Software Quality Assurance Plan
for PORFLOW Flow-Field Extraction Tool (GoldSimFlows)

G. P. Flach, B. T. Butcher

JUNE 2013

Savannah River National Laboratory
Washington Savannah River Company
Savannah River Site
Aiken, SC 29808

Prepared for the U.S. Department of Energy Under
Contract Number DE-AC09-96SR18500

SRNL

DISCLAIMER

Printed in the United States of America

Prepared For
U.S. Department of Energy

Key Words:
Performance Assessment
Flow Simulation
Sensitivity
Uncertainty

Retention: Permanent

Software Quality Assurance Plan for PORFLOW Flow-Field Extraction Tool (GoldSimFlows)

G. P. Flach, B. T. Butcher

JUNE 2013

Savannah River National Laboratory
Washington Savannah River Company
Savannah River Site
Aiken, SC 29808

Prepared for the U.S. Department of Energy Under Contract Number DE-AC09-96SR18500

SRNL™

REVIEWS AND APPROVALS

6/5/2013

G. P. Flach, Design Agency / Cognizant Technical Function (CTF) — Date
Radiological Performance Assessment

6/5/13

G. A. Taylor, Independent Reviewer — Date
Radiological Performance Assessment

6/12/2013

W. A. Drown, Cognizant Quality Function (CQF) — Date
SRNL Quality

6/6/13

B. T. Butcher, Design Authority — Date
Radiological Performance Assessment

6/6/13

D. A. Crowley, Design Authority Manager — Date
Manager, Radiological Performance Assessment

6/12/13

R. S. Aylward, Manager, — Date
Environmental Restoration Technology

EXECUTIVE SUMMARY

This Software Quality Assurance Plan (SQAP) describes the software, "GoldSimFlows", a Fortran90 program that extracts flow field data from a series of steady-state 2D PORFLOW simulations for selected regions in the model domain and writes the data to a tab-delimited spreadsheet-type file. This software retrieves selected output from a steady-state 2D PORFLOW flow simulation, calculates Eh and pH transition times based on pore volume counts, and writes these data for user-defined sub-regions to a tab-delimited file intended for GoldSim import. This document describes both lifecycle requirements and verification techniques for the application of this software tool.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

CQF	Cognizant Quality Function
CTF	Cognizant Technical Function
DOE	U.S. Department of Energy
GSA	General Separations Area
HPC	High Performance Computing
PA	Performance Assessment
QA	Quality Assurance
QAP	Quality Assurance Procedure
SQAP	Software Quality Assurance Plan
SRNL	Savannah River National Laboratory
SRS	Savannah River Site
V&V	Verification and Validation

1.0 INTRODUCTION

The Radiological Performance Assessment group of Savannah River National Laboratory (SRNL) conducts Performance Assessments (PA) of Savannah River Site (SRS) low-level waste facilities to provide a reasonable assurance that the Performance Objectives of DOE Order 435.1, *Radioactive Waste Management,* (DOE, 1999) will be met. Each PA includes an analysis of contaminant transport through the saturated zone between the disposal unit and compliance boundary. The aquifer transport analysis is based on the velocity and saturation fields produced by a General Separations Area (GSA) regional groundwater flow model referred to as GSA/PORFLOW (Flach 2004). PORFLOW refers to a commercial code used to simulate groundwater flow (ACRi). Probabilistic uncertainty and sensitivity analysis is typically conducted in parallel to PORFLOW modeling for PA's using GoldSim, a Windows-based commercial program for carrying out dynamic, probabilistic simulation of complex systems. The GoldSim system model includes vadose zone and aquifer transport submodels, but does not simulate multi-dimensional vadose zone flow because of the complexities of time-varying material properties and solution of the non-linear Richards (unsaturated flow) equation. Instead, GoldSim imports PORFLOW vadose zone flow fields generated for the nominal case and additional flow sensitivity cases as needed. A software tool, "GoldSimFlows", has been written for extracting flow field information from PORFLOW output for import to GoldSim.

2.0 SCOPE

This SQAP follows the guidelines and minimum content requirements specified in the Savannah River Site *1Q Quality Assurance Manual*, Procedure (QAP) 20-1, "Software Quality Assurance" (SRS, 2013). The GoldSimFlows software application has been classified as Level "C" software per QAP 20-1, Attachment 8.1, which classifies "software applications used to comply with regulatory laws, environmental permits or regulations and/or commitments to compliance" as Level "C" software. The GoldSimFlows software application together with other Performance Assessment software is used to provide a reasonable assurance that the Performance Objectives of DOE Order 435.1 will be met for the various SRS low-level waste disposal units. This SQAP addresses the life cycle requirements and verification techniques for the GoldSimFlows software application as Level "C" software.

3.0 ROLES AND RESPONSIBILITIES

Oversight and assignment of all PA related work, including use of the GoldSimFlows software, is provided by Dave Crowley, Manager of the Radiological Performance Assessments group. The Design Authority responsible for implementation of software Quality Assurance (QA) for PA related work is Tom Butcher. The Cognizant Technical Function (CTF) and Design Agency associated with the GoldSimFlows software is Greg Flach who also maintains and is the sole user of this software. The Cognizant Quality Function (CQF) is Wayne Drown of the SRNL QA group. Should other SRNL business programs choose to use GoldSimFlows at the same Level "C" functional classification,

oversight will be provided by the appropriate Manager. The CTF, CQF and this SQAP can remain unchanged.

4.0 SOFTWARE LIFE CYCLE

SRS Manual 1Q, Procedure 20-1 will be used to assure quality throughout the Software Life Cycle. Software specific implementation of QAP 20-1 for GoldSimFlows is discussed below.

4.1 FUNCTION

The functional requirements phase defines the software capabilities required to perform the task of interest that will be designed into the software by in-house or vendor software designers. This software was created to import PORFLOW vadose zone flow fields into a spreadsheet-type format that could be directly used in a GoldSim model to perform probabilistic sensitivity and uncertainty analyses.

GoldSimFlows extracts flow field data from a series of steady-state 2D PORFLOW simulations for selected regions in the model domain and writes the data to a tab-delimited spreadsheet-type file. The flow field data are vertical Darcy velocity, horizontal Darcy velocity, flowrate, saturation, pore volume, Eh transition time, and pH transition time. The regions of interest are rectangular and defined by lower-left and upper-right cell vertex indices in Tecplot specified in an input file typically named “GoldSimZones.txt”. Groups of rectangular regions can also be analyzed. The main output file is typically named GoldSim_*.tab where * specifies a specific tank/vault and scenario/case. EhpH_*.tab provides just the Eh and pH transition time information. GoldSim_*.out provides more detailed information than is presented in GoldSim_*.tab. GoldSim_*.tec is enables convenient Tecplot plotting of the variables “time darcyV darcyH saturation flow volume volumePore volumeWater”. GoldSim_*.log is a log file focused on reading input files.

The software must be able to execute on a SRNL multi-platform environment using the current operating system of choice. At present the platform of choice is an SRNL HPC workstation running Linux.

4.2 EVALUATION

The GoldSimFlows code was developed by Greg Flach as a personal calculation tool, and first applied in its current configuration to a Special Analysis for the Saltstone Disposal Facility (Jordan and Flach 2013). Following initial code development and multiple uses controlled by design-checking of each application, the GoldSimFlows code was found to have general applicability to E-Area PA, Saltstone PA and Tank Closure PA analyses.

A software classification evaluation was completed which determined that this application meets the Level “C” software criteria (Butcher, 2013). GoldSimFlows is existing software currently used solely by the software developer. According to QAP 20-1, Attachment 8.2, GoldSimFlows is further classified as "Level C Existing Software". Required components of the SQAP are "Evaluation" and "Configuration Control", with other elements optionally

applied using a graded approach. Additional optional components deemed appropriate for GoldSimFlows are Design, (Verification and Validation, V&V) Testing, Acceptance Testing, Access Control, Cyber Security Control, Problem Reporting and Corrective Action, and Software Retirement. Requirements and Implementation are not warranted because GoldSimFlows is existing software. Formal User Instructions, and Operation and Maintenance procedures are not warranted, provided the developer (Greg Flach) continues to be the sole user of GoldSimFlows. The SQAP will be modified should additional Users be desired.

The selected optional elements are discussed further below. Software Design and initial V&V and Acceptance Testing are documented as part of this SQAP.

4.3 DESIGN

The primary function of GoldSimFlows is to retrieve selected output from a steady-state 2D PORFLOW flow simulation. The simulation data are retrieved from the main PORFLOW output/archive file, conventionally named "MAIN.sav". The PORFLOW simulation is assumed to be composed of a sequence of steady-state flow simulations that make up a pseudo-transient flow simulation. The flow simulations must reside in subdirectories named TI01, TI02, etc. Appendix A contains the GoldSimFlows Fortran90 source code.

Input specifications

The application of GoldSimFlows can be illustrated for a cylindrical tank disposal system. The input specifications for GoldSimFlows are shown by the annotated example input file in Table 1. Figure 1 illustrates the PORFLOW grid and material zones for the cross-section associated with this example. Appendix B lists the PORFLOW input files for the cylindrical coordinate simulations. The grid in Figure 1 is a simple analog of the cylindrical disposal unit (half-width) surrounded by soil. The outer shell is a concrete barrier, and the interior is a grout waste. A hypothetical fast-flow path fully penetrates the disposal cell. The grid dimensions in terms of cell vertices are 12 x 10.

Table 1 is a cylindrical coordinate example of GoldSimFlows input. For Cartesian ("xyz") coordinates, an additional line is required in the input file after the geometry specification, namely, the geometric factor between the unit thickness 2D PORFLOW simulation and the 3D physical geometry. For example, if the PORFLOW length unit is a centimeter, half the physical feature is modeled (due to symmetry), and the physical geometry is 100 ft long, then the geometric factor is 2 halves*100 ft*30.48 cm/ft. For cylindrical geometries the geometric factor is 2π and handled internally by GoldSimFlows.

Grid regions of interest are specified through grid vertex indices defining the lower-left and upper-right corners of the zone; thus grid zones must be rectangular with respect to indices. The grid indices of interest are most easily determined by probing a Tecplot data zone corresponding to the grid vertices. In the example input, the indices (1,3) to (9,8) capture the concrete, grout, and fast-flow features, i.e., the whole disposal unit. The "Eh" and "pH"

values are the numbers of pore volume flushes at which Eh and pH transitions occur. Arbitrary values of 100 and 1000 pore volumes are chosen for Eh and pH transitions, respectively, in this example. The "xFlow" value is the lateral Darcy flow that is not explicitly represented in the PORFLOW model; see SRNL-L6200-2010-00026, Rev. 1 "PORFLOW Modeling Supporting the H-Tank Farm Performance Assessment" for further information (Jordan et al., 2010). In this example the crossflow rate is arbitrarily set to 100 cm/yr. The "group" parameter allows the user to combine two or more rectangular zones into the union of these zones. All zones with the same "group" value #### greater than 0 are combined to form a new region named "Group####". The group number is arbitrary (but less than or equal to 9999). In this example, the grout regions on each side of the fast-flow path are combined into a third combined zone that will be named "Group0003".

Table 1. Annotated example input file to GoldSimFlows.

```
./Flow/CaseB_xrt                !path to PORFLOW simulation
MAIN.sav                        !name of PORFLOW archive file
4                               !number of steady-state flow periods
xrt                             !geometry (xyz = cartesian, xrt =
cylindrical)
.                               !path to output directory
GoldSim_CaseB_xrt.log           !name of log file
GoldSim_CaseB_xrt.out           !name of detailed output file
GoldSim_CaseB_xrt.tec           !name of Tecplot output file
FlowDetail_CaseB_xrt.tab        !name of flow detail tab file
GoldSim_CaseB_xrt.tab           !name of GoldSim tab file
EhpH_CaseB_xrt.tab              !name of Eh/pH tab file
CaseB_xrt                       !label
.false.                         !write crossflow field? (.false. or .true.)
!zone label      iLL   jLL   iUR   jUR    Eh    pH    xFlow group
Domain            1     1     12    10    1e20  1e20  0     0
DisposalCell      1     3     9     8     100   1000  0     0
DisposalXflow     1     3     9     8     100   1000  100   0
GroutInner        1     4     4     7     100   1000  0     3
GroutOuter        5     4     8     7     100   1000  0     3
```

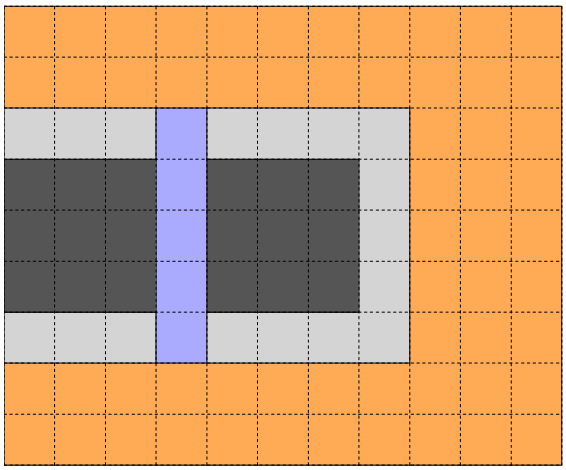

Figure 1. Example disposal cell cross-section (half-width) showing PORFLOW grid and material zone.

Output files

The various output files produced by GoldSim flows are named and briefly identified in Table 1. The primary file of interest is "GoldSim_CaseB_xrt.tab", as this file is intended for subsequent GoldSim modeling.

4.4 TESTING

Verification & Validation Testing

Two PORFLOW model geometries were evaluated by V&V tests devised for GoldSimFlows. The geometries included the example disposal unit defined by cylindrical coordinates described in Section 4.3 above. A second disposal unit geometry, defined by a rectangular prism using Cartesian coordinates was also evaluated. Appendix C contains the PORFLOW input files for Cartesian coordinates case.

Table 2 compares GoldSimFlows output in the files "GoldSim_CaseB_x[y,r][z,t].tab" (xyz/Cartesian and xrt/cylindrical coordinates) to independent PORFLOW output and spreadsheet calculations. The GoldSimFlows "darcyV", "darcyU", and "sat" values are compared to average values from the independent PORFLOW "STAT.out" files. PORFLOW apparently gives equal weight to each grid cell when averaging, whereas GoldSimFlows uses volume weighting. For Cartesian coordinates and a uniform 2D grid, these averaging schemes produce the same result. For cylindrical coordinates, the PORFLOW STATistics values differ from the GoldSimFlows results. The "flow" values for comparison come from PORFLOW "FLUX.out", and the "pore" values from "FLUXC.out". The GoldSimFlows and PORFLOW FLUX outputs are the same. The Eh and pH reference

values are computed from the flow and pore volume data, and agree with GoldSimFlows output. To verify the GoldSimFlows averaging in cylindrical coordinates for "darcyV", "darcyU", and "sat" values, separate manual calculations were performed for the 0 - 100 yr period and the "DOMAIN" region as shown in Table 3. The individual cell values were extracted from the "MAIN.sav" PORFLOW output file. The averages were computed using values weighted by $\pi(r_{outer}^2 - r_{inner}^2)$. The hand-calculated values agree with the GoldSimFlows output shown in Table 2.

The electronic files associated with this test case are located at:

\\godzilla-01\hpc_project\projwork23\pa\GoldSimFlows Test Case

Table 2. Comparison of GoldSimFlows output to reference values for software test cases "CaseB_xyz" and "CaseB_xrt".

