:MKD_PU | m ³ /kg | log-uniform | [5.00E-4, 1.00E+1] | 7.10E-2 |
| Th(IV) | TH+4:MKD_TH | m ³ /kg | log-uniform | [5.00E-4, 1.00E+1] | 7.10E-2 |
| U(IV) | U+4:MKD_U | m ³ /kg | log-uniform | [5.00E-4, 1.00E+1] | 7.10E-2 |
| U(VI) | U+6:MKD_U | m ³ /kg | log-uniform | [3.00E-5, 2.00E-2] | 7.70E-4 |

4.2 Transport Results

Radionuclide transport calculations for the Culebra were performed with the SECOTP2D code suite as described in Section 4.1. All calculations were performed on the WIPP Alpha Cluster under formal run control procedures (Long, 2010).

The prediction of interest in the Culebra transport calculations is the cumulative release of radionuclides at the LWB during the 10,000-year regulatory period. In each transport simulation, 1 kg of each of four radionuclides is released at the center of the waste panel area during the first 50 years after repository closure. This unit mass release at the initial simulation time is subsequently scaled and time shifted in the CCDFGF analysis (Camphouse, 2010). The radionuclides transported in the Culebra are ²⁴¹Am, ²³⁴U, ²³⁰Th and ²³⁹Pu. Pu may be present in either the Pu(III) or Pu(IV) oxidation state; U may be present as U(IV) or U(VI), depending on the values of OXSTAT; mixed high- and low-oxidation states do not occur.

Transport calculations were performed for both full-mining and partial-mining scenarios. The partial-mining scenario assumes potash is only mined outside the LWB, while full mining assumes that all reserves are exploited both inside and outside the LWB. The effect of mining enters the transport calculations through the Culebra flow-field computed using MODFLOW (see Section 3).

A total of 300 realizations (three replicates of 100 vectors each) were required for both full- and partial-mining conditions; 600 Culebra transport simulations were performed. Along with the input files referenced above, the output (CAMDAT database) files from these simulations are stored in CMS library LIBPABC09_ST2D, class PABC09-0 on the WIPP PA Alpha Cluster. The naming convention for the CAMDAT database files is: ST2D3_PABC09_Rr_Mm_Vvvv.CDB, where $r \in \{1, 2, 3\}$, $m \in \{F, P\}$, and $vvv \in \{001, 002, \dots, 100\}$. See (Long, 2010) for a complete description of the run control procedures used to perform the Culebra transport calculations. In each transport simulation, 1 kg of each of four radionuclides (²⁴¹Am, ²³⁴U, ²³⁰Th, and ²³⁹Pu) is released in the Culebra at the location of observation well C-2737 (the center of the WIPP waste panels). Transport of the ²³⁰Th daughter product of ²³⁴U decay is calculated and tracked as a separate species. In the following discussion, ²³⁰Th will refer to the ²³⁴U daughter product and ²³⁰ThA will refer to that released at the center of the waste panel area.

4.2.1 Radionuclide Transport Under Partial-Mining Conditions

Under partial-mining conditions, only ^{234}U and its daughter product ^{230}Th had a significant number of vectors (37 and 21, respectively) where radionuclides were transported to the LWB during the course of the 10,000-year simulation. Only five vectors of ^{239}Pu and ^{230}ThA reached a 10^{-9} kg cumulative release threshold (Table 4-5), and ^{241}Am had no vectors surpassing a 10^{-9} kg criterion. The 10^{-9} kg cumulative release reporting or significance threshold is somewhat arbitrary, but is chosen to compare to results reported previously in PABC-2004 (Lowry and Kanney, 2005).

Table 4-5. Number of vectors exceeding the 10^{-9} kg cumulative release threshold during the 10,000 year simulation

| | partial | | | full | | |
|--------------------|---------|----|----|------|----|----|
| | R1 | R2 | R3 | R1 | R2 | R3 |
| ^{241}Am | 0 | 0 | 0 | 8 | 10 | 3 |
| ^{239}Pu | 3 | 1 | 1 | 20 | 27 | 22 |
| ^{234}U | 11 | 14 | 12 | 48 | 50 | 47 |
| ^{230}Th | 5 | 10 | 6 | 36 | 38 | 42 |
| ^{230}ThA | 2 | 3 | 0 | 21 | 31 | 29 |

The total numbers of vectors with releases greater than the 10^{-9} kg criterion, tabulated in Table 4-5, are shown in Figure 4-5 as histograms of log cumulative releases. The individual histograms in Figure 4-5 are arranged similar to the rows and columns in Table 4-5 through Table 4-8. The histograms give more information about the distribution of vectors with cumulative releases greater than the 10^{-9} kg criterion, rather than just the total number of them.

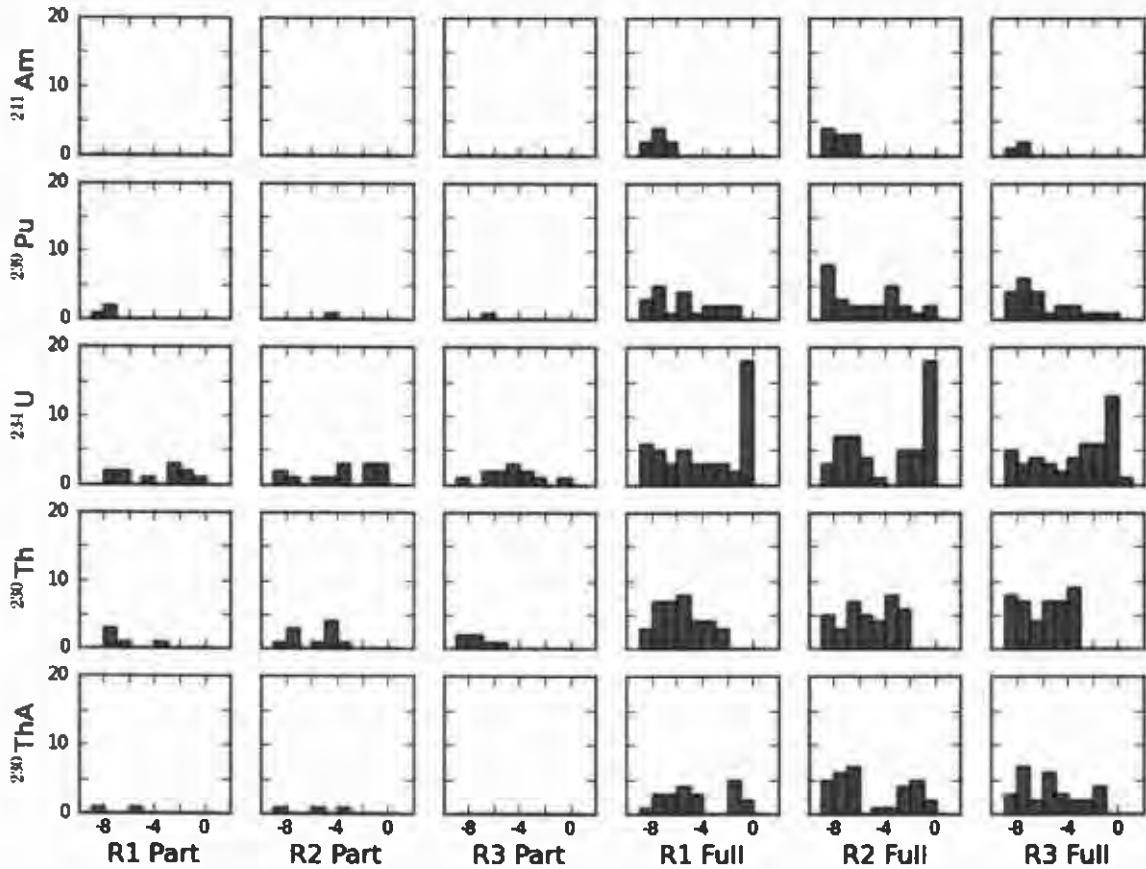


Figure 4-5. Histograms showing distribution of cumulative releases greater than 10^{-9} kg, each x-axis is \log_{10} cumulative release (kg), each y-axis is number of vectors per bin (bins are one log cycle wide).

Table 4-6 shows the maximum cumulative release predicted for each group of 100 vectors, including all vectors (not just those greater than the 10^{-9} kg criterion). These values correspond to the rightmost bar in the histograms shown in Figure 4-5. The results for ^{241}Am are very low and are likely caused by numerical dispersion, as the concentrations reported are so low as to be physically unrealistic. The maximum masses reported reaching the WIPP LWB are smaller than can validly be simulated using SECOTP2D and can therefore be considered to be effectively zero.

Table 4-6. Maximum cumulative release (kg) within each group of 100 vectors at 10,000 years

| | partial | | | full | | |
|--------------------|---------|---------|---------|----------------------|----------------------|----------------------|
| | R1 | R2 | R3 | R1 | R2 | R3 |
| ^{241}Am | 4.6e-27 | 7.4e-20 | 4.3e-23 | 1.1e-07 | 7.5e-07 [#] | 8.6e-08 |
| ^{239}Pu | 3.9e-08 | 3.3e-05 | 5.7e-07 | 3.3e-02 | 2.5e-01 [#] | 1.1e-01 |
| ^{234}U | 7.1e-01 | 7.5e-01 | 2.5e-01 | 9.9e-01 | 9.9e-01 | 1.0e+00 [#] |
| ^{230}Th | 9.9e-04 | 1.0e-04 | 1.9e-06 | 4.2e-03 [#] | 2.6e-03 | 9.2e-04 |
| ^{230}ThA | 8.5e-06 | 9.5e-04 | 2.6e-12 | 2.1e-01 [#] | 1.8e-01 | 6.6e-02 |

[#] Indicates the largest maximum release for each radionuclide

The identities of the vectors that produced the largest cumulative releases in Table 4-6 are listed in Table 4-7. In R1, the highest releases of ^{234}U and both ^{230}Th are from V098. Similarly, in R2 and R3, the same vector that had the highest release of ^{234}U also had the highest release of ^{230}Th .

Table 4-7. Vector number corresponding to maximum cumulative release to WIPP LWB over 10,000 years (see Table 4-6)

| | partial | | | full | | |
|--------------------|---------|------|------|-------------------|-------------------|-------------------|
| | R1 | R2 | R3 | R1 | R2 | R3 |
| ^{241}Am | V018 | V099 | V053 | V033 | V099 [#] | V057 |
| ^{239}Pu | V071 | V050 | V040 | V100 | V050 [#] | V004 |
| ^{234}U | V098 | V033 | V036 | V098 | V088 | V036 [#] |
| ^{230}Th | V098 | V033 | V036 | V057 [#] | V012 | V084 |
| ^{230}ThA | V098 | V045 | V097 | V098 [#] | V045 | V097 |

[#] Indicates the largest maximum release for each radionuclide

Table 4-8 shows the median cumulative releases for each set of 100 vectors (including those less than the 10^{-9} kg threshold, which were left out of Table 4-5 and Figure 4-5). The median concentrations for all radionuclides in Table 4-8 are so low as to be physically unrealistic (not just ^{241}Am), and are essentially zero. These near-zero median concentrations indicate that under partial-mining conditions at least half of the vectors have essentially no simulated transport to the LWB.

Table 4-8. Median concentration at 10,000 years across all 100 vectors for each species and mining scenario

| | partial | | | full | | |
|--------------------|---------|---------|---------|---------|---------|---------|
| | R1 | R2 | R3 | R1 | R2 | R3 |
| ^{241}Am | 0.0e+00 | 0.0e+00 | 0.0e+00 | 1.3e-17 | 3.7e-17 | 3.6e-17 |
| ^{239}Pu | 0.0e+00 | 0.0e+00 | 0.0e+00 | 3.5e-14 | 3.2e-15 | 3.4e-14 |
| ^{234}U | 4.0e-39 | 0.0e+00 | 0.0e+00 | 3.2e-10 | 1.1e-09 | 2.0e-10 |
| ^{230}Th | 2.7e-38 | 0.0e+00 | 1.9e-39 | 2.1e-11 | 4.5e-12 | 8.6e-13 |
| ^{230}ThA | 0.0e+00 | 0.0e+00 | 0.0e+00 | 1.4e-13 | 1.2e-15 | 1.7e-14 |

4.2.2 Radionuclide Transport Under Full-Mining Conditions

Under full-mining conditions, all five simulated species had significant numbers of vectors transporting cumulative releases greater than the 10^{-9} kg threshold to the LWB during the course of the 10,000-year simulation (see the right half of Table 4-5). Releases greater than 10^{-9} kg were calculated for all simulated radionuclides for all three replicates.

The simulated constituents sorted by the numbers of vectors exceeding the 10^{-9} kg cumulative release threshold were ^{234}U , ^{230}Th , ^{230}ThA , ^{239}Pu , and ^{241}Am , in descending order. Nearly half of the ^{234}U vectors surpassed the 10^{-9} kg cumulative release threshold.

Figure 4-5 shows the distribution of these vectors with cumulative releases greater than the 10^{-9} kg threshold; there are more vectors plotted than in the partial-mining case, and they are on average at higher concentrations. Figure 4-5 shows that approximately 15 vectors from each full-mining replicate

transported at least 10% of the ^{234}U mass injected to the LWB in 10,000 years (tall bars in the ^{234}U row between 10^{-1} and 10^0).

Table 4-6 shows that the maximum full-mining ^{234}U releases (all three replicates) essentially transported the entire 1 kg of injected mass beyond the WIPP LWB. The identities of the vectors that produced the largest cumulative releases are listed in Table 4-7.

The five vectors with the largest cumulative releases to the LWB (all from the full-mining scenario) have their radionuclide distributions plotted at 10,000 years in Figure 4-6 through Figure 4-10, which all share the same \log_{10} concentration color scale.

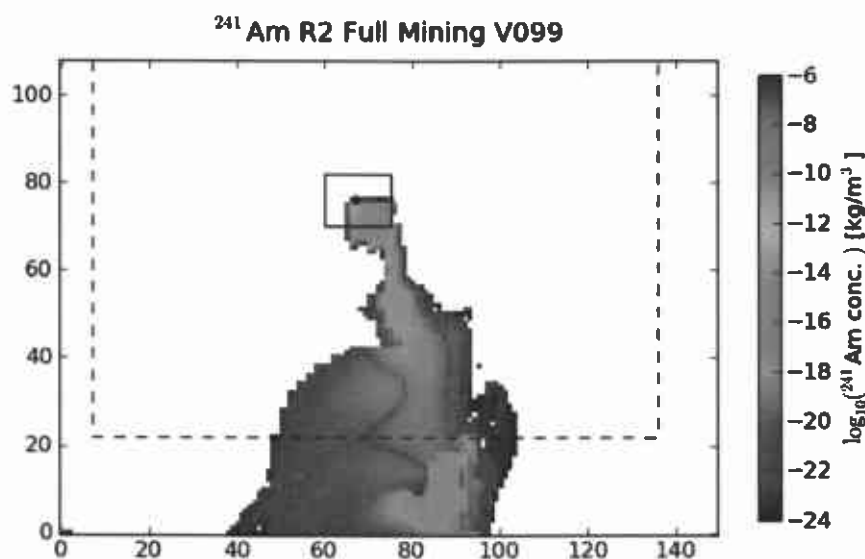


Figure 4-6. \log_{10} -distribution of ^{241}Am at 10,000 years for the vector with the largest cumulative release of ^{241}Am to the LWB (dashed line); SECOTP2D cell-based coordinates

Concentrations of ^{241}Am are relatively low compared to the other radionuclides, even for the vector with the largest cumulative release (see blues and greens in Figure 4-6). ^{234}U concentrations are shown to be highest south of the LWB (see darkest reds in Figure 4-8), consistent with the statement that nearly all the mass had been transported that far already.

The two ^{230}Th distributions are slightly different (they also correspond to different vectors and flow-fields). ^{230}ThA has higher concentrations near the release point (see Figure 4-10), while ^{230}Th is more evenly distributed because it is a decay product of ^{234}U (see Figure 4-9).

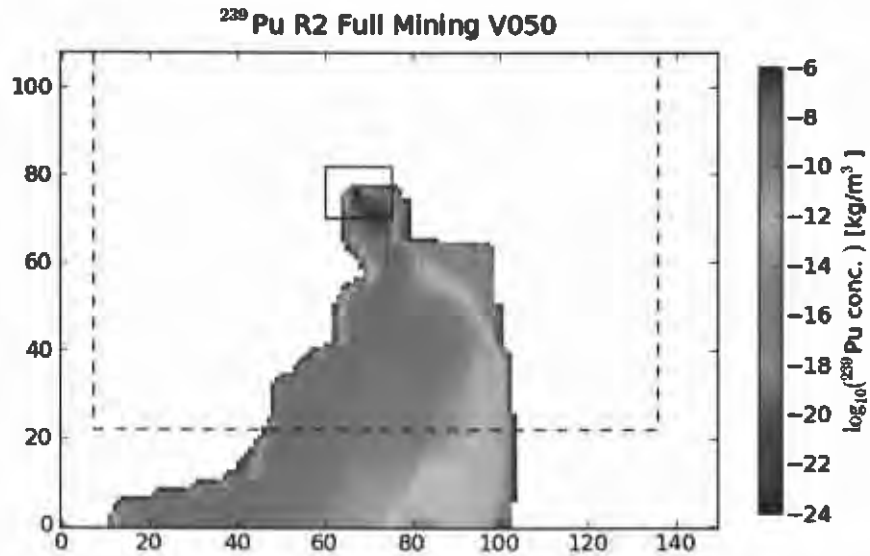


Figure 4-7. \log_{10} -distribution of ^{239}Pu at 10,000 years for the vector with the largest cumulative release of ^{239}Pu to the LWB (dashed line); SECOTP2D cell-based coordinates

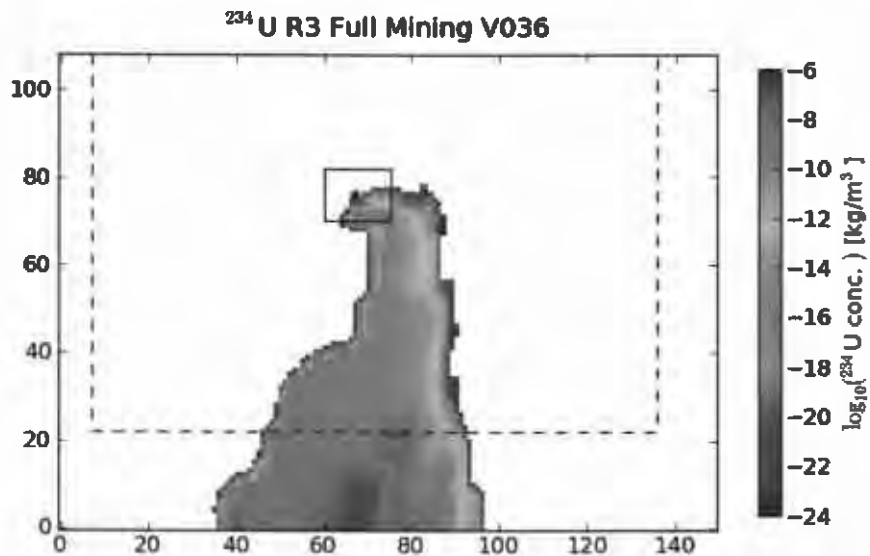


Figure 4-8. \log_{10} -distribution of ^{234}U at 10,000 years for the vector with the largest cumulative release of ^{234}U to the LWB (dashed line); SECOTP2D cell-based coordinates

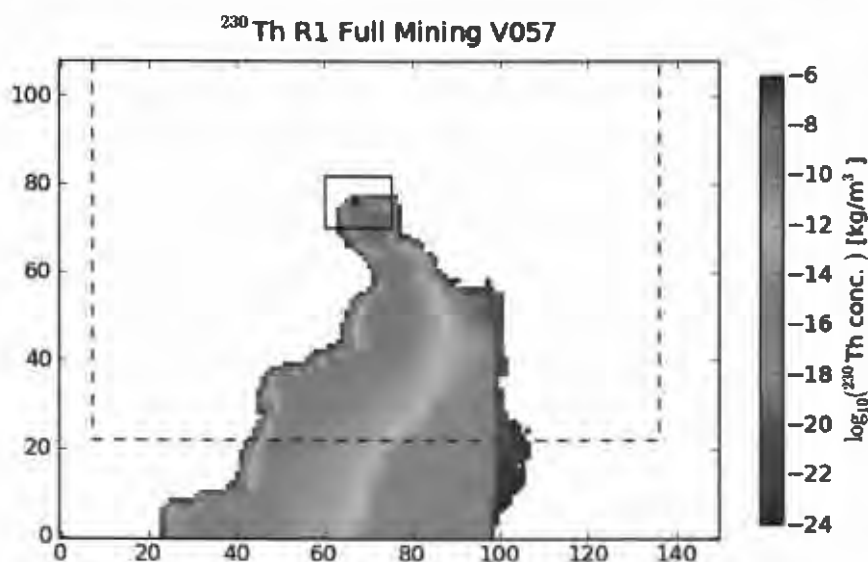


Figure 4-9. \log_{10} -distribution of ^{230}Th (daughter product) at 10,000 years for the vector with the largest cumulative release of ^{230}Th to the LWB (dashed line); SECOTP2D cell-based coordinates

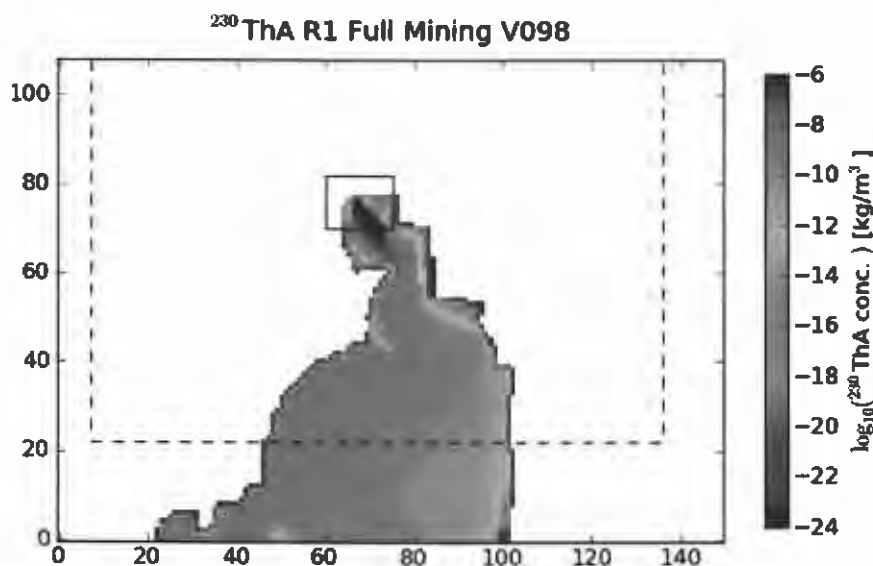


Figure 4-10. \log_{10} -distribution of ^{230}ThA (released ^{230}Th) at 10,000 years for the vector with the largest cumulative release of ^{230}ThA to the LWB (dashed line); SECOTP2D cell-based coordinates

4.3 Transport Summary

In summary, under partial-mining conditions, few vectors showed significant transport of radionuclides to the LWB during the 10,000-year simulation, while under full-mining conditions, a large portion of the vectors showed mass transport to the LWB. ^{234}U and its ^{230}Th daughter product had the largest cumulative masses transported to the WIPP LWB. Comparing these results to those for the PABC-2004, there is much more transport of radionuclides through the Culebra to the LWB. Similar to the PABC-

2004, ^{234}U and ^{230}Th have the largest number of vectors with significant releases to the WIPP LWB during the 10,000-year simulation.

These significant increases in radionuclide transport to the LWB can be attributed to three main changes since the PABC-2004:

1. **Speedup due to mining:** The definition of minable potash, obtained from the BLM (Cranston, 2009), has changed significantly, especially inside the LWB (see Figure 3-8). Minalable potash ore is now located in close proximity (approximately 670 m) to the center of the WIPP disposal panels.
2. **Speedup due to decrease in radionuclide retardation:** The lower limits of the matrix distribution coefficient (K_d) distributions were decreased several orders of magnitude, as requested by EPA (Kelly, 2009). Lower limits of the K_d ranges for Am(III) and Pu(III) were reduced from $2.0\text{E-}2$ to $5.0\text{E-}3$ m^3/kg ; lower limits for Pu(IV), Th(IV), and U(IV) were reduced from $7.0\text{E-}1$ to $5.0\text{E-}4$ m^3/kg ; the lower limit for U(VI) was not changed. Lower K_d values result in smaller retardation coefficients (see equation 11 in Section 4.1.4), and were requested to reflect the increase in organic ligand content in the WIPP inventory.
3. **Speedup due to MODFLOW calibration:** The consistent presence of a high-transmissivity zone in the southeastern portion of the WIPP site in all the calibrated realizations of the MODFLOW model (Hart et al., 2009) creates a pathway for radionuclides to leave the LWB (see Figure 4-3). This feature was confirmed through the interpretation of pumping tests at SNL-14, and the data from this test were included as calibration targets (Hart et al., 2009). The presence of this pathway leads to predictions of advective particle travel times to the LWB more consistent with CCA model predictions than PABC-2004 (see Figure 3-10).

5 Summary and Conclusions

The 100 transmissivity fields resulting from calibration to both steady-state and transient observed freshwater heads in the Culebra (Hart et al., 2009) were modified to account for potential effects due to mining potash from the Salado Formation above the repository. A definition of the areal extent of minable potash was obtained from the Bureau of Land Management (Cranston, 2009) and used to define areas where Culebra transmissivity was increased by a randomly sampled mining factor ($1 \leq \text{MINP_FACT} \leq 1000$). Two mining scenarios were developed: a full-mining scenario with all minable potash removed and a partial-mining scenario with only potash outside the WIPP LWB removed.

The mining-modified transmissivities were inputs to a MODFLOW flow model, which was used by DTRKMF to compute advective particle tracks from a release point at the center of the WIPP waste panels to the edge of the WIPP LWB. Results show that for the partial-mining scenario, the median particle travel time of 22,376 years is 3.03 times greater than for the non-mining scenario (7,374 years); the median particle travel time for the partial-mining scenario in PABC-2004 was 7.06 times greater than for the non-mining scenario. In contrast to the PABC-2004, the full-mining scenario decreased median travel time to 5,084 years, a factor of 1.45 *faster* than for the non-mining scenario; the median particle

travel time for the full-mining scenario in PABC-2004 was 3.84 times slower than for the non-mining scenario. For the partial-mining scenario, the increase in transmissivity due to mining increases the relative flow rate through the mining zones, with a corresponding decrease in flow through the non-mining zones. This decrease in flow through the non-mining zones produces longer travel times for the partial-mining scenario. For the full-mining scenario, the potash definition from BLM (Cranston, 2009) locates minable potash ore half the distance from the C-2737 release point than in PABC-2004 (see Figure 3-8). This new shortened distance from the release point to minable potash (in the full-mining scenario) reverses the slowing-down trend observed in the PABC-2004 analysis.

Like the PABC-2004 calculations, a very weak positive correlation was found between the travel times and the random mining factor (the higher the random mining factor, the longer the particle travel time – see Figure 3-21 and Figure 3-22). As the mining factor is increased, the flow through the non-mining areas is decreased, producing longer travel times and the positive correlation. Most of the advective particle travel time variability is due to differences in the base T-fields and not the random mining factor.

Culebra radionuclide transport calculations used the same MODFLOW flow-fields as DTRKMF, but refined the solution mesh to focus on a smaller subset of the domain focused on the WIPP LWB. The flow-fields computed using this smaller refined grid were additionally multiplied by a randomly sampled multiplier ($1 \leq \text{CLIMTIDX} \leq 2.25$) to account for potential increases in hydraulic gradients due to climate change (Corbet and Swift, 1996).

Many more Culebra radionuclide transport vectors showed significant movement of radionuclides to the LWB during the 10,000-year transport simulations under full-mining conditions, compared to PABC-2004. The mass of ^{234}U and its ^{230}Th daughter product transported to the WIPP LWB during the 10,000 year regulatory period were the largest of the five simulated radionuclides, with a few vectors essentially transporting the entire injected mass of ^{234}U to the LWB. The large increase in radionuclide transport can be attributed to three factors:

1. changes in the definition of minable potash,
2. changes in the lower limit of the K_d ranges, and
3. presence of a high-transmissivity pathway out of the southeast portion of the LWB, supported by the SNL-14 pumping test.

The cumulative mass of each radionuclide transported to the WIPP LWB, reported every 50 years, is the prediction used from this analysis in the subsequent CCDFGF analysis (Camphouse, 2010). The source term introduced in the Culebra in this modeling exercise is a unit mass at the beginning of the 10,000 year regulatory period, which is scaled and time-shifted based on other sampled parameters to produce predicted release to the accessible environment (the subsurface outside the WIPP LWB).

6 Bibliography

- Bertram, S.G., 1995. *Record of FEP Screening Work, FEP ID#NS-11: Subsidence Associated with Mining Inside or Outside the Controlled Area*. Carlsbad, NM: Sandia National Laboratories. ERMS 230761.
- Camphouse, R.C., 2010. *Analysis Package for CCDFGF: CRA-2009 Performance Assessment Baseline Calculation*. Carlsbad, NM: Sandia National Laboratories. In Progress.
- Chavez, M., 2006. *NP 19-1 Software Requirements, Revision 12*. Carlsbad, NM: Sandia National Laboratories.
- Chavez, M., 2008. *NP 9-1 Analyses, Revision 7*. Carlsbad, NM: Sandia National Laboratories.
- Clayton, D.J., 2009a. *Analysis Plan for the CRA-2009 Performance Assessment Baseline Calculation, AP-145 Revision 0*. Carlsbad, NM: Sandia National Laboratories. ERMS 551603.
- Clayton, D.J., 2009b. *Update to the K_d values for the PABC-2009*. Carlsbad, NM: Sandia National Laboratories. ERMS 552395.
- Clayton, D.J., Camphouse, R.C., Garner, J.W., Ismail, A.E., Kirchner, T.B., Kuhlman, K.L. and Nemer, M.B., 2010. *Summary Report of the CRA-2009 Performance Assessment Baseline Calculation*. Carlsbad, NM: Sandia National Laboratories. In progress.
- Clayton, D.J., Dunagan, S., Garner, J.W., Ismail, A.E., Kirchner, T.B., Kirkes, G.R. and Nemer, M.B., 2008. *Summary Report of the 2009 Compliance Recertification Application Performance Assessment*. Carlsbad, NM: Sandia National Laboratories. ERMS 548862.
- Corbet, T., 1995. *Record of FEP Screening Work FEP ID# NS-9 Justification of SECO2D Approximation for PA Transport Calculation*. Carlsbad, NM: Sandia National Laboratories. ERMS 230802.
- Corbet, T. and Swift, P., 1996. *Parameters required for SECOFL2D: Climate Index*. Carlsbad, NM: Sandia National Laboratories. ERMS 237465.
- Cotsworth, E., 2005. *EPA Letter on Conducting the Performance Assessment Baseline Change (PABC) Verification Test*. Washington, D.C.: U.S. Environmental Protection Agency, Office of Radiation and Indoor Air. ERMS 538858.
- Cotsworth, E., 2009. *EPA CRA-2009 First Set of Completeness Comments*. Washington, D.C.: U.S. Environmental Protection Agency, Office of Radiation and Indoor Air. ERMS 551444.
- Cranston, C.C., 2009. *Minable Patash Ore*. Carlsbad, NM: Bureau of Land Management. ERMS 551120.
- ESRI, 1998. *ESRI Shapefile Technical Description - An ESRI White Paper*. Redlands, CA: Environmental Systems Research Institute, Inc.
- Harbaugh, A.W., Banta, E.R., Hill, M.C. and McDonald, M.G., 2000. *MODFLOW 2000: The U.S. Geological Survey Modular Ground-Water Model User Guide*. Reston, VA: U.S. Geological Survey. OFR 00-92.

- Hart, D.B., 2010. *WIPP PA Validation Document for MODFLOW-2000 Version 1.6: addenda for direct solver*. Carlsbad, NM: Sandia National Laboratories. ERMS 552422.
- Hart, D.B., Beauheim, R.L. and McKenna, S.A., 2009. *Analysis Report for Task 7 of AP-114: Calibration of Culebra Transmissivity Fields*. Carlsbad, NM: Sandia National Laboratories. ERMS 552391.
- Hart, D.B., Holt, R.M. and McKenna, S.A., 2008. *Analysis Report for Task 5 of AP-114: Generation of Revised Base Transmissivity Fields*. Carlsbad, NM: Sandia National Laboratories. ERMS 549597.
- Holt, R.M., 1997. *Conceptual Model for Transport Processes in the Culebra Dolomite Member of the Rustler Formation*. Carlsbad, NM: Sandia National Laboratories. SAND97-0194.
- Holt, R.M. and Powers, D.W., 1988. *Facies Variability and Post-Depositional Alteration Within the Rustler Formation in the Vicinity of the Waste Isolation Pilot Plant, Southeastern New Mexico*. Carlsbad, NM: WIPP Project Office. ERMS 242145.
- Johnson, P.B., 2009. *Routine Calculations Report In Support of Task 6 of AP-114: Potentiometric Surface, Adjusted to Equivalent Freshwater Heads, of the Culebra Dolomite member of the Rustler Formation near the WIPP Site, Revision 2*. Carlsbad, NM: Sandia National Laboratories. ERMS 551116.
- Kelly, T.E., 2009. *EPA Third Letter Requesting Additional Information on the CRA-2009*. Washington, D.C.: U.S. EPA Office of Radiation and Indoor Air. ERMS 552374.
- Kirchner, T., 2008. *User's Guide to CVS, ReadScript.py, and Related Utilities, Version 1.00*. Carlsbad, NM: Sandia National Laboratories. ERMS 550579.
- Kirchner, T.B., 2010. *Generation of the LHS Samples for the AP-145 (PABC09) PA calculations*. Carlsbad, NM: Sandia National Laboratories. In Progress.
- Kuhlman, K.L., 2009. *AP-144 Analysis Plan for the Calculation of Culebra Flow and Transport for CRA2009PABC*. Sandia National Laboratories. ERMS 551676.
- Leigh, C.D., Kanney, J.F., Brush, L.H., Garner, J.W., Kirkes, G.R., Lowry, T., Nemer, M.B., Stein, J.S., Vugrin, E.D., Wagner, S. and Kirchner, T.B., 2004. *2004 Compliance Recertification Application Performance Assessment Baseline Calculation, Revision 0*. Carlsbad, NM: Sandia National Laboratories. ERMS 541521.
- Long, J.J., 2010. *Execution of Performance Assessment Codes for the CRA-2009 Performance Assessment Baseline Calculation*. Carlsbad, NM: Sandia National Laboratories. In Progress.
- Lowry, T.L., 2003a. *Analysis Report, Task 5 of AP-088: Evaluation of Mining Scenarios*. Carlsbad, NM: Sandia National Laboratories. ERMS 531138.
- Lowry, T.L., 2003b. *Analysis Report, Tasks 2 & 3 of AP-100: Grid Size Conversion and Generation of SECOTP2D Input*. Carlsbad, NM: Sandia National Laboratories. ERMS 531137.
- Lowry, T.S., 2004. *Analysis Report for Inclusion of Omitted Areas in Mining Transmissivity Calculations in Response to EPA Comment G-11*. Carlsbad, NM: Sandia National Laboratories. ERMS 538218.

- Lowry, T.S. and Kanney, J.F., 2005. *Analysis Report for the CRA-2004 PABC Culebra Flow and Transport Calculations*. Carlsbad, NM: Sandia National Laboratories. ERMS 541508.
- MacKinnon, R.J. and Freeze, G., 1997a. *Summary of EPA-Mandated Performance Assessment Verification Test (Replicate 1) and Comparison With the Compliance Certification Application Calculations, Revision 1*. Carlsbad, NM: Sandia National Laboratories. ERMS 422595.
- MacKinnon, R.J. and Freeze, G., 1997b. *Summary of Uncertainty and Sensitivity Analysis Results for the EPA-Mandated Performance Assessment Verification Test, Rev 1*. Carlsbad, NM: Sandia National Laboratories. ERMS 420669.
- MacKinnon, R.J. and Freeze, G., 1997c. *Supplemental Summary of EPA-Mandated Performance Assessment Verification Tests (All Replicates) and Comparison With the Compliance Certification Application Calculations, Revision 1*. Carlsbad, NM: Sandia National Laboratories. ERMS 414880.
- Meigs, L.C., Beauheim, R.L. and Jones, T.L., eds., 2000. *Interpretation of Tracer Tests Performed in the Culebra Dolomite at the Waste Isolation Pilot Plant Site*. Carlsbad, NM: Sandia National Laboratories. SAND97-3109.
- Powers, D.W., Lambert, S.J., Shaffer, S.-E., Hill, L.R. and Weart, W.D., eds., 1978. *Geological Characterization Report - (WIPP)*. Carlsbad, NM: Sandia National Laboratories. SAND78-1596.
- Ramsey, J.L., 1997. *WIPP PA User's Manual for SECOTP2D, Version 1.41*. Carlsbad, NM: Sandia National Laboratories. ERMS 245734.
- Ramsey, J.L. and Wallace, M., 1996. *Analysis Package for the Culebra Flow and Transport Calculations (Task 3) of the Performance Assessment Analyses Supporting the Compliance Certification Application*. Carlsbad, NM: Sandia National Laboratories. ERMS 240516.
- Rudeen, D.K., 2003. *User's Manual for DTRKMF Version 1.00*. Carlsbad, NM: Sandia National Laboratories. ERMS 523246.
- Salari, K. and Blaine, R., 1996. *WIPP PA User's Manual for SECOTP2D, Version 1.30*. Carlsbad, NM: Sandia National Laboratories. ERMS 236695.
- U.S. DOE, 1996. *Title 40 CFR Part 191 Compliance Certification Application for the Waste Isolation Pilot Plant*. Carlsbad, NM: U.S. Department of Energy Waste Isolation Pilot Plant, Carlsbad Area Office. DOE/CAO-1996-2184.
- U.S. DOE, 2004. *Title 40 CFR Part 191 Compliance Recertification Application for the Waste Isolation Pilot Plant*. Carlsbad, NM: U.S. Department of Energy Waste Isolation Pilot Plant, Carlsbad Field Office. DOE/WIPP 2004-3231.
- U.S. EPA, 1993. *Title 40 CFR Part 191: Environmental Radiation Protection Standards for the Management and Disposal of Spent Nuclear Fuel, High-Level and Transuranic Radioactive Wastes; Final Rule*. Federal Register, Vol 58, 66398-66416.

