SOFTWARE SUMMARY FORM

01. Summary Date:	02. Summary prepared by (Name and phone)	03. Summary Action:
4/27/93	Mark V. Muller, 522-3222	
04. Software Date: December 1992	05. Software Title: BIGFLOW 1.0	New
06. Short Title: BIGFLOW 1.0		07. Internal Software ID:

08. Software Type:	09. Processing Mode:	10. APPLICATION AREA
☐ Automated Data System	☐ Interactive	a. General: ☒ Scientific/Engineering ☐ Bibliographic/Textual ☐ Process Control ☐ Management Business
☒ Computer Program	☐ Batch	☐ Computer Systems Support/Utility ☐ Other
☐ Subroutine/Module	☒ Combination	b. Specific: Hydrogeologic Flow Simulation Model

11. Submitting Organization and Address:	12. Technical Contact(s) and Phone:
CNWRA	Ross Bagtzoglou, 522-5182

13. Narrative:
BIGFLOW (provisional name) is an evolving Fortran 77 computer code for simulating three-dimensional Darcy-type flow in a heterogeneous porous medium. BIGFLOW is currently being modified from the 3D finite difference BIGFLO code previously developed at the MIT Parson's Laboratory (Author: R. Ababou; Contact at MIT: Prof. L.W. Gelhar). The BIGFLO code accommodates steady as well as transient flow regimes in saturated, variably saturated, or purely unsaturated porous media with highly heterogeneous hydrodynamic properties. The evolving BIGFLOW code will contain the features of the previous BIGFLO code as a subset. A companion code (DATAFLOW) will be used for data processing.

14. Keywords:
Finite Differences; Three-Dimensional; Porous Media; Heterogeneous; Saturated; Unsaturated; Hydrogeology

15. Computer Model and Manufacturer	16. Computer Operating System:	17. Programming Language(s):	18. Number of Source Program Statements:
Cray/XMP	UNICOS	FORTRAN	20,000 (estimated)
19. Computer Memory Requirements: Variable (dynamic allocation)	20. Tape Drives: N/A	21. Disk/Drum Units: N/A	22. Terminals: N/A

23. Other Operational Requirements

None.

24. Software Availability:	25. Documentation Availability:
☐ Available ☐ Limited ☒ In-House ONLY	☐ Available ☐ Inadequate ☒ In-House ONLY
☐ Active ☐ Inactive	

26. Submission Package Status:
CCB Acceptance Criteria: Met ☒ Not Met ☐ Software QA Assessment: Successful ☒ Unsuccessful ☐

Code Custodian: _____ Date: 10/19/93

CNWRA Form TOP-4-1 (11/92)

SOFTWARE RELEASE NOTICE

01. SRN Number: PA-SRN-004

02. Project Title: Hydrogeologic Flow Simulation, CNWRA Version 1.1

Project No.

03. SRN Title: Bigflow 1.0

04. Originator/Requester: Thomas J. Ratchford

Date: 12/14/93

05. Summary of Actions

- ■ Release of new code admitted to CM System
- □ Release of modified code:
  - □ Enhancements made
  - □ Corrections made
- ■ Change of access code

06. Persons Authorized Access

Name	RO/RW	A/C/D

07. Element Manager Approval: Date:

08. Remarks:

CNWRA Form TOP-6 (08/93)

```
gemstone.14 ~/bigflo/ver11/VCS => ls
ALFA.Z*           BETA.Z*           COMBIG.orig.Z*   HIC.Z*
IN1.Z*            INPUT1.Z*         KSAT.Z*          Makefile.Z*
bcl10.F.Z*        bcl20.F.Z*        bcl30.F.Z*       bcnl10.F.Z*
bcnl20.F.Z*       bcnl30.F.Z*       bflux1.F.Z*      bflux2.F.Z*
bflux3.F.Z*       bhead1.F.Z*       bhead2.F.Z*      bhead3.F.Z*
bhzero.F.Z*       bigflo.cpp.Z*     bigflo.f.ori.Z*  bigflo.pre.Z*
cg.F.Z*           combig.H.Z*       dnorm2.F.Z*      dnorm3.F.Z*
dscale.F.Z*       iccg.F.Z*         icfac.F.Z*       lcond.F.Z*
mainflo.F.Z*      micfac.F.Z*       midflo.F.Z*      nlcond.F.Z*
norm2.F.Z*        norm3.F.Z*        sip1.F.Z*        sip2.F.Z*
sipfac.F.Z*       subflo.F.Z*       sysl1.F.Z*       sysnl1.F.Z*
sysnl2.F.Z*       theta10.F.Z*      theta20.F.Z*     theta30.F.Z*
theta31.F.Z*      theta32.F.Z*      tstep.F.Z*       tsys1.F.Z*
whead.F.Z*        wprobe.F.Z*       writ10.F.Z*      writ11.F.Z*
writ22.F.Z*       x.bigflow.co.Z*   x.bigflow.te.Z*
gemstone.15 ~/bigflo/ver11/VCS =>
```

```
total 803
   2 VCS/          792 comp.1.Z        1 hpm0/           1 test/
   1 analysis/       1 coverage/       2 non-ansi.1      3 wkdir/

bigflo_ver11/VCS:
total 3140
 192 ALFA*           2 bhzero.F*         8 sipfac.F*
 192 BETA*         576 bigflo.cpp*     136 subflo.F*
   8 COMBIG.orig*  576 bigflo.f.ori*     3 sysl1.F*
 192 HIC*          584 bigflo.pre*       5 sysnl1.F*
   4 IN1*           19 cg.F*             4 sysnl2.F*
   4 INPUT1*        11 combig.H*        20 theta10.F*
 192 KSAT*           2 dnorm2.F*        12 theta20.F*
   3 Makefile*       2 dnorm3.F*        12 theta30.F*
   7 bcl10.F*        9 dscale.F*        15 theta31.F*
   7 bcl20.F*       20 iccg.F*          15 theta32.F*
   7 bcl30.F*        3 icfac.F*          4 tstep.F*
  10 bcnl10.F*      10 lcond.F*          2 tsysl.F*
   9 bcnl20.F*      36 mainflo.F*       12 whead.F*
   8 bcnl30.F*       4 micfac.F*         9 wprobe.F*
   6 bflux1.F*      60 midflo.F*        29 writ10.F*
   6 bflux2.F*      19 nlcond.F*         7 writ11.F*
   6 bflux3.F*       3 norm2.F*          3 writ22.F*
   6 bhead1.F*       3 norm3.F*          1 x.bigflow.co*
   6 bhead2.F*      13 sipl.F*           1 x.bigflow.te*
   5 bhead3.F*      20 sip2.F*

bigflo_ver11/analysis:
total 2751
   1 ANALYZ.CRD     78 mainflo.LCL       8 mainflo.pty
   1 mainflo.ALT     1 mainflo.PAR     448 mainflo.rpt
   8 mainflo.BLK     1 mainflo.VAL    1056 mainflo.src
1080 mainflo.COM     8 mainflo.cpx       5 mainflo.tre
   1 mainflo.ENT    21 mainflo.exp
   5 mainflo.FCN    29 mainflo.met

bigflo_ver11/coverage:
total 5384
  10 CUMSUM.ANA     52 OUT12          1136 mainflo.f
  64 ESTATS        400 OUT13           168 mainflo.post
 192 HEAD_T1         1 OUTBAD           82 mainflo.tot
 192 HEAD_T2         0 PAST              1 names.seg
  28 HISTORY      1416 anotat.f77        8 report1.f77
  19 OUT10          28 history.cum     160 summary.f77
  51 OUT11        1376 mainflo*

bigflo_ver11/hpm0:
total 1080
 192 HEAD_T1      19 OUT10        52 OUT12        1 OUTBAD
 192 HEAD_T2      51 OUT11       568 OUT13        5 perf.data

bigflo_ver11/test:
total 43
   1 bigflow.perf.j    4 opt.rpt           9 perf.rpt
   4 bigflow.perf.l   16 perf.data        9 perf.rpt.old

bigflo_ver11/wkdir:
total 8153
   3 Makefile      78 bflux3.1    120 iccg.1       24 sipl.f      128 theta30.1
   1 analysis/      6 bhead1.F      3 icfac.F     112 sipl.1       15 theta31.F
   7 bcl10.F       17 bhead1.f     14 icfac.f      20 sip2.F       26 theta31.f
  18 bcl10.f       85 bhead1.1     66 icfac.1      31 sip2.f      136 theta31.1
  90 bcl10.1        6 bhead2.F     10 lcond.F     144 sip2.1       15 theta32.F
   7 bcl20.F       83 bhead2.1     21 lcond.f       8 sipfac.F     26 theta32.f
  18 bcl20.f        5 bhead3.F     88 lcond.1      18 sipfac.f    136 theta32.1
  87 bcl20.1       16 bhead3.f   1704 mainflo*     91 sipfac.1      4 tstep.F
```

```
  7 bcl30.F       79 bhead3.l     36 mainflo.F   136 subflo.F      14 tstep.f
 18 bcl30.f        2 bhzero.F     36 mainflo.f   144 subflo.f      64 tstep.l
 84 bcl30.l       13 bhzero.f     50 mainflo.l   456 subflo.l       2 tsysl.F
 10 bcnl10.F      63 bhzero.l    136 mainflo.m     3 sysll.F       13 tsysl.f
 21 bcnl10.f      19 cg.F          4 micfac.F     14 sysll.f       63 tsysl.l
112 bcnl10.l      30 cg.f         15 micfac.f     68 sysll.l       12 whead.F
  9 bcnl20.F     120 cg.l         70 micfac.l      5 sysnll.F      23 whead.f
 20 bcnl20.f      11 combig.H     60 midflo.F     16 sysnll.f      92 whead.l
112 bcnl20.l      11 combig.h     71 midflo.f     72 sysnll.l       9 wprobe.F
  8 bcnl30.F       2 dnorm2.F    216 midflo.l      4 sysnl2.F      20 wprobe.f
 19 bcnl30.f      13 dnorm2.f     19 nlcond.F     15 sysnl2.f      81 wprobe.l
 92 bcnl30.l      62 dnorm2.l     30 nlcond.f     70 sysnl2.l      29 writl0.F
  6 bfluxl.F       2 dnorm3.F    184 nlcond.l     20 thetal0.F     40 writl0.f
 17 bfluxl.f      13 dnorm3.f      3 norm2.F      31 thetal0.f    120 writl0.l
 81 bfluxl.l      62 dnorm3.l     14 norm2.f     144 thetal0.l      7 writl1.F
  6 bflux2.F       9 dscale.F     65 norm2.l      12 theta20.F     18 writl1.f
 17 bflux2.f      20 dscale.f      3 norm3.F      23 theta20.f     70 writl1.l
 80 bflux2.l      81 dscale.l     14 norm3.f     128 theta20.l      3 writ22.F
  6 bflux3.F      20 iccg.F       64 norm3.l      12 theta30.F     14 writ22.f
 17 bflux3.f      31 iccg.f       13 sipl.F       23 theta30.f     68 writ22.l

bigflo_verll/wkdir/analysis:
total 59
 44 subflo.f.Z    15 writl0.f.Z
```

Subject Analysis of Bigflow was reviewed and suggestions were noted. A.C. Baytzoglou 2/8/94

BIGFLOW Fortran Program Static and Dynamic Analysis

June 28, 1993

Earl S. Marwil
John E. Tolli
Scientific Computing Unit
Idaho National Engineering Laboratory

1. Introduction

This analysis was performed on the Cray version of the software as provided by Southwest Research Institute (SwRI).

The program was analyzed using the Craft (Cross Reference Analysis of Fortran) tool, FORWARN, the Fortran 77 analyzer, and PC-Metric. These tools provide static analysis, coverage analysis, and complexity analysis.

One sample problem was supplied along with the source code. Because the program runs much slower with the performance tracing turned on, the input file "INPUT1" was modified to run a shorter problem (1200 seconds) rather than the original one which was for 20000 seconds.

2. References

[1] N.H. Marshall and E.S. Marwil, Cross Reference Analysis of Fortran (CRAFT), EG&G-CATT-9198, EG&G Idaho, Inc., July 1991.
[2] Fortran 77 Analyzer User's Manual, National Bureau of Standards, NBS GCR 81-359, 1981
[3] FORWARN User's Guide, Quibus Enterprises, Inc., July 1991.
[4] PC-Metric User's Guide, SET Laboratories, Inc., 1987.

3. Functions

The BIGFLOW program contains 45 Fortran routines.

BIGFLOW has no alternate entry points and no extraneous subroutines.

4. Common Block Irregularities

There are 7 common blocks in the BIGFLOW program.

All common block declarations are consistent.

There are many instances of a common block being declared in a routine in which none of its elements are otherwise referenced. This is primarily due to all common blocks being included in a single comdeck.

There are 35 common block variables which are never used.

5. Interface Irregularities

No exceptions to report.

6. Local Variable Irregularities

Parameter declarations and values are consistent. All parameters are used where declared.

Local integer variables "jp1", "jp2", and "jp3" are defined but not used in subroutine "bcl30". Local real variable "yyhbb" is defined but not used in subroutine "theta30".

7. Fortran Extensions

Main program routine "mainflo" has a name which is 7 characters long. Subroutines "theta10", "theta20", "theta30", "theta31", and "theta32" all have names which are 7 characters long.

Format statements 202 and 1300 in subroutine "subflo" both have more than 19 continuation lines

8. Optimization

The following table summarizes the performance data gathered from execution of the sample problem. Only those routines exercised by the sample problem are shown (see "Coverage Analysis" for a list of routines not exercised by the sample problem, i.e., coverage = 0%). The table lists all program modules in descending order according to CPU time. To optimize code execution time, emphasis should be placed on those modules which appear highest in the listing.

The performance data show that a high percentage of the overall execution time (81.962%) is spent in the first 3 routines listed (BHEAD3, BFLUX3, BCNL30). This is due primarily to the following (applies to some or all of the 3 routines):

1) a low percentage of floating point operations which are performed in vector mode (%Vflops is small)

2) a high level of memory conflicts (MC/MR > 1)

3) a high rate of instruction buffer fetches (IBFR > 1).

A detailed optimization analysis effort should focus on these 3 areas.

PERFORMANCE DATA FOR BIGFLOW

ROUTINE NAME	Time	%ExTime	%AccumT	%Vflops	IFact	MC/MR	IBFR
BHEAD3	42.679	36.024	36.024	0.00000	0.00	1.272	0.718
BFLUX3	30.637	25.859	61.883	0.00000	0.00	1.192	0.308
BCNL30	23.788	20.079	81.962	0.00000	0.00	2.229	1.031
NLCOND	8.288	6.995	88.957	99.95661	0.00	0.238	0.086
THETA30	4.873	4.113	93.070	99.87878	0.00	0.239	0.196
SUBFLO	2.590	2.186	95.256	87.88327	0.00	0.419	0.661
CG	1.800	1.519	96.776	99.11501	0.00	0.168	0.076
DSCALE	0.916	0.773	97.549	99.95377	0.01	0.203	0.016
SYSNL1	0.910	0.768	98.317	99.98613	0.00	0.139	0.220
WHEAD	0.601	0.507	98.824	42.19134	0.00	1.164	0.959
DNORM3	0.474	0.400	99.225	97.21663	0.03	0.145	0.013
NORM3	0.439	0.370	99.595	96.26370	0.02	0.155	0.018
NORM2	0.293	0.248	99.843	96.67481	0.04	0.168	0.027
DNORM2	0.161	0.136	99.979	96.72326	0.02	0.140	0.020
WRIT10	0.010	0.009	99.987	93.95059	0.00	1.131	0.683
WRIT11	0.005	0.005	99.992	91.22159	0.00	1.221	0.713
MIDFLO	0.005	0.004	99.996	96.87376	0.00	0.620	0.892
TSTEP	0.005	0.004	100.000	0.00000	0.71	2.166	1.215
MAINFLO	0.000	0.000	100.000	100.00000	0.00	3.586	1.130
Totals (All Traced Routines)	118.474	100.000	100.000	91.49476	0.01	0.863	0.582

Key:

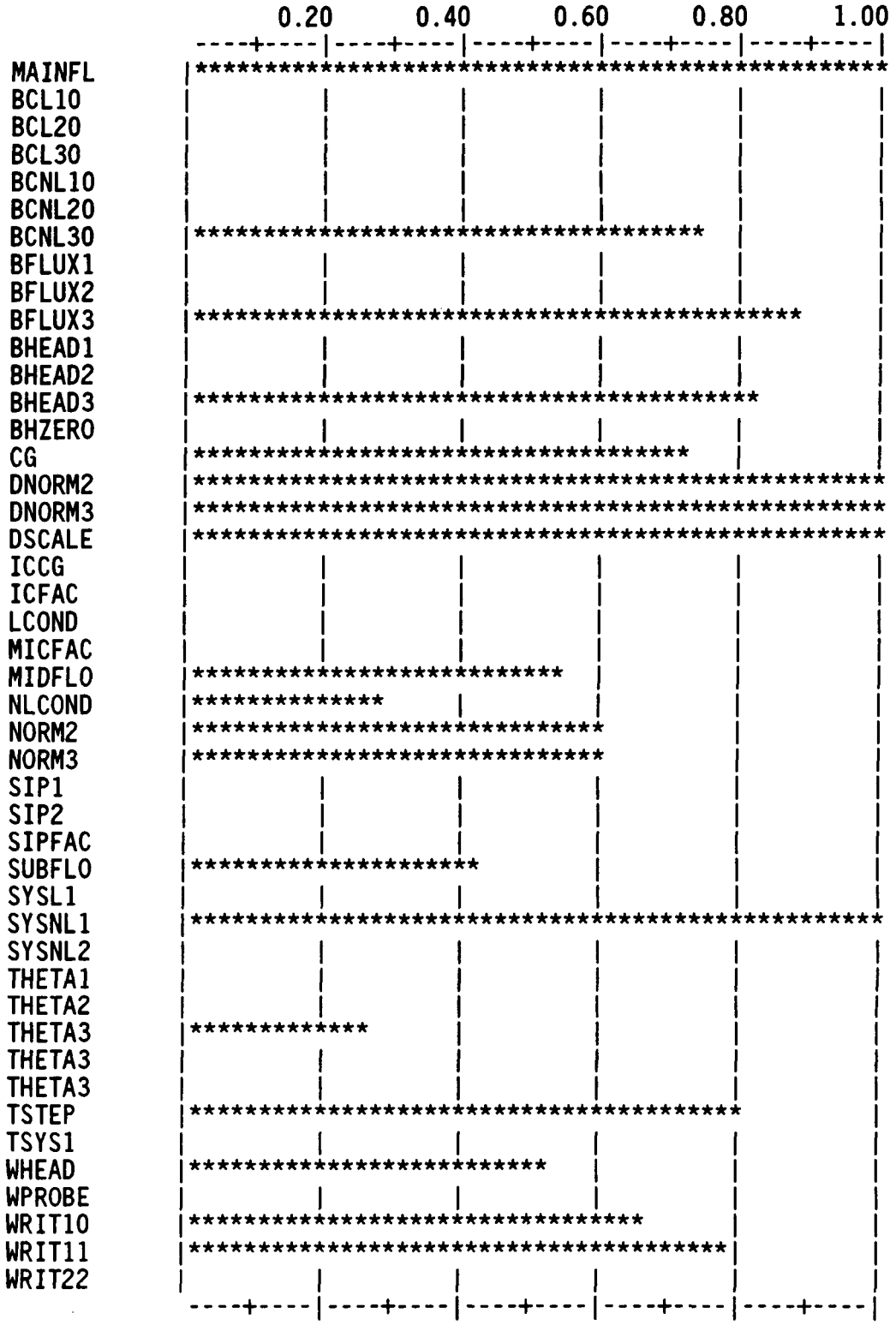
- %AccumT - accumulated percentage of total CPU time
- %ExTime - percentage of total CPU time
- %Vflops - percentage of floating point operations due to vector floating point operations
- IBFR - Instruction Buffer Fetch Rate (megafetches/sec)
- IFact - Inline Factor (total calls to routine / average time spent in routine for each call)
- MC - number of memory conflicts
- MR - number of memory references
- Time - total CPU time (sec)

9. Coverage Analysis

One sample problem was supplied. A coverage analysis shows that this problem yielded a 31% segment coverage of BIGFLOW. Sample problems provided with simulation programs typically achieve 35% to 50% coverage. A statement of software quality cannot be made for routines that have low coverage, i.e., large portions of the code are untested.

Note that 26 routines have 0% coverage. These routines are not tested with the supplied sample problem. Two routines achieve 20%-39% coverage, 5 routines achieve 40%-59% coverage, 5 routines achieve 60%-79% coverage, 2 routines achieve 80%-99% coverage, and 5 routines achieve 100% coverage. The following table shows the percent coverage for each routine.

Module Name	Number of Segments in module	Number of Segments Executed	Percent Segment Coverage
MAINFL	1	1	100.0
BCL10	68	0	0.0
BCL20	68	0	0.0
BCL30	68	0	0.0
BCNL10	86	0	0.0
BCNL20	86	0	0.0
BCNL30	86	63	73.3
BFLUX1	52	0	0.0
BFLUX2	52	0	0.0
BFLUX3	52	46	88.5
BHEAD1	64	0	0.0
BHEAD2	64	0	0.0
BHEAD3	64	52	81.2
BHZERO	13	0	0.0
CG	96	69	71.9
DNORM2	7	7	100.0
DNORM3	7	7	100.0
DSCALE	38	38	100.0
ICCG	91	0	0.0
ICFAC	7	0	0.0
LCOND	58	0	0.0
MICFAC	7	0	0.0
MIDFLO	297	162	54.5
NLCOND	85	24	28.2
NORM2	15	9	60.0
NORM3	15	9	60.0
SIP1	76	0	0.0
SIP2	122	0	0.0
SIPFAC	15	0	0.0
SUBFLO	989	412	41.7
SYSL1	7	0	0.0
SYSNL1	7	7	100.0
SYSNL2	7	0	0.0
THETA1	57	0	0.0
THETA2	57	0	0.0
THETA3	57	15	26.3
THETA3	97	0	0.0
THETA3	97	0	0.0
TSTEP	5	4	80.0
TSYS1	7	0	0.0
WHEAD	97	50	51.5
WPROBE	38	0	0.0
WRIT10	93	62	66.7
WRIT11	9	7	77.8
WRIT22	17	0	0.0
Totals	3401	1044	30.7


0.20
0.40
0.60
0.80
1.00
MAINFL
BCL10
BCL20
BCL30
BCNL10
BCNL20
BCNL30
BFLUX1
BFLUX2
BFLUX3
BHEAD1
BHEAD2
BHEAD3
BHZERO
CG
DNORM2
DNORM3
DSCALE
ICCG
ICFAC
LCOND
MICFAC
MIDFLO
NLCOND
NORM2
NORM3
SIP1
SIP2
SIPFAC
SUBFLO
SYSL1
SYSNL1
SYSNL2
THETA1
THETA2
THETA3
THETA3
THETA3
TSTEP
TSYS1
WHEAD
WPROBE
WRIT10
WRIT11
WRIT22

coverage = 0.	BCL10	BCL20	BCL30	BCNL10	BCNL20
	BFLUX1	BFLUX2	BHEAD1	BHEAD2	BHZERO
	ICCG	ICFAC	LCOND	MICFAC	SIP1
	SIP2	SIPFAC	SYSL1	SYSNL2	THETA1
	THETA2	THETA3	THETA3	TSYS1	WPROBE
	WRIT22				
0.20 <= coverage < 0.40	NLCOND	THETA3			
0.40 <= coverage < 0.60	MIDFLO	NORM2	NORM3	SUBFLO	WHEAD
0.60 <= coverage < 0.80	BCNL30	CG	TSTEP	WRIT10	WRIT11
0.80 <= coverage < 0.85	BHEAD3				
0.85 <= coverage < 0.90	BFLUX3				
coverage = 1.00	MAINFL	DNORM2	DNORM3	DSCALE	SYSNL1

Program coverage for this run =0.31

10. Complexity Analysis

Some key metrics are the number of executable statements (sloc), the number of non-blank comments (ncomt), McCabe's extended cyclomatic complexity (vg2), the number of branching statements (cgoto, ugoto, bIF, and lIF), and Halstead's predicted number of errors in (re)writing the code (bhat). Measures are normalized per 100 executable statements for ease of comparison and are listed in the table below.

The branching measures for this code indicate few unconditional GO TO statements and logical IFs for most program modules; however, some modules have higher such measures, e.g., "bhead1", "bhead2", "bhead3". Overall, this code appears to be fairly well structured.

All routines have a good ratio of non-blank comments to source code.

McCabe's extended cyclomatic complexity (vg2), normalized per 100 lines of source code, indicates high values. Generally, the routines with the highest complexity are those most likely to have defects. As a guideline, normalized measures of 15 or greater should be considered complex. A software maintenance program should focus on those routines with the highest measures.

Complexity Report by Subprogram for MAINFLO

Name	loc	sloc	cmnt	ncomt	ncomt /sloc	vg2 /sloc	cgoto	cgoto /sloc	ugoto	ugoto /sloc	bIF	bif /sloc	lIF	lif /sloc	Bhat
MAINFLO	19	6	523	5188	633.3	16.7	0	0.0	0	0.0	0	0.0	0	0.0	0
BCL10	290	113	112	111	98.2	28.3	0	0.0	15	13.3	6	5.3	15	13.3	3
BCL20	293	113	112	110	97.3	38.9	0	0.0	15	13.3	6	5.3	15	13.3	3
BCL30	282	113	122	120	106.2	28.3	0	0.0	15	13.3	6	5.3	15	13.3	3
BCNL10	347	146	132	130	89.0	26.0	0	0.0	15	10.3	6	4.1	21	14.4	4
BCNL20	336	140	129	127	90.7	35.7	0	0.0	15	10.7	6	4.3	21	15.0	4
BCNL30	322	137	124	124	90.5	27.7	0	0.0	15	10.9	6	4.4	21	15.3	4
BFLUX1	265	100	107	107	107.0	30.0	0	0.0	10	10.0	6	6.0	10	10.0	2
BFLUX2	262	100	107	107	107.0	36.0	0	0.0	10	10.0	6	6.0	10	10.0	2
BFLUX3	254	97	101	101	104.1	30.9	0	0.0	10	10.3	6	6.2	10	10.3	2
BHEAD1	247	94	103	103	109.6	44.7	0	0.0	10	10.6	0	0.0	28	29.8	2
BHEAD2	256	94	100	100	106.4	57.4	0	0.0	10	10.6	0	0.0	28	29.8	2
BHEAD3	236	91	90	90	98.9	46.2	0	0.0	10	11.0	0	0.0	28	30.8	2
BHZERO	165	20	115	115	575.0	35.0	0	0.0	0	0.0	0	0.0	0	0.0	0
CG	463	136	231	231	169.9	52.2	0	0.0	3	2.2	10	7.4	18	13.2	4
DNORM2	163	13	78	78	600.0	30.8	0	0.0	0	0.0	0	0.0	0	0.0	0
DNORM3	161	12	133	1331	108.3	33.3	0	0.0	0	0.0	0	0.0	0	0.0	0
DSCALE	250	44	181	181	411.4	54.5	0	0.0	0	0.0	2	4.5	0	0.0	1
ICCG	459	133	252	252	189.5	50.4	0	0.0	3	2.3	6	4.5	18	13.5	4
ICFAC	169	13	143	1431	100.0	30.8	0	0.0	0	0.0	0	0.0	0	0.0	0
LCOND	284	89	129	129	144.9	34.8	0	0.0	0	0.0	3	3.4	0	0.0	2
MICFAC	178	16	94	94	587.5	25.0	0	0.0	0	0.0	0	0.0	0	0.0	1
MIDFLO	1609	807	785	766	94.9	31.0	0	0.0	13	1.6	42	5.2	82	10.2	12
NLCOND	659	406	182	182	44.8	12.1	0	0.0	4	1.0	4	1.0	4	1.0	13
NORM2	173	24	84	84	350.0	37.5	0	0.0	0	0.0	2	8.3	0	0.0	0
NORM3	171	23	101	101	439.1	39.1	0	0.0	0	0.0	2	8.7	0	0.0	0
SIP1	369	116	178	176	151.7	42.2	0	0.0	5	4.3	5	4.3	16	13.8	3
SIP2	584	251	229	229	91.2	27.9	0	0.0	12	4.8	9	3.6	23	9.2	7
SIPFAC	258	52	230	230	442.3	17.3	0	0.0	0	0.0	2	3.8	0	0.0	2
SUBFLO	3008	1583	1254	1219	77.0	38.2	0	0.0	78	4.9	128	8.1	224	14.2	33
SYSL1	186	19	95	95	500.0	21.1	0	0.0	0	0.0	0	0.0	0	0.0	1
SYSNL1	207	19	103	103	542.1	21.1	0	0.0	0	0.0	0	0.0	0	0.0	1
SYSNL2	190	19	211	2111	110.5	21.1	0	0.0	0	0.0	0	0.0	0	0.0	1
THETA10	501	255	167	160	62.7	19.2	0	0.0	0	0.0	8	3.1	0	0.0	5
THETA20	485	239	174	167	69.9	20.5	0	0.0	0	0.0	8	3.3	0	0.0	6
THETA30	493	247	190	183	74.1	19.8	0	0.0	0	0.0	8	3.2	0	0.0	6

Name	loc	sloc	cmnt	ncomt	ncomt /sloc	vg2 /sloc	cgoto	cgoto /sloc	ugoto	ugoto /sloc	bIF	bif /sloc	lIF	lif /sloc	Bhat
THETA31	573	327	190	183	56.0	19.9	0	0.0	0	0.0	24	7.3	0	0.0	7
THETA32	573	327	189	182	55.7	19.9	0	0.0	0	0.0	24	7.3	0	0.0	7
TSTEP	173	16	89	89	556.3	18.8	0	0.0	0	0.0	0	0.0	2	12.5	0
TSYS1	161	10	106	106	1060.0	40.0	0	0.0	0	0.0	0	0.0	0	0.0	0
WHEAD	380	157	171	171	108.9	37.6	0	0.0	3	1.9	21	13.4	23	14.6	2
WPROBE	264	48	133	133	277.1	75.0	0	0.0	0	0.0	8	16.7	13	27.1	1
WRIT10	651	136	123	123	90.4	38.2	0	0.0	0	0.0	2	1.5	43	31.6	4
WRIT11	261	24	107	107	445.8	20.8	0	0.0	0	0.0	3	12.5	1	4.2	1
WRIT22	196	36	69	69	191.7	27.8	0	0.0	0	0.0	0	0.0	1	2.8	1

Legend of Metrics in Report

loc -- lines of code
sloc -- number of executable statements
cmnt -- total number of commnts
ncomt -- number of non-blank COMMENT statements
100*ncomt/sloc -- percent, nonblank comments to number of executable statements
100*vg2/sloc -- percent, extended complexity of number of executable statements
cgoto -- number of COMPUTED GO TO statements
100*cgoto/sloc -- percent, computed GOTO's to number of executable statements
ugoto -- number of UNCONDITIONAL GO TO statements
100*ugoto/sloc -- percent, unconditional GOTO's to number of executable statements
bIF -- number of BLOCK IF statements
100*bif/sloc -- percent, Block IF statements to number of executable statements
lIF -- number of LOGICAL IF statements
100*lif/sloc -- percent, logical IF statements to number of executable statements
Bhat -- Halstead's predicted number of errors in writing code

CNWRA 92-026

BIGFLOW: A MULTI-DIMENSIONAL CODE FOR FLOW IN HETEROGENEOUS AND VARIABLY SATURATED GEOLOGIC MEDIA (THEORY AND USER'S MANUAL — VERSION 1.0)

Prepared for

Nuclear Regulatory Commission
Contract NRC-02-88-005

Prepared by

Amvrossios C. Bagtzoglou
Rachid Ababou
Ashok Nedungadi

Center for Nuclear Waste Regulatory Analyses
San Antonio, Texas

December 1992

ABSTRACT

The objective of this report is to document the BIGFLOW code, Version 1.0, including the underlying mathematical and numerical models, and to present test problems, benchmarks, and applications of the code. BIGFLOW is actually a software package composed of a simulation code (BIGFLOW), and an interactive data processing code (DATAFLOW). The simulation code can be used for detailed modeling of transient or steady state flow systems in three-dimensional (3D) or lower-dimensional, unsaturated, partially saturated, or saturated, heterogeneous porous media. A preliminary version of this code was initially developed in 1985 by R. Ababou. Since then, it has undergone extensive modifications and improvements at the Center for Nuclear Waste Regulatory Analyses (CNWRA). Over the years, BIGFLOW has been used for studying macro-scale behavior of saturated and unsaturated flow processes in randomly heterogeneous media, for performance assessment of a high-level nuclear waste geologic repository in the presence of layers and fault, as well as for hydrologic simulations such as strip-source infiltration in layered soils, groundwater flow through lakes, etc.

The BIGFLOW code solves linear and nonlinear porous media flow equations based on Darcy's law, appropriately generalized to account for 3D, deterministic or random heterogeneity. It is written in ANSI Standard Fortran 77, is free of machine-dependent directives, and is portable without modifications to a variety of computer systems, mainframes, and workstations. An implicit finite difference scheme is used for discretization. Optionally, a modified Picard scheme is used for linearizing unsaturated flow equations (outer iterations), and a preconditioned iterative method is used for solving the resulting matrix systems (inner iterations). Iterative matrix solvers which have been extensively used so far are the Strongly Implicit Procedure (SIP) and Diagonally Scaled Conjugate Gradient (DSCG). These solution modules were especially coded to take advantage of sparsity and symmetry of the finite difference systems. The data processor (DATAFLOW), also written in ANSI Standard Fortran 77, allows interactive data entry, manipulation, and analysis of BIGFLOW's 3D datasets. A statistical analysis module is also included within DATAFLOW.

In addition to presenting detailed input instructions, this report presents the mathematical and numerical basis of BIGFLOW, as well as case studies aimed at testing and verifying the code. In particular, the code has been extensively used for simulating highly heterogeneous 3D flow systems, on large numerical grids comprising on the order of 10^6 to 10^7 nodes. For such large grids, analyses of computational performance were carried out using Cray-2 and Cray-Y/MP8 supercomputers. In addition, a number of smaller one-dimensional (1D) and two-dimensional (2D) flow tests were also developed in order to verify and benchmark BIGFLOW against other independently developed codes such as PORFLOW and CMVSFS (Connection Machine Variably Saturated Flow Simulator).

CONTENTS

CONTENTS (Cont'd)

FIGURES

FIGURES (Cont'd)

TABLES

ACKNOWLEDGMENTS

This report was prepared to document work performed by the Center for Nuclear Waste Regulatory Analyses (CNWRA) for the U.S. Nuclear Regulatory Commission (NRC) under Contract NRC-02-88-005. The activities reported here were performed on behalf of the NRC Office of Nuclear Regulatory Research (RES), Division of Regulatory Applications. The report is an independent product of the CNWRA and does not necessarily reflect the views or regulatory position of the NRC.

This report has benefitted greatly from contributions by Linda Tweedy and Michael Muller (SwRI) who conducted simulations and produced some of the graphics included herein. Discussions with Professor Art Warrick (University of Arizona) have assisted in the coding of some analytical solutions and subsequent comparisons. The superb quality of work by CNWRA support staff Ms. Mary Ann Gruhlke and Mr. Arturo Ramos is greatly appreciated. The technical reviewers of this document, Mikko Ahola, Bob Baca, and Budhi Sagar have been very generous with their time and effort. Their suggestions and contributions made this report a much more enjoyable User's Guide.

1 INTRODUCTION AND OVERVIEW

1.1 INTRODUCTION

BIGFLOW is a numerical solver for the partial differential equations governing three-dimensional (3D) flow in a heterogeneous porous medium. It accommodates steady and transient flows in saturated, partially saturated, or unsaturated media, possibly with highly heterogeneous or spatially random hydrodynamic coefficients.

The BIGFLOW package was initially developed at Massachusetts Institute of Technology (MIT) by Dr. Rachid Ababou for investigating fully 3D saturated-unsaturated flow in random porous media (Ababou, 1988). The MIT precursor of BIGFLOW was named BIGFLO. The current BIGFLOW package is the result of subsequent modifications aimed at enhancing its scope, flexibility, and computational efficiency. The package comprises two modular Fortran 77 codes, respectively, for simulating and pre- or postprocessing 3D flow fields in heterogeneous porous media. The simulator code is named BIGFLOW, and the data processing code is named DATAFLOW. In addition, auxiliary graphics display programs were also developed in conjunction with the main package. Partial documentation of BIGFLOW's main functions and applications can be found in several publications of the hydrology literature.

The interactive data processor, DATAFLOW, serves as a convenient tool for certain preprocessing tasks such as setting up boundary conditions, initial conditions, and material properties, and for postprocessing tasks such as cell-by-cell calculation of 3D flux vectors, spatial-statistical analyses, extraction of sub-dimensional datasets, and so on. Auxiliary graphic techniques for displaying large 3D datasets have also been developed, as mentioned earlier. BIGFLOW and DATAFLOW were written in Fortran 77 in full accordance with the ANSI standard, except for the possible use of the non-standard INCLUDE statement described below. The current version of BIGFLOW, possibly distributed under the alias name BIGFLO-PACK, will run on most systems including Vax's Virtual Management System (VMS) and Cray's Unix Cray Operating System (UNICOS), provided a fully American National Standards Institute, Inc. (ANSI) compatible Fortran 77 compiler is available.

One of the few modifications that may be required with some compilers concerns the non-standard INCLUDE statement. This statement is used in many places to include a set of common blocks from a Fortran file COMBIG the declared common blocks being the same in all the subroutines of BIGFLOW. If the compiler does not accept the INCLUDE statement for example, CFT77 on CRAY-2, or Fortran 200 on CYBER-205, then one may use a text editor to actually include the contents of the COMBIG file in place of each occurrence of the "INCLUDE 'COMBIG' " instruction. Naturally, doing this can be tedious and should be avoided if possible. Finally, with compilers that do accept the INCLUDE statement, a compilation error may still occur if the name of the include file is not exactly correct. On MicroVax machines running VMS, the compiler looks in the current directory for an include file named "COMBIG.FOR" or possibly "COMBIG". On machines with the UNIX-based operating system, the compiler looks in the current directory for an include file named "COMBIG". The quotes are not part of the file name.

The porous medium flow simulation by BIGFLOW satisfies locally the hydrodynamic constitutive laws of Darcy (1856) and Buckingham (1907). The governing partial differential equations are expressed either in terms of hydraulic head for saturated flow, or in terms of moisture content and

pressure head for partially saturated and unsaturated flow. The latter formulation is a mixed variable version of Richards' equation (Richards, 1931). The differential equations are discretized by an implicit finite difference scheme, two-point backward Euler in time, and seven-point centered in space. The spatial mesh is a regular rectangular lattice. The time step is generally variable and self-adjusted. The computational domain is a rectangular parallelepiped, whose coordinate system may be inclined at arbitrary angles with respect to the natural, horizontal-vertical coordinate system.

Multidimensional inputs to BIGFLOW code are specified as either node-by-node or block-by-block with the aid of the interactive data processor, DATAFLOW. The multidimensional input functions are the initial conditions, the planar boundary conditions, and up to six 3D parameters for spatially variable hydrodynamic coefficients and nonlinear constitutive relations. The boundary conditions are of three types: (i) pressure or hydraulic head, (ii) normal flux, or (iii) zero pressure gradient for gravitational drainage. The latter type of boundary conditions are described in Ababou (1988) and McCord (1991). Spatially variable and mixed-type boundary conditions can be defined separately for each of the six boundary planes.

Low-dimensional flow systems are simulated by reducing the size of the grid to just a few nodes, that is, at least one interior node plus four boundary nodes along any unmodeled dimension. The two-dimensional (2D) or one-dimensional (1D) solution is obtained from the numerical output at the middle node. This is efficient for 2D problems. This feature has been frequently implemented in previous applications of BIGFLOW. For 1D problems, dedicated 1D flow simulators should be preferred if available. Low-dimensional datasets such as transects and cross-sections can be either generated by the simulation code or extracted from existing 3D datasets by using the data processing code DATAFLOW. Finally, irregular boundaries may be introduced, in some cases, by selecting extremely small or large hydraulic conductivity values in subregions located between the physical flow domain and the outer computational boundaries. There is, however, no direct option for explicitly specifying irregular boundaries. The boundary conditions themselves can be specified for each node of the planar boundaries, and can, therefore, be spatially variable on these planes without any restriction.

The flow code solves nonlinear equations in the case of unsaturated or partially saturated flow, and linear equations in the special case of fully saturated flow. In the latter case, a sparse symmetric matrix system is generated at each time step. Steady flow problems can be solved either directly, based on steady state equations equivalent to one infinite time step, or indirectly, by time-stepping the transient flow equations up to very large times. For nonlinear flow problems, a modified Picard scheme is used. This yields a sequence of sparse symmetric matrix systems converging to the solution of the nonlinear system at each time step (outer iterations). The linearized systems have the same seven-diagonal structure as in the linear case.

In all cases, the symmetric matrix systems are solved by a fast preconditioned iterative solver (inner iterations). Several sparse solution modules that qualify as fast preconditioned iterative solvers are presently available, including the Strongly Implicit Procedure (SIP), and Diagonal Scaling Conjugate Gradients (DSCG). All arrays are stored in-core, and a diagonal-by-diagonal matrix storage scheme is used. This data structure minimizes in-core memory requirements and central processing unit (CPU) time. Furthermore, the code dynamically allocates the correct dimensions to the various arrays that describe the flow system. For a given set of inputs, the code computes the total size of the master array needed for storage of all vector-matrix variables, and starts the simulation only if the declared size is sufficient.

The total in-core memory required to simulate fully heterogeneous 3D systems is modest, about 10 words per node for saturated flow, and roughly twice as much for partially saturated or unsaturated flow. Thus, owing to the efficiency and sparseness of discretization and solution procedures, relatively large simulations are feasible in reasonable amounts of computer time and with minimal memory requirements. In the case of saturated flow, the grid size can be on the order of two hundred thousand nodes for a minicomputer such as Microvax-2, and typically one to ten million nodes for the 1985-90 generation of supercomputers such as Cray-2. For partially saturated flow problems, the feasible grid size can decrease by up to one order of magnitude relative to the linear case, depending on the nature of nonlinearities and initial-boundary conditions. For transient unsaturated flow the actual computational domain size can be made variable in time, provided this makes sense physically, for example, infiltration in initially dry soils. This option is available exclusively for unsaturated or partially saturated flow.

Hydraulic properties for saturated media are the hydraulic conductivity (K_s) and the specific storativity (S_s), both spatially variable if needed. For partially saturated or unsaturated media, the hydraulic conductivity (K) is an exponential function of pressure head (h) with a given slope (α) up to a given bubbling pressure (h_b) where K reaches the saturation value (K_s). All three coefficients K_s, α, and h_b can be spatially variable. Finally, the soil moisture retention curve $\theta(h)$ can be an arbitrary function with several parameters, two of which can be spatially variable. The code computes the soil moisture capacity by a chord-slope differentiation of $\theta(h)$. In addition, the specific storativity (S_s) is taken into account in regions of positive pressure. Note that S_s can be spatially variable for purely saturated flow but is assumed constant in space for partially saturated flow. In this case the spatial variability of the soil moisture curve and the conductivity curve is more relevant.

The governing equations are discretized by a seven-point centered finite difference scheme in 3D space, and a backwards (fully implicit) finite difference scheme in time. Since the discrete system is fully implicit in time, the case of a steady state is handled by making the transient term zero and solving for only one (virtually infinite) time step. The nonlinear flow problem, unsaturated or partially saturated, is approximately linearized by a modified Picard, or approximate Newton, iteration scheme. Thus, an approximate linear system has to be solved at each iteration step, and this is repeated for each time step.

The linear or linearized finite difference system is solved iteratively by using one of several preconditioned iterative methods. For the nonlinear case, this yields a doubly iterative cycle: outer iterations for linearization, and inner iterations for solution of a linear system. The available matrix solvers used in this code are all based on approximate factorization for the preconditioning stage. The "SIP-based" solvers use a particular type of approximate Lower Upper (LU) factorization and iterate using a modified Picard scheme (different versions are available). The Incomplete Choleski Conjugate Gradient (ICCG) solver uses an approximate Choleski factorization of the symmetric matrix, and solves the preconditioned system iteratively by the conjugate gradients method. The SIP and ICCG solvers differ both in the preconditioning method and in the iterative solution method they use.

The inputs and outputs for/from the BIGFLOW code can be processed interactively (for the most part) by using the companion code DATAFLOW, a special-purpose data processor for 1D, 2D, or 3D data processing, and statistical analysis. Random field generation must be handled separately by using a 3D Turning Bands code (CTURN). This code is not a part of the BIGFLOW package.