CaseB_xyz	tBeg	tEnd	Domain_darcyV	DisposalCell_darcyV	DisposalXflow_darcyV	GroutInner_darcyV	GroutOuter_darcyV
CaseB_xyz	0	100	-1.00E+01	-9.60E-03	-9.60E-03	-1.14E-02	-1.23E-02
PORFLOW			-1.00E+01	-9.60E-03	-9.60E-03	nd	nd
CaseB_xyz	100	1000	-1.00E+01	-5.61E-02	-5.61E-02	-3.34E-02	-3.47E-02
PORFLOW			-1.00E+01	-5.61E-02	-5.61E-02	nd	nd
CaseB_xyz	1000	10000	-1.00E+01	-6.83E+00	-6.83E+00	-5.08E-02	-5.26E-02
PORFLOW			-1.00E+01	-6.83E+00	-6.83E+00	nd	nd
CaseB_xyz	10000	100000	-1.00E+01	-1.01E+01	-1.01E+01	-7.98E+00	-8.22E+00
PORFLOW			-1.00E+01	-1.01E+01	-1.01E+01	nd	nd
poreVolumes							
transitionTime							
CaseB_xrt	tBeg	tEnd	Domain_darcyV	DisposalCell_darcyV	DisposalXflow_darcyV	GroutInner_darcyV	GroutOuter_darcyV
CaseB_xrt	0	100	-1.00E+01	-9.07E-03	-9.07E-03	-1.22E-02	-1.23E-02
PORFLOW			-6.59E+00	-9.62E-03	-9.62E-03	nd	nd
CaseB_xrt	100	1000	-1.00E+01	-4.91E-02	-4.91E-02	-4.99E-02	-2.99E-02
PORFLOW			-6.61E+00	-5.68E-02	-5.68E-02	nd	nd
CaseB_xrt	1000	10000	-1.00E+01	-4.73E+00	-4.73E+00	-7.51E-02	-4.49E-02
PORFLOW			-8.67E+00	-5.41E+00	-5.41E+00	nd	nd
CaseB_xrt	10000	100000	-1.00E+01	-1.01E+01	-1.01E+01	-6.99E+00	-8.57E+00
PORFLOW			-9.96E+00	-9.91E+00	-9.91E+00	nd	nd
poreVolumes							
transitionTime							

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	Group0003_darcyV	Domain_darcyH	DisposalCell_darcyH	DisposalXflow_darcyH	GroutInner_darcyH
CaseB_xyz	0	100	-1.18E-02	3.79E+00	4.20E-04	4.20E-04	2.10E-05
PORFLOW			-1.18E-02	3.79E+00	4.20E-04	4.20E-04	nd
CaseB_xyz	100	1000	-3.41E-02	3.77E+00	1.34E-03	1.34E-03	-2.50E-03
PORFLOW			-3.41E-02	3.77E+00	1.34E-03	1.34E-03	nd
CaseB_xyz	1000	10000	-5.17E-02	8.56E-01	1.20E-03	1.20E-03	-1.72E-03
PORFLOW			-5.17E-02	8.56E-01	1.20E-03	1.20E-03	nd
CaseB_xyz	10000	100000	-8.10E+00	-8.73E-02	-1.23E-01	-1.23E-01	6.31E-01
PORFLOW			-8.10E+00	-8.73E-02	-1.23E-01	-1.23E-01	nd
poreVolumes							
transitionTime							
CaseB_xrt	tBeg	tEnd	Group0003_darcyV	Domain_darcyH	DisposalCell_darcyH	DisposalXflow_darcyH	GroutInner_darcyH
CaseB_xrt	0	100	-1.23E-02	1.96E+00	7.13E-04	7.13E-04	3.00E-05
PORFLOW			-1.20E-02	1.83E+00	5.03E-04	5.03E-04	nd
CaseB_xrt	100	1000	-3.42E-02	1.95E+00	3.25E-03	3.25E-03	-2.12E-03
PORFLOW			-3.70E-02	1.82E+00	2.34E-03	2.34E-03	nd
CaseB_xrt	1000	10000	-5.14E-02	5.94E-01	2.24E-03	2.24E-03	-1.01E-03
PORFLOW			-5.55E-02	4.83E-01	1.66E-03	1.66E-03	nd
CaseB_xrt	10000	100000	-8.23E+00	-1.47E-01	-4.48E-01	-4.48E-01	5.56E-01
PORFLOW			-7.86E+00	-1.40E-01	-2.42E-01	-2.42E-01	nd
poreVolumes							
transitionTime							

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	GroutOuter_darcyH	Group0003_darcyH	infiltration	Domain_flow	DisposalCell_flow	DisposalXflow_flow
CaseB_xyz	0	100	7.24E-04	3.73E-04	1.00E+01	6.71E+07	4.98E+04	4.98E+04
PORFLOW			nd	3.73E-04	1.00E+01	6.71E+07	4.98E+04	4.98E+04
CaseB_xyz	100	1000	5.59E-03	1.54E-03	1.00E+01	6.71E+07	2.78E+05	2.78E+05
PORFLOW			nd	1.54E-03	1.00E+01	6.71E+07	2.78E+05	2.78E+05
CaseB_xyz	1000	10000	4.83E-03	1.56E-03	1.00E+01	6.71E+07	3.33E+07	3.33E+07
PORFLOW			nd	1.56E-03	1.00E+01	6.71E+07	3.33E+07	3.33E+07
CaseB_xyz	10000	100000	-8.58E-01	-1.14E-01	1.00E+01	6.71E+07	4.95E+07	4.95E+07
PORFLOW			nd	-1.14E-01	1.00E+01	6.71E+07	4.95E+07	4.95E+07
poreVolumes								
transitionTime								
CaseB_xrt	tBeg	tEnd	GroutOuter_darcyH	Group0003_darcyH	infiltration	Domain_flow	DisposalCell_flow	DisposalXflow_flow
CaseB_xrt	0	100	8.34E-04	6.61E-04	1.00E+01	3.80E+07	2.07E+04	2.07E+04
PORFLOW			nd	4.72E-04	1.00E+01	3.80E+07	2.07E+04	2.07E+04
CaseB_xrt	100	1000	6.95E-03	5.00E-03	1.00E+01	3.80E+07	1.02E+05	1.02E+05
PORFLOW			nd	2.86E-03	1.00E+01	3.80E+07	1.02E+05	1.02E+05
CaseB_xrt	1000	10000	4.85E-03	3.59E-03	1.00E+01	3.80E+07	9.51E+06	9.51E+06
PORFLOW			nd	2.19E-03	1.00E+01	3.80E+07	9.51E+06	9.51E+06
CaseB_xrt	10000	100000	-8.24E-01	-5.29E-01	1.00E+01	3.80E+07	2.05E+07	2.05E+07
PORFLOW			nd	-1.97E-01	1.00E+01	3.80E+07	2.05E+07	2.05E+07
poreVolumes								
transitionTime								

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	GroutInner_flow	GroutOuter_flow	Group0003_flow	Domain_sat	DisposalCell_sat	DisposalXflow_sat
CaseB_xyz	0	100	2.16E+04	2.41E+04	4.57E+04	8.78E-01	1.00E+00	1.00E+00
PORFLOW			nd	nd	4.57E+04	8.78E-01	1.00E+00	1.00E+00
CaseB_xyz	100	1000	8.24E+04	8.68E+04	1.69E+05	8.78E-01	1.00E+00	1.00E+00
PORFLOW			nd	nd	1.69E+05	8.78E-01	1.00E+00	1.00E+00
CaseB_xyz	1000	10000	1.47E+05	1.51E+05	2.98E+05	8.52E-01	9.74E-01	9.74E-01
PORFLOW			nd	nd	2.98E+05	8.52E-01	9.74E-01	9.74E-01
CaseB_xyz	10000	100000	1.59E+07	1.68E+07	3.27E+07	7.05E-01	6.51E-01	6.51E-01
PORFLOW			nd	nd	3.27E+07	7.05E-01	6.51E-01	6.51E-01
poreVolumes								
transitionTime								
CaseB_xrt	tBeg	tEnd	GroutInner_flow	GroutOuter_flow	Group0003_flow	Domain_sat	DisposalCell_sat	DisposalXflow_sat
CaseB_xrt	0	100	3.66E+03	1.39E+04	1.75E+04	8.36E-01	1.00E+00	1.00E+00
PORFLOW			nd	nd	1.75E+04	8.67E-01	1.00E+00	1.00E+00
CaseB_xrt	100	1000	1.96E+04	4.21E+04	6.17E+04	8.36E-01	1.00E+00	1.00E+00
PORFLOW			nd	nd	6.17E+04	8.67E-01	1.00E+00	1.00E+00
CaseB_xrt	1000	10000	3.44E+04	7.20E+04	1.06E+05	8.20E-01	9.73E-01	9.73E-01
PORFLOW			nd	nd	1.06E+05	8.46E-01	9.69E-01	9.69E-01
CaseB_xrt	10000	100000	2.23E+06	9.82E+06	1.21E+07	7.14E-01	6.59E-01	6.59E-01
PORFLOW			nd	nd	1.21E+07	7.05E-01	6.50E-01	6.50E-01
poreVolumes								
transitionTime								

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	GroutInner_sat	GroutOuter_sat	Group0003_sat	Domain_pore	DisposalCell_pore	DisposalXflow_pore
CaseB_xyz	0	100	1.00E+00	1.00E+00	1.00E+00	2.23E+09	8.62E+08	8.62E+08
PORFLOW			nd	nd	1.00E+00	2.23E+09	8.62E+08	8.62E+08
CaseB_xyz	100	1000	1.00E+00	1.00E+00	1.00E+00	2.35E+09	9.87E+08	9.87E+08
PORFLOW			nd	nd	1.00E+00	2.35E+09	9.87E+08	9.87E+08
CaseB_xyz	1000	10000	1.00E+00	1.00E+00	1.00E+00	2.29E+09	9.26E+08	9.26E+08
PORFLOW			nd	nd	1.00E+00	2.29E+09	9.26E+08	9.26E+08
CaseB_xyz	10000	100000	7.00E-01	7.01E-01	7.01E-01	2.27E+09	9.02E+08	9.02E+08
PORFLOW			nd	nd	7.01E-01	2.27E+09	9.02E+08	9.02E+08
poreVolumes								
transitionTime								
CaseB_xrt	tBeg	tEnd	GroutInner_sat	GroutOuter_sat	Group0003_sat	Domain_pore	DisposalCell_pore	DisposalXflow_pore
CaseB_xrt	0	100	1.00E+00	1.00E+00	1.00E+00	1.25E+09	3.32E+08	3.32E+08
PORFLOW			nd	nd	1.00E+00	1.25E+09	3.32E+08	3.32E+08
CaseB_xrt	100	1000	1.00E+00	1.00E+00	1.00E+00	1.30E+09	3.77E+08	3.77E+08
PORFLOW			nd	nd	1.00E+00	1.30E+09	3.77E+08	3.77E+08
CaseB_xrt	1000	10000	1.00E+00	1.00E+00	1.00E+00	1.27E+09	3.55E+08	3.55E+08
PORFLOW			nd	nd	1.00E+00	1.27E+09	3.55E+08	3.55E+08
CaseB_xrt	10000	100000	6.94E-01	7.03E-01	7.01E-01	1.29E+09	3.73E+08	3.73E+08
PORFLOW			nd	nd	6.99E-01	1.29E+09	3.73E+08	3.73E+08
poreVolumes								
transitionTime								

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	GroutInner_pore	GroutOuter_pore	Group0003_pore	Domain_Eh	DisposalCell_Eh	DisposalXflow_Eh
CaseB_xyz	0	100	3.18E+08	3.18E+08	6.36E+08	-999	3774	1246
PORFLOW			nd	nd	6.36E+08			
CaseB_xyz	100	1000	3.18E+08	3.18E+08	6.36E+08	-999	3774	1246
PORFLOW			nd	nd	6.36E+08			
CaseB_xyz	1000	10000	3.18E+08	3.18E+08	6.36E+08	-999	3774	1246
PORFLOW			nd	nd	6.36E+08			
CaseB_xyz	10000	100000	2.08E+08	2.08E+08	4.17E+08	-999	3774	1246
PORFLOW			nd	nd	4.17E+08			
poreVolumes						nc	100	100
transitionTime						nc	3774	1246
CaseB_xrt	tBeg	tEnd	GroutInner_pore	GroutOuter_pore	Group0003_pore	Domain_Eh	DisposalCell_Eh	DisposalXflow_Eh
CaseB_xrt	0	100	4.92E+07	1.80E+08	2.30E+08	-999	4727	1330
PORFLOW			nd	nd	2.30E+08			
CaseB_xrt	100	1000	4.92E+07	1.80E+08	2.30E+08	-999	4727	1330
PORFLOW			nd	nd	2.30E+08			
CaseB_xrt	1000	10000	4.92E+07	1.80E+08	2.30E+08	-999	4727	1330
PORFLOW			nd	nd	2.30E+08			
CaseB_xrt	10000	100000	3.22E+07	1.18E+08	1.50E+08	-999	4727	1330
PORFLOW			nd	nd	1.50E+08			
poreVolumes						nc	100	100
transitionTime						nc	4727	1330

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	GroutInner_Eh	GroutOuter_Eh	Group0003_Eh	Domain_pH	DisposalCell_pH	DisposalXflow_pH
CaseB_xyz	0	100	11255	11184	11218	-999	22333	3521
PORFLOW								
CaseB_xyz	100	1000	11255	11184	11218	-999	22333	3521
PORFLOW								
CaseB_xyz	1000	10000	11255	11184	11218	-999	22333	3521
PORFLOW								
CaseB_xyz	10000	100000	11255	11184	11218	-999	22333	3521
PORFLOW								
poreVolumes			100	100	100	nc	1000	1000
transitionTime			11255	11184	11218	nc	22333	3521
CaseB_xrt	tBeg	tEnd	GroutInner_Eh	GroutOuter_Eh	Group0003_Eh	Domain_pH	DisposalCell_pH	DisposalXflow_pH
CaseB_xrt	0	100	11350	11157	11193	-999	23783	4387
PORFLOW								
CaseB_xrt	100	1000	11350	11157	11193	-999	23783	4387
PORFLOW								
CaseB_xrt	1000	10000	11350	11157	11193	-999	23783	4387
PORFLOW								
CaseB_xrt	10000	100000	11350	11157	11193	-999	23783	4387
PORFLOW								
poreVolumes			100	100	100	nc	1000	1000
transitionTime			11350	11157	11193	nc	23783	4387