U.S. EPA, 1996. *Title 40 CFR Part 194: Criteria for the Certification and Recertification of the Waste Isolation Pilot Plant's Compliance with the 40 CFR Part 191 Disposal Regulations; Final Rule*. Federal Register, Vol. 61, 5223-5245.

U.S. EPA, 1998. *Criteria for the Certification and Recertification of the Waste Isolation Pilot Plant's Compliance with the Disposal Regulations: Certification Decision: Final Rule*. Federal Register, Vol. 63. ERMS 251924.

Vine, J.D., 1963. *Surface Geology of the Nash Draw Quadrangle, Eddy County, New Mexico*. U.S. Geological Survey. Bulletin 1141-B.

Appendix 1 Mining Modification Process Narrative

This appendix describes each step of the mining modification process in sufficient detail to allow it to be reproduced. Tables of input and output files are listed in the Culebra flow chapter of the run control document for PABC-2009 (Long, 2010)

The mining modification (i.e., non-VMS) portion of the analysis in this report can be broken down into 9 major steps:

0. Convert mining areas delineated by a binary ESRI shapefile to ASCII MODFLOW grid data;
1. Checkout AP-114 Task 7 results from CVS repository `Tfields` and modify directory structure associated with realization names (remove `Update` & `Update2` from some of the path names);
2. Enlarge mining areas to include both MODFLOW cells directly over mined areas and cells affected by nearby mined areas (i.e., within the 45° angle-of-draw);
3. Modify hydraulic conductivity fields and internal boundary conditions according to mining shape definition (from step 3) and mining factors from LHS on VMS (see Section A2.5);
4. Run MODFLOW and DTRKMF for each of 700 scenarios (600 mined + 100 non-mined) using 100 meter square elements;
5. Post-process DTRKMF output for plotting CDF and particle track figures.
6. Convert MODFLOW input data from 100 m to 50 m elements (4× total number of elements);
7. Run MODFLOW for each of the 600 mined scenarios using 50 m elements; and
8. Convert MODFLOW budget files from binary to ASCII, renaming them for ftp transfer to VMS to be used as input to VTRAN2 (see Appendix 3).

The commercial off-the-shelf software (COTS) used in the mining modification portion of the analysis is summarized in Table A1-1, and software are discussed when used in the rest of this appendix.

Table A1-1. Commercial off-the-shelf software used in mining modification analysis

| Program | Version | Platform | Use |
|----------|--------------------|------------|-------------------------------|
| Python | 2.3.4 | Linux | script interpreter |
| Bash | 3.00.15(1) | Linux | script interpreter |
| MATLAB | 7.9.0.529 (R2009b) | Windows XP | script interpreter / plotting |
| Gnuplot | 4.2.3 | Windows XP | plotting |
| MS-Excel | 2007-SP1 | Windows XP | plotting / regression |
| Surfer | 9.7.543 | Windows XP | plotting |

A1.1 Step 0 – Shapefile Conversion and Mapping to Grid

The first step is a binary-to-ASCII file conversion step (in Python) followed by a mapping of the results onto the MODFLOW grid (in MATLAB), and is the only portion of the analysis in this appendix not done on the WIPP PA Pentium 4 cluster. Step 0 is run on a Dell Precision 690 workstation (with two Intel Xeon processors) running Windows XP, Service Pack 2. The shapefile and scripts are located in the CVS repository `MiningMod` with the other files used in the Linux portion of this analysis (in the archive

potash_shapefile.zip, located in the package Auxiliary). The conversion portion of this step was done using the Python script `convert_shapefile_to_ASCII.py` (see §A4.1 for script listing and Table A1-2 for input/output files), which converts the binary ESRI shapefile polyline data that is used to define the extent of the minable potash ore into a series of ASCII text files (the specification for ESRI shapefiles is outlined in an ESRI (1998) white paper). The single shapefile is comprised of all the files with the base name `Measured_Minable_Ore_UTMNAD27` (see file listing in Table A1-5), but the single binary file with the extension `.shp` contains the polyline data defining the extent of the potash. Since the mining definition is specified as a doubly-connected region (i.e., an area with holes cut out of it), care must be taken to handle polylines nested inside one another (see Figure A1-1); it is known *a priori* that none of the individual polylines intersect.

Table A1-2. Input and output files for Python script `convert_shapefile_to_ASCII.py`

| File | Remark |
|--|--|
| Inputs¹ | |
| <code>convert_shapefile_to_ASCII.in</code> | Python input listing filenames |
| <code>Measured_Minable_Ore_UTMNAD27.shp</code> | single file from potash shapefile containing coordinates of polylines defining potash areas |
| Outputs | |
| <code>polyline_dump_matlab_partnnnn.dat</code> | 54 ASCII listings of UTM NAD27 coordinates each containing data for an individual polyline |
| <code>polyline_dump_for_gnuplot.dat</code> | ASCII listing of all parts, each separated by two blank lines for plotting to check conversion |
| 1. $nnnn \in \{0000,0001, \dots, 0053\}$ | |

The shapefile has two sets of data important to the present use; the main portion is a listing of double-precision NAD27 UTM x, y coordinates (Zone 13) of the vertices in the polyline in meters (192,528 pairs). The second set of data in the shapefile (needed for this analysis) is a listing of the 53 integer line numbers where this single long list of coordinates is broken into 54 parts.

The 54 parts defined in the shapefile used here are each saved into a different text file (to be read back in by the MATLAB script in the next step), and all the data is also saved into a single text file for plotting and checking in Gnuplot and Surfer; the multi-file (lines 172-191 of §A4.1.2) and single-file (lines 144-170) formats have the same information in them (see Table A1-3 for a comparison of the first few points in two corresponding files). The original binary shapefile and the ASCII files were plotted simultaneously in the program Surfer as a check that the data were extracted from the binary file correctly. Surfer has the ability to plot ESRI shapefiles and the text format exported by the scripts described in this section. The original shapefile and the exported text files were found to be the same.

Table A1-3. Listing of the first five data points in the files `polyline_dump_matlab_part0000.dat` and `polyline_dump_for_gnuplot.dat`.

| | |
|------------------------|---|
| 589072.390 3612760.518 | # data from Measured_Minable_Ore_UTMNAD27.shp |
| 589078.150 3612765.663 | # part: 0 |
| 589077.955 3612765.806 | 589072.4 3612760.5 |
| 589072.857 3612769.527 | 589078.1 3612765.7 |
| 589067.671 3612773.240 | 589078.0 3612765.8 |
| | 589072.9 3612769.5 |
| | 589067.7 3612773.2 |

The MATLAB script `import_polyline_determine_mined_areas.m` (§A4.2 and Table A1-4) utilizes the built-in MATLAB function `inpolygon()` to compare coordinates of model-cell centers to the polylines exported to ASCII in the previous step, determining whether each model cell is inside or outside each of the 54 potash polygons. For the mining-area definitions, none of the polygons intersect and the most of the smaller polygons are inside the largest polygon (the polyline corresponding to part 1), and therefore define small cutout areas where no minable potash is found inside a larger potash area. Model cells are checked with respect to each of the 54 parts of the overall shapefile (lines 58-63 of the MATLAB script, §A4.2). The final calculation of whether a cell is minable or not is determined by comparing what types of polygons (e.g., black, red, and green in Figure A1-1) the cell is located within (lines 65-73 in §A4.2). The following cases are exhaustive for the given set of data:

1. cells outside all polygons have no potash (white areas in Figure A1-1);
2. cells inside one polygon have potash (either black or green areas in Figure A1-1); and
3. cells inside more than one polygon have no potash (red areas in Figure A1-1), because there are no triply-nested polylines.

The two output files `mining_areas_matrix.dat` and `mining_areas_xyz.dat` contain the listing of the cells inside (value of 1) and outside (value of 0) in two different formats (saved as a list of *x, y* coordinates and mining index for plotting in Gnuplot and also saved as a matrix of mining indices, with no coordinates – used later in the analysis).

Table A1-4. Input and output files for MATLAB script `import_polyline_determine_mined_areas.m`

| File | Remark |
|--|--|
| Inputs¹ | |
| <code>polyline_dump_matlab_partnnnn.dat</code> | 54 ASCII listings of UTM NAD27 coordinates each containing data for an individual polyline (see outputs in Table A1-2) |
| Outputs | |
| <code>mining_areas_matrix.dat</code> | ASCII matrix with indices indicating MODFLOW model cells with minable potash |
| <code>mining_areas_xyz.dat</code> | ASCII <i>x,y,z</i> listing of MF2K model cells with minable potash and their UTM NAD27 coordinates |

1. *nnnn* ∈ {0000,0001,...,0053}

This process of mapping the shapefile provided by BLM onto the model grid was done in one step for PABC-2004 using the COTS Windows software GMS (Lowry, 2003a). This approach was attempted, but problems appeared associated with the way GMS 7.0 maps the current shapefile, resulting in GMS crashing and not performing the mapping. Technical and developer support from the GMS vendor could not resolve the problem, aside from the assurance that this will be fixed in some future release. Due to this limitation, the manipulation and mapping of the shapefiles to the MODFLOW model grid is given explicitly in terms of the Python and MATLAB scripts here listed and described.

Table A1-5. Directory listing of input scripts, input shapefile, intermediate files, and output files for shapefile conversion (step 0). Files are located in a zip archive saved in the CVS repository MiningMod in the package Auxiliary.

```

C:\shapefile_conversion>dir
Volume in drive C is DriveC
Volume Serial Number is 542A-10F7

Directory of C:\shapefile_conversion

10/07/2009  01:49 PM    <DIR>          .
10/07/2009  01:49 PM    <DIR>          ..
09/30/2009  04:40 PM                93 convert_shapefile_to_ASCII.in
10/07/2009  10:30 AM            9,184 convert_shapefile_to_ASCII.py
09/30/2009  04:59 PM            2,258 import_polyline_determine_mined_areas.m
06/22/2009  05:27 PM                178 Measured_Minable_Ore_UTMNAD27.dbf
06/22/2009  05:27 PM            426 Measured_Minable_Ore_UTMNAD27.prj
06/22/2009  05:27 PM            132 Measured_Minable_Ore_UTMNAD27.sbn
06/22/2009  05:27 PM            116 Measured_Minable_Ore_UTMNAD27.sbx
06/22/2009  05:27 PM    3,080,816 Measured_Minable_Ore_UTMNAD27.shp
06/22/2009  05:27 PM            2,239 Measured_Minable_Ore_UTMNAD27.shp.xml
06/22/2009  05:27 PM                108 Measured_Minable_Ore_UTMNAD27.shx
09/28/2009  11:06 AM    1,395,622 mining_areas_matrix.dat
09/28/2009  11:06 AM    4,359,400 mining_areas_xyz.dat
09/30/2009  04:35 PM    3,851,461 polyline_dump_for_gnuplot.dat
09/30/2009  04:35 PM    3,240,336 polyline_dump_matlab_part0000.dat
09/30/2009  04:35 PM            766,608 polyline_dump_matlab_part0001.dat
09/30/2009  04:35 PM            49,872 polyline_dump_matlab_part0002.dat
09/30/2009  04:35 PM            58,944 polyline_dump_matlab_part0003.dat
09/30/2009  04:35 PM            33,288 polyline_dump_matlab_part0004.dat
09/30/2009  04:35 PM            33,504 polyline_dump_matlab_part0005.dat
09/30/2009  04:35 PM            8,568 polyline_dump_matlab_part0006.dat
09/30/2009  04:35 PM            16,152 polyline_dump_matlab_part0007.dat
09/30/2009  04:35 PM            20,832 polyline_dump_matlab_part0008.dat
09/30/2009  04:35 PM            6,552 polyline_dump_matlab_part0009.dat
09/30/2009  04:35 PM            12,024 polyline_dump_matlab_part0010.dat
09/30/2009  04:35 PM            9,912 polyline_dump_matlab_part0011.dat
09/30/2009  04:35 PM            11,136 polyline_dump_matlab_part0012.dat
09/30/2009  04:35 PM            14,832 polyline_dump_matlab_part0013.dat
09/30/2009  04:35 PM            8,280 polyline_dump_matlab_part0014.dat
09/30/2009  04:35 PM            10,776 polyline_dump_matlab_part0015.dat
09/30/2009  04:35 PM            6,096 polyline_dump_matlab_part0016.dat
09/30/2009  04:35 PM            5,568 polyline_dump_matlab_part0017.dat
09/30/2009  04:35 PM            7,320 polyline_dump_matlab_part0018.dat
09/30/2009  04:35 PM            9,744 polyline_dump_matlab_part0019.dat
09/30/2009  04:35 PM            10,200 polyline_dump_matlab_part0020.dat
09/30/2009  04:35 PM            4,680 polyline_dump_matlab_part0021.dat
09/30/2009  04:35 PM            4,056 polyline_dump_matlab_part0022.dat
09/30/2009  04:35 PM            7,632 polyline_dump_matlab_part0023.dat
09/30/2009  04:35 PM            5,784 polyline_dump_matlab_part0024.dat
09/30/2009  04:35 PM            5,544 polyline_dump_matlab_part0025.dat
09/30/2009  04:35 PM            5,400 polyline_dump_matlab_part0026.dat
09/30/2009  04:35 PM            3,864 polyline_dump_matlab_part0027.dat
09/30/2009  04:35 PM            5,448 polyline_dump_matlab_part0028.dat
09/30/2009  04:35 PM            4,128 polyline_dump_matlab_part0029.dat
09/30/2009  04:35 PM            3,432 polyline_dump_matlab_part0030.dat
09/30/2009  04:35 PM            6,552 polyline_dump_matlab_part0031.dat
09/30/2009  04:35 PM            4,512 polyline_dump_matlab_part0032.dat
09/30/2009  04:35 PM            5,592 polyline_dump_matlab_part0033.dat
09/30/2009  04:35 PM            4,824 polyline_dump_matlab_part0034.dat
09/30/2009  04:35 PM            3,480 polyline_dump_matlab_part0035.dat
09/30/2009  04:35 PM            2,808 polyline_dump_matlab_part0036.dat
09/30/2009  04:35 PM            3,672 polyline_dump_matlab_part0037.dat
09/30/2009  04:35 PM            2,568 polyline_dump_matlab_part0038.dat
09/30/2009  04:35 PM            1,944 polyline_dump_matlab_part0039.dat
09/30/2009  04:35 PM            1,440 polyline_dump_matlab_part0040.dat
09/30/2009  04:35 PM            24,888 polyline_dump_matlab_part0041.dat
09/30/2009  04:35 PM            27,888 polyline_dump_matlab_part0042.dat
09/30/2009  04:35 PM            1,512 polyline_dump_matlab_part0043.dat
09/30/2009  04:35 PM            2,472 polyline_dump_matlab_part0044.dat
09/30/2009  04:35 PM            1,152 polyline_dump_matlab_part0045.dat
09/30/2009  04:35 PM            360 polyline_dump_matlab_part0046.dat
09/30/2009  04:35 PM            264 polyline_dump_matlab_part0047.dat
09/30/2009  04:35 PM            24,072 polyline_dump_matlab_part0048.dat
09/30/2009  04:35 PM            27,000 polyline_dump_matlab_part0049.dat
09/30/2009  04:35 PM            11,544 polyline_dump_matlab_part0050.dat
09/30/2009  04:35 PM            22,032 polyline_dump_matlab_part0051.dat
09/30/2009  04:35 PM            8,664 polyline_dump_matlab_part0052.dat
09/30/2009  04:35 PM            40,920 polyline_dump_matlab_part0053.dat
67 File(s)          17,322,705 bytes
 2 Dir(s)          149,001,261,056 bytes free

```

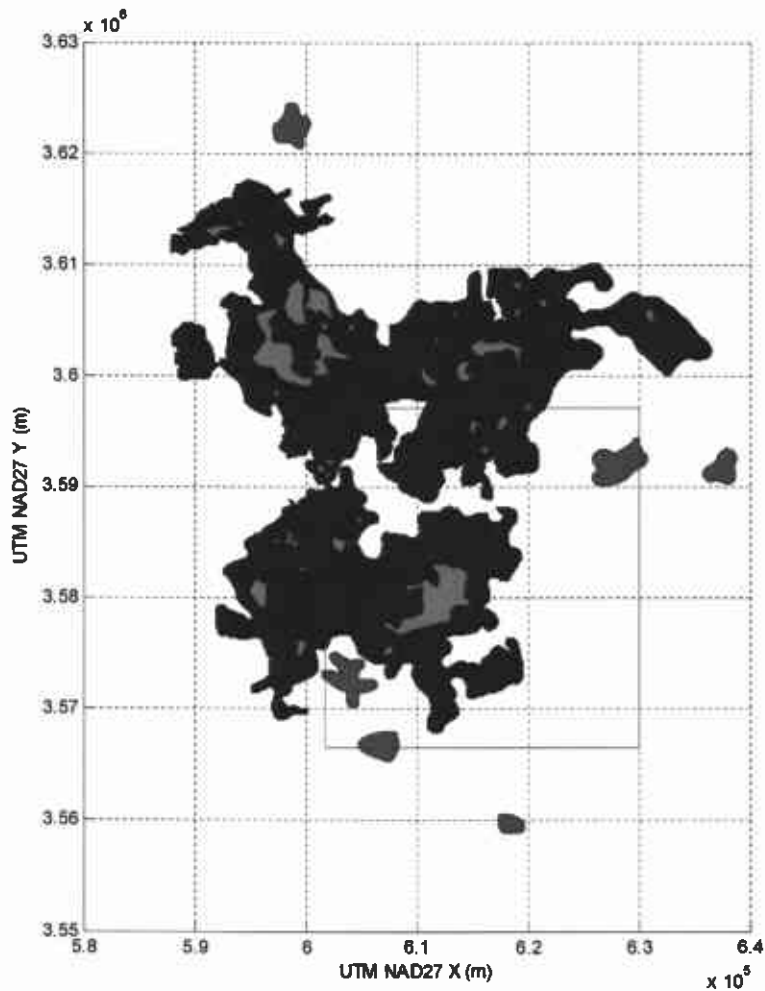


Figure A1-1. 54 polygons defining mining areas, color-coded by their relationship to the large black polygon. Green are outside the large polygon and define mined areas, red are inside the black polygon and define non-mined areas. Blue rectangle indicates PABC-2009 MODFLOW model extents.

The ASCII file `mining_areas_xyz.dat` was uploaded to the Pentium 4 PA cluster and added to the CVS repository `MiningMod` in the package `Inputs`, to be used as input to the Python script that enlarges mining areas to account for the angle-of-draw effect (see next section).

Figure A1-1 illustrates the relationship between the different parts of the polyline and shows the extent of the minable potash beyond the MODFLOW model domain (the area plotted in Figure A1-2). Figure A1-2 indicates the MODFLOW cells (small blue dots) that were determined to be located in mined areas. Both figures were generated by the MATLAB script (§A4.2: Figure A1-1 in lines 28, 35, 37, 41-43 and Figure A1-2 lines 76-77) that computes the mapping of model cells onto the potash mining definitions.

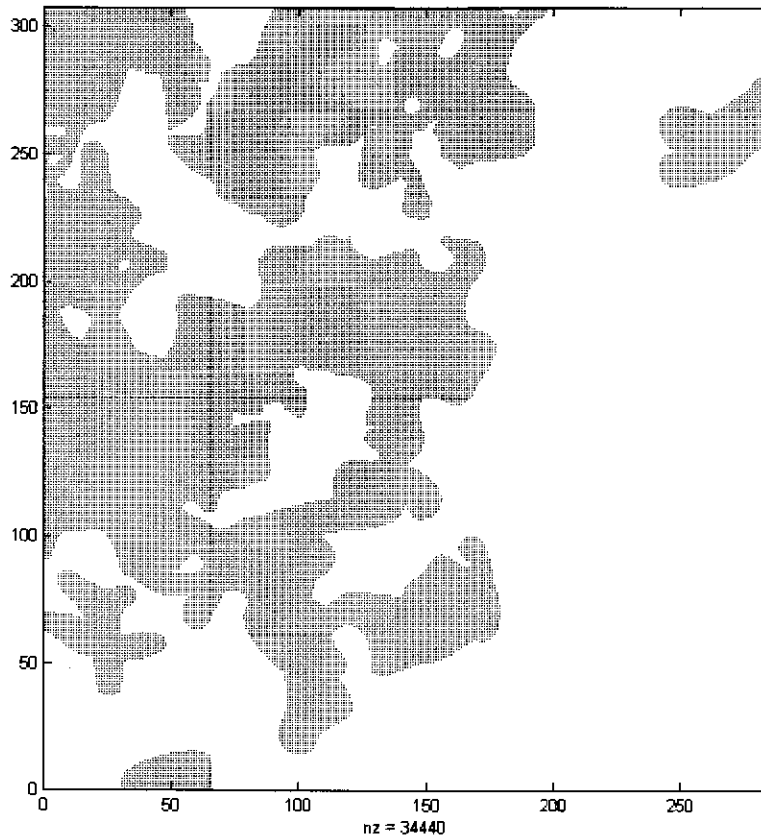


Figure A1-2. MODFLOW model domain with mined model cells marked with a small blue dot. 34,440 of the total 87,188 elements have mining (some cells in the northwest portion of the domain are inactive – see Figure 3-2)

A1.2 Step 1 – MODFLOW File Checkout and Directory Re-structuring

The creation of the directory structure and the process of checking out AP-114 Task 7 MODFLOW files from the CVS repository `Tfield` are handled by the QA-controlled Python program `ReadScript.py`, using the input file `mining_mod.inp` (itself saved in the `RunControl` package of the `MiningMod` CVS repository). See Figure A1-3 for the directory structure in the CVS repository used in this analysis. The Bash shell script `run_mining_mods.sh` (§A4.3) is the main script called by `ReadScript.py` (see last line of input file); this Bash script calls several other scripts and performs minor file manipulations (see Figure A1-4 for the dependency of scripts). The 100 realizations of flow models resulting from AP-114 Task 7 (Hart et al., 2009) are not in a flat directory structure in the CVS repository `Tfields`; most of the 100 final realizations were updated (sometimes twice) as part of the re-calibration effort associated with the freshwater head value at SNL-8 (Hart et al., 2009). The Bash script `run_mining_mods.sh` processes the directory structure and creates the intermediate file `keepers_short` at lines 25-45 (§A4.3).

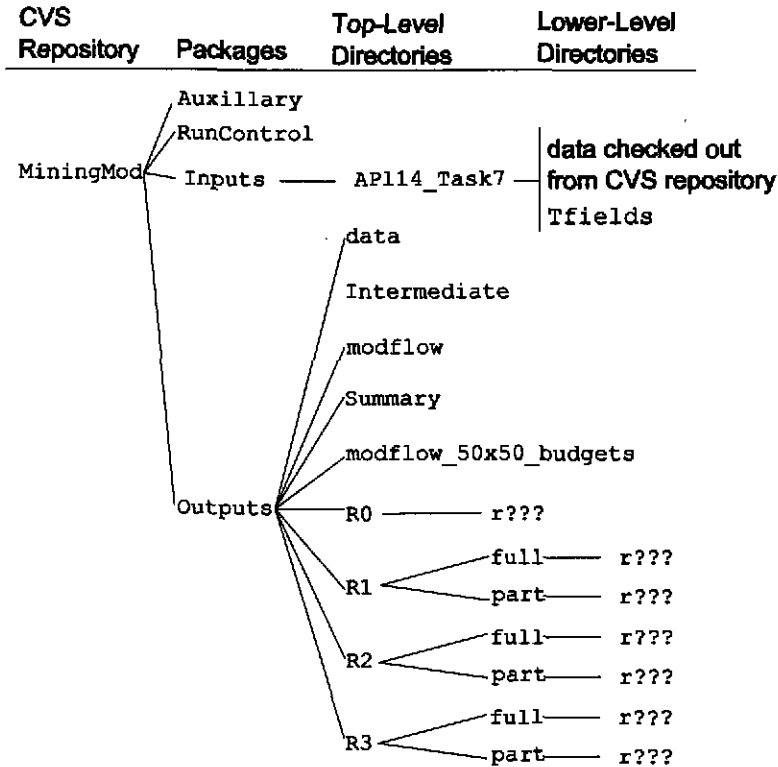


Figure A1-3. Directory structure of MiningMod CVS repository; r??? indicates the 100 realizations (not numbered sequentially) selected in Hart et al., (2009) after simplifying the directory structure (see Table A1-6).

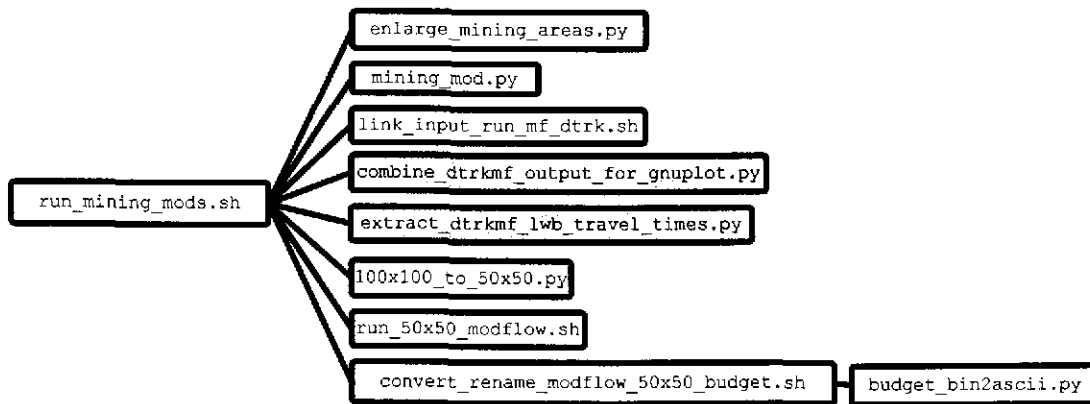


Figure A1-4. Dependency between scripts used in mining modifications on Pentium 4 PA cluster. In the middle column, the execution order is from top to bottom. The two gray boxes indicate scripts only used for post-processing DTRKMF output.

The scripts and input files used in this analysis are in the Inputs package of the CVS repository; the input files used directly from AP-114 Task 7 are exported from that CVS repository (Tfields) into the directory Inputs/AP114_Task7 for use in this analysis, but are not checked back into the MiningMod CVS repository. The Outputs package contains files generated as part of this analysis, or

input files modified from their original state in AP-114 Task 7. The two subdirectories `data` and `modflow` in the `Outputs` package (see Figure A1-3) contain MODFLOW input files that are modified for the 50 m grid from their use in the original 100 m grid. The `Intermediate` top-level subdirectory contains results from intermediate steps in the analysis that are checked back into CVS for completeness. The `Summary` top-level directory contains post-processed DTRKMF results used for plotting figures. The `R0`, `R1`, `R2`, and `R3` top-level directories contain the subdirectories where MODFLOW is executed for each realization; each `r???` leaf in Figure A1-3 represents 100 subdirectories such as `r007`, `r203`, `r861`, etc. (see Table A1-6). The `Auxiliary` package contains a zip archive with the shapefile and scripts associated with step 0 and an MS-Excel spreadsheet associated with the regression analysis shown in Figure 3-21 and Figure 3-22, neither of which are run on the Pentium 4 PA cluster.

Table A1-6. Listing of file keepers (original column) with equivalent short name (short column) after modification by `run_mining_mods.sh`

| ID | Original | Short | ID | Original | Short | ID | Original | Short | ID | Original | Short |
|----|--------------|-------|----|--------------|-------|----|--------------|-------|-----|--------------|-------|
| 1 | Update/r256 | r256 | 26 | Update/r060 | r060 | 51 | Update/r883 | r883 | 76 | Update/r002 | r002 |
| 2 | Update/r902 | r902 | 27 | Update/r012 | r012 | 52 | Update/r045 | r045 | 77 | Update/r053 | r053 |
| 3 | Update/r038 | r038 | 28 | r078 | r078 | 53 | Update2/r137 | r137 | 78 | Update/r861 | r861 |
| 4 | Update/r486 | r486 | 29 | r104 | r104 | 54 | Update/r279 | r279 | 79 | Update2/r465 | r465 |
| 5 | Update/r984 | r984 | 30 | Update/r052 | r052 | 55 | Update/r694 | r694 | 80 | Update/r511 | r511 |
| 6 | r028 | r028 | 31 | Update2/r007 | r007 | 56 | Update/r361 | r361 | 81 | Update/r191 | r191 |
| 7 | Update/r298 | r298 | 32 | r102 | r102 | 57 | Update/r040 | r040 | 82 | Update/r142 | r142 |
| 8 | Update/r791 | r791 | 33 | Update/r809 | r809 | 58 | Update/r571 | r571 | 83 | Update/r061 | r061 |
| 9 | Update/r910 | r910 | 34 | r029 | r029 | 59 | Update/r440 | r440 | 84 | Update/r055 | r055 |
| 10 | Update/r054 | r054 | 35 | Update/r921 | r921 | 60 | Update/r070 | r070 | 85 | Update/r664 | r664 |
| 11 | r098 | r098 | 36 | Update/r051 | r051 | 61 | Update/r273 | r273 | 86 | Update/r515 | r515 |
| 12 | Update2/r327 | r327 | 37 | Update/r655 | r655 | 62 | Update/r669 | r669 | 87 | Update2/r017 | r017 |
| 13 | Update/r657 | r657 | 38 | Update/r823 | r823 | 63 | r084 | r084 | 88 | Update/r001 | r001 |
| 14 | r083 | r083 | 39 | Update2/r981 | r981 | 64 | Update/r568 | r568 | 89 | Update/r203 | r203 |
| 15 | Update/r808 | r808 | 40 | r082 | r082 | 65 | Update/r037 | r037 | 90 | Update2/r940 | r940 |
| 16 | r090 | r090 | 41 | Update/r328 | r328 | 66 | Update/r276 | r276 | 91 | Update/r034 | r034 |
| 17 | Update/r982 | r982 | 42 | r027 | r027 | 67 | Update/r431 | r431 | 92 | Update/r095 | r095 |
| 18 | r076 | r076 | 43 | Update/r522 | r522 | 68 | Update/r634 | r634 | 93 | Update/r041 | r041 |
| 19 | Update/r010 | r010 | 44 | Update/r506 | r506 | 69 | Update/r652 | r652 | 94 | Update/r260 | r260 |
| 20 | Update/r059 | r059 | 45 | Update2/r207 | r207 | 70 | Update/r752 | r752 | 95 | Update2/r013 | r013 |
| 21 | Update/r006 | r006 | 46 | r073 | r073 | 71 | Update/r064 | r064 | 96 | Update/r009 | r009 |
| 22 | Update/r640 | r640 | 47 | Update/r707 | r707 | 72 | Update/r032 | r032 | 97 | Update2/r489 | r489 |
| 23 | r097 | r097 | 48 | Update/r814 | r814 | 73 | r004 | r004 | 98 | Update/r727 | r727 |
| 24 | Update/r922 | r922 | 49 | Update/r508 | r508 | 74 | Update/r058 | r058 | 99 | Update/r806 | r806 |
| 25 | Update/r631 | r631 | 50 | Update/r092 | r092 | 75 | r074 | r074 | 100 | r024 | r024 |

A1.3 Step 2 – Enlarge Mining Areas

The main Bash shell script `run_mining_mods.sh` (line 62 §A4.3) calls the Python script `enlarge_mining_areas.py` (§A4.4 and Table A1-7), which uses the input file `mining_areas_xyz.dat` as the definition of the mined areas (output by `import_polyline_determine_mined_areas.m`, §A4.2), and the UTM NAD27 Zone 13 coordinates of the corners of the WIPP LWB (specified in lines 4 and 5 of the input file `enlarge_mining_areas.in` §A4.4.1). The script writes two arrays as output, indicating where mining effects will be applied in both the full (`New_Full_09.dat`) and partial (`New_Part_09.dat`) mining scenarios.

Table A1-7. Input and output file for Python script `enlarge_mining_areas.py`

| File ¹ | Remark |
|--------------------------------|---|
| Inputs | |
| Inputs/mining_areas_xyz.dat | X,Y coordinates of 100x100-m cells with index indicating presence of minable potash |
| Inputs/enlarge_mining_areas.in | Python script input with filenames, model grid, and WIPP LWB coordinates |
| Outputs | |
| Outputs/mining_factor.dat | matrix form of input indices for plotting |
| Outputs/mining_factor.datx | x-coordinates for matrix form of plotting |
| Outputs/mining_factor.daty | y-coordinates for matrix form of plotting |
| Outputs/New_Full_09.dat | full mining index as a matrix |
| Outputs/New_Part_09.dat | partial mining index as a matrix |

1. CVS repository MiningMod

In the Python mining area enlargement script (§A4.4.2), the grid-defined mining areas are read in as (x,y,index) triplets (lines 118-131), converted from the indices 0 (no mining) and 1 (mining) to 0 and 10 (line 127), then the index associated with each model cell center that is located inside the WIPP LWB is increased by 1 (see script lines 135-143 and Table A1-8).

Table A1-8. Mining indices used in `expand_mining_areas.py`

| | Mined area | Non-mined area |
|------------------|------------|----------------|
| Inside WIPP LWB | 11 | 1 |
| Outside WIPP LWB | 10 | 0 |

Mining-related subsidence effects are expanded by cycling through the original array of model cells, and propagating effects from this into a new matrix using the stencil comprised of a 5x5 square of cells centered on the mined cell, with 1 additional affected cell located along the 4 compass directions (see script lines 168-204 and Figure 3-5).

Inside the WIPP LWB, MODFLOW cells with minable potash are compared to both mining-affected cells and the potash shapefile in Figure 3-12 through Figure 3-14 in Section 3.4.2 of the main text.

Table A1-9. Excerpts from mining_areas_matrix.dat (top), New_Full_09.dat (middle), and New_Part_09.dat (bottom) for rows 101-129 and columns 71-121 (area including NW corner of WIPP LWB). 0 is not mined or affected, 1 is mined or affected by mining (see Figure A1-5 for location of area).

The table consists of three vertically stacked sections, each representing a different data source: mining_areas_matrix.dat (top), New_Full_09.dat (middle), and New_Part_09.dat (bottom). Each section contains a grid of 51 rows (representing rows 101-151) and 51 columns (representing columns 71-121). The cells in the grid are filled with either a '0' or a '1', indicating whether a specific area is mined or affected by mining. The top section (mining_areas_matrix.dat) shows a sparse distribution of '1's, primarily in the upper-left and lower-right quadrants. The middle section (New_Full_09.dat) shows a more dense and irregular pattern of '1's, with a significant concentration in the lower-left quadrant. The bottom section (New_Part_09.dat) shows a pattern of '1's that is similar to the middle section but with some differences in density and distribution.