1.2 OVERVIEW

1.2.1 Evolution of BIGFLOW Code

A precursor of the BIGFLOW code was first developed, debugged, and tested at MIT on MICROVAX-1 and MICROVAX-2 minicomputers running the VMS operating system. It was subsequently modified to execute jobs on a CRAY-2 supercomputer, under the UNICOS operating system and using the CFT77 Fortran compiler. These modifications eliminated a few minor non-standard features, such as instructions with more than 20 continuation lines. A slightly modified version of the code is also running presently on a CYBER-205 supercomputer.

1.2.2 Main Features of BIGFLOW

The BIGFLOW code, Version 1.0, is based on a finite difference approximation of the flow equation governing the hydraulic head (saturated flow) or the pressure head (partially saturated or unsaturated flow). The governing equation is obtained from the generalized Darcy equation and the continuity equation without source/sink terms. The computational flow domain is a 3D parallelepiped, discretized into an orthogonal grid ($i\Delta x$, $j\Delta y$, and $k\Delta z$). Simulations of 1D and 2D problems can be performed by shrinking the grid along the unmodeled directions to just five nodes — three internal nodes, plus two boundary nodes. In the case of transient infiltration in dry soil, or any similar "sharp front" problem, the size of the computational domain can be made time-dependent, automatically computed by the code's algorithms. For instance, combining the previous features will allow one to simulate 1D, 2D, or 3D transient infiltration problems with variable domain size. Simulating transient groundwater flow problems is also possible, but only with fixed domain size, and no sinks or sources except at boundaries.

The boundary conditions accepted by the code are, respectively, a fixed head, a fixed flux normal to the boundary, and a null head gradient normal to the boundary. The prescribed fluxes and heads can vary arbitrarily node-by-node over each of the six plane boundaries. More generally, the type of boundary condition (head, flux, or gradient) can vary node-by-node on each plane boundary. Note finally that in the case of unsaturated or partially saturated flow, the direction of the gravity vector, with respect to the coordinate system attached to the flow domain, can be prescribed freely by the user. In other words, the parallelepiped domain may be at an arbitrary angle with respect to the vertical.

The input hydraulic properties for saturated flow are the saturated hydraulic conductivity K_s [LT^{-1}] and the specific storativity S_s [L^{-1}]. Both can be chosen arbitrarily variable in 3D space, that is defined by the finite difference grid. In the case of partially saturated or unsaturated flow, the hydraulic conductivity K(h) is assumed to be an exponential function of pressure head (h), with slope parameter (α), up to a bubbling pressure (h_b), where K(h) reaches the saturation value K_s. Each of the three coefficients K_s, α, and h_b, can be chosen spatially variable over the 3D grid. On the other hand, the moisture retention curve $\theta(h)$ may be an arbitrary nonlinear function involving several parameters, two of which can be spatially variable. The current version of the code includes subroutines for the following types of $\theta(h)$ functions: piecewise linear, exponential, and "van Genuchten." The moisture capacity $C=d\theta/dh$ is automatically computed from $\theta(h)$ by a chord-slope difference approximation. In the case of partially saturated flow, where regions of positive as well as negative pressure may exist, the input specific storativity S_s is only taken into account where pressure is positive, and it is assumed constant in space.

Finally, note that for each of the cases considered above both steady and transient flow regimes can be specified, at least in principle. However, steady flow problems may not have solutions for certain boundary conditions, for example, 1D, 2D, or 3D flow with flux q_0 at top, flux $q_1 < q_0$ at bottom, null flux on all other boundaries, and no sinks or sources. Moreover, experience suggests that some "hard" steady flow problems, both strongly nonlinear and randomly variable in space, are better solved in a transient mode with increasingly large time steps as time goes on. As an example, see the 3D steady rainfall infiltration problem in randomly heterogeneous soil, which was actually solved in the transient mode (Ababou, 1988). On the other hand, linear problems of steady groundwater flow in random aquifers were solved directly in the steady state mode (Ababou, 1988).

1.2.3 BIGFLOW Capabilities and Limitations

1.2.3.1 Randomness, Three-Dimensionality, and Spatial Resolution

Loosely speaking, the numerical flow model has been "optimized" for applications requiring a fine spatial resolution over fairly large 3D domains. All the algorithms of BIGFLOW, including the discretization scheme, are extremely sparse. The solution method is iterative rather than direct, and the modular structure of the code allows for testing various kinds of preconditioned iterative methods that are reputed to be fast (at present only SIP has been extensively tested). In its present version, the BIGFLOW code has been tested for a wide variety of flow problems in saturated and unsaturated porous media that are highly heterogeneous and/or random.

1.2.3.2 Flow Domain and Grid Geometry

BIGFLOW assumes that the 3D domain has the shape of a parallelepiped rectangle. However, that rectangle can be slanted at any angle with respect to the vertical. Furthermore, it is possible to let the size of the computational domain vary with time for certain transient flow problems. Finally, the finite difference mesh is a fixed regular network of orthogonal links and nodes. In brief, the model's grid is essentially regular and non-adaptive.

1.2.3.3 Source Terms

There are no volumetric source or sink terms in the interior of the flow domain. However, it is possible to impose local boundary sources and sinks via the flexible boundary condition routines.

1.2.3.4 Boundary Conditions

Boundary conditions can be of three types and are assumed constant in time (fixed head, fixed flux, or null head gradient). It should be noted that any of these three types of boundary conditions can be imposed at any chosen boundary node. For instance, in the case of unsaturated flow, it is possible to prescribe a surface infiltration source with a geometry approximating a circle. The same can be done for groundwater flow. It is also possible to prescribe a spatially variable head over selected boundaries. Complex boundary conditions of this nature must be stored in a file, which can be created quite efficiently by selecting routine INBC from the menu of the interactive processor DATAFLOW. A brief description of the interactive use of the DATAFLOW code will be given later.

1.2.3.5 Water Table

The classical 2D equation of unconfined flow with a free water table cannot be solved directly with the BIGFLOW code. However, one may consider instead solving a partially saturated flow problem in 3D space, including both the regions below and above the water table. This is possible in principle, but has not been fully tested. Alternatively, a more trivial approach may be justified in some cases: the unconfined problem may be treated as an equivalent confined aquifer problem, in the obvious way. Reducing the problem's dimensionality from 3 to 2 may be achieved by simulating a 3D confined flow in a thin slab.

1.2.3.6 Solution Algorithms

The governing equation is the usual combination of (generalized) Darcy equation and mass conservation equation, expressed in terms of hydraulic head for saturated flow, or pressure head for partially saturated flow. This equation is then discretized by a seven-point centered finite difference scheme in 3D space, and a fully implicit "Euler-backwards" finite difference scheme in time. The steady state case corresponds to an infinite time step (or more appropriately $\Delta t^{-1}=0$). The code includes an option for choosing either transient or steady state solution algorithms.

The nonlinear finite difference equations of partially saturated or unsaturated flow are approximately linearized by a modified Picard iteration scheme. Thus, an approximate linearized system must be solved at each iteration step, and this is repeated several times for each time step. In the case of saturated flow, of course, this nonlinear iteration loop reduces to just one step.

For saturated flow as well as linearized unsaturated flow, the resulting matrix system is solved iteratively by using a fast sparse preconditioned iterative method. For the nonlinear case, this gives a nested iteration scheme: the outer iteration loop is for linearization, and the inner iteration loop is for matrix solution. The few available matrix solvers in the current version of the BIGFLOW code are different variants of the Strongly Implicit Procedure (SIP). The SIP solver is based on an approximate LU factorization for preconditioning, and a modified Picard-type iteration scheme for converging towards the exact solution. Other routines corresponding to various Conjugate Gradient (CG) solvers have been considered, but are not fully tested at this point. In contrast, both the linear and nonlinear implementations of SIP have been extensively tested for all sorts of flow problems, including the case of randomly heterogeneous soils and aquifers (Ababou, 1988).

1.2.3.7 Code Size, Array Sizes, and Storage Requirements

The BIGFLOW source file is moderately large, about 0.5 Mbytes, or 13,000 lines of code. In most cases, nearly 100 percent of the storage that is required at link time and execution time is allocated to the master array ABIG, according to the dimension declared in the main program unit MAINFLO. The size of this master array can and should be modified as needed. In case of insufficient array size, BIGFLOW will return an error message in the output file OUTBAD and stop execution. The error message will actually show the minimum required size of the ABIG array for the job at hand. Re-running with the correct array size will fix the problem, provided that the size is compatible with the characteristics of the computer system.

Given the sparse iterative method(s) used to solve the finite difference algebraic systems in BIGFLOW, the required size of the master array is simply proportional to N, the total number of nodes

of the 3D grid. Thus, the required storage will be about 12N for a simulation of steady saturated flow with spatially variable conductivity, and possibly up to 20N for a simulation of transient unsaturated flow with spatially variable hydrodynamic parameters. These numbers are only indicative. It is not difficult to evaluate the exact factor of proportionality by looking at the BIGFLOW outputs in files OUT10 and/or OUTBAD. More details on "Inputs/Outputs" will be given in subsequent sections.

Experience shows that fairly large 3D problems can be handled routinely on minicomputers like the MICROVAX-2. The linker will accept jobs when the size of the master array is below 1-1.5 million words. For saturated flow, this corresponds roughly to a grid size of 100,000 nodes, although problems twice as large have also been solved on a MICROVAX-2 at MIT. For unsaturated flow, the admissible grid size may be about 50,000 nodes, but limitations due to relatively large CPU times have generally limited our simulations of unsaturated flow to 30,000 node grids on MICROVAX-2. On the other hand, the four-quadrant CRAY-2 will accept jobs requiring nearly 250 million words of memory, at least in principle. Thus, BIGFLOW has been used to solve several large 3D problems of stochastic groundwater flow up to 1 million nodes and of stochastic unsaturated flow up to 300,000 nodes (Ababou, 1988) on the CRAY-2 machine with CFT77 compiler.

An important detail of implementation should be emphasized in connection with problem size. When changing the dimension of the ABIG array in BIGFLOW, one should also modify the parameter LPAR located just below the ABIG dimension statement, in such a way that it has exactly the same value as the dimension of ABIG. Failure to modify LPAR may result in unpredictable output. Only these two lines of code, the ABIG dimension and the LPAR parameter statements, need to be modified by the user.

1.2.4 Previous Applications of BIGFLOW in the Literature

Several patterns of material heterogeneity were tested in various implementations of the BIGFLOW package by the code developer. These included, for example, perfectly layered structures generated with the aid of the data processing code and 3D isotropic, as well as anisotropic (imperfectly stratified) random functions of space generated by the turning bands method of Tompson and others (1989). Several examples of such generic applications of the BIGFLOW code for both saturated and unsaturated flow systems can be found in Ababou (1991b), and Ababou and others (1992a). The code is also being used at the Center for Nuclear Waste Regulatory Analyses (CNWRA) to simulate plausible patterns of variably saturated flow in layered, faulted, and inclined geological formations for Iterative Performance Analysis (IPA) of the Yucca Mountain site [see Bagtzoglou et al., (1992a) for an interim report on these simulations].

In other applications, Townley and others (1991 and 1992b) presented a study of flow through lakes, which receive groundwater on their upgradient side and discharge lake water to the regional aquifer on their downgradient side. Initially, a systematic analysis based on 2D simulations was conducted by these authors. Further analysis based on 3D simulations conducted with the BIGFLOW package revealed that the dividing surface defining the capture zone was shallower (vertically) and narrower (horizontally) than predicted with the 2D analysis.

Finally, the BIGFLOW code was also used to analyze patterns of contaminant transport in real or simulated 3D formations. For instance, Tompson and Gelhar (1990) used stochastic groundwater velocity fields generated by the BIGFLOW code to simulate stochastic contaminant transport and analyze the resulting concentration fields. Trefry and Townley (1991) and Townley and others (1992a) used the

BIGFLOW package in a preliminary analysis of 3D heterogeneous flow and transport pathways at the Koongarra uranium orebody, which is the site of the Alligator Rivers Natural Analogue Project (studied by the INTRAVAL group). Bagtzoglou and others (1992b) have presented flux vector and particle tracking results using the BIGFLOW code and its postprocessors. Their analysis was aimed at evaluating the effects of some common geological features, such as layering and fault presence, on groundwater travel time evaluations.

Specific applications and comparisons of BIGFLOW with field data include:

- **Random Porous Media and Stochastic Theories**
  - Saturated Flow (3D Random): Ababou and others (1985, 1989); and Ababou (1988).
  - Saturated Flow and Transport (3D Random): Tompson and Gelhar (1990).
  - Unsaturated Flow (3D Random): Ababou (1988); and Polmann and others (1991).
- **Other Types of Geologic Heterogeneity**
  - Unsaturated Flow with Vertical and Horizontal Layers: Ababou (1988).
  - Unsaturated Flow with Dipping Horizontal Layers and Vertical Fault: Bagtzoglou and others (1992a).
  - Unsaturated Flow and Transport with Vertical Fault: Bagtzoglou and others (1992b).
- **Experimental and Field-Related Studies**
  - Saturated Flow Through Lakes: Townley and others (1991, 1992b).
  - Saturated Flow and Transport at the Koongarra Uranium Ore Body: Trefry and Townley (1991); and Townley and others (1992a).
  - Unsaturated Flow, Strip Source Infiltration and Drainage, Las Cruces Trench Experiment 1: Ababou (1988).
- **Computational Aspects and Benchmark Tests**
  - Saturated and Unsaturated Flow in Homogeneous and Heterogeneous media: Ababou (1988, 1991a,b); and Bagtzoglou and others (1992c).
  - Saturated Flow in Homogeneous and Heterogeneous Porous Media: Ababou and others (1989,1992a).

2 MATHEMATICAL MODEL AND NUMERICAL METHOD

2.1 INTRODUCTION

The main purpose of this section is to present the direct numerical approach and, by the same token, to address the main computational issues in the case of large, multidimensional, saturated, or unsaturated flow systems through highly heterogeneous porous media. The most obvious type of application we have in view, although not the only one, is the direct simulation of single-realization stochastic flow fields with the help of the BIGFLOW numerical code. Even though the model problems will not include partially saturated flow systems where both saturated and unsaturated zones coexist, this possibility is not ruled out in principle from our modeling approach. The BIGFLOW code is based on a seven-point centered finite difference scheme in space, and a fully implicit one-step (Euler backwards) scheme in time.

2.2 GOVERNING EQUATIONS FOR SATURATED/UNSATURATED FLOW

Variably saturated flow in a heterogeneous porous medium is assumed to be governed at the local scale by the mass conservation equation, and by the generalized Darcy or Darcy-Buckingham equation relating flux to the pressure gradient. This relation is linear for saturated flow and nonlinear (quasi-linear) for unsaturated flow. In both cases, the coefficient of proportionality is called the hydraulic conductivity of the medium.

Local mass conservation in a slightly compressible and variably saturated porous medium without source/sink terms is expressed by the equation

$$\frac{\partial}{\partial t}\left[M(h) + \theta(h)\right] = -\frac{\partial q_i}{\partial x_i} \qquad (i = 1,2,3) \tag{2-1}$$

where q_i is the flux vector or specific discharge rate (L/T), h is the water pressure head, θ(h) is the volumetric soil water content (L^3/L^3) relative to the incompressed soil matrix, and M(h) is an elastic storage term (L^3/L^3) due to the combined compressibility of water and solid porous matrix. This term may be assumed negligible for unsaturated flow (M = 0 if h < 0), and proportional to pressure head for saturated flow (M = S_sh if h > 0).

The generalized Darcy equation for variably saturated flow can be expressed in an arbitrary (x_1,x_2,x_3) coordinate system as

$$q_i = -K(h)\frac{\partial}{\partial x_i}\left(h + g_j x_j\right) \tag{2-2}$$

where implicit summation on repeated indices is used. In this equation, K(h) is the unsaturated hydraulic conductivity (L/T), h is the water pressure head relative to atmospheric pressure, negative in the unsaturated zone and positive in the saturated zone (L), and g_i is a cosine vector of unit length corresponding to the acceleration of gravity with a minus sign [take g_i=(-1,0,0) if the first axis is vertical and positive downwards]. Note that the water content θ(h) and the conductivity K(h) are in general spatially variable functions of pressure head h. It should be noted that the current version of BIGFLOW

assumes that the principal components of the hydraulic conductivity tensor are alligned with the coordinate axes (i.e., $K_{ij} = 0$ for $i \neq j$).

Equations (2-1) and (2-2), and the forthcoming numerical model, are applicable in principle to the general case of variably saturated flow. That is, they are applicable to the case where the flow domain is partially saturated and partially unsaturated. However, the present discussion can be simplified by considering separately: (i) purely saturated flow ($h \geq 0$); and (ii) purely unsaturated flow ($h \leq 0$). The assumption of purely unsaturated flow is justified only if it can be shown that positive pressures do not appear at any time within the flow domain. This was indeed the case for sufficiently low rate infiltration in dry soils, even in the presence of significant heterogeneity (layers, blocks, and random fields). In fact, most of the unsaturated simulations presented in this report (except as noted) did not produce noticeable saturation zones. Accordingly, the mixed case of partially saturated flow, as handled by the BIGFLOW code, will not be treated in detail in this section.

Inserting the Darcy equation into the mass conservation equation yields the Richards' equation of unsaturated flow, here generalized to accommodate the case of a fully heterogeneous three-dimensional (3D) porous medium

$$\frac{\partial \theta(h)}{\partial t} - \frac{\partial}{\partial x_i}\left[K(h)\left(\frac{\partial h}{\partial x_i} + g_i\right)\right] = 0 \qquad (2\text{-}3)$$

On the other hand, in the case of saturated flow ($h \geq 0$), a new variable H for the total hydraulic potential is introduced

$$H = h + g_j x_j \qquad (2\text{-}4)$$

where g_j is the gravity vector as defined in Eq. (2-2), and the summation of repeated indices is over the three dimensions. Inserting a linear storativity term $M = S_s h$ in Eq. (2-1), and noting that the saturated moisture content θ_s and conductivity K_s do not depend on pressure, yields a linear parabolic equation governing saturated flow in a 3D heterogeneous porous medium

$$S_s \frac{\partial H}{\partial t} - \frac{\partial}{\partial x_i}\left(K_s \frac{\partial H}{\partial x_i}\right) = 0 \qquad (2\text{-}5)$$

Note that S_s is the "specific storativity", that is, the volume of water produced, per unit volume of the porous medium, for a unit decrease of hydraulic head ($L^3/L^3/L$). Finally, if a steady groundwater flow regime exists for the given boundary conditions, it is a solution of the linear elliptic equation

$$-\frac{\partial}{\partial x_i}\left(K_s \frac{\partial H}{\partial x_i}\right) = 0 \qquad (2\text{-}6)$$

As explained above, unsaturated and saturated flow models are developed separately based on the nonlinear equation [Eq. (2-3)], and the linear equation [Eq. (2-5)], respectively. However, the actual numerical computations implemented in BIGFLOW are based on a single, modularly structured program that can either solve the general problem of transient, variably saturated flow, or more specialized flow problems such as saturated and/or steady state flows. Accordingly, special options exist in BIGFLOW for simulating steady and/or saturated flow simply by skipping some of the algorithms. These special options are very convenient, but they are not always the best way to solve the problem. For instance,

the case of purely saturated flow can be treated like the general case of variably saturated flow for numerical testing purposes, or if the user is uncertain about the nature of flow under given initial-boundary conditions. Also, for strongly nonlinear flow problems, steady state solutions can sometimes be obtained at lesser cost by simulating transient flow starting from some well chosen initial conditions. The infinite-time steady state flow can be detected using mass balance criteria as described in Bagtzoglou and others (1992a). Examples of steady state solutions by both methods, direct steady state solution or transient time-stepping, can be found in Section 3.

2.3 HETEROGENEOUS HYDRODYNAMIC COEFFICIENTS AND CONSTITUTIVE RELATIONS

Consider first the special case of steady groundwater flow. In this case, only one spatially variable parameter need be specified, the saturated hydraulic conductivity $K_s(x)$.

In the stochastic approach the heterogeneity of the porous medium may be represented by a single realization of a statistically homogeneous random conductivity field in 3D space. Because conductivity is necessarily non-negative, a log-normal probability distribution is assumed for K_s, and accordingly, a Gaussian log-conductivity (ℓnK_s) is generated. The mean of this random field is constant, and its two-point covariance depends only on the separation vector; these statistical moments determine entirely the N-point spatial structure of the conductivity field. In practice, the Turning Band Method (TBM) can be used to generate single or multiple realizations of $\ell n\, K_s(x)$, with the desired statistical properties, on the nodes of the numerical grid. For details on the 3D TBM random field generator, see Tompson and others (1989).

In the more general case of transient unsaturated flow, the governing equation is nonlinear and depends on two constitutive relations, the moisture retention curve relating moisture content to pressure head, $\theta(h)$, and the hydraulic conductivity curve relating conductivity to pressure head, K(h). For a heterogeneous medium, these are functions of both pressure and spatial location. Below, we present the assumed analytical forms of these functional relationships as used in BIGFLOW for the unsaturated flow simulations presented in this report. Spatial dependence is modeled by taking some or all of the parameters in these nonlinear relations to be spatially variable as desired. In the stochastic case, this means that the parameters in the nonlinear relations were single realizations of random fields, again generated by the TBM (Ababou, 1988). Furthermore, cross-correlated random field parameters can also be generated by the TBM method, for example, based on the cross-correlation model proposed by Ababou and others for unsaturated media [Ababou (1988), Ababou (1991a)].

In BIGFLOW, the unsaturated conductivity-pressure relation is assumed to be a truncated exponential function involving at most three spatially variable parameters, that is

$$K(h,x) = K_s(x)\, exp\{\alpha(x)\,[h - h_b(x)]\} \qquad \text{if } h \le h_b(x)$$

$$K(h,x) = K_s(x) \qquad \text{if } h \ge h_b(x) \tag{2-7}$$

The parameters are K_s [saturated conductivity, (L/T)], h_b [bubbling pressure, or air entry pressure head, (L)], and α [scaling parameter, (L^{-1})]. Each or all of them can be taken spatially variable as desired in three spatial dimensions.

For a given length scale of interest, L, the product αL represents a convection/diffusion ratio. Indeed, it will be seen later that the discretized equations involve a grid Peclet number of the form $Pe=\alpha\Delta x$. The inverse $\alpha^{-1} = \lambda_\theta$ is a moisture dispersivity length scale (Ababou, 1991a). Alternatively, λ_θ can also be interpreted as a pore size distribution index (Yeh et al., 1985), an equivalent capillary fringe thickness (White and Sully, 1987).

A survey of the literature indicates that the exponential conductivity model is in good agreement with measured conductivity curves in a variety of soils, at least within a moderate range of soil water pressures (Ababou 1981, Bresler 1978). In their stochastic solutions, Mantoglou and Gelhar (1987a,b,c) used the same exponential model with random $K_s(x)$ and $\alpha(x)$, but zero bubbling pressure ($h_b = 0$).

For the soil moisture retention curve $\theta(h,x)$, BIGFLOW allows a choice between several functional forms, including a truncated exponential similar in form to Eq. (2-7), and the van Genuchten function (van Genuchten, 1980), among others. These relations are also allowed to include some spatially variable, 3D parameters. For instance, the van Genuchten function is modified and generalized as follows to accommodate the case of spatially variable parameters

$$\theta(h,x) = \theta_r(x) + \frac{\theta_s(x) - \theta_r(x)}{\left\{1 + [-\beta(x)\,h]^{n(x)}\right\}^{1-\frac{1}{n(x)}}} \qquad \text{if } h \le 0 \tag{2-8}$$

$$\theta(h,x) = \theta_s(x) \qquad \text{if } h \ge 0$$

where n is a dimensionless shape factor (a real number, not an integer), β is an inverse pressure head scale factor (L^{-1}), θ_s is the saturated water content, or effective porosity of the medium, and θ_r is the residual water content at very high or infinite negative pressure. Parameter θ_r is an empirical adjustment which should be taken equal to zero unless a better fit to experimental curves is obtained by using some nonzero value.

In this presentation, it is assumed *a priori* that all the parameters of Eqs. (2-2) and (2-8) are spatially variable. For instance, both the conductivity curve and the moisture retention curve were assumed spatially variable in simulations of transient infiltration in perfectly layered soils (Ababou, 1988), and in a layered and faulted rock formation (Bagtzoglou et al., 1992a,b). Finally, note that Eqs. (2-7) and (2-8) are only meant to describe the nonlinear relationships that were used in the particular applications discussed in this report. The numerical model itself is by no means limited to these particular functional forms. It is expected that future versions of BIGFLOW will incorporate a more extensive set of multi-parameter, spatially variable, nonlinear relations K(h,x) and θ(h,x), in the form of additional modules.

2.4 FINITE DIFFERENCE DISCRETIZATION IN 3D SPACE

In the case of highly heterogeneous or random porous media, and in the case of highly nonlinear flows, the fine details of the flow field must be adequately resolved in all three spatial dimensions in order to obtain meaningful solutions. High order discretization schemes such as pseudo-spectral methods, spectral finite elements, and some other weighted residual schemes, may work well for relatively smooth flow fields. In the heterogeneous case however, fine grid resolution remains a necessary requirement

even when using high order schemes; these schemes can therefore require a very significant increase of computational work relative to lower order schemes (for similar levels of accuracy).

The BIGFLOW code is based on a low-order, seven-point centered finite difference scheme in space, and a fully implicit one-step (Euler backwards) finite difference scheme in time. The spatial mesh can be rectangular, but must be uniform along each direction. On the other hand, the time step can be variable. As will be seen, the resulting coefficient matrix is symmetric and very sparse, having only seven nonzero diagonals. It also possesses interesting algebraic properties that make it well suited for fast iterative solvers such as the Strongly Implicit Procedure (SIP) and Preconditioned Conjugate Gradients (PCG).

The finite difference discretization scheme for both unsaturated and saturated flow is developed below, keeping in mind that the latter can be obtained by specializing the former. The analogy between saturated and unsaturated flow leads us to introduce a single designation (P) for the pressure head (h) and hydraulic head (H=h+z). Indeed, this is the strategy adopted for implementing the saturated-unsaturated equations in the BIGFLOW code. Accordingly, let P = h for unsaturated flow, and P = H for saturated flow.

In addition, let the vector ∇ designate the gradient operator

$$\nabla(.) = \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \frac{\partial}{\partial x_3} \right) \tag{2-9}$$

With these notations, the unsaturated flow Eq. (2-3) becomes

$$\frac{\partial \theta}{\partial t} - \nabla [K \nabla(P)] - g.\nabla(K) = 0 \tag{2-10}$$

where in general $\theta=\theta(P,x)$, and $K=K(P,x)$. As observed previously, the equation for purely saturated flow is a special case of the former. For instance, to obtain the equation governing steady state saturated flow, let $\theta = 0$, $g = 0$, and $K = K_s$, which yields

$$- \nabla [K_s \nabla (P)] = 0 \tag{2-11}$$

where in general $K_s=K_s(x)$ is spatially variable or random in 3D space.

Let us now briefly develop the seven-point centered finite difference approximation of Eq. (2-10) in 3D space. We start with the flux discretization. Recall that the first component of the flux vector is given by the generalized Darcy equation

$$q_1 = - K(P,x) \left(\frac{\partial P}{\partial x_1} + g_1 \right) \tag{2-12}$$

This flux component is now approximated by a two-point centered difference scheme as follows

$$q_1\left(x_{i+\frac{1}{2},j,k}\right) \approx -K_{i+\frac{1}{2},j,k}\left(\frac{P_{i+1,j,k}-P_{i,j,k}}{\Delta x_1}+g_1\right) \tag{2-13}$$

where the pressure is evaluated at the nodes x(i,j,k) of the regular orthogonal finite difference grid, while the flux and the conductivity are both evaluated at the mid-nodal points x(i+1/2,j,k), which define a staggered grid distinct from the original grid (i,j,k). A similar scheme is used for discretizing the flux divergence. For instance, $\partial q_1/\partial x_1$ is approximated at the grid points by the two-point centered finite difference

$$\frac{\partial q_1}{\partial x_1}\left(x_{i,j,k}\right) \approx \frac{q_1\left(x_{1+\frac{1}{2},j,k}\right)-q_1\left(x_{1-\frac{1}{2},j,k}\right)}{\Delta x_1} \tag{2-14}$$

Using similar approximations for q_2, q_3, $\partial q_2/\partial x_2$, and $\partial q_3/\partial x_3$, one obtains finally a seven-point finite difference approximation of the spatial operators of Eqs. (2-10) and (2-11) in terms of the nodal pressure P, evaluated at the grid points x(i,j,k). By the same token, the flux components q_1, q_2, and q_3 are evaluated on three different staggered grids, one for each flux component. Nevertheless, there is a provision in the postprocessing modules for generating cell-averaged flux components. This alternative form of the flux is useful for subsequent simulations of solute transport, for example, by particle tracking methods.

In order to write down explicitly the discretized spatial operators, it will be convenient to use a shorthand notation for triple indices, as shown below in square brackets

[0]	= (i,j,k)
[i ± ½]	= (i ± ½,j,k)
[i ± 1]	= (i ± 1,j,k)

The hat sign (^) will also be used to designate discretized differential operators. Accordingly, the discretized spatial operators corresponding to $\nabla[K\nabla(P)]$ and $\mathbf{g}\cdot\nabla(K)$ are given by

$$
\begin{aligned}
-\hat{\nabla}[K\hat{\nabla}(P)\,] = {} & \\
+ \left\{ \frac{K\left[i-\frac{1}{2}\right]+K\left[i+\frac{1}{2}\right]}{(\Delta x_1)^2} + \frac{K\left[j-\frac{1}{2}\right]+K\left[j+\frac{1}{2}\right]}{(\Delta x_2)^2} + \frac{K\left[k-\frac{1}{2}\right]+K\left[k+\frac{1}{2}\right]}{(\Delta x_3)^2} \right\} & P\,[0] \\
- \frac{K\left[i+\frac{1}{2}\right]}{(\Delta x_1)^2} P\,[i+1] - \frac{K\left[j+\frac{1}{2}\right]}{(\Delta x_2)^2} P\,[j+1] - \frac{K\left[k+\frac{1}{2}\right]}{(\Delta x_3)^2} & P\,[k+1] \\
- \frac{K\left[i-\frac{1}{2}\right]}{(\Delta x_1)^2} P\,[i-1] - \frac{K\left[j-\frac{1}{2}\right]}{(\Delta x_2)^2} P\,[j-1] - \frac{K\left[k-\frac{1}{2}\right]}{(\Delta x_3)^2} & P\,[k-1]
\end{aligned}
\tag{2-15}
$$

and

$$
\begin{aligned}
-g\cdot\hat{\nabla}(K) = - \Bigg\{ & \frac{g_1}{\Delta x_1}\left(K\left[i+\frac{1}{2}\right]-K\left[i-\frac{1}{2}\right]\right) \\
& + \frac{g_2}{\Delta x_2}\left(K\left[j+\frac{1}{2}\right]-K\left[j-\frac{1}{2}\right]\right) \\
& + \frac{g_3}{\Delta x_3}\left(K\left[k+\frac{1}{2}\right]-K\left[k-\frac{1}{2}\right]\right)\Bigg\}
\end{aligned}
\tag{2-16}
$$

where $\hat{\nabla}$ is the first order difference operator that approximates the gradient operator ∇. These discrete spatial operators are valid for saturated flow [Eq. (2-15)], as well as unsaturated flow [Eqs. (2-15) and (2-16)], provided a different interpretation of the "pressure" variable's, as explained earlier. A sketch of the seven-point finite difference molecule corresponding to Eq. (2-15) is shown in Figure 2-1.

2.5 BOUNDARY CONDITIONS

So far, only the spatial finite difference scheme in the interior of the computational domain (Ω) is defined. Let us now discuss the same discretization scheme under specific boundary conditions on the boundary (Γ). First, the boundary conditions accepted by the BIGFLOW code are described.

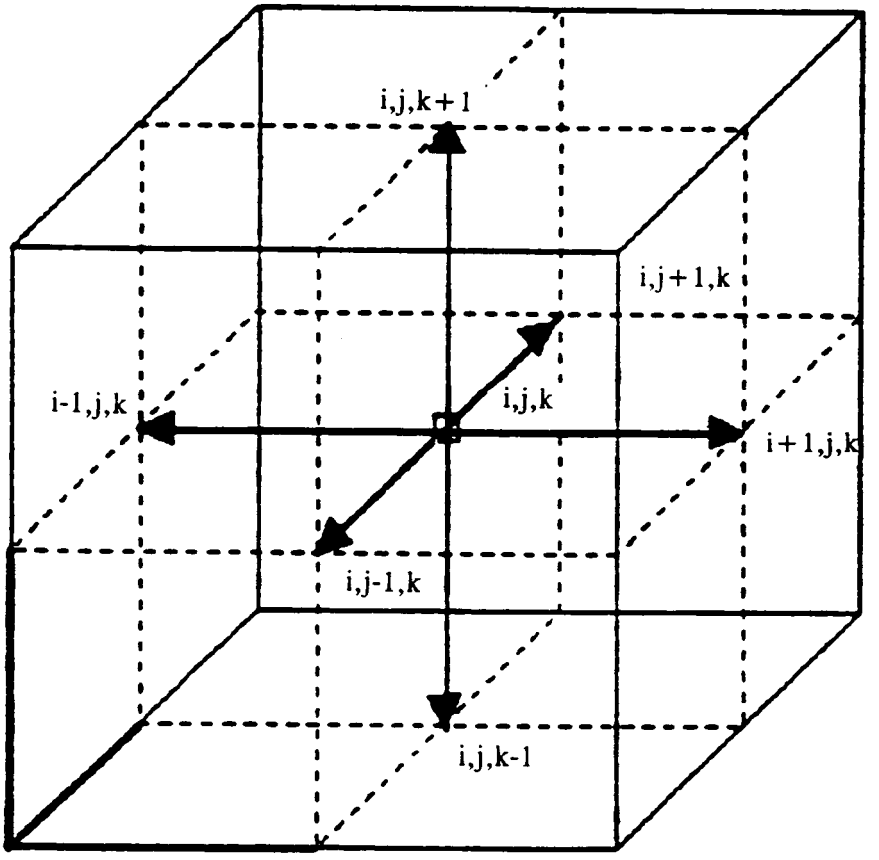


Figure 2-1. Seven-point finite difference molecule corresponding to the saturated groundwater flow equation

The boundary conditions implemented in BIGFLOW are of three types. The following classification is based on physics (the mathematical classification is indicated in parentheses).

1. Pressure condition (Dirichlet): $P = P_B(x)$, $x \in \Gamma_1$.

2. Normal flux condition (Neumann): $q \cdot n = q_B(x)$, $x \in \Gamma_2$.

3. Gravity drainage or null pressure gradient (Neumann): $\nabla P \cdot n = 0$, $x \in \Gamma_3$.

In these equations, *n* represent the vector normal to the boundary Γ_i, and $\Gamma = \Gamma_1+\Gamma_2+\Gamma_3$. Each Γ_i may consist of a disconnected set of boundary nodes. That is, the different types of boundary conditions can co-exist on any of the six planar faces composing the boundary of the hexahedral domain. For example, two types of conditions will co-exist, on a single planar face, in the following case: water supply or withdrawal at fixed pressure on part of ground surface, and zero flux on the remaining part of ground surface. This situation occurs in the test problem entitled "Flow of Water Through a Hole in a Box Under Pressure" (Section 3), which mimics the case of an artesian spring.

In the case of unsaturated flow, note that the second and third types of conditions are nonlinear. Since the boundary condition equations are eliminated by a process known as matrix condensation, they become part of the interior domain equations, which are thereafter linearized by Picard-type iterations. In short, this implies that the nonlinear boundary conditions are themselves linearized by the same iterative process. Therefore, the correct solution of the original nonlinear boundary value problem is obtained as the number of iterations increases, provided that the Picard iterations do not diverge.

There is no particular difficulty in accommodating Dirichlet-type conditions in the above equations, since fixed pressure conditions can be enforced exactly at the boundary nodes. However, Neumann-type conditions (fixed flux or pressure gradient) must be approximated by using a centered finite difference scheme as in Eq. (2-14). The order of accuracy of this approximation is the same as that in the interior of the domain, provided that the physical boundary is assumed to be located precisely at a mid-nodal point rather than at a node. The third type of condition, gravity drainage, is peculiar to unsaturated flow. In BIGFLOW, it can be implemented only for the lower horizontal boundary (if any). It was noted in the literature (McCord, 1991) that BIGFLOW and its precursor (Ababou, 1988) is one of a few multi-dimensional flow codes which provide the gravity drainage condition. McCord tested the effectiveness of this gravity drainage condition in the case of hillslope infiltration with deep water table on a truncated domain located far above the water table. He compared the results obtained with fixed head, fixed flux, and gravity drainage to the solution obtained on the full domain extending down to the water table. He concluded that the best solution, which minimizes errors with respect to water flux and (therefore) tracer transport, is obtained with the gravity drainage condition.

The centered mid-nodal formulation of Neumann-type conditions has the advantage of preserving the sparsity structure, symmetry, and positive definite characteristics of the finite difference system, as obtained after elimination of boundary pressures from the system and linearization. More precisely, note that the condensed matrix will be positive-definite if at least one boundary node is under fixed pressure. It is worth noting that this important algebraic property holds not only for saturated flow, but also for linearized unsaturated flow. More precisely, this is true in the case of a Picard-type nonlinear iterative solver as used in BIGFLOW.

2.6 MID-NODAL CONDUCTIVITIES

In order to completely define spatial discretization, it remains to be seen how the mid-nodal conductivities appearing in Eqs. (2-15) and (2-16) are to be evaluated. In the most general case, a mid-nodal conductivity coefficient like K[i+ 1/2] is a function of both pressure and spatial location, these variables being defined on a staggered grid (i+ 1/2,j,k). That is

$$K_{i+\frac{1}{2},j,k} = K\left(P_{i+\frac{1}{2},j,k},\ x_{i+\frac{1}{2},j,k}\right),\quad etc. \tag{2-17}$$

In BIGFLOW, one can choose to approximate the mid-nodal conductivities by any of the following schemes: (i) geometric weighing; (ii) harmonic weighing; or (iii) arithmetic weighing. The arithmetic weighing is not recommended, except for testing purposes. Both geometric and harmonic weighing schemes have been frequently used in the literature. However, it is recommended to use the geometric weighing scheme.

If the geometric scheme is chosen, the mid-nodal conductivity between two neighboring nodes is defined as the geometric average of nodal conductivities

$$\hat{K}_{i+\frac{1}{2},j,k} = \sqrt{K\left(P_{ijk},x_{ijk}\right)\cdot K\left(P_{i+1,j,k},\ x_{i+1,j,k}\right)} \tag{2-18}$$

In particular, for the exponential conductivity function given by Eq. (2-7) with a zero "bubbling pressure," this yields

$$\hat{K}_{i+\frac{1}{2},j,k} = \sqrt{K_s\left(x_{i+1,j,k}\right)\cdot K_s\left(x_{i,j,k}\right)}\cdot exp\left[\frac{\alpha\left(x_{i+1,j,k}\right)\cdot P_{i+1,j,k} + \alpha\left(x_{i,j,k}\right)\cdot P_{i,j,k}}{2}\right] \tag{2-19}$$

Finally, for the special case of saturated flow, this scheme reduces to a geometric mean of saturated nodal conductivities, that is, the same as Eq. (2-19), but without dependence on pressure.

The preference given to geometric over harmonic and arithmetic schemes is rather empirical. In terms of order of accuracy, the three schemes are equivalent (they preserve the second order accuracy of the spatial finite difference operator). Rather, the choice of geometric mean is motivated in part by the form of the large scale effective conductivity from approximate analytical solutions of 3D flow in random porous media. This is briefly reviewed below.

In the case of saturated flow in a random, statistically isotropic medium, the large scale effective conductivity is known to be exactly equal to the geometric mean in two dimensions; it is bounded above and below by the arithmetic and harmonic means in three dimensions, in which case it is always closer to the geometric mean (Matheron, 1967; Bakr et al., 1978; Ababou, 1988; and others). Similarly, in the case of unsaturated flow with randomized exponential conductivity function, the analytical results of Mantoglou and Gelhar [1987(a,b,c)] indicate that the effective conductivity is roughly proportional to $K_g \exp\{\langle\alpha h\rangle\}$, where K_g is the ensemble geometric mean of the random saturated conductivity, h is pressure head, and the <brackets> represent the ensemble mean operator. One should observe the similarity of this expression with Eq. (2-19).

In summary, the geometric weighing scheme [Eq. (2-18)] seems justified by analogy with the above-mentioned theoretical results for random, saturated as well as unsaturated, multi-dimensional

porous media. In addition, extensive numerical experiments by Vauclin and others (1979), and Haverkamp and Vauclin (1979), indicate that the geometric scheme yields the most accurate results in the case of one-dimensional (1D) infiltration in homogeneous soils.

2.7 TIME DISCRETIZATION

Having developed the spatial discretization scheme, one can now proceed to discretize the transient equation of unsaturated flow in time. We choose to implement a fully implicit one-step finite difference scheme, known to be first order accurate in time. From the semi-discretized form of Eq. (2-10), this choice leads to the fully discretized equation

$$\frac{\theta^{n+1}(P) - \theta^{n}(P)}{\Delta t_{n+1}} - \hat{\nabla}\left[K(P)\hat{\nabla}P\right]^{n+1} - g\cdot\hat{\nabla}\left[K(P)\right]^{n+1} = 0 \qquad (2\text{-}20)$$

where

$$\Delta t_{n+1} = t_{n+1} - t_n \qquad (2\text{-}21)$$

is the variable time step. The spatial difference operator $\hat{\nabla}$ is the same that appeared in Eqs. (2-15) and (2-16). For clarity, the direct dependence of θ and K on spatial location has been omitted, as well as the discrete-space index (i,j,k). It is understood that, in general, $\theta=\theta(P_{ijk},x_{ijk})$, $K=K(P_{ijk},x_{ijk})$, and that P stands for P_{ijk} in Eq. (2-20).

The fully implicit Euler backward scheme was selected among other one-step implicit schemes in view of numerical experiments reported in the literature (e.g., Vauclin et al., 1979, among others). The Euler forward explicit scheme was ruled out because of the well known fact that it requires a stringent stability condition, $\Delta t \leq 2\ D\ \Delta x^2$ for the linear diffusion equation (a similar condition is likely required for the nonlinear diffusion equation). In the case of unsaturated flow, the nonlinear soil moisture diffusivity (D=K/C) may become quite large in wet soils. Thus, the time step may have to be taken dramatically small in order to satisfy the explicit scheme stability condition, and there may be additional instabilities due to the nonlinear gravitational term.

On the other hand, it can be shown by Fourier analysis that implicit schemes are linearly stable, regardless of time step size (unconditionally stable). However, for nonlinear diffusion problems, the proof of unconditional stability is based on Fourier analysis assuming "frozen coefficients." Due to this approximation, one should keep in mind that the nonlinear stability of the implicit scheme is not truly guaranteed for strongly nonlinear equations like unsaturated flow. It is nonetheless probable that the implicit scheme allows larger time steps than the explicit scheme, as demonstrated experimentally by Vauclin and others (1979), and others. The current version of BIGFLOW has no provision for implementation of an explicit time-integration scheme. For more on nonlinear stability of the discretized unsaturated flow equation, see Ababou (1990).

2.8 ITERATIVE LINEARIZATION

In the case of unsaturated flow, an approximate linearized solution method must be devised to deal with the nonlinear system. For instance, implementing a Picard iteration scheme will transform the sparse and symmetric nonlinear system into a more tractable sequence of equally sparse and symmetric

matrix systems. The question of solving large symmetric matrix systems, and of coupling the matrix solution process with the nonlinear iteration process, will be discussed later. Here, the iterative linearization approach that transforms the nonlinear system into a sequence of linear systems is developed.