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	GroutInner_pH	GroutOuter_pH	Group0003_pH	Domain_flow	DisposalCell_flow	DisposalXflow_flow
CaseB_xyz	0	100	23067	22343	22695	3.01	0.01	0.06
PORFLOW						3.01	0.01	0.06
CaseB_xyz	100	1000	23067	22343	22695	25.64	0.25	2.78
PORFLOW						28.65	0.26	2.85
CaseB_xyz	1000	10000	23067	22343	22695	263.19	323.59	3559.50
PORFLOW						291.83	323.85	3562.35
CaseB_xyz	10000	100000	23067	22343	22695	2659.86	4934.27	54277.02
PORFLOW						2951.69	5258.12	57839.37
poreVolumes			1000	1000	1000	0	0	100
transitionTime			23067	22343	22695			
CaseB_xrt	tBeg	tEnd	GroutInner_pH	GroutOuter_pH	Group0003_pH	Domain_flow	DisposalCell_flow	DisposalXflow_flow
CaseB_xrt	0	100	24366	21987	22427	3.04	0.01	0.07
PORFLOW						3.04	0.01	0.07
CaseB_xrt	100	1000	24366	21987	22427	26.41	0.24	2.68
PORFLOW						29.45	0.25	2.75
CaseB_xrt	1000	10000	24366	21987	22427	268.68	240.91	2649.96
PORFLOW						298.13	241.16	2652.71
CaseB_xrt	10000	100000	24366	21987	22427	2649.50	4955.20	54507.15
PORFLOW						2947.64	5196.35	57159.86
poreVolumes			1000	1000	1000	0	0	100
transitionTime			24366	21987	22427			

Table 2 (continued)

CaseB_xyz	tBeg	tEnd	GroutInner_flow	GroutOuter_flow	Group0003_flow	comment
CaseB_xyz	0	100	0.01	0.01	0.01	pore volumes in this TI
PORFLOW			0.01	0.01	0.01	cumulative through this TI
CaseB_xyz	100	1000	0.23	0.25	0.24	pore volumes in this TI
PORFLOW			0.24	0.25	0.25	cumulative through this TI
CaseB_xyz	1000	10000	4.16	4.27	4.22	pore volumes in this TI
PORFLOW			4.40	4.52	4.46	cumulative through this TI
CaseB_xyz	10000	100000	6857.18	7258.35	7057.76	pore volumes in this TI
PORFLOW			6861.58	7262.87	7062.22	cumulative through this TI
poreVolumes			0	0	0	crossflow
transitionTime						
CaseB_xrt	tBeg	tEnd	GroutInner_flow	GroutOuter_flow	Group0003_flow	comment
CaseB_xrt	0	100	0.01	0.01	0.01	pore volumes in this TI
PORFLOW			0.01	0.01	0.01	cumulative through this TI
CaseB_xrt	100	1000	0.36	0.21	0.24	pore volumes in this TI
PORFLOW			0.37	0.22	0.25	cumulative through this TI
CaseB_xrt	1000	10000	6.30	3.59	4.17	pore volumes in this TI
PORFLOW			6.66	3.81	4.42	cumulative through this TI
CaseB_xrt	10000	100000	6222.99	7479.82	7210.52	pore volumes in this TI
PORFLOW			6229.66	7483.63	7214.94	cumulative through this TI
poreVolumes			0	0	0	crossflow
transitionTime						

Notes:
1) PORFLOW darcyV, darcyU, sat values are taken from STAT.out
2) PORFLOW flow values are taken from FLUX.out
3) PORFLOW pore values are taken from FLUXC.out
4) PORFLOW apparently give equal weight to each cell, i.e., does not volume weight. For uniform xyz grid, GoldSimFlows and PORFLOW produce the same averages. For uniform xrt grid, they are different due to cylindrical coordinates
5) “nd” = no data; “nc” = no calculation

Table 3. Comparison of GoldSimFlows output to reference values for software test cases "CaseB_xrt", zone "DOMAIN", and time period 0-100 yrs.

radius	area	weight	sJ1	sJ2	sJ3	sJ4	sJ5	sJ6	sJ7	sJ8	sJ9	avgSaturation
0			from MAIN.sav									
100	31415.93	0.008264	0.8364	0.6947	1.0000	1.0000	1.0000	1.0000	1.0000	0.9080	0.7579	
200	94247.78	0.024793	0.8365	0.6951	1.0000	1.0000	1.0000	1.0000	1.0000	0.9058	0.7574	
300	157079.6	0.041322	0.8366	0.6960	1.0000	1.0000	1.0000	1.0000	1.0000	0.9012	0.7564	
400	219911.5	0.057851	0.8369	0.6978	1.0000	1.0000	1.0000	1.0000	1.0000	0.8939	0.7547	
500	282743.3	0.07438	0.8375	0.7011	1.0000	1.0000	1.0000	1.0000	1.0000	0.8831	0.7523	
600	345575.2	0.090909	0.8387	0.7070	1.0000	1.0000	1.0000	1.0000	1.0000	0.8679	0.7487	
700	408407	0.107438	0.8407	0.7172	1.0000	1.0000	1.0000	1.0000	1.0000	0.8463	0.7436	
800	471238.9	0.123967	0.8441	0.7339	1.0000	1.0000	1.0000	1.0000	1.0000	0.8148	0.7366	
900	534070.8	0.140496	0.8487	0.7600	0.7514	0.7495	0.7504	0.7532	0.7580	0.7662	0.7276	
1000	596902.6	0.157025	0.8512	0.7666	0.7506	0.7466	0.7454	0.7447	0.7433	0.7389	0.7206	
1100	659734.5	0.173554	0.8521	0.7679	0.7495	0.7441	0.7414	0.7385	0.7342	0.7273	0.7171	
sum	**3801327**	1										**0.8361**

radius	area	weight	vJ1	vJ2	vJ3	vJ4	vJ5	vJ6	vJ7	vJ8	vJ9	avgDarcyV
0			from MAIN.sav									
100	31415.93	0.008264	-0.1860	-0.0697	-0.0099	-0.0106	-0.0114	-0.0111	-0.0108	-4.8381	-9.8326	
200	94247.78	0.024793	-0.2339	-0.0860	-0.0099	-0.0110	-0.0120	-0.0114	-0.0108	-4.8301	-9.8247	
300	157079.6	0.041322	-0.3499	-0.1256	-0.0112	-0.0132	-0.0134	-0.0119	-0.0107	-4.8121	-9.8067	
400	219911.5	0.057851	-0.5836	-0.2050	-0.0049	-0.0017	-0.0016	-0.0051	-0.0080	-4.7800	-9.7760	
500	282743.3	0.07438	-1.0409	-0.3687	-0.0105	-0.0118	-0.0123	-0.0115	-0.0108	-4.7350	-9.7296	
600	345575.2	0.090909	-1.9379	-0.7043	-0.0093	-0.0108	-0.0122	-0.0114	-0.0107	-4.6737	-9.6684	
700	408407	0.107438	-3.7280	-1.4407	-0.0107	-0.0138	-0.0143	-0.0119	-0.0105	-4.6182	-9.6129	
800	471238.9	0.123967	-7.3839	-3.2008	-0.0043	-0.0024	-0.0020	-0.0046	-0.0084	-4.6686	-9.6635	
900	534070.8	0.140496	-15.0494	-19.2202	-22.6935	-23.2809	-24.5726	-26.8709	-31.1206	-22.2036	-10.2536	
1000	596902.6	0.157025	-18.4420	-20.1252	-21.1369	-21.1715	-21.0262	-20.5563	-19.1146	-14.3992	-10.3858	
1100	659734.5	0.173554	-19.3872	-20.0323	-20.0989	-19.5906	-18.6753	-17.2396	-15.1021	-12.2864	-10.3690	
sum	**3801327**	1										**-10.0000**

radius	area	weight	uJ1	uJ2	uJ3	uJ4	uJ5	uJ6	uJ7	uJ8	uJ9	avgDarcyU
0			from MAIN.sav									
100	31415.93	0.008264	-0.0564	-0.0598	0.0000	0.0007	0.0001	-0.0004	0.0000	4.8272	0.1674	
200	94247.78	0.024793	-0.1093	-0.1160	0.0000	0.0015	0.0000	-0.0008	0.0000	8.0375	0.2869	
300	157079.6	0.041322	-0.2172	-0.2311	0.0015	0.0021	-0.0004	-0.0021	0.0001	12.5154	0.4706	
400	219911.5	0.057851	-0.4135	-0.4455	-0.0003	0.0010	0.0002	0.0005	0.0001	17.1411	0.6982	
500	282743.3	0.07438	-0.7753	-0.8595	-0.0010	0.0005	0.0008	0.0018	0.0001	21.7678	0.9876	
600	345575.2	0.090909	-1.4289	-1.6922	0.0000	0.0022	0.0006	0.0000	0.0002	26.3383	1.3609	
700	408407	0.107438	-2.5202	-3.4518	0.0019	0.0043	-0.0002	-0.0022	0.0003	30.8394	1.8193	
800	471238.9	0.123967	-3.9128	-7.4284	0.0025	0.0037	-0.0001	-0.0014	0.0022	35.3791	2.2487	
900	534070.8	0.140496	-3.5025	-6.0245	0.1251	0.4660	0.8286	1.4706	2.7831	23.6307	2.0276	
1000	596902.6	0.157025	-1.6638	-1.4455	0.2865	0.8025	1.3662	2.2774	3.8892	7.0503	1.2015	
1100	659734.5	0.173554	-0.4200	-0.2252	0.1585	0.3498	0.5656	0.8701	1.2674	1.5483	0.3690	
sum	**3801327**	1										**1.9642**

Acceptance Testing

Acceptance testing of the installed GoldSimFlows software shall be conducted upon initial installation on each computer system using either the cylindrical or Cartesian coordinates example described above. The output from the test case run is considered acceptable and the software is ready for use if the comparison produces identical values with those found in Table 2 for the case selected.

4.5 CONFIGURATION CONTROL

Configurations of GoldSimFlows will be identified and documented by the CTF through paper and electronic copies of the source code kept on file. Records of superseded configurations will be retained in the software file indefinitely.

4.6 ACCESS CONTROL

Operation of GoldSimFlows will be overseen by the Manager, Radiological Performance Assessments, who will assign tasks as needed, and by the CTF, who will ensure assigned tasks are consist with the model capabilities. Access to the software will be limited to the CTF. Access will be controlled by residence of the software on SRNL High Performance Computing (HPC) file storage space to which only the CTF (and system administrators) has access. Should additional Users be desired, the SQAP will be modified to include development of a User Manual, and an expanded access control method. Access for other potential business programs will be controlled by the respective Manager.

4.7 CYBER SECURITY CONTROL

The HPC system is a part of SRNL's accreditation boundary which meets cyber security requirements for the site (SRS Manual 10Q). HPC cyber security will control access to and use of this software.

4.8 PROBLEM REPORTING AND CORRECTIVE ACTION

The CTF shall report software problems/issues to the Manager, Radiological Performance Assessment. The Manager will notify recipients of information generated by GoldSimFlows, and with the input from the CTF and information recipients, determine appropriate corrective action. Problem reporting and corrective action for other potential business programs will be controlled by the respective Manager.

4.9 SOFTWARE RETIREMENT

If the CTF decides that this sotware is not needed, the aforementioned software shall no longer be governed by this Software Quality Assurance Plan. At that time, support for the software shall be terminated and routine use of the software shall be prevented in association with PA or CA-related work by deleting the software from all SRS computers on which it resides.

5.0 REFERENCES

Butcher, B. T., 2013, OSR 19-337, *Software Classification Document*, Classification Document Number B-SWCD-A-00641 "GoldSimFlows", May 28, 2013.

DOE, 1999, *USDOE Order 435.1 Radioactive Waste Management Manual*, U. S. Department of Energy, July 9, 1999.

Flach, G. P., 2004, *Groundwater Flow Model of the General Separations Area Using PORFLOW (U)*, WSRC-TR-2004-00106, Rev. 0., July 15, 2004.

Jordan J., G. Flach and D. Schep, 2010, *PORFLOW Modeling Supporting the H-Tank Farm Performance Assessment*, SRNL-L6200-2010-00026, Rev. 1, November 18, 2010.

Jordan, J. M., and G. P. Flach, *PORFLOW Modeling Supporting the FY13 Saltstone Special Analysis*, SRNL-STI-2013-00280, Rev. 0, May 2013.

SRS, 2013, *1Q Quality Assurance Manual*, Procedure 20-1, "Software Quality Assurance", Revision 14, October 15, 2012.