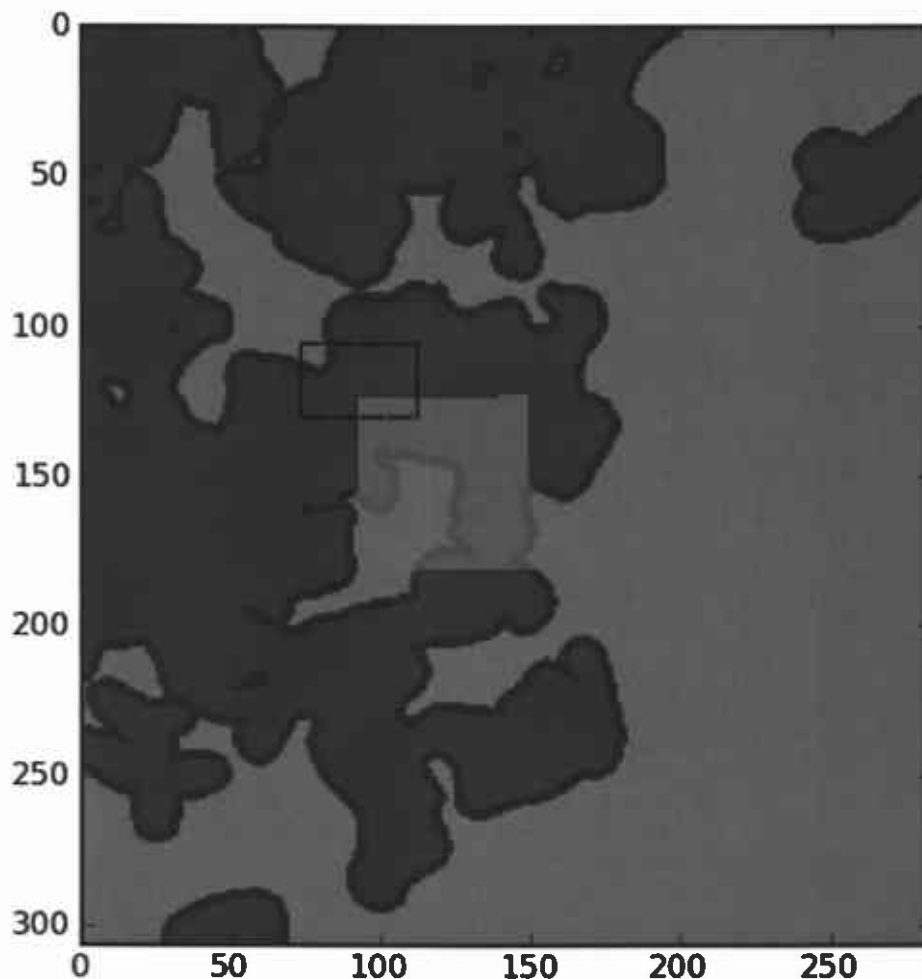


Figure A1-5. Matrices of full, partial, and unexpanded mining factors (region corresponding to Table A1-9 marked with dashed line); maroon cells have minable potash in both mining scenarios, purple indicates the halo of cells affected by mining that are not directly mined, orange cells are only mined in the full-mining scenario (with the corresponding halo of affected full-mining only cells in dark gray), and the background of model cells are unaffected by mining.

A1.4 Step 3 – Perform Mining Modifications

The main Bash shell script `run_mining_mods.sh` (§A4.3) at line 67 calls the Python script `mining_mods.py` (§A4.5.2 and Table A1-10), which uses the definitions of the mining-affected areas from the previous step and the mining factors from VMS (see section A2.5, `sum1_st2d_pabc09_r{1,2,3}.tbl`) to modify the hydraulic conductivity arrays to account for the potential increases in hydraulic conductivity due to mining-related subsidence.

After reading in the input files (lines 113-116 §A4.5.2), the Python script `mining_mods.py` first modifies the boundary conditions in certain mined areas (lines 123-144). Model cells that are specified head and mined (excluding the outermost ring of constant-head cells – handled in lines 146-165) are changed to normal cells, resulting in two new MODFLOW IBOUND arrays, `init_bnds_full.inf` and `init_bnds_part.inf` (lines 167-176).

Lastly, the horizontal hydraulic conductivity is modified by the mining factor (read in lines 186-202), resulting in six new hydraulic conductivity fields for each realization (3 replicates with different mining factors \times 2 cases of either partial or full mining, see lines 209-252).

A1.4.1 Modification of Boundary Conditions

The MODFLOW IBOUND array is used to indicate whether cells are active (1), inactive (0), or specified head (negative integers) (Harbaugh et al., 2000). East of the composite M2/H2 + M3/H3 halite margin there exist essentially lithostatic pressure conditions, where pressure is equal to the weight of the overburden. The MODFLOW model developed in AP-114 Tasks 5 and 7 represents this lithostatic area using both very low hydraulic conductivity and specified head set to the land surface elevation (Hart et al., 2008; 2009). Applying the mining factor to these areas without changing the cell from constant head to active would greatly increase the flux flowing into the domain, since the constant-head boundary conditions are a potentially infinite source of water. Since the halite-sandwiched region is not believed to be a large source of water, constant-head cells that are

1. interior to the modeling domain (i.e., not in rows 1 or 307 or in columns 1 or 284),
2. east of the H2/M2 or H3/M3 Rustler halite margins (i.e., in the “halite-sandwiched” region), and
3. affected by mining modifications

are converted to active cells (see individual IBOUND arrays in Figure A1-6 and a composite figure of all three arrays in Figure A1-7).

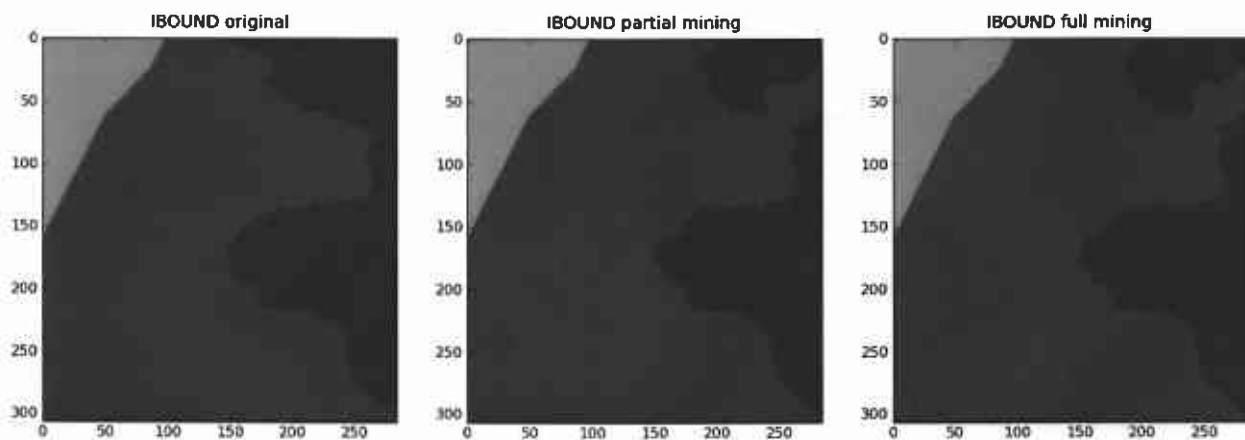


Figure A1-6. MODFLOW IBOUND arrays for original, partial-mining, and full-mining cases. Green is inactive or no-flow (zero), brick red is active (one), blue is specified head (negative integers). The only difference between full- and partial-mining scenarios is near row 175, column 150.

The cells in the outermost row and columns of the domain (aside from those cells in the inactive or no-flow region) are maintained as constant head, even if they correspond to mined regions – only constant-head cells in the interior of the model domain corresponding to the halite-sandwiched region of the Culebra are changed to active cells.

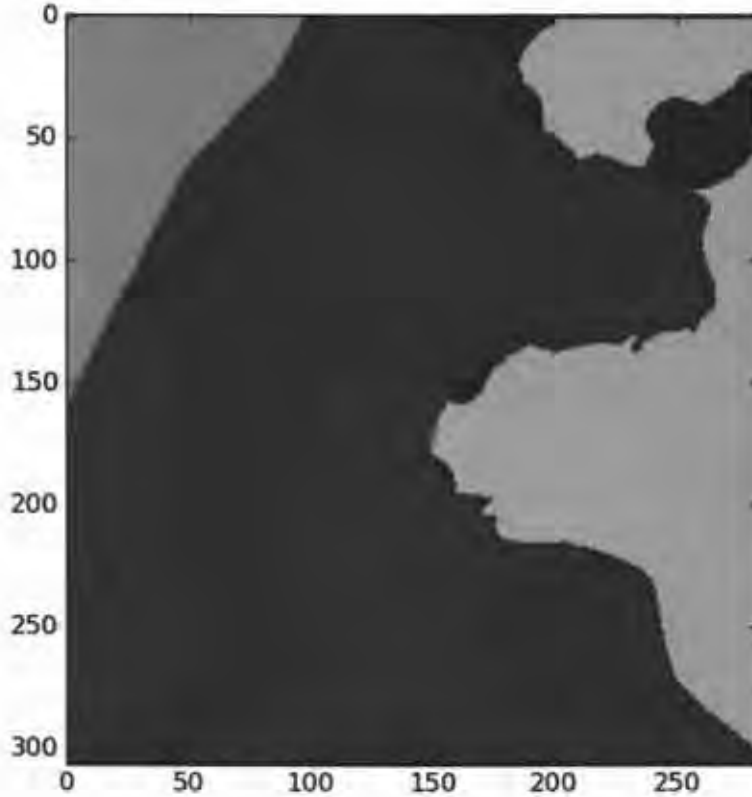


Figure A1-7. Overlay of three original, partial-mining, and full-mining IBOUND arrays given separately in Figure A1-6. The only difference between full and partial mining is a small tan sliver near row 175, column 150.

Table A1-10. Input and output files for Python script `mining_mod.py`

| File ^{1,2} | Remark |
|--|--|
| Inputs | |
| Inputs/keepers_short | list of 100 realizations with "Update" or "Update2" stripped off (Table A1-6). |
| Inputs/mining_mod.in | Python script input with filenames |
| Outputs/New_Full_09.dat | full mining index (Table A1-7 outputs) |
| Outputs/New_Part_09.dat | partial mining index (Table A1-7 outputs) |
| Inputs/AP114_Task7/data/init_bnds.inf ³ | original MF2K IBOUND array |
| Inputs/sum1_st2d_pabc09_r{1,2,3}.tbl | mining factors from VMS; replicates 1,2,3 |
| Inputs/AP114_Task7/Outputs/r??*/modeled_K_field.mod ³ | 100 calibrated T-fields from AP-114 Task 7 |
| Outputs | |
| Outputs/data/init_bnds_full.inf | MF2K IBOUND array for full mining |
| Outputs/data/init_bnds_part.inf | MF2K IBOUND array for partial mining |
| Outputs/Rn/M/r??*/modeled_mined_K_field.mod | 600 MF2K input mining-modified T-fields |

1. CVS repository MiningMod
2. $n \in \{1,2,3\}$; $M \in \{\text{full,part}\}$; $r??? \in \{r256, r902, \text{etc.}\}$ (see Table A1-6)
3. Inputs/AP114_Task7/ is the working directory where files checked out from CVS repository Tfields are located, see (Long, 2010)

A1.4.2 Modification of Transmissivity Fields

The mining factors are saved in three table files (sum1_st2d_pabc09_r{1,2,3}.tbl); the data in these tables, along with the names of the MODFLOW 2000 T-field realizations (MF2K RZN – taken from the file keepers) are given in two different ways in Table A1-11 and Table A1-12. Table A1-11 is presented from the viewpoint of the 100 MODFLOW flow-fields and the mining factors that were applied to them for the three replicates. Table A1-12 is presented from the viewpoint of the SECOTP2D vectors, and which MODFLOW flow-field and mining factor was used in each replicate.

Each T-field is modified inside the potash-affected areas uniformly by a mining factor between 1 and 1,000 (MINP_FACT) applied to a selected realization (the integer TRANS_IDX), both variables being sampled by LHS (Kirchner, 2010) for each replicate.

Table A1-11. Mining factors applied to each MODFLOW realization. Keepers id refers to the order the realizations are listed in the keepers file (Table A1-6). First five columns are sorted by MODFLOW realization name, last five columns sorted by order in keepers file.

| MF2K rzn | keepers id | R1 | R2 | R3 | keepers id | MF2K rzn | R1 | R2 | R3 |
|-------------|---------------|-------|-------|-------|---------------|-------------|-------|-------|-------|
| r001 | 88 | 374.4 | 487.7 | 774.5 | 1 | r256 | 552.2 | 271.0 | 829.2 |
| r002 | 76 | 766.0 | 252.9 | 72.50 | 2 | r902 | 444.3 | 778.6 | 439.4 |
| r004 | 73 | 356.6 | 249.1 | 992.4 | 3 | r038 | 149.2 | 124.6 | 795.4 |
| r006 | 21 | 847.3 | 756.8 | 737.6 | 4 | r486 | 484.9 | 515.1 | 38.70 |
| r007 | 31 | 84.52 | 432.4 | 592.5 | 5 | r984 | 947.2 | 237.0 | 163.0 |
| r009 | 96 | 384.8 | 665.2 | 867.6 | 6 | r028 | 104.2 | 228.6 | 391.3 |
| r010 | 19 | 240.1 | 979.7 | 59.88 | 7 | r298 | 170.5 | 690.7 | 802 |
| r012 | 27 | 774.2 | 143.7 | 875.2 | 8 | r791 | 821.7 | 170.2 | 208.8 |
| r013 | 95 | 783.4 | 927.2 | 310.9 | 9 | r910 | 124.5 | 503.6 | 67.93 |
| r017 | 87 | 593.3 | 614.0 | 347.2 | 10 | r054 | 217.6 | 402.0 | 976.7 |
| r024 | 100 | 718.3 | 980.8 | 377.4 | 11 | r098 | 681.8 | 948.3 | 112.6 |
| r027 | 42 | 579.4 | 633.7 | 17.75 | 12 | r327 | 39.37 | 357.3 | 452.9 |
| r028 | 6 | 104.2 | 228.6 | 391.3 | 13 | r657 | 810.8 | 86.49 | 886.4 |
| r029 | 34 | 968.6 | 322.9 | 707.1 | 14 | r083 | 857.1 | 479.9 | 716.7 |
| r032 | 72 | 637.4 | 291.5 | 191.8 | 15 | r808 | 796.7 | 392.2 | 423.0 |
| r034 | 91 | 80.05 | 62.14 | 937.3 | 16 | r090 | 340.7 | 425.9 | 548.9 |
| r037 | 65 | 566.1 | 674.3 | 416.3 | 17 | r982 | 869.8 | 373.0 | 530.2 |
| r038 | 3 | 149.2 | 124.6 | 795.4 | 18 | r076 | 435.1 | 36.28 | 851.6 |
| r040 | 57 | 337.4 | 996.2 | 143.2 | 19 | r010 | 240.1 | 979.7 | 59.88 |
| r041 | 93 | 531.6 | 207.4 | 813.2 | 20 | r059 | 937.6 | 831.2 | 492.3 |
| r045 | 52 | 732.0 | 415.2 | 354.7 | 21 | r006 | 847.3 | 756.8 | 737.6 |
| r051 | 36 | 803.3 | 451.5 | 837.5 | 22 | r640 | 751.6 | 599.5 | 766.5 |
| r052 | 30 | 583.5 | 539.0 | 153.5 | 23 | r097 | 898.1 | 898.8 | 126.1 |
| r053 | 77 | 721.0 | 657.3 | 221.1 | 24 | r922 | 289.7 | 729.1 | 676.3 |
| r054 | 10 | 217.6 | 402.0 | 976.7 | 25 | r631 | 616.3 | 713.6 | 918.5 |
| r055 | 84 | 186.2 | 904.5 | 28.11 | 26 | r060 | 58.89 | 876.3 | 383.6 |
| r058 | 74 | 956.1 | 780.6 | 587.5 | 27 | r012 | 774.2 | 143.7 | 875.2 |
| r059 | 20 | 937.6 | 831.2 | 492.3 | 28 | r078 | 920.7 | 847.9 | 602.0 |
| r060 | 26 | 58.89 | 876.3 | 383.6 | 29 | r104 | 309.0 | 888.1 | 842.8 |
| r061 | 83 | 298.5 | 16.25 | 648.9 | 30 | r052 | 583.5 | 539.0 | 153.5 |
| r064 | 71 | 399.4 | 103.4 | 189.2 | 31 | r007 | 84.52 | 432.4 | 592.5 |
| r070 | 60 | 362.8 | 705.3 | 562.3 | 32 | r102 | 30.83 | 495.7 | 274.8 |
| r073 | 46 | 983.4 | 582.3 | 505.1 | 33 | r809 | 13.61 | 962.8 | 252.9 |
| r074 | 75 | 91.84 | 936.0 | 728.1 | 34 | r029 | 968.6 | 322.9 | 707.1 |
| r076 | 18 | 435.1 | 36.28 | 851.6 | 35 | r921 | 258.5 | 805.7 | 107.8 |
| r078 | 28 | 920.7 | 847.9 | 602.0 | 36 | r051 | 803.3 | 451.5 | 837.5 |
| r082 | 40 | 325.0 | 827.2 | 471.1 | 37 | r655 | 499.1 | 118.6 | 983.0 |
| r083 | 14 | 857.1 | 479.9 | 716.7 | 38 | r823 | 230.7 | 910.9 | 281.4 |
| r084 | 63 | 320.7 | 605.5 | 488.0 | 39 | r981 | 111.8 | 527.7 | 172.3 |
| r090 | 16 | 340.7 | 425.9 | 548.9 | 40 | r082 | 325.0 | 827.2 | 471.1 |
| r092 | 50 | 413.8 | 49.06 | 689.4 | 41 | r328 | 696.7 | 194.6 | 757.3 |

| MF2K rzn | keepers id | R1 | R2 | R3 | keepers id | MF2K rzn | R1 | R2 | R3 |
|-------------|---------------|-------|-------|-------|---------------|-------------|-------|-------|-------|
| r095 | 92 | 203.6 | 737.1 | 326.2 | 42 | r027 | 579.4 | 633.7 | 17.75 |
| r097 | 23 | 898.1 | 898.8 | 126.1 | 43 | r522 | 404.7 | 682.8 | 139.2 |
| r098 | 11 | 681.8 | 948.3 | 112.6 | 44 | r506 | 505.9 | 769.6 | 460.9 |
| r102 | 32 | 30.83 | 495.7 | 274.8 | 45 | r207 | 671.8 | 307.5 | 298.8 |
| r104 | 29 | 309.0 | 888.1 | 842.8 | 46 | r073 | 983.4 | 582.3 | 505.1 |
| r137 | 53 | 261.0 | 442.7 | 331.8 | 47 | r707 | 67.09 | 286.8 | 638.7 |
| r142 | 82 | 659.5 | 569.5 | 558.1 | 48 | r814 | 832.1 | 383.8 | 699.4 |
| r191 | 81 | 744.2 | 183.5 | 954.3 | 49 | r508 | 478.8 | 811.7 | 656.5 |
| r203 | 89 | 648.4 | 91.27 | 401.7 | 50 | r092 | 413.8 | 49.06 | 689.4 |
| r207 | 45 | 671.8 | 307.5 | 298.8 | 51 | r883 | 705.1 | 748.4 | 921.9 |
| r256 | 1 | 552.2 | 271.0 | 829.2 | 52 | r045 | 732.0 | 415.2 | 354.7 |
| r260 | 94 | 888.6 | 958.4 | 516.5 | 53 | r137 | 261.0 | 442.7 | 331.8 |
| r273 | 61 | 872.8 | 261.7 | 364.2 | 54 | r279 | 995.3 | 620.6 | 960.4 |
| r276 | 66 | 137.4 | 797.5 | 890.6 | 55 | r694 | 514.0 | 552.0 | 574.7 |
| r279 | 54 | 995.3 | 620.6 | 960.4 | 56 | r361 | 455.0 | 5.562 | 946.9 |
| r298 | 7 | 170.5 | 690.7 | 802.0 | 57 | r040 | 337.4 | 996.2 | 143.2 |
| r327 | 12 | 39.37 | 357.3 | 452.9 | 58 | r571 | 603.1 | 59.7 | 444.3 |
| r328 | 41 | 696.7 | 194.6 | 757.3 | 59 | r440 | 663.4 | 77.07 | 902.6 |
| r361 | 56 | 455.0 | 5.562 | 946.9 | 60 | r070 | 362.8 | 705.3 | 562.3 |
| r431 | 67 | 912.6 | 541.0 | 237.4 | 61 | r273 | 872.8 | 261.7 | 364.2 |
| r440 | 59 | 663.4 | 77.07 | 902.6 | 62 | r669 | 1.019 | 572.6 | 747.9 |
| r465 | 79 | 908.9 | 24.43 | 615.9 | 63 | r084 | 320.7 | 605.5 | 488.0 |
| r486 | 4 | 484.9 | 515.1 | 38.70 | 64 | r568 | 977.1 | 344.8 | 86.60 |
| r489 | 97 | 529.8 | 136.2 | 668.9 | 65 | r037 | 566.1 | 674.3 | 416.3 |
| r506 | 44 | 505.9 | 769.6 | 460.9 | 66 | r276 | 137.4 | 797.5 | 890.6 |
| r508 | 49 | 478.8 | 811.7 | 656.5 | 67 | r431 | 912.6 | 541.0 | 237.4 |
| r511 | 80 | 625.9 | 851.8 | 266.1 | 68 | r634 | 153.5 | 469.3 | 250.2 |
| r515 | 86 | 466.9 | 865.4 | 302.1 | 69 | r652 | 179.5 | 318.3 | 4.342 |
| r522 | 43 | 404.7 | 682.8 | 139.2 | 70 | r752 | 247.5 | 338.3 | 214.2 |
| r568 | 64 | 977.1 | 344.8 | 86.60 | 71 | r064 | 399.4 | 103.4 | 189.2 |
| r571 | 58 | 603.1 | 59.70 | 444.3 | 72 | r032 | 637.4 | 291.5 | 191.8 |
| r631 | 25 | 616.3 | 713.6 | 918.5 | 73 | r004 | 356.6 | 249.1 | 992.4 |
| r634 | 68 | 153.5 | 469.3 | 250.2 | 74 | r058 | 956.1 | 780.6 | 587.5 |
| r640 | 22 | 751.6 | 599.5 | 766.5 | 75 | r074 | 91.84 | 936.0 | 728.1 |
| r652 | 69 | 179.5 | 318.3 | 4.342 | 76 | r002 | 766.0 | 252.9 | 72.50 |
| r655 | 37 | 499.1 | 118.6 | 983.0 | 77 | r053 | 721.0 | 657.3 | 221.1 |
| r657 | 13 | 810.8 | 86.49 | 886.4 | 78 | r861 | 44.18 | 156.2 | 91.25 |
| r664 | 85 | 544.4 | 220.1 | 787.5 | 79 | r465 | 908.9 | 24.43 | 615.9 |
| r669 | 62 | 1.019 | 572.6 | 747.9 | 80 | r511 | 625.9 | 851.8 | 266.1 |
| r694 | 55 | 514.0 | 552.0 | 574.7 | 81 | r191 | 744.2 | 183.5 | 954.3 |
| r707 | 47 | 67.09 | 286.8 | 638.7 | 82 | r142 | 659.5 | 569.5 | 558.1 |
| r727 | 98 | 424.7 | 648.6 | 42.97 | 83 | r061 | 298.5 | 16.25 | 648.9 |
| r752 | 70 | 247.5 | 338.3 | 214.2 | 84 | r055 | 186.2 | 904.5 | 28.11 |
| r791 | 8 | 821.7 | 170.2 | 208.8 | 85 | r664 | 544.4 | 220.1 | 787.5 |
| r806 | 99 | 272.8 | 179.3 | 627.4 | 86 | r515 | 466.9 | 865.4 | 302.1 |
| r808 | 15 | 796.7 | 392.2 | 423.0 | 87 | r017 | 593.3 | 614.0 | 347.2 |
| r809 | 33 | 13.61 | 962.8 | 252.9 | 88 | r001 | 374.4 | 487.7 | 774.5 |
| r814 | 48 | 832.1 | 383.8 | 699.4 | 89 | r203 | 648.4 | 91.27 | 401.7 |
| r823 | 38 | 230.7 | 910.9 | 281.4 | 90 | r940 | 196.3 | 365.2 | 534.8 |
| r861 | 78 | 44.18 | 156.2 | 91.25 | 91 | r034 | 80.05 | 62.14 | 937.3 |
| r883 | 51 | 705.1 | 748.4 | 921.9 | 92 | r095 | 203.6 | 737.1 | 326.2 |
| r902 | 2 | 444.3 | 778.6 | 439.4 | 93 | r041 | 531.6 | 207.4 | 813.2 |
| r910 | 9 | 124.5 | 503.6 | 67.93 | 94 | r260 | 888.6 | 958.4 | 516.5 |
| r921 | 35 | 258.5 | 805.7 | 107.8 | 95 | r013 | 783.4 | 927.2 | 310.9 |
| r922 | 24 | 289.7 | 729.1 | 676.3 | 96 | r009 | 384.8 | 665.2 | 867.6 |
| r940 | 90 | 196.3 | 365.2 | 534.8 | 97 | r489 | 529.8 | 136.2 | 668.9 |
| r981 | 39 | 111.8 | 527.7 | 172.3 | 98 | r727 | 424.7 | 648.6 | 42.97 |
| r982 | 17 | 869.8 | 373.0 | 530.2 | 99 | r806 | 272.8 | 179.3 | 627.4 |
| r984 | 5 | 947.2 | 237.0 | 163.0 | 100 | r024 | 718.3 | 980.8 | 377.4 |

Table A1-12. Mining factors and MODFLOW flow simulation corresponding to each SECOTP2D vector and replicate

| vector | R1 | | | R2 | | | R3 | | |
|--------|--------------|-------------|--------------|--------------|-------------|--------------|--------------|-------------|--------------|
| | TRANS IDX | MF2K RZN | MINP FACT | TRANS IDX | MF2K RZN | MINP FACT | TRANS IDX | MF2K RZN | MINP FACT |
| 1 | 46 | r073 | 983.4 | 73 | r004 | 249.1 | 47 | r707 | 638.7 |
| 2 | 79 | r465 | 908.9 | 86 | r515 | 865.4 | 75 | r074 | 728.1 |
| 3 | 93 | r041 | 531.6 | 91 | r034 | 62.14 | 70 | r752 | 214.2 |
| 4 | 30 | r052 | 583.5 | 75 | r074 | 936.0 | 23 | r097 | 126.1 |
| 5 | 88 | r001 | 374.4 | 24 | r922 | 729.1 | 87 | r017 | 347.2 |
| 6 | 36 | r051 | 803.3 | 41 | r328 | 194.6 | 51 | r883 | 921.9 |
| 7 | 96 | r009 | 384.8 | 77 | r053 | 657.3 | 50 | r092 | 689.4 |
| 8 | 81 | r191 | 744.2 | 69 | r652 | 318.3 | 53 | r137 | 331.8 |
| 9 | 95 | r013 | 783.4 | 36 | r051 | 451.5 | 94 | r260 | 516.5 |
| 10 | 51 | r883 | 705.1 | 1 | r256 | 271.0 | 16 | r090 | 548.9 |
| 11 | 42 | r027 | 579.4 | 80 | r511 | 851.8 | 30 | r052 | 153.5 |
| 12 | 85 | r664 | 544.4 | 70 | r752 | 338.3 | 78 | r861 | 91.25 |
| 13 | 39 | r981 | 111.8 | 66 | r276 | 797.5 | 81 | r191 | 954.3 |
| 14 | 26 | r060 | 58.89 | 10 | r054 | 402.0 | 10 | r054 | 976.7 |
| 15 | 94 | r260 | 888.6 | 55 | r694 | 552.0 | 49 | r508 | 656.5 |
| 16 | 83 | r061 | 298.5 | 38 | r823 | 910.9 | 88 | r001 | 774.5 |
| 17 | 78 | r861 | 44.18 | 79 | r465 | 24.43 | 97 | r489 | 668.9 |
| 18 | 19 | r010 | 240.1 | 100 | r024 | 980.8 | 96 | r009 | 867.6 |
| 19 | 69 | r652 | 179.5 | 87 | r017 | 614.0 | 3 | r038 | 795.4 |
| 20 | 32 | r102 | 30.83 | 68 | r634 | 469.3 | 7 | r298 | 802.0 |
| 21 | 62 | r669 | 1.019 | 71 | r064 | 103.4 | 33 | r809 | 252.9 |
| 22 | 16 | r090 | 340.7 | 4 | r486 | 515.1 | 64 | r568 | 86.60 |
| 23 | 29 | r104 | 309.0 | 65 | r037 | 674.3 | 31 | r007 | 592.5 |
| 24 | 77 | r053 | 721.0 | 2 | r902 | 778.6 | 76 | r002 | 72.50 |
| 25 | 3 | r038 | 149.2 | 50 | r092 | 49.06 | 82 | r142 | 558.1 |
| 26 | 20 | r059 | 937.6 | 13 | r657 | 86.49 | 44 | r506 | 460.9 |
| 27 | 68 | r634 | 153.5 | 57 | r040 | 996.2 | 1 | r256 | 829.2 |
| 28 | 1 | r256 | 552.2 | 14 | r083 | 479.9 | 59 | r440 | 902.6 |
| 29 | 18 | r076 | 435.1 | 22 | r640 | 599.5 | 46 | r073 | 505.1 |
| 30 | 27 | r012 | 774.2 | 93 | r041 | 207.4 | 20 | r059 | 492.3 |
| 31 | 97 | r489 | 529.8 | 18 | r076 | 36.28 | 79 | r465 | 615.9 |
| 32 | 92 | r095 | 203.6 | 61 | r273 | 261.7 | 61 | r273 | 364.2 |
| 33 | 76 | r002 | 766.0 | 56 | r361 | 5.562 | 83 | r061 | 648.9 |
| 34 | 98 | r727 | 424.7 | 45 | r207 | 307.5 | 9 | r910 | 67.93 |
| 35 | 13 | r657 | 810.8 | 43 | r522 | 682.8 | 69 | r652 | 4.342 |
| 36 | 70 | r752 | 247.5 | 96 | r009 | 665.2 | 42 | r027 | 17.75 |
| 37 | 22 | r640 | 751.6 | 60 | r070 | 705.3 | 55 | r694 | 574.7 |
| 38 | 60 | r070 | 362.8 | 52 | r045 | 415.2 | 48 | r814 | 699.4 |
| 39 | 7 | r298 | 170.5 | 20 | r059 | 831.2 | 34 | r029 | 707.1 |
| 40 | 4 | r486 | 484.9 | 23 | r097 | 898.8 | 41 | r328 | 757.3 |
| 41 | 17 | r982 | 869.8 | 92 | r095 | 737.1 | 85 | r664 | 787.5 |
| 42 | 58 | r571 | 603.1 | 59 | r440 | 77.07 | 74 | r058 | 587.5 |
| 43 | 64 | r568 | 977.1 | 85 | r664 | 220.1 | 100 | r024 | 377.4 |
| 44 | 14 | r083 | 857.1 | 15 | r808 | 392.2 | 45 | r207 | 298.8 |
| 45 | 15 | r808 | 796.7 | 76 | r002 | 252.9 | 80 | r511 | 266.1 |
| 46 | 25 | r631 | 616.3 | 48 | r814 | 383.8 | 2 | r902 | 439.4 |
| 47 | 74 | r058 | 956.1 | 21 | r006 | 756.8 | 98 | r727 | 42.97 |
| 48 | 99 | r806 | 272.8 | 51 | r883 | 748.4 | 72 | r032 | 191.8 |
| 49 | 90 | r940 | 196.3 | 3 | r038 | 124.6 | 25 | r631 | 918.5 |
| 50 | 71 | r064 | 399.4 | 37 | r655 | 118.6 | 38 | r823 | 281.4 |
| 51 | 35 | r921 | 258.5 | 7 | r298 | 690.7 | 58 | r571 | 444.3 |
| 52 | 48 | r814 | 832.1 | 44 | r506 | 769.6 | 36 | r051 | 837.5 |
| 53 | 53 | r137 | 261.0 | 49 | r508 | 811.7 | 86 | r515 | 302.1 |
| 54 | 75 | r074 | 91.84 | 19 | r010 | 979.7 | 73 | r004 | 992.4 |
| 55 | 41 | r328 | 696.7 | 29 | r104 | 888.1 | 84 | r055 | 28.11 |
| 56 | 44 | r506 | 505.9 | 81 | r191 | 183.5 | 6 | r028 | 391.3 |
| 57 | 10 | r054 | 217.6 | 28 | r078 | 847.9 | 39 | r981 | 172.3 |
| 58 | 24 | r922 | 289.7 | 27 | r012 | 143.7 | 99 | r806 | 627.4 |
| 59 | 11 | r098 | 681.8 | 63 | r084 | 605.5 | 22 | r640 | 766.5 |
| 60 | 56 | r361 | 455.0 | 42 | r027 | 633.7 | 28 | r078 | 602.0 |
| 61 | 55 | r694 | 514.0 | 39 | r981 | 527.7 | 89 | r203 | 401.7 |

| vector | R1 | | | R2 | | | R3 | | |
|--------|--------------|-------------|--------------|--------------|-------------|--------------|--------------|-------------|--------------|
| | TRANS IDX | MF2K RZN | MINP FACT | TRANS IDX | MF2K RZN | MINP FACT | TRANS IDX | MF2K RZN | MINP FACT |
| 62 | 63 | r084 | 320.7 | 12 | r327 | 357.3 | 40 | r082 | 471.1 |
| 63 | 57 | r040 | 337.4 | 98 | r727 | 648.6 | 92 | r095 | 326.2 |
| 64 | 84 | r055 | 186.2 | 99 | r806 | 179.3 | 52 | r045 | 354.7 |
| 65 | 86 | r515 | 466.9 | 25 | r631 | 713.6 | 57 | r040 | 143.2 |
| 66 | 23 | r097 | 898.1 | 16 | r090 | 425.9 | 21 | r006 | 737.6 |
| 67 | 72 | r032 | 637.4 | 11 | r098 | 948.3 | 27 | r012 | 875.2 |
| 68 | 5 | r984 | 947.2 | 30 | r052 | 539.0 | 37 | r655 | 983.0 |
| 69 | 28 | r078 | 920.7 | 31 | r007 | 432.4 | 13 | r657 | 886.4 |
| 70 | 73 | r004 | 356.6 | 94 | r260 | 958.4 | 90 | r940 | 534.8 |
| 71 | 33 | r809 | 13.61 | 17 | r982 | 373.0 | 91 | r034 | 937.3 |
| 72 | 87 | r017 | 593.3 | 58 | r571 | 59.70 | 43 | r522 | 139.2 |
| 73 | 34 | r029 | 968.6 | 40 | r082 | 827.2 | 60 | r070 | 562.3 |
| 74 | 40 | r082 | 325.0 | 8 | r791 | 170.2 | 8 | r791 | 208.8 |
| 75 | 21 | r006 | 847.3 | 62 | r669 | 572.6 | 77 | r053 | 221.1 |
| 76 | 80 | r511 | 625.9 | 53 | r137 | 442.7 | 24 | r922 | 676.3 |
| 77 | 49 | r508 | 478.8 | 74 | r058 | 780.6 | 17 | r982 | 530.2 |
| 78 | 67 | r431 | 912.6 | 47 | r707 | 286.8 | 54 | r279 | 960.4 |
| 79 | 9 | r910 | 124.5 | 64 | r568 | 344.8 | 4 | r486 | 38.70 |
| 80 | 91 | r034 | 80.05 | 9 | r910 | 503.6 | 95 | r013 | 310.9 |
| 81 | 50 | r092 | 413.8 | 32 | r102 | 495.7 | 68 | r634 | 250.2 |
| 82 | 6 | r028 | 104.2 | 54 | r279 | 620.6 | 65 | r037 | 416.3 |
| 83 | 59 | r440 | 663.4 | 82 | r142 | 569.5 | 14 | r083 | 716.7 |
| 84 | 82 | r142 | 659.5 | 89 | r203 | 91.27 | 66 | r276 | 890.6 |
| 85 | 37 | r655 | 499.1 | 6 | r028 | 228.6 | 26 | r060 | 383.6 |
| 86 | 61 | r273 | 872.8 | 35 | r921 | 805.7 | 12 | r327 | 452.9 |
| 87 | 31 | r007 | 84.52 | 97 | r489 | 136.2 | 5 | r984 | 163.0 |
| 88 | 12 | r327 | 39.37 | 83 | r061 | 16.25 | 18 | r076 | 851.6 |
| 89 | 65 | r037 | 566.1 | 67 | r431 | 541.0 | 71 | r064 | 189.2 |
| 90 | 38 | r823 | 230.7 | 46 | r073 | 582.3 | 19 | r010 | 59.88 |
| 91 | 8 | r791 | 821.7 | 78 | r861 | 156.2 | 63 | r084 | 488.0 |
| 92 | 89 | r203 | 648.4 | 84 | r055 | 904.5 | 15 | r808 | 423.0 |
| 93 | 2 | r902 | 444.3 | 33 | r809 | 962.8 | 67 | r431 | 237.4 |
| 94 | 100 | r024 | 718.3 | 34 | r029 | 322.9 | 11 | r098 | 112.6 |
| 95 | 54 | r279 | 995.3 | 95 | r013 | 927.2 | 32 | r102 | 274.8 |
| 96 | 47 | r707 | 67.09 | 88 | r001 | 487.7 | 93 | r041 | 813.2 |
| 97 | 45 | r207 | 671.8 | 72 | r032 | 291.5 | 35 | r921 | 107.8 |
| 98 | 52 | r045 | 732.0 | 5 | r984 | 237.0 | 56 | r361 | 946.9 |
| 99 | 43 | r522 | 404.7 | 90 | r940 | 365.2 | 29 | r104 | 842.8 |
| 100 | 66 | r276 | 137.4 | 26 | r060 | 876.3 | 62 | r669 | 747.9 |

Figure A1-8 and Figure A1-9 show examples of the changes in the T-field due to the mining modifications for a single realization (r440). In Figure A1-8, the color scale is the same between all three plots, and the horizontal black line indicates the location of the longitudinal cross-section (Figure A1-9).

