A THREE-DIMENSIONAL GROUND-WATER-FLOW MODEL MODIFIED TO REDUCE COMPUTER-MEMORY REQUIREMENTS AND BETTER SIMULATE CONFINING-BED AND AQUIFER PINCHOUTS

By P. Patrick Leahy

U. S. GEOLOGICAL SURVEY

Water-Resources Investigations 82-4023

Trenton, New Jersey

1982
CONTENTS

Abstract .......................................................... 1
Introduction ..................................................... 1
Memory reduction ................................................ 2
Simulating confining bed and aquifer pinchouts.............. 6
Input requirements .............................................. 8
Comparison of memory requirements and cost for simulation
of the regional aquifer system ................................ 9
Conclusions ....................................................... 12
References ....................................................... 12
Appendix 1: Data deck instructions............................ 14
Appendix 2: Listing of source code............................ 23

ILLUSTRATIONS

Figure 1.--Map showing location of northern Atlantic Coastal
Plain .............................................................. 3
2.--Numbering scheme for finite-difference mesh used
by the Tresco (1975) model and the modified
model ............................................................ 5
3.--Generalized cross sections and conceptualizations
of aquifer and confining-bed pinchouts.............. 7
4.--Finite-difference grid used to discretize the
northern Atlantic Coastal Plain aquifer system. 10

TABLES

Table 1.--Cost savings realized by modified model for
sample problems .............................................. 12

III
A THREE-DIMENSIONAL GROUND-WATER-FLOW MODEL MODIFIED TO REDUCE 
COMPUTER-MEMORY REQUIREMENTS AND BETTER SIMULATE 
CONFINING-BED AND AQUIFER PINCHOUTS

By P. Patrick Leahy

ABSTRACT

The Trescott (1975) computer program for modeling ground-water flow in three dimensions has been modified to (1) treat confining-bed and aquifer pinchouts more realistically and (2) reduce the computer-memory requirements needed for the input data. Simulating aquifer systems having nonrectangular external boundaries with the original program may result in many nodes that are not involved in the numerical solution of the problem, but that require computer memory.

The Trescott program and the modified program are used to develop a cost comparison of the computer programs for large field problems. Steady-state simulations of the northern Atlantic Coastal Plain regional aquifer system and the New Jersey Coastal Plain are used. The comparison shows a 30 percent cost savings for the regional model, which has 11,560 unused nodes of a total of 27,200. The New Jersey model shows an 8 percent savings, where 1,540 of the 14,790 nodes are unused.

Appendices provide data-deck instructions and a listing of the FORTRAN source code.

INTRODUCTION

A computer model (Trescott, 1975) for simulating ground-water flow in three dimensions was developed by the U.S. Geological Survey. Hydrologists of the Survey have applied this program to numerous field problems. The structure of the FORTRAN code, however, is inconsistent with efficient and cost-effective simulation of certain physical and hydrologic situations. In particular, confining-bed and aquifer pinchouts are not easily simulated with the Trescott model. Also, simulating complex external boundaries (shape of the entire aquifer system) results in inefficient use of computer storage. Developing a model that effectively handles these situations is an outgrowth of the study of the northern Atlantic Coastal Plain aquifer system begun in 1979 by the Geological Survey. This project is part of the Regional Aquifer System Analysis (RASA) program that began in 1977. These studies define the geology, hydrology, and geochemistry of the aquifer system and the ground-water-flow regime on a regional scale.

All the regional studies use computer models to simulate flow within an aquifer system, but, unlike many studies that stress the importance of the predictive capabilities of calibrated
ground-water-flow models, the RASA studies simulate primarily for analysis (Bennett, 1979). Thus, computer models are used to test hypotheses to increase knowledge of ground-water-flow systems. With this approach, construction of a ground-water-flow model of the Coastal Plain sediments has begun (Meisler, 1980).

The northern Atlantic Coastal Plain includes about 50,000 mi², extending from the North Carolina-South Carolina border, through Long Island, N.Y. (fig. 1). The external boundary of the fresh-water system is complex, as shown by the shape of the Fall Line onshore and the 10,000 milligrams-per-liter chloride-concentration line offshore (Meisler, 1981). The concentration line is assumed to approximate the seaward limit of the freshwater-flow system. Ten aquifers and nine intervening confining beds in the ground-water system have been delineated. Because of the large area of the project and the presence of numerous confining-bed and aquifer pinchouts in the system, a modified version of the Trescott model was needed. This report presents (1) conceptual changes in the existing flow model required to reduce computer-memory requirements and better simulate confining-bed and aquifer pinchouts, (2) modifications required in the model input, (3) the modified source code, and (4) a cost comparison of the modified and Trescott (1975) versions of the three-dimensional ground-water-flow model.