APPENDIX A: GOLDSIMFLOWS FORTRAN90 SOURCE CODE

```
program GoldSimFlows

use stuff

implicit none

integer, parameter :: LONG=SELECTED_REAL_KIND(9,99)
integer, parameter :: LineLength=200, FileNameLength=80, PathLength=500, &
                      PathPlusFileLength=581, maxMtyp=50, maxZones=100, maxTI=100
integer, parameter :: isup=10, &
                      isav=11, &
                      ilog=21, &
                      iout=22, &
                      itec=23, &
                      itab=24, &
                      itab2=25, &
                      itab3=26
integer :: iostat_flag, iunt
integer :: npx,npy, nnx,nny, nex,ney
integer :: i,j,k, l, m,mm, n,nn, nTmp, nTI, iBeg, iEnd
integer :: im,jm, ip,jp, nZone, nGroup, iGroupPrev
integer :: inMin, inMax, jnMin, jnMax
integer :: ieMin, ieMax, jeMin, jeMax
integer :: ipMin, ipMax, jpMin, jpMax
integer, dimension(:), allocatable :: iTmp
integer, dimension(:,:), allocatable :: mtyp
integer, dimension(maxZones) :: iLower, jLower, iUpper, jUpper, iGroup, iGroup2

real(LONG) :: zero, pi, twoPi
real(LONG) :: time, tBeg, tEnd
real(LONG) :: xmm,xpm,xmp,xpp, ymm,ypm,ymp,ypp
real(LONG) :: flowMinimum, fluxMinimum
real(LONG) :: flowIn, flowOut, flowNet, &
              flowIn2, flowOut2, flowNet2
real(LONG) :: dvol, darcyV, darcyH, volume, volumePore, volumeWater, &
              poreCount, saturation, darcyVert, darcyHori, &
              crossflow, darea, dflow, infiltration, fullGeometry

real(LONG), dimension(:), allocatable :: Tmp
real(LONG), dimension(:,:), allocatable :: xc,yc, x,y, p,s,u,v,mois, &
                                           fcxm,fcxp,fcym,fcyp
real(LONG), dimension(:,:), allocatable :: fxxm,fxxp,fxym,fxyp
real(LONG), dimension(:,:), allocatable :: dxm,dxp,dym,dyp
real(LONG), dimension(:,:), allocatable :: y_xm,y_xp,y_ym,y_yp

real(LONG), dimension(4) :: area, flow, flux
real(LONG), dimension(4,2) :: areaIO, flowIO
real(LONG), dimension(4,maxMtyp) :: areaMtyp, flowMtyp, fluxMtyp
real(LONG), dimension(maxMtyp) :: volumeMtyp, volumePoreMtyp, volumeWaterMtyp, &
                                  saturationMtyp,  darcyVertMtyp,  darcyHoriMtyp
real(LONG), dimension(2,maxTI) :: times
real(LONG), dimension(maxTI) :: infiltrations
real(LONG), dimension(maxZones*2) :: volumes, Eh, pH, xflow, EhTime, pHtime
real(LONG), dimension(maxZones*2,4) :: areas
real(LONG), dimension(maxZones*2,maxTI,4) :: flows, fluxes
real(LONG), dimension(maxZones*2,maxTI) :: flowsIn, flowsIn2, saturations, &
                                           velocitiesV, velocitiesH, poreCounts, &
                                           poreCumulative, volumesPore, &
                                           volumesWater
```

```
character(len=FileNameLength) :: FileName, SavFileName
character(len=PathLength) :: Path, PathOutput
character(len=PathPlusFileLength) :: PathPlusFile
character(len=LineLength) :: Line
character(len=3) :: xyz
character(len=80) :: string, string2, tag, sZone
character(len=1) :: tab
character(len=4) :: TI
character(len=40), dimension(maxZones*2) :: zoneName
character(len=10000) :: sGoldSim
character(len=5), dimension(4) :: sideName

logical :: exit_flag, firstTI, first, writeCrossflow


!DEFINE CONSTANTS
tab = "      "
zero = 0._LONG
pi = 2._LONG*acos(zero)
twoPi = 2._LONG*pi
sideName(1) = "Lower"
sideName(2) = "Upper"
sideName(3) = "Left "
sideName(4) = "Right"

!READ HIGH-LEVEL SPECIFICATIONS; OPEN OUTPUT FILES
if (.false.) then
  iunt = isup
  open (unit=iunt, file='debug.sup', status='old', action='read')
else
  iunt = 5
end if

read(iunt,'(a)') Path

read(iunt,'(a)') SavFileName

read(iunt,*) nTI

read(iunt,'(a)') xyz

if      (xyz .eq. 'xyz') then
  !fullGeometry = 2._LONG * 600._LONG * 30.48_LONG !2 halves x 600' length in cm
  read(iunt,*) fullGeometry
else if (xyz .eq. 'xrt') then
  fullGeometry = twoPi
else
  stop "invalid xyz value"
end if

read(iunt,'(a)') PathOutput

read(iunt,'(a)') FileName
PathPlusFile = trim(PathOutput)//"/"//trim(Filename)
open (unit=ilog, file=PathPlusFile, status='unknown', action='write')

read(iunt,'(a)') FileName
PathPlusFile = trim(PathOutput)//"/"//trim(Filename)
open (unit=iout, file=PathPlusFile, status='unknown', action='write')

read(iunt,'(a)') FileName
PathPlusFile = trim(PathOutput)//"/"//trim(Filename)
open (unit=itec, file=PathPlusFile, status='unknown', action='write')
```

```
read(iunt,'(a)') FileName
PathPlusFile = trim(PathOutput)//"/"//trim(Filename)
open (unit=itab, file=PathPlusFile, status='unknown', action='write')

read(iunt,'(a)') FileName
PathPlusFile = trim(PathOutput)//"/"//trim(Filename)
open (unit=itab2, file=PathPlusFile, status='unknown', action='write')

read(iunt,'(a)') FileName
PathPlusFile = trim(PathOutput)//"/"//trim(Filename)
open (unit=itab3, file=PathPlusFile, status='unknown', action='write')

read(iunt,'(a)') tag

read(iunt,*) writeCrossflow

n = 0
do
  n = n + 1
  if      (xyz .eq. 'xyz') then
    read (iunt,*,iostat=iostat_flag) &
      zoneName(n), iLower(n), jLower(n), iUpper(n), jUpper(n), &
      Eh(n), pH(n), xflow(n), iGroup(n)
  else if (xyz .eq. 'xrt') then
    read (iunt,*,iostat=iostat_flag) &
      zoneName(n), jLower(n), iLower(n), jUpper(n), iUpper(n), &
      Eh(n), pH(n), xflow(n), iGroup(n)
  end if
  call IostatCheck (iostat_flag,"iunt", exit_flag)
  if (exit_flag) exit  !end-of-file
end do
nZone = n-1

!IDENTIFY GROUPS
first = .true.
nGroup = 0
iGroupPrev = 0
do n=1, nZone
  if (iGroup(n) .ne. 0) then
    if (first) then
      nGroup = nGroup + 1
      iGroup2(nGroup) = iGroup(n)
      iGroupPrev = iGroup(n)
      first = .false.
    else if (iGroup(n) .ne. iGroupPrev) then
      nGroup = nGroup + 1
      iGroup2(nGroup) = iGroup(n)
      iGroupPrev = iGroup(n)
    else
      continue
    end if
  end if
end do

!WRITE FILE HEADERS
write(iout,'(a)') "= = = = = Flow fields = = = = ="
write(iout,'(2a)')    "  path: ", trim(Path)//"/"//"TInn/"
write(iout,'(a,i2)')        "   TIs: ", nTI
write(iout,'(2a)')    "  file: ", trim(SavFileName)
write(iout,'(2a)')    "  geom: ", trim(xyz)
write(iout,'(a,1pe10.3,a)')&
                         "factor: ", fullGeometry, " (fullGeometry factor)"
```

```
write(iout,'(a)')    ""

if (nGroup .gt. 0) then
  write(iout,'(a)') "= = = = = Groups = = = = ="
  write(iout,'(a,i4)') "nGroup: ", nGroup
  do n=1, nGroup
    write(iout,'(a,i4)') "Group: ", iGroup2(n)
    do m=1, nZone
      if (iGroup(m) .eq. iGroup2(n)) then
        write(iout,'(a, i4)') "      Zone: ", m
      end if
    end do
  end do
end if
write(iout,'(a)') ""

write(itab,'(a)') &
  "id  tBeg   tEnd   zone   material     boundary     flow"


!LOOP OVER TIs = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
firstTI = .true.
do m=1, nTI
write(ilog,*) "TI: ", m

!OPEN PORFLOW SAVE FILE
write(string,'(i2)') m
if (m .lt. 10) then
  TI = "TI0"//trim(adjustl(string))
else
  TI = "TI"//trim(adjustl(string))
end if
PathPlusFile = trim(Path)//"/"//trim(TI)//"/"//trim(SavFileName)

open (unit=isav, file=PathPlusFile, status='old', action='read')

if (firstTI) then
  !READ MESH SIZE FROM PORFLOW ARCHIVE FILE
  !note: for 'xyz', PORFLOW "X" is x
  !                 PORFLOW "Y" is y
  !      for 'xrt', PORFLOW "X" is y
  !                 PORFLOW "Y" is r (x)
  !
  do
    read (isav,'(a)',iostat=iostat_flag) Line
    call IostatCheck (iostat_flag,"isav", exit_flag)
    if (exit_flag) exit  !end-of-file

    if       (index(Line,"X DIRECTION NODES") .ne. 0) then
      read (Line,*) npx
    else if  (index(Line,"Y DIRECTION NODES") .ne. 0) then
      read (Line,*) npy
      exit
    end if
  end do
  write(ilog,*) "npx, npy, npx*npy: ", npx, npy, npx*npy

  nnx = npx - 1
  nny = npy - 1
  write(ilog,*) "nnx, nny, nnx*nny: ", nnx, nny, nnx*nny

  nex = nnx - 1
  ney = nny - 1
```

```
  write(ilog,*) "nex, ney, nex*ney: ", nex, ney, nex*ney

  write(iout,'(a)') "= = = = = Grid Size = = = = ="
  if      (xyz .eq. 'xyz') then
    write(iout,'(a)') "Cartesian coordinates (xyz)"
  else if (xyz .eq. 'xrt') then
    write(iout,'(a)') "Cylindrical coordinates (xrt)"
    write(iout,'(a)') "Tecplot indices swapped relative to PORFLOW"
    write(iout,'(a)') "First index = horizontal; second index = vertical"
  end if

  if      (xyz .eq. 'xyz') then
    write(iout,'(a,2i6)') "element indices: ", nex, ney
    write(iout,'(a,2i6)') "  nodal indices: ", nnx, nny
    write(iout,'(a,2i6)') "PORFLOW indices: ", npx, npy
  else if (xyz .eq. 'xrt') then
    write(iout,'(a,2i6)') "element indices: ", ney, nex
    write(iout,'(a,2i6)') "  nodal indices: ", nny, nnx
    write(iout,'(a,2i6)') "PORFLOW indices: ", npy, npx
  end if
  write(iout,'(a)') ""

  !ALLOCATE ARRAYS
  allocate ( Tmp(npx*npy*4))
  allocate (iTmp(npx*npy*4))

  allocate (xc(nnx,nny))
  allocate (yc(nnx,nny))

  allocate (   x(0:nnx,0:nny))
  allocate (   y(0:nnx,0:nny))
  allocate (mtyp(0:nnx,0:nny))
  allocate (   p(0:nnx,0:nny))
  allocate (   s(0:nnx,0:nny))
  allocate (   u(0:nnx,0:nny))
  allocate (   v(0:nnx,0:nny))
  allocate (mois(0:nnx,0:nny))

  allocate (fcxm(nex,ney))
  allocate (fcxp(nex,ney))
  allocate (fcym(nex,ney))
  allocate (fcyp(nex,ney))

  allocate (fxxm(nex,ney))
  allocate (fxxp(nex,ney))
  allocate (fxym(nex,ney))
  allocate (fxyp(nex,ney))

  allocate (dxm(nex,ney))
  allocate (dxp(nex,ney))
  allocate (dym(nex,ney))
  allocate (dyp(nex,ney))

  allocate (y_xm(nex,ney))
  allocate (y_xp(nex,ney))
  allocate (y_ym(nex,ney))
  allocate (y_yp(nex,ney))

  firstTI = .false.
end if

!READ TIMES FROM ARCHIVE FILE
rewind(isav)
```

```
do
  read (isav,'(a)',iostat=iostat_flag) Line
  call IostatCheck (iostat_flag,"isav", exit_flag)
  if (exit_flag) exit  !end-of-file
  if       (index(Line,"PROBLEM TITLE:") .ne. 0) then
    iBeg = index(Line,",",.true.) + 2
    iEnd = index(Line,"-",.true.) - 1
    read(Line(iBeg:iEnd),*) tBeg

    iBeg = index(Line,"-",.true.) + 1
    iEnd = index(Line,"YRS",.true.) - 1
    read(Line(iBeg:iEnd),*) tEnd

    exit
  end if
end do

!WRITE FLOW FIELD IDENTIFIER
write(iout,'(a)') "= = = = = Flow field = = = = ="
write(iout,'(2a)')  "  file: ", trim(PathPlusFile)
write(iout,'(a,f12.0)')    "  tBeg: ", tBeg
write(iout,'(a,f12.0)')    "  tEnd: ", tEnd
write(iout,'(a)')   ""

!READ ARRAYS FROM PORFLOW ARCHIVE FILE
rewind(isav)
call getReal("XC  ",isav,Tmp,nTmp,time)
if (nTmp .ne. nnx*nny) stop "(nTmp .ne. nnx*nny)"
write(ilog,*) 'XC'
call stuffMatrixReal(1,nnx,1,nny,Tmp,xc)
write(ilog,*) xc(1,1), xc(nnx/2,nny/2), xc(nnx,nny)

rewind(isav)
call getReal("YC  ",isav,Tmp,nTmp,time)
if (nTmp .ne. nnx*nny) stop "(nTmp .ne. nnx*nny)"
write(ilog,*) 'YC'
call stuffMatrixReal(1,nnx,1,nny,Tmp,yc)
write(ilog,*) yc(1,1), yc(nnx/2,nny/2), yc(nnx,nny)

rewind(isav)
call getReal("X   ",isav,Tmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
write(ilog,*) 'X'
call stuffMatrixReal(0,nnx,0,nny,Tmp,x)
write(ilog,*) x(0,0), x(nnx/2,nny/2), x(nnx,nny)

rewind(isav)
call getReal("Y   ",isav,Tmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
write(ilog,*) 'Y'
call stuffMatrixReal(0,nnx,0,nny,Tmp,y)
write(ilog,*) y(0,0), y(nnx/2,nny/2), y(nnx,nny)

rewind(isav)
call getInteger("MTYP",isav,iTmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
write(ilog,*) 'MTYP'
call stuffMatrixInteger(0,nnx,0,nny,iTmp,mtyp)
write(ilog,*) mtyp(0,0), mtyp(nnx/2,nny/2), mtyp(nnx,nny)

rewind(isav)
call getReal("P   ",isav,Tmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
```

```
write(ilog,*) 'P'
call stuffMatrixReal(0,nnx,0,nny,Tmp,p)
write(ilog,*) p(0,0), p(nnx/2,nny/2), p(nnx,nny)

rewind(isav)
call getReal("S   ",isav,Tmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
write(ilog,*) 'S'
call stuffMatrixReal(0,nnx,0,nny,Tmp,s)
write(ilog,*) s(0,0), s(nnx/2,nny/2), s(nnx,nny)

rewind(isav)
call getReal("U   ",isav,Tmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
write(ilog,*) 'U'
call stuffMatrixReal(0,nnx,0,nny,Tmp,u)
write(ilog,*) u(0,0), u(nnx/2,nny/2), u(nnx,nny)

rewind(isav)
call getReal("V   ",isav,Tmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
write(ilog,*) 'V'
call stuffMatrixReal(0,nnx,0,nny,Tmp,v)
write(ilog,*) v(0,0), v(nnx/2,nny/2), v(nnx,nny)

rewind(isav)
call getReal("MOIS",isav,Tmp,nTmp,time)
if (nTmp .ne. npx*npy) stop "(nTmp .ne. npx*npy)"
write(ilog,*) 'MOIS'
call stuffMatrixReal(0,nnx,0,nny,Tmp,mois)
write(ilog,*) mois(0,0), mois(nnx/2,nny/2), mois(nnx,nny)

rewind(isav)
call getReal("FC  ",isav,Tmp,nTmp,time)
write(ilog,*) 'FC'
n = 0
do j=1,ney
  do i=1,nex
    if (i .eq. 1) then
      n = n + 1; fcxm(i,j) = Tmp(n)
    else
                fcxm(i,j) = fcxp(i-1,j)
    end if
      n = n + 1; fcxp(i,j) = Tmp(n)
    if (j .eq. 1) then
      n = n + 1; fcym(i,j) = Tmp(n)
    else
                fcym(i,j) = fcyp(i,j-1)
    end if
      n = n + 1; fcyp(i,j) = Tmp(n)
  end do
end do
if (nTmp .ne. n) stop "(nTmp .ne. n)"
write(ilog,*) fcxm(1,1)    , fcxp(1,1)    , fcym(1,1)    , fcyp(1,1)
write(ilog,*) fcxm(nex,ney), fcxp(nex,ney), fcym(nex,ney), fcyp(nex,ney)


!FLUX CALCULATIONS
!note: for 'xyz', PORFLOW "X" is x
!                 PORFLOW "Y" is y
!      for 'xrt', PORFLOW "X" is y
!                 PORFLOW "Y" is r (x)
do j=1,ney
```

```
  do i=1,nex
    im = i
    ip = i+1
    jm = j
    jp = j+1

    xmm = xc(im,jm)
    xpm = xc(ip,jm)
    xmp = xc(im,jp)
    xpp = xc(ip,jp)

    ymm = yc(im,jm)
    ypm = yc(ip,jm)
    ymp = yc(im,jp)
    ypp = yc(ip,jp)

    dxm(i,j) = sqrt((xpm - xmm)**2 + (ypm - ymm)**2)
    dxp(i,j) = sqrt((xpp - xmp)**2 + (ypp - ymp)**2)

    dym(i,j) = sqrt((xmp - xmm)**2 + (ymp - ymm)**2)
    dyp(i,j) = sqrt((xpp - xpm)**2 + (ypp - ypm)**2)

    if (xyz .eq. 'xyz') then
      fxxm(i,j) = fcxm(i,j)/dym(i,j) !not used
      fxxp(i,j) = fcxp(i,j)/dyp(i,j)
      fxym(i,j) = fcym(i,j)/dxm(i,j)
      fxyp(i,j) = fcyp(i,j)/dxp(i,j)
      infiltration = -fxyp(nex,ney)
    else if (xyz .eq. 'xrt') then
      y_xm(i,j) = 0.5*(ymm + ymp)
      y_xp(i,j) = 0.5*(ypm + ypp)
      y_ym(i,j) = 0.5*(ymm + ypm)
      y_yp(i,j) = 0.5*(ymp + ypp)

      fxxm(i,j) = fcxm(i,j)/   (dym(i,j)*y_xm(i,j)) !not used
      fxxp(i,j) = fcxp(i,j)/   (dyp(i,j)*y_xp(i,j))
      fxym(i,j) = fcym(i,j)/max(dxm(i,j)*y_ym(i,j),1.e-20)
      fxyp(i,j) = fcyp(i,j)/   (dxp(i,j)*y_yp(i,j))
      infiltration = -fxxp(nex,ney)
    end if
  end do
end do

!LOOP OVER GRID ZONES + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + +
!note: for 'xyz', PORFLOW "X" is x
!                 PORFLOW "Y" is y
!      for 'xrt', PORFLOW "X" is y
!                 PORFLOW "Y" is r (x)
!note: iMin, iMax, jMin, jMax are Tecplot corner node indices

do n=1, nZone
  sZone = trim(zoneName(n))
  crossflow = xflow(n)

  inMin = iLower(n)
  jnMin = jLower(n)
  inMax = iUpper(n)
  jnMax = jUpper(n)

  ieMin = inMin
  jeMin = jnMin
  ieMax = inMax - 1
  jeMax = jnMax - 1
```

```
ipMin = ieMin + 1
ipMax = ieMax + 1
jpMin = jeMin + 1
jpMax = jeMax + 1

write(iout,'(a,a,a)') "= = = = = Zone: ", trim(sZone), " = = = = ="
write(iout,'(a)') "Requested indices"
if      (xyz .eq. 'xyz') then
  write(iout,'(a,4i6)') "(Tecplot)nodal: ", inMin, jnMin, inMax, jnMax
  write(iout,'(a,4i6)') "       element: ", ieMin, jeMin, ieMax, jeMax
  write(iout,'(a,4i6)') "       PORFLOW: ", ipMin, jpMin, ipMax, jpMax
else if (xyz .eq. 'xrt') then
  write(iout,'(a,4i6)') "(Tecplot)nodal: ", jnMin, inMin, jnMax, inMax
  write(iout,'(a,4i6)') "       element: ", jeMin, ieMin, jeMax, ieMax
  write(iout,'(a,4i6)') "       PORFLOW: ", jpMin, ipMin, jpMax, ipMax
end if

ieMin = min(max(1,inMin),nex)
jeMin = min(max(1,jnMin),ney)
ieMax = max(min(nex,ieMax),ieMin)
jeMax = max(min(ney,jeMax),jeMin)

write(iout,'(a)') "  (checking indices)"
if      (xyz .eq. 'xyz') then
  if (ieMin.ne.inMin  ) write(iout,'(a,i4,a,i4)') &
    "  Warning: X- boundary moved from nodal index ",inMin," to ",ieMin
  if (jeMin.ne.jnMin  ) write(iout,'(a,i4,a,i4)') &
    "  Warning: Y- boundary moved from nodal index ",jnMin," to ",jeMin
  if (ieMax.ne.inMax-1) write(iout,'(a,i4,a,i4)') &
    "  Warning: X+ boundary moved from nodal index ",inMax," to ",ieMax+1
  if (jeMax.ne.jnMax-1) write(iout,'(a,i4,a,i4)') &
    "  Warning: Y+ boundary moved from nodal index ",jnMax," to ",jeMax+1
else if (xyz .eq. 'xrt') then
  if (jeMin.ne.jnMin  ) write(iout,'(a,i4,a,i4)') &
    "  Warning: X- boundary moved from nodal index ",jnMin," to ",jeMin
  if (ieMin.ne.inMin  ) write(iout,'(a,i4,a,i4)') &
    "  Warning: Y- boundary moved from nodal index ",inMin," to ",ieMin
  if (jeMax.ne.jnMax-1) write(iout,'(a,i4,a,i4)') &
    "  Warning: X+ boundary moved from nodal index ",jnMax," to ",jeMax+1
  if (ieMax.ne.inMax-1) write(iout,'(a,i4,a,i4)') &
    "  Warning: Y+ boundary moved from nodal index ",inMax," to ",ieMax+1
end if

inMin = ieMin
jnMin = jeMin
inMax = ieMax + 1
jnMax = jeMax + 1

ipMin = ieMin + 1
ipMax = ieMax + 1
jpMin = jeMin + 1
jpMax = jeMax + 1

write(iout,'(a)') "Accepted indices"
if      (xyz .eq. 'xyz') then
  write(iout,'(a,4i6)') "(Tecplot)nodal: ", inMin, jnMin, inMax, jnMax
  write(iout,'(a,4i6)') "       element: ", ieMin, jeMin, ieMax, jeMax
  write(iout,'(a,4i6)') "       PORFLOW: ", ipMin, jpMin, ipMax, jpMax
else if (xyz .eq. 'xrt') then
  write(iout,'(a,4i6)') "(Tecplot)nodal: ", jnMin, inMin, jnMax, inMax
  write(iout,'(a,4i6)') "       element: ", jeMin, ieMin, jeMax, ieMax
  write(iout,'(a,4i6)') "       PORFLOW: ", jpMin, ipMin, jpMax, ipMax
```

```
  end if
  write(iout,'(a)') ""

  !min flow and flux for division
  flowMinimum = 1.e-20
  fluxMinimum = 1.e-20

  !For each boundary compute areas, flows, and fluxes
  area = zero
  flow = zero
  areaIO = zero
  flowIO = zero
  areaMtyp = -999._LONG
  flowMtyp = zero

  !Lower boundary
  k = 1
  if      (xyz .eq. 'xyz') then
    j = jeMin
    do i=ieMin,ieMax
      darea =  dxm(i,j)
      dflow = fcym(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .ge. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) + dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) - dflow
      end if
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  else if (xyz .eq. 'xrt') then
    i = ieMin
    do j=jeMin,jeMax
      darea =  dym(i,j)*y_xm(i,j)
      dflow = fcxm(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .ge. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) + dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) - dflow
      end if
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  end if
  flux(k) = flow(k)/area(k)
  do l=1,maxMtyp
    if (areaMtyp(k,l) .gt. zero) fluxMtyp(k,l) = flowMtyp(k,l)/areaMtyp(k,l)
  end do

  !Upper boundary
  k = 2
```

```
  if      (xyz .eq. 'xyz') then
    j = jeMax
    do i=ieMin,ieMax
      darea =  dxp(i,j)
      dflow = fcyp(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .le. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) - dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) + dflow
      end if
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  else if (xyz .eq. 'xrt') then
    i = ieMax
    do j=jeMin,jeMax
      darea =  dyp(i,j)*y_xp(i,j)
      dflow = fcxp(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .le. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) - dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) + dflow
      end if
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  end if
  flux(k) = flow(k)/area(k)
  do l=1,maxMtyp
    if (areaMtyp(k,l) .gt. zero) fluxMtyp(k,l) = flowMtyp(k,l)/areaMtyp(k,l)
  end do

  !Left boundary
  k = 3
  if      (xyz .eq. 'xyz') then
    i = ieMin
    do j=jeMin,jeMax
      darea =  dym(i,j)
      dflow = fcxm(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .ge. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) + dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) - dflow
      end if
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
```

```
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  else if (xyz .eq. 'xrt') then
    j = jeMin
    do i=ieMin,ieMax
      darea = max(dxm(i,j)*y_ym(i,j),1.e-20)
      dflow =    fcym(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .ge. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) + dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) - dflow
      end if
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  end if
  flux(k)  = flow(k) /area(k)
  do l=1,maxMtyp
    if (areaMtyp(k,l) .gt. zero) fluxMtyp(k,l) = flowMtyp(k,l)/areaMtyp(k,l)
  end do

  !Right boundary
  k = 4
  if      (xyz .eq. 'xyz') then
    i = ieMax
    do j=jeMin,jeMax
      darea =  dyp(i,j)
      dflow = fcxp(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .le. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) - dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) + dflow
      end if
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  else if (xyz .eq. 'xrt') then
    j = jeMax
    do i=ieMin,ieMax
      darea =  dxp(i,j)*y_yp(i,j)
      dflow = fcyp(i,j)
      area(k) = area(k) + darea
      flow(k) = flow(k) + dflow
      if (dflow .le. zero) then
        areaIO(k,1) = areaIO(k,1) + darea
        flowIO(k,1) = flowIO(k,1) - dflow
      else
        areaIO(k,2) = areaIO(k,2) + darea
        flowIO(k,2) = flowIO(k,2) + dflow
      end if
```

```
      if (areaMtyp(k,mtyp(i,j)) .lt. zero) areaMtyp(k,mtyp(i,j)) = 0 !flag active