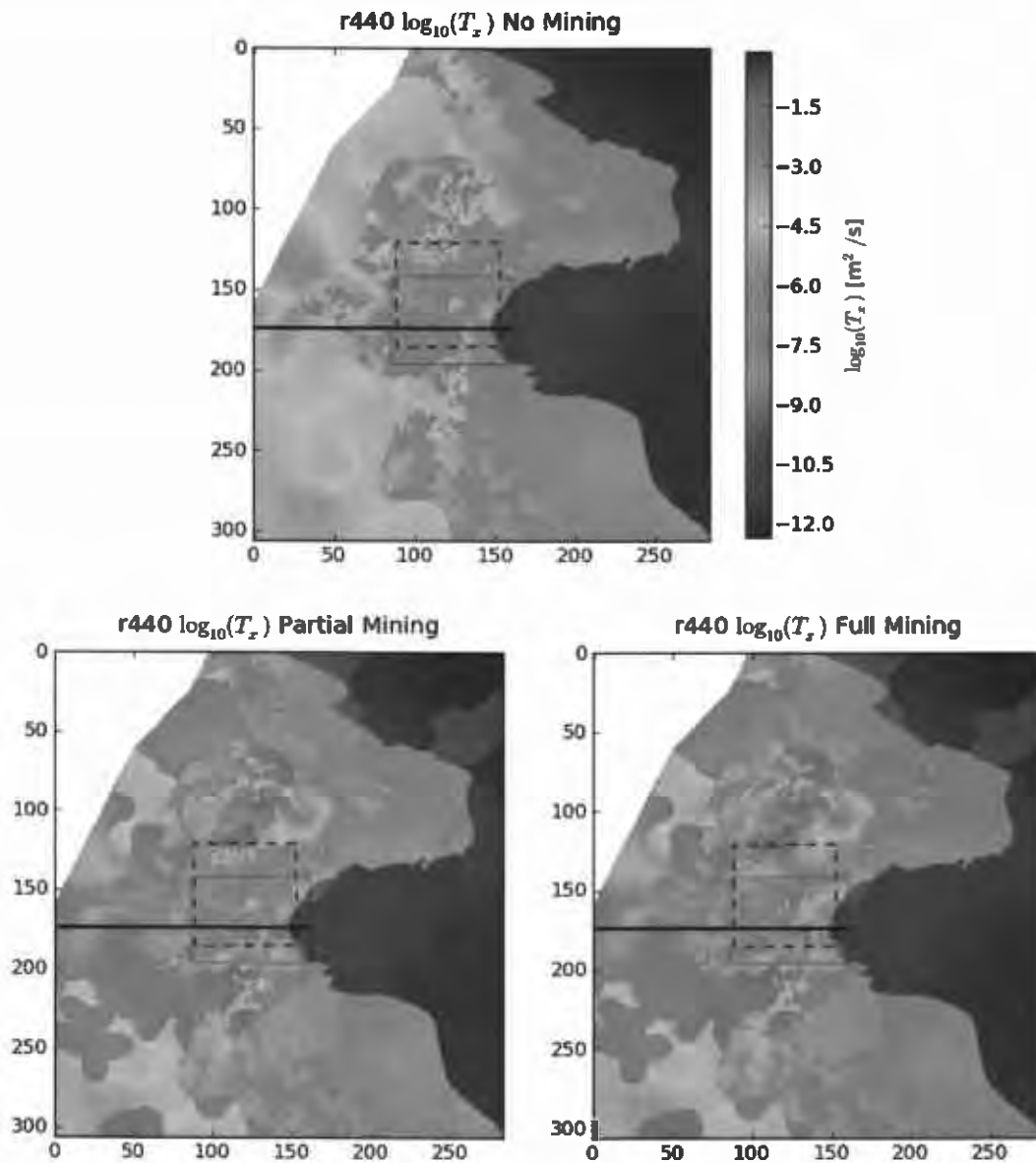


Figure A1-8. Comparison of r440 R2 for non-mined, fully mined, and partially mined scenarios (same color scale in all three plots). Coordinates are cell-based (each cell 100 m on a side); heavy black line indicates location of cross section (Figure A1-9), while LWB (black dashed) and SECOT2D domain (red) are shown for comparison.

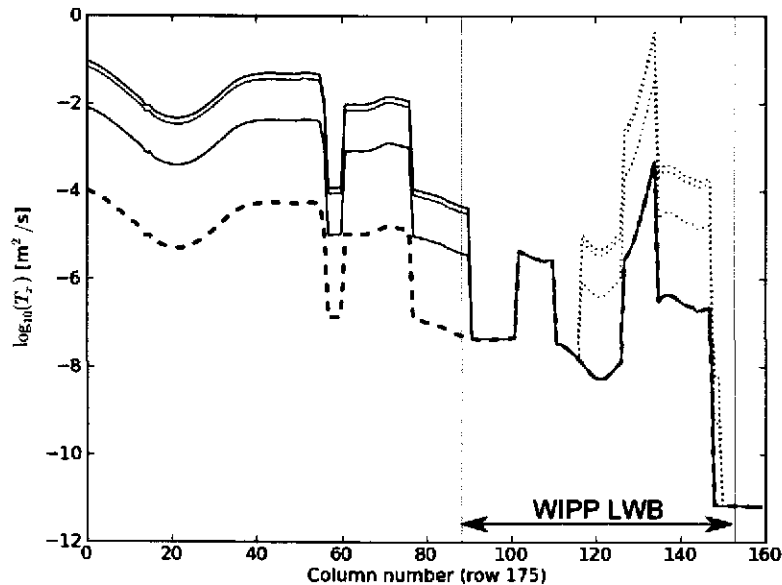


Figure A1-9. Longitudinal cross section through the WIPP LWB (see Figure A1-8 for location) showing all seven T-fields associated with r440. Dashed black line is original non-mined T field, three solid lines are partial-mining scenarios, three dotted lines are full-mining scenarios. Light vertical lines indicate extent of LWB.

A1.5 Step 4 – Run MODFLOW and DTRKMF for Each Realization

Once the mining-modified T-fields and IBOUND arrays are created for the 100 m × 100 m mesh corresponding to each realization and mining scenario, the main Bash shell script `run_mining_mods.sh` (§A4.3) at line 93 calls the Bash shell script `link_input_run_mf_dtrk.sh` (§A4.6), which creates symbolic links to the required MODFLOW (Table A1-13) and DTRKMF (Table A1-14) input files that are the same between all of the 700 scenarios run. Linking is faster and uses less disk space than making hundreds of duplicates of common input files. T distribution files (different between each of the 700 realizations) were already put into their corresponding subdirectories by the `mining_mods.py` script (see Section A1.4.2). The files to be linked are listed in a text input file named `files`, which is read by the Bash shell script. The output from running DTRKMF is used to create CDF plots of the travel time to the WIPP LWB from the C-2737 release point.

Table A1-13. Input and output files for MODFLOW flow calculations (100 m × 100 m grid)

| File ^{1,2} | Remark |
|---|------------------------------|
| Inputs | |
| Inputs/AP114_Task7/modflow/mf2k_head.ba ⁶ | basic package |
| Inputs/AP114_Task7/modflow/mf2k_head.dis ⁵ | discretization package |
| Inputs/AP114_Task7/modflow/mf2k_head.nam ⁵ | name list |
| Inputs/AP114_Task7/modflow/mf2k_head.de4.nam ^{3,5} | name list when DE4 used |
| Inputs/AP114_Task7/modflow/mf2k_head.oc ⁵ | output control package |
| Inputs/AP114_Task7/modflow/mf2k_head.rch ⁵ | recharge package |
| Inputs/AP114_Task7/modflow/mf2k_culebra.lmg ⁵ | LMG solver package |
| Inputs/AP114_Task7/modflow/mf2k_culebra.lpf ⁵ | LPF package |
| Inputs/mf2k_culebra.de4 ³ | DE4 solver package |
| Inputs/AP114_Task7/data/init_bnds.inf ^{4,5} | non-mined IBOUND array |
| Inputs/AP114_Task7/data/init_head.mod ⁵ | initial head array |
| Inputs/AP114_Task7/data/elev_bot.mod ⁵ | bottom elevation array |
| Inputs/AP114_Task7/data/elev_top.mod ⁵ | top elevation array |
| Inputs/AP114_Task7/Outputs/r??/modeled_A_field.mod ⁵ | 100 A field arrays |
| Inputs/AP114_Task7/Outputs/r??/modeled_R_field.mod ⁵ | 100 R field arrays |
| Inputs/AP114_Task7/Outputs/r??/modeled_S_field.mod ⁵ | 100 S field arrays |
| Outputs/Rn/M/r??/modeled_K_field.mod | 600 mined K field arrays |
| Outputs/R0/r??/modeled_K_field.mod | 100 non-mined K field arrays |
| Outputs/data/init_bnds_M.inf | 2 mined IBOUND arrays |
| Outputs | |
| Outputs/Rn/M/r??/modeled_head.lst | 600 ASCII narrative listings |
| Outputs/R0/r??/modeled_head.lst | 100 ASCII narrative listings |
| Outputs/Rn/M/r??/modeled_flow.bud | 600 binary flow budgets |
| Outputs/R0/r??/modeled_flow.bud | 100 binary flow budgets |
| Outputs/Rn/M/r??/modeled_head.bin | 600 binary head files |
| Outputs/R0/r??/modeled_head.bin | 100 binary head files |
| Outputs/Rn/M/r??/mf2k_stdout | 600 screen outputs |
| Outputs/R0/r??/mf2k_stdout | 100 screen outputs |
| Outputs/Rn/M/r??/lmg_err.tmp | error file (not saved) |

1. CVS repository MiningMod
2. $n \in \{1,2,3\}$; $M \in \{\text{full,part}\}$; $r?? \in \{r256, r902, \text{etc.}\}$ (see Table A1-6)
3. these input files are only used when the LMG solver fails to converge in the allotted time
4. this input file is only used in the 100 non-mined realizations
5. Inputs/AP114_Task7/ is the working directory where files checked out from CVS repository Tfields are located, see (Long, 2010)

The Bash shell script (§A4.6) has two main loops. The first loop (lines 26-70) goes over the 100 original T-fields in their non-mined state (realization R0 in the directory structure — Figure A1-3), setting up input files, running MODFLOW and DTRKMF, for the 100 m grid. The second loop (lines 78-145) does largely the same thing for each of the 600 mining-modified T-fields (100 realizations × 3 replicates R{1,2,3} × 2 mining types {full,part}). Screen output from running MODFLOW and DTRKMF is redirected to text files (mf2k_stdout and dtrkmf_stdout) that are checked into CVS.

Approximately once per every 500 times the algebraic multigrid solver package (LMG) does not converge due to problems associated with large changes in transmissivity as a result of mining (e.g., very high and low transmissivity cells juxtaposed without a smooth transition). To automate the analysis in the case where the model fails to converge, similar to an approach used in AP-114 Task 7 (Hart et al., 2009), a time limit is placed on the execution of MODFLOW when using the LMG solver via the built-in Bash command `ulimit` (see line 8 of the Bash shell script `link_input_run_mf_dtrk.sh`). In each step of the innermost loop, after running MODFLOW, the script checks for the presence of an error file, `lmg_err.tmp` (lines 118-120 in §A4.6), which would be generated if the simulation failed to converge and was terminated by the shell for running longer than the allotted time. When this error file is found, the simulation is restarted using the direct solver package (DE4), which always converges but runs at least 10-20 times slower than the LMG solver. The DE4 solver is part of the standard MODFLOW-2000 executable, but previously was not tested for WIPP use. New tests were added to the test suite to cover the direct solver (Hart, 2010). The time limit defined using `ulimit` is extended to an unlimited amount of time (line 125), and the simulation is re-run with the DE4 solver, concatenating the screen output from this second simulation onto the output from the first simulation (the screen-redirectioned output file in these restarted cases will include screen output from both simulations).

Table A1-14. Input and output files for DTRKMF particle tracking calculations (100 m × 100 m grid)

| File ^{1,2} | Remark |
|---|--|
| Inputs | |
| <code>Inputs/dtrkmf/wippctrl.inp</code> | DTRKMF input file |
| <code>Inputs/dtrkmf/dtrkmf.in</code> | DTRKMF screen input |
| <code>Outputs/Rn/M/r??/modeled_flow.bud</code> | 600 MF2K mined budgets (see Table A1-13) |
| <code>Outputs/R0/r??/modeled_flow.bud</code> | 100 MF2K non-mined budgets |
| <code>Inputs/AP114_Task7/data/elev_bot.mod^{3,4}</code> | Culebra bottom elevation (<code>fort.34</code>) ³ |
| <code>Inputs/AP114_Task7/data/elev_top.mod^{3,4}</code> | Culebra top elevation (<code>fort.33</code>) ³ |
| <code>Inputs/AP114_Task7/modflow/mf2k_head.dis⁴</code> | MF2K discretization package |
| Outputs | |
| <code>Outputs/Rn/M/r??/dtrk.dbg</code> | 600 ASCII narrative listings |
| <code>Outputs/R0/r??/dtrk.dbg</code> | 100 ASCII narrative listings |
| <code>Outputs/Rn/M/r??/dtrk.out</code> | 600 ASCII particle tracks |
| <code>Outputs/R0/r??/dtrk.out</code> | 100 ASCII particle tracks |
| <code>Outputs/Rn/M/r??/dtrkmf_stdout</code> | 600 screen output |
| <code>Outputs/R0/r??/dtrkmf_stdout</code> | 100 screen output |

1. CVS repository `MiningMod`
2. $n \in \{1,2,3\}$; $M \in \{\text{full,part}\}$; $r?? \in \{r256,r902, \text{etc.}\}$ (see Table A1-6)
3. DTRKMF expects these files to be opened on Fortran units 33 and 34, therefore the input files are linked to files named `fort.33` and `fort.34`.
4. `Inputs/AP114_Task7/` is the working directory where files checked out from CVS repository `Tfields` are located, see (Long, 2010)

A1.6 Step 5 – Post-Process DTRKMF Output for Plotting

The main Bash shell script `run_mining_mods.sh` (§A4.3) at lines 104 and 110 calls two Python scripts `combine_dtrkmf_output_for_gnuplot.py` (§A4.7.2) and `extract_dtrkmf_lwb_travel_times.py` (§A4.8.2) to post-process the output from DTRKMF for producing CDF and particle track figures (see Table A1-15). These data are not used further in PA,

but the scripts are included here and the results are checked into CVS for completeness. Each of the 700 realizations produces a DTRKMF output file (`dtrk.out`), where each record or line in the file is a point along the path taken by a particle. The first three columns of output are the time, DTRKMF column, and DTRKMF row indicating the location of each particle. The scripts perform two conversions on this output, printing the results in two different formats.

The row and column output of DTRKMF is converted to UTM NAD27 Zone 13 coordinates (meters), the travel times are adjusted to a 4-m Culebra transport thickness (from the 7.75-m thickness used in the flow calculations), and the results are saved in one of two formats. The Python script `combine_dtrkmf_output_for_gnuplot.py` puts all 100 realizations for each of the seven different mining / replicate combinations into a single file (resulting in seven large text files with two carriage returns between the points associated with particle tracks for each realization); this allows plotting of the “horsetail plots” of particles using Gnuplot (Figure 3-11 through Figure 3-14). The Python script `extract_dtrkmf_lwb_travel_times.py` extracts the last particle location from each file (the last row), saving them to seven different files, each with 100 rows indicating travel time to the WIPP LWB for each of the realizations. These data are then further processed to create an additional seven files with the particles sorted by travel time and a cumulative probability column (going from 0.01 to 1.0 in steps of 0.01), for creation of a CDF plot (Figure 3-9).

Table A1-15. Input and output files for DTRKMF post-processing Python scripts

| File ^{1,2} | Remark |
|--|---|
| Inputs | |
| Inputs/combine_dtrkmf_output_for_gnuplot.in | Python inputs with filenames and coordinates |
| Inputs/extract_dtrkmf_lwb_travel_times.in | Python inputs with filenames and coordinates |
| Outputs/R0/r???.dtrk.out | 100 DTRKMF ASCII particle tracks |
| Outputs/Rn/M/r???.dtrk.out | 600 DTRKMF ASCII particle tracks |
| Outputs | |
| Outputs/Summary/ dtrk_all_realizations_Rn_M.dat | 6 summaries of 100 full particle traces |
| Outputs/Summary/ dtrk_nomining_all_realizations.dat | summary of 100 non-mined particle traces |
| Outputs/Summary/wiplwb_results_Rn_M.out | 6 summaries of time and location each realization's particle exits the LWB |
| Outputs/Summary/wiplwb_results_Rn_M.sort | 6 summaries of sorted time and cumulative probability for particles exiting the LWB |
| Outputs/Summary/ wiplwb_results_nomining.out | summaries of time and location each non-mined realization's particle exits the LWB |
| Outputs/Summary/ wiplwb_results_nomining.sort | summary of sorted time and cumulative probability for non-mined realization's particles exiting the LWB |

1. CVS repository \$CVSLIB/MiningMod
2. $n \in \{1,2,3\}$; $M \in \{\text{full,part}\}$; $r??? \in \{r256, r902, \text{etc.}\}$ (see Table A1-6)

A1.7 Step 6 – Convert MODFLOW Input Data to Transport Mesh

The SECOTP2D transport mesh is comprised of 50 m square elements, while the MODFLOW mesh used for calibration is made up of 100 m square elements. Each square 100 m MODFLOW element is subdivided into four equal-sized square elements, with the properties of the larger cells copied directly into the smaller cells without averaging or smoothing. The boundary head is the only property that is smoothed in the transition from the coarser to the finer mesh. The main Bash shell script `run_mining_mods.sh` (§A4.3) at line 121 calls the Python script `100x100_to_50x50.py` (§A4.9.2 and Table A1-16), which reads in the two MODFLOW IBOUND arrays (full and partial mining), the top and bottom elevation arrays, the starting head array and the four different parameter fields for each realization: 300 non-mined parameter fields (anisotropy, recharge, and storage) + 100 realizations × 3 replicates × 2 mining types for hydraulic conductivity = 900 total expanded parameter fields. The corresponding quadrupled input files are saved to re-run MODFLOW for creating the inputs to SECOTP2D (the original 100 unmodified T-fields are not re-run, since they were only needed for comparison in the particle track and CDF results and figures).

Table A1-16. Input and output files for Python script 100x100_to_50x50.py

| File ^{1,2,3} | Remark |
|--|--|
| Inputs | |
| Inputs/100x100_to_50x50.in | Python input file with filenames |
| Outputs/data/init_bnds_full.inf | full-mining IBOUND (see Table A1-7 outputs) |
| Outputs/data/init_bnds_part.inf | partial-mining IBOUND (see Table A1-7 outputs) |
| Inputs/AP114_Task7/data/elev_top.mod ⁴ | Culebra top elevation field |
| Inputs/AP114_Task7/data/elev_bot.mod ⁴ | Culebra bottom elevation field |
| Inputs/AP114_Task7/data/init_head.mod ⁴ | Initial and specified head field |
| Inputs/AP114_Task7/Outputs/r???:modeled_W_field.mod ⁴ | 300 calibrated non-mined property fields |
| Outputs/Rn/r???:modeled_K_field.mod | 600 mined T-fields |
| Outputs | |
| Outputs/data/init_bnds_full.inf.50 | full-mining IBOUND for 50 m grid |
| Outputs/data/init_bnds_part.inf.50 | partial-mining IBOUND for 50 m grid |
| Outputs/data/elev_top.mod.50 | Culebra top elevation for 50 m grid |
| Outputs/data/elev_bot.mod.50 | Culebra bottom elevation for 50 m grid |
| Outputs/data/init_head.mod.50 | Initial and specified head for 50 m grid |
| Outputs/R0/r???:modeled_W_field.mod.50 | 300 non-mined properties for 50 m grid |
| Outputs/Rn/r???:modeled_K_field.mod.50 | 600 mined T-fields for 50 m grid |

1. CVS repository MiningMod
2. $n \in \{1,2,3\}$; $M \in \{\text{full,part}\}$; $r??? \in \{r256,r902, \text{etc.}\}$ (see Table A1-6)
3. $W \in \{A,R,S\}$
4. Inputs/AP114_Task7/ is the working directory where files checked out from CVS repository Tfields are located, see (Long, 2010)

At lines 148-159 of the 100x100_to_50x50.py Python script, the top and bottom elevations are converted to the finer mesh; Table A1-17 illustrates a small subsection of the two matrices. Since no manipulation besides expanding the grid was necessary, the values were treated as strings (no conversion to floating point) to increase the execution speed of the script and prevent changes to the input data due to conversion. The bottom elevation array was treated similarly; the value in each cell is exactly 7.75 m lower than the top elevation, in both the coarse and finer mesh grids.

Table A1-17. Northwest corner of Culebra top elevation field; top half is original field (4x4), bottom half is expanded field corresponding to same area (8x8). Four and 16 cells are colored-coded to illustrate correspondence.

| | | | | | | | |
|--------|--------|--------|--------|--------|--------|--------|--------|
| 911.91 | 911.99 | 912.08 | 912.18 | | | | |
| 911.89 | 911.98 | 912.09 | 912.21 | | | | |
| 911.86 | 911.96 | 912.08 | 912.21 | | | | |
| 911.79 | 911.90 | 912.03 | 912.18 | | | | |
| 911.91 | 911.91 | 911.99 | 911.99 | 912.08 | 912.08 | 912.18 | 912.18 |
| 911.91 | 911.91 | 911.99 | 911.99 | 912.08 | 912.08 | 912.18 | 912.18 |
| 911.89 | 911.89 | 911.98 | 911.98 | 912.09 | 912.09 | 912.21 | 912.21 |
| 911.89 | 911.89 | 911.98 | 911.98 | 912.09 | 912.09 | 912.21 | 912.21 |
| 911.86 | 911.86 | 911.96 | 911.96 | 912.08 | 912.08 | 912.21 | 912.21 |
| 911.86 | 911.86 | 911.96 | 911.96 | 912.08 | 912.08 | 912.21 | 912.21 |
| 911.79 | 911.79 | 911.90 | 911.90 | 912.03 | 912.03 | 912.18 | 912.18 |
| 911.79 | 911.79 | 911.90 | 911.90 | 912.03 | 912.03 | 912.18 | 912.18 |

The initial head array was smoothed (see lines 162-219) along the exterior two rows or columns of the domain and therefore the input file was converted to floating point (unlike the top elevation array). The smoothing was accomplished by first copying the data into the finer mesh without averaging (line 171), then applying a smoothing stencil to the head data (see script lines 180-216 and examples in Table A1-18 and Figure A1-10).

Table A1-18. Initial head along north edge of model domain (row 1) in original (top) and expanded (bottom) starting head.

| | | | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 942.800 | 942.834 | 942.867 | 942.899 | 942.931 | | | | | |
| 942.780 | 942.808 | 942.825 | 942.842 | 942.858 | 942.875 | 942.891 | 942.907 | 942.923 | 942.939 |

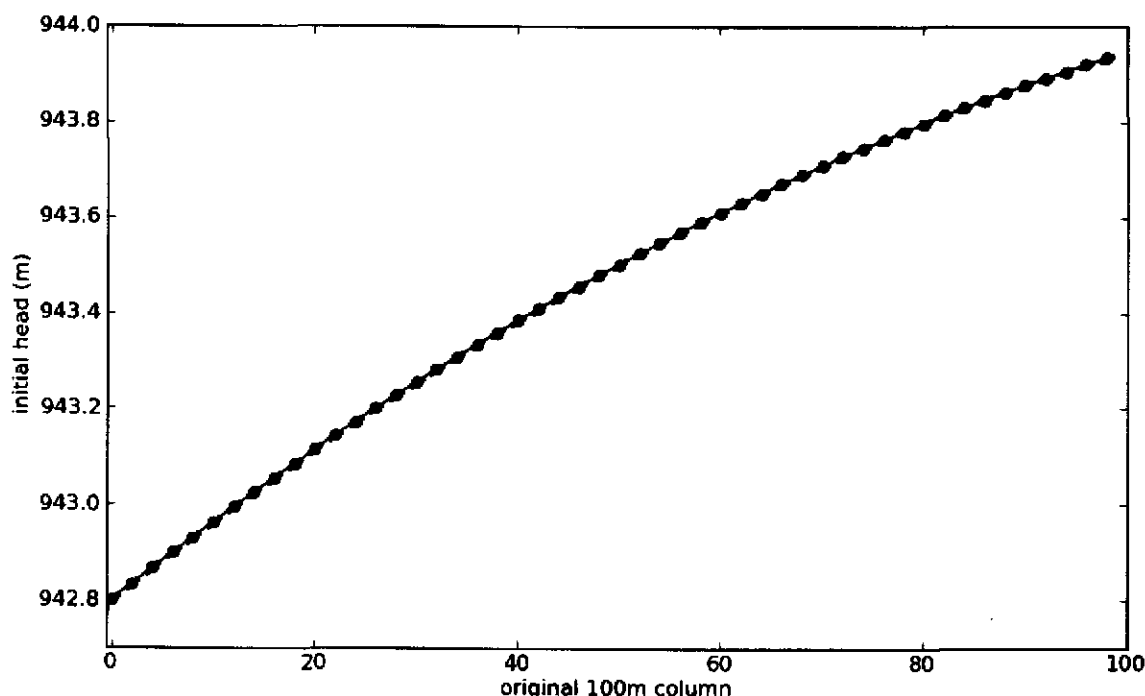


Figure A1-10. Comparison of initial head arrays for original and expanded grids. First 100 columns of first row (north boundary) in the original (red circles) and first two rows of the expanded (blue dots) model

The MODFLOW IBOUND arrays (one for full and another for partial mining) are expanded and copied into the quadrupled grid in the same manner as the top and bottom elevation arrays (see lines 222-240 of script and Table A1-19). The anisotropy (A), recharge (R), and storativity (S) arrays are expanded to the finer grid for each of the 100 realizations – 300 fields total – in lines 243-260 of the Python script. Like the IBOUND arrays, the top & bottom elevation arrays, and the T-fields, these A-, R-, and S-fields are read in and written as strings, avoiding the string-to-float conversion overhead and potential rounding errors.

Table A1-19. Section of IBOUND arrays at southeast corner of domain for 100 m grid (top) and 50 m grid (bottom). -1 (in red) indicates specified head; 1 indicates an active cell.

| | | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 |
| -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |

Table A1-20. Filenames of corresponding MODFLOW input files for 100 m and 50 m grid problems

| Type | Original 100 m grid input filename | Quadrupled 50 m grid input filename |
|------------------------|------------------------------------|-------------------------------------|
| Culebra elevations | elev_{top,bot}.mod | elev_{top,bot}.mod.50 |
| Initial heads | init_head.mod | init_head.mod.50 |
| IBOUND array | init_bnds_{part,full}.inf | init_bnds_{part,full}.inf.50 |
| T,A,R and S arrays | modeled_{T,A,R,S}_field.mod | modeled_{T,A,R,S}_field.mod.50 |
| Name list | mf2k_head.nam | mf2k_head_50.nam |
| Discretization package | mf2k_head.dis | mf2k_head_50.dis |

Aside from the data arrays, there are a few MODFLOW input files that reference the number of rows and columns in the domain, and therefore must also be modified (see Table A1-20). These input files are modified from the original versions checked out from AP-114 Task 7 in the Bash shell script run_mining_mods.sh (§A4.3 lines 130-152) using sed substitute “s/original/replace/” commands.

A1.8 Step 7 – Run MODFLOW Using the Finer Transport Mesh

Very similar to the Bash shell script link_input_run_mf_dtrkmf.sh (§A4.6) discussed in Section A1.5, the main Bash shell script run_mining_mods.sh (§A4.3) at line 162 calls the Bash shell script run_50x50_modflow.sh (§A4.10), which prepares the required input files and executes MODFLOW for each of the 600 cases (100 realizations × 2 mining types (full or partial) × 3 replicates (R1,R2,R3) =

600), but unlike the script that executes the 100 m MODFLOW model, the script that drives the 50 m MODFLOW model does not additionally run DTRKMF after each flow simulation. The 2-minute time limit is enforced to handle cases where the LMG solver does not converge (see previous discussion for details); the fraction of runs that do not converge for the quadrupled grid is a larger proportion than for the original coarser mesh, approximately 1 in 100. When the LMG solver converges, it does so in less than 10 seconds; the direct solver always works, but has a run time of several minutes.

The only result used from running MODFLOW with the finer mesh is the cell-by-cell flow budget, which is used as an input (after some pre-processing) for SECOTP2D.

A1.9 Step 8 – Convert Binary MODFLOW Budget Files to ASCII for VTRAN2

The main Bash shell script `run_mining_mods.sh` (§A4.3) at line 173 calls the Bash shell script `convert_rename_modflow_50x50_budget.sh` (§A4.11), which loops over the 600 MODFLOW simulations using the finer transport mesh, calling the Python script `budget_bin2ascii.py` (§A4.12) in each directory, renaming and moving the resulting ASCII output to a common directory.

A1.9.1 Binary to ASCII Budget File Conversion

The Python script `budget_bin2ascii.py` (§A4.12.2) reads the MODFLOW binary cell-by-cell budget file, and writes an ASCII file that is comprised of two matrices, separated by a blank line. The binary budget files for the Culebra MODFLOW model have the structure outlined in Table A1-21 (two header records and some data). The first header record is formatted in little-endian format with the following data: two integers (`kstp` and `kper`), a 16-character string (`text`), and three integers (`ncol`, `nrow`, and `nlay` – see line 130 of §A4.12.2). Based on the value in the `text` field in the first header; the value is either read in as `qx` (FLOW RIGHT FACE), `qy` (FLOW FRONT FACE), or it is skipped (lines 150, 154, and 158). The second header is not used.

Table A1-21. General structure of a data block in a MODFLOW budget files

| |
|--|
| Header1: (<code>kstp</code> , <code>kper</code> , <code>text</code> , <code>ncol</code> , <code>nrow</code> , <code>nlay</code>) |
| Header2: (discarded) |
| data |

The ASCII budget is saved as two large matrices in scientific notation (specified in line two of the input file) in the natural, unwrapped format. Each matrix for the PABC-2009 MODFLOW model is 307 rows by 284 columns, with a blank line between the matrices.

A single realization (`r440`) is plotted for each of the two mining scenarios and three replicates in Figure A1-11, to illustrate the effect mining type and mining multiplier have on the resulting transport flowfield, here only showing the portion of the flowfield used for transport analysis in SECOTP2D. The WIPP LWB, WIPP panels, and release point are indicated in the six plots.

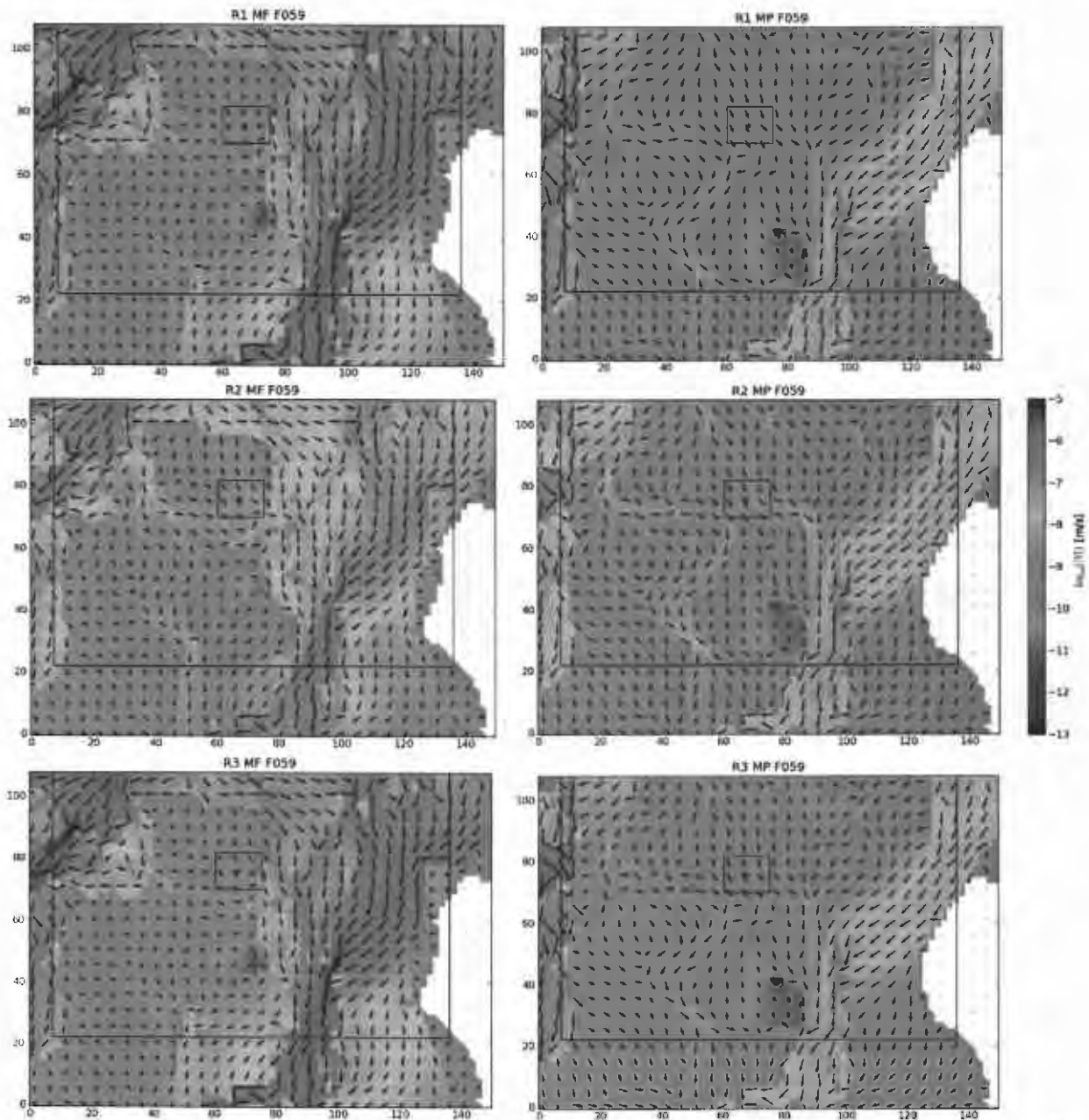


Figure A1-11. VTRAN2 output (\log_{10} Darcy velocity in m/s) corresponding to a single MODFLOW realization (r440). Left column is full mining, right column is partial mining; mining factors: R1=663.4, R2=77.07, R3=902.6. Color scale is the same between all figures, relative vector length scaling is not.

Appendix 2 SECOTP2D Transport Process Narrative

This appendix describes the methods used to organize and implement the SECOTP2D Culebra radionuclide transport calculations.

The SECOTP2D solute transport code is used in conjunction with a preprocessor (PRESECOTP2D) and a postprocessor (POSTSECOTP2D). In addition, several utility codes are used to sample from the probability distributions for subjectively uncertain parameters, define the mesh, set material properties and set model parameters. These codes and their use are described in the following sections. All of these codes are located in the WIPP Software Configuration Management System (SCMS) (Long, 2010) and are run on the WIPP PA Alpha Cluster. All codes used in the analysis, with the exception of the VTRAN2 utility code, are qualified per Nuclear Waste Management Procedure NP 19-1: Software Requirements (Chavez, 2006). The VTRAN2 utility is qualified for use in this analysis per Nuclear Waste Management Procedure NP 9-1: Analyses (Chavez, 2008). VTRAN2 code validation testing is presented in Appendix 3.

The computations are divided into several steps according to the number of times a particular code is run, for efficiency, and to allow for inspection of intermediate results. Digital Command Language (DCL) run control scripts have been written to orchestrate the calculations in each step. Using these scripts, the calculations were performed under formal run control procedures by the WIPP PA Run Control Coordinator. See the PABC-2009 Run Control Report (Long, 2010) for detailed information on the WIPP PA Run Control System.