MEMORY REDUCTION

To understand the method of memory reduction, knowledge of the nodal numbering or indexing scheme used in the unmodified Trescott model is required. The finite-difference approach requires subdividing of the aquifer system into discrete blocks, which are assumed to have uniform hydraulic properties. A point in the center of the finite-difference block is known as the node and is commonly numbered to conveniently locate the position of the node in the discretized system. A finite-difference approximation of the differential equations of three-dimensional ground-water-flow results in a system of algebraic equations. A generalized equation for this system at node \( i,j,k \) is:

\[
B_{i,j,k} h_{i-1,j,k} + D_{i,j,k} h_{i,j-1,k} + E_{i,j,k} h_{i,j+1,k} + F_{i,j,k} h_{i+1,j,k} + G_{i,j,k} h_{i,j,k} = H_{i,j,k} h_{i,j,k} + Z_{i,j,k} h_{i,j,k} (1)
\]

where \( B_{i,j,k}, D_{i,j,k}, E_{i,j,k}, F_{i,j,k}, G_{i,j,k}, H_{i,j,k}, S_{i,j,k}, Z_{i,j,k} \) are

[1]
Figure 1.— Location of northern Atlantic Coastal Plain.
constants, as defined by Trescott (1975). The constants are the harmonic means of the appropriate hydraulic parameters for adjacent blocks. These coefficients provide the correct hydraulic connection between adjacent nodes.

\[ S' \] is storage coefficient;
\[ \Delta t \] is length of time step;
\[ h_{i,j,k} \] is the hydraulic head (known) at time \( t \);
\[ h_{i,j,k} \] is the hydraulic head (unknown) at time \( t + \Delta t \);
\[ i \] is the node number in the y-direction;
\[ j \] is the node number in the x-direction;
\[ k \] is the node number in the z-direction.

The location of these parameters in a finite-difference mesh is shown on the left of figure 2. In this example, there are a total of 27 finite-difference blocks, 3 blocks in the x-direction, 3 in the y-direction, and 3 in the z-direction. The blocks can be indexed by the \( i,j,k \) notation of Equation 1. The indices of each block are shown within the block of figure 2. However, for computation, a more efficient notation is numbering the nodes sequentially (from 1 to 27 in this example), as shown by Trescott, (1975, p. II-5). The \( i,j,k \) indices are converted to one index, \( N \), by the following equation:

\[
N = i + (j-1)I0 + (k-1)I0*J0
\]  

(2)

where \( I0 \) is the total number of blocks in the y-direction and \( J0 \) is the total number of blocks in the x-direction.

In a finite-difference model, the model grid is always rectangular, whereas the modeled area is generally not. Hence, some nodes in the grid are usually outside the modeled area. Nodes outside the modeled area are designated as no-flow nodes. In the Trescott model, all nodes are numbered, and data are stored at all nodes, including the nodes outside the modeled area. Memory can, therefore, be saved by eliminating the storage of data at these nodes. Because regional models of the scope of the northern Atlantic Coastal Plain need many nodes to represent the aquifer system, wasted computer-memory is not only costly but may affect the choice of the grid spacing used in the model and the computer on which the model will be run. That is, computers have a finite-memory limit, which restricts the size of model that can be run, regardless of cost.

In an aquifer system that contains a large percentage of nodes outside the modeled area, a more efficient method of indexing is numbering only the nodes within the modeled area. These nodes will be referred to as active nodes in the remainder of the report, as opposed to inactive nodes, which are defined in this report as nodes outside the modeled area. The modified model
Model block showing block number \( N = 6 \) and indices \( i=3, j=2 \) and \( k = 1 \). \( N = i + (j-1) \times O + (k-1) \times O \times J \).

In this example \( IO \) is 3 and \( JO \) is 3. The inactive nodes are denoted with an \( X \).