mtyp's
      areaMtyp(k,mtyp(i,j)) = areaMtyp(k,mtyp(i,j)) + darea
      flowMtyp(k,mtyp(i,j)) = flowMtyp(k,mtyp(i,j)) + dflow
    end do
  end if
  flux(k) = flow(k)/area(k)
  do l=1,maxMtyp
    if (areaMtyp(k,l) .gt. zero) fluxMtyp(k,l) = flowMtyp(k,l)/areaMtyp(k,l)
  end do

  !Flow balance using ***net*** flow from each boundary
  flowIn = zero
  flowOut = zero
  do k=1,4
    if      (k.eq.1 .or. k.eq.3) then !Lower and left boundaries
      if (flow(k) .ge. zero) then
        flowIn = flowIn + flow(k)
      else
        flowOut = flowOut - flow(k)
      end if
    else if (k.eq.2 .or. k.eq.4) then !Upper and right boundaries
      if (flow(k) .le. zero) then
        flowIn = flowIn - flow(k)
      else
        flowOut = flowOut + flow(k)
      end if
    end if
  end do
  flowNet = flowIn - flowOut

  !Flow balance 2 using ***in and out*** flows from each boundary
  flowIn2 = zero
  flowOut2 = zero
  do k=1,4
    flowIn2 = flowIn2 + flowIO(k,1)
    flowOut2 = flowOut2 + flowIO(k,2)
  end do
  flowNet2 = flowIn2 - flowOut2

  !Volume average saturation and vertical Darcy velocity
  dvol = zero
  darcyV = zero
  darcyH = zero

  volume = zero
  volumePore = zero
  volumeWater = zero
  saturation = zero
  darcyVert = zero
  darcyHori = zero

  volumeMtyp = -999._LONG
  volumePoreMtyp = -999._LONG
  volumeWaterMtyp = -999._LONG
  saturationMtyp = zero
  darcyVertMtyp = zero
  darcyHoriMtyp = zero

  do j=jeMin,jeMax
    do i=ieMin,ieMax
      if (volumeMtyp(mtyp(i,j)) .lt. zero) volumeMtyp(mtyp(i,j)) = 0 !flag active
mtyp's
```

```
      if      (xyz .eq. 'xyz') then
        dvol = 0.5*(dxm(i,j)+dxp(i,j)) * 0.5*(dym(i,j)+dyp(i,j)) * 1.0_LONG
        darcyV = v(i,j)
        darcyH = u(i,j)
      else if (xyz .eq. 'xrt') then
        dvol = 0.5*(dxm(i,j)+dxp(i,j)) * &
               0.5*(dym(i,j)*y_xm(i,j)+dyp(i,j)*y_xp(i,j))
        darcyV = u(i,j)
        darcyH = v(i,j)
      end if
      mm = mtyp(i,j)

      volume = volume + dvol
      volumeMtyp(mm) = volumeMtyp(mm) + dvol

      volumePore = volumePore + dvol*mois(i,j)/s(i,j)
      volumePoreMtyp(mm) = volumePoreMtyp(mm) + dvol*mois(i,j)/s(i,j)

      volumeWater = volumeWater + dvol*mois(i,j)
      volumeWaterMtyp(mm) = volumeWaterMtyp(mm) + dvol*mois(i,j)

      saturation = saturation + dvol*s(i,j)
      saturationMtyp(mm) = saturationMtyp(mm) + dvol*s(i,j)

      darcyVert = darcyVert + dvol*darcyV
      darcyVertMtyp(mm) = darcyVertMtyp(mm) + dvol*darcyV

      darcyHori = darcyHori + dvol*darcyH
      darcyHoriMtyp(mm) = darcyhoriMtyp(mm) + dvol*darcyH
    end do
  end do
  saturation = saturation/volume
  darcyVert = darcyVert/volume
  darcyHori = darcyHori/volume
  do l=1,maxMtyp
    if (volumeMtyp(l) .gt. zero) then
      saturationMtyp(l) = saturationMtyp(l)/volumeMtyp(l)
       darcyVertMtyp(l) =  darcyVertMtyp(l)/volumeMtyp(l)
       darcyHoriMtyp(l) =  darcyHoriMtyp(l)/volumeMtyp(l)
    end if
  end do

  !Pore volumes
  poreCount = (flowIn2*(tEnd - tBeg))/volumePore
  poreCount = poreCount*(1._LONG + crossflow/infiltration)

  !MAIN OUTPUT FILE . . . . . .

  !Flow balance for whole zone
  write(iout,'(a)') "Flow balance for unit dimension using NET boundary flows"
  write(iout,'(a,es12.3e3)') "   in flow: ", flowIn
  write(iout,'(a,es12.3e3)') "  out flow: ", flowOut
  write(iout,'(a,es12.3e3)') "  net flow: ", flowNet
  write(iout,'(a,f12.8,a)')  "  rel diff: ", flowNet/flowIn*100," %"
  write(iout,'(a)') ""
  write(iout,'(a)') "Flow balance for full geometry"
  write(iout,'(a,es12.3e3)') "   in flow: ", flowIn*fullGeometry
  write(iout,'(a,es12.3e3)') "  out flow: ", flowOut*fullGeometry
  write(iout,'(a,es12.3e3)') "  net flow: ", flowNet*fullGeometry
  write(iout,'(a,f12.8,a)')  "  rel diff: ", flowNet/flowIn*100," %"
  write(iout,'(a)') ""
  write(iout,'(a)') "Flow balance for unit dimension using IN and OUT boundary
flows"
```

```
write(iout,'(a,es12.3e3)') "   in flow: ", flowIn2
write(iout,'(a,es12.3e3)') "  out flow: ", flowOut2
write(iout,'(a,es12.3e3)') "  net flow: ", flowNet2
write(iout,'(a,f12.8,a)')  "  rel diff: ", flowNet2/flowIn2*100," %"
write(iout,'(a)') ""
write(iout,'(a)') "Flow balance for full geometry"
write(iout,'(a,es12.3e3)') "   in flow: ", flowIn2*fullGeometry
write(iout,'(a,es12.3e3)') "  out flow: ", flowOut2*fullGeometry
write(iout,'(a,es12.3e3)') "  net flow: ", flowNet2*fullGeometry
write(iout,'(a,f12.8,a)')  "  rel diff: ", flowNet/flowIn*100," %"
write(iout,'(a)') ""
write(iout,'(a,es12.3e3)') "Infiltration: ", infiltration
write(iout,'(a,es12.3e3)') "   Crossflow: ", crossflow
write(iout,'(a,es12.3e3)') "  Pore count: ", poreCount
write(iout,'(a)') ""


!Loop over boundaries writing areas, flows, and fluxes
do k=1,4
  write(iout,'(a)') trim(sideName(k))//" boundary"

  write(iout,'(a,es12.3e3)')   "  area: ", area(k)
  do l=1,maxMtyp
    write(string,*) l; string=adjustl(string)
    if (areaMtyp(k,l) .gt. zero) write(iout,'(a,es12.3e3,f10.5)') &
      "    mtyp"//trim(string)//": ", areaMtyp(k,l), areaMtyp(k,l)/area(k)
  end do

  if      (k.eq.1 .or. k.eq.3) then !Lower and left boundaries
    if (flow(k) .ge. zero) then
      string = " inflow"
    else
      string = " outflow"
    end if
  else if (k.eq.2 .or. k.eq.4) then !Upper and right boundaries
    if (flow(k) .ge. zero) then
      string = " outflow"
    else
      string = " inflow"
    end if
  end if

  write(iout,'(a,es12.3e3,a)') "  flow: ", flow(k), trim(string)
  do l=1,maxMtyp
    write(string,*) l; string=adjustl(string)
    if (areaMtyp(k,l) .gt. zero) write(iout,'(a,es12.3e3,f10.5)') &
      "    mtyp"//trim(string)//": ", flowMtyp(k,l), &
      flowMtyp(k,l)/(sign(1.0_LONG,flow(k))*max(abs(flow(k)),flowMinimum))
  end do

  write(iout,'(a,es12.3e3)')   "  flux: ", flux(k)
  do l=1,maxMtyp
    write(string,*) l; string=adjustl(string)
    if (areaMtyp(k,l) .gt. zero) write(iout,'(a,es12.3e3,f10.5)') &
      "    mtyp"//trim(string)//": ", fluxMtyp(k,l), &
      fluxMtyp(k,l)/(sign(1.0_LONG,flux(k))*max(abs(flux(k)),fluxMinimum))
  end do
end do

!Volume average saturation and vertical Darcy velocity
write(iout,'(a)') ""
write(iout,'(a)') "Volumes and volume-averages"
```

```
write(iout,'(2(a,es12.3e3),a)') &
  "  volume: ", volume, " per unit dimension,", &
  volume*fullGeometry, " full geometry"
do l=1,maxMtyp
  write(string,*) l; string=adjustl(string)
  if (volumeMtyp(l) .gt. zero) write(iout,'(a,es12.3e3,f10.5)') &
    "    mtyp"//trim(string)//": ", volumeMtyp(l), volumeMtyp(l)/volume
end do

write(iout,'(2(a,es12.3e3),a)') &
  "  volumePore: ", volumePore, " per unit dimension,", &
  volumePore*fullGeometry, " full geometry"
do l=1,maxMtyp
  write(string,*) l; string=adjustl(string)
  if (volumePoreMtyp(l) .gt. zero) &
    write(iout,'(a,es12.3e3,f10.5)') "    mtyp"//trim(string)//": ", &
      volumePoreMtyp(l), volumePoreMtyp(l)/volumePore
end do

write(iout,'(2(a,es12.3e3),a)') &
  "  volumeWater: ", volumeWater, " per unit dimension,", &
  volumeWater*fullGeometry, " full geometry"
do l=1,maxMtyp
  write(string,*) l; string=adjustl(string)
  if (volumeWaterMtyp(l) .gt. zero) &
    write(iout,'(a,es12.3e3,f10.5)') "    mtyp"//trim(string)//": ", &
      volumeWaterMtyp(l), volumeWaterMtyp(l)/volumeWater
end do

write(iout,'(a,es12.3e3)')   "  saturation: ", saturation
do l=1,maxMtyp
  write(string,*) l; string=adjustl(string)
  if (volumeMtyp(l) .gt. zero) write(iout,'(a,es12.3e3,f10.5)') &
    "    mtyp"//trim(string)//": ", saturationMtyp(l)
end do

write(iout,'(a,es12.3e3)')   "  Darcy vertical: ", darcyVert
do l=1,maxMtyp
  write(string,*) l; string=adjustl(string)
  if (volumeMtyp(l) .gt. zero) write(iout,'(a,es12.3e3,f10.5)') &
    "    mtyp"//trim(string)//": ", darcyVertMtyp(l)
end do

write(iout,'(a,es12.3e3)')   "  Darcy horizontal: ", darcyHori
do l=1,maxMtyp
  write(string,*) l; string=adjustl(string)
  if (volumeMtyp(l) .gt. zero) write(iout,'(a,es12.3e3,f10.5)') &
    "    mtyp"//trim(string)//": ", darcyHoriMtyp(l)
end do
write(iout,'(a)') ""

!Excel file (values per unit radian)
do k=1,4
  write(itab,'(a,2(a,f8.0),3(a,a),(a,es12.3e3))') &
    trim(tag), tab,tBeg, tab,tEnd, tab,trim(sZone), tab,"Zone", &
    tab,trim(sideName(k)), tab,flow(k)
  do l=1,maxMtyp
    write(string,*) l; string=adjustl(string)
    if (areaMtyp(k,l) .gt. zero) &
      write(itab,'(a,2(a,f8.0),3(a,a),(a,es12.3e3))') &
        trim(tag), tab,tBeg, tab,tEnd, tab,trim(sZone), &
        tab,"Mtyp"//trim(string), &
        tab,trim(sideName(k)), tab,flowMtyp(k,l)
```

```
    end do
  end do

  !Accumulate results for Excel and Tecplot (values per unit radian)
  !          (,TI)
        times(1,m) = tBeg
        times(2,m) = tEnd
  infiltrations(m) = infiltration

  !         (zone)
        volumes(n) = volume

  !    (zone,side)
        areas(n,1) = area(1)
        areas(n,2) = area(2)
        areas(n,3) = area(3)
        areas(n,4) = area(4)

  !      (zone,TI)
      flowsIn(n,m) = flowIn
     flowsIn2(n,m) = flowIn2
  saturations(n,m) = saturation
  velocitiesV(n,m) = darcyVert
  velocitiesH(n,m) = darcyHori
  volumesPore(n,m) = volumePore
 volumesWater(n,m) = volumeWater
   poreCounts(n,m) = poreCount

  ! (zone,TI,side)
      flows(n,m,1) = flow(1)
      flows(n,m,2) = flow(2)
      flows(n,m,3) = flow(3)
      flows(n,m,4) = flow(4)

     fluxes(n,m,1) = flux(1)
     fluxes(n,m,2) = flux(2)
     fluxes(n,m,3) = flux(3)
     fluxes(n,m,4) = flux(4)

!END LOOP OVER GRID ZONES  + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + +
end do

!END LOOP OVER TIs = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
end do

!ACCUMULATE GROUP RESULTS
if (nGroup .gt. 0) then
  do l=1, nGroup
    n = nZone + l

    write(string,*) iGroup2(l)
    if (iGroup2(l) .lt. 10) then
      zoneName(n) = "Group000"//trim(adjustl(string))
    else if (iGroup2(l) .lt. 100) then
      zoneName(n) = "Group00"//trim(adjustl(string))
    else if (iGroup2(l) .lt. 1000) then
      zoneName(n) = "Group0"//trim(adjustl(string))
    else
      zoneName(n) = "Group"//trim(adjustl(string))
    end if

    !         (zone)
          volumes(n) = 0
```

```
!    (zone,side)
       areas(n,1) = 0
       areas(n,2) = 0
       areas(n,3) = 0
       areas(n,4) = 0
do m=1,nTI
!      (zone,TI)
     flowsIn(n,m) = 0
    flowsIn2(n,m) = 0
 saturations(n,m) = 0
 velocitiesV(n,m) = 0
 velocitiesH(n,m) = 0
 volumesPore(n,m) = 0
volumesWater(n,m) = 0
  poreCounts(n,m) = 0
! (zone,TI,side)
     flows(n,m,1) = 0
     flows(n,m,2) = 0
     flows(n,m,3) = 0
     flows(n,m,4) = 0
    fluxes(n,m,1) = 0
    fluxes(n,m,2) = 0
    fluxes(n,m,3) = 0
    fluxes(n,m,4) = 0
end do

do nn=1, nZone
  if (iGroup(nn) .eq. iGroup2(l)) then
    !         (zone)
                Eh(n) =                   Eh(nn) !assumed to be the
                pH(n) =                   pH(nn) !same values for each
             xflow(n) =                xflow(nn) !member of group!
           volumes(n) = volumes(n) + volumes(nn)
    !    (zone,side)
           areas(n,1) = areas(n,1) + areas(nn,1)
           areas(n,2) = areas(n,2) + areas(nn,2)
           areas(n,3) = areas(n,3) + areas(nn,3)
           areas(n,4) = areas(n,4) + areas(nn,4)
    do m=1,nTI
    !      (zone,TI)
         flowsIn(n,m) =      flowsIn(n,m) +      flowsIn(nn,m)
        flowsIn2(n,m) =     flowsIn2(n,m) +     flowsIn2(nn,m)
     saturations(n,m) =  saturations(n,m) +  saturations(nn,m)*volumes(nn)
     velocitiesV(n,m) =  velocitiesV(n,m) +  velocitiesV(nn,m)*volumes(nn)
     velocitiesH(n,m) =  velocitiesH(n,m) +  velocitiesH(nn,m)*volumes(nn)
     volumesPore(n,m) =  volumesPore(n,m) +  volumesPore(nn,m)
    volumesWater(n,m) = volumesWater(n,m) + volumesWater(nn,m)
      poreCounts(n,m) =   poreCounts(n,m) +   poreCounts(nn,m)*volumes(nn)
    ! (zone,TI,side)
         flows(n,m,1) =  flows(n,m,1) +  flows(nn,m,1)
         flows(n,m,2) =  flows(n,m,2) +  flows(nn,m,2)
         flows(n,m,3) =  flows(n,m,3) +  flows(nn,m,3)
         flows(n,m,4) =  flows(n,m,4) +  flows(nn,m,4)
        fluxes(n,m,1) = fluxes(n,m,1) + fluxes(nn,m,1)
        fluxes(n,m,2) = fluxes(n,m,2) + fluxes(nn,m,2)
        fluxes(n,m,3) = fluxes(n,m,3) + fluxes(nn,m,3)
        fluxes(n,m,4) = fluxes(n,m,4) + fluxes(nn,m,4)
    end do
  end if
end do
do m=1,nTI
  saturations(n,m) = saturations(n,m)/volumes(n)
  velocitiesV(n,m) = velocitiesV(n,m)/volumes(n)
```

```
      velocitiesH(n,m) = velocitiesH(n,m)/volumes(n)
       poreCounts(n,m) =  poreCounts(n,m)/volumes(n)
    end do
  end do
end if

!COUNT PORE VOLUMES
do n=1,nZone+nGroup
  poreCumulative(n,1) = poreCounts(n,1)
  do m=2,nTI
    poreCumulative(n,m) = poreCumulative(n,m-1) + poreCounts(n,m)
  end do
end do

!COMPUTE Eh TIME
EhTime = -999._LONG
do n=1,nZone+nGroup
  do m=2,nTI
    if (poreCumulative(n,m) .ge. Eh(n)) then
      EhTime(n) = (Eh(n) - poreCumulative(n,m-1))/poreCounts(n,m) &
                  *(times(2,m) - times(1,m)) + times(1,m)
      exit
    end if
  end do
end do

!COMPUTE pH TIME
pHtime = -999._LONG
do n=1,nZone+nGroup
  do m=2,nTI
    if (poreCumulative(n,m) .ge. pH(n)) then
      pHtime(n) = (pH(n) - poreCumulative(n,m-1))/poreCounts(n,m) &
                  *(times(2,m) - times(1,m)) + times(1,m)
      exit
    end if
  end do
end do

!WRITE Eh AND pH TIMEs
write(iout,'(a)') "= = = = = Eh and pH transition pore volumes and times = = = = ="
do n=1,nZone+nGroup
  if (Eh(n) .le. 100000) then
    write(string,'(f12.0)') Eh(n)
  else
    write(string,'(1pe7.0)') Eh(n)
  end if

  if (pH(n) .le. 100000) then
    write(string2,'(f12.0)') pH(n)
  else
    write(string2,'(1pe7.0)') pH(n)
  end if

  write(iout,'(3(a,a),2(f12.0,a))') &
    trim(zoneName(n)), tab, &
    trim(adjustl(string)), tab, &
    trim(adjustl(string2)), tab, &
    EhTime(n), tab, &
    pHtime(n)
end do

sGoldSim = trim(tag)//tab//"Zone"//tab//"EhPoreVol"//tab// &
           "pHporeVol"//tab//"EhTime"//tab//"pHtime"
```

```
write(itab3,'(a)') trim(sGoldSim)
do n=1,nZone+nGroup
  sGoldSim = trim(tag)//tab//trim(zoneName(n))
  if (Eh(n) .le. 100000) then
    write(string,'(f12.0)') Eh(n)
  else
    write(string,'(1pe7.0)') Eh(n)
  end if
  sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  if (pH(n) .le. 100000) then
    write(string,'(f12.0)') pH(n)
  else
    write(string,'(1pe7.0)') pH(n)
  end if
  sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  write(string,'(f12.0)')   EhTime(n)
  sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  write(string,'(f12.0)')   pHtime(n)
  sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  write(itab3,'(a)') trim(sGoldSim)
end do

!LOOP OVER TIs FOR TECPLOT (values per full circle)
write(itec,'(a)') "VARIABLES = time darcyV darcyH saturation flow volume volumePore
volumeWater"
do n=1,nZone+nGroup
  sZone = trim(zoneName(n))
  write(itec,'(a)') "ZONE T="//trim(tag)//"_"//trim(sZone)
  do m=1,nTI
    write(itec,'(8(es12.3e3))') &
      times(1,m), -velocitiesV(n,m), velocitiesH(n,m), saturations(n,m), &
      flowsIn2(n,m)*fullGeometry, volumes(n)*fullGeometry, &
      volumesPore(n,m)*fullGeometry, volumesWater(n,m)*fullGeometry
    write(itec,'(8(es12.3e3))') &
      times(2,m), -velocitiesV(n,m), velocitiesH(n,m), saturations(n,m), &
      flowsIn2(n,m)*fullGeometry, volumes(n)*fullGeometry, &
      volumesPore(n,m)*fullGeometry, volumesWater(n,m)*fullGeometry
  end do
end do

!LOOP OVER TIs FOR GOLDSIM (values per full circle)
sGoldSim = trim(tag)//tab//"tBeg"//tab//"tEnd"
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"darcyV"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
end do
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"darcyH"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
end do
string = "infiltration"
sGoldSim = trim(sGoldSim)//tab//trim(string)
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"flow"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
end do
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"sat"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
end do
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"pore"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
```

```
end do
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"Eh"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
end do
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"pH"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
end do
if (writeCrossflow) then
do n=1,nZone+nGroup
  string = trim(zoneName(n))//"_"//"crossflow"
  sGoldSim = trim(sGoldSim)//tab//trim(string)
end do
end if
write(itab2,'(a)') trim(sGoldSim)

do m=1,nTI
  sGoldSim = trim(tag)
  write(string,*) nint(times(1,m))
  sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  write(string,*) nint(times(2,m))
  sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  do n=1,nZone+nGroup
    write(string,'(es14.5e3)')  velocitiesV(n,m)
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  do n=1,nZone+nGroup
    write(string,'(es14.5e3)')  velocitiesH(n,m)
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  write(string,'(es14.5e3)') infiltrations(m)
  sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  do n=1,nZone+nGroup
    write(string,'(es14.5e3)') flowsIn2(n,m)*fullGeometry
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  do n=1,nZone+nGroup
    write(string,'(es14.5e3)') saturations(n,m)
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  do n=1,nZone+nGroup
    write(string,'(es14.5e3)') volumesPore(n,m)*fullGeometry
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  do n=1,nZone+nGroup
    write(string,*) nint(EhTime(n))
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  do n=1,nZone+nGroup
    write(string,*) nint(pHtime(n))
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  if (writeCrossflow) then
  do n=1,nZone+nGroup
    write(string,'(es14.5e3)') xflow(n)
    sGoldSim = trim(sGoldSim)//tab//trim(adjustl(string))
  end do
  end if
  write(itab2,'(a)') trim(sGoldSim)
end do

end program GoldSimFlows
```

```
module stuff

implicit none

contains


subroutine stuffMatrixReal(iBeg,iEnd,jBeg,jEnd,arrayReal,matrixReal)
integer, parameter :: LONG=SELECTED_REAL_KIND(9,99)
integer, intent(in) :: iBeg, iEnd, jBeg, jEnd
integer :: i, j, n
real(LONG), dimension(:), intent(in) :: arrayReal
real(LONG), dimension(iBeg:,jBeg:), intent(out) :: matrixReal
n = 0
do j=jBeg,jEnd
  do i=iBeg,iEnd
    n = n + 1
    matrixReal(i,j) = arrayReal(n)
  end do
end do
end subroutine stuffMatrixReal


subroutine stuffMatrixInteger(iBeg,iEnd,jBeg,jEnd,arrayInteger,matrixInteger)
integer, parameter :: LONG=SELECTED_REAL_KIND(9,99)
integer, intent(in) :: iBeg, iEnd, jBeg, jEnd
integer :: i, j, n
integer, dimension(:), intent(in) :: arrayInteger
integer, dimension(iBeg:,jBeg:), intent(out) :: matrixInteger
n = 0
do j=jBeg,jEnd
  do i=iBeg,iEnd
    n = n + 1
    matrixInteger(i,j) = arrayInteger(n)
  end do
end do
end subroutine stuffMatrixInteger


subroutine getReal(name,iunt,arrayReal,nValues,time)

integer, parameter :: LONG=SELECTED_REAL_KIND(9,99)
integer, intent(in) :: iunt
integer :: iostat_flag, i, nTmp
integer, intent(out) :: nValues

real(LONG) :: Tmp
real(LONG), intent(out) :: time
real(LONG), dimension(:), intent(out) :: arrayReal

character(len=4), intent(in) :: name
character(len=4) :: string
character(len=200) :: Line

logical ::  exit_flag

time = -999
nValues = -999
do
  read(iunt,'(a)',iostat=iostat_flag) Line
  call IostatCheck (iostat_flag,"iunt", exit_flag)
```

```
  if (exit_flag) exit  !end-of-file

  if (index(Line,"%RECORD%") .ne. 0) then !found start of record
    read(Line(49:),*) Tmp
    read(iunt,'(a)') Line
    read(Line,*) string
    read(Line(50:),*) nTmp
    if (string .eq. name) then
      time = Tmp
      nValues = nTmp
      read(iunt,*) (arrayReal(i),i=1,nValues)
      exit
    end if
  end if
end do

end subroutine getReal


subroutine getInteger(name,iunt,arrayInteger,nValues,time)

integer, parameter :: LONG=SELECTED_REAL_KIND(9,99)
integer, intent(in) :: iunt
integer :: iostat_flag, i, nTmp
integer, intent(out) :: nValues
integer, dimension(:), intent(out) :: arrayInteger

real(LONG) :: Tmp
real(LONG), intent(out) :: time

character(len=4), intent(in) :: name
character(len=4) :: string
character(len=200) :: Line

logical ::  exit_flag

time = -999
nValues = -999
do
  read(iunt,'(a)',iostat=iostat_flag) Line
  call IostatCheck (iostat_flag,"iunt", exit_flag)
  if (exit_flag) exit  !end-of-file

  if (index(Line,"%RECORD%") .ne. 0) then !found start of record
    read(Line(49:),*) Tmp
    read(iunt,'(a)') Line
    read(Line,*) string
    read(Line(50:),*) nTmp
    if (string .eq. name) then
      time = Tmp
      nValues = nTmp
      read(iunt,*) (arrayInteger(i),i=1,nValues)
      exit
    end if
  end if
end do

end subroutine getInteger


subroutine IostatCheck (iostat_flag, string, exit_flag)

integer  , intent(in) :: iostat_flag
```

```
character(len=*), intent(in) :: string

logical, intent(out) ::  exit_flag

if      (iostat_flag .eq. 0) then !OK
  exit_flag = .false.

else if (iostat_flag .lt. 0) then !END-OF-FILE
  exit_flag = .true.

else if (iostat_flag .gt. 0) then !ERROR
  write (*,'(2a)') '*** READ ERROR *** ', string
  stop

end if

end subroutine IostatCheck


end module stuff
```