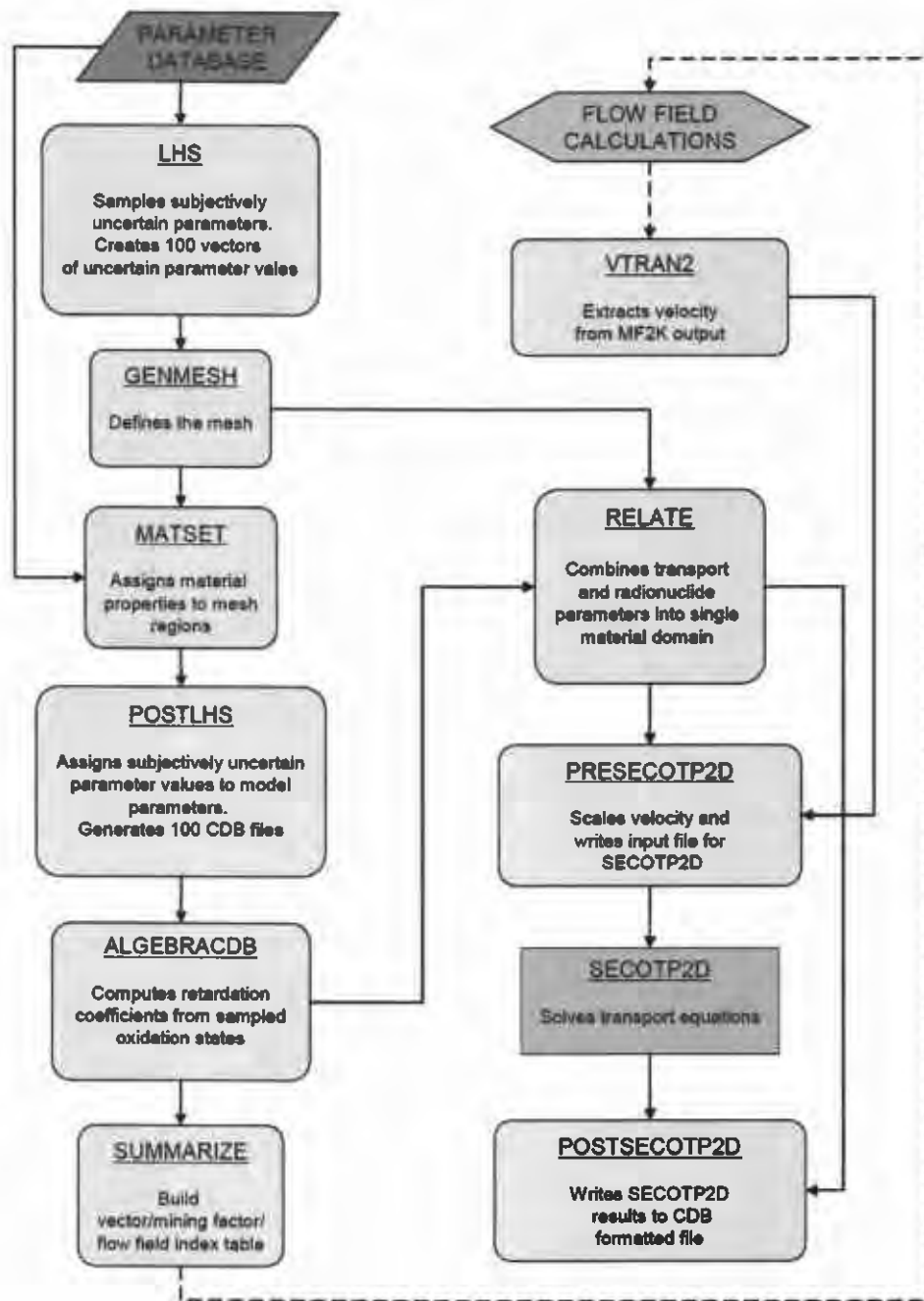


Figure A2-1. Culebra transport calculation flowchart

A2.1 Step 0 – Sampling of Subjectively Uncertain Parameters

The subjectively uncertain Culebra transport parameters were not sampled in isolation. Instead, all subjectively uncertain parameters used in the PABC-2009 were sampled at once at the beginning of the PA calculations in order to impose the appropriate correlations. The Latin hypercube sampling code LHS was used for this purpose. The LHS sampling is reported in (Kirchner, 2010).

A2.2 Step 1 – Mesh Generation and Material Property Assignment

Step 1 is run once. The Step 1 script uses GENMESH version 6.08 to generate the computational grid and MATSET version 9.10 to assign material properties to element blocks. The Step 1 script, GENMESH and MATSET executables, script input and log files, as well as the input and output files for GENMESH and MATSET are listed in Table A2-1.

The GENMESH input file specifies a uniform computational mesh consisting of 50 m square cells, with 151 cells in the *x*-direction, and 109 cells in the *y*-direction. All cells are 4 m thick.

The MATSET input file specifies 13 element blocks. Block 1 is for Culebra physical transport properties. Block 2 is for global properties. Block 3 is for reference constants. Blocks 4 through 7 are for the atomic weights and half-lives of radionuclide species ²⁴¹Am, ²³⁹Pu, ²³⁰Th, and ²³⁴U, respectively. Blocks 8 through 13 are for the distribution coefficients and molecular diffusion coefficients for the species/oxidation state combinations Am(III), Pu(III), Pu(IV), Th(IV), U(IV), and U(VI), respectively.

MATSET assigns the values of deterministic model parameters to the appropriate element blocks. For uncertain model parameters, the median value is assigned to the appropriate element blocks.

Table A2-1. Step 1 input and output files

| | File Names | CMS Library | CMS Class |
|----------------|----------------------------|----------------|--------------------|
| SCRIPT | EVAL_GENERIC_STEP1.COM | LIBPABC09_EVAL | GENERIC_STEP1_V1.4 |
| Input | EVAL_ST2D_PABC09_STEP1.INP | LIBPABC09_EVAL | PABC09-0 |
| Log | EVAL_ST2D_PABC09_STEP1.LOG | LIBPABC09_ST2D | PABC09-0 |
| GENMESH | GM_PA96.EXE | LIBGM | PA96 |
| Input | GM_ST2D_PABC09.INP | LIBPABC09_ST2D | PABC09-0 |
| Output | GM_ST2D_PABC09.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | GM_ST2D_PABC09.DBG | Not kept | Not kept |
| MATSET | MATSET_QA0910.EXE | LIBMS | QA0910 |
| Input | MS_ST2D_PABC09.INP | LIBPABC09_ST2D | PABC09-0 |
| Input | GM_ST2D_PABC09.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | MS_ST2D_PABC09.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | MS_ST2D_PABC09.DBG | Not kept | Not kept |

A2.3 Step 2 – Uncertain Parameter Assignments

Step 2 is run once per replicate. The Step 2 script uses POSTLHS version 4.07A to assign the sampled parameter values to the appropriate element block properties. The Step 2 script, POSTLHS executable, script input and log files, as well as the input and output files for POSTLHS are shown in Table A2-2.

POSTLHS loops over all 100 vectors in the replicate. The result is 100 CAMDAT database (CDB) files per replicate, with each CDB file containing the sampled parameter values for that replicate/vector combination. Sampled values for physical parameters for replicates R1, R2, and R3 are shown in Table A2-3.

Table A2-2. Step 2 input and output files

| | File names^{1,2} | CMS Library | CMS Class |
|----------------|---------------------------------|--------------------|--------------------|
| SCRIPT | EVAL_GENERIC_STEP2.COM | LIBPABC09_EVAL | GENERIC_STEP2_V1.2 |
| Input | EVAL_ST2D_PABC09_STEP2_Rr.INP | LIBPABC09_EVAL | PABC09-0 |
| Log | EVAL_ST2D_PABC09_STEP2_Rr.LOG | LIBPABC09_ST2D | PABC09-0 |
| POSTLHS | POSTLHS_QA0407A.EXE | LIBLHS | QA0407A |
| Input | LHS3_DUMMY.INP | LIBPABC09_LHS | PABC09-0 |
| Input | LHS2_PABC09_Rr.TRN | LIBPABC09_LHS | PABC09-0 |
| Input | MS_ST2D_PABC09.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | LHS3_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | LHS3_ST2D_PABC09_Rr.DBG | LIBPABC09_ST2D | PABC09-0 |

1. $r \in \{1,2,3\}$
2. $vvv \in \{001,002, \dots, 100\}$ for each r

| vector | R1 | | | | R2 | | | | R3 | | | |
|--------|---------|---------|---------|----------|---------|---------|---------|----------|---------|---------|---------|----------|
| | APOROS | DPOROS | HMBLKLT | CLIMTIDX | APOROS | DPOROS | HMBLKLT | CLIMTIDX | APOROS | DPOROS | HMBLKLT | CLIMTIDX |
| 61 | 1.91E-3 | 1.68E-1 | 3.31E-1 | 1.23E+0 | 4.50E-3 | 1.14E-1 | 3.51E-1 | 1.16E+0 | 2.52E-3 | 1.05E-1 | 4.29E-1 | 2.10E+0 |
| 62 | 1.01E-4 | 1.28E-1 | 4.38E-1 | 1.00E+0 | 4.01E-4 | 1.20E-1 | 3.42E-1 | 1.64E+0 | 9.02E-4 | 1.86E-1 | 2.81E-1 | 1.01E+0 |
| 63 | 8.75E-3 | 1.41E-1 | 1.57E-1 | 1.22E+0 | 1.39E-3 | 1.67E-1 | 2.97E-1 | 1.22E+0 | 2.39E-4 | 1.09E-1 | 2.98E-1 | 1.06E+0 |
| 64 | 9.38E-3 | 2.38E-1 | 1.78E-1 | 1.81E+0 | 5.82E-4 | 1.18E-1 | 4.71E-1 | 1.02E+0 | 2.96E-3 | 1.30E-1 | 3.11E-1 | 1.07E+0 |
| 65 | 2.37E-3 | 2.03E-1 | 3.93E-1 | 1.07E+0 | 1.70E-4 | 1.91E-1 | 4.74E-1 | 1.01E+0 | 1.93E-4 | 1.65E-1 | 8.08E-2 | 2.05E+0 |
| 66 | 5.85E-3 | 1.86E-1 | 3.28E-1 | 1.04E+0 | 2.59E-4 | 1.89E-1 | 2.64E-1 | 1.14E+0 | 7.25E-4 | 1.08E-1 | 4.25E-1 | 1.04E+0 |
| 67 | 5.22E-3 | 1.45E-1 | 1.28E-1 | 1.25E+0 | 1.83E-3 | 1.15E-1 | 4.22E-1 | 1.02E+0 | 3.20E-4 | 1.70E-1 | 4.61E-1 | 1.05E+0 |
| 68 | 2.64E-3 | 1.03E-1 | 4.56E-1 | 2.01E+0 | 8.61E-3 | 1.88E-1 | 2.53E-1 | 1.17E+0 | 8.25E-4 | 1.37E-1 | 3.36E-1 | 1.98E+0 |
| 69 | 2.56E-3 | 1.54E-1 | 1.46E-1 | 1.21E+0 | 1.16E-3 | 1.31E-1 | 3.56E-1 | 1.09E+0 | 7.07E-3 | 1.17E-1 | 5.41E-2 | 1.08E+0 |
| 70 | 1.10E-3 | 1.11E-1 | 4.72E-1 | 1.57E+0 | 6.89E-3 | 1.45E-1 | 4.49E-1 | 1.19E+0 | 1.73E-4 | 1.77E-1 | 3.13E-1 | 1.03E+0 |
| 71 | 4.26E-4 | 1.28E-1 | 2.19E-1 | 1.98E+0 | 1.93E-3 | 2.19E-1 | 3.58E-1 | 1.05E+0 | 3.67E-4 | 2.36E-1 | 3.38E-1 | 1.21E+0 |
| 72 | 1.98E-4 | 1.78E-1 | 4.74E-1 | 2.02E+0 | 3.03E-4 | 1.06E-1 | 1.03E-1 | 1.01E+0 | 1.27E-4 | 1.61E-1 | 1.50E-1 | 1.05E+0 |
| 73 | 1.07E-4 | 1.80E-1 | 7.48E-2 | 1.66E+0 | 1.48E-3 | 1.49E-1 | 2.43E-1 | 1.79E+0 | 2.28E-3 | 1.43E-1 | 2.56E-1 | 1.17E+0 |
| 74 | 4.01E-4 | 2.46E-1 | 4.91E-1 | 2.18E+0 | 7.55E-3 | 1.46E-1 | 3.31E-1 | 1.19E+0 | 6.32E-4 | 1.58E-1 | 3.53E-1 | 2.24E+0 |
| 75 | 3.68E-4 | 1.88E-1 | 8.65E-2 | 2.08E+0 | 3.16E-3 | 2.48E-1 | 6.96E-2 | 1.74E+0 | 1.28E-3 | 1.39E-1 | 7.34E-2 | 1.79E+0 |
| 76 | 1.34E-3 | 1.64E-1 | 4.82E-1 | 1.15E+0 | 1.10E-3 | 1.61E-1 | 2.28E-1 | 1.11E+0 | 9.70E-3 | 1.64E-1 | 9.49E-2 | 1.08E+0 |
| 77 | 8.20E-3 | 1.87E-1 | 1.62E-1 | 1.22E+0 | 1.05E-3 | 1.06E-1 | 1.07E-1 | 1.70E+0 | 5.36E-3 | 1.14E-1 | 3.72E-1 | 1.02E+0 |
| 78 | 6.13E-3 | 1.61E-1 | 1.26E-1 | 1.07E+0 | 1.27E-3 | 1.11E-1 | 1.96E-1 | 1.10E+0 | 1.01E-4 | 1.69E-1 | 2.13E-1 | 1.24E+0 |
| 79 | 3.81E-3 | 1.42E-1 | 3.57E-1 | 1.02E+0 | 4.77E-4 | 1.86E-1 | 4.52E-1 | 2.19E+0 | 1.01E-3 | 1.83E-1 | 8.82E-2 | 1.07E+0 |
| 80 | 1.77E-3 | 1.25E-1 | 5.64E-2 | 1.50E+0 | 7.69E-4 | 1.12E-1 | 2.18E-1 | 1.03E+0 | 4.97E-3 | 1.14E-1 | 4.12E-1 | 1.13E+0 |
| 81 | 3.04E-3 | 1.19E-1 | 1.65E-1 | 1.18E+0 | 1.20E-3 | 1.40E-1 | 1.61E-1 | 1.11E+0 | 1.06E-4 | 1.25E-1 | 4.84E-1 | 1.01E+0 |
| 82 | 2.00E-4 | 2.33E-1 | 1.72E-1 | 1.04E+0 | 1.65E-4 | 1.82E-1 | 2.45E-1 | 1.24E+0 | 3.69E-3 | 1.19E-1 | 3.60E-1 | 1.53E+0 |
| 83 | 3.19E-3 | 1.79E-1 | 2.89E-1 | 1.91E+0 | 2.57E-3 | 1.28E-1 | 3.07E-1 | 1.13E+0 | 4.08E-3 | 1.62E-1 | 3.90E-1 | 1.93E+0 |
| 84 | 2.50E-4 | 1.63E-1 | 4.11E-1 | 2.11E+0 | 2.93E-4 | 1.48E-1 | 5.86E-2 | 1.07E+0 | 2.23E-4 | 1.45E-1 | 1.64E-1 | 1.04E+0 |
| 85 | 1.37E-4 | 2.22E-1 | 9.59E-2 | 1.12E+0 | 6.72E-4 | 1.60E-1 | 3.01E-1 | 1.15E+0 | 2.83E-3 | 1.10E-1 | 1.62E-1 | 1.71E+0 |
| 86 | 4.68E-4 | 1.88E-1 | 1.84E-1 | 1.10E+0 | 1.24E-4 | 1.83E-1 | 4.87E-1 | 2.12E+0 | 2.93E-4 | 1.86E-1 | 3.23E-1 | 1.18E+0 |
| 87 | 1.44E-4 | 1.17E-1 | 4.54E-1 | 1.08E+0 | 1.19E-4 | 1.68E-1 | 3.20E-1 | 1.07E+0 | 4.41E-3 | 1.41E-1 | 2.36E-1 | 1.16E+0 |
| 88 | 1.53E-3 | 1.34E-1 | 7.10E-2 | 1.08E+0 | 3.29E-3 | 1.36E-1 | 2.51E-1 | 2.19E+0 | 1.15E-3 | 1.87E-1 | 1.05E-1 | 1.91E+0 |
| 89 | 2.45E-3 | 1.23E-1 | 1.93E-1 | 1.84E+0 | 4.94E-3 | 1.78E-1 | 3.70E-1 | 1.22E+0 | 1.17E-4 | 1.12E-1 | 2.26E-1 | 1.10E+0 |
| 90 | 3.39E-3 | 1.83E-1 | 6.59E-2 | 1.12E+0 | 2.94E-3 | 1.62E-1 | 2.23E-1 | 1.18E+0 | 8.97E-3 | 1.89E-1 | 3.69E-1 | 1.11E+0 |
| 91 | 1.18E-3 | 1.12E-1 | 1.85E-1 | 2.20E+0 | 3.76E-3 | 1.66E-1 | 3.45E-1 | 1.55E+0 | 2.31E-3 | 1.80E-1 | 2.64E-1 | 1.00E+0 |
| 92 | 3.93E-4 | 1.14E-1 | 9.39E-2 | 1.23E+0 | 1.02E-4 | 1.58E-1 | 8.14E-2 | 1.12E+0 | 3.41E-4 | 1.06E-1 | 2.32E-1 | 1.03E+0 |
| 93 | 6.39E-3 | 1.38E-1 | 1.21E-1 | 1.19E+0 | 6.35E-4 | 1.90E-1 | 3.63E-1 | 1.05E+0 | 1.10E-3 | 1.42E-1 | 1.90E-1 | 1.23E+0 |
| 94 | 1.84E-3 | 1.17E-1 | 3.36E-1 | 1.14E+0 | 8.34E-4 | 1.55E-1 | 3.96E-1 | 1.02E+0 | 3.53E-3 | 2.49E-1 | 4.55E-1 | 1.14E+0 |
| 95 | 7.32E-3 | 1.08E-1 | 3.47E-1 | 1.10E+0 | 1.32E-4 | 1.35E-1 | 2.14E-1 | 2.25E+0 | 1.73E-3 | 1.84E-1 | 2.23E-1 | 1.06E+0 |
| 96 | 6.64E-3 | 1.82E-1 | 1.38E-1 | 2.24E+0 | 2.16E-4 | 1.63E-1 | 1.11E-1 | 1.23E+0 | 1.89E-3 | 2.44E-1 | 1.45E-1 | 1.13E+0 |
| 97 | 6.56E-4 | 1.71E-1 | 4.31E-1 | 1.78E+0 | 5.80E-3 | 1.71E-1 | 3.80E-1 | 1.03E+0 | 2.50E-4 | 1.90E-1 | 3.97E-1 | 1.25E+0 |
| 98 | 1.56E-4 | 1.59E-1 | 3.88E-1 | 1.70E+0 | 3.89E-4 | 2.34E-1 | 2.89E-1 | 1.86E+0 | 3.34E-3 | 1.46E-1 | 5.92E-2 | 1.09E+0 |
| 99 | 2.98E-4 | 1.00E-1 | 2.23E-1 | 1.17E+0 | 1.54E-4 | 1.41E-1 | 4.43E-1 | 1.24E+0 | 1.26E-3 | 1.68E-1 | 9.97E-2 | 1.06E+0 |
| 100 | 1.72E-4 | 1.16E-1 | 2.03E-1 | 1.20E+0 | 2.70E-3 | 1.70E-1 | 4.68E-1 | 1.04E+0 | 4.42E-4 | 1.66E-1 | 3.44E-1 | 1.02E+0 |

A2.4 Step 3 - Evaluation of Oxidation-State-Dependent Parameters

Step 3 is run once per replicate. The Step 3 script loops over all 100 vectors in the replicate, invoking ALGEBRACDB version 2.35 and RELATE version 1.43 for each vector. The Step 3 script, ALGEBRACDB and RELATE executables, script input and log files, as well as the input and output files for ALGEBRACDB and RELATE are shown in Table A2-5

The ALGEBRACDB code selects the sampled distribution coefficient and molecular diffusion coefficient based upon the value of the sampled oxidation state. The distribution coefficient is then used to calculate the retardation factor. The RELATE code is used to remove all element blocks except for CULEBRA (all required non-CULEBRA block properties are transferred to the CULEBRA block). Sampled values for chemical parameters are shown in Table A2-4.

| VECTOR | R1 | | | | | R2 | | | | | R3 | | | | |
|--------|--------|---------|---------|---------|---------|--------|---------|---------|---------|---------|--------|---------|---------|---------|---------|
| | OXSTAT | MKD_AM | MKD_PU | MKD_TH | MKD_U | OXSTAT | MKD_AM | MKD_PU | MKD_TH | MKD_U | OXSTAT | MKD_AM | MKD_PU | MKD_TH | MKD_U |
| 60 | H | 5.40E-2 | 5.17E-3 | 6.63E-3 | 1.55E-2 | H | 1.87E-2 | 2.23E-3 | 4.04E-3 | 5.88E-5 | L | 5.47E-2 | 1.28E-2 | 2.37E-2 | 5.33E-1 |
| 61 | L | 4.16E-2 | 8.40E-2 | 4.07E-3 | 1.27E+0 | L | 2.95E-1 | 1.45E-2 | 3.02E-3 | 4.75E-3 | L | 8.95E-2 | 8.35E-2 | 4.83E-3 | 9.88E-3 |
| 62 | L | 1.04E-1 | 1.75E-2 | 8.67E-1 | 4.61E+0 | L | 7.51E-3 | 3.15E-1 | 2.06E-1 | 7.12E-1 | L | 6.36E-2 | 5.74E-3 | 4.14E-1 | 5.94E-2 |
| 63 | L | 1.78E-2 | 6.87E-2 | 2.47E-2 | 9.16E-1 | H | 3.47E-1 | 5.16E-1 | 6.44E-3 | 1.41E-3 | H | 2.14E-2 | 5.83E-2 | 1.23E-3 | 1.27E-4 |
| 64 | L | 4.75E-2 | 2.33E-2 | 8.33E+0 | 5.43E-3 | H | 4.01E-2 | 1.52E-3 | 1.34E-2 | 6.19E-3 | L | 3.53E-2 | 5.04E-3 | 2.61E-2 | 1.09E+0 |
| 65 | H | 2.45E-2 | 1.62E-3 | 7.02E+0 | 1.03E-2 | H | 1.29E-1 | 1.13E+0 | 9.76E+0 | 8.39E-5 | H | 6.32E-3 | 8.21E-1 | 1.24E-2 | 4.40E-3 |
| 66 | H | 2.78E-2 | 7.53E-1 | 2.36E-3 | 1.29E-2 | H | 9.34E-2 | 8.91E+0 | 7.52E-3 | 1.99E-3 | L | 1.11E-2 | 4.48E-2 | 2.94E-1 | 3.06E-1 |
| 67 | H | 1.51E-1 | 7.05E-3 | 3.07E-2 | 1.34E-4 | H | 8.07E-3 | 6.39E+0 | 2.15E-2 | 4.60E-3 | L | 1.38E-1 | 1.44E-2 | 5.13E+0 | 1.39E-1 |
| 68 | H | 1.01E-1 | 2.87E-2 | 5.30E-3 | 1.19E-4 | L | 1.76E-2 | 7.19E-3 | 4.00E-2 | 7.67E-4 | H | 6.93E-2 | 2.27E+0 | 8.56E+0 | 8.29E-5 |
| 69 | L | 1.59E-1 | 5.13E-2 | 2.94E-1 | 1.77E+0 | L | 5.91E-2 | 5.58E-3 | 4.75E-2 | 1.26E+0 | H | 1.85E-2 | 3.67E+0 | 1.03E-1 | 2.65E-4 |
| 70 | H | 3.97E-1 | 4.54E-2 | 9.69E+0 | 1.50E-4 | L | 2.47E-2 | 1.54E-1 | 9.00E-3 | 3.44E+0 | L | 1.42E-1 | 2.96E-1 | 5.92E-1 | 5.01E+0 |
| 71 | H | 5.91E-2 | 1.40E-3 | 1.18E-1 | 4.00E-4 | L | 1.80E-1 | 1.81E-1 | 2.32E-2 | 4.79E-1 | L | 2.59E-1 | 4.87E-2 | 4.72E-3 | 2.29E+0 |
| 72 | L | 8.40E-3 | 1.28E-2 | 3.68E-2 | 3.19E+0 | H | 2.40E-2 | 3.64E+0 | 5.51E-1 | 2.79E-4 | L | 1.09E-2 | 2.60E-1 | 1.32E-1 | 1.28E-1 |
| 73 | H | 2.07E-1 | 6.84E-1 | 5.60E-3 | 1.16E-3 | H | 8.12E-2 | 1.29E-1 | 2.01E+0 | 1.22E-2 | H | 1.99E-1 | 4.53E+0 | 5.24E-4 | 3.78E-3 |
| 74 | H | 8.45E-2 | 2.58E-3 | 1.11E-1 | 9.83E-5 | H | 1.08E-2 | 1.65E-3 | 4.81E+0 | 4.62E-5 | L | 5.97E-2 | 4.13E-2 | 4.09E-3 | 1.67E-2 |
| 75 | H | 1.88E-2 | 5.14E+0 | 2.32E+0 | 3.41E-4 | L | 5.59E-3 | 8.68E-2 | 3.73E-1 | 1.14E+0 | L | 1.38E-2 | 1.72E-1 | 1.29E+0 | 9.05E-1 |
| 76 | H | 2.19E-1 | 4.77E+0 | 9.97E-3 | 1.21E-2 | H | 6.96E-3 | 7.12E-4 | 1.76E+0 | 1.40E-2 | L | 1.03E-2 | 3.43E-1 | 3.62E-3 | 4.47E-3 |
| 77 | H | 6.92E-3 | 1.71E-2 | 3.38E+0 | 4.51E-5 | L | 3.19E-1 | 3.17E-2 | 9.98E-2 | 8.18E-1 | L | 6.26E-2 | 1.67E-2 | 2.71E-2 | 1.38E+0 |
| 78 | L | 5.06E-2 | 2.56E-2 | 1.28E-3 | 1.17E+0 | L | 2.25E-2 | 7.15E-2 | 3.91E-1 | 5.80E-2 | L | 3.41E-2 | 4.39E-2 | 4.71E+0 | 3.15E-2 |
| 79 | L | 1.22E-1 | 2.98E-1 | 5.20E-4 | 7.64E-2 | L | 2.48E-1 | 6.16E-3 | 5.25E-4 | 1.26E-2 | L | 4.79E-2 | 3.34E-1 | 2.99E-2 | 7.30E-4 |
| 80 | L | 3.34E-1 | 1.94E-1 | 5.71E-1 | 5.74E-4 | L | 1.05E-2 | 3.35E-2 | 2.53E-1 | 1.78E+0 | L | 8.06E-2 | 1.08E-1 | 2.84E-3 | 6.05E-1 |
| 81 | L | 6.74E-3 | 1.88E-2 | 1.49E-2 | 1.86E-2 | H | 3.09E-2 | 3.35E-2 | 1.58E+0 | 1.99E-2 | L | 5.17E-2 | 1.87E-1 | 6.43E-3 | 1.90E-2 |
| 82 | L | 1.35E-1 | 8.01E-2 | 4.57E-2 | 2.48E-2 | L | 8.15E-3 | 2.65E-1 | 3.00E-1 | 2.72E-1 | L | 7.63E-3 | 1.15E-1 | 5.62E-1 | 2.76E+0 |
| 83 | H | 8.89E-2 | 6.17E+0 | 7.21E-1 | 6.19E-4 | L | 1.62E-1 | 1.01E-1 | 6.05E-1 | 2.03E-3 | H | 5.49E-3 | 6.43E-2 | 4.35E-1 | 6.65E-4 |
| 84 | H | 5.27E-2 | 5.72E-3 | 1.68E-1 | 1.11E-2 | L | 1.65E-2 | 2.82E-1 | 1.30E-3 | 2.69E-2 | H | 2.86E-1 | 5.12E+0 | 2.06E-3 | 1.36E-4 |
| 85 | L | 2.38E-1 | 3.48E-2 | 6.60E-1 | 6.16E-2 | L | 2.07E-1 | 1.64E-1 | 2.82E-3 | 9.85E-4 | H | 3.63E-2 | 1.50E-2 | 2.21E-1 | 2.75E-3 |
| 86 | H | 5.64E-2 | 5.00E-1 | 2.28E-2 | 9.18E-3 | L | 1.83E-1 | 1.12E-2 | 1.30E+0 | 2.65E+0 | L | 4.48E-2 | 3.15E-1 | 8.30E-2 | 1.61E-3 |
| 87 | H | 2.82E-1 | 1.60E+0 | 4.02E+0 | 7.12E-5 | L | 1.17E-1 | 2.64E-2 | 6.58E-2 | 5.50E-1 | L | 9.24E-3 | 2.03E-2 | 4.94E-1 | 1.43E-1 |
| 88 | H | 4.06E-2 | 4.20E-2 | 1.57E-3 | 1.44E-3 | H | 3.83E-2 | 4.84E-3 | 2.46E-3 | 1.23E-4 | H | 9.57E-2 | 1.57E+0 | 1.91E+0 | 4.53E-5 |
| 89 | L | 5.39E-3 | 8.55E-3 | 5.43E+0 | 1.65E-1 | L | 1.51E-2 | 2.65E-2 | 1.43E-3 | 2.97E-1 | H | 3.93E-1 | 3.05E-1 | 1.82E-2 | 4.81E-3 |
| 90 | H | 1.86E-1 | 5.90E-4 | 3.89E-1 | 7.65E-5 | H | 2.54E-2 | 5.89E+0 | 8.04E-1 | 3.09E-4 | L | 3.81E-1 | 1.31E-1 | 1.82E+0 | 7.09E-1 |
| 91 | L | 2.95E-1 | 1.34E-1 | 7.83E-1 | 1.95E-1 | L | 6.62E-3 | 9.52E-2 | 3.15E+0 | 9.47E-2 | H | 3.23E-2 | 5.72E+0 | 2.37E+0 | 3.54E-3 |
| 92 | H | 1.17E-2 | 3.22E-2 | 6.10E-2 | 6.62E-3 | L | 1.34E-2 | 3.53E-2 | 5.89E-3 | 3.16E+0 | H | 3.06E-2 | 1.87E-3 | 9.53E-4 | 7.49E-3 |
| 93 | H | 3.08E-1 | 2.24E-3 | 3.27E+0 | 1.62E-3 | H | 1.91E-1 | 2.04E-3 | 6.78E-3 | 1.56E-2 | L | 6.07E-3 | 2.84E-1 | 1.15E+0 | 4.43E+0 |
| 94 | H | 3.73E-1 | 1.89E-1 | 2.76E-2 | 3.66E-5 | L | 7.73E-2 | 3.84E-2 | 1.84E-2 | 7.10E-4 | H | 2.61E-2 | 2.23E-2 | 1.68E-3 | 2.41E-4 |
| 95 | L | 3.92E-2 | 3.59E-1 | 3.39E-3 | 1.46E-1 | H | 9.15E-3 | 2.51E-2 | 1.31E-1 | 5.97E-4 | H | 8.81E-3 | 4.06E-2 | 6.02E+0 | 1.75E-2 |
| 96 | H | 2.50E-1 | 2.67E+0 | 5.82E-4 | 5.41E-3 | H | 8.71E-3 | 4.13E+0 | 3.45E-3 | 7.62E-3 | H | 1.73E-1 | 3.70E-3 | 1.05E-2 | 3.07E-5 |
| 97 | L | 1.12E-1 | 2.55E-1 | 1.78E-1 | 3.51E-1 | L | 1.47E-1 | 4.15E-2 | 4.28E+0 | 4.33E-3 | H | 2.80E-2 | 1.33E-1 | 7.40E-4 | 2.15E-3 |
| 98 | H | 9.65E-3 | 1.79E-3 | 7.46E-4 | 4.29E-5 | L | 3.31E-1 | 2.48E-2 | 2.25E-1 | 1.20E-1 | H | 2.23E-2 | 1.70E-3 | 3.19E+0 | 1.21E-3 |
| 99 | H | 2.58E-2 | 9.64E-2 | 6.43E-3 | 1.79E-3 | H | 1.40E-2 | 1.80E-1 | 7.39E+0 | 1.50E-4 | L | 3.57E-1 | 2.54E-1 | 2.75E+0 | 5.72E+0 |
| 100 | H | 3.41E-1 | 7.64E-4 | 2.65E-3 | 4.13E-3 | H | 5.39E-3 | 5.10E-3 | 2.26E-3 | 1.16E-4 | H | 1.33E-2 | 5.89E-4 | 3.80E+0 | 9.68E-3 |

Table A2-5. Step 3 script, executables, input, and output files

| | File names^{1,2} | CMS Library | CMS Class |
|-------------------|---------------------------------|--------------------|------------------|
| SCRIPT | EVAL_ST2D_STEP3.COM | LIBPABC09_EVAL | ST2D_STEP3_V1.0 |
| Input | EVAL_ST2D_PABC09_STEP3_Rr.INP | LIBPABC09_EVAL | PABC09-0 |
| Log | EVAL_ST2D_PABC09_STEP3_Rr.LOG | LIBPABC09_ST2D | PABC09-0 |
| ALGEBRACDB | ALGEBRACDB_PA96.EXE | LIBALG | PA96 |
| Input | ALG_ST2D_PABC09.INP | LIBPABC09_ST2D | PABC09-0 |
| Input | LHS3_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | ALG_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | ALG_ST2D_PABC09_Rr_Vvvv.DBG | Not kept | Not kept |
| RELATE | RELATE_PA96.EXE | LIBREL | |
| Input | REL_ST2D_PABC09.INP | LIBPABC09_ST2D | PABC09-0 |
| Input | GM_ST2D_PABC09.CDB | LIBPABC09_ST2D | PABC09-0 |
| Input | ALG_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | REL_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | REL_ST2D_PABC09_Rr_Vvvv.DBG | Not kept | Not kept |

1. $r \in \{1,2,3\}$
2. $vvv \in \{001,002, \dots, 100\}$ for each r

A2.5 Step 4 – Tabulation of Mining Factors and Flow-Field Indices

Step 4 is run once per replicate. The Step 4 script fetches all 100 ALGEBRACDB output files produced in Step 3, then runs SUMMARIZE version 2.20 on them. The Step 4 script, SUMMARIZE executable, script input and log files as well as the input and output files for SUMMARIZE are shown in Table A2-6.

The SUMMARIZE code is used to construct tables of the uncertain mining factor parameter CULEBRA:MINP_FAC and the flow-field index parameter CULEBRA:TRANSIDX (flow-field index = INT[CULEBRA:TRANSIDX]). These tables are transferred to the WIPP PA Pentium Cluster for use in the Culebra flow calculations (see Section A1.4). Each table contains four columns: the vector number, time (not used), MINP_FAC, and TRANSIDX.

Table A2-6. Step 4 script, executables, input and output files

| | File names ^{1,2} | CMS Library | CMS Class |
|------------------|-------------------------------|----------------|--------------------|
| SCRIPT | EVAL_ST2D_STEP4.COM | LIBPABC09_EVAL | GENERIC_STEP4_V1.0 |
| Input | EVAL_ST2D_PABC09_STEP4_Rr.INP | LIBPABC09_EVAL | PABC09-0 |
| Output | SUM1_ST2D_PABC09_Rr.INP | LIBPABC09_ST2D | PABC09-0 |
| Log | EVAL_ST2D_PABC09_STEP4_Rr.LOG | LIBPABC09_ST2D | PABC09-0 |
| SUMMARIZE | SUMMARIZE_QA_Q220.EXE | LIBSUM | QA0220 |
| Input | SUM1_ST2D_PABC09_Rr.INP | LIBPABC09_ST2D | PABC09-0 |
| Input | ALG_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | SUM1_ST2D_PABC09_Rr.TBL | LIBPABC09_ST2D | PABC09-0 |
| Output | SUM1_ST2D_PABC09_Rr.LOG | Not kept | Not kept |
| Output | SUM1_ST2D_PABC09_Rr_ERROR.LOG | Not kept | Not kept |

1. $r \in \{1,2,3\}$
2. $vvv \in \{001,002, \dots, 100\}$ for each r

A2.6 Step 5 – Flow-Field Extraction

The rules for converting the MODFLOW output data (volumetric flux) to SECOTP2D input data (Darcy velocity) described in Section 4.1.2 were implemented in the Fortran code VTRAN2. VTRAN2 neither models physical phenomena nor solves differential equations that model physical phenomena. Rather, it is a utility code that processes the output data produced by a modeling code and formats that data for use in another modeling code. VTRAN2 has been qualified for this analysis per Nuclear Waste Management Procedure NP 9-1: Analysis (Chavez, 2008). The source code listing, build information, and verification testing for VTRAN2 are provided in Appendix 3.

The VTRAN2 code takes five command line arguments, four required and one optional. All arguments are the names of input or output files, descriptions of which follow:

1. **cmd_file** is an input ASCII format command file. The command file describes the MODFLOW mesh, the SECOTP2D mesh, the x - and y -direction offsets between the two meshes and the format that was used to write the MODFLOW velocities into the ASCII budget file (see below).
2. **bud_file** is an input ASCII format MODFLOW budget file containing the volumetric flux values for each cell in the groundwater flow modeling mesh.
3. **trn_file** is a binary format output file containing the groundwater flow velocities for the transport domain (including the ghost cells) in the format required by the SECOTP2D transport code.
4. **dbg_file** is an output ASCII format diagnostic/debug file containing information about the VTRAN2 run.
5. **txt_file** is an optional output ASCII format file containing the same data as the **trn_file**.

A sample command line that executes VTRAN2 is shown below:

```
$ VTRAN2 TEST.CMD TEST.BUD TEST.TRN TEST.DBG TEST.TXT
```


The flow-fields obtained from the Culebra flow calculations introduce the concept of mining scenarios into the transport calculations. The Step 5 script is run once per replicate/mining scenario combination. The script loops over all 100 flow-field indices in the replicate/scenario combination, using the VTRAN2 utility code to extract the mining-modified Culebra flow-fields (corresponding to the sub-domain used in the Culebra transport calculations) from the MODFLOW output files. The script, executable, script input and log files, along with the input and output files for the VTRAN2 utility are shown in Table A2-7.