Let us describe how the highly nonlinear spatial and temporal operators of the unsaturated flow system are linearized iteratively using a Picard-type approach. The modified Picard iteration scheme defined below approximates Eq. (2-20) as a sequence of systems (k=0,1,2 ...) where the unsaturated conductivities appear linearly at each iteration level

$$\frac{\theta^{n+1,k+1} - \theta^{n+1,k}}{\Delta t_n + 1} - \hat{\nabla}\left[K^{n+1,k}\hat{\nabla}\left(P^{n+1,k+1} - P^{n+1,k}\right) \right] \approx$$

$$- \left[\frac{\theta^{n+1,k} - \theta^{n}}{\Delta t_{n+1}} - \hat{\nabla}\left(K^{n+1,k}\,\hat{\nabla}P^{n+1,k}\right) - g\cdot\hat{\nabla}\left(K^{n+1,k}\right) \right] \tag{2-22}$$

Note that k+1 represents the current iteration level, while n+1 represents the current time step. The residual term on the right hand side of Eq. (2-22) is known, since it depends only on the previous iteration level (n+1,k), and on the previous time step (n+1,0), simply denoted (n). The spatial operator on the left hand side operates on a pressure increment rather than pressure itself. This incremental formulation, or "modified" Picard scheme, was obtained by subtracting a known quantity from both sides of the standard Picard equation. The modified Picard scheme is not only more elegant, but also computationally more stable than the standard Picard scheme with respect to round-off errors (Ababou, 1988).

There remains a pressure-dependent moisture content in Eq. (2-22). To obtain a fully linear system at each iteration level, the storage term is now linearized by using a first order difference approximation of the θ(P)-increment. Specifically, one may construct a "chord-slope" approximation of the soil moisture capacity ($C=\partial\theta/\partial P$) as follows

$$C\left(P^k\right) \approx \frac{\theta\left(P^k\right) - \theta\left(P^0\right)}{P^k - P^0} \tag{2-23}$$

This chord-slope formula is now inserted in a first order difference approximation of the θ(P)-increment

$$\frac{\theta^{k+1} - \theta^{k}}{\Delta t_{n+1}} \approx \frac{C\left(P^k\right)}{\Delta t_{n+1}}\left(P^{k+1} - P^k\right) \tag{2-24}$$

where all variables are implicitly taken at time level (n+1), except for P^0 which is the pressure at iteration level (0), that is, the solution of the previous time step.

Alternatively, a pointwise evaluation of moisture capacity could be used instead of Eq. (2-24). However, as noted in Ababou (1988), the chord-slope scheme should be favored, as it appears to improve the convergence of nonlinear iterations (Huyakorn et al., 1984), and is mass-conservative in the sense defined by Milly (1985). Note that the resulting finite difference system is a discrete approximation of the mixed form equation, rather than the pressure-based Richards' equation. This mixed form is identical to that developed in Ababou (1988). A similar mixed form approach was adopted by Bouloutas (1989) and Celia and others (1990).

Combining Eqs. (2-22) and (2-24) finally leads to a sequence of fully linear systems of equations to be solved for incremental pressures. Omitting the spatial index (i,j,k) for convenience, the iterative sequence is given by

$$\frac{C^{n+1,k}}{\Delta t_{n+1}} \delta P^{n+1,k+1} - \hat{\nabla}\left[K^{n+1,k} \hat{\nabla} \left(\delta P^{n+1,k+1}\right)\right] \approx R^{n+1,k} \tag{2-25}$$

where δP is the unknown pressure increment taken over consecutive iteration levels, that is

$$\delta P^{n+1,k+1} = P^{n+1,k+1} - P^{n+1,k} \tag{2-26}$$

and R is a known linearized residual, given by

$$R^{n+1,k} = -\left[\frac{C^{n+1,k}}{\Delta t_{n+1}} \left(P^{n+1,k} - P^{n}\right) - \hat{\nabla}\left(K^{n+1,k} \hat{\nabla} P^{n+1,k}\right) - g \cdot \hat{\nabla} K^{n+1,k}\right] \tag{2-27}$$

With just one iteration in the outer iteration loop (k), the nonlinear system solver yields a simple linearized implicit finite difference scheme. For 1D flow and spatially constant soil properties, the finite difference equation, in terms of pressure head (h), is

$$C_i^n \frac{h_i^{n+1} - h_i^n}{\Delta t_n} = K^n_{i+\frac{1}{2}} \left(\frac{h_{i+1}^{n+1} - h_i^{n+1}}{\Delta z}\right) - K^n_{i-\frac{1}{2}} \left(\frac{h_i^{n+1} - h_{i-1}^{n+1}}{\Delta z}\right) + g \cdot \frac{K^n_{i+\frac{1}{2}} - K^n_{i-\frac{1}{2}}}{\Delta z} \tag{2-28}$$

The superscript n indicates the time level, and g is a gravity factor which takes the value $g = 0$ for horizontal flow and $g = \pm 1$ for vertical flow, depending on the chosen orientation of the z axis.

The form of the finite difference system [Eq. (2-28)] suggests that, while the nonlinear diffusion term is treated implicitly, the nonlinear "gravity term" is in fact treated explicitly as it appears only on the right hand side of the linearized system. Based on this remark, it is interesting to examine how this discrepancy (implicit versus explicit) affects the numerical stability of the solution. The proposed method is to develop a Fourier stability analysis of Eq. (2-28) with partially frozen coefficients. This is analogous to the usual frozen coefficients analysis except that here the nonlinearity of the gravity term is taken into account rather than simply "frozen." This is achieved by using the following quasi-linear approximation

$$g \frac{K^n_{i+\frac{1}{2}} - K^n_{i-\frac{1}{2}}}{\Delta z} = g \frac{K_i^n}{\Delta z} \frac{\alpha}{2} \left(h_{i+1}^n - h_{i-1}^n\right) + O\left(\Delta z^2\right) \tag{2-29}$$

where $\alpha = d(lnK)/dh$.

Admittedly, this approximation will not be accurate unless the absolute value of $\alpha\ (h_{i+1} - h_i) \approx \alpha \Delta z\ (\partial h/\partial z)$ is on the order of unit or less. Nevertheless, even rough indications on the numerical stability of the nonlinear unsaturated flow system can be useful given the lack of theoretical results in this area. With this provision, inserting Eq. (2-29) into Eq. (2-28) yields the mixed implicit/explicit scheme

$$- \bar{D}_{i-\frac{1}{2}} \cdot h_{i-1}^{n+1} + \left(1 + \bar{D}_{i-\frac{1}{2}} + \bar{D}_{i+\frac{1}{2}}\right) h_i^{n+1} - \bar{D}_{i+\frac{1}{2}} \cdot h_{i+1}^{n+1} \approx$$

$$- \frac{1}{2} g\ \alpha \Delta z \bar{D}_i \cdot h_i^n + \frac{1}{2}\ g\ \alpha\ \Delta z\ \bar{D}_i\ h_{i+1}^n \tag{2-30}$$

where $\bar{D}$ is the dimensionless diffusion coefficient

$$\bar{D}_{i\pm\frac{1}{2}} = \frac{K_{i\pm\frac{1}{2}}}{C_i}\ \frac{\Delta t}{\Delta z^2} \tag{2-31}$$

Now, a standard Fourier stability analysis of Eq. (2-30) with "frozen" diffusion coefficients yields the complex time amplification factor

$$\rho \approx \frac{1 + j\alpha g \Delta z\ \bar{D}_i \sin(k\Delta x)}{1 + \left(\bar{D}_{i+\frac{1}{2}} + \bar{D}_{i-\frac{1}{2}}\right)[1 - \cos(k\Delta z)] - j\left(\bar{D}_{i+\frac{1}{2}} - \bar{D}_{i-\frac{1}{2}}\right) \sin(k\Delta z)} \tag{2-32}$$

where k is the Fourier mode or wave number taking discrete values in (π/L,..., nπ/L). Applying the inequality $|\rho| \leq 1$ to Eq. (2-32) finally leads to the necessary and sufficient stability conditions

$$|\alpha g \Delta z| \leq 2\sqrt{1 + \left(2\ \frac{K_i}{C_i}\ \frac{\Delta t}{\Delta_z^2}\right)^{-1}} \tag{2-33}$$

In summary, Eqs. (2-25) and (2-27) define a linearized system of equations approximating the unsaturated flow equation at each time step (n+1) and each iteration level (k+1). The following points should be recalled:

1. The spatial index (i,j,k) has been omitted for clarity.

2. The nonlinearity and spatial variability of hydrodynamic coefficients were fully taken into account although not shown explicitly.

3. The spatial difference operator ∇ was previously defined through Eqs. (2-15) and (2-16).

4. The dependent variable P represents either pressure head for unsaturated flow or total hydraulic head for saturated flow.

Obviously, the linearization sequence is not needed for saturated flow. In that case, the equations reduce to the zero iteration, k = 0, taken with the coefficients appropriate for saturated flow, as described earlier in Eq. (2-11). Finally, Eqs. (2-25) through (2-27) are also valid in the general case of variably saturated flow. This flow regime was actually assumed *a priori* for all "unsaturated" test problems, thereby allowing pressures to be positive as well as negative, as the case may be.

2.9 STRUCTURE OF THE ALGEBRAIC SYSTEM

In this section, it will be most convenient to reformulate the finite difference equations in matrix-vector form. Bold face upper cases are used for matrices ($\boldsymbol{A}$), and bold face lower cases for vectors ($\boldsymbol{p}$).

Consider the discretized Eqs. (2-25) to (2-27), which are either linear (for saturated flow) or linearized (for unsaturated flow). In the case of unsaturated flow, let $\boldsymbol{C}$ be the diagonal matrix of specific moisture capacities of Eq. (2-23), $\boldsymbol{K}$ the matrix of unsaturated conductivities arising from the elliptic operator of Eq. (2-15), $\boldsymbol{\delta p}$ the vector of incremental pressures of Eq. (2-26), and $\boldsymbol{b}$ the right hand side vector. The latter includes the residual term on the right hand side of Eq. (2-27), as well as additional terms obtained after elimination of boundary values from the linearized system (matrix condensation). With these notations, the iterative sequence of linearized unsaturated flow systems can be written equivalently as

$$\left[\frac{\boldsymbol{C}^{n+1,k}}{\Delta t_{n+1}} + \boldsymbol{K}^{n+1,k}\right] \delta \boldsymbol{p}^{n+1,k+1} = \boldsymbol{b}^{n+1,k} \tag{2-34}$$

The case of saturated flow leads to a similar algebraic equation, with $\boldsymbol{C}$ a diagonal matrix of specific storativities rather than moisture capacities, and $\boldsymbol{K}$ a matrix of saturated rather than unsaturated conductivities. The more general case of variably saturated flow also leads to a similar equation, with $\boldsymbol{C}$ a diagonal matrix containing both specific storativities and moisture capacities.

The simplest case is that of steady saturated flow, which can be used as a base case to analyze generic properties of the algebraic system [Eq. (2-34)]. To reduce the transient Eq. (2-34) to a steady state equation, one may either take a single infinite time step ($\Delta t \to \infty$), or else use the steady state system [Eq. (2-11)] after expressing it in matrix-vector form. When solving directly for steady state, the method adopted in BIGFLOW is to let $\Delta t^{-1} = 0$ in Eq. (2-34), where Δt^{-1} is a distinct Fortran variable equal to the inverse time step. Either way, for a linear steady state flow problem, the result is of the form

$$\boldsymbol{K} \cdot \boldsymbol{p} = \boldsymbol{b} \tag{2-35}$$

where $\boldsymbol{p}$ is the vector of pressures (total pressures here), and $\boldsymbol{b}$ is a vector containing the boundary terms that arise after matrix condensation. Note that the structure of the conductivity matrix, $\boldsymbol{K}$, would remain unchanged in the case of unsaturated flow (within each Picard iteration). The properties of $\boldsymbol{K}$ are as follows.

In both Eqs. (2-34) and (2-35), the conductivity matrix $\boldsymbol{K}$ is a weakly diagonal-dominant symmetric matrix with only seven nonzero diagonal lines. In the transient case, the system is more strongly diagonally dominant and therefore better conditioned as Δt decreases. This is due to the

presence of a diagonal storage matrix of the form $C/\Delta t$. This important property should be kept in mind when experimenting with transient solution strategies: smaller time steps may be advantageous since they may require fewer iterations to achieve convergence.

More precisely, it can be shown that the coefficient matrices of systems [Eqs. (2-34) and (2-35)] are symmetric, positive-definite M-matrices. In an M-matrix, all diagonal terms are strictly positive and all off-diagonal terms are negative or null. Symmetry and positive-definiteness are required for Conjugate Gradient (CG) matrix solvers. The M-matrix property is a also required for some preconditioned iterative solvers. The reader is referred for example to Meijerink and Van der Vorst (1977), who proved the existence of Incomplete Choleski (IC) factorizations, and Chen (1988), who proved a theorem on convergence of the Strongly Implicit Procedure (SIP), both requiring the M-matrix property. It should be emphasized that these algebraic properties do not necessarily carry on to other spatial discretization methods (Ababou, 1988; and Ababou et al., 1989).

The appropriate strategy for solving the above finite difference matrix systems depends, in part, on their size. As explained previously, the matrices are sparse but can be very large when simulating 3D heterogeneous flow systems on high resolution grids. Consider for instance the case of a cubic grid of size $N = n^3$. If N is on the order of 10^5 to 10^6, direct solution by substitution (Gauss) or by exact triangular factorization (Choleski) seems infeasible due to prohibitive computational work and storage requirements. Indeed, the banded triangular matrices arising in the solution process are full within a band of size $O(n^2)$, even though the original system is very sparse. Here, n is the uni-dimensional size of the cubic grid, for example, $n=100$ for a million node grid. Typically, a standard band solver like Gauss or Choleski will require $O(n^7)$ floating point operations and $O(n^5)$ words of storage.

Besides being large, the coefficient matrix will also have highly heterogeneous coefficients due to spatial variability of material properties, nonlinearity, or both. This can lead to large matrix condition number, a measure of the difficulty of solving the associated system. The condition number of matrix $\boldsymbol{K}$ is typically proportional to the squared number of nodes along the largest side of the grid. See Golub and Van Loan (1989) for a precise definition of condition numbers, Ababou (1988), and Ababou and others (1992a) for more details on the condition number of conductivity matrices. The influence of condition number, system size, and coefficient variability on the convergence of iterative solvers, is briefly analyzed below. See also Ababou and others (1992a) for convergence of PCG on very large matrix systems arising in saturated flow.

2.10 OVERVIEW OF ITERATIVE MATRIX SOLVERS

Given the severe conditions described above, the optimal solution strategy may be a combination of direct and iterative methods. In other words, the sparse matrix system is first approximated by a direct factorization or matrix splitting method (preconditioning step). The resulting preconditioned system, which is also sparse, is then solved iteratively to obtain the solution of the original system (iterative steps).

Matrix solution methods considered in BIGFLOW are the SIP and various PCG solvers. The SIP solver and the Incomplete Choleski Conjugate Gradients (ICCG) are two types of iterative methods where the preconditioner is based on an approximate triangular factorization of the sparse, symmetric coefficient matrix. On the other hand, the Diagonal Scaling Conjugate Gradient (DSCG) solver involves a matrix preconditioner that is particularly inexpensive in terms of storage and computer time, since it

consists only of a symmetric diagonal scaling of the coefficient matrix. The solvers which have been most extensively debugged and successfully tested in BIGFLOW are: (i) DSCG solver; and (ii) several variants of the SIP solver. Therefore, DSCG and SIP are the recommended solvers, although the user is free to experiment with other solvers as needed.

The SIP, DSCG, and other solvers were coded in BIGFLOW as separate modules. These solver modules employ a specialized vector storage scheme, suitable for the 3D finite difference system at hand. For example, the symmetric seven-diagonal coefficient matrix is stored as four vectors, one for the main diagonal and three for the lower nonzero off-diagonals. Each vector is in fact a 3D array a(i,j,k), with triple index representing cartesian grid coordinates. In what follows, the BIGFLOW implementations of SIP and DSCG are described for the specific matrix systems at hand. For completeness, a brief review of the literature is given below and in the following sections describing DSCG and SIP, respectively.

Successful vectorization of a solver module can speed-up computations by two orders of magnitude on vector supercomputers. Numerical analysis of triangular factorization preconditioners have shown that a suitable renumbering of the nodes can circumvent to some extent their inherent lack of ability to vectorize. Although this approach was not pursued in BIGFLOW, the interested reader is referred to Van der Vorst (1981) for Vector-ICCG implementations. On vector machines, DSCG is essentially fully vectorizable. Thus, BIGFLOW's DSCG module executes almost entirely in vector mode when implemented on a Cray-2 or Cray-Y/MP system, and this without any modification of the code or its modules.

Finally, we point out that the method of choice depends on its ability to be implemented concurrently (for parallel machines). Dougherty (1991), and Bagtzoglou and others (1992c), obtained satisfactory results in terms of computer time with massively parallel versions of DSCG coded in the CM-FORTRAN language on the Connection Machine CM-2. Ababou and others (1992a) obtained similar timings with a coarse-grained parallel implementation of DSCG on Cray-Y/MP8, using Cray's autotasking utility. In the latter work, the simulations were performed using the unmodified, ANSI Fortran 77, BIGFLOW code.

It is, therefore, concluded that the method of choice for ill-conditioned steady state linear problems — which require the largest amount of iterations — may be either DSCG, or perhaps one of the newly developed vector variants of ICCG mentioned above. On the other hand, for relatively well conditioned transient systems and/or for nonvector machines, the standard SIP solver may remain competitive. However, in transient and/or nonlinear flows, the preconditioned matrix system must be computed anew at each time step and/or Picard iteration. On any given computer system, the DSCG preconditioner requires fewer operations than that of SIP or ICCG. Overall, DSCG should be the first choice, particularly on vector and parallel machines, and SIP the second choice.

2.11 DIAGONAL SCALING CONJUGATE GRADIENT (DSCG SOLVER)

2.11.1 Brief Review of Conjugate Gradient (CG) and Preconditioned CG Solvers

The CG method is an effective solver for symmetric matrix systems that yields the exact solution after at most N iterations for an $N \times N$ matrix (Hestenes and Stiefel, 1952). For large sparse matrices, CG may be considered as an iterative accelerator of convergence rather than an exact solver, and it works best with preconditioners that tend to cluster the eigenvalues of the system (Kershaw, 1978;

and Gambolati and Perdon, 1984). Numerical experiments with various Preconditioned CG solvers were developed by Kershaw (1978), Kuiper (1981,1987), Gambolati and Perdon (1984), Jackson and Robinson (1985), and Meyer and others (1989), among many others.

One popular preconditioner is the IC factorization (Du Pont et al., 1968; Meijerink and Van der Vorst 1977; and Gustafsson 1978). The IC factorization is applicable, in principle, only to M-matrices like the finite difference matrices arising in BIGFLOW. Two variants of IC factorization were coded for use as CG preconditioners in BIGFLOW; however, they have not been extensively tested to date, and should be avoided in routine simulations. Furthermore, these IC preconditioners are computationally intensive and do not vectorize well on current supercomputers. This is due to the inherent recursiveness of triangular factorization. The precise "loop dependencies" that prevent vectorization depend much on the particular multi-dimensional difference stencil, node ordering scheme, and programming style.

For these reasons, the simple Diagonal Scaling (DS) algorithm may be an attractive alternative to the more "expensive" preconditioners reviewed above. The symmetric DS algorithm is a fully vectorizable, extremely sparse, and symmetric preconditioner, which can be advantageously used in conjunction with CG. The combination of DS and CG yields the DSCG solver. In what follows, we will now focus exclusively on DS in conjunction with CG iterations. First, the basic algorithm is presented (below), and secondly, a summary assessment of the computational efficiency of this solver is provided.

2.11.2 Basic DSCG Algorithms

The DSCG solver is based on the following algorithms. First, let the finite difference system be expressed in the generic algebraic form

$$A \cdot y = b \tag{2-36}$$

where y is the vector of nodal pressures ($P_{i,j,k}$); b is the vector containing boundary terms as well as residual terms from nonlinear iteration and/or time-stepping; and A is the seven-diagonal, heterogeneous conductivity matrix. Recall that A is symmetric positive-definite, weakly diagonal dominant, and possesses the M-matrix property, having strictly positive diagonal elements and negative off-diagonal elements.

A diagonal-by-diagonal storage scheme is used for the sparse symmetric matrix A. Specifically, only four vectors are stored, corresponding to the main diagonal and three nonzero off-diagonal lines in the lower half of the matrix. Each "vector" is in fact represented by a triple-indexed array variable (one index per spatial dimension). This specialized algebra and data structure minimizes both storage and Central Processing Unit (CPU) time. The total memory required for solving heterogeneous 3D systems with DSCG is modest, about 13 words per node for saturated flow, and up to twice as much for unsaturated flow.

The DSCG algorithm is implemented in two steps, first by applying symmetric DS to the original system, and secondly by solving the scaled symmetric system using the CG method. First, symmetric DS is implemented as shown in Figure 2-2. Secondly, given an initial guess for the (scaled) solution vector, the iterative CG algorithm (Golub and Van Loan, 1989) is implemented to solve the (scaled) symmetric system as indicated in Figure 2-3.

- Diagonal preconditioner: $\mathbf{D}_0 = \text{diag}(\mathbf{A})$
- Scaled coefficient matrix: $\mathbf{A}_* = \mathbf{D}_0^{-1/2}\, \mathbf{A}\, \mathbf{D}_0^{-1/2}$
- Scaled right-hand side: $\mathbf{b}_* = \mathbf{D}_0^{-1/2}\, \mathbf{b}$
- Scaled system: $\mathbf{A}_*\, \mathbf{y}_* = \mathbf{b}_*$
- Scaled solution: $\mathbf{y}_* = \mathbf{D}_0^{+1/2}\, \mathbf{y}$

Figure 2-2. Symmetric diagonal scaling

0. Initialize parameters: $\mu_{old} = \mu_{new} = \omega = 0$

   Initialize residual vector: $\mathbf{r} = \mathbf{b} - \mathbf{A}\,\mathbf{y}$

   Initialize search vector: $\mathbf{p} = \mathbf{0}$

1. Update μ-parameter: $\mu_{old} = \mu_{new}$ $\mu_{new} = 1/(\mathbf{r}^T.\mathbf{r})$
2. Update search vector: $\mathbf{p} = \mathbf{r} + (\mu_{old}/\mu_{new})\, \mathbf{p}$
3. Compute auxiliary vector: $\mathbf{z} = \mathbf{A}\,\mathbf{p}$
4. Compute ω-parameter: $\omega = [\mu_{new}\, (\mathbf{p}^T \cdot \mathbf{z})]^{-1}$
5. Update solution vector: $\mathbf{y} = \mathbf{y} + \omega\mathbf{p}$

   Update residual vector: $\mathbf{r} = \mathbf{r} - \omega\mathbf{z}$

6. GO TO step 1 IF stopping criterion NOT satisfied, ELSE STOP.

Figure 2-3. Conjugate gradient iterations

To obtain the full DSCG solver, one should replace the **A**, **y**, and **b**, of Figure 2-3 by the $\mathbf{A}_s$, $\mathbf{y}_s$, and $\mathbf{b}_s$ quantities defined in Figure 2-2. Note that D_0 is the main diagonal of the unscaled matrix A. In Step 6, the stopping criterion may be a number of iterations or an error norm, such as the L_2- or L_∞-norm of the error vector $(y_{new}-y_{old})$. In Step 3, the matrix-vector product $\mathbf{z} = \mathbf{Ap}$ is computed as a sum of seven shifted dot products, one for each nonzero diagonal line of **A**. All other array operations are straight dot products, except for the L_∞-norm of error.

2.11.3 Convergence Behavior of DSCG Solver

The convergence behavior of DSCG, particularly for flow in heterogeneous porous media, was analyzed by Dougherty (1990), Bagtzoglou and others (1992c,d), and Ababou and others (1992a). Below, an expansion on some results and observations from the work of Ababou and others with emphasis on those aspects independent of hardware can be found. Certain practical consequences of the convergence behavior of DSCG will be discussed. For hardware-dependent aspects, such as vectorization, parallelization, and actual timings, the reader is referred to the above discussion and references.

For a wide class of iterative solvers that includes CG and DSCG, the number of iterations required to decrease the error by, for instance, six orders of magnitude is known to be approximately proportional to the square root of the condition number of the coefficient matrix. In the case at hand, the condition number is typically $O(n^2)$, where n represents the unidirectional size of the grid along its largest side (Ababou, 1988). For each iteration, the computational work or number of operations is proportional to N, the multidimensional number of nodes. Multiplying by the estimated number of iterations yields a total work on the order $O(N^p)$, with exponent $p = 4/3$ for a 3D-cubic grid [$p = 3/2$ for a two-dimensional (2D) square grid; $p = 2$ for a one-dimensional (1D) grid].

These order of magnitude estimates give indications on the relation between computer time and problem size. However, they have several shortcomings: (i) the convergence rate estimate does not indicate the influence of conductivity heterogeneity and spatial structure; (ii) it is only a worst case estimate obtained from an approximate error upper bound; (iii) this worst case estimate must break down as the number of iterations approaches the number of equations (N), since the CG method gives the exact solution in no more than N iterations (within machine precision); and (iv) the assumption that computational work per iteration is proportional to grid size (N) does not take into account possible nonproportional speedups due to vector and parallel processing.

In the special case of constant conductivity, DS has no effect and the DSCG solver is equivalent to the conventional CG solver. Let the number of iterations (I) be defined as that required to decrease the L_∞-norm of error by six orders of magnitude. Tests on grids ranging from $(8)^3$ up to $(128)^3$ nodes (over two million nodes) showed approximately linear increase of I(n), with respect to unidirectional grid size (n). This behavior, depicted in Figure 2-4 (Bagtzoglou et al., 1992c), is in agreement with the simplified theory presented above. The difference in the slope observed between BIGFLOW and CMVSFS is attributed to the use of a relative error versus an absolute error as the algorithm stopping criterion. What is important to keep in mind, however, is that both codes show an almost exact linear behavior, characteristic of their independent agreement with the theory.

The convergence behavior of the DSCG solver can be, however, more complicated (as indicated in the theoretical discussion above). For illustration, Figure 2-5 depicts several curves of the L_∞-norm of error versus iteration count for test problems with different degrees of heterogeneity, grid sizes, and

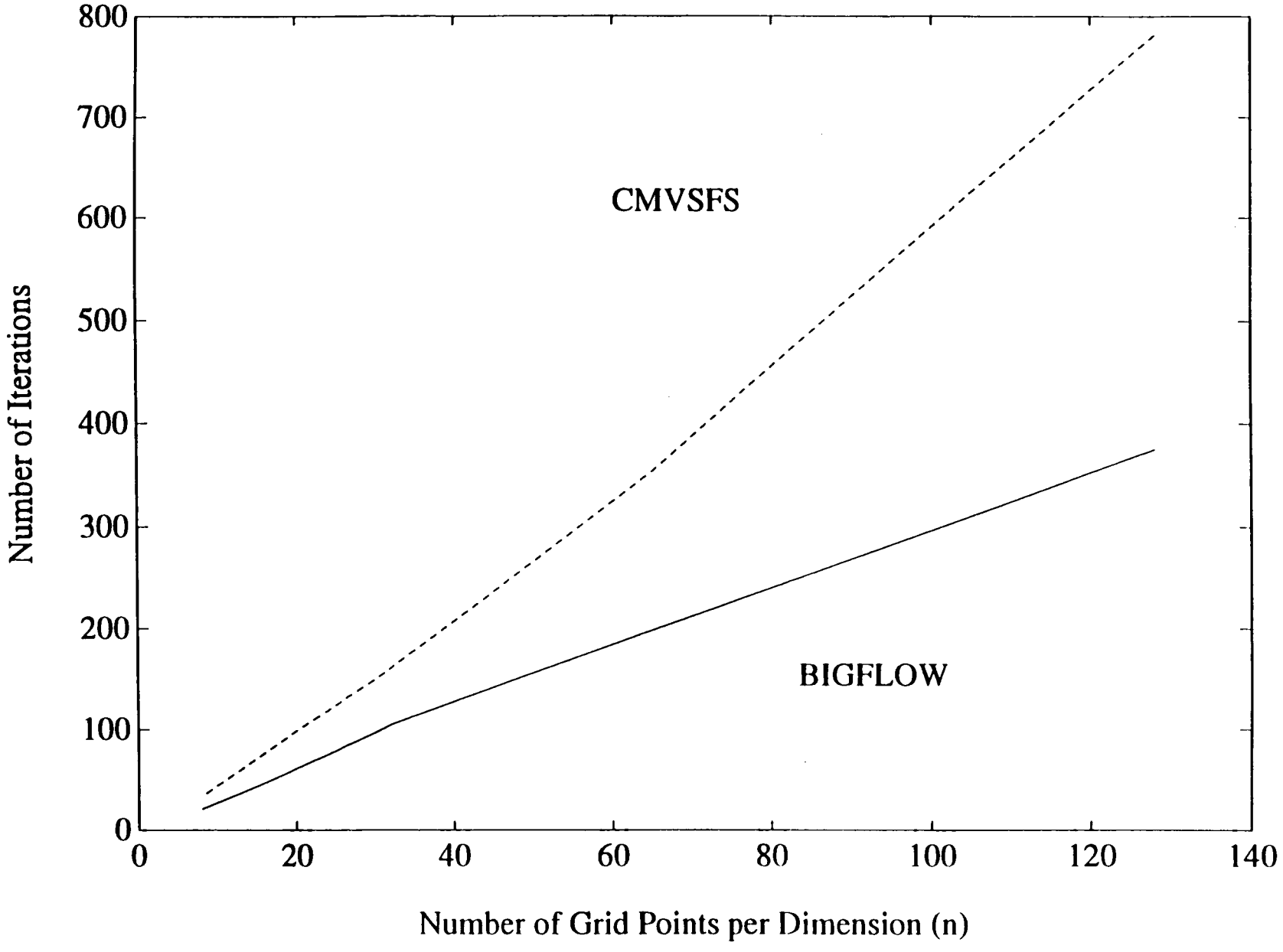


Figure 2-4. Number of DSCG iterations as a function of unidirectional grid size for BIGFLOW and CMVSFS

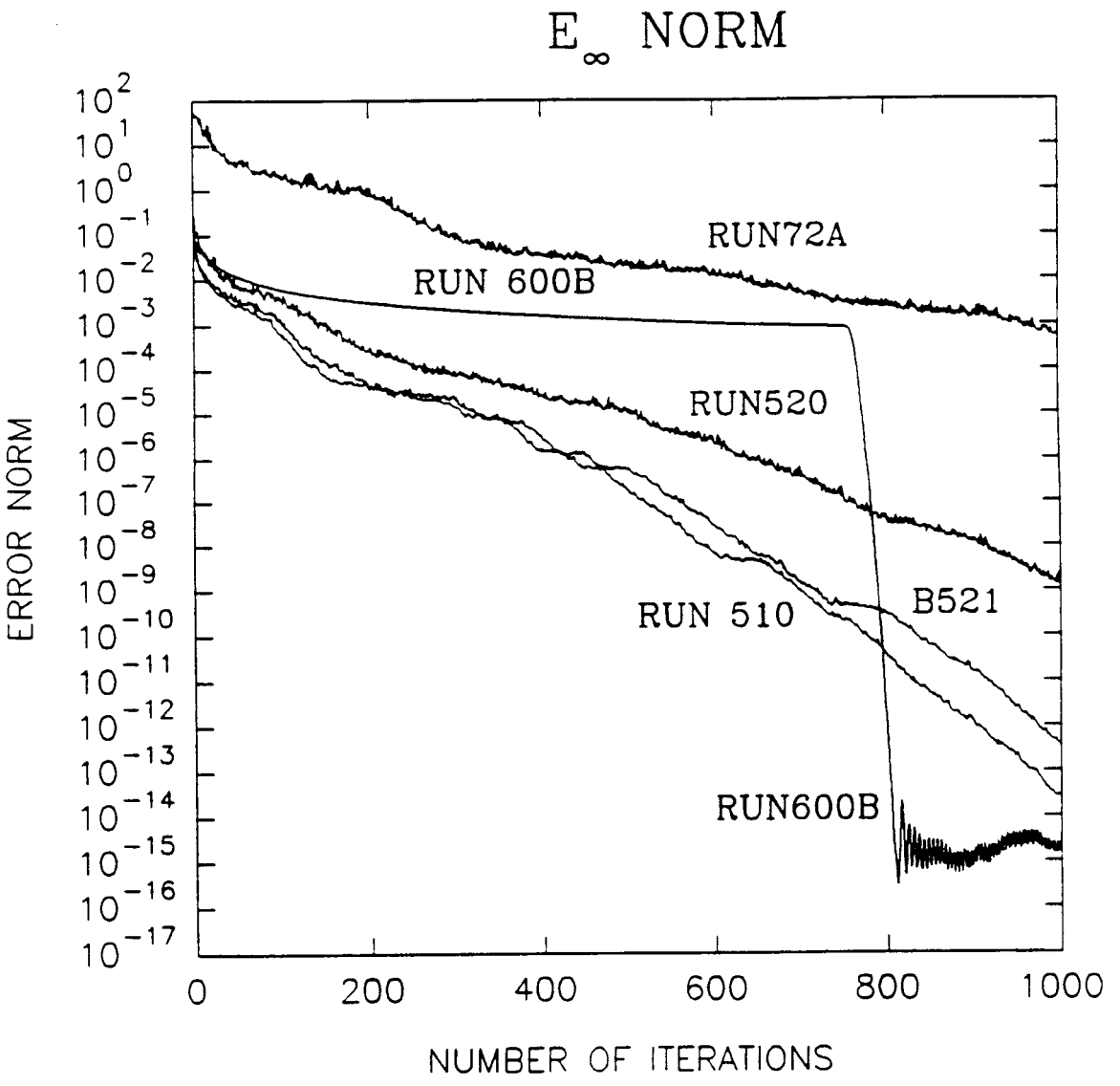


Figure 2-5. Convergence of DSCG iterations (L_∞-norm of error versus iteration count) for the test problems described in Table 2-1

log-conductivity structures. These test problems are summarily described in Table 2-1. Grid sizes range from a few thousand nodes up to 7.6 million nodes. The degree of heterogeneity is represented by σ, the standard deviation of log-conductivity (lnK).

In Figure 2-5, note the singular behavior exhibited by curve number 600B. Initially, convergence is slow as expected, due to the large unidirectional size of the grid (n = 1001). However, after over 700 iterations, the error drops quickly to machine precision and cannot decrease further. This behavior is in agreement with the fact that the CG method always yields the exact solution, within machine precision, after N iterations at most (N is not very large for this problem). On the other hand, comparing curves labeled number 510 (σ = 1) and number 520 ($\sigma = \sqrt{3}$) indicates the influence of degree of heterogeneity (slower convergence). Comparing curves number 520 (Gaussian distribution) and number B521 (binary distribution) demonstrates the equally important influence of spatial structure. And comparing curves number 520 (1 million nodes) and number 72A (7.6 million nodes) shows the influence of grid size (slower convergence).

The convergence behavior reported in Figure 2-5 complements the theoretical convergence rate estimates given earlier. Taken together, these results could be used to assess the typical number of iterations required for analogous saturated flow problems. In the case of constant or mildly heterogeneous conductivity, the theoretical estimate given earlier indicates how CPU time grows with grid size. For highly variable coefficients, this estimate must be corrected based on empirical tests, such as those shown in Figure 2-5 and Table 2-1. Note, for instance, that after 1000 iterations, the error decreases by 12 orders of magnitude for moderate variability (Test #510), compared to "only" 9 orders of magnitude for larger variability (Test #520).

Finally, the empirical tests also demonstrate that essentially exact solutions can be obtained in a finite number of iterations (Test #600B). With the CG method, it is known theoretically that the number of iterations can never exceed the total number of equations (or grid points). Empirical tests indicate that, for grids with very large aspect ratio, the number of iterations to achieve essentially exact solution is on the order of the number of nodes along the largest dimension of the grid. Indeed, Test #600B, with a quasi "one-dimensional" grid of size 1001 $\times$ 5 $\times$ 5, required roughly 1000 iterations (more precisely 800 iterations) to achieve essentially exact solution. These indications may be useful to BIGFLOW users who wish to tailor their grid size and geometry based in part on computational criteria.

2.11.4 Computational Efficiency of DSCG Solver

Following the approach developed by Ababou and others (1985, 1989), Meyer and others (1989) analyzed performance of DSCG and other preconditioned solvers for single realizations of stochastic groundwater flow on grids on the order of one million equations. The grid sizes and statistical properties were similar or identical to those previously used in million node simulations with the nonvectorized SIP solver (Ababou, 1988). Comparing the SIP timings by Ababou (1988), and using certain equivalence rules to convert computer times from different machines, Meyer and others (1989) concluded in favor of DSCG over SIP. The advantage of DSCG over SIP can be explained in part by the more efficient vectorization of DSCG on the use of the vector-parallel Alliant machine.

The efficiency of the DSCG solver was also demonstrated more recently in large-scale numerical experiments by Ababou and others (1992a). In the latter work, heterogeneous conductivity systems on the order of ten million equations were accurately solved in times on the order of minutes, using Cray-2

Table 2-1. Summary of test problems for the DSCG solver of BIGFLOW

Test Number	ln(K) Distribution	Grid Size $N = n_1 \times n_2 \times n_3$
600B	Constant $\sigma=0$	$N = 1001\times5\times5$
510	Gaussian isotropic $\sigma=1$	$N = 101\times101\times101$
520	Gaussian isotropic $\sigma=\sqrt{3}$	$N = 101\times101\times101$
521	Binary isotropic $\sigma=\sqrt{3}$	$N = 101\times101\times101$
72A	Gaussian anisotropic $\sigma=\sqrt{3}$	$N = 178\times120\times357$

and Cray-Y/MP8 computers. Furthermore, it was found that SIP was roughly 20 times less efficient than DSCG when executing on the same Cray-2 machine in vector mode. Finally, efficient vectorization and coarse-grained parallelization, typically 85 - 90 percent of the entire BIGFLOW code, were achieved on a Cray-Y/MP8 system with eight concurrent processors. Parallelization was performed by using Cray's autotasking utility. Both vectorization and parallelization were achieved without special directives or any other modification of the code (Ababou et al., 1992a). The results are summarized below.

The performance of the DSCG-based code, expressed in CPU seconds, depends on grid size and number of iterations and on machine-dependent additive and multiplicative factors. Following Ababou (1988), timings can be expressed approximately in the form

$$T(I,N) = (aI + b)\, N \tag{2-37}$$

where T is the total CPU time (seconds), a represents specific iterative work (seconds/iteration/million nodes), and b represents work spent outside the iterative solution process or overhead (seconds/million nodes). As before, I is the number of iterations, and N is the number of nodes in multi-dimensional space. Note that I may be a pre-selected number of iterations, or alternatively, the number of iterations to decrease the error by a certain amount (say, six orders of magnitude). In view of the theoretical estimate given earlier and confirmed experimentally, the number of iterations is proportional to $N^{1/3}$ for cubic grids.

For the DSCG-based code running serially on Cray-2 and Cray-YMP machines, it was found empirically:

- $a_{(\text{Serial Cray-2})} = 0.48$ seconds per iteration per million nodes; and
- $a_{(\text{Serial Cray-Y/MP})} \approx 0.20$ seconds per iteration per million nodes.

These constants were obtained from timings of several large test problems with randomly heterogeneous conductivities, with most, but not all, involving cubic grids. The Cray dependency analyzer and optimizer (fpp) was used on both machines; the aggressive optimization option was used on the Cray-Y/MP. Note that the Cray-Y/MP machine is faster than Cray-2 by a factor around 2.5 for these types of problems (in serial mode).

The serial Cray-2 timings were analyzed in detail using the flowtrace utility. It was found that the fpp dependency analyzer decreased a by just a few percent. All inner loops vectorized with or without fpp. The overhead constant b was found to be sensitive to Inputs/Outputs (I/Os); with unformatted I/Os, this constant was found to be 28 seconds per million nodes, compared to 116 seconds per million nodes with formatted I/O's.

Coarse-grained parallelization was studied by allowing the BIGFLOW code to run concurrently on k processors of the Cray-Y/MP8 in dedicated mode ($1 \leq k \leq 8$). Again, the DSCG solver was used for solving random conductivity problems involving one to several million grid points. The BIGFLOW source code was not modified for multiprocessing. Instead, the Cray autotasking software performed the necessary code modifications and enhancements (using a compiler option to inline the CG solver module). Estimates of speedups and of parallelizable fraction of code were obtained by comparing cumulated CPU times to wall clock times, and by applying Amdahl's law.

Let k denote the number of processors, T_k the parallel CPU time or wall clock time for k concurrent processors running in dedicated mode, and T_1 the serial CPU time for a single processor. Define f as the fraction of parallelizable code, measured in serial CPU time units. Decomposing T_1 into parallelizable and nonparallelizable parts yields

$$T_1 = fT_1 + (1-f)T_1 \tag{2-38}$$

The serial-to-parallel speedup ratio is given by $r(k) = T_1/T_k$, and satisfies $r \geq 1$. Substituting T_k yields the following relation, known as Amdahl's law

$$r(k) = \frac{k}{k(1-f) + f} \tag{2-39}$$

If the parallelizable fraction f is known, Amdahl's law can be used to obtain speedups. Amdahl's law can also be inverted to obtain the fraction f from observed speedups. Defining

$$\rho(k) = r/k \tag{2-40}$$

as the average speedup per active processor, it is interesting to note that f can be expressed in the simple form

$$f = \rho(k-1)/\rho(k) \tag{2-41}$$

In other words, Eq. (2-41) simply means that the parallelizable fraction of code is given by the ratio of "per processor speedup" for k-1 and k processors, respectively.

Figure 2-6 depicts two speedup curves r(k) obtained for a 1 million node test problem (lower curve) and for a 7.6 million node test problem (upper curve). The test problems were described earlier

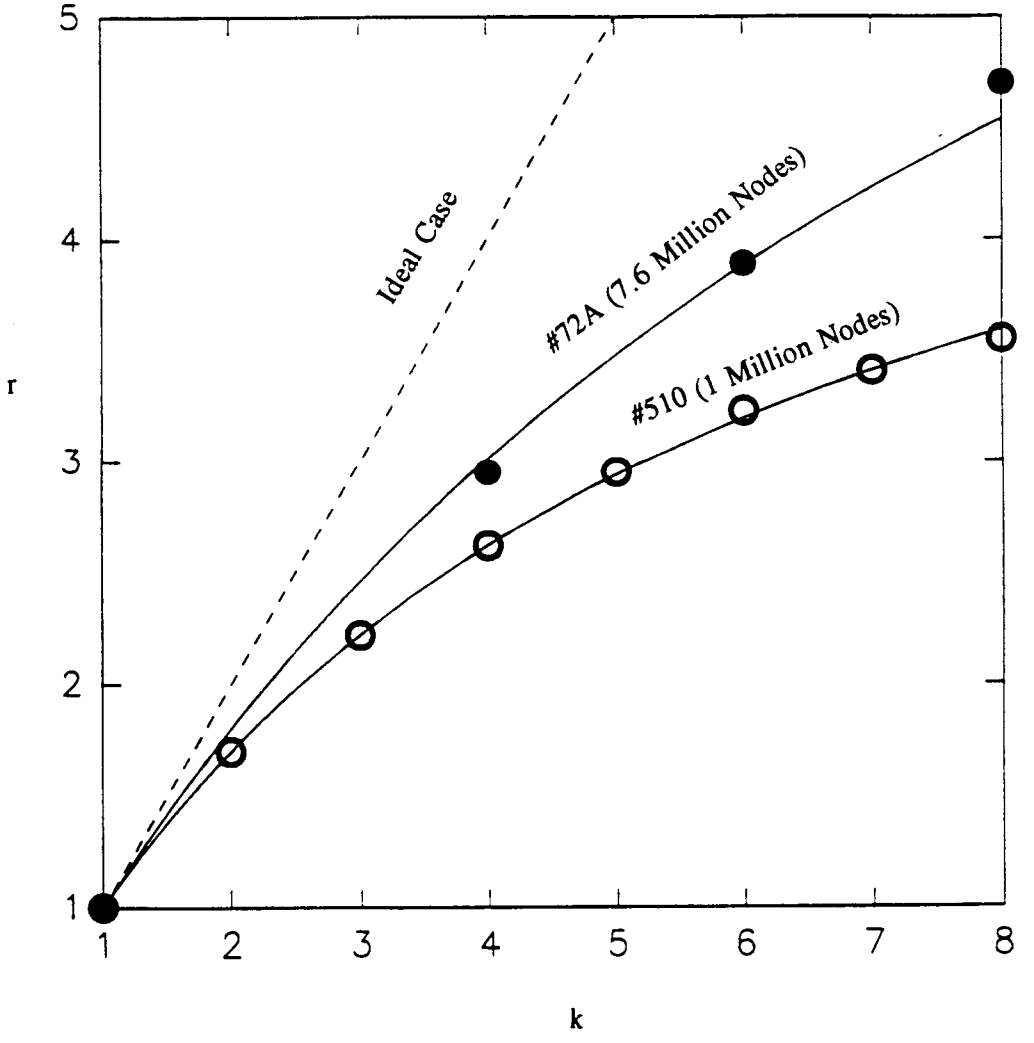


Figure 2-6. Speedup curves: Parallel/serial speedup ratio (r) versus number of Cray-Y/MP8 processors running concurrently (k)

in Table 2-1 (test numbers 510 and 72A, respectively). Figure 2-5 shows both actual speedups (circles), and analytical curves r(k) (solid lines) from Amdahl's law. These curves were obtained after evaluation of the parallelizable fraction "f". The dashed straight line represents the ideal case f = 100 percent, corresponding to a fully parallelizable code.