APPENDIX B: PORFLOW INPUT FILES FOR "CASEB_XRT"

COOR.dat

```
0.00000       0.00000
100.000       0.00000
200.000       0.00000
300.000       0.00000
400.000       0.00000
500.000       0.00000
600.000       0.00000
700.000       0.00000
800.000       0.00000
900.000       0.00000
1000.00       0.00000
1100.00       0.00000
0.00000       100.000
100.000       100.000
200.000       100.000
300.000       100.000
400.000       100.000
500.000       100.000
600.000       100.000
700.000       100.000
800.000       100.000
900.000       100.000
1000.00       100.000
1100.00       100.000
0.00000       200.000
100.000       200.000
200.000       200.000
300.000       200.000
400.000       200.000
500.000       200.000
600.000       200.000
700.000       200.000
800.000       200.000
900.000       200.000
1000.00       200.000
1100.00       200.000
0.00000       300.000
100.000       300.000
200.000       300.000
300.000       300.000
400.000       300.000
500.000       300.000
600.000       300.000
700.000       300.000
800.000       300.000
900.000       300.000
1000.00       300.000
1100.00       300.000
0.00000       400.000
100.000       400.000
200.000       400.000
300.000       400.000
400.000       400.000
500.000       400.000
600.000       400.000
700.000       400.000
800.000       400.000
```