EXPLANATION

Model block showing block number \( N = 6 \) and indices \( ij=6 \) and \( k = 1 \). \( N = ij+(k-1) \times IOJO \).

In this example \( IOJO \) is 7, the total number of active nodes in a layer.

Figure 2.—Numbering scheme for finite-difference mesh used by the Tréscott (1975) model and the modified model.
uses this indexing scheme and takes advantage of the accompanying reduction in computer-memory. The indexing scheme used in the modified model is shown on the right of figure 2. In the example, there are two inactive nodes in each layer, thus, numbering only the active nodes results in a total of 21 finite-difference blocks. In comparison, the standard Trescott indexing scheme results in 27 blocks requiring storage. The i,j indices are not directly used in this indexing scheme; however, a variable ij is used. In any layer, the number of a node is i,j, where ij refers only to the active nodes in the layer. The nodal index N is given by the following equation:

\[ N = ij + (k-1) \times IOJO \]  

where; ij has a range from 1 to IOJO,

k is the number of the layer of interest, and

IOJO is the number of active nodes in any layer.

SIMULATING CONFINING BED AND AQUIFER PINCHOUTS

The Trescott model code uses the transmissivity (T) matrix as a flag for determining if a node is included (T>0) or excluded (T=0) in the computational scheme. The solution process sweeps through all the nodes sequentially, and a check of the value of transmissivity determines the disposition of the node in the model. However, in certain hydrologic situations, the transmissivity matrix alone is not adequate to define the disposition of the node in the model.

Figure 3 shows typical cross sections, with pinchouts of aquifers and confining beds, and the model conceptualizations of the system in a discretized domain. In case one, the confining bed pinches out. In both the Trescott and modified models, the lateral flow in the aquifers is controlled by the transmissivities; the vertical connection between the aquifers is controlled by the appropriate vertical leakance (K_v/b). In areas where the confining bed is absent, the vertical hydraulic properties of the aquifer are used in the model, whereas the vertical properties of the confining bed are used where it is present. Typically, the vertical leakance of the aquifers is much larger than that of the confining beds.

An example of an aquifer pinching out between two adjacent confining beds is shown in case 2. The conceptualization used in the modified model is such that lateral flow in the aquifer is controlled by the transmissivity. Where the aquifer is absent, the transmissivity is specified as zero (no lateral flow), and all flow is in the vertical direction, controlled by the vertical hydraulic properties of the adjacent confining beds. In contrast,
Confining Bed

Case 1—Confining Bed Pinchout

Arrows show vertical and horizontal connection between nodes. Length of arrows represents amount of hydraulic connection.

Case 2—Aquifer Pinchout

Physical Configuration

Model Conceptualization

Figure 3.—Generalized cross sections and conceptualizations of aquifer and confining-bed pinchouts.
the logic used in the Trescott model requires that an artificial value of transmissivity greater than zero be specified in the areas where the aquifer is absent. If the transmissivity specified is very small, the lateral flow in this area becomes negligible, and the appropriate hydraulic connections are simulated. Hence, to simulate aquifer pinchouts accurately with the Trescott model artificial values of transmissivity are required to ensure that these areas are included in the solution scheme. These artificial values cause unnecessary computational effort (lateral flow), and model results may be sensitive to the value of transmissivity specified.

To model subcrops and pinchouts accurately, the modified model was developed, so that the transmissivity matrix alone is not used to decide whether a node is included in the computational scheme. The modified model checks for vertical hydraulic connections as well as horizontal connections. If both the transmissivity and vertical leakance ($K_v/b$, variable TK in the model) are zero, the node is skipped in the computational scheme. However, if either the transmissivity or TK are nonzero, the node is retained.

Besides the changes in program logic, the user must calculate and enter appropriate transmissivity and TK values to match the modified conceptualization of the system. However, TK and transmissivity values are read into the model exactly as before.

In areas where an aquifer is absent between adjacent confining beds, the aquifer transmissivity should be set to zero, and appropriate confining-bed leakance values should be entered to control the vertical flow from one confining bed to another. The head computed for the node represents the head at the contact of adjacent confining beds.

In areas where a confining bed pinches out, the vertical flow between adjacent aquifers is controlled by the vertical hydraulic properties of the aquifers. The harmonic mean of the TK values of adjacent aquifers is commonly used as the effective TK between them. The harmonic mean is $2*TK1*TK2/(TK1 + TK2)$ where $TK1$ and $TK2$ are $K_v/b$ of the adjacent aquifers.