Table A2-7. Step 5 script, executable, input and output files

| | File Names^{1,2,3} | CMS Library | CMS Class |
|---------------------|-----------------------------------|--------------------|------------------|
| SCRIPT | EVAL_ST2D_STEP5.COM | LIBPABC09_EVAL | ST2D_STEP5_V1.0 |
| Script Input | EVAL_ST2D_PABC09_STEP5_Rr_Mm.INP | LIBPABC09_EVAL | PABC09-0 |
| Log | EVAL_ST2D_PABC09_STEP5_Rr_Mm.LOG | LIBPABC09_MF2K | PABC09-0 |
| VTRAN2 | VTRAN2.EXE | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Input | VTRAN2_PABC09.INP | LIBPABC09_MF2K | PABC09-0 |
| Input | MF2K_PABC09_Rr_Mm_Ffff.OUT | LIBPABC09_MF2K | PABC09-0 |
| Output | MF2K_PABC09_Rr_Mm_Ffff.TRN | LIBPABC09_MF2K | PABC09-0 |
| Output | VTRAN2_ST2D_PABC09_Rr_Mm_Ffff.DBG | Not kept | Not kept |

1. $r \in \{1,2,3\}$
2. $m \in \{F,P\}$
3. $fff \in \{001,002, \dots, 100\}$ for each r and each m

A2.7 Step 6 – Transport Calculations

The Step 6 script runs the SECOTP2D suite of codes (PRESECOTP2D version 1.22, SECOTP2D version 1.41A, and POSTSECOTP2D version 1.04) to calculate radionuclide transport through the Culebra. Two DCL run control scripts are used in Step 6. The master script is invoked once for each replicate/scenario combination. The master script loops over all 100 vectors for each replicate/scenario combination. For each vector, the master script performs the following steps:

- Use GROPECDB version 2.12 to extract the value of the CULEBRA:TRANSIDX parameter from the ALGEBRACDB output file generated in Step 3. The value of this parameter indicates the flow-field index to use with the vector.
- Write input file for slave script
- Run the slave script

The slave script then runs PRESECOTP2D, SECOTP2D, and POSTSECOTP2D for that vector. Both the master and the slave scripts produce log files to record their actions. In the paragraphs that follow, the input and output files for the generic case are described, then the procedure followed to re-run certain replicate/scenario/vector combinations with a modified PRESECOTP2D input file to overcome numerical stability problems is described.

Table A2-8. Step 6 script, executables, input and output files (general case)

| | File Names ^{1,2,3,4,5} | CMS Library | CMS Class |
|----------------------|---------------------------------------|----------------|-----------------|
| MASTER SCRIPT | EVAL_ST2D_STEP6_MASTER.COM | LIBPABC09_EVAL | ST2D_STEP6_V1.0 |
| Input | EVAL_ST2D_PABC09_STEP6_Rr_Mm.INP | LIBPABC09_EVAL | PABC09-0 |
| Log | EVAL_ST2D_PABC09_STEP6_Rr_Mm.LOG | LIBPABC09_ST2D | PABC09-0 |
| SLAVE SCRIPT | EVAL_ST2D_STEP6_MASTER.COM | LIBPABC09_EVAL | ST2D_STEP6_V1.0 |
| Log | EVAL_ST2D_PABC09_STEP6_Rr_Mm_Vvvv.LOG | LIBPABC09_ST2D | PABC09-0 |
| GROPECDB | GROPECDB_PA96.EXE | LIBGR | PA96 |
| Input | GROPE_ST2D_PABC09.INP | LIBPABC09_ST2D | PABC09-0 |
| Input | ALG_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | GROPE_ST2D_PABC09_Rr_Mm_Vvvv.TXT | Not kept | Not kept |
| PRESECOTP2D | PRESECOTP2D_QA0122.EXE | LIBST2D | QA0122 |
| Input | ST2D1_PABC09.INP | LIBPABC09_ST2D | PABC09-0 |
| Input | REL_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Input | MF2K_PABC09_Rr_Mm_Ffff.TRN | LIBPABC09_MF2K | PABC09-0 |
| Output | ST2D2_PABC09_Rr_Mm_Vvvv.INP | Not kept | Not kept |
| Output | ST2D1_PABC09_Rr_Mm_Vvvv.PRP | Not kept | Not kept |
| Output | ST2D1_PABC09_Rr_Mm_Vvvv.VEL | Not kept | Not kept |
| Output | ST2D1_PABC09_Rr_Mm_Vvvv.DBG | Not kept | Not kept |
| SECOTP2D | SECOTP2D_QA0141A.EXE | LIBST2D | QA0141A |
| Input | ST2D2_PABC09_Rr_Mm_Vvvv.INP | Not kept | Not kept |
| Input | ST2D1_PABC09_Rr_Mm_Vvvv.PRP | Not kept | Not kept |
| Input | ST2D1_PABC09_Rr_Mm_Vvvv.VEL | Not kept | Not kept |
| Output | ST2D2_PABC09_Rr_Mm_Vvvv.BIN | Not kept | Not kept |
| Output | ST2D2_PABC09_Rr_Mm_Vvvv.DBG | Not kept | Not kept |
| POSTSECOTP2D | POSTSECOTP2D_QA0104.EXE | LIBST2D | QA0104 |
| Input | ST2D2_PABC09_Rr_Mm_Vvvv.BIN | Not kept | Not kept |
| Input | REL_ST2D_PABC09_Rr_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | ST2D3_PABC09_Rr_Mm_Vvvv.CDB | LIBPABC09_ST2D | PABC09-0 |
| Output | ST2D3_PABC09_Rr_Mm_Vvvv.DBG | LIBPABC09_ST2D | PABC09-0 |

1. $r \in \{1,2,3\}$
2. $m \in \{F,P\}$
3. $fff \in \{001,002, \dots, 100\}$ for each m
4. $vvv \in \{001,002, \dots, 100\}$ for each m
5. flow-field index matched with number through the TRANSIDX parameter for each vector

In the few instances where SECOTP2D failed due to numerical instability using the generic numerical control parameters, a new PRESECOTP2D input file was submitted by the analyst and the case was re-run in a manner similar to that described above. In order to track these cases, a special tag ("MOD") was inserted into the PRESECOTP2D input file name, as well as the master script input file and log file names. The vectors for each combination of replicate and mining scenario that required modified PRESECOTP2D input files are shown in Table A2-9. The modified file names for each vector are shown in

Table A2-10. Other files have the same names as for the generic case. Files in the libraries from the previous runs were replaced with files from the re-run.

Table A2-9. Step 6 modified input runs

| | Full Mining (MF) | Partial Mining (MP) |
|-----------|-------------------------|----------------------------|
| R1 | 6, 35, 63, 65, 88, 95 | 46, 88 |
| R2 | - | 2, 9, 26, 27, 40, 99 |
| R3 | 49 | 32, 52, 53, 69 |

Table A2-10. Step 6 modified input run file names

| | File Names^{1,2,3} | CMS Library | CMS Class |
|----------------------|---|--------------------|------------------|
| MASTER SCRIPT | | | |
| Input | EVAL_ST2D_PABC09_STEP6_Rr_Mm_Vvvv_MOD.INP | LIBPABC09_EVAL | PABC09-0 |
| Log | EVAL_ST2D_PABC09_STEP6_Rr_Mm_Vvvv_MOD.LOG | LIBPABC09_ST2D | PABC09-0 |
| PRESECOTP2D | | | |
| Input | ST2D1_PABC09_Rr_Vvvv.INP | LIBPABC09_ST2D | PABC09-0 |
| | <ol style="list-style-type: none"> 1. $r \in \{1,2,3\}$ 2. $m \in \{F,P\}$ 3. $vvv \in \{001,002, \dots, 100\}$ for each m | | |

Appendix 3 VTRAN2 Utility

This appendix provides the source listing, build information, and a summary of testing for the Fortran utility code VTRAN2. VTRAN2 was originally written for PABC-2004 (Lowry and Kanney, 2005) and was used to extract the groundwater flow velocity data from ASCII versions of the MODFLOW cell-by-cell flow budget files (see Sections 4.1.1 and A2.6). The VTRAN source code and input files were trivially changed from PABC-2004 to accommodate a MODFLOW grid 568 elements wide (previously 448 elements).

A3.1 VTRAN2 Build Info

The Fortran source code file for the VTRAN2 utility (VTRAN2.F) is shown in Table A3-1. This file is archived in CMS library LIBPABC09_MF2K, class VTRAN2_V1.1 on the WIPP PA Alpha Cluster.

Table A3-1. VTRAN2.F source listing

```

PROGRAM VTRAN
C
C
C      nrowmf - number of rows cells in the mf2k grid
C      ncolmf - number of columns in the mf2k grid
C
C      ncx - number of cells in x-direction in the st2d grid
C      ncy - number of cells in y-direction in the st2d grid
C
C      jshftx - x offset of transport domain (# of cells in col direction)
C      ishfty - y offset of transport domain (# of cells in row direction)
C
C
C      PARAMETER (nfmt = 2, mxfile=5)
C
C      CHARACTER*80 author, date, title
C      CHARACTER*80 filenm(mxfile)
C      CHARACTER*80 fmat(nfmt),rdfmt
C      CHARACTER*80 fmcmd, fnmbud, fnmtrn, fnmdbg, fnmvel
C
C      INTEGER ierr
C      INTEGER iunscr, iuncmd, iunbud, iuntrn, iundbg, iunvel
C      INTEGER nfiles, nfiler
C      INTEGER nrowmf, ncolmf, ncx, ncy
C      INTEGER istart, ishft, jstart, jshift
C      INTEGER match, irdfmt
C      DOUBLE PRECISION time
C      DOUBLE PRECISION ayz_inv, axz_inv, dx, dy, dz
C      DOUBLE PRECISION, ALLOCATABLE :: qxin(:, :), qyin(:, :),
C      DOUBLE PRECISION, ALLOCATABLE :: qxout(:, :), qyout(:, :)
C
C      LOGICAL wrtvel
C
C-----
C.....Setup
C-----
C.....Assign file unit numbers
C
C      nfiles = 5
C
C      iunscr = 6
C      iuncmd = 11
C      iunbud = 12
C      iuntrn = 13
C      iundbg = 14
C      iunvel = 15
C
C.....Valid budget file input formats
C      modified these format statements to accommodate the 2009PABC
C      MODFLOW model which was bigger than the previous 448 width
C
C      fmat(1) = '(568e16.8)'
C      fmat(2) = '(568e24.16)'
```

```

C-----
C....Process command line (get file names)
C-----

C      WRITE(iunscr,*) 'VTRAN >> Processing command line'

C....Required args (1-4) are fnmcmd, fnmbud, fnmtrn, fnmdbg
C....Optional arg (5) is fnmvel

      CALL filcmdlin( nfiles, nfiler, filenm )

C      write(iunscr,*) 'nfiler = ', nfiler

      IF ( nfiler .GT. nfiles ) THEN
        CALL QAABORT( 'VTRAN>> Too many command line arguments' )
      ELSE
        IF ( nfiler .LT. nfiles-1 ) THEN
          CALL QAABORT( 'VTRAN>> Too few command line arguments' )
        ENDIF
      ENDIF

      fnmcmd = filenm(1)
      fnmbud = filenm(2)
      fnmtrn = filenm(3)
      fnmdbg = filenm(4)

      IF (nfiler .eq. nfiles ) THEN
        wrtvel = .true.
        fnmvel = filenm(5)
      ELSE
        wrtvel = .false.
        fnmvel = 'None'
      ENDIF

C      write(iunscr,*) 'fnmcmd is ', fnmcmd
C      write(iunscr,*) 'fnmbud is ', fnmbud
C      write(iunscr,*) 'fnmtrn is ', fnmtrn
C      write(iunscr,*) 'fnmdbg is ', fnmdbg
C      write(iunscr,*) 'fnmvel is ', fnmvel

C....Open Diagnostics/Debug file

      OPEN (UNIT=iundbg, FILE=fnmdbg, STATUS='UNKNOWN', IOSTAT=ierr)
      IF ( ierr .NE. 0 ) THEN
        CALL QAABORT ('Error opening command file')
      ENDIF

C-----
C....Process command file
C-----

C      WRITE(iunscr,*) 'VTRAN >> Processing command file'

C....Open command file

      OPEN (UNIT=iuncmd, FILE=fnmcmd, STATUS='OLD',
+        READONLY, IOSTAT=ierr)
      IF ( ierr .NE. 0 ) THEN
        WRITE(iundbg,*) 'Error opening command file'
        CALL QAABORT ('Error opening command file')
      ENDIF

C....Read from command file

      READ (iuncmd,*)
      READ (iuncmd,10) author
      READ (iuncmd,*)
      READ (iuncmd,10) date
      READ (iuncmd,*)
      READ (iuncmd,10) title
      READ (iuncmd,*)
      READ (iuncmd,10) rdformat
      READ (iuncmd,*)
      READ (iuncmd,*) iscrn
      READ (iuncmd,*)
      READ (iuncmd,*) ncolmf, nrowmf
      READ (iuncmd,*)
      READ (iuncmd,*) jshftx, ishfty, ncx, ncy
      READ (iuncmd,*)
      READ (iuncmd,*) dx, dy, dz

```

```

10  FORMAT(A80)
11  FORMAT(3e01.3)
C....Close command file

      CLOSE (UNIT=iuncmd,STATUS='KEEP')

C....Send diagnostic output to screen or to debug file

      IF (iscrn .EQ. 0 ) THEN
        iunscr = iundbg
      ENDIF

C....Echo input

      WRITE(iunscr,20) fnmcmd, fnmbud, fnmtrn, fnmdbg, fnmvel

20  FORMAT (1X, 'command file           = ',A80/
+         1X, 'budget file             = ',A80/
+         1X, '(binary) velocity transfer file = ',A80/
+         1X, 'diagnostic/debug file    = ',A80/
+         1X, '(ascii) velocity output file = ',A80
+         )

      WRITE(iunscr,50) author, date, title, rdformat
      WRITE(iunscr,100) iscrn, ncolmf, nrowmf,
+         jshftx, ishfty, ncx, ncy,
+         dx, dy, dz

50  FORMAT(1X,' author   = ',A80/
+         1X,' date     = ',A80/
+         1X,' title    = ',A80/
+         1X,' format   = ',A80
+         )

100 FORMAT(1X,' iscrn   = ',i5 /
+         1X,' ncolmf  = ',i5 /
+         1X,' nrowmf  = ',i5 /
+         1X,' jshftx  = ',i5 /
+         1X,' ishfty  = ',i5 /
+         1X,' ncx     = ',i5 /
+         1X,' ncy     = ',i5 /
+         1X,' dx     = ',e10.4 /
+         1X,' dy     = ',e10.4 /
+         1X,' dz     = ',e10.4
+         )

C....Assign correct format number

      irdfmt = 0
      match = 0
      DO i=1,nfmat
        IF( LLE(rdfmat,fmat(i)) .AND. LLE(fmat(i),rdfmat)) THEN
          irdfmt = i
          match = 1
        ENDIF
      ENDDO

      IF ( match .ne.1 ) THEN
        WRITE(iunscr,*) 'Invalid input format'
        CALL QAABORT ('Invalid input format')
      ENDIF

      WRITE(iunscr,150) irdfmt, rdformat
150  FORMAT(1X,'Using input format (',i2, ') = ',A80)

C....Sanity check. Since ghost cells are added, we must have:
C....jshftx >= 1 and ishfty >= ncy+1

      IF ( jshftx .LT. 1 ) THEN
        WRITE(iunscr,*) 'Invalid jshftx value'
        CALL QAABORT ('Invalid jshftx value')
      ENDIF

      IF ( ishfty .LT. (ncy+1) ) THEN
        WRITE(iunscr,*) 'Invalid ishfty value'
        CALL QAABORT ('Invalid ishfty value')
      ENDIF

C-----
C....Allocate memory
C-----

```

```

C....Mf2k grid is (1:ncolmf,1:nrowmf).
C....Ghost cells placed around transport domain, so ST2D grid is
C....(0:ncx,0:ncy). Thus qxout and qyout are padded to account for
C....required ghost cells

      ALLOCATE( qxin(1:ncolmf,1:nrowmf),
+            qyin(1:ncolmf,1:nrowmf),
+            qxout(0:ncx+1,0:ncy+1),
+            qyout(0:ncx+1,0:ncy+1),
+            STAT=ierr )

      IF ( ierr .NE. 0 ) THEN
        WRITE(iunscr,*) 'Error allocating memory'
        CALL QABORT ('Error allocating memory')
      ENDIF

C-----
C....Read budget file
C-----

      WRITE(iunscr,*) 'VTRAN >> Reading budget file'

      OPEN (UNIT=iunbud, FILE=fnmbud, FORM='FORMATTED',
+        STATUS='OLD', IOSTAT=ierr)
      IF ( ierr .NE. 0 ) THEN
        WRITE(iunscr,*) 'Error opening budget file'
        CALL QABORT ('Error opening budget file')
      ENDIF

      DO i=1,nrowmf
        READ(iunbud,rdfmt) (qxin(j,i),j=1,ncolmf)
      END DO
      READ(iunbud,*)
      DO i=1,nrowmf
        READ(iunbud,rdfmt) (qyin(j,i),j=1,ncolmf)
      END DO

C....Close budget file

      CLOSE (UNIT=iunbud,STATUS='KEEP')

C....Budget file contains volume fluxes, so must divide
C....by area of cell face perpendicular to flow direction
C....to get specific discharge (darcy velocity)

C....X direction

      ayz_inv = 1.d0/(dy*dz)
      DO i=1,nrowmf
        DO j=1,ncolmf
          qxin(j,i) = qxin(j,i) * ayz_inv
        END DO
      END DO

C....Y direction

      axz_inv = 1.d0/(dx*dz)
      DO i=1,nrowmf
        DO j=1,ncolmf
          qyin(j,i) = qyin(j,i) * axz_inv
        END DO
      END DO

C-----
C....Process velocities
C-----

      WRITE(iunscr,*) 'VTRAN >> Processing velocities'

C....Now grab velocities for internal cells and ghost cells.
C....Let (l,m) be indices of the ST2D grid cells, ranging from 0:ncx+1
C....and 0:ncy+1, respectively. We must compute the corresponding
C....MF2K indices. The computed mf2k indices must account for:
C.... 1) The offset of the ST2D grid origin
C.... 2) The opposite sense of the y-coord in the two meshes
C.... 3) ST2D face-centered velocities of a given cell are defined
C....    at the trailing edges of cells (defined according to sense of
C....    the ST2D axes) while the MF2K face-centered velocities are
C....    defined at the "right" and "front" faces of the cell.

```

```

DO m=0,ncy+1
  DO l=0,ncx+1
    j = jshftx + 1
    i = ishfty + 1 - m
    qxout(l,m) = qxin(j-1,i)
    qyout(l,m) = qyin(j,i)
  END DO
END DO

C.... For ST2D, face centered velocities defined at trailing edges
C of cells. Ghost cells are placed around the computational domain,
C but cells on left and bottom do not have defined velocities associated
C with them. Consider the x-dimension with limits [0,x1], with ncx
C regular cells and a ghost cell on each side of the domain. Then
C u(0,m) is not defined,
C u(1,m) = u at x=0, and
C u(ncx+1,m) = u at x1
C Similarly, Consider the y-dimension with limits [0,y1], with ncy
C regular cells and a ghost cell on each side of the domain. Then
C v(l,0) is not defined,
C v(l,1) = v at y=0, and
C v(l,ncy+1) = v at y1

C.....Zero out the undefined components

DO m=0,ncy+1
  qxout(0,m) = 0.D0
END DO

DO l=0,ncx+1
  qyout(l,0) = 0.D0
END DO

C....Change sign of y-velocities. Modflow convention is that
C....flow is positive in direction of increasing row numbers.
C....But row numbers increase in negative y-direction.

DO m=0,ncy+1
  DO l=0,ncx+1
    qyout(l,m) = -qyout(l,m)
  END DO
END DO

C-----
C....Write velocity transfer file
C-----

WRITE(iunscr,*) 'VTRAN >> Writing velocity transfer file'

C....Open the file

OPEN (UNIT=iuntrn, FILE=fnmtrn, FORM='UNFORMATTED',
- STATUS='UNKNOWN',IOSTAT=ierr)
IF ( ierr .NE. 0 ) THEN
  WRITE(iunscr,*) 'Error opening velocity transfer file'
  CALL QABORT ('Error opening velocity transfer file')
ENDIF

C....Write the following line because sf2d wrote it and
C....st2d1 expects it (but does not use them)

time = 0.d0
WRITE(iuntrn) ncx, ncy, time

C....Write velocities to output file. Include the undefined
C....components, since ST2D1 expects them. (ST2D1 reads
C....them in, but does not write them to the velocity file
C....it passes to ST2D2)

WRITE(iuntrn) ( ( qxout(l,m), l=0,ncx+1),m=0,ncy+1 )
WRITE(iuntrn) ( ( qyout(l,m), l=0,ncx+1),m=0,ncy+1 )

C....Close output file

CLOSE (UNIT=iuntrn,STATUS='KEEP')

C-----
C....Write ascii velocity output file
C-----

IF ( wrtvel ) THEN

```



```

WRITE(iunscr,*) 'VTRAN >> Writing ascii velocity output file'

C....Open the file

C   OPEN (UNIT=iunvel, FILE=fmvel, FORM='FORMATTED',
C   +     STATUS='UNKNOWN',IOSTAT=ierr)

      irecl = 568*(23+1)
      OPEN (UNIT=iunvel, FILE=fmvel, FORM='FORMATTED',
      +     STATUS='UNKNOWN',RECL=irecl,IOSTAT=ierr)

      IF ( ierr .NE. 0 ) THEN
        WRITE(iunscr,*) 'Error opening ascii velocity output file'
        CALL QABORT ('Error opening ascii velocity output file')
      ENDIF

C....Write the following line because sf2d wrote it and
C....st2dl expects it (but does not use them)

      time = 0.d0
      WRITE(iunvel,200) ncx, ncy, time
200  FORMAT(1x,2(i5,2x),e16.8)

C....Write velocities to output file

C   changed implied do-loops to one explicit, one implied
C   and added a blank line between the arrays for easier reading
C
      DO m=0,ncy+1
        WRITE(iunvel,rdformat) ( qxout(l,m), l=0,ncx+1)
      enddo
      write(iunvel,*) '      '
      do m=0,ncy+1
        WRITE(iunvel,rdformat) ( qyout(l,m), l=0,ncx+1)
      enddo

C....Close output file

      CLOSE (UNIT=iunvel,STATUS='KEEP')

      ENDIF

C-----
C....Clean up
C-----

      WRITE(iunscr,*) 'VTRAN >> Cleaning up'

      DEALLOCATE(qxin,qxout,qyin,qyout)

      WRITE(iunscr,*) 'VTRAN >> Normal Completion'
      CLOSE (UNIT=iundbg,STATUS='KEEP')

C   Signal normal completion

      STOP 'VTRAN >> Normal Completion'
      END

```

The VTRAN2 utility was compiled using the compile and link commands listed in Table A 3-2. The files and storage locations associated with building the VTRAN2 utility are given in Table A3-3.

Table A 3-2. VTRAN2 VMS build and link commands

```

$ F90 /DEBUG/NOOPTIMIZE VTRAN2.F
$ LINK /EXEC=VTRAN2.EXE VTRAN2.OBJ, CAMCON_LIB/LIB, CAMSUPES_LIB/LIB

```

Table A3-3. VTRAN2 build info

| Description | File | CMS Library | CMS Class |
|-------------|------|-------------|-----------|
|-------------|------|-------------|-----------|

| | | | |
|-------------------|------------|----------------|-------------|
| Source code | VTRAN2.F | LIBPABC09_MF2K | VTRAN2_V1.1 |
| VTRAN2 executable | VTRAN2.EXE | LIBPABC09_MF2K | VTRAN2_V1.1 |

A3.2 VTRAN2 Verification

Test cases for the VTRAN2 utility were slightly modified from those developed for Lowry and Kanney (2005) and executed by the Run Control Coordinator. The executable and input/output files used for the test are shown in Table A3-4.

Table A3-4. VTRAN2 test files

| Description | File | CMS Library | CMS Class |
|-----------------------------|------------------|----------------|-------------|
| VTRAN2 executable | VTRAN2.EXE | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Test 1 | | | |
| Input command file | VTRAN_TEST_1.CMD | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Input budget file | VTRAN_TEST_1.BUD | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Output binary velocity file | VTRAN_TEST_1.TRN | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Output ASCII velocity file | VTRAN_TEST_1.VEL | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Output debug file | VTRAN_TEST_1.DBG | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Test 2 | | | |
| Input command file | VTRAN_TEST_2.CMD | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Input budget file | VTRAN_TEST_2.BUD | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Output binary velocity file | VTRAN_TEST_2.TRN | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Output ASCII velocity file | VTRAN_TEST_2.VEL | LIBPABC09_MF2K | VTRAN2_V1.1 |
| Output debug file | VTRAN_TEST_2.DBG | LIBPABC09_MF2K | VTRAN2_V1.1 |

The VTRAN2 utility code was verified using two test cases. Case 1 demonstrates the conversion of volumetric flux to Darcy velocities, the sign change of the y-direction velocities, and the inclusion of ghost nodes. Case 2 demonstrates that the indexing selects the correct subregion.

Both test cases use the mesh layout shown in Figure A3-1. The volume fluxes are specified on the 10×15 cell mesh. The 4×3 cell sub-region outlined in red represents the transport domain. The dashed lines indicate the ghost nodes. We run VTRAN2 such that it writes the output in both ASCII and binary format, so we can visually inspect the ASCII file to verify the results.

A3.2.1 VTRAN2 Verification Test Case 1

In this test we use a uniform volume flux ($Q_x = Q_y = 1$), and choose $dx = 1$, and $dy = dz = 2$ in the VTRAN2 command file such that $A_x = 4$ and $A_y = 2$. Thus, for the transport mesh, we will have $u = 0.25$ and $v = 0.5$ for all cell faces except for the left and bottom boundaries. $u = 0$ at the left boundary and $v = 0$ at the bottom because the respective velocity at these faces is undefined in the SECOTP2D convention.

The command file (VTRAN2_TEST_1.CMD), and the budget file (VTRAN2_TEST_1.BUD) are shown in Table A3-5 and Table A3-6, respectively. Running VTRAN2 with these input files produces the ASCII velocity file (VTRAN2_TEST_1.VEL) and the diagnostic file (VTRAN2_TEST_1.DBG) shown in Table

A3-7 and Table A3-8, respectively. We note that u and v are 0.25 and -0.5 , as expected. We also note that the 4×3 cell transport domain has been appropriately padded with ghost cells to make a 6×5 array for each velocity component.

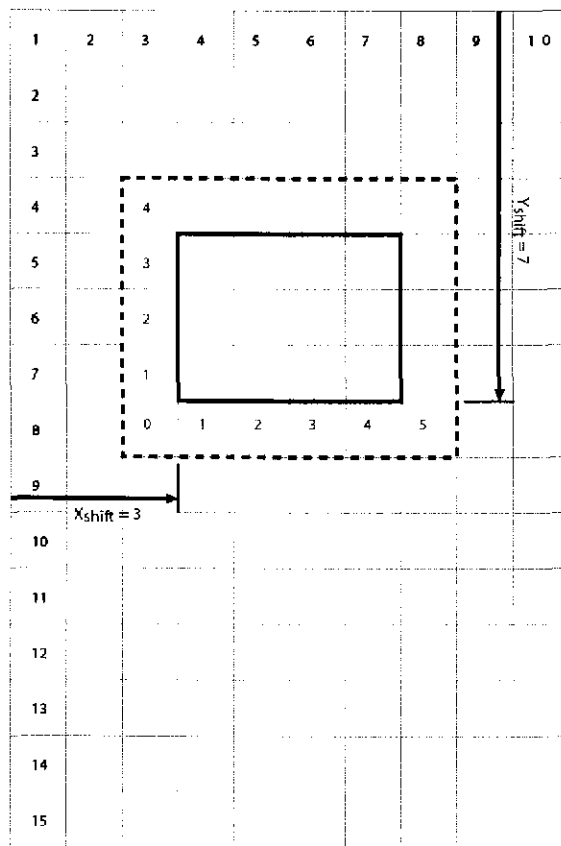


Figure A3-1. Mesh for VTRAN2 verification

Table A3-5. VTRAN2_TEST_1.CMD

```
* author
Joseph F. Kanney
* date
2003.09.24
* title
vtran2 test 1
* input format type
(568e16.8)
* iscrn > 0 will print to screen, otherwise to dbg file
0
* ncol nrow
10 15
* jshftx ishfty ncx ncy
3 7 4 3
* dx dy dz (3e10.3)
1.000E+00 2.000E+00 2.000E+00
```


VTRAN2 command file. In this way, we can visually inspect the velocity file to verify that the correct row and column indices are extracted.

The command file (VTRAN2_TEST_2.CMD), and the budget file (VTRAN2_TEST_2.BUD) are shown in Table A3-9 and Table A3-10, respectively. Running VTRAN2 with these input files produces the ASCII velocity file (VTRAN2_TEST_2.VEL) and the diagnostic file (VTRAN2_TEST_2.DBG) shown in Table A3-11 and Table A3-12, respectively. We note from the velocity component values that the correct translation of indices between the two meshes has been effected.

Table A3-9. VTRAN2_TEST_2.CMD

```
* author
Joseph F. Kanney
* date
2003.09.24
* title
vtran2 test 2
* input format type
(568e16.8)
* iscrn > 0 will print to screen, otherwise to dbg file
0
* ncol nrow
10 15
* jshftx ishfty ncx ncy
3 7 4 3
* dx dy dz (3e10.3)
1.000E+00 1.000E+00 1.000E+00
```

Table A3-10. VTRAN2_TEST_2.BUD

```
0.10100000E+01 0.10200000E+01 0.10300000E+01 0.10400000E+01 0.10500000E+01 0.10599999E+01 0.10700001E+01 0.10800000E+01 0.10900000E+01 0.11000000E+01
0.20100000E+01 0.20200000E+01 0.20300000E+01 0.20400000E+01 0.20500000E+01 0.20599999E+01 0.20699999E+01 0.20799999E+01 0.20899999E+01 0.20999999E+01
0.30100000E+01 0.30200000E+01 0.30300000E+01 0.30400000E+01 0.30500000E+01 0.30599999E+01 0.30699999E+01 0.30799999E+01 0.30899999E+01 0.30999999E+01
0.40100000E+01 0.40200000E+01 0.40300000E+01 0.40400000E+01 0.40500000E+01 0.40599999E+01 0.40700002E+01 0.40799999E+01 0.40900002E+01 0.40999999E+01
0.50100000E+01 0.50200000E+01 0.50300000E+01 0.50400000E+01 0.50500000E+01 0.50599999E+01 0.50700002E+01 0.50799999E+01 0.50900002E+01 0.50999999E+01
0.60100000E+01 0.60200000E+01 0.60300000E+01 0.60400000E+01 0.60500000E+01 0.60599999E+01 0.60700002E+01 0.60799999E+01 0.60900002E+01 0.60999999E+01
0.70100000E+01 0.70200000E+01 0.70300000E+01 0.70400000E+01 0.70500000E+01 0.70599999E+01 0.70700002E+01 0.70799999E+01 0.70900002E+01 0.70999999E+01
0.80100000E+01 0.80200000E+01 0.80299997E+01 0.80400000E+01 0.80500000E+01 0.80600004E+01 0.80699997E+01 0.80799999E+01 0.80900002E+01 0.81000004E+01
0.90100000E+01 0.90200000E+01 0.90299997E+01 0.90400000E+01 0.90500000E+01 0.90600004E+01 0.90699997E+01 0.90799999E+01 0.90900002E+01 0.91000004E+01
0.10010000E+02 0.10020000E+02 0.10030000E+02 0.10040000E+02 0.10050000E+02 0.10060000E+02 0.10070000E+02 0.10080000E+02 0.10090000E+02 0.10100000E+02
0.11010000E+02 0.11020000E+02 0.11030000E+02 0.11040000E+02 0.11050000E+02 0.11060000E+02 0.11070000E+02 0.11080000E+02 0.11090000E+02 0.11100000E+02
0.12010000E+02 0.12020000E+02 0.12030000E+02 0.12040000E+02 0.12050000E+02 0.12060000E+02 0.12070000E+02 0.12080000E+02 0.12090000E+02 0.12100000E+02
0.13010000E+02 0.13020000E+02 0.13030000E+02 0.13040000E+02 0.13050000E+02 0.13060000E+02 0.13070000E+02 0.13080000E+02 0.13090000E+02 0.13100000E+02
0.14010000E+02 0.14020000E+02 0.14030000E+02 0.14040000E+02 0.14050000E+02 0.14060000E+02 0.14070000E+02 0.14080000E+02 0.14090000E+02 0.14100000E+02
0.15010000E+02 0.15020000E+02 0.15030000E+02 0.15040000E+02 0.15050000E+02 0.15060000E+02 0.15070000E+02 0.15080000E+02 0.15090000E+02 0.15100000E+02
0.10100000E+01 0.10200000E+01 0.10300000E+01 0.10400000E+01 0.10500000E+01 0.10599999E+01 0.10700001E+01 0.10800000E+01 0.10900000E+01 0.11000000E+01
0.20100000E+01 0.20200000E+01 0.20300000E+01 0.20400000E+01 0.20500000E+01 0.20599999E+01 0.20699999E+01 0.20799999E+01 0.20899999E+01 0.20999999E+01
0.30100000E+01 0.30200000E+01 0.30300000E+01 0.30400000E+01 0.30500000E+01 0.30599999E+01 0.30699999E+01 0.30799999E+01 0.30899999E+01 0.30999999E+01
0.40100000E+01 0.40200000E+01 0.40300000E+01 0.40400000E+01 0.40500000E+01 0.40599999E+01 0.40700002E+01 0.40799999E+01 0.40900002E+01 0.40999999E+01
0.50100000E+01 0.50200000E+01 0.50300000E+01 0.50400000E+01 0.50500000E+01 0.50599999E+01 0.50700002E+01 0.50799999E+01 0.50900002E+01 0.50999999E+01
0.60100000E+01 0.60200000E+01 0.60300000E+01 0.60400000E+01 0.60500000E+01 0.60599999E+01 0.60700002E+01 0.60799999E+01 0.60900002E+01 0.60999999E+01
0.70100000E+01 0.70200000E+01 0.70300000E+01 0.70400000E+01 0.70500000E+01 0.70599999E+01 0.70700002E+01 0.70799999E+01 0.70900002E+01 0.70999999E+01
0.80100000E+01 0.80200000E+01 0.80299997E+01 0.80400000E+01 0.80500000E+01 0.80600004E+01 0.80699997E+01 0.80799999E+01 0.80900002E+01 0.81000004E+01
0.90100000E+01 0.90200000E+01 0.90299997E+01 0.90400000E+01 0.90500000E+01 0.90600004E+01 0.90699997E+01 0.90799999E+01 0.90900002E+01 0.91000004E+01
0.10010000E+02 0.10020000E+02 0.10030000E+02 0.10040000E+02 0.10050000E+02 0.10060000E+02 0.10070000E+02 0.10080000E+02 0.10090000E+02 0.10100000E+02
0.11010000E+02 0.11020000E+02 0.11030000E+02 0.11040000E+02 0.11050000E+02 0.11060000E+02 0.11070000E+02 0.11080000E+02 0.11090000E+02 0.11100000E+02
0.12010000E+02 0.12020000E+02 0.12030000E+02 0.12040000E+02 0.12050000E+02 0.12060000E+02 0.12070000E+02 0.12080000E+02 0.12090000E+02 0.12100000E+02
0.13010000E+02 0.13020000E+02 0.13030000E+02 0.13040000E+02 0.13050000E+02 0.13060000E+02 0.13070000E+02 0.13080000E+02 0.13090000E+02 0.13100000E+02
0.14010000E+02 0.14020000E+02 0.14030000E+02 0.14040000E+02 0.14050000E+02 0.14060000E+02 0.14070000E+02 0.14080000E+02 0.14090000E+02 0.14100000E+02
0.15010000E+02 0.15020000E+02 0.15030000E+02 0.15040000E+02 0.15050000E+02 0.15060000E+02 0.15070000E+02 0.15080000E+02 0.15090000E+02 0.15100000E+02
```

Table A3-11. VTRAN2_TEST_2.VEL

| | | | | | | |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|--|
| 4 | 3 | 0.00000000E+00 | | | | |
| 0.00000000E+00 | 0.80299997E+01 | 0.80400000E+01 | 0.80500002E+01 | 0.80600004E+01 | 0.80699997E+01 | |
| 0.00000000E+00 | 0.70300002E+01 | 0.70400000E+01 | 0.70500002E+01 | 0.70599999E+01 | 0.70700002E+01 | |
| 0.00000000E+00 | 0.60300002E+01 | 0.60400000E+01 | 0.60500002E+01 | 0.60599999E+01 | 0.60700002E+01 | |
| 0.00000000E+00 | 0.50300002E+01 | 0.50400000E+01 | 0.50500002E+01 | 0.50599999E+01 | 0.50700002E+01 | |
| 0.00000000E+00 | 0.40300002E+01 | 0.40400000E+01 | 0.40500002E+01 | 0.40599999E+01 | 0.40700002E+01 | |
| 0.00000000E+00 | 0.00000000E+00 | 0.00000000E+00 | 0.00000000E+00 | 0.00000000E+00 | 0.00000000E+00 | |
| -0.70300002E+01 | -0.70400000E+01 | -0.70500002E+01 | -0.70599999E+01 | -0.70700002E+01 | -0.70799999E+01 | |
| -0.60300002E+01 | -0.60400000E+01 | -0.60500002E+01 | -0.60599999E+01 | -0.60700002E+01 | -0.60799999E+01 | |
| -0.50300002E+01 | -0.50400000E+01 | -0.50500002E+01 | -0.50599999E+01 | -0.50700002E+01 | -0.50799999E+01 | |
| -0.40300002E+01 | -0.40400000E+01 | -0.40500002E+01 | -0.40599999E+01 | -0.40700002E+01 | -0.40799999E+01 | |

Table A3-12. VTRAN2_TEST_2.DBG

```

command file           = VTRAN2_TEST_2.CMD
budget file           = VTRAN2_TEST_2.BUD
(binary) velocity transfer file = VTRAN2_TEST_2.TRN
diagnostic/debug file = VTRAN2_TEST_2.DBG
(ascii) velocity output file = VTRAN2_TEST_2.VEL
author   = Joseph F. Kanney
date     = 2003.09.24
title    = vtran2 test 2
format   = (568e16.8)
iscrn    = 0
ncolmf   = 10
nrowmf   = 15
jshftx   = 3
ishfty   = 7
ncx      = 4
ncy      = 3
dx       = 0.1000E+01
dy       = 0.1000E+01
dz       = 0.1000E+01
Using input format ( 1) = (568e16.8)
VTRAN >> Reading budget file
VTRAN >> Processing velocities
VTRAN >> Writing velocity transfer file
VTRAN >> Writing ascii velocity output file
VTRAN >> Cleaning up
VTRAN >> Normal Completion
    
```

A4 Script Source Listing

The scripts listed in this appendix neither model physical phenomena nor solve differential equations that model physical phenomena. Rather they are utility codes that process inputs and outputs for other modeling codes (i.e., MODFLOW, DTRKMF, and SECOTP2D).