900.000	400.000
1000.00	400.000
1100.00	400.000
0.00000	500.000
100.000	500.000
200.000	500.000
300.000	500.000
400.000	500.000
500.000	500.000
600.000	500.000
700.000	500.000
800.000	500.000
900.000	500.000
1000.00	500.000
1100.00	500.000
0.00000	600.000
100.000	600.000
200.000	600.000
300.000	600.000
400.000	600.000
500.000	600.000
600.000	600.000
700.000	600.000
800.000	600.000
900.000	600.000
1000.00	600.000
1100.00	600.000
0.00000	700.000
100.000	700.000
200.000	700.000
300.000	700.000
400.000	700.000
500.000	700.000
600.000	700.000
700.000	700.000
800.000	700.000
900.000	700.000
1000.00	700.000
1100.00	700.000
0.00000	800.000
100.000	800.000
200.000	800.000
300.000	800.000
400.000	800.000
500.000	800.000
600.000	800.000
700.000	800.000
800.000	800.000
900.000	800.000
1000.00	800.000
1100.00	800.000
0.00000	900.000
100.000	900.000
200.000	900.000
300.000	900.000
400.000	900.000
500.000	900.000
600.000	900.000
700.000	900.000
800.000	900.000
900.000	900.000
1000.00	900.000
1100.00	900.000

TYPE.dat

```
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    2    2    2    2    4    2    2    2    2
    1    1    1    1    3    3    3    3    4    3    3    3    2    1    1    1
    1    3    3    3    3    4    3    3    3    2    1    1    1    1    3    3
    3    3    4    3    3    3    2    1    1    1    1    2    2    2    2    4
    2    2    2    2    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
```

.../TI01/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cylindrical, CaseB_cyl, TI01, 0-100 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 11 by 13 NODEs
COORdinates CYLIndrical X R from file "../../Mesh2d_cyl/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (11,13)
LOCAte ID=INSIDE as nodes (1,1) to (11,13), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d_cyl/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (11,1), (1,13), (11,13) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  -0.00981  0.0  0.0  0.00981             !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                  !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                              !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                     !reference pressure (N/cm^2)
PROPerty for P is HARMonic                       !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
```

```
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 1:      !ID=SOIL     !Sand
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 8.83E+03 ky = 1.58E+04 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !HQ concrete VG
MATErial   ID=CONCRETE DENSity  2.51
MATErial   ID=CONCRETE POROsity =  0.168
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !HQ concrete VG
MATErial   ID=GROUT DENSity  2.40
MATErial   ID=GROUT POROsity =  0.58
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr 0,
Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !same as CONCRETE
MATErial   ID=FASTFLOW DENSity  2.51
MATErial   ID=FASTFLOW POROsity =  0.168
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for Y-: 0    !no flow
BOUNdary FLUX  of P for Y+: 0    !no flow
BOUNdary VALUe of P for X-: 0    !pressure bottom of model
BOUNdary FLUX  of P for X+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
```

```
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
!
/ Initial Condition
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                      NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
```

```
!
FLUX for C in ID=SOIL     to "FLUXC.out"
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

.../TI02/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cylindrical, CaseB_cyl, TI02, 100-1000 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 11 by 13 NODEs
COORdinates CYLIndrical X R from file "../../Mesh2d_cyl/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (11,13)
LOCAte ID=INSIDE as nodes (1,1) to (11,13), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d_cyl/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (11,1), (1,13), (11,13) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  -0.00981  0.0  0.0  0.00981            !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                 !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                             !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                    !reference pressure (N/cm^2)
PROPerty for P is HARMonic                      !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
```

```
!
!FOR material type 1:      !ID=SOIL     !Sand
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 8.83E+03 ky = 1.58E+04 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !HQ concrete VG
MATErial   ID=CONCRETE DENSity  2.51
MATErial   ID=CONCRETE POROsity =  0.168
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !HQ concrete VG
MATErial   ID=GROUT DENSity  2.40
MATErial   ID=GROUT POROsity =  0.58
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr 0,
Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !same as GROUT
MATErial   ID=FASTFLOW DENSity  2.40
MATErial   ID=FASTFLOW POROsity =  0.58
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for Y-: 0    !no flow
BOUNdary FLUX  of P for Y+: 0    !no flow
BOUNdary VALUe of P for X-: 0    !pressure bottom of model
BOUNdary FLUX  of P for X+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
```

```
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
!
/ Initial Condition
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                       NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
!
FLUX for C in ID=SOIL     to "FLUXC.out"
```

```
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

.../TI03/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cylindrical, CaseB_cyl, TI03, 1000-10000 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 11 by 13 NODEs
COORdinates CYLIndrical X R from file "../../Mesh2d_cyl/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (11,13)
LOCAte ID=INSIDE as nodes (1,1) to (11,13), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d_cyl/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (11,1), (1,13), (11,13) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  -0.00981  0.0  0.0  0.00981             !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                 !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                             !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                    !reference pressure (N/cm^2)
PROPerty for P is HARMonic                      !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 1:      !ID=SOIL     !Sand
```

```
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 8.83E+03 ky = 1.58E+04 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !HQ concrete VG
MATErial   ID=CONCRETE DENSity  2.51
MATErial   ID=CONCRETE POROsity =  0.168
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !HQ concrete VG
MATErial   ID=GROUT DENSity  2.40
MATErial   ID=GROUT POROsity =  0.58
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr 0,
Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !same as SOIL
MATErial   ID=FASTFLOW DENSity  2.66
MATErial   ID=FASTFLOW POROsity =  0.38
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for Y-: 0    !no flow
BOUNdary FLUX  of P for Y+: 0    !no flow
BOUNdary VALUe of P for X-: 0    !pressure bottom of model
BOUNdary FLUX  of P for X+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
```

```
!
/ Initial Condition
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                      NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
!
FLUX for C in ID=SOIL     to "FLUXC.out"
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
```

```
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

.../TI04/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cylindrical, CaseB_cyl, TI04, 10000-100000 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 11 by 13 NODEs
COORdinates CYLIndrical X R from file "../../Mesh2d_cyl/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (11,13)
LOCAte ID=INSIDE as nodes (1,1) to (11,13), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d_cyl/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (11,1), (1,13), (11,13) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  -0.00981  0.0  0.0  0.00981             !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                 !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                             !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                    !reference pressure (N/cm^2)
PROPerty for P is HARMonic                      !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 1:      !ID=SOIL      !Sand
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
```

```
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 8.83E+03 ky = 1.58E+04 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !same as SOIL
MATErial   ID=CONCRETE DENSity  2.66
MATErial   ID=CONCRETE POROsity =  0.38
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 8.83E+03 ky = 1.58E+04 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !same as SOIL
MATErial   ID=GROUT DENSity  2.66
MATErial   ID=GROUT POROsity =  0.38
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 8.83E+03 ky = 1.58E+04 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !Gravel
MATErial   ID=FASTFLOW DENSity  2.60
MATErial   ID=FASTFLOW POROsity =  0.30
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 4.73E+06 ky = 4.73E+06 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.45746, alpha 1.43e-01, Sr
0.0524, Sg 0, m 0.314, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for Y-: 0    !no flow
BOUNdary FLUX  of P for Y+: 0    !no flow
BOUNdary VALUe of P for X-: 0    !pressure bottom of model
BOUNdary FLUX  of P for X+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
!
/ Initial Condition
```

```
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                       NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
!
FLUX for C in ID=SOIL     to "FLUXC.out"
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
```

```
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

APPENDIX C: PORFLOW INPUT FILES FOR "CASEB_XYZ"

COOR.dat

```
0.00000       0.00000
100.000       0.00000
200.000       0.00000
300.000       0.00000
400.000       0.00000
500.000       0.00000
600.000       0.00000
700.000       0.00000
800.000       0.00000
900.000       0.00000
1000.00       0.00000
1100.00       0.00000
0.00000       100.000
100.000       100.000
200.000       100.000
300.000       100.000
400.000       100.000
500.000       100.000
600.000       100.000
700.000       100.000
800.000       100.000
900.000       100.000
1000.00       100.000
1100.00       100.000
0.00000       200.000
100.000       200.000
200.000       200.000
300.000       200.000
400.000       200.000
500.000       200.000
600.000       200.000
700.000       200.000
800.000       200.000
900.000       200.000
1000.00       200.000
1100.00       200.000
0.00000       300.000
100.000       300.000
200.000       300.000
300.000       300.000
400.000       300.000
500.000       300.000
600.000       300.000
700.000       300.000
800.000       300.000
900.000       300.000
1000.00       300.000
1100.00       300.000
0.00000       400.000
100.000       400.000
200.000       400.000
300.000       400.000
400.000       400.000
500.000       400.000
600.000       400.000
700.000       400.000
800.000       400.000
```