INPUT REQUIREMENTS

Model-program modifications required to implement the memory reduction occur in every subroutine. Also, minor modifications to the input requirements of the program were necessary. Input instructions are given in Appendix I, and the modified model program is listed in Appendix II. The model input is essentially identical to the standard flow model, making the model easily compatible with available input data decks with minor additions. Data additions include (1) defining the starting and ending column of active or modeled nodes in each row in the model and (2) the
total number of active nodes in a composite layer. For example, if layer 1 in the model has active nodes in columns 5 through 15 and layer 2 has active nodes in columns 8 through 25, the composite layer would have active nodes in columns 5 through 25. The starting and ending position of active or modeled nodes in each row in the model are input as two additional data arrays in the model. The size of each array is 10, the number of rows in the model. Within the modeled part of the finite-difference mesh, nodes representing areas where an aquifer is absent can still be indicated by zero transmissivity.

Even though data at unmodeled nodes are not stored, model data values are input for all nodes in the same format as in the standard flow model. The modified program uses the information that defines unused nodes to skip over unneeded data on input. Use of the same input structure makes conversion simpler than if a new input structure were used. On output, printout positions are maintained for all nodes, but, at unused nodes, values are printed as blanks. This makes it readily apparent which nodes are unused.

The program keeps track of the physical location of the adjacent nodes in relation to the node of interest during computation. This assures that the correct lateral hydraulic connections (coefficients TR and TC) are computed in SUBROUTINE COEF, and the solution algorithm is properly formulated in SUBROUTINE SOLVE.

To insure compatibility with the standard Trescott model, the example problem in Trescott (1975, Appendix IV) was simulated with the modified model. In this simulation, the storage for the zero-transmissivity nodes surrounding the active nodes, with the exception of a single dummy node in both rows 1 and 20, was eliminated from the Y-vector (the variable that contains storage for all array data). There were 74 inactive nodes per layer in the modified model simulation out of a total of 400 nodes per layer. The modified model reproduced exactly the results presented in Trescott (1975) for the sample problem.

COMPARISON OF MEMORY REQUIREMENTS AND COST FOR SIMULATION OF THE REGIONAL AQUIFER SYSTEM

The preliminary finite-difference grid used to discretize the northern Atlantic Coastal Plain aquifer system is shown in figure 4. The grid consists of 85 rows, 32 columns, and 10 layers, totalling 27,200 nodes. Because of the complex external geometry, not all of these nodes are actually needed to represent the aquifer system. The modeled or active part of the grid is bounded on the west by the Fall Line (Brown, 1972) and on the east by the seaward limit of the 10,000 mg/L isochlor (Meisler, 1981).
Figure 4.—Finite-difference grid used to discretize the northern Atlantic Coastal Plain aquifer system.
The outline on the grid defines the maximum number of active nodes for a composite of all the layers in the aquifer system. Although some inactive nodes are present within the outline, their number is small. The outline surrounds 15,640 nodes, or 57.5 percent of the 27,200 in the entire grid. The FORTRAN code was modified to store data only for nodes contained within the area defined by the outline (fig. 4).

Simulations of the northern Atlantic Coastal Plain aquifer system model were used to compare costs of the modified model and the Trescott model. Table 1 shows the results. As previously stated, 42.5 percent of the total 27,200 nodes are inactive in this model (simulation 1). The length of the Y-vector required for the simulation is smaller with the modified version. In general, savings were largely on memory costs; savings were also appreciable, however, on cost of execution time. The modified model executes faster because fewer time-consuming disk input and output operations associated with a virtual memory system are required with the smaller memory space. In general, the overall savings increase with an increasing ratio of inactive to total number of nodes. This is demonstrated by simulation 2 (table 1), which is identical to simulation 1 except that only the inactive nodes in the artificial zero transmissivity border (columns 1 and 32, rows 1 and 85) have been removed from storage. With this change, 91.6 percent of the total 27,200 nodes require storage space in the Y-vector.

As a further test of the modified model, the Coastal Plain aquifers of New Jersey were simulated both by the modified and the standard Trescott versions. The modified model duplicated the results of the Trescott model. A cost comparison of these simulations is given in table 1, simulation 3. The total number of nodes in these simulations is 14,790, of which 10 percent are inactive. Because, overall, there are fewer total nodes in this simulation than in the regional model, the memory savings are less. As expected, the overall savings, in comparison to simulation 1, are less.
Table 1.--Cost savings realized by modified model for sample problems.