A4.1 Python script convert_shapefile_to_ASCII.py

A4.1.1 Input File

```
1 Measured_Minable_Ore_UTM_NAD27
  polyline_dump_matlab_part .dat
3 polyline_dump_for_guuplot.dat
```

A4.1.2 Script

```
1 # Copyright (c) 2008
2 # Zach Steindler (steiza@gmail.com, http://code.google.com/p/pyshapefile/)
3 #
4 # All rights reserved.
5 #
6 # Redistribution and use in source and binary forms, with or without modification,
7 # are permitted provided that the following conditions are met:
8 #
9 # Redistributions of source code must retain the above copyright notice,
10 # this list of conditions and the following disclaimer.
11 #
12 # Redistributions in binary form must reproduce the above copyright notice,
13 # this list of conditions and the following disclaimer in the documentation
14 # and/or other materials provided with the distribution.
15 #
16 # Neither the name of the original author nor the names of contributors
17 # may be used to endorse or promote products derived from this software
18 # without specific prior written permission.
19 #
20 # THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
21 # AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
22 # IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
23 # ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
24 # LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
25 # DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
26 # SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
27 # CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
28 # OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
29 # OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
30
31 import struct
32 _POINT_PAIR_SIZE = 16
33
34 class ShapefileData(object):
35     """A python class for loading shapefiles."""
36
37     def __init__(self, base_filename):
38         """Initializes ShapefileData with path to shapefile data.
39
40         Args:
41             base_filename: Base path to shapefile data. base_filename + '.shp'
42                           and base_filename + '.dbf' should exist.
43         """
44         self.base_filename = base_filename
45
46     def load_bounding_box(self):
47         """Loads boundary information from .shp file"""
48         shp_fd = open('%s.shp' % self.base_filename, 'rb')
49         shp_fd.seek(36)
50
51         self.file_xmin, self.file_ymin, self.file_xmax, self.file_ymax = \
52             struct.unpack('dddd', shp_fd.read(32))
53
54         shp_fd.close()
55
56     def load_objects(self, x = 0, y = 0, radius = 0):
57         """Loads information from .shp file; only loads information within
58         bounds if specified.
59
60         Args:
61             x: x coordinate at center of bounding box
62             y: y coordinate at center of bounding box
63             radius: size of bounding box
64         """
65         self.load_bounding_box()
66         self.items = {}
67
68         if x == 0 or y == 0 or radius == 0:
69             self.data_xmin = self.file_xmin
70             self.data_xmax = self.file_xmax
71             self.data_ymin = self.file_ymin
72             self.data_ymax = self.file_ymax
73         else:
74             self.data_xmin = x - radius
75             self.data_xmax = x + radius
```

```

77     self.data_ymin = y - radius
78     self.data_ymax = y + radius
79
80 shp_fd = open('%s.shp' % self.base_filename, 'rb')
81 shp_fd.seek(100)
82
83 while True:
84     # Process record header
85     record_header = shp_fd.read(12)
86     if record_header == '':
87         break
88
89     record_number, record_length = struct.unpack('>LL',
90         record_header[:8])
91
92     record_shape = struct.unpack('<L', record_header[8:])[0]
93
94     # Process record body
95     if record_shape == 0:
96         # Record is null
97         pass
98
99     elif record_shape == 1:
100        # Record is a point
101        object_point = shp_fd.read(_POINT_PAIR_SIZE)
102        x, y = struct.unpack('dd', object_point)
103
104        if self.data_xmin > x or self.data_xmax < x:
105            continue
106        if self.data_ymin > y or self.data_ymax < y:
107            continue
108
109        self.items[record_number] = {'type': 'point',
110            'point': {'x': x, 'y': y}}
111
112     elif record_shape == 3 or record_shape == 5:
113        # Record is a polyline
114        object_header = shp_fd.read(40)
115        object_xmin, object_ymin, object_xmax, object_ymax, \
116            nparts, points = struct.unpack('dddidi', object_header)
117
118        if object_xmax < self.data_xmin or \
119            object_xmin > self.data_xmax or \
120            object_ymax < self.data_ymin or \
121            object_ymin > self.data_ymax:
122
123            # added partial handling of parts (KK 09/09)
124            part_list = shp_fd.read((4 * nparts) +
125                (_POINT_PAIR_SIZE * points))
126            print 'TODO: learn to handle this case?! (not needed for WIPP data)'
127
128            continue
129
130        # read in list of indicies to first point in each part (KK 09/09)
131        part_list = shp_fd.read((4 * nparts))
132        self.parts = list(struct.unpack('i'*nparts, part_list))
133
134        self.items[record_number] = {'type': 'polyline', 'points': []}
135
136        object_points = shp_fd.read(_POINT_PAIR_SIZE * points)
137        for i in xrange(0, len(object_points), _POINT_PAIR_SIZE):
138            x, y = struct.unpack('dd', object_points[i:_POINT_PAIR_SIZE])
139            self.items[record_number]['points'].append({'x': x, 'y': y})
140
141     else:
142         raise UnknownShape
143 shp_fd.close()
144
145 def dump_polyline_to_file(self, rec_num=1, fn='polyline_dump_for_gnuplot.dat'):
146     """save each part of the polyline as a group, separated
147     by two blank lines. Intended for plotting in Gnuplot.
148     Added by Kris Kuhlman, Sept 2009"""
149
150     fh = open(fn, 'w')
151     # save filename in a comment at top of file
152     fh.write('# data from %s.shp\n'%self.base_filename)
153
154     # number of parts
155     npar = len(self.parts)
156
157     # number of points (end of last part)
158     npts = len(self.items[rec_num]['points'])
159
160     self.parts.append(npts)
161
162     for i in xrange(npar):
163         fh.write('# part: %i\n%i\n'%i)
164         p = self.items[rec_num]['points'][self.parts[i]:self.parts[i+1]]
165         for pt in p:
166             # save data with 10cm precision (good enough for plotting)

```



```

167         # make file a bit smaller and quicker to parse by GnuPlot
168         fh.write('%1f %1f\n'%(pt['x'],pt['y']))
169         fh.write('\n\n')
170
171     fh.close()
172
173     def dump_polyline_to_files(self,rec_num=1,fn=['polyline_dump_matlab_part','.dat']):
174         """save each part of the polyline as a file, Intended for
175         importing into Matlab. Added by Kris Kuhlman, Sept 2009"""
176
177         # number of parts
178         npar = len(self.parts)
179
180         # number of points (end of last part)
181         npts = len(self.items[rec_num]['points'])
182
183         self.parts.append(npts)
184
185         for i in xrange(npar):
186             # save file with index between prefix and suffix given as arguments
187             fh = open('%s%4.4i%s'%(fn[0],i,fn[1]),'w')
188             p = self.items[rec_num]['points'][self.parts[i]:self.parts[i+1]]
189             for pt in p:
190                 # write points with 0.1cm precision (UTM is in meters)
191                 fh.write('%1f %1f\n'%(pt['x'],pt['y']))
192             fh.close()
193
194 # following added by KK 09/09
195 if __name__ == '__main__':
196
197     from sys import argv
198
199     # this script just drives the pyshapefile Python library
200     # to open (then dump to ASCII) the potash mining definitions shapefile
201     # read an input file with same basename but '.in' instead of '.py'
202
203     fin = open(argv[0].replace('.py','.in'),'r')
204     shpfn = fin.readline().strip()
205     matfn = fin.readline().strip().split()
206     gnufn = fin.readline().strip()
207     fin.close()
208
209     print '#####'
210     print 'shpfn:',shpfn
211     print 'matfn:',matfn
212     print 'gnufn:',gnufn
213     print '#####'
214
215     shp = ShapefileData(shpfn)
216
217     # read in shapefile polyline data
218     shp.load_objects()
219
220     # dump polylines to a single text file for plotting in GnuPlot
221     shp.dump_polyline_to_file(fn=gnufn)
222
223     # dump each part of polyline to a single file
224     # for reading / processing in Matlab
225
226     shp.dump_polyline_to_files(fn=matfn)

```

A4.2 MATLAB script import_polyline_determine_mined_areas.m

```

1  % this MATLAB script reads in the ASCII parts of the polyline defined
2  % in the BLM mining areas shapefile, determining whether each
3  % cell in the model domain is inside the mining region an
4  % integer matrix indicating whether the cell is mined is the result.
5
6  % NB: this script takes about 2 hours to run on my desktop
7
8  prefix = 'polyline_dump_matlab_part';
9  suffix = '.dat';
10 numpart = 54;
11
12 % use cells (rather than more common arrays) to allow variable-length parts
13 p = cell(numpart,2);
14
15 for i=1:numpart
16     p(i,1) = {load([prefix,sprintf('%4.4i',i-1),suffix])};
17 end
18
19 % first cell is the main polygon; all other cells will be checked with
20 % reference to that one. 0=outside, 1=inside
21
22 % since no polylines intersect, just check to see if the first point of a
23 % given polyline is inside or outside the big polyline.
24
25 p(1,2) = {-1};
26
27 % plot individual parts of polyline as areas (patches) for checking
28 patch(p{1,1}(:,1),p{1,1}(:,2),'k')
29 hold on;
30
31 for i=2:numpart
32     p(i,2) = {inpolygon(p{i,1}(1,1),p{i,1}(1,2), ...
33         p{1,1}(:,1):p{1,1}(:,2))};
34     if p{i,2} == 1
35         patch(p{i,1}(:,1),p{i,1}(:,2),'r')
36     else
37         patch(p{i,1}(:,1),p{i,1}(:,2),'g')
38     end
39 end
40
41 daspect([1,1,1])
42 xlabel('UTM NAD27 X (m)')
43 ylabel('UTM NAD27 Y (m)')
44
45 % model grid
46 nrow = 307;
47 ncol = 284;
48 x0 = 601700.0;
49 x1 = 630000.0;
50 dx = 100.0;
51 y0 = 3586500.0;
52 y1 = 3597100.0;
53 dy = 100.0;
54
55 [X,Y] = meshgrid(x0:dx:x1,y0:dy:y1);
56 result = zeros(nrow,ncol,numpart);
57
58 for i=numpart:-1:1
59     display(i);
60     tic
61     result(:,:,i) = inpolygon(X,Y,p{i,1}(:,1),p{i,1}(:,2));
62     toc
63 end
64
65 % this assumes that the only polygon that surrounds any other polygons
66 % is the "big" one that is the first entry. None of the other polygons
67 % intersect or surround one another.
68
69 % black          red          green
70 % big-polygon - nested-polygons + external-polygon
71
72 final = result(:,:,1) - sum(result(:,:, [p{: ,2}] == 1),3) + ...
73     sum(result(:,:, [p{: ,2}] == 0),3);
74
75 % plot for checking
76 figure()
77 spy(final)
78
79 save('mining_areas_matrix.dat','-ascii','final')
80 out = [reshape(X,nrow*ncol,1),reshape(Y,nrow*ncol,1), ...
81     reshape(final,nrow*ncol,1)];
82
83 save('mining_areas_xyz.dat','-ascii','out')
    
```

A4.3 Bash shell script run_mining_mods.sh

```

1 #!/bin/bash
3 # this script assumes the following directory substructure
# created in ReadScript.py, where CWP='cwd'
5 #
# CWP/Inputs
7 # CWP/Inputs/bin (MF2K & DTRKMF binaries)
# CWP/Inputs/API14.Task7
9 # CWP/Inputs/API14.Task7/data (MF2K data files)
# CWP/Inputs/API14.Task7/modflow (MF2K package files)
11 # CWP/Inputs/API14.Task7/Outputs (modcled_{K,A,R,S}_field.mod)
# CWP/Outputs/ (results of this analysis)
13 # CWP/RunControl
#
15 verbose=True
17 # the script is started from the "CWP" directory
19 echo '## run_mining_mods.sh starting directory: ' `pwd` '##'
21 # modify the path of "updated" T-fields, so they are all at the
23 # same level in the directory structure (simplifying scripts elsewhere)
25 if [ ${verbose} == True ]; then
27     fi
29     if [ -a ./Inputs/keepers_short ]; then
31         # delete any pre-existing files here,
# since file is concatenated to in next loop
33     fi
35     for d in `cat ./Inputs/keepers`; do
37         bn=`basename ${d}`
# test whether it is a compound path
# if compound, move up a directory
39         if [ ${d} != ${bn} ]; then
41             mv ./Inputs/API14.Task7/Outputs/${d}/ ./Inputs/API14.Task7/Outputs/
43         # create a new 'keepers' list without Update or Update2 directories
45     done
47     if [ ${verbose} == True ]; then
49         fi
51     cd Outputs
53     # make links of mining mod input files and scripts into
# the current Outputs working directory
55     for f in `cat ../Inputs/files_minmod`; do
57         ln -s ../Inputs/${f} ./${f}
59     # python script creates partial/full mining factor
# arrays from GIS definition of mining areas
61     # extending area to account for 45-degree angle of repose
python onlarge-mining_areas.py
63
65     # python script creates 600 mining-modified T fields
# and 2 mining-modified MODFLOW IBOUND arrays
# this script creates directory hierarchy in Outputs/
python mining_mod.py
67
69     # -----
# run 100x100 m grid MODFLOW2K and DTRKMF
71
73     if [ ${verbose} == True ]; then
75         fi
77         # MODFLOW name file using direct solver (use when LMG solver fails)
sed -e "s/LMG 8 mf2k_culebra.lmg/DE4 8 mf2k_culebra.de4/" \
<../Inputs/API14.Task7/modflow/mf2k_head.nam \
79         >../Outputs/modflow/mf2k_head.de4.nam
81     # adjust MODFLOW LMG solver tolerance and turn on PCG pre-conditioner
sed -c "s/3\0 2\2 5\4 0/3.0 2.2 5.4 1/" \
83         -e "s/2 50 1\0E-08 1.0 1/2 50 1.0E-10 1.0 1/" \
<../Inputs/API14.Task7/modflow/mf2k_culebra.lmg \
85         >../Outputs/modflow/mf2k_culebra.lmg
87     # sed doesn't modify files in-place
mv ../Outputs/modflow/mf2k_culebra.lmg \

```

```

89  ./Inputs/API14_Task7/modflow/mf2k-culebra.lmg
91  # creates links for common files into each of 600 mining-modified
92  # T fields and runs MODFLOW and DTRKMF (this is 100x100 m grid)
93  ./link_input_run_mf_dtrk.sh
94
95  # -----
96  if [ ${verbose} == True ]; then
97    echo "## extract DTRKMF output ##"
98  fi
99
100 # combine all 100 realizations of results for each
101 # mining scenario into a single file for easier plotting
102 # travel times are converted to 4m thickness
103
104 python combine_dtrkmf_output_for_gnuplot.py
105
106 # extract time required to exit LWB from dtrkmf results
107 # also sort results for plotting of travel-time CDFs
108 # travel times here are converted to 4m thickness
109
110 python extract_dtrkmf_lwb_travel_times.py
111
112 # -----
113 # convert 100x100 m grid to 50x50 m grid
114
115 if [ ${verbose} == True ]; then
116   echo "## convert MODFLOW2K inputs to 50x50m grid ##"
117 fi
118
119 # create spatially-variable model property files
120 # for 50x50m grid from 100x100m grid
121 python 100x100_to_50x50.py
122
123 # create other 50x50 MODFLOW input files from 100x100 files
124 # using find/replace in sed
125
126 if [ ${verbose} == True ]; then
127   echo "## create MF2K inputs for 50x50m grid ##"
128 fi
129
130 # change MODFLOW name file to point to expanded input files
131 # s/find/replace/ is a sed regexp, '.' in the left hand side
132 # must be escaped with a backslash, or it means "any character"
133 # a trailing "g" means apply globally (otherwise finds one occurrence)
134 sed -e "s/modeled_head\.lst/modeled_head_50.lst/" \
135     -e "s/modeled_head\.bin/modeled_head_50.bin/" \
136     -e "s/modeled_flow\.bud/modeled_flow_50.bud/" \
137     -e "s/\.mod\.mod.50/g" \
138     -e "s/\.inf\.inf.50/" \
139     -e "s/mf2k_head\.dis/mf2k_head_50.dis/" \
140     <../Inputs/API14_Task7/modflow/mf2k.head.nam \
141     >../Outputs/modflow/mf2k.head.50.nam
142
143 # input file for direct solver
144 # this is used when LMG doesn't work, which is ~1% of the time
145 sed -e "s/LMG      8 mf2k_culebra\.lmg/DE4      8 mf2k_culebra.de4/" \
146     <../Outputs/modflow/mf2k.head.50.nam \
147     >../Outputs/modflow/mf2k.head.50.de4.nam
148
149 # change MODFLOW discretization file to new # rows & cols
150 sed -e "s/307 284/614 568/" \
151     <../Inputs/API14_Task7/modflow/mf2k.head.dis \
152     >../Outputs/modflow/mf2k.head.50.dis
153
154 # -----
155 # run MODFLOW on refined grid to create budget files
156 # for SECOTP2D, located inside each subdirectory
157
158 if [ ${verbose} == True ]; then
159   echo "## run MODFLOW2K on 50x50 m grid ##"
160 fi
161
162 ./run_50x50_modflow.sh
163
164 # -----
165 # convert budget files to ASCII and copy resulting ASCII files
166 # (all with same name, but in different directories) into a
167 # single directory, renaming files appropriately
168
169 if [ ${verbose} == True ]; then
170   echo "## convert binary budgets to ASCII and move VTRAN2 input files to single location ##"
171 fi
172
173 ./convert_rename_modflow_50x50_budget.sh
174
175 # the files saved into this directory are inputs to VTRAN2,
176 # which prepares these files for use by SECOTP2D
177
178 if [ ${verbose} == True ]; then
179   echo "## end of 'run_mining_mods.sh' Bash shell script ##"
180 fi

```

A4.4 Python script enlarge_mining_areas.py

A4.4.1 Input File

```

601700.0 3566500.0
2 284 307
100.0 100.0
4 616941.0 610495.0 610567.0 617015.0 616941.0
3585109.0 3585068.0 3578623.0 3578681.0 3585109.0
6 mining_areas.xyz.dat
mining_factor.dat
8 New_Full_09.dat
New_Part_09.dat
10 True
    
```

A4.4.2 Script

```

# this script takes the output of ArcGIS and creates the full and partial mining
2 # factor files that are used in the mining modification calculation.

4 from math import floor
from itertools import izip
6 from sys import exit, argv

8 #####
# read in input data
10 try:
    # input name has same name as script ending with ".in", instead of ".py"
12     scrfname = argv[0]
    inpfname = scrfname.replace('.py', '.in')
14     fin = open(inpfname, 'r')
except IOError:
16     print 'ERROR: %s input file "%s" not found.' %(scrfname, inpfname)
    exit(1)
18
x0,y0 = [float(x) for x in fin.readline().strip().split()] # UTM NAD27 (m) modflow grid origin
20 nx,ny = [int(x) for x in fin.readline().strip().split()] # number of cells
dx,dy = [float(x) for x in fin.readline().strip().split()] # cell size (meters)
22
wipp = []
24 wipp.append([float(x) for x in fin.readline().rstrip().split()]) # x-coordinate (closed CCW loop)
wipp.append([float(x) for x in fin.readline().rstrip().split()]) # y
26 wipp = zip(*wipp)

28 # file with GIS mining data
30 fnminingin = fin.readline().strip()
# file for saving mining factor into matrix form for debugging
32 fnminifactout = fin.readline().strip()
# files for full and partial mining indices (output for other scripts)
34 fnfullminout = fin.readline().strip()
fnpartminout = fin.readline().strip()
36

# print status/debugging to screen?
38 verbose = fin.readline().strip()
fin.close()
40
verbose = verbose[0] == 'T' or verbose[0] == 't'
42
if verbose:
44     print 'echo input'
    print '=====',
46     print "x0,y0",x0,y0
    print "nx,ny",nx,ny
48     print "dx,dy",dx,dy
    print "wipp",wipp
50     print "fnminingin",fnminingin
    print "fnminifactout",fnminifactout
52     print "fnfullminout",fnfullminout
    print "fnpartminout",fnpartminout
54     print '-----',

56 #####
58 ncl = ny*nx
60 #####
62 # define functions

64 def wipp-inside(x,y):
    """ given a line segment between points P0=(x0,y0) and P1=(x1,y1)
66     a third point P=(x,y) has the following relationship to the line segment.
    f = (y-y0)(x1-x0) - (x-x0)(y1-y0)
68     if f<0 then P is to the right of the line,
    if f>0 it is on the left (=0 is on the line)
70     if a point is on the left of all CCW line segments defining
    a convex polygon (such as the wipp boundary), then it is inside the polygon
72     http://ozviz.wasp.uwa.edu.au/~pbourke/geometry/insidepoly/"""
74     f = True
    
```

```

76     for (p0,p1) in zip(wipp[0:-1],wipp[1:]):
77         # to be inside the following must be > 0 for all 4 sides
78         if (y-p0[1])*(p1[0]-p0[0])-(x-p0[0])*(p1[1]-p0[1]) <= 0.0:
79             f = False
80             break
81     return f

82 def matsave(filename,m):
83     """Writes file as a list of lists as a 2D array (passed in as integers)
84     Outer list is rows, inner lists are columns (C-major order)."""
85     f = open(filename,'w')
86     for row in m:
87         f.write(' '.join(['%2i'%val for val in row]) +'\n')
88     f.close()

89 def minsave(filename,m):
90     """Save mining matrices, taking the 3 cells of padding into account"""
91     f = open(filename,'w')
92     for row in m[3:-3]: # leave off first and last 3 rows
93         # leave off first and last 3 columns of each row
94         f.write(' '.join(['%2i'%val for val in row[3:-3]]) +'\n')
95     f.close()

96 def reshapev2m(v,order='C'):
97     """Reshape a vector that was previously reshaped in C-style order from a matrix,
98     back into a matrix (here a list of lists)."""
99     if order=='C':
100         # row-major order (MODEFLOW, C, Python/numpy)
101         m = [None]*ny
102         for i,(lo,hi) in enumerate(izip(xrange(0, nel-nx+1, nx), xrange(nx, nel+1, nx))):
103             m[i] = v[lo:hi]
104         return m
105     elif order=='F':
106         # column-major order (Fortran, Matlab)
107         m = [None]*nx
108         for i,(lo,hi) in enumerate(izip(xrange(0, nel-ny+1, ny), xrange(ny, nel+1, ny))):
109             m[i] = v[lo:hi]
110         # transpose rows/columns and flip in y-direction
111         return zip(*m)[::-1]
112
113 #####
114
115 # read in definition of mining on 100x100 grid
116 if verbose:
117     print 'reading GIS data:',fminingin
118
119 fin = open(fminingin,'r')
120 m = []
121 for line in fin:
122     # data are "X Y index" (index=1.0 for potash mining, index=0.0 for outside potash)
123     m.append([float(x) for x in line.strip().split()])
124     m[-1][2] = 10*int(m[-1][2]) # convert to integer and make index (0,10)
125
126 if verbose:
127     print len(m),'lines of data read;',len(m[0]),'columns'
128     # save coordinates of points inside LWB for checking algorithm
129
130 count = 0
131
132 # check if inside LWB
133 for pt in m:
134     if wipp_inside(pt[0],pt[1]):
135         # add one
136         pt[2] += 1
137         count += 1
138
139 if verbose:
140     print count,'cells inside LWB, out of',len(m)
141
142 min1 = reshapev2m(zip(*m)[2],order='F')
143
144 # save un-expanded mining areas in matrix format for comparison
145 if verbose:
146     print 'saving matrix indicating areas with ore to',fminifactout
147     matsave(fminifactout,min1)
148     matsave(fminifactout+'x',reshapev2m(zip(*m)[0],order='F'))
149     matsave(fminifactout+'y',reshapev2m(zip(*m)[1],order='F'))
150
151 # 0 = not mined, outside LWB
152 # 1 = not mined, inside LWB
153 # 10 = mined, outside LWB
154 # 11 = mined, inside LWB
155
156 # pad the target array (list of lists) to allow easier indexing at boundaries
157 minp = [None]*(ny+6)
158 minf = [None]*(ny+6)
159 for i in xrange(len(minp)):
160     minp[i] = [0]*(nx+6)
161     minf[i] = [0]*(nx+6)

```

```

166 v5 = [1,1,1,1,1]
168 if verbose:
    print 'expand mining'
170 for i in xrange(ny):
    j2 = i+3
172     for j in xrange(nx):
        j2 = j+3
174         # is there mining in this cell?
        # either inside or outside LWB (full mining case)
176         if minl[i][j] > 9:
            # expand the mining using a 5x5 square, centered on this cell
178             minf[i2-2][j2-2:j2+3] = v5
            minf[i2-1][j2-2:j2+3] = v5
180             minf[i2 ][j2-2:j2+3] = v5
            minf[i2+1][j2-2:j2+3] = v5
182             minf[i2+2][j2-2:j2+3] = v5
184             # expand mining an additional cell in compass directions
            minf[i2+3][j2] = 1
186             minf[i2-3][j2] = 1
            minf[i2 ][j2-3] = 1
188             minf[i2 ][j2+3] = 1
190             # is there mining in this cell AND is it outside the LWB?
            # (partial mining case)
192             if minl[i][j] == 10:
                # expand the mining using a 5x5 square
194                 minp[i2-2][j2-2:j2+3] = v5
                 minp[i2-1][j2-2:j2+3] = v5
196                 minp[i2 ][j2-2:j2+3] = v5
                 minp[i2+1][j2-2:j2+3] = v5
198                 minp[i2+2][j2-2:j2+3] = v5
200                 # expand mining an additional cell in compass directions
                 minp[i2+3][j2] = 1
202                 minp[i2-3][j2] = 1
                 minp[i2 ][j2-3] = 1
204                 minp[i2 ][j2+3] = 1
206 if verbose:
    print 'saving full mining factor array:', ffullminout
208 minsave(ffullminout, minf)
210 if verbose:
    print 'saving partial mining factor array:', fpartminout
212 minsave(fpartminout, minp)

```

A4.5 Python script mining_mod.py

A4.5.1 Input File

```

1 284 307
%llc
3 ../Inputs/keepers_short
  New_Full_09.dat
5 New_Part_09.dat
  ../Inputs/AP114.Task7/data/init_bnds.inf
7 init_bnds_full.inf
  init_bnds_part.inf
9 sum1_st2d_pabc09_r1.tbl sum1_st2d_pabc09_r2.tbl sum1_st2d_pabc09_r3.tbl
  ../Inputs/AP114.Task7/Outputs
11 modeled_K_field.mod
  modeled_mined_K_field.mod
13 True
    
```

A4.5.2 Script

```

1 # This script does the actual mining modifications to the T fields and
  # MODFLOW boundary conditions.
3 #
  # INPUTS: 2 integer arrays indicating where full/partial mining occurs
5 #          100 MODFLOW T fields from calibration
  #          1 MODFLOW IBOUND array (specification of active/BC/inactive status)
7 #          3 mining factor lists (100 numbers 1<=x<=1000)
  #
9 # OUTPUTS: 600 mining-modified T fields (100 realizations, 2 mining types, 3 replicates)
  #          2 mining-modified MODFLOW ibound arrays
11 #
  # this script should be run from the ./Outputs directory
13
14 from glob import glob
15 from os import makedirs,remove
16 from os.path import split,exists
17 from itertools import chain
18 from sys import exit,argv
19
20 #####
21 # read in input data
  try:
22     # input name has same name as script ending with ".in", instead of ".py"
23     scrfname = argv[0]
24     inpfname = scrfname.replace('.py','.in')
25     fin = open(inpfname,'r')
26 except IOError:
27     print 'ERROR: %s input file "%s" not found.' %(scrfname,inpfname)
28     exit(1)
29
30 # number of columns, rows in model grid
31 nx,ny = [int(x) for x in fin.readline().rstrip().split()]
32 outfmt = fin.readline().strip() # python format for K fields
33 fnrznlist = fin.readline().strip() # filename of shortened realization list (keepers_short)
34 fnfullminin = fin.readline().strip() # full mining indicator field (input)
35 fnpartminin = fin.readline().strip() # partial mining "" "" (input)
36 fnIBOUNDin = fin.readline().strip() # path to MF2K IBOUND array file (input)
37 fnIBOUNDfout = fin.readline().strip() # path to full-mining-modified IBOUND (out)
38 fnIBOUNDpout = fin.readline().strip() # "" partial "" ""
39 fnminifact = fin.readline().strip().split() # list of 3 filenames for mining factors from VMS
40 pathrzn = fin.readline().strip() # base path to all 100 AP114-Task 7 realizations
41 fnKfieldin = fin.readline().strip() # filename of K (aka T) field (in)
42 fnKfieldout = fin.readline().strip() # filename for mining-modified K field (out)
43 verbose = fin.readline().strip() # True to print status output (False = no)
44 fin.close()
45
46 verbose = verbose[0] == 'T' or verbose[0] == 't'
47
48 if verbose:
49     print 'echo input'
50     print '====='
51     print 'nx,ny',nx,ny
52     print 'outfmt',outfmt
53     print 'fnrznlist',fnrznlist
54     print 'fnfullminin',fnfullminin
55     print 'fnpartminin',fnpartminin
56     print 'fnIBOUNDin',fnIBOUNDin
57     print 'fnIBOUNDfout',fnIBOUNDfout
58     print 'fnIBOUNDpout',fnIBOUNDpout
59     print 'fnminifact',fnminifact
60     print 'pathrzn',pathrzn
61     print 'fnKfieldin',fnKfieldin
62     print 'fnKfieldout',fnKfieldout
63     print '====='
64
65 ncl = nx*ny
66
67 #####
68 # define functions
69 def intload(filename):
    
```



```

71     """Reads file (a 2D integer array) as a list of lists.
72     Outer list is rows, inner lists are columns."""
73     f = open(filename, 'r')
74     m = [[int(v) for v in line.rstrip().split()] for line in f]
75     f.close()
76     return m
77
78 def intsave(filename, m):
79     """Writes file as a list of lists as a 2D integer array, space-separated format '%2i'.
80     Outer list is rows, inner lists are columns."""
81     f = open(filename, 'w')
82     for row in m:
83         f.write(' '.join(['%2i' % col for col in row]) + '\n')
84     f.close()
85
86 def floatload(filename):
87     """Reads file (a list of real numbers, one number each row) into a list of floats."""
88     f = open(filename, 'r')
89     m = [float(line.rstrip()) for line in f]
90     f.close()
91     return m
92
93 def floatsave(filename, m):
94     """Writes file as a list of floats, one value per row, format '%.11e'."""
95     f = open(filename, 'w')
96     f.write('\n'.join([outfmt % num for num in m]) + '\n')
97     f.close()
98
99 def debugsave(filename, m, n):
100     f = open(filename, 'w')
101     for val, flag in zip(m, n):
102         f.write(outfmt % val + ' %5i' % flag + '\n')
103     f.close()
104
105 def reshape2v(m):
106     """Reshapes a rectangular matrix into a vector in same fashion as numpy.reshape().
107     which is C-major order"""
108     return list(chain(*m))
109
110 #####
111 if verbose:
112     print 'read files'
113 # read in mining definitions and modflow ibound array (all 2D integer matrices)
114 full = intload(fnfullminin)
115 part = intload(fnpartminin)
116 ibound = intload(fnIBOUNDin)
117
118 iboundf = []
119 iboundp = []
120
121 if verbose:
122     print 'adjust BC'
123 # turn off constant head nodes where mining modification happens
124 for j in xrange(ny):
125     rowf = []
126     rowp = []
127     for i in xrange(nx):
128         # 1 in mining file indicates mining
129         # -1 in ibound array indicates constant head
130         # new ibound array for full mining
131         if full[j][i] == 1 and ibound[j][i] == -1:
132             rowf.append(1)
133         else:
134             rowf.append(ibound[j][i])
135
136         # new ibound array for partial mining
137         if part[j][i] == 1 and ibound[j][i] == -1:
138             rowp.append(1)
139         else:
140             rowp.append(ibound[j][i])
141
142     iboundf.append(rowf)
143     iboundp.append(rowp)
144
145 # make the northeast far row/column is still specified head at the outermost row/col
146 # these are columns 99:284 in row 1 and all rows in rightmost column for
147 # CRA 2009 PABC MODFLOW model
148 for i in xrange(98, nx):
149     iboundf[0][i] = -1
150     iboundp[0][i] = -1
151
152 for j in xrange(ny):
153     iboundf[j][-1] = -1
154     iboundp[j][-1] = -1
155
156 # make sure south row is all constant head
157 for j in xrange(nx):

```

```

150     iboundf[-1][j] = -1
151     iboundp[-1][j] = -1
161     # make sure south part of western boundary (row 159:307) is all constant head
163     for j in xrange(159,ny):
164         iboundf[j][0] = -1
165         iboundp[j][0] = -1
167     # create similar subdirectories in the Outputs subdir for modified input
168     # files, rather than just putting them back in the Inputs directory
169     # with the unmodified files checked out of CVS
170     for dir in ['./Outputs/data', './Outputs/modflow']:
171         if not exists(dir):
172             makedirs(dir)
173
174     # save two mining-modified MODFLOW IBOUND arrays to disk
175     intsave('./Outputs/data/'+fnIBOUNDfout,iboundf)
176     intsave('./Outputs/data/'+fnIBOUNDpout,iboundp)
177
178     if verbose:
179         print 'reshape mining'
180         # reshape 2D matrix of integer mining indicators to
181         # vector same shape as T fields
182         full = reshaped2v(full)
183         part = reshaped2v(part)
184         min = [full,part]
185
186     if verbose:
187         print 'read mining factors'
188         # read in 3 lists of random mining factors
189         # skip first 3 rows (header), mining factor is third column
190         minp_fact = [None]*3
191         transidx = [None]*3
192
193     for i,infu in enumerate(fnminifact):
194         fh = open(infu,'r')
195         data = fh.read().strip()
196         fh.close()
197
198         # read in list of mining factors (floating point 1.0 to 1000.0)
199         minp_fact[i] = [float(line.split()[2]) for line in data.splitlines()[3:]]
200
201         # read in mapping from order in "keepers" -> SECOTF vector order (integer index 1:100)
202         transidx[i] = [int(float(line.strip().split()[3])) for line in data.splitlines()[3:]]
203
204     # loop over all 100 realizations ()
205     fin = open(fnrcallist,'r')
206     rzns = [x.strip() for x in fin.readlines()] # list of 100 MODFLOW realizations in proper order
207     fin.close()
208
209     for i,rzn in enumerate(rzns):
210         if verbose:
211             print i,rzn # print status to screen
212
213         # path into realization directory
214         d = pathrzn + '/' + rzn
215
216         # get the original T fields from the calibration results (1D double-precision vector)
217         K = floatload(d + '/' + fnKfieldin)
218
219         # write unmodified T field to R0 directory
220         # create rzn subdirectories
221         path = './R0/' + rzn
222         if not exists(path):
223             makedirs(path)
224         Kfn = path + '/' + fnKfieldin
225         if exists(Kfn):
226             remove(Kfn)
227         floatsave(Kfn+'.bak',K) # to compare with the linked version
228
229         # perform mining modifications and copy results to directory
230         # creating directory structure for 3 replicates and full/partial mining
231         for j,r in enumerate(['R1','R2','R3']):
232             for m,t in enumerate(['full','part']):
233                 Kmin = [None]*nel
234                 for n,val in enumerate(K):
235                     if min[m][n] == 1:
236                         # apply mining factor, lookup the current flow realization (0-based)
237                         # in the transidx list (1-based) and take that mining factor
238                         Kmin[n] = val*minp_fact[j][transidx[j].index(i+1)]
239                     else:
240                         Kmin[n] = val
241
242                 # save mining modified K into directory tree
243                 # create realization directory if it doesn't exist yet
244                 mfn = '/' + join([r,t,rzn])
245                 if not exists(mfn):
246                     makedirs(mfn)

```

```
247         floatsave(mfn +'/' + fnKfieldout ,Kmin)
249         # debugging version has both K values and mining factors .
251         # (for comparison with original K field)
         if verbose:
             debugsave(mfn +'/' + fnKfieldout+'.dbg',Kmin,min[m])
```

A4.6 Bash shell script link_input_run_mf_dtrk.sh

```

#!/bin/bash
2 verbose=Truc
4 crfile=lmg_err.tmp
6
8 # set a 1 minute timeout for LMG solver (usually runs in ~2 seconds)
ulimit -S -t 60
10 # use absolute paths with ln -s
path='pwd'
12
14 # this should point to the working directory where ReadScript.py was run
# which the parent directory of the current Output directory
# where this script was started
16 based='dirname ${path}'
18
19 if [ ${verbose} == Truc ]; then
20     echo "current dir:" ${path}
21     echo "base dir:" ${based}
22 fi
23
24 #####
# re-run no-mining scenario in the "R0" directory
r=0
26 for d in `cat ${based}/Inputs/keepers_short`; do
27     cd R0/${d}
28
29     echo "current directory:" `pwd`
30
31     # input files that are the same for every realization
32     for f in `cat ${based}/Inputs/files`; do
33         # the "files" list includes directories in the names too
34         ln -sf ${based}/${f} .
35     done
36
37     # handle ibound array separately (here use AP114 Task7 version)
38     ln -sf ${based}/Inputs/AP114_Task7/data/init_bnds.inf .
39
40     # realization-specific files
41     for x in {R,A,S,K}; do
42         ln -sf ${based}/Inputs/AP114_Task7/Outputs/${d}/modeled_${x}_field.mod .
43     done
44
45     # delete any old budget file, so dtrkmf throws
46     # an error if modflow doesn't run
47     if [ -a modeled_flow.bud ]; then
48         rm modeled_flow.bud
49     fi
50
51     # run MF2K
52     if [ ${verbose} == True ]; then
53         ${based}/Inputs/bin/mf2k.1.6.release mf2k.head.nam | tee mf2k_stdout
54     else
55         ${based}/Inputs/bin/mf2k.1.6.release mf2k.head.nam >mf2k_stdout
56     fi
57
58     # dtrkmf reads .dis file, but not .nam file (why?)
59     ln -sf ./elev_bot.mod ./fort.34
60     ln -sf ./elev_top.mod ./fort.33
61
62     # run DTRKMF
63     if [ ${verbose} == True ]; then
64         ${based}/Inputs/bin/dtrkmf.v0100 <dtrkmf.in | tee dtrkmf_stdout
65     else
66         ${based}/Inputs/bin/dtrkmf.v0100 <dtrkmf.in >dtrkmf_stdout
67     fi
68
69     cd ../../
70 done
71
72 if [ ${verbose} == True ]; then
73     echo " running mining-modified MODFLOW realizations"
74 fi
75
76 #####
# run mining modified cases in directories R1,R2 and R3
78 for r in {1,2,3}; do
79     for t in {"full","part"}; do
80         for d in `cat ${based}/Inputs/keepers_short`; do
81             cd R${r}/${t}/${d}
82
83             if [ ${verbose} == True ]; then
84                 echo "current directory:" `pwd`
85             fi
86
87             # input files that are the same for every realization
88             for f in `cat ${based}/Inputs/files`; do
89                 ln -sf ${based}/${f} .