Under the Cray autotasking utility, it was found that f = 82.5 percent for the one million node problem (number 510), and f = 89.1 percent for the 7.6 million node problem (number 72A). The corresponding speedup ratios for eight processors are 3.59 and 4.53, respectively. The sensitivity of speedup curve to grid size and grid geometry may be due to trade-offs between vector processing and multiprocessing. The largest problem, with 7.6 million grid points, executed at about 750 MFLOPS (wall clock). Having recently identified certain ambiguities in the DSCG solver and the norm calculation modules, we expect to achieve faster rates, possibly well over 1 GFLOPS, by simple modifications of these modules in the future (the current version of BIGFLOW does not incorporate such modifications).

Applying a speedup ratio of approximately 3.5 - 4.5 to the serial Cray-Y/MP timings given earlier, we have

$$a_{(\text{Parallel Cray-Y/MP8})} \approx 0.04\text{-}0.06 \text{ seconds per iteration per million nodes.}$$

This performance is on the same order as that achieved by other DSCG-based codes running on the Connection Machine CM-2 (Bagtzoglou et al., 1992c,d; and Dougherty, 1991). Thus, for a homogeneous $(128)^3$ problem, a performance of 0.070 sec/iter/million nodes was achieved on 16 K processors (Bagtzoglou et al., 1992c,d), and 0.043 sec/iter/million nodes on 32 K processors (Dougherty, 1991). Recent hardware and software optimizations made the code CMVSFS much more efficient. It can now achieve performances of 0.033 sec/iter/million nodes on a 16 K CM-200 machine. However, to keep these comparisons in perspective, it should be noted that the BIGFLOW code includes a relatively large number of features and options, and is apparently more complex than the CM-2 or CM-200 codes used for these comparisons. Also, other CM-2 timings reported in the above-cited references indicated in fact much slower computations for grid sizes not exactly equal to powers of two, a problem not encountered with BIGFLOW on the Cray-Y/MP8 machine.

2.12 THE STRONGLY IMPLICIT PROCEDURE (SIP SOLVER)

The preconditioner used in SIP is a nonsymmetric triangular factorization that has the advantage of being potentially more accurate, that is, higher order in Δx, than the IC factorization (Gustafsson, 1978). Unfortunately, the SIP preconditioner cannot be used in conjunction with CGs, since the CG iterations require a symmetric system, thus a symmetric preconditioner. Below, the SIP solver for the 3D, seven-diagonal finite difference systems at hand is developed. First, a brief review of the origin and properties of this solver is given.

The SIP was originally devised by Stone (1968) for solving the 2D heat equation, and was then extended to the 3D case by Weinstein and others (1969) in the context of oil recovery simulations. SIP has also been used for simulating 2D and 3D groundwater flow problems of moderate size (Trescott, 1975; McDonald and Harbaugh, 1984; Kuiper, 1981 and 1987). Partially saturated flow processes have been previously simulated numerically using a SIP-based method (Cooley, 1983). Here, we will implement SIP to solve very large heterogeneous flow systems in both saturated and unsaturated media,

following earlier work reported in Ababou and others (1985, 1989), Ababou (1988), and Ababou and Gelhar (1988).

The SIP solver is specifically tailored for sparse centered finite difference approximations of elliptic or parabolic equations. It is based on an approximate, nonsymmetric, triangular factorization preconditioner. A Picard-type iteration scheme is used to smooth out the residual error due to the approximately factored matrix. In the case at hand, we apply SIP to both linearized unsaturated flow and/or linear saturated flow. As before, the linear or linearized system to be solved can be expressed in the generic form

$$A \cdot y = b \tag{2-42}$$

where y represents a vector of pressures (or incremental pressures in the nonlinear case); b is a vector that contains boundary terms plus residual terms in the nonlinear and/or transient case; and A represents the conductivity matrix, with additional diagonal mass terms in the transient case.

Briefly, the preconditioned SIP iteration works as follows. Let the unknown lower and upper triangular factors be denoted L and U. The iterative solution scheme is based on splitting in the form $A = LU+E$, where E is an error matrix hopefully close to zero in some norm. This leads quite naturally to

$$L \cdot U \cdot (y_{m+1} - y_m) = \omega (b - A \cdot y_m) \tag{2-43}$$

where m is the iteration counter, and ω is a relaxation parameter. Note that the solution is obtained in terms of the incremental vector $(y_{m+1}-y_m)$, similar to the strategy adopted for the nonlinear iterations (modified Picard scheme). Again, this modified form is known to be more stable with respect to round-off errors.

It is easily seen that Eq. (2-43) is a consistent iteration scheme. If the iterations converge, the limit $m \rightarrow \infty$ yields the exact solution, y, of Eq. (2-43). However, the key problem is to choose the L and U matrices in a way that ensures fast convergence and fast solution of the preconditioned system. In SIP, the LU factorization is nonsymmetric ($L \neq U$), and such that L and U have the same sparsity structure as A. Consequently, Eq. (2-43) only involves sparse triangular systems of size N. These are solved recursively by forward (L) and backward (U) substitutions, requiring only O(N) operations per iteration.

Given the previously described structure of LU, it is easily seen that the system $LU=A$ is overdetermined; thus A cannot be factored exactly in the form LU. This is clearly indicated by the extra diagonals in the LU product matrix. On the other hand, given an error matrix E, the system $LU-A=E$ is underdetermined. We conclude that many choices of L, U, and E are possible. The particular factorization developed by Weinstein and others (1969) approximates the original partial differential equation by a nonsymmetric 13-point finite difference molecule, rather than the seven-point centered molecule. It is a generalization of the original algorithm developed by Stone (1968) for 2D problems, where the original system is five-diagonal instead of seven-diagonal.

For the 2D Laplace equation, it was shown (Stone, 1968) that LU approximates A with second order accuracy in terms of mesh size; the error matrix actually includes cross-terms like $O(\Delta x.\Delta y)$. For the more general 3D equations at hand, this finding may be loosely expressed as

$$E = LU - A = \sum_{i,j} O(\Delta x_i \, \Delta x_j) \qquad (i = 1,2,3 \; ; \; j = 1,2,3) \qquad (2\text{-}44)$$

In comparison, the IC factorization used in ICCG is only first order accurate (Gustafsson, 1978); the accuracy of the factorization is, however, only one among other measures of efficiency of the SIP and ICCG solvers.

To compute the SIP triangular factors L and U requires a nonlinear recursive algorithm that involves O(N) operations. This factorization algorithm, not detailed here, is analogous to the well-known Thomas factorization of tridiagonal systems, although more complex. In particular, the L and U factors are allowed to depend on a cyclic parameter $\gamma_m \in [0,1]$ that greatly influences convergence. The cyclic γ_m-sequence advocated by Stone (1968) was inspired by results from alternate direction methods and from a Fourier analysis of amplification rates (Stone, 1968). A variant of SIP with alternate node ordering (standard/reverse) has also been advocated (Stone, 1968; and Weinstein et al., 1969). The nonalternate version based on the standard cartesian node numbering scheme is the one that will be implemented here. This and other details have been described in Ababou (1988). An error concerning the cyclic γ_m-sequence in Stone's paper was corrected.

The convergence theory of SIP is far from complete at the time of this writing. Unlike Alternate Direction Implicit (ADI) methods, the optimal γ_m-sequence is not known, even for the simple Laplace equation. This uncertainty has prompted us to introduce an additional relaxation parameter, ω. A recent work on the convergence properties of SIP (Chen, 1988) seems to validate this relaxation strategy. Chen essentially proved the following result:

> "There exists a value of $\gamma = \gamma_0$ in [0,1], and a value $\omega = \omega_0$ in [0,2], such that if the γ-parameter satisfies $0 \leq \gamma \leq \gamma_0$, and if the relaxation parameter satisfies $0 < \omega < \omega_0$, then SIP converges."

In practice of course, the conditions assumed in this proof may not be met. For instance, γ is actually cyclic, not constant. Nonetheless, our experience indicates that choosing certain values of ω with $\omega \leq 1$ can force convergence of SIP in cases involving very large weakly diagonal dominant and heterogeneous coefficient matrices.

It should be kept in mind that the algebraic and convergence properties of the system depend on: (i) flow regime; and (ii) space-time discretization. It is known that the coefficient matrix A is seven-diagonal, symmetric positive-definite, and diagonally dominant. But more specifically, A is only weakly diagonal dominant in the steady state case, and it can even become nondominant (indefinite) if there is no Dirichlet boundary condition on any boundary node. Strict diagonal dominance always holds in the transient case, provided that the storage term $(C/\Delta t)$ remains strictly positive at all times and all locations, as will occur for low rate infiltration in a moderately dry soil, or for saturated groundwater flow with storativity effects. Diagonal dominance increases as time step decreases. As diagonal dominance is enhanced, the condition number of the matrix decreases and the SIP solver yields more accurate solutions in fewer iterations. The latter observations apply to PCG solvers as well.

2.13 NONLINEAR SOLVERS AND NESTED ITERATIONS

The iterative matrix solvers DSCG and SIP are applicable to unsaturated flow as well as saturated flow. The iterative linearization procedure was developed earlier independently from any particular matrix solver (see modified Picard algorithm). However, a strategy must be devised for coupling or nesting the iterative matrix solver with the iterative linearization procedure.

Briefly, the overall nonlinear solution strategy is as follows. For each time step (n), the linearization procedure (Picard) operates as an outer iteration loop (k), while the matrix solver (DSCG or SIP) is the inner iteration loop (m). The inner matrix iterations yield the solution of the linearized system at each outer iteration; the outer iterations yield the solution of the nonlinear system; and this is repeated for each time step. The fully nested iteration scheme can be expressed by inserting the DSCG iterations or the SIP iterations, with $y = \delta p$, in the nonlinear Picard scheme [Eq. (2-34)].

In the case of SIP — which we use for illustration — this yields

- for n=0,1,2,... (time steps),
- for k=0,1,2,... (outer linearization loop), and
- for m=0,1,2,... (inner matrix solution loop).

$$(LU)^{n+1,k}\left(\delta p_{m+1}^{n+1,k+1} - \delta p_{m}^{n+1,k+1}\right) = \omega\left(b^{n+1,k} - A^{n+1,k}\delta p_{m}^{n+1,k}\right) \qquad (2\text{-}45)$$

where the incremental pressure δp should go to zero as the correct nonlinear solution is being approached. Equation (2-45) defines the nonlinear-SIP solver. A similar algorithm, with same nested structure, is implemented for the nonlinear-DSCG solver.

In summary, the nonlinear finite difference system is iteratively linearized at each time step using the modified Picard method, and the resulting sequence of matrix systems is solved iteratively using available matrix solver modules (DSCG, SIP). Note that SIP itself uses a Picard-type iteration to smooth out errors due to its approximately factorized preconditioner. Therefore, with nonlinear SIP, we have in effect two Picard-type schemes being activated, one in the outer loop (k) for linearization, and another one in the inner loop (m) for solution of the preconditioned matrix.

Finally, a few practical points should be emphasized. In the case of nonlinear-SIP, notice that Eq. (2-45) is being solved for a double increment of pressure, (δp_{m+1}-δp_m). When this quantity becomes small in some sense, the inner iterations (m=0,1,2...) should be stopped. Likewise, the outer iterations (k=0,1,2,...) should be stopped when the pressure increment $\delta p = p^{k+1}$-p^k becomes small in some sense. In the case of initially dry media, the stopping criteria should be based on comparing the maximum absolute value of pressure increments at all grid points (L_∞-norm) to a preset tolerance. In most test cases, the tolerance was roughly 0.1 cm of pressure head for the outer loop, and 0.01 cm for the inner loop. These numbers were relatively small compared to the maximum pressure head variation over space and time, typically on the order of 100 cm for our unsaturated test problems (see applications in Section 3). The same remarks are essentially applicable to the nonlinear-DSCG solver as well.

In closing, let us point out that alternative nonlinear-SIP procedures have been previously developed, notably by Trescott and Larson (1977), and Kuiper (1981,1987), for the mildly nonlinear equation of unconfined groundwater flow. Also, a relatively complex SIP-Newton method was developed by Cooley (1983) for solving partially saturated flow with seepage faces. In all of these numerical studies, the sizes of the test problems were small, on the order of 1000 nodes or less. The paper by Kuiper (1987) compares the performance of a number of solution methods based on SIP and ICCG matrix solvers, in conjunction with various Picard and Newton strategies for dealing with the mild nonlinearity due to the variable transmissity of unconfined groundwater flow.

It is interesting to note that some of Kuiper's tests involve in particular a nonlinear Picard solver limited to 1-5 inner iterations. In comparison, the solution strategy recommended for the BIGFLOW code is somewhat more flexible: keep the number of iterations low by tightly controlling the time step size, while enforcing the iteration stopping criteria based on the L_∞-norm of pressure increments. The dynamic time-stepping algorithm, through which time-step size can be controlled, is described in Section 2.14.

2.14 TIME STEPPING STRATEGY

Given the implicit time integration scheme being used, there is no obvious stability-type limitation on time step size. In practice however, nonlinear instability and/or poor accuracy can result from large time steps. In addition, time step size also influences the condition of the algebraic system, as discussed earlier. Therefore, the choice of time step size is an important feature of the overall solution strategy for transient problems; we will discuss here mainly the case of transient unsaturated flow.

In the transient test problems, unless stated otherwise, a variable time step size related to the variation of pressure with time is used in the following fashion

$$\Delta t_n = \min \left\{ \rho \ \Delta t_{n-1}, \frac{\| p^{n+1} - p^n \|_{\max}}{|P_0 - P^0|} \ \Delta t_1 \right\} \qquad (2\text{-}46)$$

where ρ is an empirical growth factor ($\rho \approx$ 1.05-1.25), P^0 is a representative value of initial pressure, and P_0 is a typical value of pressure at the boundaries (e.g., solution of $K(P_0) = q_0$ if the boundary condition is a fixed flux $q = q_0$). The first time step Δt_1 was calculated independently so as to be fairly small (e.g., Δt_1 proportional to squared mesh size, and inversely proportional to the maximum value of soil moisture diffusivity calculated from known initial and boundary conditions).

Recall from the earlier discussion that decreasing the time step enhances diagonal dominance through the storage term ($C/\Delta t$). Also, given the strongly nonlinear character of problems such as infiltration in dry soil, the time steps cannot be taken too large if the nested iterations are to converge. By applying Eq. (2-46) one is in effect tightly limiting the magnitude of the time step. This strategy leads to improved matrix condition and faster convergence of the nonlinear-SIP or -DSCG solver at each time step.

More precisely, it was found that decreasing the time step within certain bounds did not necessarily increase the total computational work. The reason is that, with smaller time step, the larger number of time steps was almost exactly balanced by the smaller number of iterations per time step. In the case of the nonlinear-SIP solver, with the stopping criteria described earlier, only a few (typically no

more than five) inner iterations of the SIP solver were generally sufficient to solve infiltration-type problems. This suggests that, with the proper choice of time step, the approximate factorization preconditioner of SIP can be considered as a relatively accurate "noniterative solver" (Ababou, 1988).

To put this in perspective, note that the computational work for factorization, and for each inner iteration, is proportional to the total number of nodes N. Therefore, if only a few iterations are needed, that is, O(1) iterations, the computational work required to solve the matrix system is effectively proportional to N, that is, O(N). Naturally, the total work is also proportional to the number of outer iterations per time step (nonlinear Picard loop), and to the total number of time steps (which depends on time scale of simulation as well as time step size).

3 VERIFICATION AND BENCHMARK TESTING OF BIGFLOW

3.1 INTRODUCTION

The principal objective of this section is to evaluate and demonstrate the general capabilities of BIGFLOW. Six test problems with varying degrees of complexity are presented. Two types of computational testing were conducted: (i) verification; and (ii) benchmarking.

Spatially distributed models such as BIGFLOW can be tested in more or less specialized fashion. It is convenient to distinguish testing procedures aimed at verifying the consistency of well defined components of the model, and groundtruth experiments which aim at an overall assessment of the model under real field conditions (Ababou et al., 1992b).

Consistency tests purposely limit the scope of testing in order to focus on the reliability of particular solutions for precisely known model inputs. Groundtruth tests, on the other hand, attempt to assess the validity of the postulated model under conditions that are not fully controlled (e.g., due to unknown material properties), and that may lie outside the accepted range of validity of specific model postulates (e.g., inadequacy of postulated constitutive laws). Numerical tests aim at checking the consistency of the numerical implementation, without questioning basic governing equations, postulated constitutive relations, etc. The consistency of the numerical model can be tested in many different ways, which can be classified broadly as follows: (i) internal tests such as mass balance and sensitivity to space-time mesh size; and (ii) comparisons of model's outputs with other analytical, quasi-analytical, or numerical solutions, all obtained independently.

Verification testing was performed by comparing known analytical solutions of appropriate problems with equivalent simulations using BIGFLOW. The primary objective of the verification testing was to check the computational accuracy of the numerical techniques used within the code. Benchmark testing was conducted to verify the agreement of the simulation results using BIGFLOW and another code of similar capability. In this section, we used PORFLOW (Runchal and Sagar, 1992) and the Connection Machine Variably Saturated Flow Simulator (CMVSFS) to perform the benchmark testing. PORFLOW was used during the benchmarking because it was readily available and has been used in the industry over a considerable time for groundwater simulation flow and transport studies. CMVSFS, a code developed for the massively parallel computer CM-200, was used for result benchmarking and computational efficiency comparisons. It should be kept in mind that no verification, in the strict sense, was conducted since no quantitative performance measures were used. The comparisons presented in this section are only qualitative unless otherwise stated.

The test problems presented here are by no means indicative of all the capabilities of BIGFLOW. They are only meant to illustrate the main features of the code. These test problems involve both saturated and unsaturated flow in a geologic medium and were obtained from technical publications. In the following, the test problems are described

TEST-1: **One-dimensional (1D) transient saturated groundwater flow.** The 1D Richards' equation is solved with BIGFLOW for the special case of exponential K(h) and θ(h) relations. Under these conditions the problem is analogous to the heat equation with constant coefficients.

TEST-2: **1D transient vertical infiltration.** In this test problem, unsaturated flow in a vertical column was simulated. Fluid flow through the soil occurs through capillary action and gravity. A quasi-analytic solution for this problem was published by Philip (1957). This test provided a means to determine BIGFLOW's ability to solve the nonlinear Richards' equation for a relatively simple unsaturated flow problem.

TEST-3: **Two-dimensional (2D) infiltration caused by a line source above a shallow water table.** In this test problem, flow from a trickle or subsurface irrigation system consisting of a perforated pipe placed above a shallow water table is simulated. An analytical solution for this problem was published by Warrick and Lomen (1977). This test provided a means to test BIGFLOW's ability to obtain steady state results directly, without time marching.

TEST-4: **2D infiltration caused by a strip source above a shallow water table.** In this test problem, flow from multiple subsurface porous pipes placed adjacent to each other was investigated. No analytical solution was present for this problem, therefore the verification of the BIGFLOW simulation output was limited to benchmark testing with PORFLOW, which is another code with equal capability. This test provided a means to test BIGFLOW's ability to obtain steady state results using a time marching approach.

TEST-5: **2D infiltration in a heterogeneous medium.** This test problem is physically similar to TEST-4 with the exception of the presence of an obstacle (modeled by a zone with a lower saturated conductivity). No analytical solution was present for this problem, therefore the verification of the BIGFLOW simulation output was limited to benchmark testing with PORFLOW. The objective of placing an obstacle in the path of the water flow was to determine BIGFLOW's ability to simulate realistic field conditions. It further showed the computational efficiency of BIGFLOW in the presence of high conductivity contrasts.

TEST-6: **Flow of water through a hole in a box under pressure.** This test problem makes use of BIGFLOW's ability to simulate boundary conditions of different types, co-existing on the same planar face. It is a mathematical/numerical analog of water withdrawal at fixed pressure, and mimics the case of flow from an artesian spring.

3.2 INTERNAL TESTS (MESH SIZE SENSITIVITY AND MASS BALANCE)

Internal tests are generally based on results of numerical analysis. In the first place, space-time discretization methods must be selected so as to be theoretically consistent with governing equations: discretized equations must converge to governing equations as space-time mesh size goes to zero. Typically, however, numerical analysis does not give sufficient information on the rate of convergence of discrete equations to governing equations. The reader is referred to Ababou (1990), and Ababou and others (1992b), for analyses of stability and truncation errors relevant to the BIGFLOW code.

In the nonlinear case in particular, it was concluded that numerical analysis, while indicating the order of accuracy of the discretization, does not give sufficiently reliable information on the accuracy to be expected for noninfinitesimal space-time mesh size. For these reasons, numerical experimentation is required in order to better evaluate the effects of space-time mesh size. This may be particularly useful in the case of heterogeneous as well as nonlinear material properties, where truncation error estimates become very difficult to evaluate.

Consider for instance, the case of transient strip-source flux infiltration in a 2D, perfectly layered soil with alternating sand/silt layers of equal thickness (Ababou, 1988). Figure 3-1 shows a comparison of pressure profiles on a vertical transect coinciding with the axis of symmetry of the strip-source. The solution is obtained by BIGFLOW with a geometric weighing of mid-nodal conductivities. The layering consists of alternating sand/silt layers of equal thickness. The "fine" mesh corresponds to three grid spacings per layer, and the "coarse" mesh to just one grid spacing per layer. While some detailed fluctuations of the pressure field have been lost in passing from fine to coarse discretization, Figure 3-1 indicates that the coarser discretization does preserve the global features of the wetted region, including the location of the wetting front. However, the distribution of fluxes in the "coarse" simulation needs to be verified as well; it may not be as accurate as that of pressure or moisture.

Similar numerical experiments can be conducted for testing time-step discretization. In such internal tests, numerical solutions obtained with the smallest space-time mesh sizes are considered to be essentially exact in comparison to coarser discretizations. However, in the presence of highly contrasted vertical fault zones the time step is an essential stability criterion (Bagtzoglou et al., 1992b).

In addition to discretization errors, the numerical implementation of spatially distributed flow models entails errors due to approximate solution of nonlinear (though quasilinear) systems, and round-off errors due to limited machine precision for floating-point operations. When using iterative solution schemes, the combined solution errors and round-off errors can be estimated numerically by computing "on-line" the norm of the residual, or incremental solution between consecutive iterations. The magnitude of this error norm, and its rate of convergence toward zero, give approximate indications of the errors incurred in the solution process (see, for instance, Ababou et al., 1989, for iterative solutions of large linear systems).

In the case of steady saturated flow, the only available quantities for mass balance checks are the total discharge rates through 2D sections. When zero flux conditions are used on four lateral boundaries and hydraulic head conditions on the remaining boundaries, the longitudinal discharge rates through fixed head boundaries should be equal, and the transverse discharge rate through any longitudinal section should be zero. These principles have been used to evaluate mass balance errors for large single realizations of stochastic groundwater flow (Ababou, 1988).

Conservation principles such as global mass balance are useful as internal checks of overall accuracy in the following restricted sense. Since the exact solution must satisfy exactly global mass conservation, the accuracy of the numerical solution can be assessed by evaluating the discrepancy between the net discharge rate entering the system and the rate of change of mass (converted to volume) of water present in the system. However, even if global mass balance is found to be satisfied accurately, this does not rule out the possibility of significant local errors which may cancel out on average, for example, spatial oscillations as shown in Bagtzoglou and others (1992a).

If significant mass balance discrepancies are found, they may be due to spatial discretization errors, time discretization errors, nonlinear solution errors, and/or round-off errors. Mass balance calculations should be consistent with the space-time discretization scheme used in the numerical model. Consider the case of transient unsaturated flow based on the mixed form of Richard's equation with a volumetric source term (s), and prescribed boundary conditions

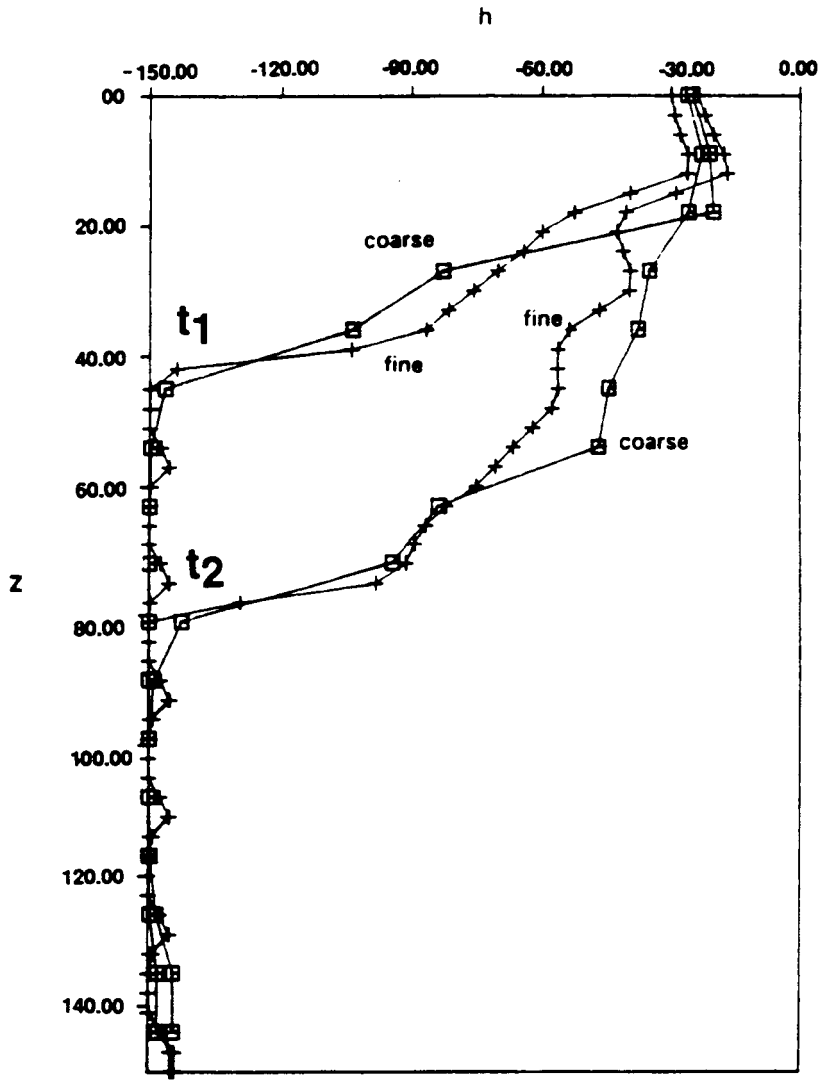


Figure 3-1. Comparison of pressure profiles for the fine mesh (three cells per layer: crosses) and coarse mesh (one cell per layer: square boxes) along vertical transect through the axis of the strip at times t = 0.3 day and t = 1 day

$$\frac{\partial \theta}{\partial t} = \vec{\nabla} \cdot \vec{q} + s \quad \text{in} \quad \Omega \subset \mathbb{R}^3 \tag{3-1}$$

$$\frac{\partial h}{\partial z} = 0 \rightarrow \vec{q} \cdot \vec{n} = K(h_B) \quad \text{on} \quad \bar{\Omega}_o \tag{3-2}$$

$$h = h_B \quad \text{on} \quad \bar{\Omega}_1 \tag{3-3}$$

$$\vec{q} \cdot \vec{n} = q_B \quad \text{on} \quad \bar{\Omega}_2 \tag{3-4}$$

where $\bar{\Omega}_o + \bar{\Omega}_1 + \bar{\Omega}_3 = \bar{\Omega}$ represents the boundary of computational domain Ω, and s is a distributed volumetric single/source density within a subdomain Ω_s of Ω (e.g., due to water uptake in the root zone). Integrating Eq. (3-1) over Ω and applying Green's divergence theorem yields

$$\frac{\partial M}{\partial t} = \iint_{\bar{\Omega}} \vec{q} \cdot \vec{n} \, d\sigma + S \tag{3-5}$$

where M is the total mass (converted to volume of water), and S the total sink-source term (m^3/s)

$$M = \iiint_{\Omega} \theta \, d\tau \tag{3-6}$$

$$S = \iiint_{\Omega_s} S \, d\tau \tag{3-7}$$

To evaluate mass balance ex-post, Eqs. (3-5) to (3-7) must be discretized in both space and time. For consistency, the discretization schemes must be identical to those used for numerical solution. Consider for instance, the case of implicit second-order finite differences. In this case, the total mass M must be evaluated by summing water contents over node-centered cells. The elementary mass of a node-centered cell is

$$m(i_1, i_2, i_3) = \theta[h(i_1, i_2, i_3)] \, \Delta x_1 \, \Delta x_2 \, \Delta x_3 \tag{3-8}$$

and the total mass is

$$m = \sum_{i_1, i_2, i_3} M(i_1, i_2, i_3) \tag{3-9}$$

The boundary fluxes on the right hand side of Eq. (3-5) must be evaluated using the same midnodal scheme as that used in the interior domain. Thus, on the fixed pressure boundary ($\bar{\Omega}_1$), the boundary flux is evaluated by a scheme of the form

$$q_{1/2} = - K_{1/2}\left(\frac{h_1 - h_0}{\Delta x} + g\right) \tag{3-10}$$

where the index 1/2 indicates the mid-nodal location adjacent to the nodal boundary under consideration. The boundary integral in Eq. (3-7) must be evaluated by assuming piecewise constant boundary fluxes over node-centered boundary cells.

Equation (3-5) must now be discretized according to the same implicit scheme that was used for numerical solution. Let $\hat{M}$ represent the discrete estimate of total mass, $\hat{S}$ the discrete estimate of total source term, and $\hat{Q}_B$ the discrete estimate of total boundary flux (net discharge rate at boundaries). Discretizing Eq. (3-5) yields

$$\hat{Q}_M(t) = \frac{\hat{M}(t) - \hat{M}(t-\Delta t)}{\Delta t} - \hat{S}(t) = \hat{Q}_B(t) + E_Q(t) \tag{3-11}$$

This equation expresses the approximate equality of net boundary discharge rate ($\hat{Q}_B$ on the right hand side), and total rate of change of mass inside the domain ($\hat{Q}_M$ on left-hand side). The term E_Q represents the absolute mass balance error in terms of discharge rates (m^3/s).

For transient flow, relative measures of global mass balance errors can be defined as

$$e_Q(t) = \frac{\hat{Q}_M(t) - \hat{Q}_B(t)}{\hat{Q}_B(t)} \tag{3-12}$$

$$e_V(t) = \frac{\int_o^t \{\hat{Q}_M(t') - \hat{Q}_B(t')\}\,dt'}{\int_o^t \hat{Q}_B(t')dt'} \tag{3-13}$$

respectively, in terms of discharge rate (Q), and time-accumulated discharge rate (V). On the other hand, in the case of steady flow without source terms, a convenient measure of relative mass balance error can be devised by distinguishing inlet boundaries and outlet boundaries. Thus, the computed boundary discharge rate is decomposed as

$$\hat{Q}_B = |\hat{Q}_{in}| - |\hat{Q}_{out}| \tag{3-14}$$

and the following relative mass balance error can be defined in terms of steady state boundary discharge rates

$$e_Q = - \frac{|\hat{Q}_{in}| - |\hat{Q}_{out}|}{\frac{1}{2}\left(|\hat{Q}_{in}| + |\hat{Q}_{out}|\right)} \tag{3-15}$$

Similar measures could be constructed for local mass balance at the mesh size. Local mass balance measures, although highly desirable, are not computed by the current version of BIGFLOW. We also submit that conservation principles other than mass balance may be useful, although they have not been traditionally used in the context of porous media flow. One such principle is curl conservation, which expresses the irrotational properties of hydraulic gradient. Let

$$\vec{J} = \vec{\nabla}(h+z) = \frac{\vec{q}}{K} \tag{3-16}$$

since $\vec{J}$ is a gradient, its curl must vanish locally, that is

$$\vec{\nabla} \times \vec{J} = \vec{0} \tag{3-17}$$

Integrating over the computational domain yields

$$\iiint_{\Omega} \vec{\nabla} \times \vec{J} = \vec{0} \tag{3-18}$$

Applying Green-like theorems to this volume integral leads to the following boundary integral identity

$$\iint_{\bar{\Omega}} \vec{J} \times \vec{n} d\sigma = \vec{0} \tag{3-19}$$

The discrepancy between this boundary integral and $\vec{0}$ would give a global measure of the degree to which the numerical solution preserves the irrotational character of the hydraulic gradient vector. Note that the hydraulic gradient is directly related to flux (via Darcy's law).

Figure 3-2 depicts the temporal variation of the mass balance error measure $e_Q(t)$ for test problem TEST-4. It can be seen that at time t = 30 hours the error is 0.7 percent indicating excellent mass balance as steady state conditions are attained. Note that this simulation was conducted by marching in time and using transient pressure head results as initial conditions for subsequent simulations.

3.3 TEST PROBLEMS

3.3.1 1D Transient Saturated Groundwater Flow

Consider the 1D, mixed variable, form of Richards' equation (2-3) with the following nonlinear constitutive relations

$$\begin{aligned} K(h) &= K_s \exp(\alpha h) \\ \theta(h) &= \theta_s \exp(\beta h) \end{aligned} \tag{3-20}$$

In the case $\alpha = \beta$, this equation becomes equivalent to the linear heat equation, with a constant diffusion heat coefficient $D = K_s/(\alpha\theta_s)$, governing the total hydraulic head (P = h + z).

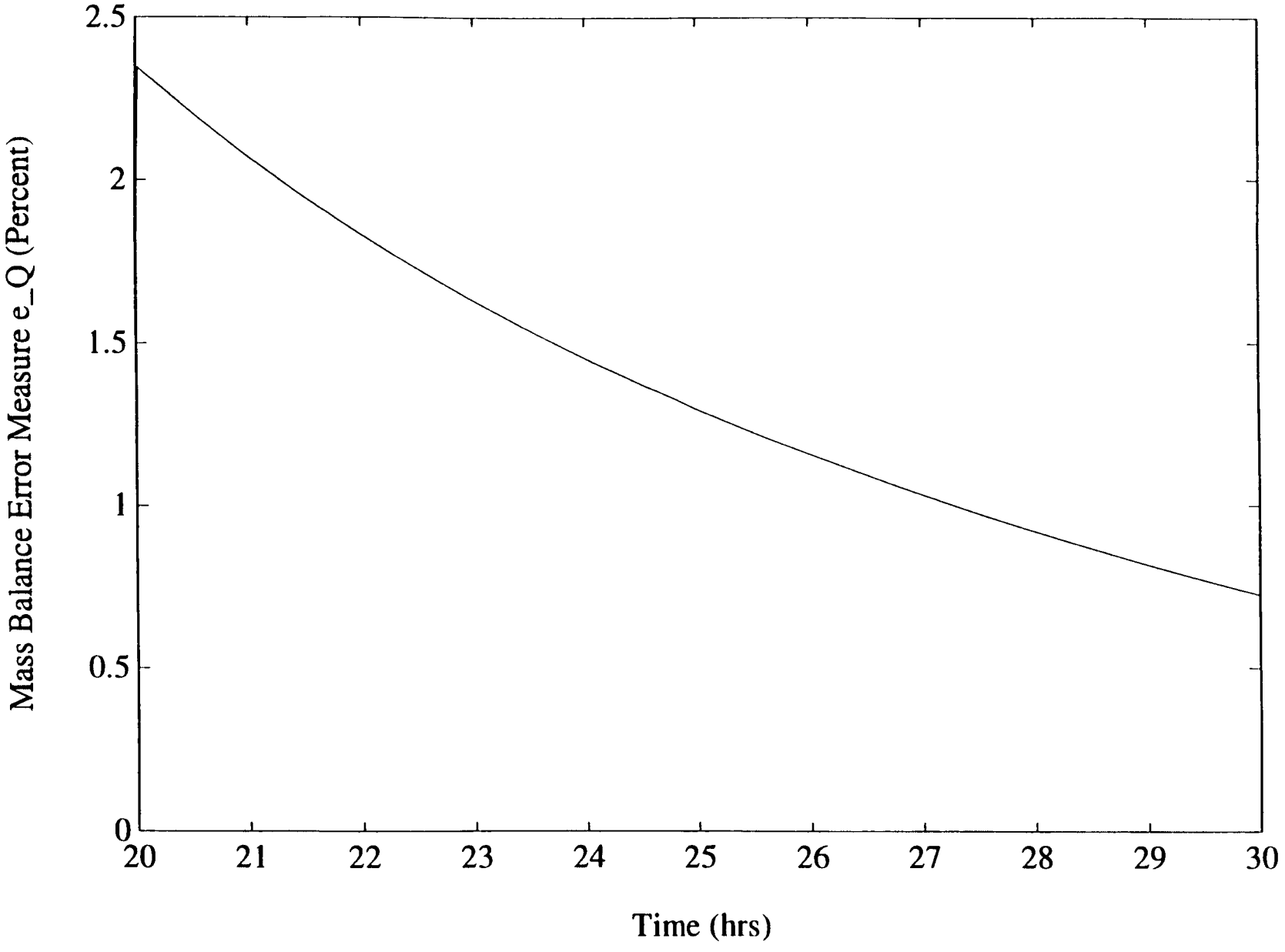


Figure 3-2. Temporal variation of mass balance error e_Q for TEST-4

The analytical solution to the 1D heat equation

$$\begin{aligned} \frac{\partial P}{\partial t} &= \frac{\partial^2 P}{\partial x^2}, \qquad 0 \le x \le 1,\ t \ge 0 \\ P(0,t) &= 1. \\ P(1,t) &= 0. \\ P(x,0) &= 0. \end{aligned} \tag{3-21}$$

is given by the rapidly convergent series

$$\begin{aligned} P(x,t) &= \sum_{n=1}^{\infty} n \cdot (a_n - b_n) \\ a_n &= erf\left(\frac{n + x/2}{\sqrt{t}}\right) + erf\left(\frac{n - x/2}{\sqrt{t}}\right) \\ b_n &= erf\left(\frac{n-1+x/2}{\sqrt{t}}\right) + erf\left(\frac{n+1-x/2}{\sqrt{t}}\right) \end{aligned} \tag{3-22}$$

where erf(x) is the error function

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-s^2)\, ds \tag{3-23}$$

When expressed in terms of dimensional quantities, the solution to the usual groundwater flow equation in a domain of size L, with conductivity K, storativity S, and D = K/S, may be obtained from Eq. (3-22) by substituting (x/L) for x, and (Dt/L^2) for t. This exact solution is compared with the output of the numerical flow simulator. Figure 3-3 shows the transient solution for fixed hydraulic heads at the two end points, as obtained by BIGFLOW using variable time steps and a nonlinear system solver. The analytical and numerical results are identical.

3.3.2 1D Transient Vertical Infiltration

For this test problem the physical setting was a vertical, homogeneous soil column. The initial condition was a uniform pressure head. The pressure head at the upper boundary was held at a value corresponding to saturation. The bottom boundary was held constant at the initial pressure head. Transient infiltration of moisture in the vertical direction resulted from capillary forces and gravity.

The simulation of the movement of the moisture front was performed with BIGFLOW and the pressure head profiles were obtained at three different times: 0.005 days, 0.05 days, and 0.1 days. These profiles were graphically plotted and compared with the semi-analytical solution of Philip (1957). A total of 101 nodes was used that were uniformly spaced over the length of the domain (300 cm).

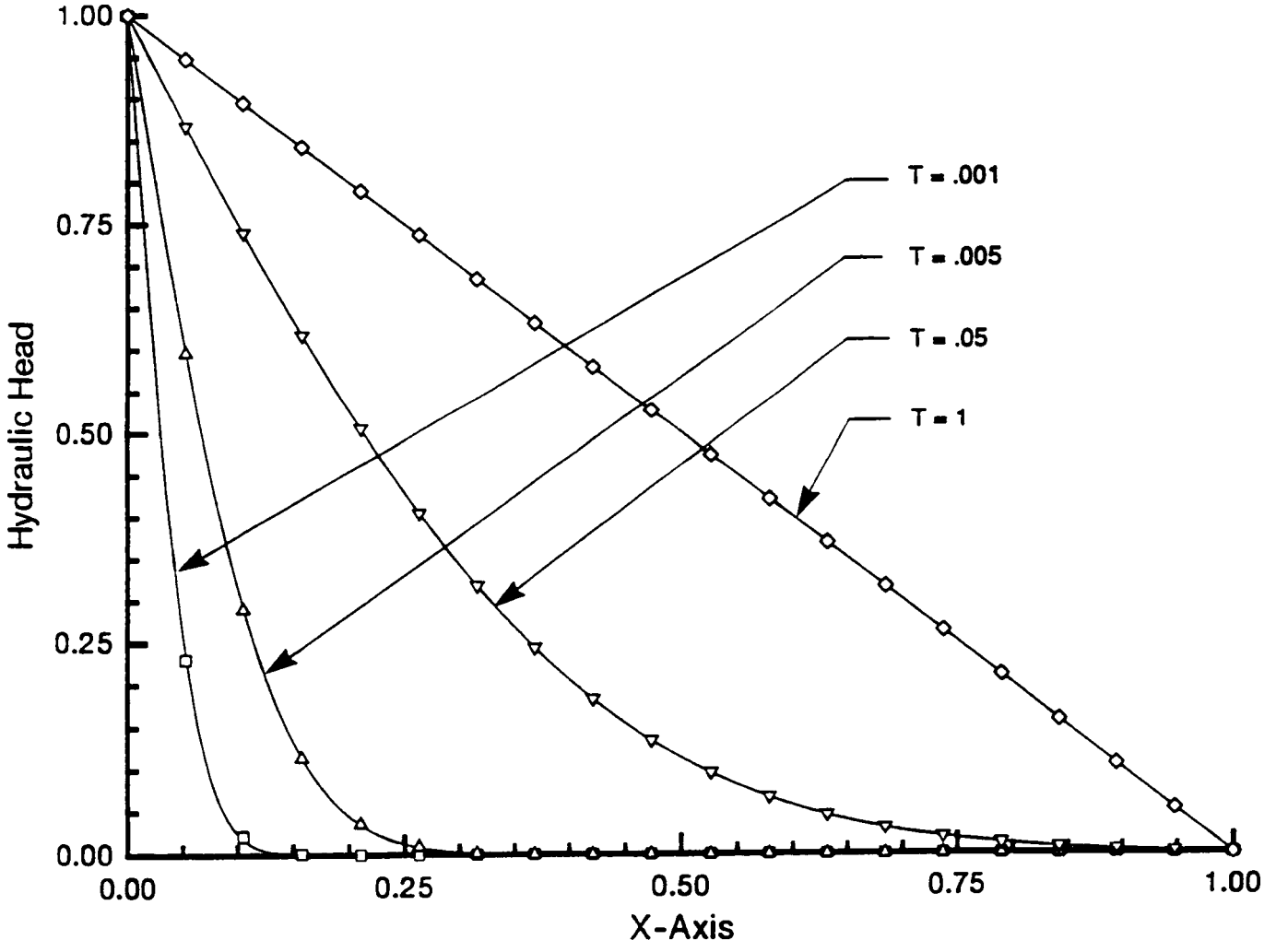


Figure 3-3. Numerical (solid curve) and analytical (discrete marks) solutions for 1D diffusion in a homogeneous medium (TEST-1). Solution is shown at t = 0.001, 0.005, 0.05, and 1.

The soil properties used are

K_s = 5.85 cm/hr
α = 0.073 cm^{-1}
θ_s = 0.3
θ_r = 0.055
n = 2.0304
m = 0.5075

A graphical comparison of BIGFLOW and the semi-analytical solution of Figure 3-4(a) shows that overall BIGFLOW performs a successful simulation of vertical movement of a moisture front in a homogeneous unsaturated soil. The differences observed near the saturation level are attributed to the effect of the difference in the number of nodes used for the moisture front calculations. Figure 3-4(b) depicts the comparison of BIGFLOW with CMVSFS for the same problem. The results are in excellent agreement, except from the very initial time steps. The speed of the moisture front is calculated to be 33 cm/hr for both codes. This speed is attained at time $t_{grav} \approx 0.8$ hr, after which gravity flow dominates, as predicted theoretically.

3.3.3 2D Infiltration Caused by a Line Source Above a Shallow Water Table

This problem describes flow from a single trickle or subsurface irrigation system of porous pipe that is placed above a shallow water table. Figure 3-5(a) shows an illustration of the physical setting of this test problem. No flow boundaries are imposed on all sides of the cube, and the initial pressure head is described by a linear distribution with zero at the bottom and -122 cm at the top. The water table is at a finite depth of 122 cm from the top surface.