900.000	400.000
1000.00	400.000
1100.00	400.000
0.00000	500.000
100.000	500.000
200.000	500.000
300.000	500.000
400.000	500.000
500.000	500.000
600.000	500.000
700.000	500.000
800.000	500.000
900.000	500.000
1000.00	500.000
1100.00	500.000
0.00000	600.000
100.000	600.000
200.000	600.000
300.000	600.000
400.000	600.000
500.000	600.000
600.000	600.000
700.000	600.000
800.000	600.000
900.000	600.000
1000.00	600.000
1100.00	600.000
0.00000	700.000
100.000	700.000
200.000	700.000
300.000	700.000
400.000	700.000
500.000	700.000
600.000	700.000
700.000	700.000
800.000	700.000
900.000	700.000
1000.00	700.000
1100.00	700.000
0.00000	800.000
100.000	800.000
200.000	800.000
300.000	800.000
400.000	800.000
500.000	800.000
600.000	800.000
700.000	800.000
800.000	800.000
900.000	800.000
1000.00	800.000
1100.00	800.000
0.00000	900.000
100.000	900.000
200.000	900.000
300.000	900.000
400.000	900.000
500.000	900.000
600.000	900.000
700.000	900.000
800.000	900.000
900.000	900.000
1000.00	900.000
1100.00	900.000

TYPE.dat

```
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    2    2    2    2    4    2    2    2    2
    1    1    1    1    3    3    3    3    4    3    3    3    2    1    1    1
    1    3    3    3    3    4    3    3    3    2    1    1    1    1    3    3
    3    3    4    3    3    3    2    1    1    1    1    2    2    2    2    4
    2    2    2    2    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
```

.../TI01/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cartestian, CaseB, TI01, 0-100 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 13 by 11 NODEs
COORdinates X Y from file "../../Mesh2d/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (13,11)
LOCAte ID=INSIDE as nodes (1,1) to (13,11), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (13,1), (1,11), (13,11) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  0.0  -0.00981  0.0  0.00981             !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                  !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                              !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                     !reference pressure (N/cm^2)
PROPerty for P is HARMonic                       !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
```

```
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 1:      !ID=SOIL     !Sand
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !HQ concrete VG
MATErial   ID=CONCRETE DENSity  2.51
MATErial   ID=CONCRETE POROsity =  0.168
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !HQ concrete VG
MATErial   ID=GROUT DENSity  2.40
MATErial   ID=GROUT POROsity =  0.58
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr 0,
Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !same as CONCRETE
MATErial   ID=FASTFLOW DENSity  2.51
MATErial   ID=FASTFLOW POROsity =  0.168
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for X-: 0    !no flow
BOUNdary FLUX  of P for X+: 0    !no flow
BOUNdary VALUe of P for Y-: 0    !pressure bottom of model
BOUNdary FLUX  of P for Y+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
```

```
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
!
/ Initial Condition
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                      NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
```

```
!
FLUX for C in ID=SOIL     to "FLUXC.out"
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

.../TI02/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cartestian, CaseB, TI02, 100-1000 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 13 by 11 NODEs
COORdinates X Y from file "../../Mesh2d/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (13,11)
LOCAte ID=INSIDE as nodes (1,1) to (13,11), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (13,1), (1,11), (13,11) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  0.0  -0.00981  0.0  0.00981            !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                  !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                              !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                     !reference pressure (N/cm^2)
PROPerty for P is HARMonic                       !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
```

```
!
!FOR material type 1:      !ID=SOIL     !Sand
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !HQ concrete VG
MATErial   ID=CONCRETE DENSity  2.51
MATErial   ID=CONCRETE POROsity =  0.168
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !HQ concrete VG
MATErial   ID=GROUT DENSity  2.40
MATErial   ID=GROUT POROsity =  0.58
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr 0,
Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !same as GROUT
MATErial   ID=FASTFLOW DENSity  2.40
MATErial   ID=FASTFLOW POROsity =  0.58
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for X-: 0    !no flow
BOUNdary FLUX  of P for X+: 0    !no flow
BOUNdary VALUe of P for Y-: 0    !pressure bottom of model
BOUNdary FLUX  of P for Y+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
```

```
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
!
/ Initial Condition
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                      NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
!
FLUX for C in ID=SOIL     to "FLUXC.out"
```

```
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

.../TI03/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cartestian, CaseB, TI03, 1000-10000 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 13 by 11 NODEs
COORdinates X Y from file "../../Mesh2d/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (13,11)
LOCAte ID=INSIDE as nodes (1,1) to (13,11), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (13,1), (1,11), (13,11) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  0.0  -0.00981  0.0  0.00981             !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                 !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                             !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                    !reference pressure (N/cm^2)
PROPerty for P is HARMonic                      !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 1:      !ID=SOIL     !Sand
```

```
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !HQ concrete VG
MATErial   ID=CONCRETE DENSity  2.51
MATErial   ID=CONCRETE POROsity =  0.168
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 2.93E-03 ky = 2.93E-03 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr
0, Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !HQ concrete VG
MATErial   ID=GROUT DENSity  2.40
MATErial   ID=GROUT POROsity =  0.58
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 2.02E-01 ky = 2.02E-01 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.9433, alpha 2.0856e-6, Sr 0,
Sg 0, m 0.485, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !same as SOIL
MATErial   ID=FASTFLOW DENSity  2.66
MATErial   ID=FASTFLOW POROsity =  0.38
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for X-: 0    !no flow
BOUNdary FLUX  of P for X+: 0    !no flow
BOUNdary VALUe of P for Y-: 0    !pressure bottom of model
BOUNdary FLUX  of P for Y+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
```

```
!
/ Initial Condition
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                      NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
!
FLUX for C in ID=SOIL     to "FLUXC.out"
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
```

```
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

.../TI04/RUN.dat

```
/ Main input file
TITLe TEST CASE FOR GOLDSIMFLOWS, Cartestian, CaseB, TI04, 10000-100000 YRS
!
/ Units: cm, g, yr
!
/ CPUs
CPU 4  !recent versions of Porflow
!
/ Allocate space for user-defined variables
ALLOcate HEAD
!
/ Finite-element mesh
GRID is 13 by 11 NODEs
COORdinates X Y from file "../../Mesh2d/COOR.dat"
LOCAte ID=DOMAIN as nodes (1,1) to (13,11)
LOCAte ID=INSIDE as nodes (1,1) to (13,11), FIELd only
!
/ Material types
MATErial type data from "../../Mesh2d/TYPE.dat"
!
/ Subregions
LOCAte MATErial type 1 as subregion ID=SOIL
LOCAte MATErial type 2 as subregion ID=CONCRETE
LOCAte MATErial type 3 as subregion ID=GROUT
LOCAte MATErial type 4 as subregion ID=FASTFLOW
!
LOCAte UNIOn of ID=CONCRETE and ID=GROUT    as ID=CEMENT
LOCAte UNIOn of ID=CEMENT   and ID=FASTFLOW as ID=DISPCELL
!
LOCAte ID=OUTNODES by IJ indices: (1,1), (13,1), (1,11), (13,11) EXTErior
!
WRITE MTYP for ID=DOMAIN to "MTYP.dat"
!
/ Materials
/ Global settings
GRAVity components  0.0  -0.00981  0.0  0.00981             !xyz components, |g/gc|
(N/g)
DENSity of fluid 0.9982                                 !fluid density (g/cm^3)
VISCosity of fluid 3.19e-15                             !fluid viscosity (N-yr/cm^2)
REFErence P is 10.13                                    !reference pressure (N/cm^2)
PROPerty for P is HARMonic                      !spatial averaging at faces
!
/ Properties
ALLOcate TABLe space for 18000 words
!
/ Exterior Node Specification
MULTiphase ID=OUTNODES properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 1:      !ID=SOIL      !Sand
MATErial   ID=SOIL DENSity  2.66
MATErial   ID=SOIL POROsity =  0.38
```

```
HYDRaulic  ID=SOIL ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=SOIL properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 2:      !ID=CONCRETE !same as SOIL
MATErial   ID=CONCRETE DENSity  2.66
MATErial   ID=CONCRETE POROsity =  0.38
HYDRaulic  ID=CONCRETE ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=CONCRETE properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 3:      !ID=GROUT    !same as SOIL
MATErial   ID=GROUT DENSity  2.66
MATErial   ID=GROUT POROsity =  0.38
HYDRaulic  ID=GROUT ss= 1.e-3  kx = 1.58E+04 ky = 8.83E+03 kz = 0
MULTiphase ID=GROUT properties using VAN Genuchten n 1.40995, alpha 0.02954, Sr
0.354, Sg 0, m 0.291, krmin 1.e-20
!
!FOR material type 4:      !ID=FASTFLOW !Gravel
MATErial   ID=FASTFLOW DENSity  2.60
MATErial   ID=FASTFLOW POROsity =  0.30
HYDRaulic  ID=FASTFLOW ss= 1.e-3  kx = 4.73E+06 ky = 4.73E+06 kz = 0
MULTiphase ID=FASTFLOW properties using VAN Genuchten n 1.45746, alpha 1.43e-01, Sr
0.0524, Sg 0, m 0.314, krmin 1.e-20
!
/ Boundary conditions
BOUNdary FLUX  of P for X-: 0    !no flow
BOUNdary FLUX  of P for X+: 0    !no flow
BOUNdary VALUe of P for Y-: 0    !pressure bottom of model
BOUNdary FLUX  of P for Y+: 10   !infiltration (cm/yr)
!
/ Outputs
!SAVE P S U V MOIS FC to "MAIN.sav" every 1 step
!
STATistics for U in ID=DOMAIN   to "STAT.out"
STATistics for U in ID=SOIL     to "STAT.out"
STATistics for U in ID=CONCRETE to "STAT.out"
STATistics for U in ID=GROUT    to "STAT.out"
STATistics for U in ID=FASTFLOW to "STAT.out"
STATistics for U in ID=DISPCELL to "STAT.out"
!
STATistics for V in ID=DOMAIN   to "STAT.out"
STATistics for V in ID=SOIL     to "STAT.out"
STATistics for V in ID=CONCRETE to "STAT.out"
STATistics for V in ID=GROUT    to "STAT.out"
STATistics for V in ID=FASTFLOW to "STAT.out"
STATistics for V in ID=DISPCELL to "STAT.out"
!
STATistics for S in ID=DOMAIN   to "STAT.out"
STATistics for S in ID=SOIL     to "STAT.out"
STATistics for S in ID=CONCRETE to "STAT.out"
STATistics for S in ID=GROUT    to "STAT.out"
STATistics for S in ID=FASTFLOW to "STAT.out"
STATistics for S in ID=DISPCELL to "STAT.out"
!
FLUX for P in ID=DOMAIN   to "FLUX.out"
FLUX for P in ID=SOIL     to "FLUX.out"
FLUX for P in ID=CONCRETE to "FLUX.out"
FLUX for P in ID=GROUT    to "FLUX.out"
FLUX for P in ID=FASTFLOW to "FLUX.out"
FLUX for P in ID=DISPCELL to "FLUX.out"
!
/ Initial Condition
```

```
TIME 0
SET P to: 0                        !set pressure head (cm) to constant
!SET P to LINEar function: 0 -1(Y)      !set head (cm) to constant
!READ UNFOrmatted "BINARY.sav" and make a fresh STARt
!
/ Solve
PROPerty P UPWI
MATRix LUDE for P
CONVergence for P    1.e-6, 10 iterations max
DIAGnostic output: TIME P S U V for node (2,2) every 1 steps
!DIAGnostic output: TIME V BP DP RP for node (2,2) every 1 steps
!
!!Migrate to solution neighborhood
CONVergence for P    1.e-6, 10 iterations max
SOLVe STEAdy 5
RELAx S 0.7
SOLVe STEAdy 5
RELAx S 0.3
SOLVe STEAdy 15
RELAx S 0.1
SOLVe STEAdy 45
!
!!Mixed purpose
CONVergence for P    1.e-6, 30 iterations max
RELAx S 0.03
SOLVe STEAdy 50
RELAx S 0.01
SOLVe STEAdy 50
!
!!Suppress noise / sharpen mass balance
CONVergence for P    1.e-6, 100 iterations max
RELAx S 0.003
SOLVe STEAdy 20
RELAx S 0.001
SOLVe STEAdy 20
RELAx S 0.0003
SOLVe STEAdy 20
RELAx S 0.0001
SOLVe STEAdy 20
RELAx S 0.00001
SOLVe STEAdy 20
RELAx S 0.000001
SOLVe STEAdy 20
!
/ Compute auxiliary variables
SET HEAD by LINEar function:  +1(P) +1(Y) +0  !h=p+z
!
/ Outputs
SAVE P S U V MOIS FC to "MAIN.sav"                      NOW
CLOSe "MAIN.sav"
SAVE FC S            to "FLOW.sav" using COMPact format NOW
CLOSe "FLOW.sav"
SAVE P S U V MOIS FC to "BINARY.sav" UNFOrmatted        NOW
!
/ Transport run to determine void volumes
SET S=1
SET C=1
!
FLUX for C in ID=SOIL     to "FLUXC.out"
FLUX for C in ID=CONCRETE to "FLUXC.out"
FLUX for C in ID=GROUT    to "FLUXC.out"
FLUX for C in ID=FASTFLOW to "FLUXC.out"
FLUX for C in ID=DISPCELL to "FLUXC.out"
```

```
!
SOLVE C 1.e-7 1.e-7
!
END
QUIT
```

APPENDIX D: Independent Review Summary

Independent Reviewer comments:

Technical Review of the GoldSim to PORFLOW flow abstractor system.

The two-dimensional flows from the PORFLOW model must be abstracted to one-dimensional flows to be used by the GoldSim model. This is accomplished by a "system" of two programs, one a makefile and the other a FORTRAN program. This technical review checked the coding of both in addition to assessing its efficacy via a test problem.

Coding check

A line-by-line check of the coding was performed. Both the makefile and the FORTRAN program appear to be coded correctly. The only issue with the makefile is that it is essentially coded so that its time periods and vaults are hardwired. While not an issue with the current analysis, it does detract from the generality of the program. This shows up near the bottom of the program:

```
if [ "${Vault}" = "Vault1" ]; then tis=59; fi;\
```

where "tis' is the number of time periods for a vault. If one were to modify the number of time periods, which are stored in an array, one must also change these hardwired values. It might be better to have "tis" set to the array length rather than the hardcoded value. It also might be a good idea to turn the "tmp" file generated by the makefile into a permanent file. As the makefile is an important and integral piece of the code system, a copy should be included in Appendix A.

Output Check

An exhaustive output check of the test case is documented in Reference 1 and herein. It shows that the program performs as expected.

References

Flach, G., "GoldSimFlows program", SRNL-L3200-2013-00024, Rev. 1, June 4, 2013.

CTF Response:

The scope of this software QA plan is intended to be just the FORTRAN program named "GoldSimFlows", for the reason that (as observed by the Independent Reviewer) the Makefile is problem-specific and currently reflects FY2013 Saltstone Special Analysis parameters. The plan going forward is to modify the Makefile, or manually create the input to GoldSimFlows, on a project-by-project basis. The specific actions would be design-checked at that time. Considering the scope of the SQAP only the FORTRAN program is listed in Appendix A.

THIS PAGE INTENTIONALLY LEFT BLANK

Distribution

R. S. Aylward, 773-42A
B. T. Butcher, 773-43A
D. A. Crowley, 773-43A
W. A. Drown, 773-41A
G. P. Flach, 773-42A
B. H. Lester, 705-1C
S. L. Marra, 773-A
K. H. Rosenberger, 705-1C
R. E. Sheppard, 705-1C
G. A. Taylor, 773-43AC. Wilson (1 file copy and 1 electronic copy), 773-43A – Rm. 213