```

```

90  done
92  # handle ibound array separately (here use mining-modified one)
    ln -sf ${based}/Outputs/data/init_bnds_${t}.inf ./init_bnds.inf
94
96  # realization-specific files (not including K, use mining-modified one)
    for x in {R,A,S}; do
98      ln -sf ${based}/Inputs/AP114-Task7/Outputs/${d}/modeled_${x}_field.mod .
    done
100
    ln -sf modeled_mined_K_field.mod modeled_K_field.mod
102
    if [ -a modeled_flow.bud ]; then
104        rm modeled_flow.bud
    fi
106
    # make sure there are no error files laying around
    if [ -s ${erfile} ]; then
108        rm -f ${erfile}
    fi
110
    # run MF2K with LMG solver
    if [ ${verbose} == True ]; then
112        ${based}/Inputs/bin/mf2k_1.6.release mf2k_head.nam | tee mf2k.stdout
    else
114        ${based}/Inputs/bin/mf2k_1.6.release mf2k_head.nam >mf2k.stdout
    fi
116
118    # check for presence of non-zero sized error file.
    # Re-run with direct solver, which is very slow but always works
120    if [ -s ${erfile} ]; then
122
        ln -sf ${based}/Outputs/modflow/mf2k_head.de4.nam .
        ln -sf ${based}/Inputs/mf2k_culebra.de4 .
124
        ulimit -S -t unlimited
        # concatenate log file to the end of other file
126        ${based}/Inputs/bin/mf2k_1.6.release mf2k_head.de4.nam \
            >>mf2k.stdout
        ulimit -S -t 60
128
    fi
130
132    ln -sf ./elev_bot.mod ./fort.34
    ln -sf ./elev_top.mod ./fort.33
134
    # DTRKMF
    if [ ${verbose} == True ]; then
136        ${based}/Inputs/bin/dtrkmf-v0100 <dtrkmf.in | tee dtrkmf.stdout
    else
138        ${based}/Inputs/bin/dtrkmf-v0100 <dtrkmf.in >dtrkmf.stdout
    fi
140
142    cd ../../..
done
144 done
done
146 ulimit -S -t unlimited

```

A4.7 Python script combine_dtrkmf_output_for_gnuplot.py

A4.7.1 Input File

```

601700.0 3597100.0
2 100.0 100.0
dtrk.out
4 dtrk-nomining-all-realizations.dat
dtrk-all-realizations .dat
6 True
    
```

A4.7.2 Script

```

# put all the particle tracks together into one file for simpler
# plotting of the "cloud" of particles using gnuplot
#
# INPUT: 700 dtrkmf output files
# OUTPUT: 7 merged files with all particle tracks together
#         each realization separated by two blank lines
#
8
from glob import glob
10 from sys import exit,argv

12 #####
# read input data
14 try:
    # input name has same name as script ending with ".in", instead of ".py"
16     scrfname = argv[0]
    inpfname = scrfname.replace('.py','.in')
18     fin = open(inpfname,'r')
except IOError:
20     print 'ERROR: %s input file "%s" not found.' %(scrfname,inpfname)
    exit(1)
22

# grid origin for dtrkmf I,J -> UTM NADE7 x,y conversion
24 x0,y0 = [float(x) for x in fin.readline().rstrip().split()]
dx,dy = [float(x) for x in fin.readline().rstrip().split()]
26 fndtrkout = fin.readline().strip() # dtrkmf output filename
fnnominingout = fin.readline().strip() # output filename
28 fnminingout = fin.readline().strip().split() # prefix & suffix of output file
verbose = fin.readline().strip()
30
verbose = verbose[0] == 'T' or verbose[0] == 't'
32
if verbose:
34     print 'echo input'
    print '-----'
36     print 'x0,y0',x0,y0
    print 'dx,dy',dx,dy
38     print 'fndtrkout',fndtrkout
    print 'fnnominingout',fnnominingout
40     print 'fnminingout',fnminingout
    print '-----'
42
#####
44
rep=['R1','R2','R3']
46 min=['full','part']
for r in rep:
48     for m in min:
        ofn = fnminingout[0]+'_'+r+'_'+m+fnminingout[1]
50         if verbose:
            print 'output:',ofn
        fout = open(ofn,'w')
52         for f in glob('/*'.join([r,m,'r???',fndtrkout])):
            if verbose:
54                 print 'input:',f
                # read in all results (except first header line)
                fh = open(f,'r')
56                 results = [l.split() for l in fh.readlines()[1:]]
                fh.close()
58                 # write a gnuplot-friendly header for each realization
                fout.write('# %s \n' % f)
                # write x,y location and time
                for pt in results:
54                     # convert from I,J to x,y
                        x = float(pt[1])*dx + x0
56                     y = y0 - float(pt[2])*dy
                        # convert from 7.75-m to 4-m Culebra
58                     t = float(pt[0])/7.75*4.0
                        fout.write('% .1f,%.1f,%.8e\n' % (x,y,t))
70
                # 2 blank lines between each realization
72                 fout.write('\n\n')
            fout.close()
74

# nomining case
76 r='R0'
    
```

```
76 if verbose:
    print 'output:',fnominingout
80 fout = open(fnominingout,'w')
81 for f in glob('/*').join(['r','r???',fndtrkout]):
82     if verbose:
83         print 'input:',f
84         fh = open(f,'r')
85         results = [l.split() for l in fh.readlines()[1:]]
86         fh.close()
87         fout.write('# %s \n' % f)
88         for pt in results:
89             x = float(pt[1])*dx + x0
90             y = y0 - float(pt[2])*dy
91             t = float(pt[0])/7.75*4.0
92             fout.write('% .1f,% .1f,% .8e\n' % (x,y,t))
93
94     fout.write('\n\n')
95 fout.close()
```

A4.8 Python script extract_dtrkmf_lwb_travel_times.py

A4.8.1 Input File

```

1 601700.0 3597100.0
  100.0 100.0
3 dtrk.out
  wipplwb_results_nomining.out
5 wipplwb_results.out .sort
  True
    
```

A4.8.2 Script

```

# extract the final time from each DIRKMF output file. This corresponds
# to the travel time from the release point at the center of the
# waste panels (C-2737) to the WIPP LWB.
#
# INPUTS: 700 DIRKMF output files (600 mining + 100 no-mining)
# OUTPUTS: 7 summary files with travel time to cross WIPP LWB
#           and x,y location of crossing point.
#           7 summary files with cumulative probability, travel
#           time to the WIPP LWB, files sorted by travel time.

from glob import glob
from sys import exit,argv

#####
# read input data
try:
    # input name has same name as script ending with ".in", instead of ".py"
    scrname = argv[0]
    inpfname = scrname.replace('.py','in')
    fin = open(inpfname,'r')
except IOError:
    print 'ERROR: %s input file "%s" not found.' %(scrname,inpfname)
    exit(1)

# grid origin for dtrkmf I,J -> UTM NAD27 x,y conversion
x0,y0 = [float(x) for x in fin.readline().rstrip().split()]
dx,dy = [float(x) for x in fin.readline().rstrip().split()]
fndtrkout = fin.readline().strip() # dtrkmf output filename
fnnominingout = fin.readline().strip() # output filename
fnminingout = fin.readline().strip().split() # prefix & suffix of output file
verbose = fin.readline().strip()

verbose = verbose[0] == 'T' or verbose[0] == 't'

if verbose:
    print 'echo input'
    print '====='
    print 'x0,y0',x0,y0
    print 'dx,dy',dx,dy
    print 'fndtrkout',fndtrkout
    print 'fnnominingout',fnnominingout
    print 'fnminingout',fnminingout
    print '====='

#####

# loop over all the mining scenarios
for r in ['R1','R2','R3']:
    for scen in ['full','part']:
        ofn = fnminingout[0]+'_'+r+'_'+scen+fnminingout[1]
        if verbose:
            print 'output:',ofn
        fout = open(ofn,'w')
        for d in glob('/*'.join([r,scen,'r???'])):
            # open file, get last row (t,col,row,...)
            ifn = d+'/' + fndtrkout
            if verbose:
                print 'input:',ifn
            fin = open(ifn,'r')
            pt = fin.readlines()[-1].split()
            fin.close()

            # save coordinates of exit point
            x = float(pt[1])*dx + x0
            y = y0 - float(pt[2])*dy

            # save travel time (converting to 4m thick Culebra
            # from 7.75m MODFLOW results)
            t = float(pt[0])/7.75*4.0

            fout.write('%%.8e,%.1f,%.1f,%s\n' % (t,x,y,d[-4:]))

        # write each 100 realizations to different file
        fout.close()

# the non-mining scenario
    
```



```

78 r='R0'
79 if verbose:
80     print 'output:',fnminingout
81     fout = open(fnminingout,'w')
82     for d in glob(r+'r???'):
83
84         ifn = d + '/' + fndtrkout
85         if verbose:
86             print 'input:',ifn
87         fin = open(ifn,'r')
88         pt = fin.readlines()[-1].split()
89         fin.close()
90
91         x = float(pt[1])*dx + x0
92         y = y0 - float(pt[2])*dy
93         t = float(pt[0])/7.75*4.0
94         fout.write('%%.8e,%.1f,%.1f,%s\n' % (t,x,y,d[-4:]))
95
96     fout.close()
97
98     #####
99     # read data back in and pre-process for making CDF figures
100
101     nr = 100 # number of realizations
102
103     # cumulative probability vector
104     p = [float(j+1)/float(nr) for j in range(nr)]
105
106     #####
107     # add first column with cumulative p values
108     # and sort each column from shortest to largest travel time
109     for ifn in glob(fnminingout[0]+'*'+fnminingout[1]):
110         fin = open(ifn,'r')
111         t = [(float(line.split(',')[0]),line.rstrip().split(',')[1]) for line in fin]
112         fin.close()
113
114         # this sorting approach taken from Python Library reference
115         tmplist = [(x[0],x) for x in t]
116         tmplist.sort()
117         t = [x for (key,x) in tmplist]
118
119         ofn = ifn.replace(fnminingout[1],fnminingout[2])
120
121         if verbose:
122             print ifn,'->',ofn
123
124         # give sorted files a different extension
125         fout = open(ofn,'w')
126         for prob,(time,name) in zip(p,t):
127             fout.write('%%.2f,%.8e,%s\n' % (prob,time,name))
128         fout.close()
    
```

A4.9 Python script 100x100_to_50x50.py

A4.9.1 Input File

```

1 284 307 100
2 ../Inputs/AP114.Task7/data/
3 elev_top.mod elev_bot.mod
4 elev_top.mod.50 elev_bot.mod.50
5 init_head.mod
6 init_head.mod.50
7 ./data/
8 init_bnds.inf init_bnds_part.inf init_bnds_full.inf
9 init_bnds.inf.50 init_bnds_part.inf.50 init_bnds_full.inf.50
10 ../Inputs/AP114.Task7/Outputs/
11 modeled_A.field.mod modeled_R.field.mod modeled_S.field.mod modeled_K.field.mod
12 modeled_A.field.mod.50 modeled_R.field.mod.50 modeled_S.field.mod.50 modeled_K.field.mod.50
13 %!le
14 True
    
```

A4.9.2 Script

```

# this script expands the MODFLOW input files from the 100x100m grid
# used in AP-144 to the 50x50m grid used by SECOTPD (AP-145)
#
# INPUTS: 2 integer IBOUND arrays
#          2 real top/bottom elevation arrays
#          1 real starting head array
#          2400 real parameter fields (4 params x 100 realizations x 3 replicates x 2 mining types)
#
# OUTPUTS: 2 integer expanded (50x50m) IBOUND arrays
#           2 expanded real top/bottom elevation arrays
#           1 expanded real starting head array
#           2400 expanded real parameter fields and associated directory structure:
#
from glob import glob
from itertools import chain, izip
from sys import exit, argv

#####
# read input data
try:
    # input name has same name as script ending with ".in", instead of ".py"
    scrfname = argv[0]
    inpfname = scrfname.replace('.py','.in')
    fin = open(inpfname,'r')
except IOError:
    print 'ERROR: %s input file "%s" not found.' %(scrfname,inpfname)
    exit(1)

# columns, rows, and realizations in 100x100m MODFLOW model
nx,ny,nf = [int(x) for x in fin.readline().rstrip().split()]
pathMF2Kdata = fin.readline().strip() # relative paths to AP114 Task7 files
fnelevin = fin.readline().strip().split() # elevation files (input)
fnelevout = fin.readline().strip().split() # " " (output)
fninitheadin = fin.readline().strip() # starting head file (input)
fninitheadout = fin.readline().strip() # " " (output)
pathMF2Kdataout = fin.readline().strip() # path to resulting modified arrays
fnIBOUNDin = fin.readline().strip().split() # IBOUND arrays (input)
fnIBOUNDout = fin.readline().strip().split() # " " (outputs)
pathMF2KOutputs = fin.readline().strip() # path to AP114 Task 7 Outputs
fnfieldsin = fin.readline().strip().split() # {A,R,S,K} fields (input)
fnfieldsout = fin.readline().strip().split() # " " (output)
outfmt = fin.readline().strip() # valid Python format for MF2K output
verbose = fin.readline().strip()
fin.close()

verbose = (verbose[0] == 'T' or verbose[0] == 't')

if verbose:
    print 'echo input'
    print '-----'
    print 'nx,ny,nf',nx,ny,nf
    print 'pathMF2Kdata',pathMF2Kdata
    print 'fnelevin',fnelevin
    print 'fnelevout',fnelevout
    print 'fninitheadin',fninitheadin
    print 'fninitheadout',fninitheadout
    print 'pathMF2Kdataout',pathMF2Kdataout
    print 'fnIBOUNDin',fnIBOUNDin
    print 'fnIBOUNDout',fnIBOUNDout
    print 'pathMF2KOutputs',pathMF2KOutputs
    print 'fnfieldsin',fnfieldsin
    print 'fnfieldsout',fnfieldsout
    print 'outfmt',outfmt
    print '-----'

nel = nx*ny

#####
# define functions
    
```

```

70 # only float{load,save} below convert from string to numeric values
72 # other functions only work with strings for increased speed and to
# eliminate possible changes in trailing significant figures from conversions
74 def matload(filename):
    """Reads file (a 2D array) as a list of lists of strings.
    Outer list is rows, inner lists are columns."""
    f = open(filename, 'r')
    m = [line.rstrip().split() for line in f]
    f.close()
    return m

82 def matsave(filename,m):
    """Writes file as a list of lists as a 2D array (passed in as strings)
    Outer list is rows, inner lists are columns
    (column-major or C-style order)."""
    f = open(filename, 'w')
    for row in m:
    88         f.write(' '.join(row) + '\n')
    f.close()

90 def floatload(filename):
    """Reads file (a list of strings, one per row) into a list of floats."""
    f = open(filename, 'r')
    94     m = [float(line.rstrip()) for line in f]
    f.close()
    96     return m

98 def floatsave(filename,m):
    """Writes file as a list floats, one value per row."""
    f = open(filename, 'w')
    100     f.write('\n'.join([outfmt % num for num in m]) + '\n')
    102     f.close()

104 def vecload(filename):
    """Reads file (a list of real numbers, one number each row)
    106     into a list (no conversion)."""
    f = open(filename, 'r')
    108     m = [line.rstrip() for line in f]
    f.close()
    110     return m

112 def vocsave(filename,m):
    """Writes file as a list floats, one value per row."""
    f = open(filename, 'w')
    114     f.write('\n'.join(m) + '\n')
    116     f.close()

118 # these 3 reshaping routines are generic
# they work on either strings, floats, or integers
120 def reshapem2v(m):
    """Reshapes a rectangular matrix into a vector in
    122     row-major (C-style) order"""
    124     return list(chain(*m))

def reshapev2m(v):
    """Reshape a vector that was previously reshaped in row-major
    126     order from a matrix, back into a matrix (here a list of lists)."""
    128     m = [None]*ny
    for i,(lo,hi) in enumerate(izip(xrange(0, ncl-nx+1, nx),
    130         xrange(nx, ncl+1, nx))):
    132         m[i] = v[lo:hi]
    return m

134 def doublem(m):
    """create a matrix that is doubled in both rows and cols, with
    136     entries copied from each larger cell into the 4 smaller cells
    associated with it."""
    138     md = [None]*(2*ny)
    for i,row in enumerate(m):
    140         drow = list(chain(*[(x,x) for x in row]))
    142         md[2*i] = drow
    144         md[2*i+1] = drow
    return md

146 #####
# top and bottom elevations (double-precision vectors)
148 if verbose:
    150     print 'top/bot elevations'

inbased = pathMF2Kdata
152 outbased = pathMF2Kdataout

154 for fin,fout in zip(fnelevin,fnelevout):
    ifn = inbased + fin
    156     ofn = outbased + fout
    if verbose:
    158         print ifn, '->', ofn
    vocsave(ofn, reshapem2v(doublem(reshapev2m(vecload(ifn)))))

```

```

160 #####
162 # initial / boundary head (double-precision vectors)
163 if verbose:
164     print 'initial head'
166 infname = inbased + fninitheadin
167 outfname = outbased + fninitheadout
168 if verbose:
169     print infname, '->', outfname
170
171 mout = doublem(reshapev2m(floatload(infname)))
172
173 # build up empty "matrices"
174 fact = []
175 res = []
176 for j in range(2*ny):
177     fact.append([0.0 for x in range(2*ux)])
178     res.append([0.0 for x in range(2*nx)])
180 # need to smooth and average heads along exterior boundaries,
181 # need to do conversion to floats (avoided elsewhere)
182
183 # first & last two rows (each boundary cell converted to 2 cells)
184 for row in [0,1,-2,-1]:
185     delta = []
186     # compute smoothing factor
187     for p,v,n in zip(mout[row][0:-2], mout[row][1:-1], mout[row][2:]):
188         # one of these two terms =0, because neighboring values are copies
189         delta.append(v-p + v-n)
190
191     # apply smoothing factor
192     for i in xrange(1,len(mout[row])-1):
193         # factor 1/4 is distance between centers of smaller cells
194         # relative to distance between centers of larger cells
195         fact[row][i] = - 0.25*delta[i-1]
196
197 mtout = zip(*mout)
198
199 # first & last two columns :: zip(*"list of lists") essentially a transpose
200 for col in [0,1,-2,-1]:
201     delta = []
202     for p,v,n in zip(mtout[col][0:-2], mtout[col][1:-1], mtout[col][2:]):
203         delta.append(v-p + v-n)
204
205     for i in xrange(1,len(mtout[col])-1):
206         fact[i][col] = - 0.25*delta[i-1]
208 # apply factors
209 for i,row in enumerate(fact):
210     for j,val in enumerate(row):
211         # don't smooth transition between Land surface constant head and
212         # that specified by the parametric surface
213         if abs(fact[i][j]) < 20.0:
214             res[i][j] = fact[i][j] + mout[i][j]
215         else:
216             res[i][j] = mout[i][j]
218 # save results
219 floatsave(outfname, reshapem2v(res))
220 #####
221 # ibound arrays (integer matrices)
222 if verbose:
223     print 'ibound'
224
225 for fin,fout in zip(fnIBOUNDin[0:1], fnIBOUNDout[0:1]):
226     ifn = inbased + fin
227     ofn = outbased + fout
228     if verbose:
229         print ifn, '->', ofn
230     matsave(ofn, doublem(matload(ifn)))
232
233 for fin,fout in zip(fnIBOUNDin[1:], fnIBOUNDout[1:]):
234     # these files were mining modified and are
235     # already in the "output" directory
236     ifn = outbased + fin
237     ofn = outbased + fout
238     if verbose:
239         print ifn, '->', ofn
240     matsave(ofn, doublem(matload(ifn)))
242 #####
243 # A,R and S fields (DP vectors, same for all 3 mining scenarios)
244 if verbose:
245     print 'anisotropy, recharge, and storativity'
246
247 inbased = pathMF2KOutputs
248 # save mining-agnostic files in the R0/r??? set of sub-directories
249 outbased = './R0/'

```

```
250 realizations = glob(inbased+'r???'')
252 /
252 for r in realizations:
254     rzn = r.split('/')[1] # get just the 'r???' part
254     for fin,fout in zip(fnfieldsin[0:3],fnfieldsout[0:3]):
256         ifn = r + '/' + fin
256         ofn = outbased + rzn + '/' + fout
258         if verbose:
258             print ifn, '->', ofn
260         vecsave(ofn, reshapedm2v(doublem(reshapev2m(vecload(ifn)))))
262
262 #####
264 # K fields (different for each realization and mining scenario)
264 if verbose:
266     print 'hydraulic conductivity'
268 # R[1-3] are replicates (not including non-mining case R0)
268 # * is {full,part} mining
270 # r?? is T-field realization
270 realizations = glob('R[1-3]*/r??/'')
272
272 for r in realizations:
274     # these files are already in ./Outputs, since they are modified
274     ifn = r + fnfieldsin[3]
276     ofn = r + fnfieldsout[3]
276     if verbose:
278         print ifn, '->', ofn
278     vecsave(ofn, reshapedm2v(doublem(reshapev2m(vecload(ifn)))))
```

A4.10 Bash shell script run_50x50_modflow.sh

```

1  #!/bin/bash
3  verbose=Truc
5  # use absolute paths with ln
   path='pwd'
7  based='dirname ${path}'
9  erfile=lmg_err.tmp
11 #set -o xtrace
13 mfexc=mf2k_1.6.release
   #mfexc=mf2k_dp_1.6.release
15
17 # set a 2 minute timeout for LMG solver (usually runs in ~4 seconds)
   ulimit -S -t 120
19 if [ ${verbose} == True ]; then
   echo "current dir:" ${path}
21   echo "base dir:" ${based}
   fi
23 #####
25 # run mining modified cases in directories R1,R2 and R3
   for r in {1,2,3}; do
27     for t in {"full","part"}; do
       for d in `cat ${based}/Inputs/keepers_short`; do
29         cd R${r}/${t}/${d}
31         if [ ${verbose} == Truc ]; then
           echo 'current dir:' R${r} ${t} ${d}
33         fi
35         # input files that are the same for every realization
           for f in `cat ${based}/Inputs/files_50`; do
37             ln -sf ${based}/${f} .
           done
39         # handle ibound array separately
           ln -sf ${based}/Outputs/data/init_bnds_${t}.inf.50 ./init_bnds.inf.50
41
43         # realization-specific files (not including K - use modified one)
           # were saved into the R0/r??? subdirectory structure
           for x in {R,A,S}; do
45             ln -sf ${based}/Outputs/R0/${d}/modeled_${x}.field.mod.50 .
           done
47
49         # make sure there are no error files laying around
           if [ -s ${erfile} ]; then
51             rm -f ${erfile}
           fi
53
55         # run MF2K with LMG solver (fast, works 99% of the time)
           if [ ${verbose} == Truc ]; then
               ${based}/Inputs/bin/${mfexc} mf2k.head_50.nam \
57             | tee mf2k_stdout.50
           else
59             ${based}/Inputs/bin/${mfexc} mf2k.head_50.nam \
               >mf2k_stdout.50
           fi
61
63         # there are two potential reasons that MF2K may need to be re-run with de4
           # -----
65         # 1) MF2K runs to completion, but the solver does not converge
           #    grep has to be used to check for this statement in the listing file
67         # 2) MF2K never finishes, therefore the timelimit must be
           #    used to kill MF2K, producing an error file.
69
71         # If either condition is met re-run with direct solver,
           # which is very slow but always works
73
75         # check for and write out the type of failure
           fail=0
           if grep 'FAILED TO CONVERGE IN TIME STEP 1 OF STRESS PERIOD 1' modeled_head_50.lst; then
77             fail=1
               messg=non-converged
           elif [ -s ${erfile} ]; then
79             fail=2
               messg=timed-out
           fi
81
83         if [[ ${fail} > 0 ]]; then
           if [ ${verbose} == True ]; then
85             echo 'Convergence problem' ${messg} 'in:' R${r} ${t} ${d} 'use DE4 solver'
           fi
87
           ln -sf ${based}/Outputs/modflow/mf2k.head_50.de4.nam .

```

```
80      ln -sf ${based}/Inputs/mf2k_culebra.de4 .
91      ulimit -S -t unlimited
92      # concatenate log file to the end of other file
93      ${based}/Inputs/bin/${mfexe} mf2k.head-50.de4.nam \
94      >>mf2k_stdout.50
95      ulimit -S -t 120
96  fi
97
98  cd ../../..
99  done
100 done
101 done
102 # unset limit set at top of script
103 ulimit -S -t unlimited
```

A4.11 Bash shell script convert_rename_modflow_50x50_budget.sh

```

#!/bin/bash
2
# this script traverses the directory structure, converting
4 # the 50x50m MF2K budget files to ASCII format, and
# copies them to a common directory (renaming them uniquely),
6 # then zips up the budget files for ftp transfer to VMS
9 verbose=Truc
buddir=modflow.50x50-budgets
10
if [ -d ${buddir} ]; then
12     echo "budget destination directory exists"
else
14     mkdir ${buddir}
fi
16
# copy output from each of 600 directories to common directory
18 # renaming files using directory structure
for r in {1,2,3}; do
20     for t in {'full','part'}; do
        index=0
22         for d in `cat ../Inputs/keepers_short`; do
24             # increment counter
                let index++
26
                # save a zero-padded integer as a string for writing as filename
28             fff=`printf "%03d" ${index}`
30
                cd R${r}/${t}/${d}
32
                # link local versions of python script and its input file
                ln -sf ../../../../Inputs/budget_bin2ascii.{py,in} .
34
                # run binary -> ascii Python converter script
36                python budget_bin2ascii.py
38
                # use F and P like VMS is expecting
                if [ ${t} == full ]; then
40                    typ=F
                else
42                    typ=P
                fi
44
                if [ ${verbose} == Truc ]; then
46                    echo "current dir:" `pwd` "-> MF2K_PABC09_R${r}_M${typ}_F${fff}.OUT"
                fi
48
                # move ascii file to common directory while modifying file name
                # to conform to what VMS expects
30                mv modeled_flow.50.asciibud \
52                ../../../../${buddir}/MF2K_PABC09_R${r}_M${typ}_F${fff}.OUT
54
                cd ../../..
            done
        done
    done
56
# zip up files by replicate, for easier transfer to VMS.
# The files are >2GB all zipped up, so it has to be
# broken into three files to stay below that 32-bit file-size limit
62
##cd ${buddir}
##for rep in {R1,R2,R3}; do
## # fastest compression is good enough, therefore use -1
66 ## # -T tests the archive after creating it
## zip -T1 modflow_budgets_${rep} MF2K_PABC09_${rep}.*.OUT
68 ##done
##cd ..

```


A4.12 Python script budget_bin2ascii.py

A4.12.1 Input File

```

modeled_flow.bud
2 %24.16e
modeled_flow.asciibud
4 True
    
```

A4.12.2 Script

```

# this script reads in the binary MODFLOW budget files, saved in
2 # the compact format. ASCII arrays (with one blank line between them)
# for "FLOW RIGHT FACE" and "FLOW FRONT FACE" (qx and qy) are saved to file
4 #
# INPUT: MODFLOW binary budget file
6 # OUTPUT: ASCII file with qx and qy matrices
#
8 # NB: this script assumes one layer in binary file, if there is more than one
# the last layer present will overwrite the previous layers in the ASCII file
10
import struct
12 from itertools import izip
from sys import exit, argv
14
if __name__ == "__main__":
16     #####
    # read input data
18     try:
        # input name has same name as script ending with ".in", instead of ".py"
20         scrfname = argv[0]
        inpfname = scrfname.replace('.py', '.in')
22         fin = open(inpfname, 'r')
    except IOError:
24         print 'ERROR: %s input file "%s" not found.' %(scrfname, inpfname)
        exit(1)
26
        fbinaryin = fin.readline().strip() # MODFLOW binary budget file
28         outfmt = fin.readline().strip() # desired ASCII format (valid Python format)
        fnasciout = fin.readline().strip() # ASCII output file
30         verbose = fin.readline().strip()
        fin.close()
32
        verbose = verbose[0] == 'T' or verbose[0] == 't'
34
        if verbose:
36             print 'echo input'
            print '-----'
38             print 'fbinaryin', fbinaryin
            print 'outfmt', outfmt
40             print 'fnasciout', fnasciout
            print '-----'
42
44     #####
    # define class and functions needed below
46     class FortranFile(file):
        """ modified from May 2007 Enthought-dev mailing list post by Neil Martinsen-Burrell """
48
        def __init__(self, fname, mode='r', buf=0):
50             file.__init__(self, fname, mode, buf)
            self.ENDIAN = '<' # little endian
52             self.di = 4 # default integer, usually 4 bytes, might be 8 on 64-bit platform
            self.dr = 4 # default real, double-precision is 8 bytes, single is 4
54
        def readReals(self):
56             """Read in an array of reals (default precision) with error checking
            prec=d is 'double precision' or 64-bit reals, while prec=f is 'single-
58             precision' with 32-bit reals."""
            if self.dr == 8:
60                 prec='d' # double
            elif self.dr == 4:
62                 prec='f' # float
            else:
64                 print 'incorrectly specified default real'
                    exit(1)
66
            h = struct.unpack(self.ENDIAN+'i', self.read(self.di))[0] # read header
68             data_str = self.read(h)
            if h % self.dr != 0:
70                 raise IOError('Error reading array of reals from data file')
            num = h/self.dr
72             reals = struct.unpack(self.ENDIAN+str(num)+prec, data_str)
            if struct.unpack(self.ENDIAN+'i', self.read(self.di))[0] != h: # check against footer
74                 raise IOError('Error reading array of reals from data file')
            return list(reals)
76
        def readInts(self):
78             """Read in an array of integers with error checking"""
            h = struct.unpack('i', self.read(self.di))[0] # read header
    
```

```

80     data_str = self.read(h)
81     len_int = struct.calcsize('i')
82     if h % len_int != 0:
83         raise IOError('Error reading array of integers from data file')
84     num = h/len_int
85     ints = struct.unpack(str(num)+'i',data_str)
86     if struct.unpack(self.ENDIAN+'i',self.read(self.di))[0] != h: # check against footer
87         raise IOError('Error reading array of integers from data file')
88     return list(ints)

90     def readRecord(self):
91         """Read a single fortran record (potentially mixed reals, ints, and characters)"""
92         dat = self.read(self.di) # read header
93         if len(dat) == 0:
94             raise IOError('Empty record header')
95         h = struct.unpack(self.ENDIAN+'i',dat)[0]
96         data_str = self.read(h)
97         if len(data_str) != h:
98             raise IOError('Didn't read enough data')
99         check = self.read(self.di)
100        if len(check) != self.di:
101            raise IOError('Didn't read enough data')
102        if struct.unpack(self.ENDIAN+'i',check)[0] != h: # check against footer
103            raise IOError('Error reading record from data file')
104        return data_str

106     def reshapev2m(v,nx,ny):
107         """Reshape a vector that was previously reshaped in C-major order from a matrix,
108         back into a C-major order matrix (here a list of lists)."""
109         m = [None]*ny
110         n = nx*ny
111         for i,(lo,hi) in enumerate(izip(xrange(0, n-nx+1, nx), xrange(nx, n+1, nx))):
112             m[i] = v[lo:hi]
113         return m

114     def floatmatsave(filehandle,m):
115         """Writes array to open filehandle.
116         Outer list is rows, inner lists are columns."""
117
118         for row in m:
119             f.write(''.join([outfmt % col for col in row]) + '\n')
120
121     #####
122
123     if __name__ == "__main__":
124
125         # open binary file
126         ff = FortranFile(fbinaryin)
127
128         # format of MODFLOW header in binary budget file
129         fmt = '<2i16s3i' # little endian, 2 integers, 16-character string, 3 integers
130
131         while True:
132             try:
133                 # read in header
134                 h = ff.readRecord()
135
136             except IOError:
137                 # exit while loop
138                 break
139
140             else:
141                 # unpack header
142                 kstp,kper,text,ncol,nrow,nlay = struct.unpack(fmt,h)
143                 if verbose:
144                     print kstp,kper,text,ncol,nrow,nlay
145
146                 # second header read but not needed
147                 throw_away = ff.readRecord()
148
149                 if text.strip() == 'FLOW RIGHT FACE':
150                     qx = ff.readReals()
151                     assert len(qx) == nrow*ncol, 'qx array is not expected size; check binary format'
152
153                 elif text.strip() == 'FLOW FRONT FACE':
154                     qy = ff.readReals()
155                     assert len(qy) == nrow*ncol, 'qy array is not expected size; check binary format'
156
157                 else: # text.strip() == 'CONSTANT HEAD' (and potentially other cases)
158                     num = ff.readInts()[0]
159
160                     # there are "num" Constant head records consisting of default integers
161                     # and reals (4 or 8 bytes each),
162                     # padded fore and aft by default integers (4 bytes each on 32-bit)
163                     # second argument to seek (1) incates a seek relative to current position
164
165                     jump = num*(3*ff.di + ff.dr)
166                     if verbose:
167                         print 'num:',num,'jump:',jump

```

```
ff.seek(jump,1)
170 ff.close()
172
173 # save ASCII file
174 f = open(fnasciiout, 'w')
floatmatsave(f, reshapev2m(qx, ncol, nrow))
176 f.write('\n')
floatmatsave(f, reshapev2m(qy, ncol, nrow))
178 f.close()
```