Due to the symmetry and the infinite extent of the problem along the direction of the porous pipe, the three-dimensional (3D) physical problem is mathematically equivalent to a 2D problem with a point source of flux q_0 at the left top corner of the 2D domain. A no flow boundary condition is imposed on the left and right edge of the 2D domain, and the initial pressure head is given by a linear distribution with zero at the bottom and -122 cm at the top. Figure 3-5(b) shows the 2D mathematical equivalent of Figure 3-5(a). The water table is at the bottom of the 2D domain.

The analytic solution is based on the solution by Warrick and Lomen (1977), who presented a solution for steady state conditions. The 2D domain [see Figure 3-5(b)] dimensions are 61 cm and 122 cm along the X and Z axis, respectively. A total of 62 nodes and 123 nodes were evenly spaced along the X and Z axis, respectively. The following exponential Gardner model was used to characterize the conductivity-pressure head relationship of the soil

$$K(h) = K_s \exp(\alpha h) \tag{3-24}$$

where

K_s (Saturated hydraulic conductivity) = 0.00112 cm/sec
α (characteristic inverse length scale) = 0.1258 cm^{-1}
h is the pressure head
K is the hydraulic conductivity

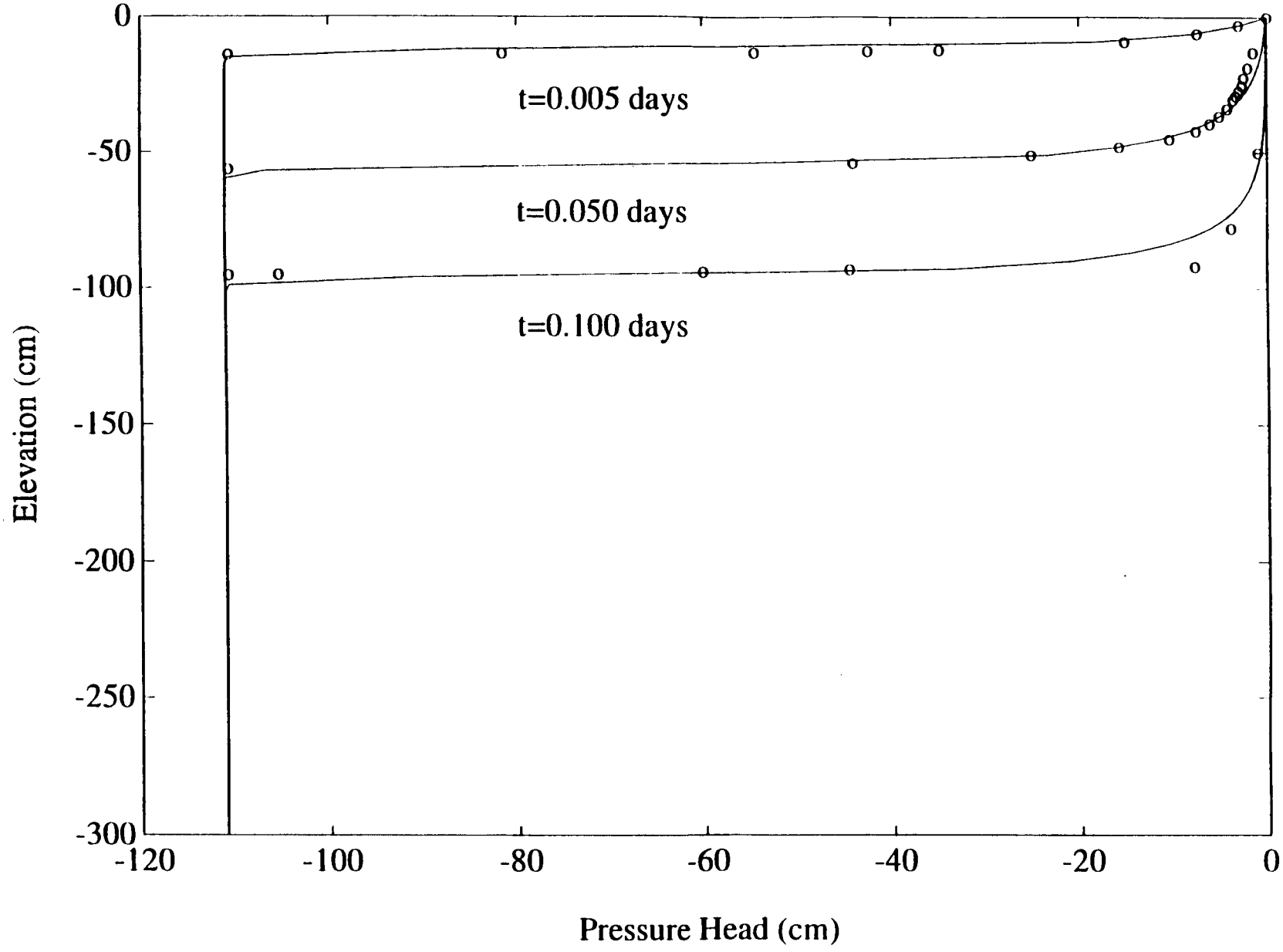


Figure 3-4(a). Comparison of numerical solution from BIGFLOW (solid line) with analytical solution (discrete marks) of Philip (1957) (TEST-2)

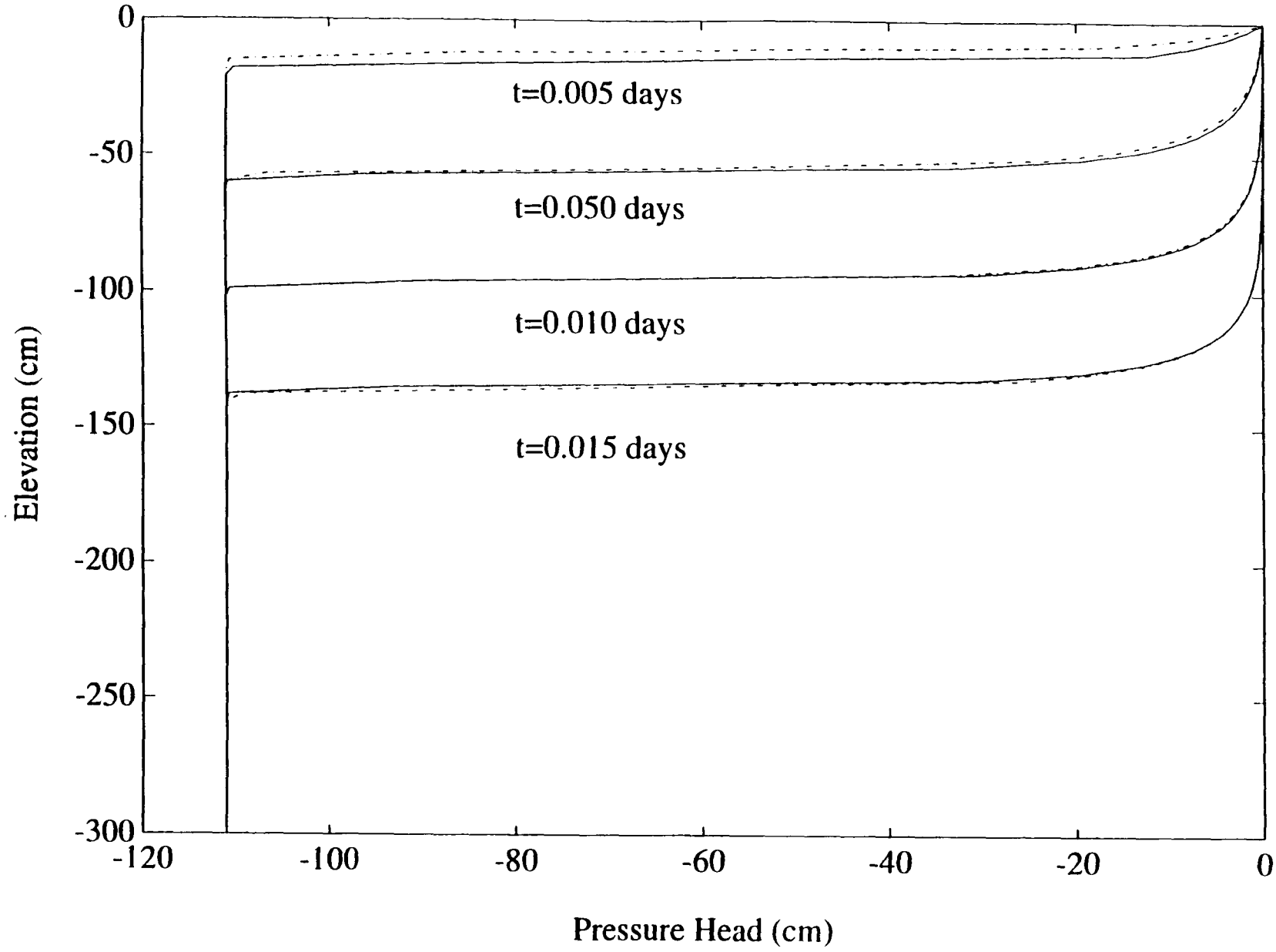
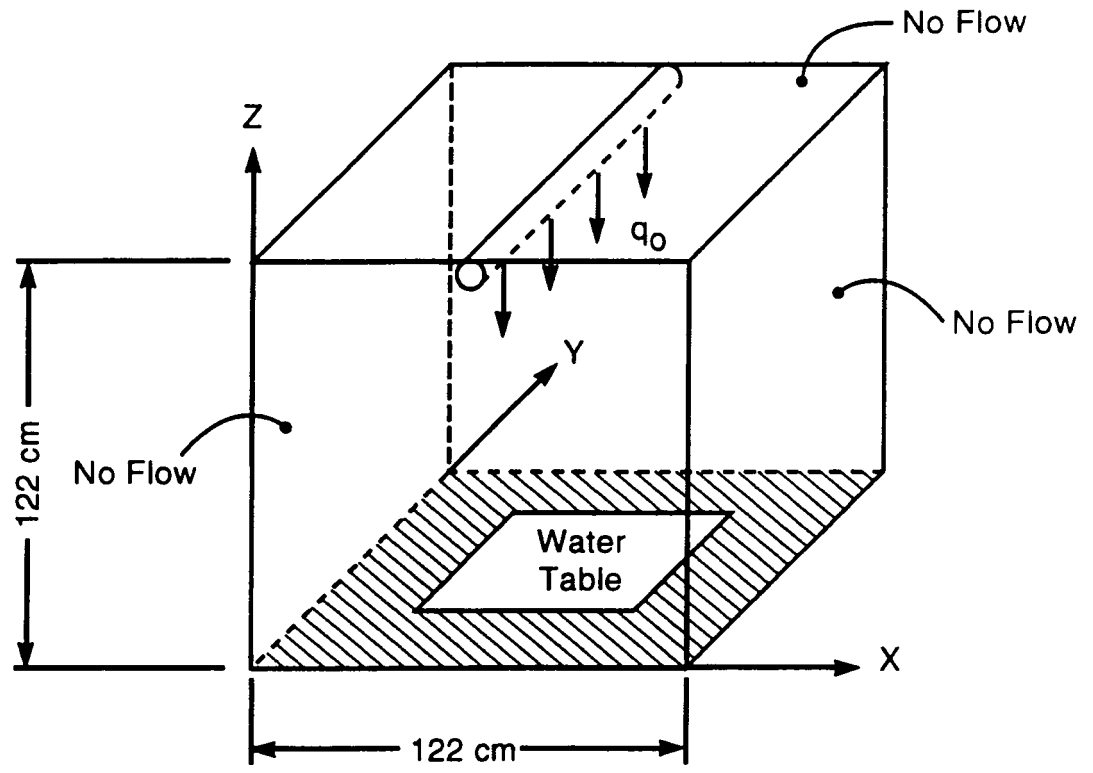


Figure 3-4(b). Comparison of numerical solution from BIGFLOW (solid line) with CMVSFS (dashed dotted line) for the 1D nonlinear vertical infiltration problem (TEST-2)



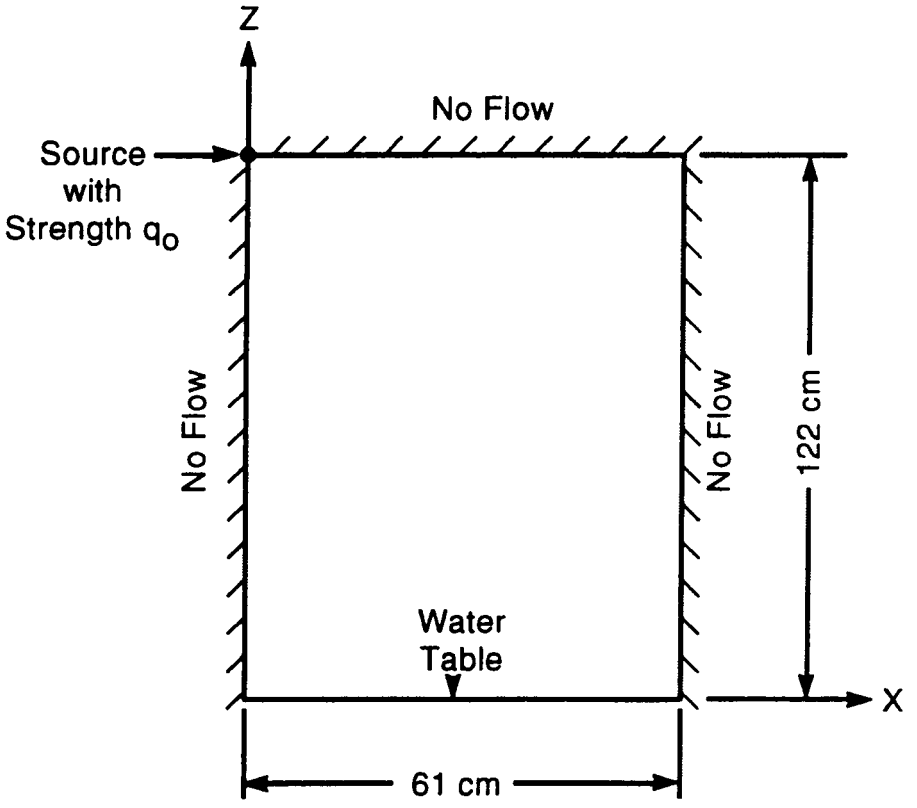


Figure 3-5. Schematic for the 2D infiltration problem with a line source (TEST-3): (a) physical 3D domain; and (b) mathematically equivalent 2D domain

The van Genuchten model used to represent the water retention curve is

$$\theta(h) = \theta_r + (\theta_s - \theta_r)\,[1 + (-\beta h)^n]^{-m} \qquad (3\text{-}25)$$

where

θ_s (saturated water moisture content) = 0.4411
θ_r (residual water moisture content) = 0.0189
n (real exponent) = 3.8720
m (real exponent) = 0.7417

The input flux q_0 is 0.00105 cm^3/sec/cm, slightly lower than the saturated hydraulic conductivity. The movement of the moisture front was analyzed by comparing steady state pressure head contours from the BIGFLOW output and those from the analytical solution of Warrick and Lomen (1977). Figure 3-6 shows this comparison. The BIGFLOW numerical solution was obtained with the nonlinear solver being set to an under-relaxation factor (HNLAX) of 0.1. The Conjugate Gradient (CG) solver (inner), and the nonlinear (outer) solver are forced to perform 200 iterations each. Although, there are very small deviations of the BIGFLOW results from the analytical solution, BIGFLOW compares well with the analytical solution. The success of this exercise demonstrated BIGFLOW's capability to reach steady state solutions directly, without marching in time.

As a further verification test, we compared the BIGFLOW results with another code, namely PORFLOW. Figure 3-7 shows PORFLOW pressure head contours at 120 hours. The PORFLOW numerical solution was obtained using the same constitutive relationships to describe the soil properties and the PORFLOW simulation was marched slowly in time. Although the PORFLOW results in Figure 3-7 correspond to only a near steady state, the similarity with Figure 3-6 provides an additional verification of the BIGFLOW results.

3.3.4 2D Infiltration Caused by a Strip Source Above a Shallow Water Table

This test problem is an extension to the problem described in Section 3.3.3. In this problem, multiple porous pipes are laid adjacent to each other (such that they are in contact). Figure 3-8(a) shows an illustration of the physical setting of this test problem. No flow boundaries are imposed on all sides of the cube, and the initial pressure head is described by a linear distribution with zero at the bottom and -122 cm at the top. The water table is at a depth of 122 cm from the top surface.

In a similar manner as described in Section 3.3.3, the symmetry and the infinite extent of the problem can be exploited to formulate a mathematically equivalent 2D problem, illustrated in Figure 3-8(b). The top boundary has two boundary conditions: (i) first half comprising a constant flux boundary with flux q_0 = 0.001 cm^3/sec/cm and (ii) second half consisting of a no flow boundary. A no flow boundary condition is imposed on the left and right edge of the 2D domain of Figure 3-8(b), and the initial pressure head is given by a linear distribution with zero at the bottom and -122 cm at the top. The water table is at the bottom of the 2D domain. The soil specifications (water retention curve, and hydraulic conductivity versus pressure head) for this test problem are identical to those described in Section 3.3.3.

The BIGFLOW simulation output was compared with PORFLOW, an independent, comparable code. Figure 3-9 shows transient pressure head comparisons at node B (X=44 cm, Z = 116 cm) [see

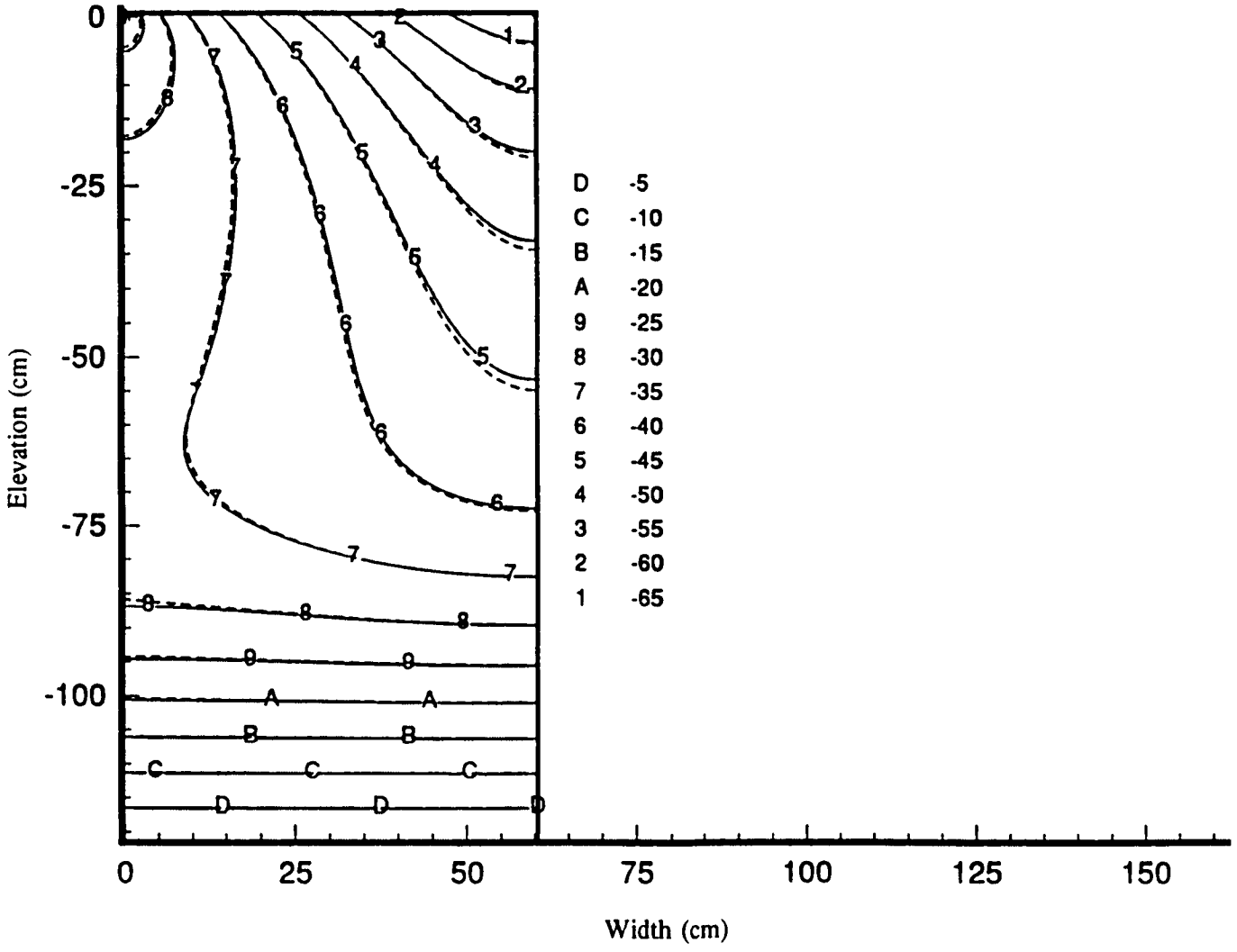


Figure 3-6. Comparison of numerical solution from BIGFLOW (solid line) with analytical solution (dashed line) of Warrick and Lomen (1977) (TEST-3). All pressure head values are in centimeters.

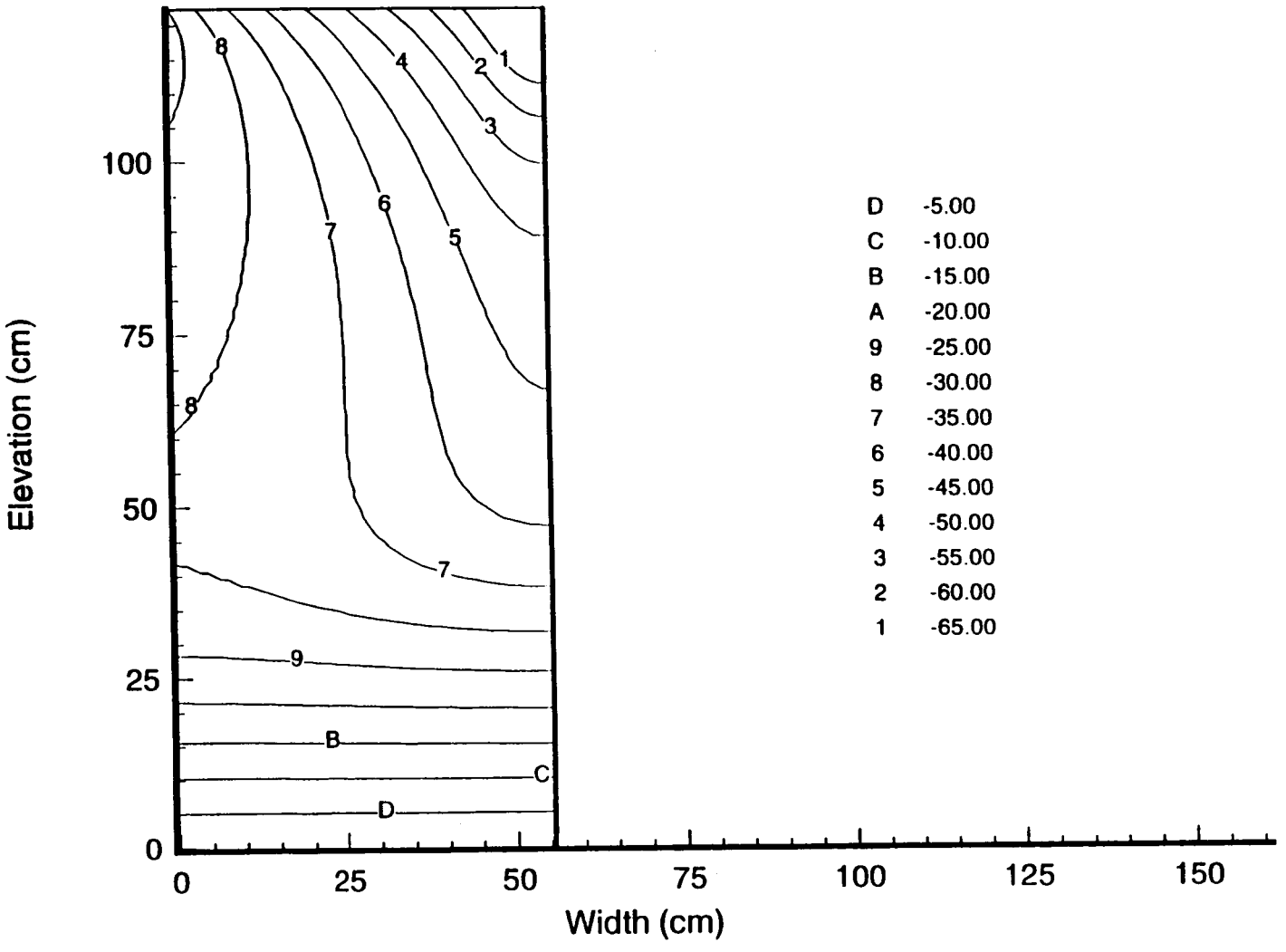


Figure 3-7. Numerical solution from PORFLOW at t = 120 hours (TEST-3). Pressure head values are in centimeters.

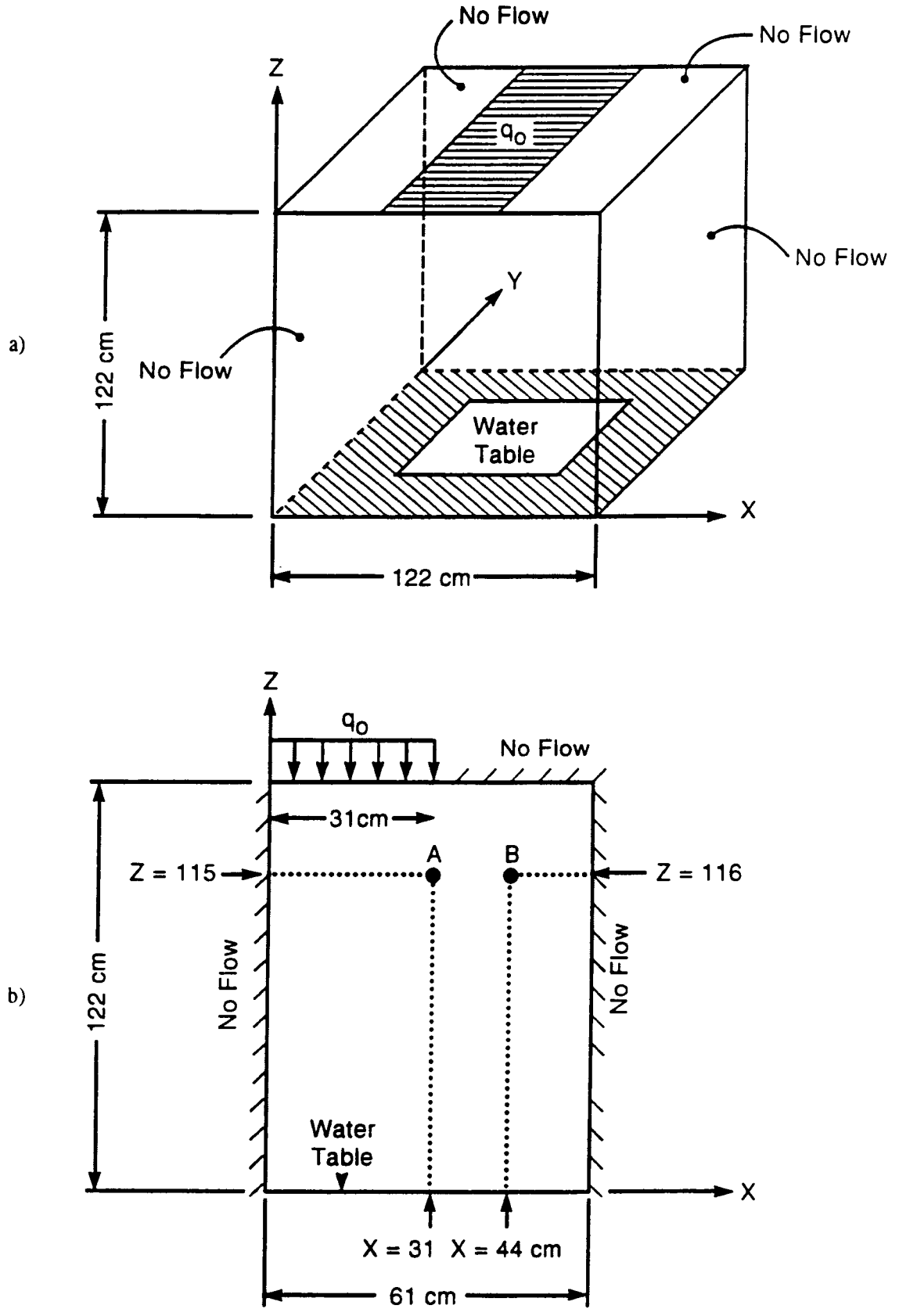


Figure 3-8. Schematic for the 2D infiltration problem with a strip source (TEST-4): (a) physical domain; and (b) mathematically equivalent 2D domain

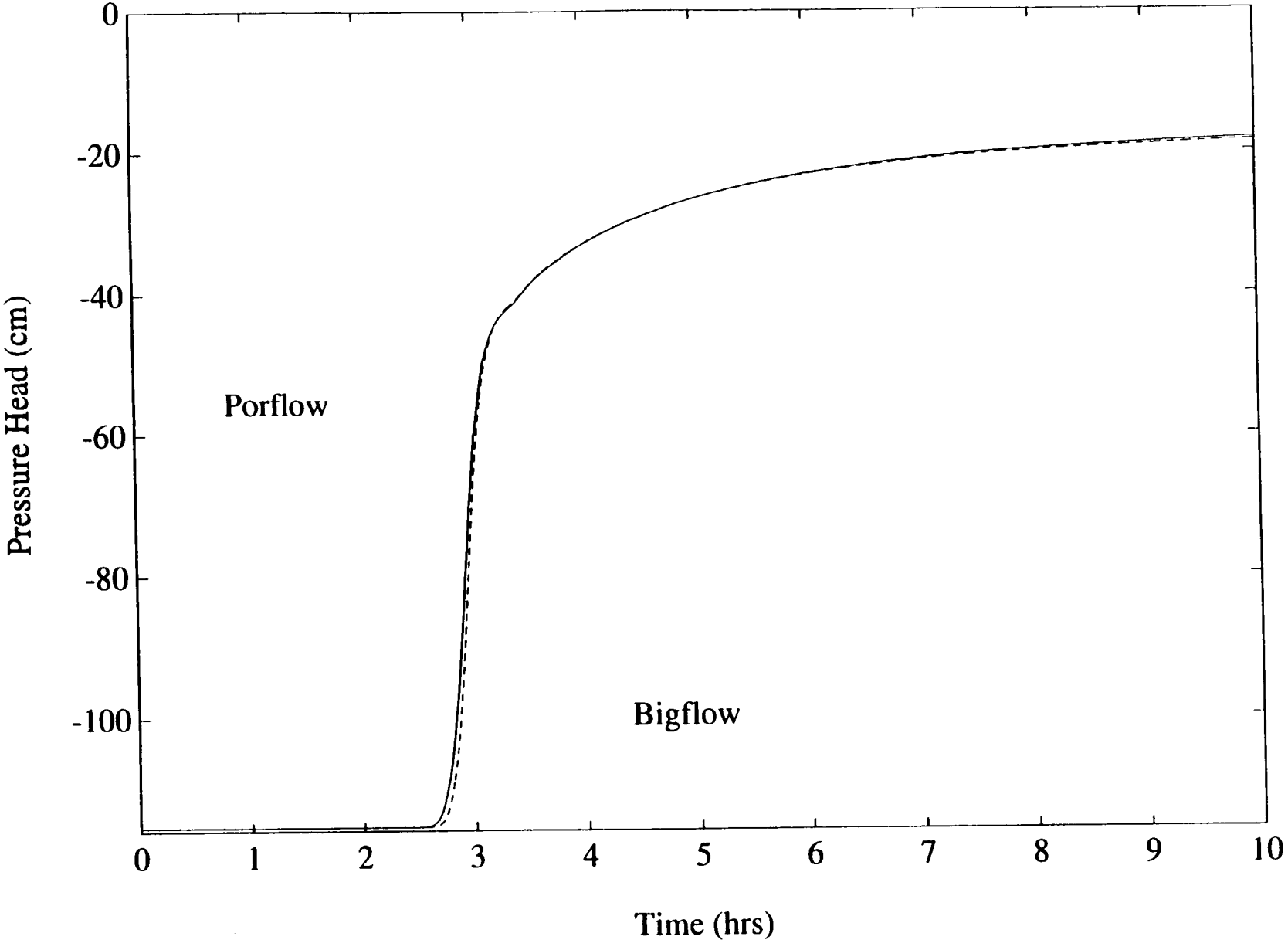


Figure 3-9. Comparison of pressure head temporal variation at node B for the BIGFLOW (dashed line) and PORFLOW (solid line) numerical solutions (TEST-4)

Figure 3-8(b)] using BIGFLOW and PORFLOW. The results are in excellent agreement.

The pressure head contours of both BIGFLOW and PORFLOW were also compared for a time snapshot at 10 hours. Figure 3-10 shows the pressure head contours from the BIGFLOW simulation and Figure 3-11 illustrates the same from the PORFLOW simulation. A comparison of the aforementioned figures reveals that both BIGFLOW and PORFLOW results are very similar but not identical. For example, the "toe" of the moisture plume has reached an elevation Z = 51 cm in the case of BIGFLOW, as opposed to an elevation of Z = 53 cm for PORFLOW. This may be a result of the different numerical techniques used within both codes, namely: BIGFLOW used the Diagonally Scaled Conjugate Gradient (DSCG) method while PORFLOW used the Alternate Direction Implicit (ADI) technique. It may also be attributed to the use of tabular constitutive relationships and the associated interpolation. This was found necessary due to the fact that PORFLOW does not accomodate the combination of the Gardner (K-h) and the van Genuchten (θ-h) relationships.

An additional verification test was conducted with the strip source spanning the whole width of the 2D domain, leading to an essentially 1D problem. Figure 3-12 depicts the comparison of the temporal variation in pressure heads at node A (X=31 cm, Z= 115 cm) for BIGFLOW and PORFLOW. Furthermore, Figure 3-13 shows a comparison of pressure heads at a vertical transact in the middle of the 2D domain at t = 3 hours. Again, the results are in very good agreement.

3.3.5 2D Infiltration in a Heterogeneous Medium

The physical setting of this problem is similar to that described in Section 3.3.4. The heterogeneity was implemented by a zone of lower saturated hydraulic conductivity as illustrated in Figure 3-14. No flow boundaries are imposed on all sides of the 2D domain, and the initial pressure head is described by a linear distribution with zero at the bottom and -122 cm at the top. The water table is at a depth of 122 cm from the top surface.

The soil specifications (water retention curve, and hydraulic conductivity versus pressure head) for this test problem are identical to those described in Section 3.3.3. Two zones are used to represent the heterogeneous medium: zone 1 which spans the entire 2D domain; and zone 2 with a width and height of 20 cm each. The X and Z coordinates of the corners of the second zone are shown in Figure 3-14. The exponential Gardner model is used to describe both the aforementioned zones. The saturated hydraulic conductivity and the characteristic inverse length scale of the Gardner model for both zones are defined as follows

ZONE 1 (entire 2D domain):
Saturated hydraulic conductivity K_s = 0.00112 cm/sec
Characteristic inverse length scale α = 0.1258 cm^{-1}

ZONE 2 (20 cm $\leq$ X $\leq$ 40 cm, 80 cm $\leq$ Y $\leq$ 100 cm):
Saturated hydraulic conductivity K_s = 2.24 x 10^{-5} cm/sec
Characteristic inverse length scale α = 0.0839 cm^{-1}

Figure 3-15 depicts the unsaturated hydraulic conductivity relationship for the two zones. The location of zone 2 in this test problem was determined such that the cross-over point, where zone 2 becomes more conductive than zone 1 is located at the interface between the two zones at point A (see

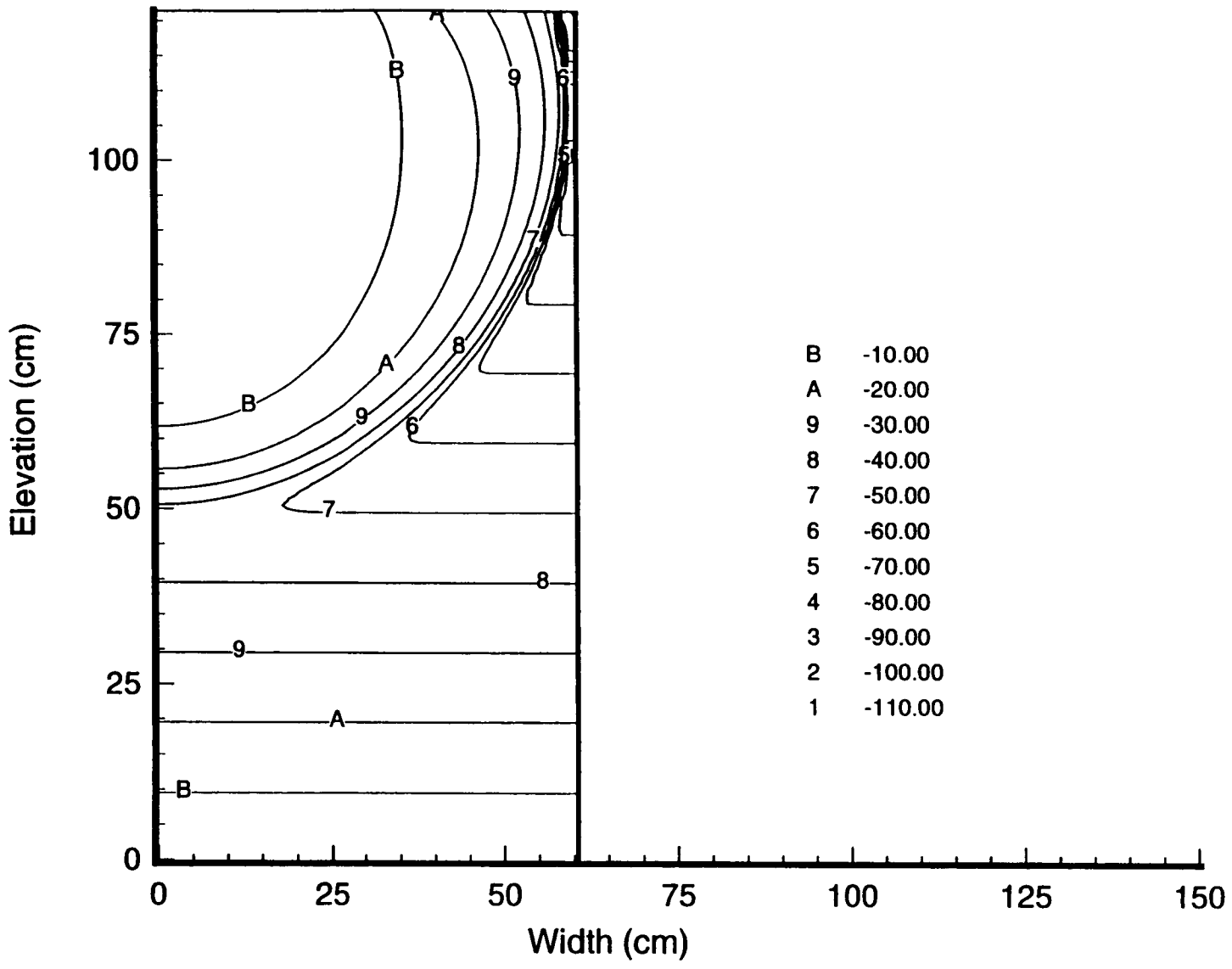


Figure 3-10. Pressure head contours for the half strip source infiltration problem (TEST-4) as obtained at t = 10 hours with BIGFLOW. Pressure head values are in centimeters.

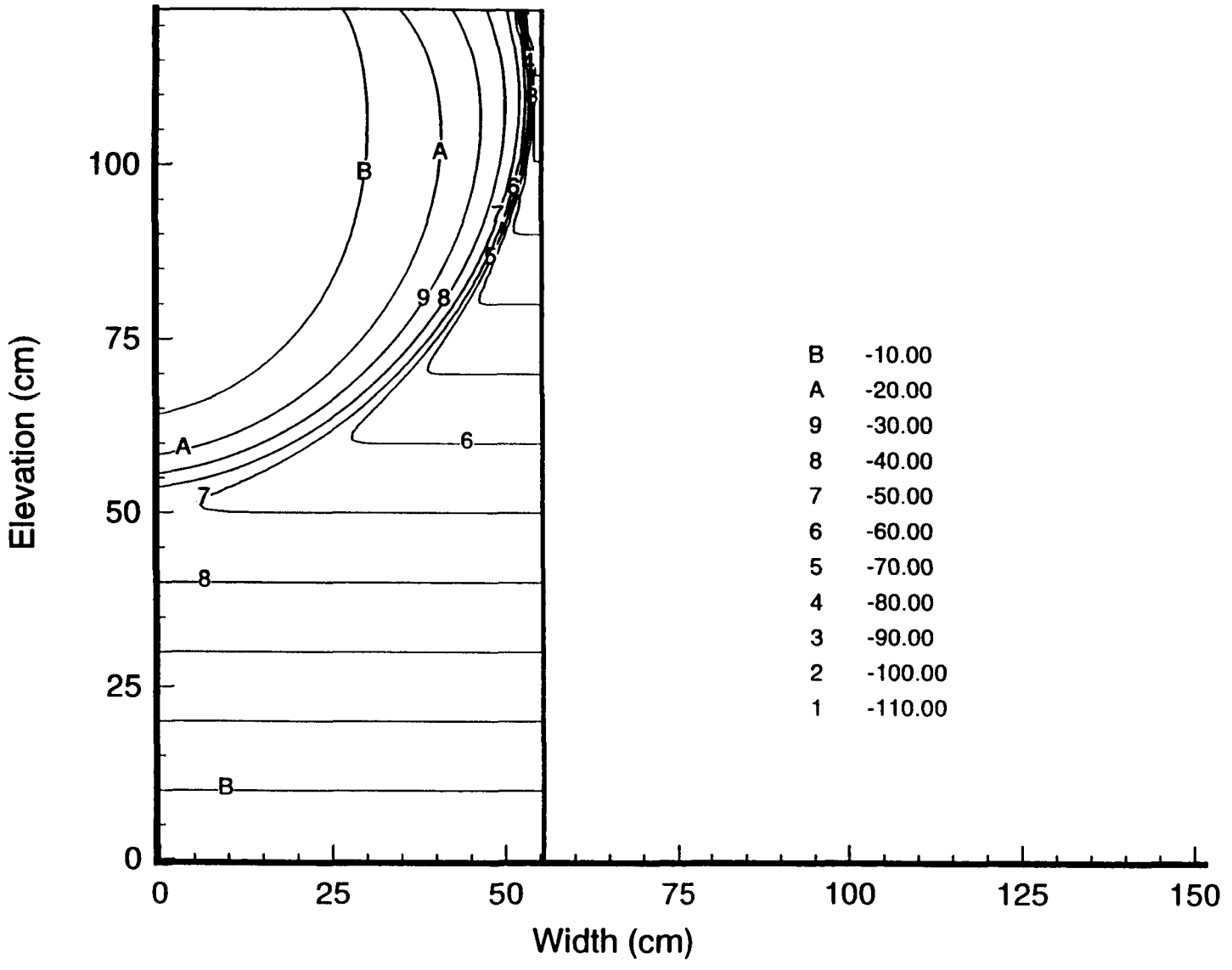


Figure 3-11. Pressure head contours for the half strip source infiltration problem (TEST-4) as obtained at t = 10 hours with PORFLOW. Pressure head values are in centimeters.

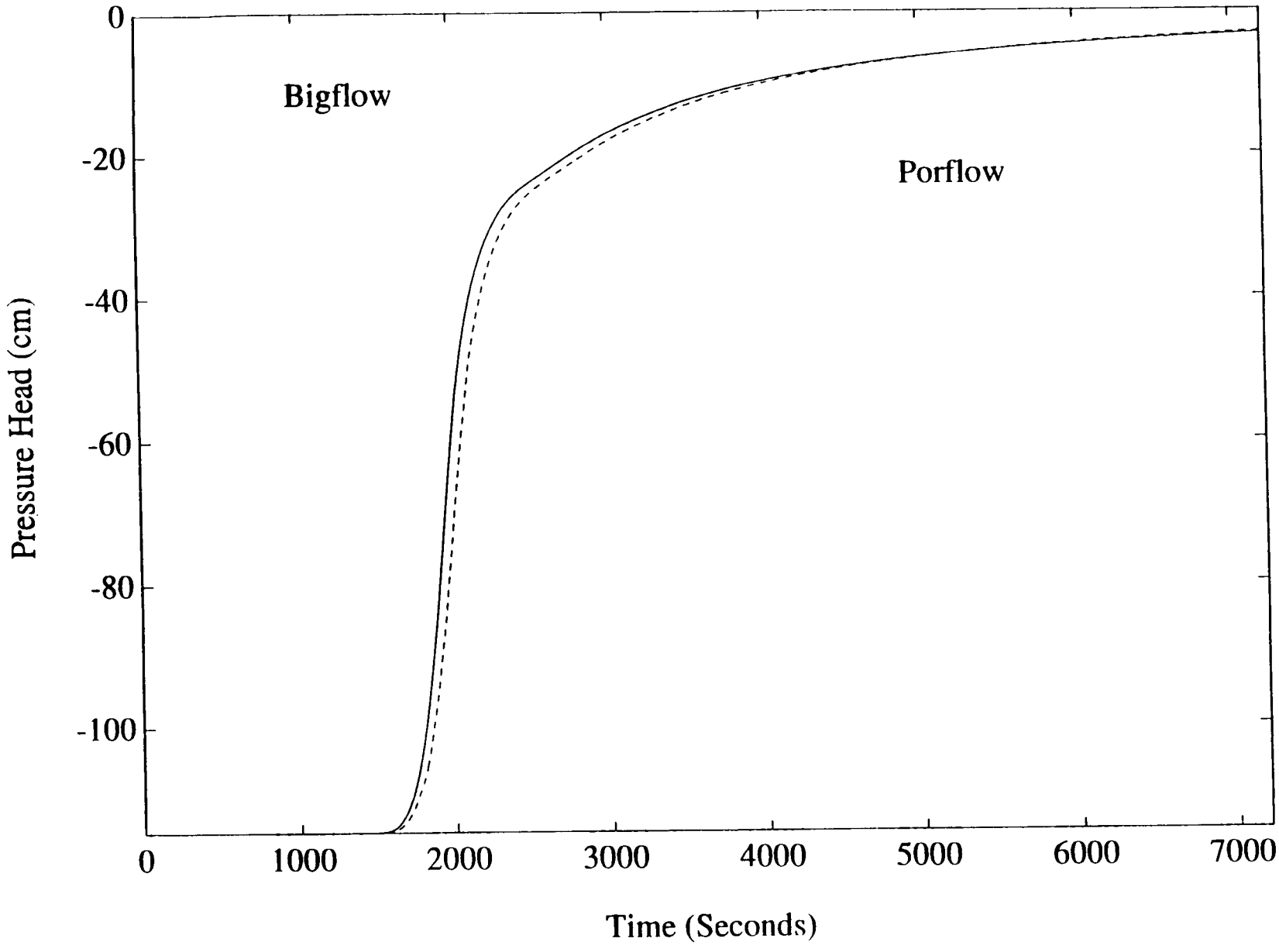


Figure 3-12. Comparison of the temporal variation in pressure heads at node A [see Figure 3-8(b)] using BIGFLOW (solid line) and PORFLOW (dashed line)

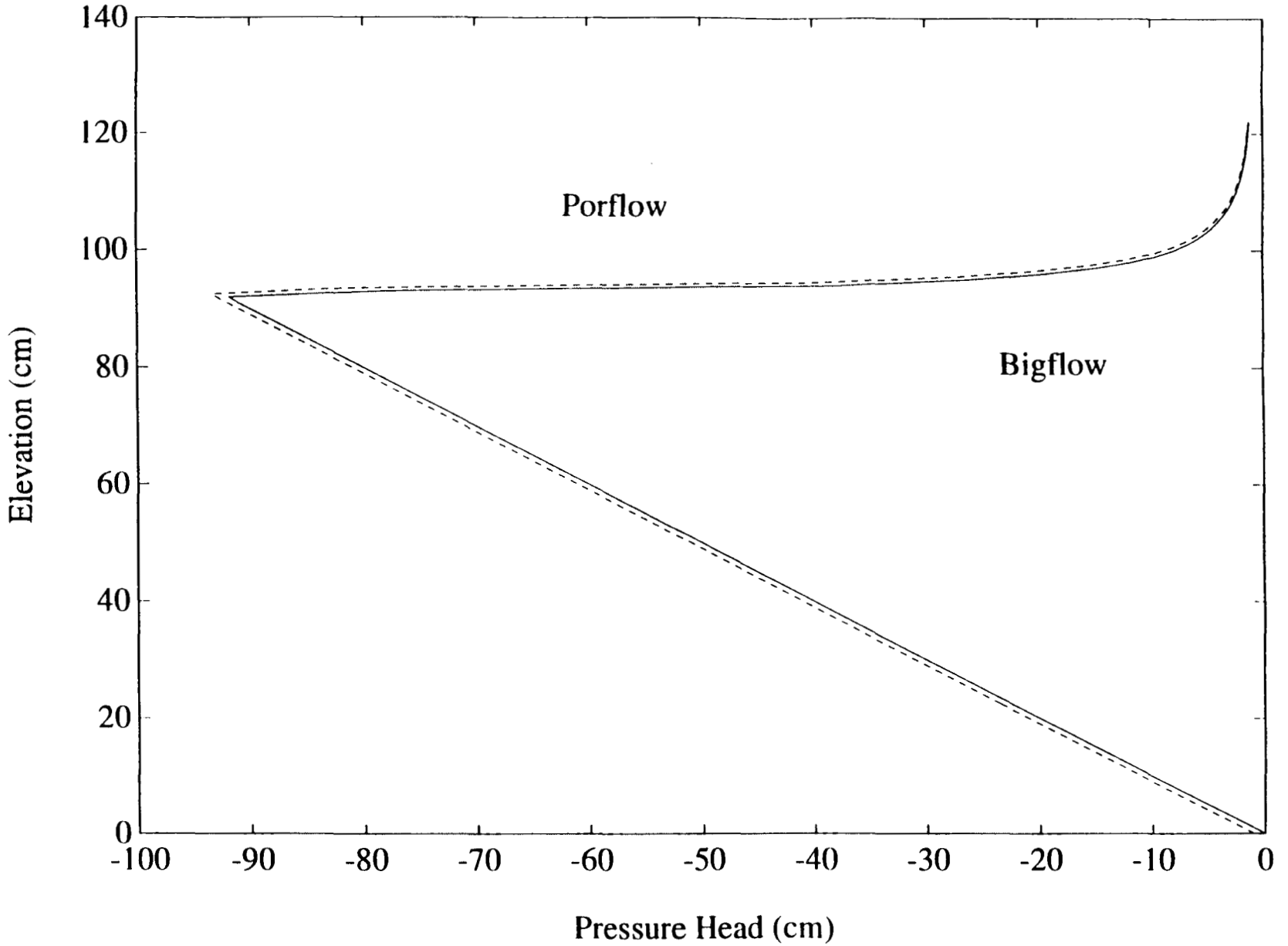


Figure 3-13. Comparison of the pressure heads at a vertical transect in the middle of the 2D domain [see Figure 3-8(b)] at time three hours using BIGFLOW (solid line) and PORFLOW (dashed line)

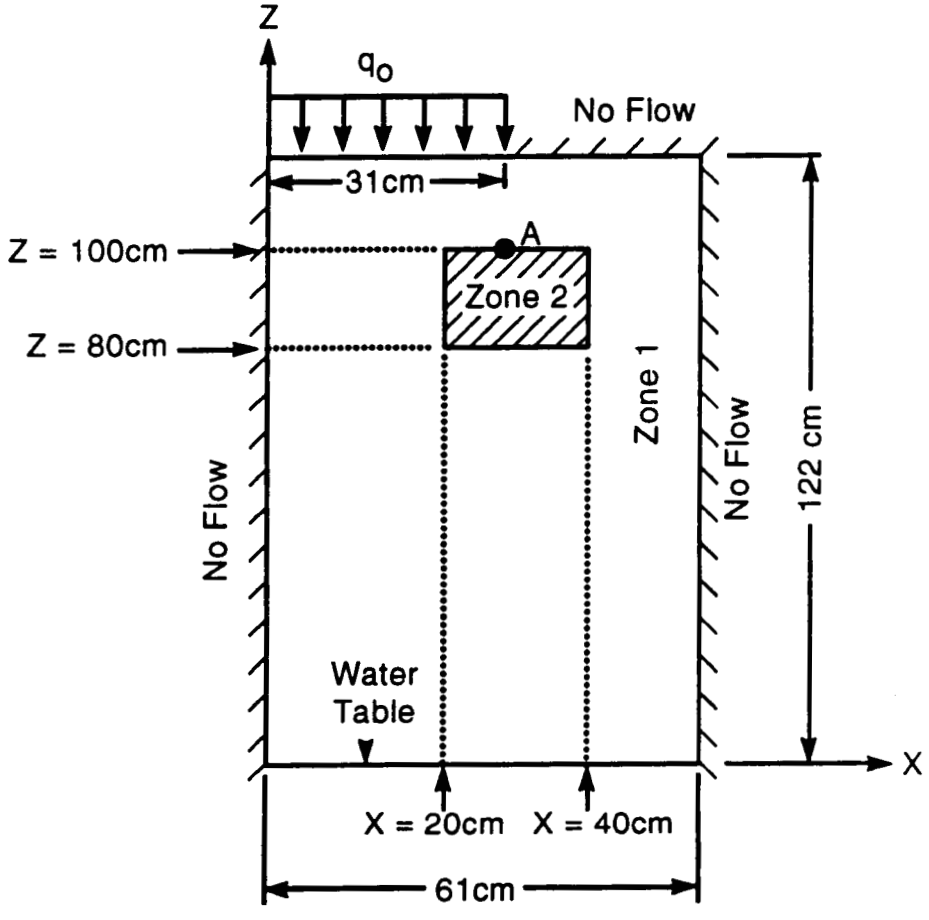


Figure 3-14. Schematic for the 2D infiltration with a strip source in a heterogeneous medium (TEST-5)

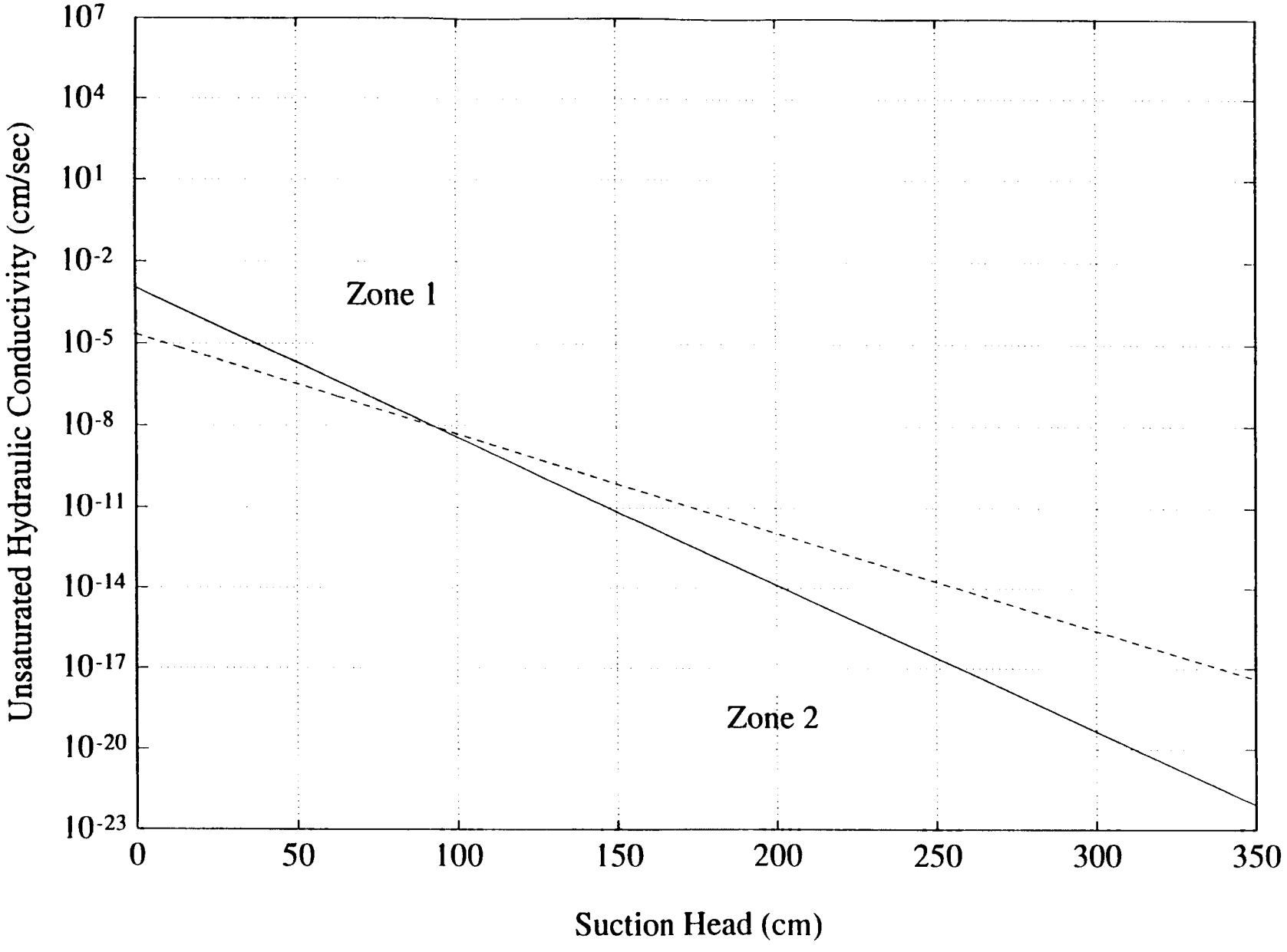


Figure 3-15. Unsaturated hydraulic conductivity relationship for the two zones of TEST-5. Zone 1 is represented by a solid line and Zone 2 is represented by a dashed line.

Figure 3-14). This makes the test problem a more challenging, computationally intensive exercise. The van Genuchten model was used to describe the water retention curve of both zones. The parameters of the van Genuchten model for both zones were selected identical to those described in Section 3.3.3.

The top of the 2D domain (see Figure 3-14) has two boundaries: (i) first half comprising a constant flux boundary with flux q_0 = 0.001 cm^3/sec/cm and (ii) second half comprising a no flow boundary. A no flow boundary condition is imposed on the left and right edge of the 2D domain of Figure 3-14, and the initial pressure head is given by a linear distribution with zero at the bottom and -122 cm at the top. The water table is at the bottom of the 2D domain.

Figure 3-16 shows the pressure head contours at a time snapshot of six hours using BIGFLOW. The PORFLOW pressure head contours at the same time are illustrated in Figure 3-17. A comparison of Figures 3-16 and 3-17 reveals a very good similarity in numerical results between the two codes. BIGFLOW completed the aforementioned simulation in 2.5 CPU hours, while PORFLOW performed the simulation in 7 CPU hours. One should also keep in mind that BIGFLOW actually solved a 3D problem with 62 × 5 × 123 nodes, five times more nodes than the number of nodes involved in the PORFLOW simulation. This comparison confirms the optimum manner with which BIGFLOW performs reliable flow simulations in heterogeneous media. Figure 3-18 shows a further comparison of the transient pressure head obtained from BIGFLOW and PORFLOW for node A located at the interface between the two zones.

The speed with which the moisture plume is travelling downwards is greater for the BIGFLOW simulations. However, at no time does the pressure head difference, between BIGFLOW and PORFLOW, become greater than 10 percent. Moreover, as the simulation approaches steady state conditions the results converge to an almost perfect agreement. Finally, it is worthwhile noticing that the BIGFLOW simulation predicts the creation of a small perched zone on top of zone 2, as indicated by the positive pressure heads.

3.3.6 Flow of Water Through a Hole in a Box Under Pressure

This problem is a mathematical analog of flow of water from an artesian spring. Even though no direct comparison with the BIGFLOW results was conducted, this problem tested the ability of the code to simulate boundary conditions of different types. In particular, this problem involved no flow and constant head boundary conditions co-existing on the same planar face. Figure 3-19 presents a schematic of this problem. In Figure 3-19(a) a vertical cross-section is depicted, whereas Figure 3-19(b) shows a horizontal plane view.

The dimensions of the cubical domain are 100 × 100 × 100 cm, discretized into 51 × 51 × 51 nodes. The flow system is saturated and the hydraulic conductivity is K = 1 cm/sec. The boundary flux is q = 1 cm^3/sec/cm and the head at the top boundary is maintained at H_o = 1000 cm. Figure 3-20 depicts a horizontal map of the steady state hydraulic head at a plane Z = 50cm. The physics of the problem are honored since the equipotential lines are perpendicular to the no flow boundaries and the lowest point of the equipotential surface is at the prescribed head region. The physical soundness of these results is further demonstrated in Figure 3-21 and 3-22 where streamlines are shown for a vertical cross-section at X = 50cm and a horizontal cross-section at Z = 50cm, respectively. Once again, the results appear to be very reasonable since the streamlines converge to the prescribed head opening of the cubical domain.

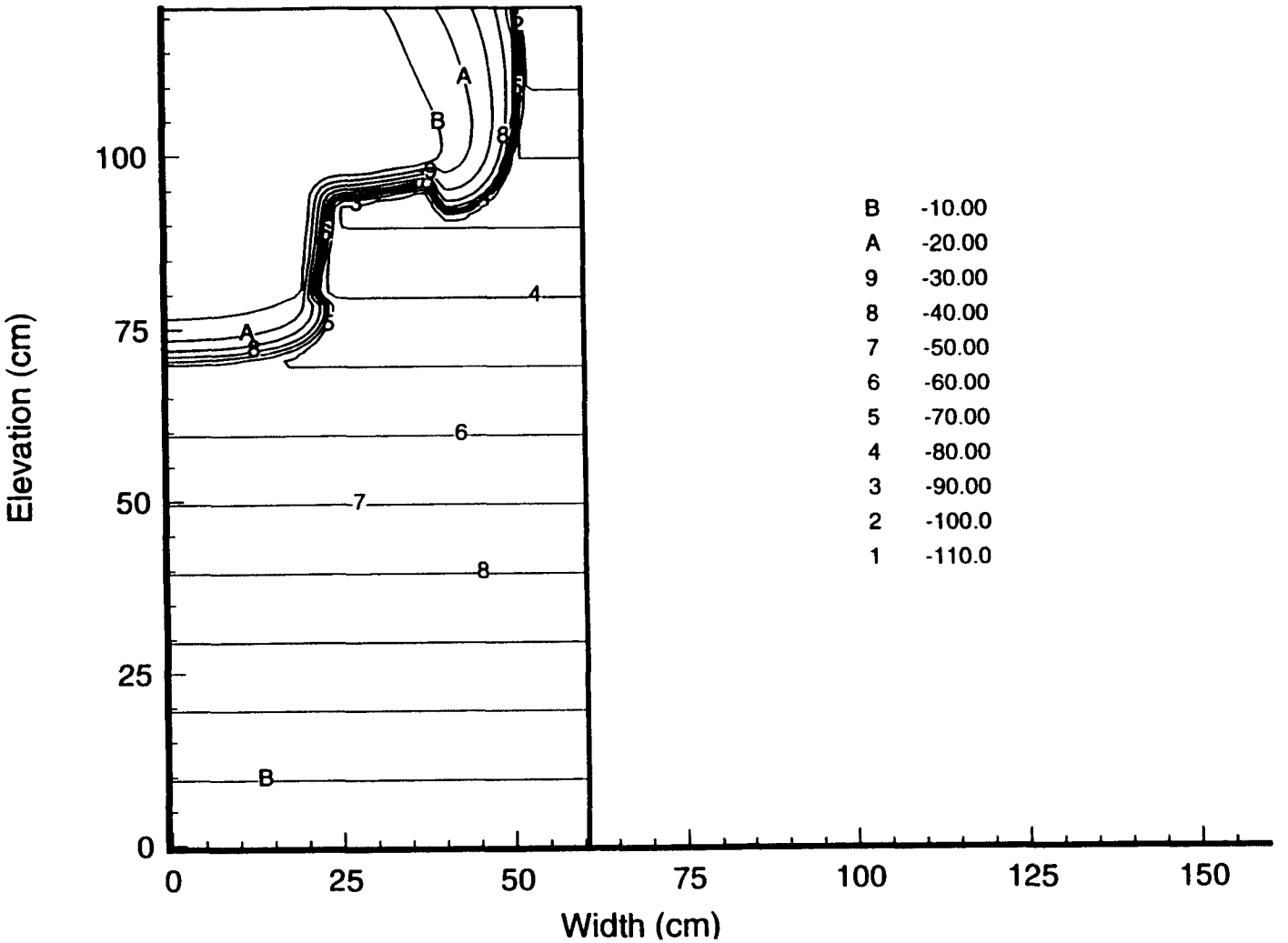


Figure 3-16. Pressure head contours for 2D infiltration in a heterogeneous medium (TEST-5) as obtained at t = 6 hours with BIGFLOW. Pressure head values are in centimeters.

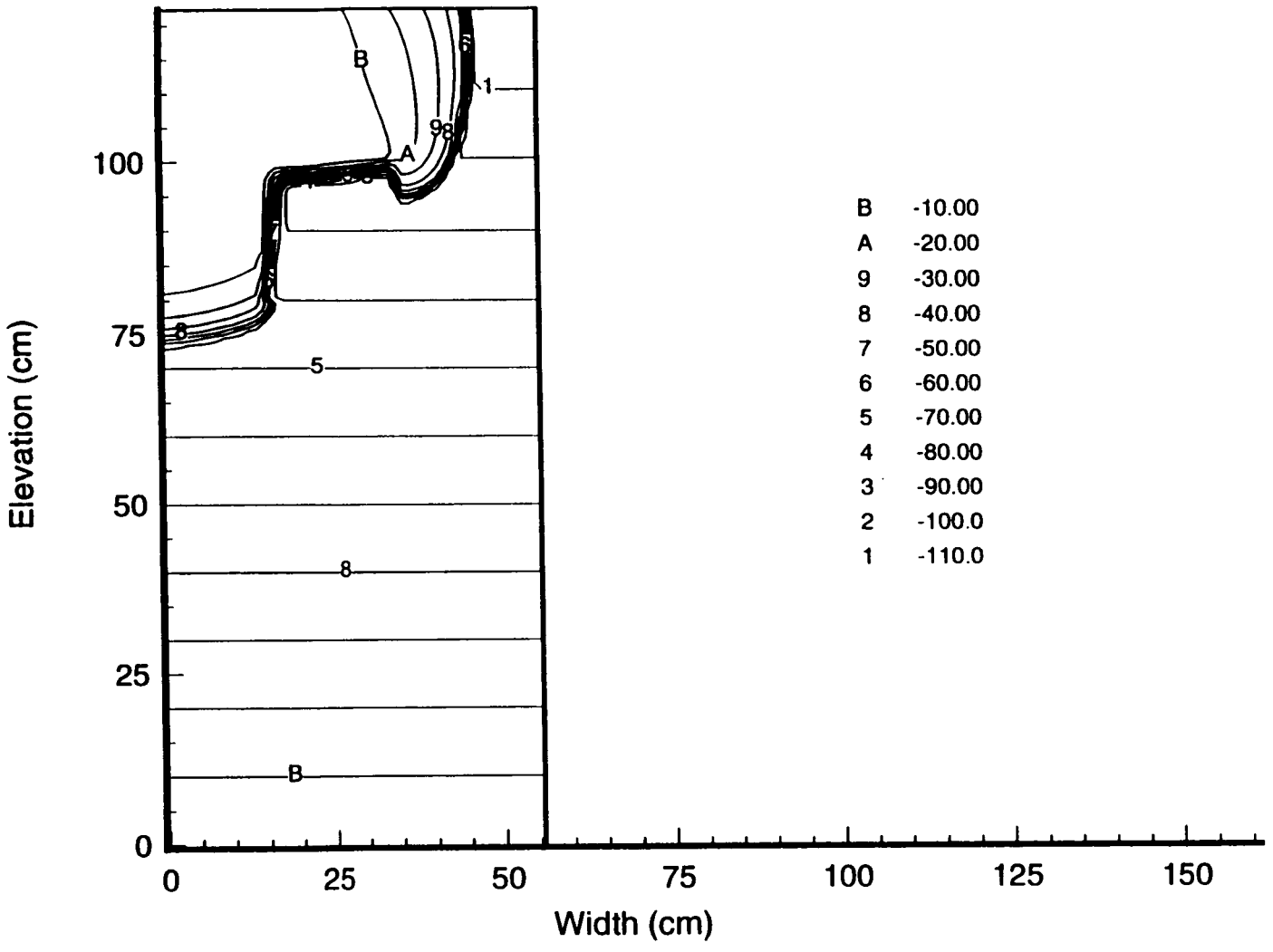


Figure 3-17. Pressure head contours for 2D infiltration in a heterogeneous medium (TEST-5) as obtained at t = 6 hours with PORFLOW. Pressure head values are in centimeters.

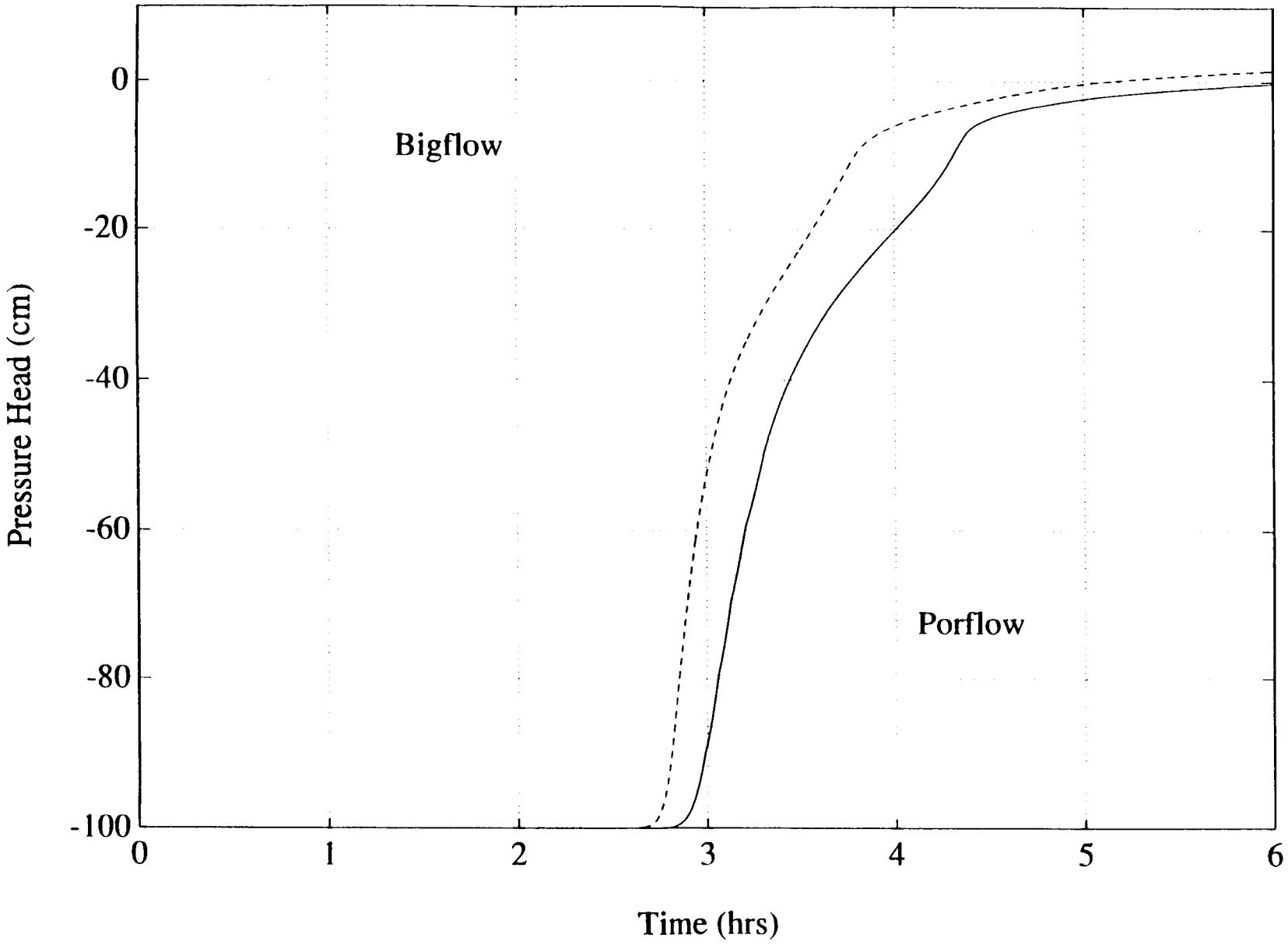
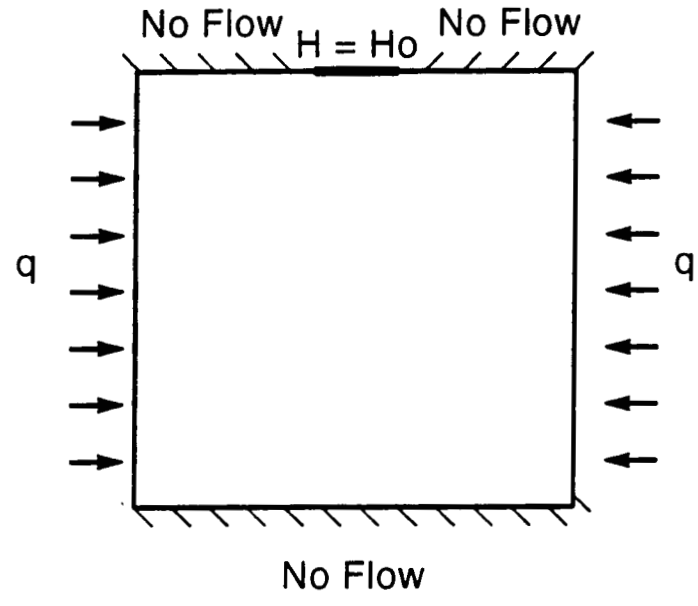


Figure 3-18. Comparison of the temporal variation in pressure heads at node A (See Figure 3-14) using BIGFLOW (dashed line) and PORFLOW (solid line)

a)



b)

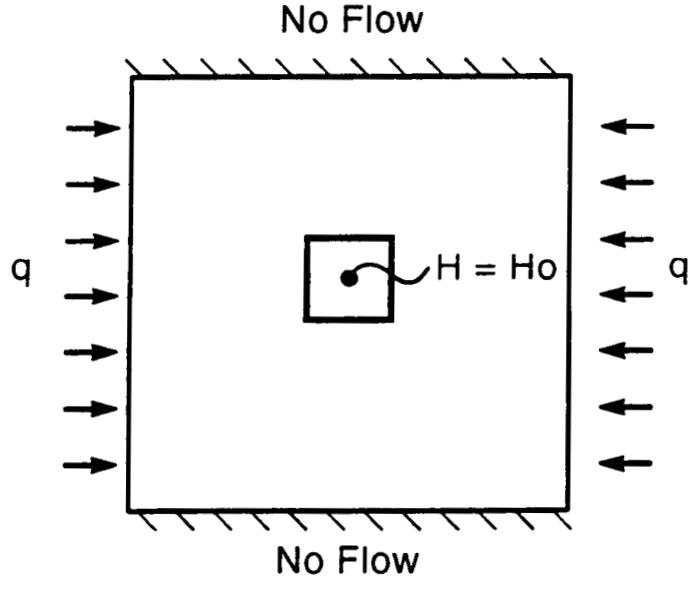


Figure 3-19. Schematic for the flow of water through a hole problem (TEST-6): (a) vertical cross-section, and (b) horizontal plane view

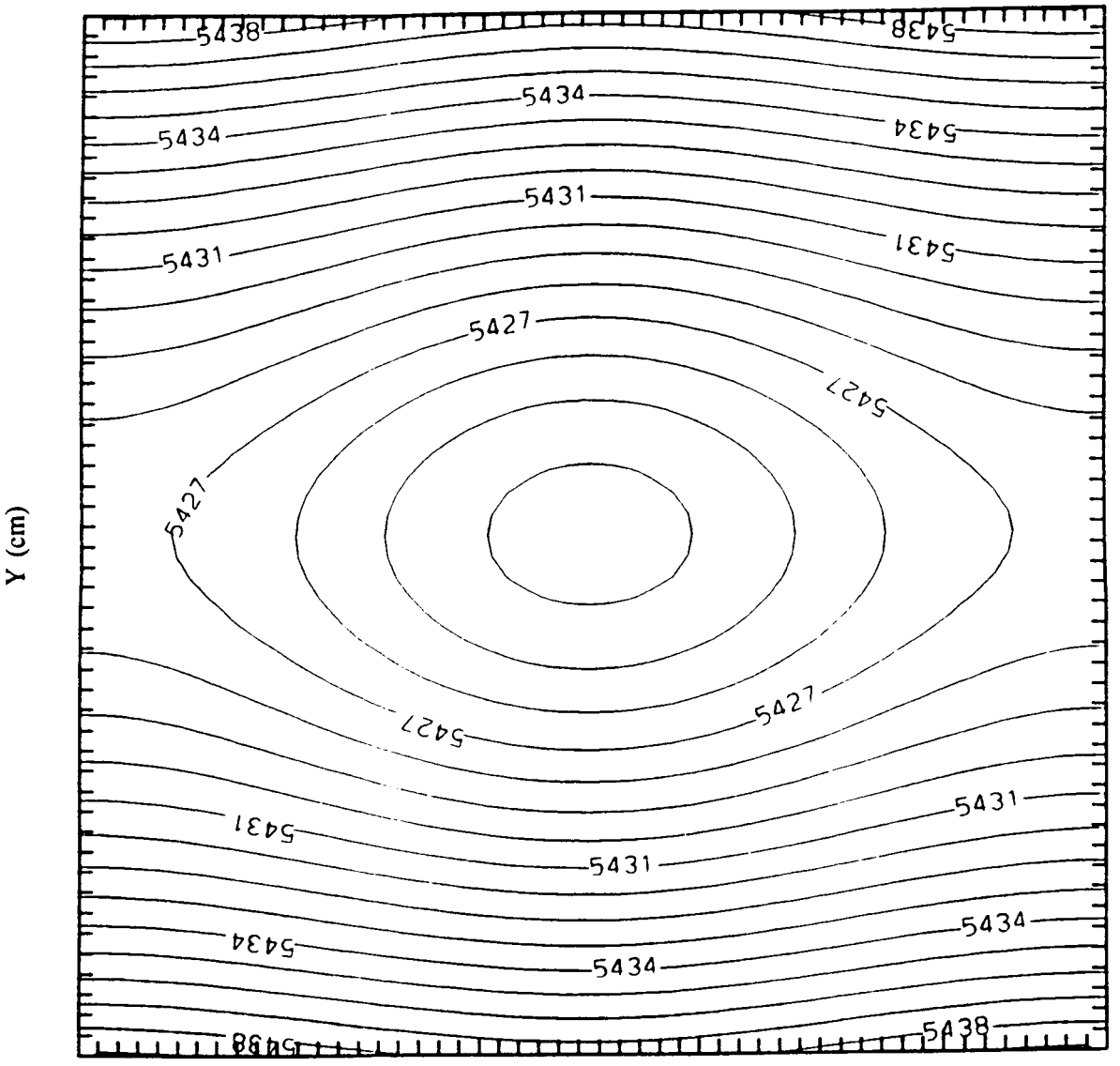


Figure 3-20. Hydraulic head contour for a horizontal plane Z = 50 cm for TEST-6

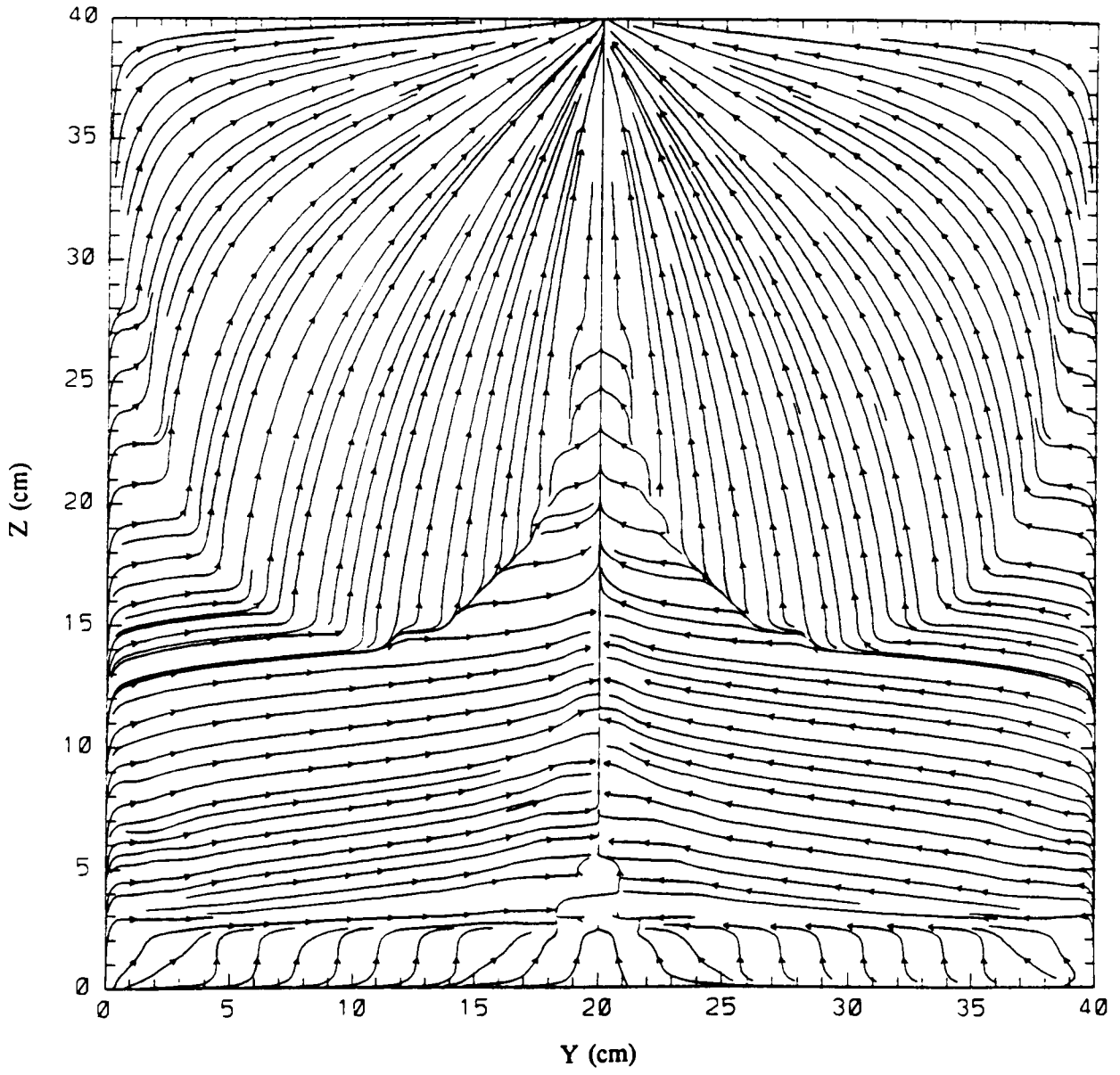


Figure 3-21. Streamlines for problem TEST-6 at a vertical plane X = 50 cm

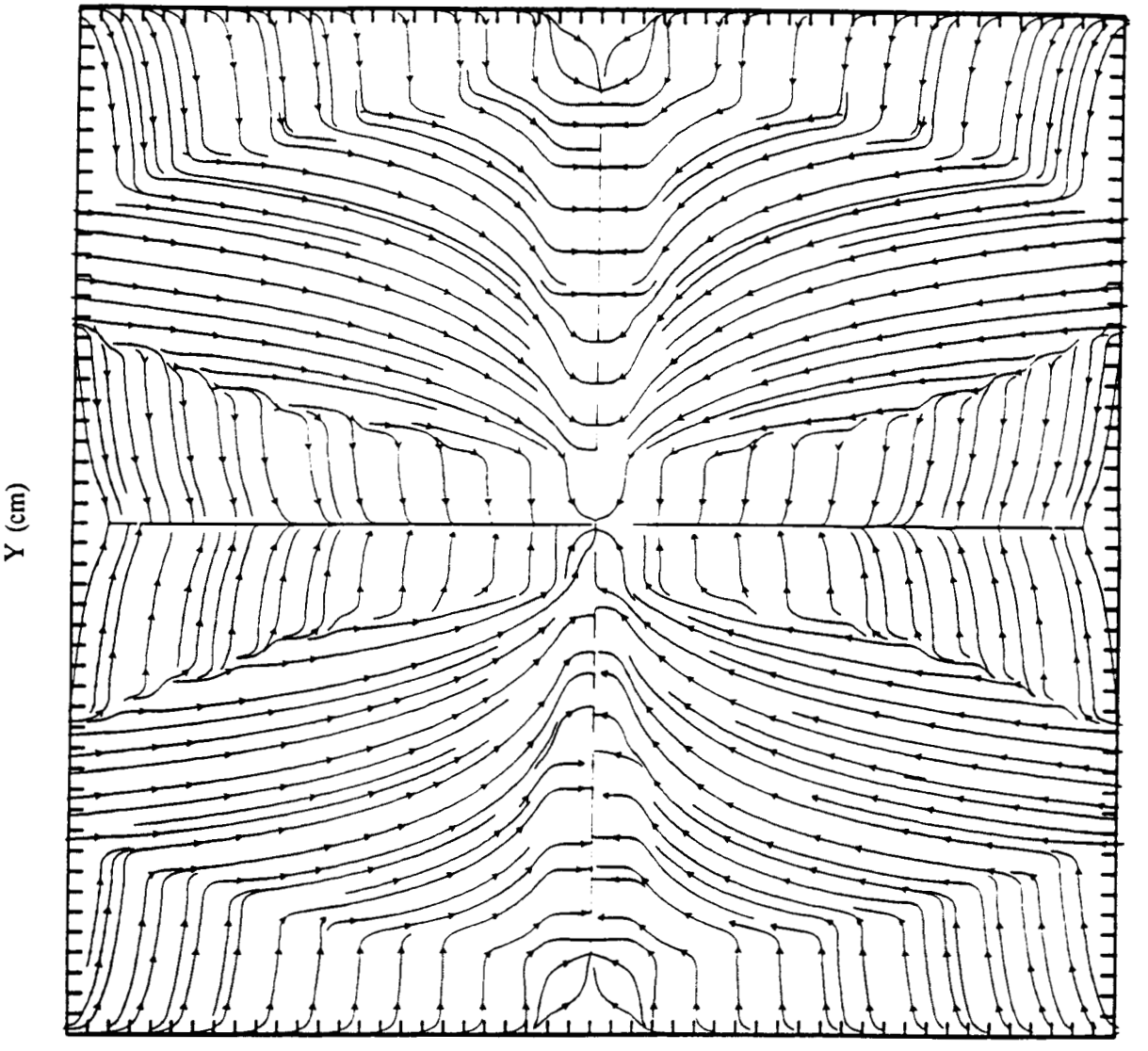


Figure 3-22. Streamlines for problem TEST-6 at a horizontal plane Z = 50 cm

4 BIGFLOW USER'S GUIDE: FLOW SIMULATIONS AND DATA PROCESSING

4.1 BIGFLOW PACKAGE DESCRIPTION

The BIGFLOW Package (Fortran source files) comprises the following components:

Flow Simulator: BIGFLOW

This Fortran file contains the entire simulation code, except for COMMON blocks located in a separate file (COMBIG) which must be located in the same directory. The BIGFLOW code does not require direct interaction with the user from the console and can, therefore, be executed in "batch" mode as well as "interactive" mode.

Auxiliary File: COMBIG

This Fortran file is an INCLUDE file which contains all the COMMON blocks used in BIGFLOW; it is required to be present in the same directory as BIGFLOW when compiling BIGFLOW.

Data Processor: DATAFLOW

This interactive Fortran program contains several routines for efficiently processing BIGFLOW Inputs/Outputs (I/Os), for example making input files for BIGFLOW, and performing analyses of multi-dimensional flow data. Outputs of DATAFLOW are in ASCII or binary form, but they are not graphics metafiles. The DATAFLOW code requires interaction with the user from the console. Thus, it can only be executed in "interactive" mode, not in "batch" mode.

Related Codes for Data Generation and Data Visualization:

Turning Band Code: CTURN

This Fortran program generates three-dimensional (3D) random fields using the Turning Band Method (TBM). While it is not considered part of the BIGFLOW package, it may be used to generate random field datasets in a format directly accepted by BIGFLOW. For a reference on the 3D Turning Band Method, see Tompson and others (1989). The TBM code (CTURN) has been extensively tested through its use in simulations of stochastic flow (Ababou, 1988; and Ababou et al., 1992a). It is a slightly modified version of the code previously used by Tompson and others (1987 and 1989), and it is available upon request from the authors of this report.

Graphics Package for Personal Computers: PLOTFLOW

This auxiliary graphics package, written by Ashok Nedungadi of Southwest Research Institute, can be used to display BIGFLOW data on personal computers under the DOS operating system. A "VGA" color monitor and graphics card are required. This package complements the BIGFLOW package described above, but it is not considered part of it, and has not been extensively tested to this date. Briefly, PLOTFLOW is composed of a set of user friendly interactive programs for visual display of one-dimensional (1D) and two-dimensional (2D) datasets compatible with BIGFLOW data formats.

The 2D data can be either scalar fields, leading to color shaded contour plots, or vector fields, leading to vector plots. A typical usage is as follows: first, the DATAFLOW processor may be used to produce 3D vector fields (flux) from scalar fields (head); second, DATAFLOW can be used to extract 1D transects and/or 2D slices from scalar or vector 3D datasets; and finally, PLOTFLOW can be used for quick visualization of these transects or slices on the screen. PLOTFLOW can be obtained upon request from the authors of this report.

4.2 BIGFLOW CODE STRUCTURE

The BIGFLOW computer code has 43 subroutines arranged in at least five levels. The first level is the main program MAINFLO, which dimensions a single "master array" (ABIG). MAINFLO calls the second level routine MIDFLO once, in such a way that the master array is broken up in a number of arrays. MIDFLO is mainly a memory manager or "middle man" which reads the basic input file INPUT1 and allocates to each of these arrays the exact dimension required for the case at hand (if the array is not needed, its dimension is reduced to one). MIDFLO then calls the third level routine SUBFLO, usually only once (except in the case of dynamic control of domain size — see "ACTIV" option).

SUBFLO is the largest subroutine of BIGFLOW (in terms of executable code). It performs a number of tasks, such as reading multi-dimensional datasets, initializing arrays, performing some limited algebra, and most importantly, managing the flow of conditional calls to many algebraic and I/O modules. These modules are fourth level routines of BIGFLOW.

The computational kernel of BIGFLOW is composed of just a small subset of the fourth-level modules, principally the matrix solver modules. The fourth level modules may in turn call fifth level modules: for example, the solver modules call certain error norm computation modules many times.

The names and functions of all the subroutines composing the BIGFLOW code (in alphabetical order) are:

No.	Subroutine	Function
1.	BCL10	Implements boundary conditions after subroutine "SYSL" has been called.
2.	BCL20	Implements boundary conditions after subroutine "SYSL" has been called.
3.	BCL30	Implements boundary conditions after subroutine "SYSL" has been called.
4.	BCNL10	Implements boundary conditions (Fixed Head) for the nonlinear system (unsaturated flow). The temporary head array is also updated at boundary nodes.
5.	BCNL20	Implements boundary conditions (Fixed Flux) for the nonlinear system (unsaturated flow). The temporary head array is also updated at boundary nodes.

No.	Subroutine	Function
6.	BCNL30	Implements boundary conditions (Zero Head Gradient) for the nonlinear system (unsaturated flow). The temporary head array is also updated at boundary nodes.
7.	BFLUX1	Computes values of normal fluxes at boundaries, that is at midnode locations adjacent to boundaries.
8.	BFLUX2	Computes values of normal fluxes at boundaries, that is at midnode locations adjacent to boundaries.
9.	BFLUX3	Computes values of normal fluxes at boundaries, that is at midnode locations adjacent to boundaries.
10.	BHEAD1	Computes values of array "AH" at boundary nodes for output purposes only.
11.	BHEAD2	Computes values of array "AH" at boundary nodes for output purposes only.
12.	BHEAD3	Computes values of array "AH" at boundary nodes for output purposes only.
13.	BHZERO	Sets the boundary node values of the head array to be zero
14.	DNORM2	Computes the L_2-Norm of the difference between two array variables
15.	DNORM3	Computes the L_∞-Norm of the difference between two array variables
16.	DSCALE	Performs a scaling of the matrix system just before and after some of the matrix solver calls if the diagonal scaling option is activated.
17.	ICCG	Solves the linear system $[A]\{\underline{h}\} = \{\underline{b}\}$ by the method of "Preconditioned Conjugate Gradients;
18.	ICFAC	Finds the L matrix in the incomplete Cholesky factorization of the original matrix [A]
19.	LCOND	Computes three 3D Arrays of midnodal saturated conductivity.
20.	NORM2	Computes the L_2-Norm of the solution error
21.	NORM3	Computes the L_∞-Norm of the solution error
22.	MICFAC	Finds the L-matrix in the modified Incomplete Cholesky Factorization of the original matrix [A].
23.	MIDFLOW	Reads part of the inputs (file INPUT 1), computes array dimensions and array addresses in array "ABIG", and calls the subroutine SUBFLO
24.	NLCOND	Computes the nonlinear unsaturated conductivity as a function of pressure head.

No.	Subroutine	Function
25.	SIP1	Solves the linear system of BIGFLOW using a standard SIP algorithm (no alternate node ordering)
26.	SIP2	Solves the linear system of BIGFLOW using either a fixed node ordering (standard) or alternate node ordering (standard/reverse) SIP algorithm.
27.	SIPFAC	Performs a SIP factorization
28.	SUBFLO	Reads various three-dimensional data sets that may be needed for the particular case at hand, and calls other fourth level modules.
29.	SYSL1	Computes coefficients of the discrete system excluding the effect of boundary conditions, terms in coefficients (ARHS, AD) that depend (linearly) on the past solution, and/or the variable time step. This routine is used for both transient and steady saturated flow.
30.	SYSNL1	Computes coefficients of the nonlinear system, excluding the effect of boundary conditions (unsaturated flow).
31.	SYSNL2	Computes coefficients of the nonlinear system, excluding the effect of boundary conditions (unsaturated flow).
32.	SYSNL3	Computes coefficients of the nonlinear system, excluding the effect of boundary conditions (unsaturated flow).
33.	THETA10	Computes the water content-pressure head curve (piecewise linear).
34.	THETA20	Computes the water content-pressure head curve (exponential with cut-off).
35.	THETA30	Computes the water content-pressure head curve (van Genuchten Function).
36.	THETA31	Computes the water content-pressure head curve (Composite Dek and Dieri soils).
37.	THETA32	Computes the water content-pressure head curve (Composite Dek and Montfavet soils).
38.	TSTEP	Computes the next time step, not including the initial (first) time step which is computed separately.
39.	TSYS1	Updates the Right Hand Side (R.H.S) and main diagonal coefficients at each new time step.
40.	WHEAD	Writes unformatted full 3D array of heads for saturated flow.

No.	Subroutine	Function
41.	WPROBE	Writes values of heads at certain nodes (single transect plane).
42.	WRIT10	Writes initial variables and certain data in a formatted output file (OUT10).
43.	WRIT11	Writes data and computed variables at certain time instants or time steps to an output file. 3D arrays are not included.
44.	WRIT22	Writes formatted data of 3D array "AH" (Heads)

Not all the subroutines are used in a given execution of the code. Figure 4-1 shows a schematic flowchart of the BIGFLOW code.

4.3 BIGFLOW EXECUTION PROCEDURE

A brief description of the recommended procedures necessary to run a BIGFLOW simulation is given below. More details on I/O files will follow.

STEP 1 - Prepare basic input file (INPUT1):

Run DATAFLOW (select INPUT1 from menu) to create the basic input file required for the BIGLOW code. The generic name of this data file is INPUT1 (see below for naming conventions).

STEP 2 - Prepare other input files (multi-dimensional datasets):

Find out which other input files are needed. Information on this will appear on the screen when DATAFLOW prompts for the names of the input files. The random field generator, CTURN may be used if the conductivity field is random. The generic name INPUT2 is reserved for the 3D initial condition file, INPUT3 is for the planar boundary condition file, and INPUT4 through INPUT9 are for 3D hydraulic property files. Only some of these data files may be needed, depending on the particular simulation at hand. For instance, none of the data files INPUT2 through INPUT9 would be needed for a flow problem with uniform initial condition, uniform boundary condition on each face of the domain, and spatially constant hydrodynamic parameters (material properties). On the other hand, if, for example, the conductivity K_x is spatially variable, then the actual file name of the generic data file INPUT5 must be specified correctly, for example, KSAT5. The conductivity file KSAT5 may be generated either with DATAFLOW, or, if random, by using the CTURN code.

STEP 3 - Input file name conventions:

Once all the required input data files have been created, their actual file names should be changed by the user to reflect the particular flow problem being simulated. This will avoid confusion with previously created data files. The actual file names, as chosen by the user, must then be specified inside the basic INPUT1 file, along with the other input variables specified in that file. The INPUT1 file itself may have a different name, for example, the user may choose to name this file IN1-TEST999 rather than INPUT1. However, at run time, it is necessary that the INPUT1 file's actual name be INPUT1,

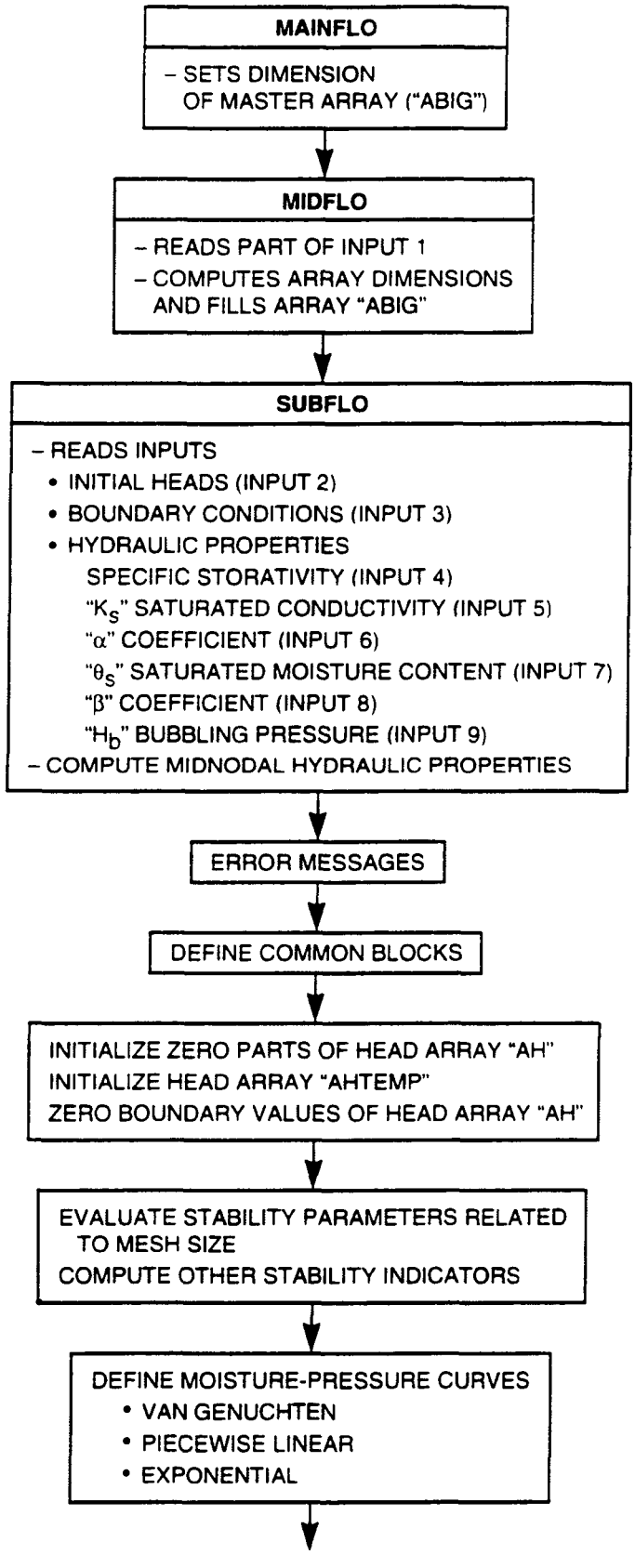


Figure 4-1. Schematic flowchart of BIGFLOW

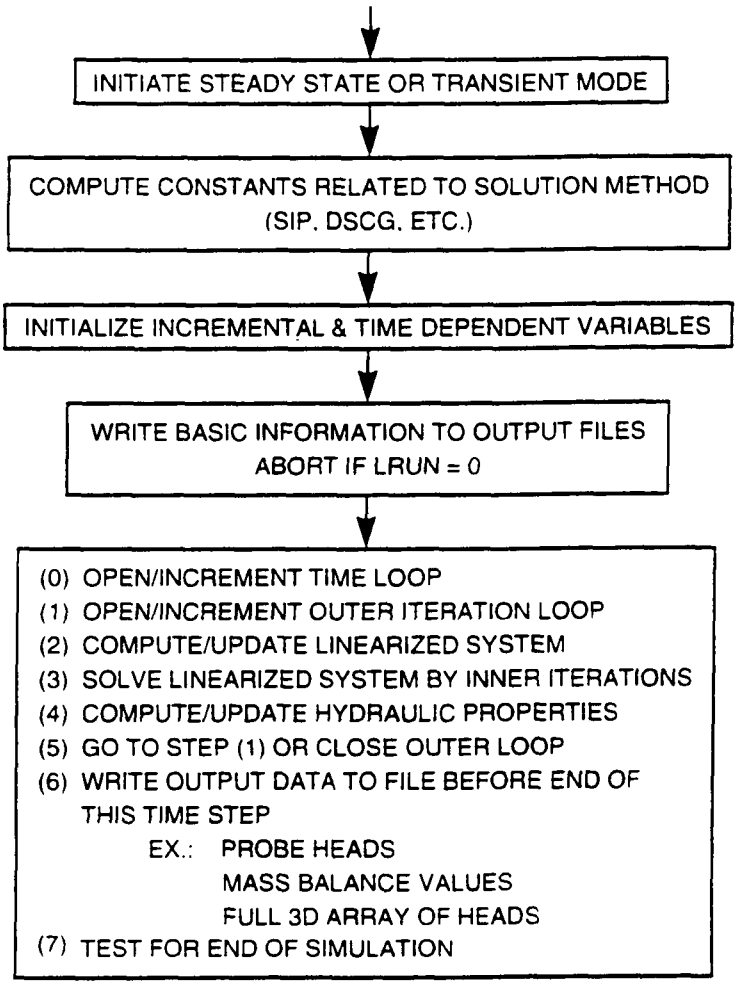


Figure 4-1 (Cont'd). Schematic flowchart of BIGFLOW

since BIGFLOW automatically searches for a file named INPUT1. Alternatively, system commands could be used to assign the actual file name of INPUT1 to the logical name "INPUT1" just before running BIGFLOW.

STEP 4 - Run BIGFLOW:

Run BIGFLOW either in "batch mode" or "interactive mode".

NOTE #1: For complex problems, it may be useful to run first with the flag LRUN=0 as a short preliminary test. Look at the output files OUT10 and OUTBAD to check for parameters computed by the code, then, if all is as expected, run a full simulation with flag LRUN=1.

NOTE #2: BIGFLOW's numerical outputs, as well as inputs, can be large. To reduce the size of BIGFLOW's 3D datasets, both inputs and outputs, choose the "unformatted" option by setting the flag LUNF to LUNF=1. Unformatted files are about four times more compact, but they are not usually portable to different computer systems, unlike formatted (ASCII) files. If portability is important, choose LUNF=0 for "formatted" files.

STEP 5 - Look for possible error messages:

At run time, two categories of errors may occur: (i) run time errors detected by the operating system (such as overflow); and (ii) errors or data conflicts detected by the BIGFLOW code itself. Error messages from the BIGFLOW code will appear in a file named OUTBAD, which is overwritten at each new simulation whether there is a BIGFLOW error or not. If there are no such errors, the message will be of the form "BIGFLOW errors:...none." On the other hand, a detected error or data conflict will cause a different message to be printed in OUTBAD, and will also cause premature end-of-execution. Note that this may occur even in the case of a short test run with the flag LRUN=0. If a BIGFLOW error occurs while LRUN=0, the error is probably due to incompatible or nonexistent input file(s). More generally, the possible errors that can be detected by BIGFLOW are of three types:

1. Insufficient dimension of the master array ABIG (main program MAINFLO).

2. Incompatible or erroneous data in input file INPUT1; the error code LBAD=-n will help locate the error (see Appendix A for a listing of various error descriptions).

3. Erroneous or nonexistent input data file INPUT2 through INPUT9. The error code LBAD=-n will help locate the problem; in particular the 1st digit of n indicates which input file was at fault. For instance, n=-52 would indicate a problem with the input file whose generic name is INPUT5 (3D saturated conductivity field). For example, if the actual file name for INPUT5 is KSAT5, then the error may be due to KSAT5 being nonexistent in the current directory (perhaps with the wrong name, too short/too long, or with an incorrect format).

STEP 6 - Examine and rename output files:

The 3D head solution is stored in file HEAD_T0 for steady state flow problems, in files HEAD_Tn or H_Tnnn for transient flow problems. These files can be formatted (ASCII) or unformatted (binary). In the transient case, the head values at certain locations may also be given at each time step (all time steps) in the mass balance file OUT13 discussed below.

The post-processing routines of the DATAFLOW code can be used to obtain other outputs such as 3D flux and 3D head gradient vectors, transects or plane sections of 3D fields, and also statistical properties. Basic data and numerical information such as iteration errors will be found in the output files OUT10, OUT11, and OUT12, depending on the case at hand. All output files should be renamed to avoid confusion with previous simulation outputs. It is advised to incorporate the generic names above as part of the new names, for example, rename OUT13 as "OUT13-TEST999" for test run 999.

4.4 BIGFLOW INPUT/OUTPUT FILES

Most of the inputs/outputs of the BIGFLOW code can be processed interactively by using the companion code DATAFLOW. The special-purpose data processor DATAFLOW can be used to: (i) create the basic BIGFLOW input file describing the nature of the flow problem and a number of options; (ii) create other BIGFLOW inputs that require two- or three- dimensional data fields; and (iii) process 1D, 2D, and 3D datasets in a number of ways. For instance, one may generate a 3D flux field from 3D head and conductivity fields, perform various statistical analyses of 3D datasets, and extract lower-dimensional datasets along transects or slices.

The generation of random field parameters can be handled independently by the 3D CTURN. The formats used in the latter code are indeed compatible with those used in BIGFLOW and DATAFLOW. This means that a random conductivity field generated with the CTURN code can be processed by DATAFLOW for statistical analysis (this is recommended), and can then be used as input to BIGFLOW for conducting stochastic flow simulation(s). Furthermore, DATAFLOW also contains a routine to interactively rescale the random conductivity field, for example, for modifying the log-conductivity standard deviation, or the geometric mean conductivity, or both.

The BIGFLOW code itself is noninteractive, and can, therefore, be run as a batch process as well as directly from the console. The required input files have the logical names INPUTj (j=1,...9). The "basic" input file is INPUT1. This short file contains the chosen names of all the other input files, as well as the basic description of the flow problem and a number of numerical options. It is recommended that the DATAFLOW code for creating INPUT1 interactively be used. The task is particularly easy when it is only required that an existing INPUT1 file be slightly modified . The remaining input files may be created using either DATAFLOW and/or the CTURN code. For example, one may use DATAFLOW to create a 3D conductivity field corresponding to a layered or block-structured medium; one may alternatively use the CTURN code to generate a 3D conductivity field for a randomly heterogeneous medium; and one may even use DATAFLOW to superimpose layers or blocks of specified conductivity onto an existing random field output of the CTURN code (or any other BIGFLOW-compatible 3D dataset for that matter).

Some of the numerical outputs from BIGFLOW can be quite large, but there are also two small output files created at the start of execution: (i) one file containing basic information on the simulation

parameters (OUT10); and (ii) a short error file (OUTBAD) which may show an error message if trouble was detected by BIGFLOW. On the other hand, output files like OUT11 and OUT12, which contain purely numerical information (residual iteration errors), can be long depending on the total number of iterations. File OUT13 contains output data that needs to be issued at each time step, such as mass balance information from which mass balance errors may be assessed, and head values at selected locations (the "probe" option). The 3D field of hydraulic head or pressure head can be very large, and is only stored at selected times in the case of transient simulations. For steady flow, there is only one 3D head output, stored in a data file named HEAD_T0. For transient flow, there may be a number of 3D head outputs issued at selected times chosen by the user. These outputs are stored in data files named either HEAD_Tn or H_Tnnn, where n or nn or nnn represents an integer with, at most, three digits ($0 \leq nnn \leq 999$). In case BIGFLOW has trouble with this numbering scheme, or exhausts all the available digits, then a "mystery file" named HEAD_TX may be created.

The format of I/O files containing 3D datasets can be either formatted (ASCII) or unformatted (binary). In order to save space on the file storage system, it is recommended to use the latter option in the case of large simulations (however, binary data files are not easily transportable among different operating systems). Also, it is advisable to rename all output files after completion of each BIGFLOW job, since the code always uses the same generic file names for its outputs. This is true also for the data processor code DATAFLOW, for example, when used to postprocess the numerical 3D head field. The result of such processing may be the creation of a 3D flux vector field from the existing 3D head and conductivity fields, the extraction of 2D plane sections of these fields, 1D, uni-directional covariance functions obtained by spatial moment analysis, etc. In some cases, the resulting data files are given generic names and may have to be renamed for clarity. But, most often, the DATAFLOW code will prompt the user for a file name.

Figure 4-2 shows the general interrelationship between the I/O files of BIGFLOW.

4.4.1 BIGFLOW Input Files

All the input files of the computer code BIGFLOW are listed and described in Table 4-1.

4.4.2 Description of Variables in File INPUT1

In the following, all the basic input variables needed for conducting a BIGFLOW simulation are listed and explained. These variables are encountered by a user while directly editing an INPUT1 file, or during an interactive session with DATAFLOW while creating a new INPUT1 file, or modifying interactively an existing INPUT1 file. Note that the names of all other input datasets required by BIGFLOW (if any) are themselves considered as basic input variables, to be specified in file INPUT1.

Variable	Explanation
IDATE	Date (integer with six digits: day-month-year)
IDRUN	Identification number for the run (at most six digits)
LRUN	Set this flag to 1 if actual simulation is required

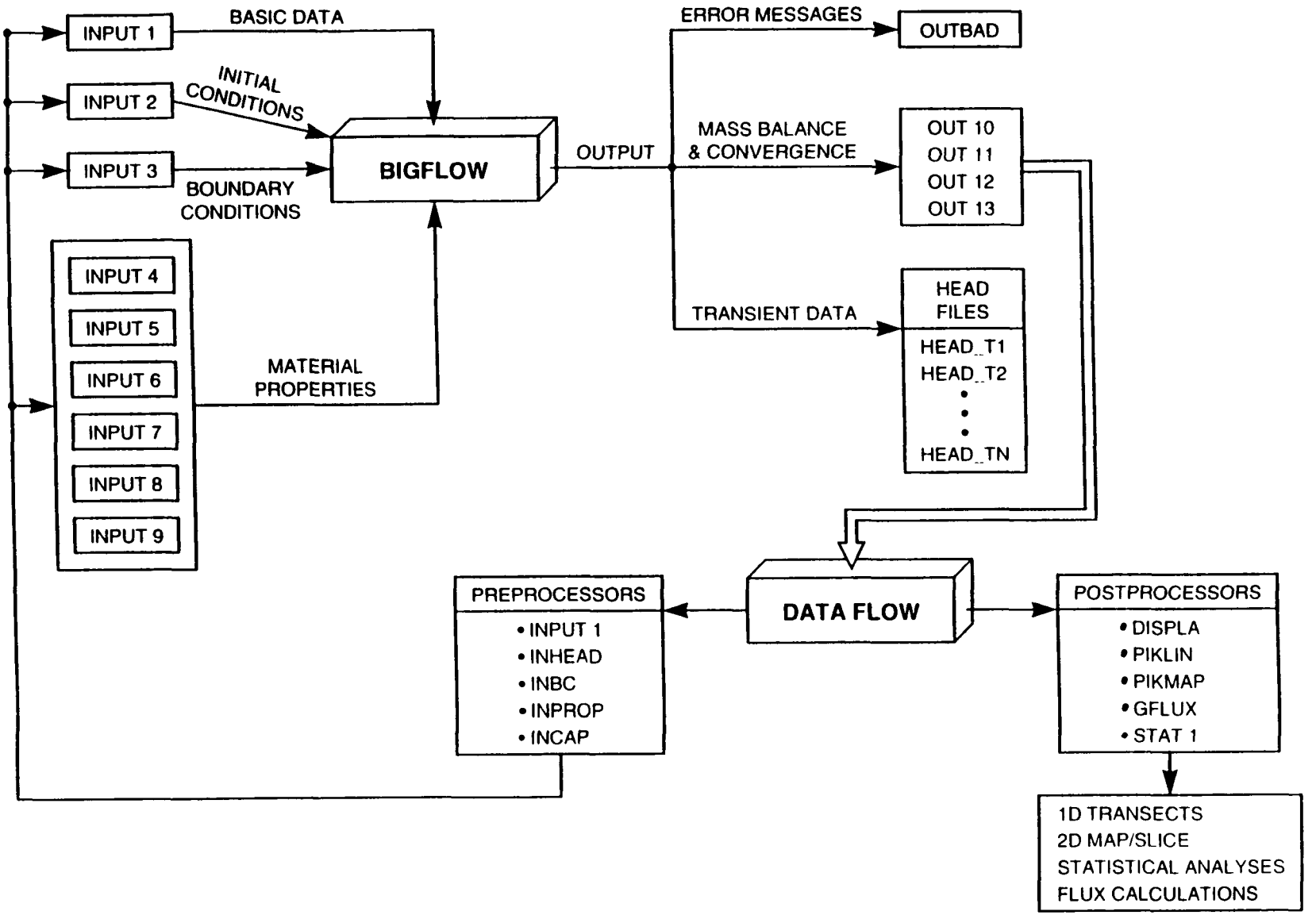


Figure 4-2. General interrelationship between the input and output files of BIGFLOW

Table 4-1. List of BIGFLOW input files

Input File	Description
INPUT1	Basic data
INPUT2	Initial heads
INPUT3	Boundary condition type and actual value
INPUT4	Specific storativity (S_s)
INPUT5	Nodal saturated conductivity (K_s)
INPUT6	Slope of the unsaturated lnK(h), h relation (α)
INPUT7	Volumetric water content at saturation (θ_s)
INPUT8	Shape coefficient or slope in the θ(h) retention curve (β)
INPUT9	Bubbling pressure head, for unsaturated conductivity and/or water retention curves (h_b)

LUNF Flag for unformatted input/output data files (LUNF=1). If LUNF = 0 then input/output data files are all formatted

GRID SIZE

NGRID1,2,3 Total number of nodes along the X, Y, and Z axis, respectively. This includes fixed head nodes, and also exterior nodes for flux or gradient boundary conditions (i.e., boundary nodes as well as interior nodes).

MING1,2,3 This is needed only if one or more boundaries will be moving during the (TRANSIENT) flow simulation. The rules are:

(1): MINGj less than or equal to NGRIDj
(2): MINGj greater than or equal to 4
(3): MINGj odd(even) if NGRIDj odd(even)

If both opposite faces (Aj,Bj) are fixed, then MINGj will automatically be set equal to NGRIDj by the BIGFLOW code

PARAMETERS FOR ACTIVATION ALGORITHM

MDELTj — Node-size of the region adjacent to the boundaries, where the flow code searches for pressure changes with respect to the initial pressure head

HDELT — Absolute value of pressure head difference, used as decision criterion to activate boundaries

LDELT — Integer code for choosing the kind of norm to be used for computing the typical pressure difference Norm(H-Hin) within search zone : LDELT=2 or <2 : Mean-square norm LDELT=3 or >3 : Absolute Maximum norm

GRAV 1,2,3 — Orientation of axes relative to gravity

BOUNDARY CONDITION TYPE ON EACH FACE

LTYPA(J), LTYPB(J) — Boundary condition type on each face: For J = 1 through 3. The index J corresponds to the coordinate axis (for example J =1 along X axis, J = 2 along Y axis, and J =3 along Z axis). Therefore, LTYPA(1) describes the Y-Z plane at X1 = 0, and LTYPB(1) describes the Y-Z plane at X1 = XL1 (maximum node along the X direction).

= 0 if activated boundary
= 1 if uniform fixed head
= 11 if nonuniform fixed head
= 2 if uniform fixed flux
= 22 if nonuniform fixed flux
= (-)12 if mixed head-AND-flux

Minus sign if flux condition is more severe than head condition, 3 if zero pressure head gradient

For example:
ENTER :
LTYPA(1), LTYPB(1) ...FOR X1=0., X1=XL1, where XL1 is the maximum node along the X direction
LTYPA(2), LTYPB(2) ...FOR X2=0., X2=XL2, where XL2 is the maximum node along the Y direction
LTYPA(3), LTYPB(3) ...FOR X3=0., X3=XL3, where XL3 is the maximum node along the Z direction

VALUE OF FIXED HEAD OR FLUX ON EACH FACE

FIXAj, FIXBj — Value of fixed head or flux on each face. If one or more faces have nonuniform boundary conditions, then all the values will be disregarded as boundary conditions (they will be given in a separate data file INPUT3). However, if this

is the case, one should still give typical values of pressure or flux, as these will be used to compute stability parameters. Also, use FIXaj=HIN for the moving boundaries.

For example:
ENTER:
FIXA1, FIXB1 ...FOR X1=0., X1=XL1
FIXA2, FIXB2 ...FOR X2=0., X2=XL2
FIXA3, FIXB3 ...FOR X3=0., X3=XL3

Figure 4-3(a) illustrates LTYPA(J) and LTYPB(J) boundaries with FIXA(J) and FIXB(J) (J=1,2,3) boundary values. Figure 4-3(b) depicts the location of planes A and B, relative to the origin of the coordinate system.

MESH SIZE

DX1, DX2, DX3 — Mesh size if LGRID = 1, domain size if LGRID=2

FLOW REGIME AND VARIOUS OPTIONS

LFLOW
= 1 : saturated (hydraulic head-based equations)
= 2 : partially saturated/unsaturated (pressure head-based equations)
= 3 : steady unsaturated flow (Kirchoff transform-based equations)
(option 3 not available)

LTRANS
= 0 : steady state flow
= 1 : transient flow

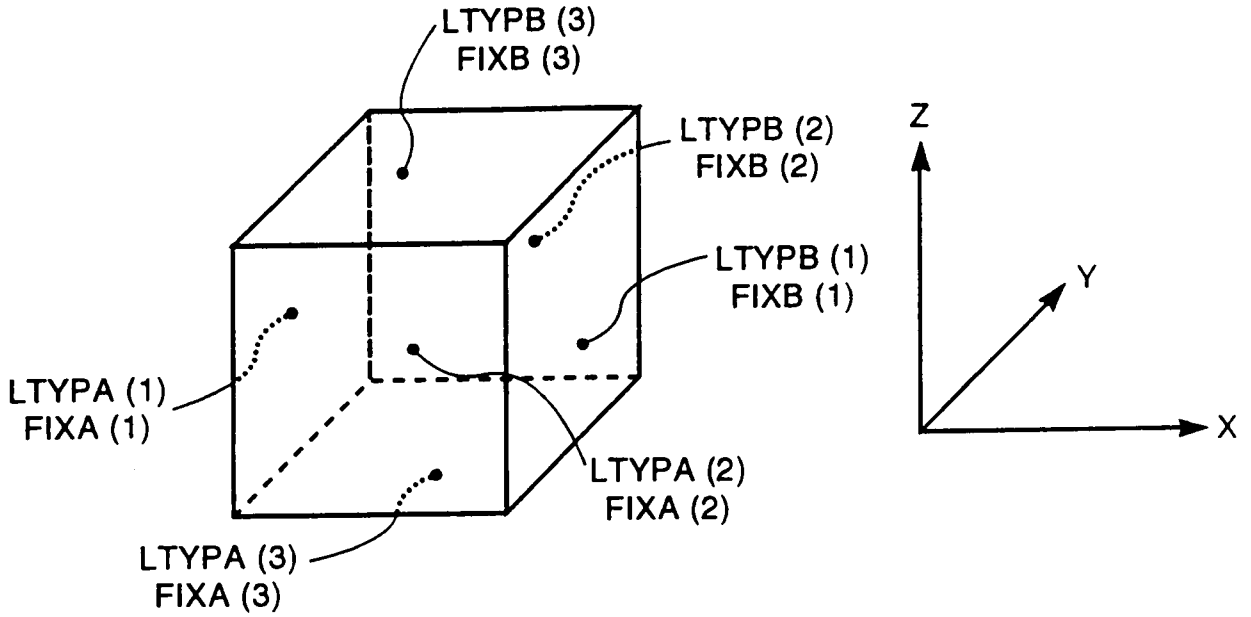
LKWNOD
Option for the scheme used in computing midnodal saturated conductivities, given the nodal values (interpolation scheme)
LKWNOD = 1: use geometric mean
= 2: use harmonic mean
= 3: use arithmetic mean

LKWNOD=1 is recommended for most applications of saturated flow; this has no influence on unsaturated flow simulations

LHIN
= 0: initial head is spatially uniform
= 1: initial head is arbitrarily non-uniform
= 10: linear profile along the X1 axis
= 20: linear profile along the X2 axis
= 30: linear profile along the X3 axis
WARNING: Head=hydraulic head for saturated flow; head=pressure head for unsaturated flow

HIN — Initial head if LHIN=0 above, or else some reference head value if LHIN=1 or 10 or 20 or 30 above

a)



b)

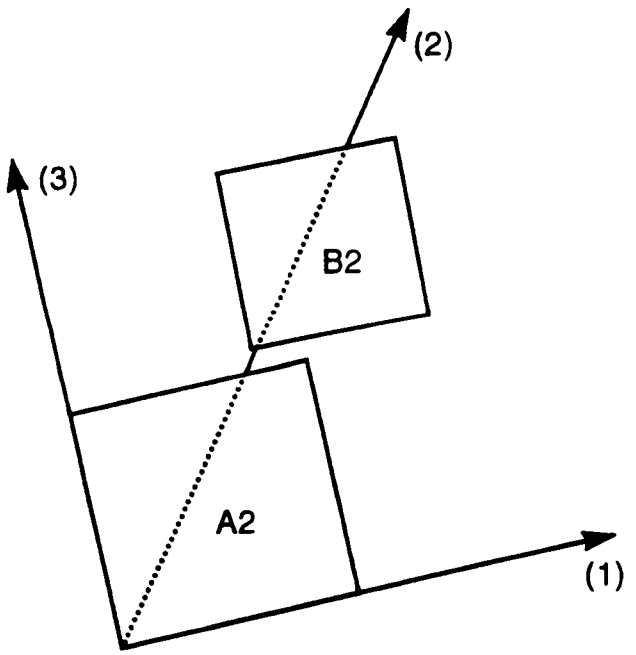


Figure 4-3. Schematic depicting: (a) the LTYPA(J) and LTYPB(J) boundary planes, and (b) the location of planes relative to the origin

HYDRAULIC PROPERTIES/STATISTICS

NOTE: The porous medium hydraulic properties are assumed *a priori* to be defined as random variables or random functions. However this interpretation is not necessary; for those properties not random, use relevant values as explained below.

Mean	Value of the data (if constant in space) or some reference value (if variable)
Standard Deviation	Zero (if constant data), else some measure of variability (maximum amplitude, etc.)
Correlation Scale	Zero (if constant data), else some fluctuation scale or wavelength
FKGM	Geometric mean
FKDEV	Standard Deviation of ln(K)
FKL1, FKL2, FKL3	Three Correlation Scales of ln(K)

SPECIFIC STORATIVITY (TRANSIENT SATURATED FLOW)

NOTE: This property can also be used as the specific storativity for unsaturated flow; in this case it must be constant (CAPDEV=0) and will only be turned on for regions with positive pressures.

CAPGM	Geometric mean of specific storativity
CAPDEV	Standard Deviation of specific storativity
CAPL1, CAPL2, CAPL3	Three correlation scales of specific storativity

HYDRAULIC PROPERTIES FOR UNSATURATED FLOW

ALFGM	Mean of coefficient α of the lnK(h) relationship (unsaturated flow regime)
ALFDEV	Standard deviation of α
ALFA1, ALFA2, ALFA3	Three correlation scales of α
TTGM	Mean of saturated volumetric water content or porosity (unsaturated flow only; not required if steady state)

TTDEV	Standard deviation of saturated volumetric water content or porosity (unsaturated flow only; not required if steady state)
TTL1, TTL2, TTL3	Three correlation scales of saturated volumetric water content or porosity (unsaturated flow only; not required if steady state)
BTGM	Mean of β coefficient of the water retention curve (unsaturated flow only; not required if steady state)
BTDEV	Standard deviation of β coefficient of the water retention curve (unsaturated flow only; not required if steady state)
BTL1, BTL2, BTL3	Three correlation scales of β coefficient of the water retention curve (unsaturated flow only; not required if steady state)
HBB	Air entry or "bubbling" pressure head (for lnK(h) and θ(h) curves; unsaturated flow only)

OTHER INPUTS FOR UNSATURATED SOIL PROPERTIES

LTHETA	= 10 ------> Piecewise linear = 20 ------> Exponential curve, with cut-off at h=HBB = 30 ------> Van-Genuchten curve = 31 ------> Dek and Dieri soils = 32 ------> Dek and Montfavet soils (LTHETA is not needed in steady state and/or saturated flow)
TTDRY	Dry residual water content (real constant) (TTDRY is not needed in steady state and/or saturated flow)
VGN	Van-Genuchten "n" parameter (real constant) (VGN is not needed in steady state and/or saturated flow)
HSMALL	Small pressure head difference used to compute unsaturated soil moisture capacity; should be small enough for a good finite difference approximation of the derivative of θ(h). (HSMALL is needed only for unsaturated flow)

TIME-STEP AND RELATED DATA ON FLOW DYNAMICS CASE OF TRANSIENT FLOW

DTIN	Initial time step
DTMIN	Minimum time step
DTMAX	Maximum time step

DTMUL — Time step multiplier for a geometric increase of time step with respect to time. Use $1 < DTMUL < 2$ for unsaturated flow, or even more for saturated flow; taking $DTMUL < 0$ will indicate another version of the time step algorithm as described in the following. The time-stepping algorithm is based on the rate of change of heads compared to the largest absolute head difference DHSTAB (computed from boundary and initial conditions), as follows:

RMUL = ABS(DTMUL) ;
DTB=DTIN*DHSTAB/MAX(H(K+1)-H(K))
DT(K+1)=Min(RMUL*DT(K) , DTB) if DTMUL > 0
DT(K+1)=RMUL*Min(DT(K) , DTB) if DTMUL < 0

DTIN — The 1st time step unless it is zero, in which case the 1st step is computed by the code so that it be very (very) small. For steady state flow, DTIN is irrelevant.

DTMIN — Minimum 1st time step. For steady state flow, DTMIN is irrelevant.

DTMAX — Maximum 1st time step. For steady state flow, DTMAX is irrelevant.
NOTE: DTIN = DTMIN = DTMAX will force the code to use a constant time step, except for the 1st one. (This is not recommended).

OTHER DATA ON NUMERICS AND FLOW DYNAMICS

DHSTAB — Reference value of head difference, used to compute time-step according to: DT(K+1)=DTIN*DHSTAB/MAX(H(K+1)-H(K)). This value may be over-ridden by the code if it is found to be too small.

CONDMA — Approximate maximum of saturated or unsaturated conductivity

SCAPMI — Approximate minimum of either specific storativity (for saturated flow) or soil moisture capacity (for unsaturated flow)

LINEAR SYSTEM SOLVER (INNER ITERATIONS)

LSOLV — Choice of the linear system solver. Two main types of preconditioned iterative solvers are available: SIP solvers, and Conjugate Gradients (CG) solvers.

- = +-110: "SIP1" solver - standard ordering: (larger storage; smaller CPU time if constant iteration parameter; larger CPU time if cyclic iteration parameter)
- = +-120: "SIP2" solver - standard ordering; (smaller storage; larger CPU time)
- = +-121: "SIP2" solver - alternate ordering; (smaller storage; larger CPU time) +-200: "CG" and "DSCG" conjugate gradients solver; (minus sign for pre-diagonal scaling recommended)

- = +210: "ICFAC+PCG" solver: Incomplete Choleski and Conjugate Gradients

- = +220: "MICFAC+PCG" solver: Modified Incomplete Choleski (IC) and Conjugate Gradients. Examples: +120 (standard SIP), -200 (diagonally scaled CG)

LNORM

Controls computation of error norm in both the linear and nonlinear iteration loops (respectively ERNORM and ENL):
0: do not compute ERNORM
-1,-2,-3,-4,-5: compute ERNORM only at the last iteration step
+1,+2,+3,+4,+5: compute ERNORM each iteration step, and apply the ERMIN convergence test
+-1: L1 norm (mean absolute value)
+-2: L2 norm (mean-squared norm)
+-3: L-infinity norm (absolute maximum)
+-4: L2 and L-infinity both computed, with L2 used for computing ERMIN
+-5: L2 and L-infinity both computed, with L-infinity used for ERMIN.
If LNORM > 0: ERNORM compared to ERMIN to end iterations
If LNORM= < 0: ERNORM not used as stopping criterion
RECOMMENDED: +4 (convergence test with quadratic norm); even safer: +5 (convergence test with infinity norm)

ERMIN

Admissible norm-of-error (convergence criterion)

ERMAX

Maximum allowed error (more under relaxed or exit) (must be given in head units)

ERMIN

Minimum allowed error (must be given in head units)

ERMIN > 0 ---> compared to error at each iteration
ERMIN < 0 ---> ABS(ERMIN) is compared to max error over 1 SIP cycle (i.e., over "MIT" iterations)
ERMIN = 0 ---> ERMIN is ignored (same for ERMAX)
LNORM < 0 ---> ERMIN is also ignored in this case
HINT: Take ERMIN < 0 for a stringent convergence criterion, take ERMAX=0 to avoid max_error stopping

ITEND

Maximum number of inner iterations for each outer iteration (OR EACH TIME STEP). Typically: 10-50 transient; 100-1000 steady; maybe more for large steady state problems.
NOTE: The actual number of iterations will always be equal to "ITEND" if LNORM < 0 or if ERMIN is very small.

LPEPIT

Option for the SIP iteration parameter:
LPEPIT=0: PIT is constant over iterations
LPEPIT=1: PIT is periodic over iterations

RECOMMENDED: LPEPIT=1

MIT — Integer period of cyclic iteration parameter:
MIT=0 if constant iteration parameter (LPEPIT=0)
MIT=4 recommended, and no more than 8 (LPEPIT=1)
Also note: for alternate-SIP, actual cycle is 2×MIT

PITMAX — Maximum value of the SIP iteration parameter (or its constant value, if not cyclic):
PITMAX=Real number between 0.0 and 1.0; however, a zero value indicates that PITMAX will be computed by the code
RECOMMENDED: Pick value close to one, or better take PITMAX=0.0 to let the code do the job

SIP SOLVER(S) RELAXATION PARAMETER

UNLAX — BLAX(1); where the formula for BLAX(M) for non-stationary relaxation is:
BLAX(M) = UNLAX+[OVLAX-UNLAX]*[1-EXP(-VARLAX*(M-1))]; M is the iteration count

OVLAX — BLAX(M) AS M-->0

VARLAX — The rate of change of BLAX(M) (half the rate if alternate-SIP is used)

Notes on the choice of solver relaxation:

- For neutral iterations (no relaxation): UNLAX=1., OVLAX=1., VARLAX=0

- For stationary under-relaxation, say BLAX=0.5: UNLAX=0.5, OVLAX=0.5, VARLAX=0

HINT: For a first try, do not relax (enter: 1,1,0). If solver diverges, under-relax (e.g., 0.5, 0.5, 0). Avoid too small under relax: keep under-relax > 0.1

NONLINEAR SYSTEM SOLVER (OUTER ITERATIONS)
CASE OF UNSATURATED FLOW

ENLMIN — Admissible norm-of-error (convergence criterion)
ENLMAX — Maximum allowed error (more under-relax...or exit)
NOTE: ENLMIN must be >0, given in head units (same for ENLMAX), ENLMIN will be ignored if LNORM < 0 (given above), ENLMIN will also be ignored if zero (same for ENLMAX)
HINT: Let ENLMAX be ignored (ENLMAX=0.)

INLMAX — Maximum number of outer (nonlinear) iterations for each time step; Typically: 10-50 transient; maybe more for steady state problems. Note that the actual number of iterations will be equal to "INLMAX" if LNORM < 0 or if ENLMIN is very small.

INLBC — Number of additional iterations to update just the heads at boundaries; this may not work at all (set INLBC=0 in most cases).
HINT: Set INLBC=0

RELAXATION OF NONLINEAR ITERATIONS FOR UNSATURATED FLOW

HNLAX — Relaxation factor for the head increment
RNLAX — Relaxation factor for the Right Hand Side residual
IMPORTANT: The flow code requires at least one of these two parameters to be equal to 1. This is because they are just different ways of doing the same thing (HNLAX ex-post; RNLAX ex-ante).

Other rules:

- HNLAX, RNLAX = 1. for neutral scheme (no relax.)
- 0 < HNLAX, RNLAX < 1. for under-relaxation
- 1 <HNLAX, RNLAX < 2. for over-relaxation
- HNLAX, RNLAX zero or negative for automatic control of relaxation: if zero, the code finds the starting value; if negative, the code uses the absolute value as a starting value.

HINT: Let HNLAX=1. and RNLAX=1. for a first try. If divergence occurs, try RNLAX=1/2,1/4, or less.

OUTPUT OPTIONS FOR TRANSIENT SIMULATIONS

KTMAX — Maximum number of time steps (end-of-simulation):
- For steady state flow, the code automatically sets this value to be 1, regardless of what is entered.
- Let KTMAX=0 if no limit on the total number of time steps (the code will disregard KTMAX).

TYMAX — Maximum time in physical units (end-of-simulation):
Note that TYMAX limits the duration of simulation in terms of simulated time, while KTMAX limits actual computer time. This parameter is ignored in case of steady state flow.

LTOUT — Flag for outputs at specified time instants
LTOUT=1 for "YES" , LTOUT=0 for "NO"

KTOUT — Number of 3D head outputs to be delivered at given times; must be less than 100 but the number of outputs is actually not limited; take KTOUT=1 for steady state flow.

TYMOUT(k) Output times, k=1 through KTOUT ; Periodic output: The last value of TYMOUT is assumed to be the period for further outputs.
For example, to get an output with period "PPP":
1 [Return]
PPP [Return]

PERIODICITY OF OUTPUTS

KOUT10 Basic information (no arrays) for each time step, delivered every "KOUT10" time steps, in file "OUT10"(about 1 page/step)

KOUT11 Basic information (no arrays) for each inner iteration, given every "KOUT11" iteration steps, in file "OUT11"

KHOUT 3D array of heads provided every "KHOUT" time steps; usually, let KHOUT=0 for transient flow. However, selecting outputs at given time instants are to be preferred: recall the LTOUT flag defined previously in this session.

Details on the choice of "KOUT10, KOUT11, KHOUT":
1,1,1 --> for steady flow with one 3D output
1,1,0 --> for steady flow without 3D output
0,0,0 --> for steady flow without 3D output and minimal simulation information
1,1,0 --> for transient flow without 3D output every "K" time steps, and with maximal simulation information
0,1,0 --> for transient flow without 3D output every "K" time steps, with minimal simulation information per t-step, and with full iterative solver output (note that this concerns only outputs at given steps; outputs at given times are obtained with flag LTOUT=1 above).
Note that setting Kxxx=0 disables that output option.
HINT: Always use KHOUT=0 for long TRANSIENT runs; the 3D heads can be written in output files at some fixed times rather than time-steps. Let KHOUT=1 to obtain 3D heads in steady state flow

FORM OF OUTPUTS, IN PARTICULAR 3D HEAD OUTPUT

LFORM1 1 (OBSOLETE: OUT10 always formatted)

LFORM2 1 (OBSOLETE: OUT11 always formatted)

LHOUT Form of the 3D head outputs (if any):
-1: for no head output at all (this will override everything)

0: for a small formatted head output, limited to a (10×10×10) window (use only for testing purposes)

1: for a complete 3D head output, (unformatted/formatted file if LUNF=1/0)

HINT: For full normal results, just enter: 1,1,1 for LFORM1, LFORM2, and LHOUT

PROBE THE HEAD SOLUTION AT CERTAIN LOCATION

IPROB1,2,3 — Location of the node, line, or plane

This in effect monitors the value of the head field at certain nodes, with output delivered every time step for these nodes. The options are:

- no probe
- probe a single node (or none)
- probe a line of nodes (along X_j)
- probe a plane of nodes (along X_j, X_k)

For example:
0,0,0 -----> no probe (this cancels probe output)
i,j,k -----> one node, at location (I1=i, I2=j, I3=k)
0,j,k -----> one line, parallel to X1, located at (I2=j, I3=k)
0,0,k -----> one plane, orthogonal to X3, located at I3=k
NOTE: Indices (i,j,k) run from 0 or 1 through NGRIDj (0="undefined"); exclude boundary nodes 1 & NGRIDj

HINT: If not sure what to do, just type 0,0,0

SPECIAL OPTIONS

LBFLUX — Compute boundary fluxes (YES=1, NO=0)

LMASS — Compute mass balance (YES=1, NO=0)
NOTE: Total mass (LMASS=1) computed only for case of transient unsaturated flow; the boundary fluxes (LBFLUX=1) are computed for any kind of flow; for transient unsaturated flow, the code assumes LBFLUX=1 and also implies LMASS=1 and vice-versa. Outputs will be stored in file "OUT13".

SYSTEM-DEPENDENT CONSTANTS

SRELPR — 10 × (Single relative precision)

RSMALL — 100 × (Smallest real number)

INPUT2 — Name (not to exceed 10 characters) for INPUT2 file

INPUT3 — Name (not to exceed 10 characters) for INPUT3 file

INPUT4 — Name (not to exceed 10 characters) for INPUT4 file

INPUT5 — Name (not to exceed 10 characters) for INPUT5 file

INPUT6	Name (not to exceed 10 characters) for INPUT6 file
INPUT7	Name (not to exceed 10 characters) for INPUT7 file
INPUT8	Name (not to exceed 10 characters) for INPUT8 file
INPUT9	Name (not to exceed 10 characters) for INPUT9 file

4.4.3 BIGFLOW Output Files

Basic data and numerical information such as iteration count and errors will be found in the output files OUT10, OUT11, and OUT12, depending on the case at hand. Table 4-2 summarizes the output files of BIGFLOW.

4.5 INTERACTIVE DATA PROCESSOR DATAFLOW

DATAFLOW is BIGFLOW's companion code that enables a user to interactively create the input files for a BIGFLOW run. DATAFLOW creates the necessary input files for BIGFLOW during an interactive session with the user, in which the user responds to specific prompts. In addition, DATAFLOW also offers postprocessing capabilities of BIGFLOW 3D output data such as cell-by-cell calculation of 3D flux vectors, spatial statistical analyses, extraction of sub-dimensional data sets (1D transects or 2D cross sections), etc. DATAFLOW is written in standard ANSI Fortran 77 and has been successfully tested on a variety of computing platforms. At the user's option, the output of DATAFLOW can be either in ASCII or binary format.

4.5.1 Features and Functions of DATAFLOW

In the following, we describe the names and functions of the most important subroutines comprising the DATAFLOW code.

No.	Subroutine	Type	Function
1	INFLO1	Preprocessor:	Creates a formatted input file named "INPUT1" by interacting with the user. This routine also allows the user to modify an existing "INPUT1" file.
2	INHEAD	Preprocessor:	Creates a 3D array of nodal values of heads (hydraulic or pressure head). This array must have the same dimensions as the head array (AH) i.e, NPP1 × NPP2 × NPP3, where NPP1 = N1 + 2, NPP2 = N2 + 2, NPP3 = N3 + 2, where N1, N2, and N3 are the number of interior nodes along the three othogonal directions.

Table 4-2. List of BIGFLOW output files

Output File	Description
OUTBAD	Error file containing errors detected in the input files or BIGFLOW warnings
OUT10	Summary of Simulation Results
OUT11	Matrix Solver iterations
OUT12	Nonlinear iterations
OUT13	Probe Heads versus time at given space locations
HEAD_T0	Full 3D Head Arrays HEAD_T0: for steady state flow HEAD_Tnnn (nnn < 100): for transient flow

No.	Subroutine	Type	Function
3	INBC	Preprocessor:	Creates 3D arrays for the boundary conditions on each of the 6 faces of the 3D grid. There are six arrays for the type of boundary condition (BC), and six other arrays for the corresponding value. Table 4-3 summarizes the boundary condition type and the corresponding value.
4	INPROP	Preprocessor:	Creates 3D array of nodal values of saturated conductivity or any other hydraulic property, except specific storativity. The following files are created:
		INPUT5 -->	Saturated conductivity (K_s)
		INPUT6 -->	α parameter in lnK(h) of the unsaturated curve
		INPUT7 -->	Saturated water content (θ_s)
		INPUT8 -->	β parameter in water retention curve
		INPUT9 -->	Bubbling pressure (h_b)
5	INCAP	Preprocessor:	Creates a 3D array of nodal values of capacities such as specific storativity. This array must have the dimensions N1×N2×N3, where N1, N2, and N3 are the number of nodes in each of the orthogonal directions.

Table 4-3. List of boundary condition types

BC Type	BC Value
TYP = 0 Activated boundary	FIX = HIN or arbitrary
TYP = 1 Fixed uniform head	FIX = Hydraulic or pressure head
TYP = 11[1] Fixed variable head	FIX = Hydraulic or pressure head
TYP = 2 Fixed uniform flux	FIX = Flux (specific discharge rate)
TYP = 22[1] Fixed variable flux	FIX = Flux (specific discharge rate)
TYP = 12[2] Fixed head and flux	FIX = Head or flux
TYP = 3	FIX = Non-applicable

NOTE #1 TYP = 11 comes with two options, one being to prescribe two distinct values of heads inside/outside a rectangle, the other being to require a linear variation of heads from one side of the rectangular face to the opposite side.

NOTE #2 TYP = 12 means that both types of boundary conditions may occur on the same face (fixed head at certain nodes, and fixed flux at others).

No.	Subroutine	Type	Function
6	DISPLA	Postprocessor:	Reads the header (basic information) of unformatted/formatted data files that are 3D inputs or outputs for BIGFLOW.
7	MODIFY	Preprocessor:	Modifies the contents of the conductivity file as created by INCOND or the turning band generator. A new file ("INPUT5") may be created by this routine.

No.	Subroutine	Type	Function
8	PIKLIN	Postprocessor:	Prints 3D results from BIGFLOW along user selected lines. One data file is created for each line. It also computes the arithmetic mean along each line (excluding the boundaries). This routine can extract 1D transects of 3D head data, 3D conductivity data, and 3D vectors of head gradients and flux. The 1D transects are stored in either a user specified file or a default file name as selected by the routine. Table 4-4 summarizes the functions of this routine.
9	PIKMAP	Postprocessor:	Selects and stores a 2D data slice taken along a plane from the 3D data field from BIGFLOW, such as 3D conductivity, 3D Head or flux. The name of the 3D file is provided interactively by the user. The 2D slice that this routine outputs will be stored in a user specified file or in the default file "MAP22".
10	GFLUX	Postprocessor:	Computes the gradient or flux of a scalar quantity in 3D space. The following table summarizes the inputs and outputs of this routine.
11	STAT1	Postprocessor:	Computes statistics of a scalar field (for example the 3D head field) or of one of the components of a vector field (for example, 3D flux field). The computed statistics are: (i) the mean; (ii) the global variance; and (iii) the correlation function along each of the three axes. It is assumed that the input represents a 3D head field. However, this could actually be any 3D array of nodal values, provided the input file has the appropriate format. If the input file is a vector field then a single component of this vector field is treated.

4.5.2 Generation of Input for BIGFLOW Using the Interactive DATAFLOW Code

All inputs (INPUT1 through INPUT9) can be generated using the interactive DATAFLOW code. Once the user invokes DATAFLOW, a DATAFLOW menu presents the following:

Preprocessors:

- INPUT1
- INHEAD
- INBC
- INPROP
- INCAP

Table 4-4. Functions of routine PIKLIN

From	To	Default File Name
3D Head	1D Head	PLOT21
3D Conductivity	1D Conductivity	PLOT51
3D Head Gradient	1D Head Gradient	PLOT71
3D Flux	1D Flux	PLOT81

Postprocessors:

- DISPLA
- MODIFY
- PIKLIN
- PIKMAP
- GFLUX
- STAT1

A concise description of each of the above selections can be obtained by entering "MENU" at the prompt. The user must first create the basic data file "INPUT1" for any BIGFLOW run. The data file may be created by entering "INPUT1" at the prompt. At this point, the user has the option to create a completely new "INPUT1" data file or modify an existing INPUT1 file. If the user chooses to modify an existing "INPUT1" file, the following rules must be observed:

- Read the definitions given on the screen
- Look at the values read from old INPUT1 file
- To change any of these values, type any character (except space bar or zero), for example "C" followed by a "RETURN". The new value may now be entered.
- To save old values, strike the "RETURN" key.
- Do not use more than 10 characters to specify the name of the "INPUT1" file.

At this point, DATAFLOW is ready to create/modify the basic data file "INPUT1". The user must now respond to a series of questions that define the variables needed for a simulation. The definition of each variable along with the choice of responses (where appropriate) is displayed on the screen. The user is referred to section 4.4.2 for a complete definition of all variables that constitute INPUT1.

The porous medium hydraulic properties are assumed *a priori* to be defined as random variables or random functions. The randomness of the hydraulic property is defined by its geometric mean, standard deviation and correlation scales along the three axes X, Y, and Z. In the case of spatially constant hydraulic data, the mean corresponds to the value of the variable, the standard deviation and the correlation scales must all be set to zero. In the case of spatially varying hydraulic data, the mean

corresponds to a reference value, the standard deviation represents a measure of variability (for example maximum amplitude) and the correlation scales represents a fluctuation scale or wavelength along each cartesian coordinate axis. This aspect is specially useful when creating a spatially varying, layered conductivity field. Routine "INPROP" must be invoked to create a spatially varying hydraulic property. "INPROP" enables a user to create the following spatially varying properties:

- Saturated conductivity (K_s)
- The "α" parameter [slope of lnk(h)]
- Saturated water content (θ_s)
- The "β" parameter [of the retention curve, θ(h)]
- The bubbling pressure (h_b)

Once the flow regime (saturated on unsaturated) is specified, the user can invoke the "BLOCKS" option to enable a zone or subdomain with hydraulic properties that are different from the imbedding domain. The following rules must be observed:

- First generate the embedding matrix (background)
- Then create one or many blocks that the user wishes to superimpose onto the embedding matrix
- Up to 500 blocks maybe embedded

The routine "INHEAD" generates a 3D array of heads and saves this to file "INPUT2". The user first enters the exact number of nodes along each of the coordinate axes X, Y, and Z. Upon entering the mesh size, the orientation of the Z axis, the user must successively input the head values FIXA and FIXB along each of the coordinate axes (see Section 4.4.2 for the definition of FIXA and FIXB). The head may either be a constant or linearly varying for hydrostatic pressure head distribution. Note that, for unsaturated flow, the pressure head is used instead of the total pressure head. INBC generates several 2D arrays for the boundary conditions (type of condition and/or value) on each of the six faces of the 3D domain and saves these to file "INPUT3". The user is prompted for the type of boundary condition (LTYPA or LTYP3) and value (FIXA and FIXB) for each of the six faces of the 3D calculation domain. See Section 4.4.2 for a description of available boundary conditions and corresponding values.

4.5.3 Postprocessing of BIGFLOW Output Using DATAFLOW

DATAFLOW offers a range of postprocessing options that can be applied to BIGFLOW output data. The following is a list of DATAFLOW's postprocessing routines:

- DISPLA
- MODIFY
- PIKLIN
- PIKMAP
- GFLUX
- STAT1

A concise description of each of the above routines can be obtained on the screen by typing "MENU" followed by a RETURN. Section 4.5.1 also describes each of the above routines. The

following describes the interaction with "PIKMAP" to create a 2D slice out of a 3D data set. All other postprocessing routines operate in a similar mode.

Upon entering the exact node dimensions of the domain, the field is specified to be a scalar or a vector. The desired 2D plane (slice) is obtained by first selecting the (X2, X3), (X1, X3), or (X1, X2) plane (LDIR=1, 2, 3, respectively) followed by entering the nodal coordinate of the selected plane. For example, in a 125 $\times$ 25 $\times$ 25 domain, the (X1, X2) plane at X3 = 100 can be obtained by selecting LDIR = 3 and the nodal coordinate = 100. The extracted slice is saved in a user specific file in formatted or unformatted form.

5 REFERENCES

Ababou, R. 1981. *Modelisation des Transfers Hydriques dans le Sol en Irrigation Localisee.* Dr.-Ing. Thesis. Grenoble, France: Institut de Mecanique de Grenoble: 218.

Ababou, R., D. McLaughlin, L.W. Gelhar, and A.F. Tompson. 1985. *Numerical Simulation of Saturated Flow Fields in Randomly Heterogeneous Porous Media.* Pre-prints; Montvillargenne, France: International Association for Hydraulic Research (IAHR) and Paris School of Mines: International Symposium on The Stochastic Approach to Subsurface Flow.

Ababou, R. 1988. *Three-Dimensional Flow in Random Porous Media.* Ph.D. Thesis. Cambridge, Massachusetts: Massachusetts Institute of Technology (MIT): Department of Civil Engineering: 2 volumes: 833.

Ababou, R., and L.W. Gelhar. 1988. A high-resolution finite difference simulator for 3D unsaturated flow in heterogeneous media. *Computational Methods in Water Resources.* Elsevier and Computational Mechanics Publications 1: 173-178.

Ababou, R., D. McLaughlin, L.W. Gelhar, and A.F. Tompson. 1989. Numerical simulation of three dimensional saturated flow in randomly heterogeneous porous media. *Transport in Porous Media* 4: 549-565.

Ababou, R. 1990. Numerical analysis of unsaturated flow equations. *Proceedings VIIIth Conference on Computational Methods in Water Resources.* G. Gambolati and others, eds. Venice, Italy: Computational Mechanics Publications and Springer-Verlag: A: 151-160.

Ababou, R. 1991a. *Approaches to Large-Scale Flow in Heterogeneous, Stratified, and Fractured Geologic Media.* NUREG/CR-5743. Washington, D.C.: Nuclear Regulatory Commission (NRC).

Ababou, R. 1991b. Three-dimensional flow in heterogeneous geologic media: High-resolution simulations. *Proceedings International Hydrology and Water Resources Symposium.* Perth, Australia: 725-731.

Ababou, R., L.W. Gelhar, and C. Hempel. 1992a. Serial and parallel performance on large matrix systems. *Cray Channels*, Summer Issue.

Ababou, R., B. Sagar, and G. Wittmeyer. 1992b. Testing procedures for spatially distributed flow models. *Advances in Water Resources* 15: 181-198.

Bagtzoglou, A.C., R. Ababou, and B. Sagar. 1992a. Effects of layering, dipping angle and faulting on two-dimensional variably saturated flow. *Technical Report No. CNWRA 92-004.* San Antonio, Texas: Center for Nuclear Waste Regulatory Analyses (CNWRA).

Bagtzoglou, A.C., R. Ababou, B. Sagar, and M.R. Islam. 1992b. Effects of some common geological features on 2D variably saturated flow. *Materials Research Society Symposium Series.* (In Review).

Bagtzoglou, A.C., G.W. Wittmeyer, R. Ababou, and B. Sagar. 1992c. Application of a massively parallel computer to flow in variably saturated heterogeneous porous media. T. F. Russell, R. E. Ewing, C. A. Brebbia, W. G. Gray, and G. F. Pinder, eds. *Numerical Methods in Water Resources*. Computational Mechanics Publications and Elsevier Applied Science: 695-703.

Bagtzoglou, A.C., G.W. Wittmeyer, and B. Sagar. 1992d. Parallel computing of subsurface flow problems with the Connection Machine (CM) system. *Report on Research Activities for Calendar Year 1991*. W. C. Patrick, ed. CNWRA 91-01A. San Antonio, Texas: CNWRA: 9-27 to 9-53.

Bakr, A.A., L.W. Gelhar, A.L. Gutjahr, and R.J. McMillan. 1978. Stochastic analysis of spatial variability in subsurface flow (1): Comparison of one- and three-dimensional flows. *Water Resources Research* 14(2).

Bouloutas, E.T. 1989. *Improved Numerical Approximations for Flow and Transport in the Unsaturated Zone*. Ph.D. Thesis. Cambridge, Massachusetts: MIT: Department of Civil Engineering.

Bresler, E. 1978. Analysis of trickle irrigation with application to design problems. *Irrigation Science* 1(1): 3-17.

Buckingham, E. 1907. *Studies on the Movement of Soil Moisture*. Bulletin No. 8. Washington, D.C.: U. S. Department of Agriculture: Bureau of Soils.

Celia, M.A., E.T. Bouloutas, and R.L. Zarba. 1990. A general mass-conservative numerical solution for the unsaturated flow equation. *Water Resources Research* 26(7): 1483-1496.

Chen, P. 1988. Convergence of SIP. *SIAM Journal Numerical Analysis* 25(2): 342-350.

Cooley, R. 1983. Some new procedures for the numerical solution of variably saturated flow problems. *Water Resources Research* 19(5): 1271-1285.

Darcy, H.P.G. 1856. *Les Fontaines Publiques de la Ville de Dijon, Exposition et Application des principes a Suivre et des Formules a Employer dans les Questions de Distribution d'Eau*. Paris, France. Victor Dalmont.

Dougherty, D.E. 1990. PCG solutions of flow problems in random porous media using mixed finite elements. *Advances in Water Resources* 13: 2-11.

Dougherty, D.E. 1991. Hydrologic applications of the Connection Machine (CM-2). *Water Resources Research* 27(12): 3137-3147.

Du Pont, T., R.P. Kendall, and H.H. Rachford. 1968. An approximate factorization procedure for solving self-adjoint elliptic difference equations. *SIAM J. Numer. Anal.* 5(3).

Gambolati, G., and A.M. Perdon. 1984. The conjugate gradients in subsurface flow and land subsidence modeling. *Fundamentals of Transport Phenomena in Porous Media*. J. Bear and M. Y. Corapcioglu, eds. The Hague, The Netherlands.

Golub, G.H., and C.F. Van Loan. 1989. *Matrix Computations*. Baltimore, Maryland: The John Hopkins University Press.

Gustafsson, I. 1978. A class of first order factorization methods. *BIT Nordisk Tidskrift for Inform. Behandling* 18: 142-156.

Haverkamp, R., and M. Vauclin. 1979. A note on estimating finite difference interblock hydraulic conductivity value for transient unsaturated flow problems. *Water Resources Research* 15(1).

Hestenes, M., and E. Stiefel. 1952. Method of conjugate gradients for solving linear systems. *National Bureau of Standards, J. Res.* 49: 409-436.

Huyakorn, P.S., S.D. Thomas, and B.M. Thompson. 1984. Techniques for making finite elements competitive in modeling flow in variably saturated porous media. *Water Resources Research* 20(8): 1099-1115.

Jackson, C.P., and P.C. Robinson. 1985. A numerical study of various algorithms related to the preconditioned conjugate gradient method. *Int. J. Num. Methods in Eng.* 21: 1315-1338.

Kershaw, D.S. 1978. The incomplete Choleski-conjugate gradient method for the iterative solution of systems of linear equations. *Journal of Comput. Physics* 2(6): 43-65.

Kuiper, L.K. 1981. A comparison of the incomplete Choleski-conjugate gradient method with the strongly implicit method as applied to the solution of two-dimensional groundwater flow equations. *Water Resources Research* 17(6): 1082-1086.

Kuiper, L.K. 1987. A comparison of iterative methods as applied to the solution of the nonlinear three-dimensional groundwater flow equation. *SIAM Journal on Sci. Stat. Computing* 8(4).

Mantoglou, A., and L.W. Gelhar. 1987a. Stochastic modeling of large-scale transient unsaturated flow systems. *Water Resources Research* 23(1): 37-46.

Mantoglou, A., and L.W. Gelhar. 1987b. Capillary tension head variance, mean soil moisture content, and effective specific soil moisture capacity of transient unsaturated flow in stratified soils. *Water Resources Research* 23(1): 47-56.

Mantoglou, A., and L.W. Gelhar. 1987c. Effective hydraulic conductivities of transient unsaturated flow in stratified soils. *Water Resources Research* 23(1): 57-67.

Matheron, G. 1967. *Elements Pour une Theorie des Milieux Poreux*. Paris, France: Masson et Cie: 164.

McCord, J.T. 1991. Application of second-type boundaries in unsaturated flow modeling. *Water Resources Research* 27(12): 3257-3260.

McDonald, M.G., and A.W. Harbaugh. 1984. *A Modular Three Dimensional Finite-Difference Groundwater Flow Model.* Reston, Virginia: U. S. Department of the Interior and U. S. Geological Survey, Nat. Center: 528.

Meijerink, J.A., and H.A. Van der Vorst. 1977. An iterative method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math Comp.*

Meyer, P.D., A.J. Valocchi, S.F. Ashby, and P.E. Saylor. 1989. A numerical investigation of the conjugate gradient method as applied to three-dimensional groundwater flow problems in randomly heterogeneous porous media. *Water Resources Research* 25(6): 1440-1446.

Milly, P.C.D. 1985. A mass-conservative procedure for time-stepping in models of unsaturated flow. *Advances in Water Resources* 8: 32-36.

Philip, J.R. 1957. Numerical solution of equations of the diffusion type with diffusivity concentration-dependent II. *Australian Journal of Physics* 10(2): 29-42.

Polmann, D.J., D. McLaughlin, S. Luis, L.W. Gelhar, and R. Ababou. 1991. Stochastic modeling of large-scale flow in heterogeneous unsaturated soils. *Water Resources Research* 27(7): 1447-1458.

Richards, L.A. 1931. Capillary conduction of liquids through porous medium. *Physics* 1: 318-333.

Runchal, A.K., and B. Sagar. 1992. *PORFLOW: A Multifluid Multiphase Model for Simulating Flow, Heat Transfer, and Mass Transport in Fractured Porous Media - User's Manual, Version 2.40.* CNWRA 92-003. San Antonio, Texas: CNWRA.

Stone, H.L. 1968. Iterative solution of implicit approximations of multi-dimensional partial differential equations. *SIAM Journal Num. Anal.* 5(3).

Tompson, A.F., R. Ababou, and L.W. Gelhar. 1987. Applications and use of the three-dimensional turning band random field generator in hydrology: Single realization problems. *Technical Report No. 313.* Cambridge, Massachusetts: MIT: Parsons Laboratory.

Tompson, A.F., R. Ababou, and L.W. Gelhar. 1989. Implementation of the three-dimensional turning bands random field generator. *Water Resources Research* 25(10): 2227-2243.

Tompson, A.F.B., and L.W. Gelhar. 1990. Numerical simulation of solute transport in three-dimensional, randomly heterogeneous porous media. *Water Resources Research* 26(10): 2541-2562.

Townley, L.R., J.V. Turner, M.G. Trefry, and A.D. Barr. 1991. Groundwater flow patterns near lakes and wetlands: Modeling and field validation. *International Hydrology and Water Resources Symposium*. Perth, Australia 3: 732-737.

Townley, L.R., A.D. Barr, S. Braumiller, M. Kawanashi, D.A. Lever, K. Miyakawa, S.T. Morris, J.P. Raffensperger, J.L. Smoot, Y. Tanaka, and M.G. Trefrey. 1992a. Hydrogeological modeling. *Alligator Rivers Analogue Project, Final Report*. ANSTO. Lucas Heights, N.S.W., Australia 6: 200. (In press).

Townley, L.R., J.V. Turner, A.D. Barr, M.G. Trefry, K.D. Wright, C.J. Harris, V. Gailitis, and C.D. Johnston. 1992b. Interaction between lakes, wetlands, and aquifers. CSIRO Division of Water Resources. *Water Resources Series:* 250. (In press).

Trefry, M.G., and L.R. Townley. 1991. Three-dimensional modeling of groundwater flow through the Koongarra uranium orebody, Northern Territory, Australia. *International Hydrology and Water Resources Symposium.* Perth, Australia 2: 623-624.

Trescott, P.C. 1975. Documentation of finite difference model for simulation of three-dimensional groundwater flow. *U. S. Geological Survey*. Open File Report 75: 438.

Trescott, P.C., and S.P. Larson. 1977. Comparison of iterative methods of solving two-dimensional flow equations. *Water Resources Research* 13(1).

Van der Vorst, H. 1981. A vectorizable variant of some ICCG methods. *SIAM J. Sci. Stat. Comput*. 3: 350-356.

van Genuchten, M.Th. 1980. A closed form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil Sci. Soc. Am. J.* 44: 892-898.

Vauclin, M., R. Haverkamp, and G. Vachaud. 1979. *Resolution Numerique d'une Equation de Diffusion Nonlineaire*. P.U.G., Grenoble, France: 183.

Warrick, A.W., and D.O. Lomen. 1977. Flow from a line source above a shallow water table. *Soil Sci. Soc. Am. J.* 41: 849-852.

Weinstein, H.G., H.L. Stone, and T.V. Kwan. 1969. Iterative procedure for solution of systems of parabolic and elliptic equations in three dimensions. *Industr. Engin. Chem, Fundamentals* 8(2).

White, I., and M.J. Sully. 1987. Macroscopic and microscopic capillary length and time scales from field infiltration. *Water Resources Research* 23(8): 1514-1522.

Yeh, T.-C., L.W. Gelhar, and A.L. Gutjhar. 1985. Stochastic analysis of unsaturated flow in heterogeneous soils: (2) Statistically anisotropic media with variable α. *Water Resources Research* 21(4): 457-464.

APPENDIX A

ERROR MESSAGES OF BIGFLOW

ERROR MESSAGES OF BIGFLOW

BIGFLOW error messages will appear in a file named "OUTBAD", which is overwritten at each new simulation whether there is a BIGFLOW error or not. If there are no such errors, the message will be of the form "BIGFLOW errors:...none." On the other hand, a detected error or data conflict will cause a different message to be printed in OUTBAD, and will also cause premature end-of-execution. In Table A-1 the error messages that may result when executing a simulation with BIGFLOW are listed.

Table A-1. Error messages of BIGFLOW

Error #	Meaning
LBAD = -1	Data inconsistent with flow type
LBAD = -2	Illegal boundary type selected
LBAD = -3	Inconsistent use of zero gradient boundary condition
LBAD = -4	Inconsistent use of moving boundary
LBAD = -5	The specified gravity vector does not have unit length
LBAD = -6	An illegal value for LBFLUX was selected
LBAD = -7	Inconsistent specification of transient outputs
LBAD = -8	Incorrect specification of input/output file formats
LBAD = -9	An invalid value for LGRID was selected
LBAD = -10	Steady state unsaturated flow currently unavailable (Kirchoff transform-based equations)
LBAD = -11	Inconsistent specification of matrix solver relaxation
LBAD = -12	Bubbling head cannot be strictly positive
LBAD = -13	(LFLOW = 1 OR LTRANS = 0) AND LMASS = 1
LBAD = -14	IF((LLLSSS.NE.110).AND.(LLLSSS.NE.120).AND.(LLLSSS.NE.121).AND .LLLSSS.NE.200).AND.(LSOLV.NE.+210).AND.(LSOLV.NE.+220))
LBAD = -15	Incorrect specification of flow type
LBAD = -16	Illegal initial head distribution selected
LBAD = -19	Illegal option selected for computation of error norm
LBAD $\leq$ -20	An error was detected in the input file "INPUTX", where X is the first digit of the error code LBAD

APPENDIX B

PARTIAL INPUT AND OUTPUT FILES FOR TEST PROBLEM # 5 (Section 3.5)

INPUT 1 FILE (GENERAL DATA)

```
IDATE (DAY-MONTH-YEAR):
71192
IDRUN (SIMULATION ID NUMBER):
10000
LRUN (0:TEST RUN/1:FULL RUN):
1
LUNF (0:FORMATTED/1:UNFORMATTED):
0
NGRID1,2,3 (FULL GRID SIZE: TOTAL No.of NODES):
62, 5, 123
MING1,2,3 (STARTING GRID SIZE: TOTAL No.of NODES):
62, 5, 123
MDELT1,2,3 (SEARCH ZONE FOR ACTIV); HDELT; LDELT :
3*1, 1., 1
GRAV1,2,3 (GRAVITY VECTOR, DEFINES AXES):
2*0., +1.
TYPE OF BNDRY CONDITIONS: LTYPA(j),LTYPB(j),j=1,2,3:
2*2
2*2
1, 22
BNDRY (or stability) CONDITIONS:(FIXAj,FIXBj)j=1,2,3:
2*0.
2*0.
2*0.
MESH SIZE DX1,DX2,DX3:
3*1.
FLOW REGIME: LFLOW=(1:SATURATED/2,3:UNSATURATED):
2
TRANSIENT/STEADY FLOW (LTRANS=1/0):
1
LKWNOD=1,2,3 (MIDNOD.KSAT:GEOM,HARMON,ARITHM MEAN):
1
LHIN (0:Hin UNIFORM, 1:NONUNIFORM, 10*i:LINEAR-Xi):
1
HIN (INITIAL OR REFERENCE HEAD VALUE):
-122.
Ksat STATISTICS (FKGM,FKDEV,FKL1,FKL2,FKL3):
1.12E-3, 1., 3*1.
STORATIVITY STATISTICS (CAPGM,CAPDEV,CAPL1,2,3):
5*0.
ALFA STATISTICS (ALFGM,ALFDEV,ALFAL1,2,3):
0.1258, 1., 3*1.
TTsat STATISTICS (TTGM,TTDEV,TTL1,TTL2,TTL3):
0.4411, 4*0.
BETA STATISTICS (BTGM,BTDEV,BTL1,BTL2,BTL3):
5.E-2, 4*0.
BUBBLE PRESSURE STATISTICS (HBBGM,HBBDEV,HBBL1,2,3):
5*0.
```

INPUT 1
(Cont'd)

```
LTHETA=Theta(h) option; TTDRY; VGN=VanGENUCHTEN "n" :
30,  1.89E-2,  3.872

HSMALL (FOR UNSAT. MOISTURE CAPACITY COMPUTATION):
0.5
TIME STEP (DTIN,DTMIN,DTMAX,DTMUL):
0.2,  0.1,  10.,  1.2
DHSTAB (TYPICAL HEAD DIFFERENCE FOR T-STEP CONTROL):
122.
CONDMA ("MAXIMUM" SAT/UNSAT CONDUCTIVITY):
1.12E-3
SCAPMI ("MINIMUM" STORATIVITY OR MOISTURE CAPACITY):
0.
LSOLV (LINEAR SYSTEM SOLVER OPTION):
-200
LNORM (OPTION FOR CHOICE OF ERROR NORM):
5
ERMIN,ERMAX (ERROR NORMS/INNER ITER: E<ERMIN-->CONV):
1.E-3,  0.
ITEND (MAX. NUMBER OF LINEAR SOLVER ITERATIONS):
50
LPEPIT (SIP ITER. PARAM. : CSTANT(0) OR CYCLIC(1)):
1
MIT (PERIOD OF SIP ITER. PARAM., IF CYCLIC):
4
PITMAX (MAX. ITER. PARAM. OF "SIP": 0 IF UNKNOWN):
0.
LINEAR SOLVER RELAXATION (UNLAX,OVLAX,VARLAX):
2*1.,  0.
ENLMIN,ENLMAX (ERR. NORM/OUTER ITER: E<ENLMIN->CONV):
1.E-3,  0.
INLMAX (MAX.No.NONLINEAR OUTER ITERATIONS); INLBC=0 :
50,  0
HNLAX,RNLAX (RELAX FACTORS FOR NONLINEAR ITERATIONS):
2*1.
KTMAX (MAX. OR TOTAL NUMBER OF TIME STEPS):
20000
TYMAX (MAX. OR TOTAL TIME OF TRANSIENT SIMULATION):
36000.
LTOUT (OPTION FOR OUTPUTS AT GIVEN TIMES):
1
KTOUT (NUMBER OF EARLY OUTPUT TIMES):
1
```

INPUT 1
(Cont'd)

EARLY OUTPUT TIMES TYMOUT(1),...,TYMOUT(ktout):
7200.
OUTPUTS OPTIONS KOUT10,KOUT11,KHOUT:
2*2000, 0
OUTPUT "FORMATS" OPTIONS LFORM1,LFORM2,LHOUT:
3*1

LOCATION OF NODE/LINE/PLANE FOR THE PROBE (IPROBj):
31, 3, 0
SPECIAL OPTIONS LBFLUX,LMASS:
2*1
MACHINE DEPENDENT CSTANTS: SRELPR,RSMALL:
1.E-6, 1.E-30
ACTUAL NAMES OF ALL INPUT FILES (INPUT1/.../INPUT9):
IN1
IN2
IN3
INPUT4
IN5
IN6
INPUT7
INPUT8
INPUT9

PARTIAL INPUT 2 FILE (INITIAL CONDITIONS)

```
10000 21192     2     2
  0 0.0000000E+00
  62     5   123
0.1000000E+01  0.1000000E+01  0.1000000E+01
0.0000000E+00  0.0000000E+00  0.1000000E+01
0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00

0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
                                  .
                                  .
                                  .
                                  .
0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
-0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01
-0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01
-0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01
-0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01
-0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01 -0.1000000E+01
                                  .
                                  .
                                  .
                                  .
-0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01
-0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01
-0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01
-0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01
-0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01
-0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01 -0.2000000E+01
                                  .
                                  .
                                  .
                                  .
-0.1220000E+03 -0.1220000E+03 -0.1220000E+03 -0.1220000E+03 -0.1220000E+03
```

PARTIAL INPUT 3 FILE (BOUNDARY CONDITIONS)

```
10000  21192     3     1
    0  0.0000000E+00
   62     5   123
 0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.1000000E+01
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00

 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
                               .
                               .
                               .
 0.0000000E+00 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02 -0.1000000E-02
-0.1000000E-02  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
 0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
```

INPUT 3
(Cont'd)

0.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00
0.0000000E+00 0.0000000E+00

PARTIAL INPUT 5 FILE (SATURATED CONDUCTIVITY K_s)

```
10000  71192      5      2
    0  0.0000000E+00
   62      5    123
0.1000000E+01  0.1000000E+01  0.1000000E+01
0.0000000E+00  0.0000000E+00  0.1000000E+01
0.1120000E-02  0.1000000E+01  0.1000000E+01  0.1000000E+01  0.1000000E+01

0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
                                   .
                                   .
                                   .
                                   .
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04  0.2240000E-04
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02  0.1120000E-02
                                   .
                                   .
```

INPUT 5
(Cont'd)

.

.

0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04
0.2240000E-04	0.2240000E-04	0.2240000E-04	0.2240000E-04	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02

.

.

.

.

0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02
0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02	0.1120000E-02

PARTIAL INPUT 6 FILE (CHARACTERISTIC INVERSE LENGTH α)

```
10000  71192     6     2
   0  0.0000000E+00
  62     5   123
0.1000000E+01  0.1000000E+01  0.1000000E+01
0.0000000E+00  0.0000000E+00  0.1000000E+01
0.1258000E+00  0.1000000E+01  0.1000000E+01  0.1000000E+01  0.1000000E+01

0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
                              .
                              .
                              .
                              .
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01  0.8390000E-01
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00  0.1258000E+00
                              .
                              .
```

INPUT 6
(Cont'd)

.
.

0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01
0.8390000E-01	0.8390000E-01	0.8390000E-01	0.8390000E-01	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00

.
.
.
.

0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00
0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00	0.1258000E+00

PARTIAL HEAD_6 OUTPUT FILE (HYDRAULIC PRESSURE HEAD AT TIME T=6 HOURS)

```
10000  71192     23      2
 2190  0.2159526E+05
   62      5    123
0.1000000E+01  0.1000000E+01  0.1000000E+01
0.0000000E+00  0.0000000E+00  0.1000000E+01
0.1120000E-02  0.1000000E+01  0.1000000E+01  0.1000000E+01  0.1000000E+01

0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
.
.
.
.
-0.6859475E+01 -0.7024314E+01 -0.7219880E+01 -0.7452353E+01 -0.7730099E+01
-0.8064729E+01 -0.8472863E+01 -0.8978525E+01 -0.9613236E+01 -0.1039152E+02
-0.1521656E+02 -0.3078397E+02 -0.4556410E+02 -0.6442622E+02 -0.7797465E+02
-0.7990353E+02 -0.7999640E+02 -0.7999988E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02  0.0000000E+00 -0.6242951E+01 -0.6242951E+01 -0.6254717E+01
-0.6278490E+01 -0.6314777E+01 -0.6364286E+01 -0.6428101E+01 -0.6507524E+01
-0.6604298E+01 -0.6720688E+01 -0.6859475E+01 -0.7024314E+01 -0.7219880E+01
-0.7452353E+01 -0.7730099E+01 -0.8064729E+01 -0.8472863E+01 -0.8978525E+01
-0.9613236E+01 -0.1039152E+02 -0.1521656E+02 -0.3078397E+02 -0.4556410E+02
-0.6442622E+02 -0.7797465E+02 -0.7990353E+02 -0.7999640E+02 -0.7999988E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.6242949E+01
-0.6242949E+01 -0.6254747E+01 -0.6278512E+01 -0.6314771E+01 -0.6364322E+01
-0.6428110E+01 -0.6507534E+01 -0.6604337E+01 -0.6720700E+01 -0.6859509E+01
-0.7024345E+01 -0.7219912E+01 -0.7452397E+01 -0.7730138E+01 -0.8064763E+01
-0.8472904E+01 -0.8978558E+01 -0.9613292E+01 -0.1039149E+02 -0.1521654E+02
-0.3078396E+02 -0.4556409E+02 -0.6442619E+02 -0.7797464E+02 -0.7990353E+02
-0.7999640E+02 -0.7999988E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
-0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02 -0.8000000E+02
.
.
```

INPUT 6
(Cont'd)

.
.

-0.1000000E+03 -0.1277587E+01 -0.1277587E+01 -0.1275357E+01 -0.1270716E+01
-0.1263502E+01 -0.1253380E+01 -0.1239860E+01 -0.1222339E+01 -0.1200040E+01
-0.1171877E+01 -0.1136669E+01 -0.1092670E+01 -0.1037945E+01 -0.9697806E+00
-0.8849020E+00 -0.7789644E+00 -0.6465534E+00 -0.4812425E+00 -0.2786942E+00
-0.5281955E-01 0.1002191E+01 0.2225468E+01 0.2599149E+01 0.2744816E+01
0.2774986E+01 0.2723646E+01 0.2604612E+01 0.2422578E+01 0.2178572E+01
0.1872746E+01 0.1504556E+01 0.1074000E+01 0.5808875E+00 0.1342259E-01
-0.6493974E+00 -0.1421050E+01 -0.2327769E+01 -0.3416213E+01 -0.4754379E+01
-0.6514933E+01 -0.1013225E+02 -0.1368492E+02 -0.1532645E+02 -0.1737570E+02
-0.1977100E+02 -0.2254700E+02 -0.2577918E+02 -0.2960512E+02 -0.3430714E+02
-0.4059699E+02 -0.5408818E+02 -0.9344700E+02 -0.9994563E+02 -0.9999971E+02
-0.1000000E+03 -0.1000000E+03 -0.1000000E+03 -0.1000000E+03 -0.1000000E+03
-0.1000000E+03 -0.1000000E+03 -0.1000000E+03 0.0000000E+00 -0.1277587E+01
-0.1275357E+01 -0.1270716E+01 -0.1263502E+01 -0.1253380E+01 -0.1239860E+01
-0.1222339E+01 -0.1200040E+01 -0.1171877E+01 -0.1136669E+01 -0.1092670E+01
-0.1037945E+01 -0.9697806E+00 -0.8849020E+00 -0.7789644E+00 -0.6465534E+00
-0.4812425E+00 -0.2786942E+00 -0.5281955E-01 0.1002191E+01 0.2225468E+01
0.2599149E+01 0.2744816E+01 0.2774986E+01 0.2723646E+01 0.2604612E+01
0.2422578E+01 0.2178572E+01 0.1872746E+01 0.1504556E+01 0.1074000E+01
0.5808875E+00 0.1342259E-01 -0.6493974E+00 -0.1421050E+01 -0.2327769E+01
-0.3416213E+01 -0.4754379E+01 -0.6514933E+01 -0.1013225E+02 -0.1368492E+02
-0.1532645E+02 -0.1737570E+02 -0.1977100E+02 -0.2254700E+02 -0.2577918E+02

.
.
.
.

0.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00
0.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00