<table>
<thead>
<tr>
<th>Simulation</th>
<th>Number of total nodes</th>
<th>Number of nodes requiring storage</th>
<th>Percentage of total storage</th>
<th>Percentage cost savings realized by use of modified model for sample problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>27,200</td>
<td>15,640</td>
<td>57.5</td>
<td>30 18 45</td>
</tr>
<tr>
<td>2</td>
<td>27,200</td>
<td>24,920</td>
<td>91.6</td>
<td>14 3.5 25.2</td>
</tr>
<tr>
<td>3</td>
<td>14,790</td>
<td>13,250</td>
<td>90.0</td>
<td>8 4.4 13</td>
</tr>
</tbody>
</table>

CONCLUSIONS

The modified version of the ground-water-flow model duplicates exactly the results of the standard Trescott model for the sample problem and other test problems. The modified model permits the simulation of confining-bed and aquifer pinchouts without the use of artificial hydraulic parameters and eliminates wasted computer memory. Simulating a large number of nodes, a large percentage of which are inactive, is less expensive if the modified model is used. The savings over the Trescott model are problem dependent, that is, the savings are related to the number of inactive nodes in the simulation. Numerical experiments with the steady-state simulation of the northern Atlantic Coastal Plain regional aquifer study showed overall savings of 30 percent. In general, appreciable savings in simulating regional aquifer systems with complex external boundaries can be realized by the modified model.

REFERENCES

Bennett, G. D., 1979, Regional ground-water systems analyses: Water Spectrum, v. 11, no. 4, p. 36-42.

REFERENCES—Continued


APPENDIX 1

DATA DECK INSTRUCTIONS

The data deck instructions have been made as compatible as possible with the Trescott version of the flow model. This allows existing data decks to be used with the modified model with minimal amount of recoding. The modified model also includes the transient leakage option (ENTRY CLAY) described by Posson, Hearne, Tracy and Frenzel (1980) and a modification to allow a user-specified maximum iteration parameter (WMAX) in ENTRY ITER. Because contouring packages for pen plotters are generally available, the line printer plotting routine (SUBROUTINE PRINTAI) was eliminated. The instructions that follow are adapted from Open-File Report 75-438.

Group I: Title, Simulation Options and Problem Dimensions

This group of cards, which are read by the main program, contains data required to dimension the model. To specify an option on card 4 punch the characters underlined in the definition. For an option not used, that section of card 4 can be left blank.

Note: Default typing of variables applies for all data input.

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-80</td>
<td>20A4</td>
<td>HEADING</td>
<td>Any title the user wishes to print on one line at the start of output</td>
</tr>
<tr>
<td>2</td>
<td>1-52</td>
<td>13A4</td>
<td>HEADING</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1-10</td>
<td>I10</td>
<td>IO</td>
<td>Number of rows</td>
</tr>
<tr>
<td></td>
<td>11-20</td>
<td>I10</td>
<td>JO</td>
<td>Number of columns</td>
</tr>
<tr>
<td></td>
<td>21-30</td>
<td>I10</td>
<td>KO</td>
<td>Number of layers</td>
</tr>
<tr>
<td></td>
<td>31-40</td>
<td>I10</td>
<td>ITMAX</td>
<td>Maximum number of iterations per time step</td>
</tr>
<tr>
<td></td>
<td>41-50</td>
<td>I10</td>
<td>NCH</td>
<td>Number of constant head nodes</td>
</tr>
<tr>
<td></td>
<td>51-60</td>
<td>I10</td>
<td>MODE</td>
<td>Number of terms used in transient leakage code</td>
</tr>
<tr>
<td></td>
<td>61-70</td>
<td>I10</td>
<td>IOJO</td>
<td>Number of modeled nodes per layer</td>
</tr>
</tbody>
</table>
NOTE: IOJO is the total number of modeled nodes for a composite layer consisting of a combination of all layers considered together. In other words, the position of JDIML1 may be determined by the external geometry of Layer 1, whereas the position of JDIML2 may be determined by the external geometry of Layer 2. The variable IOJO would be the number of modeled nodes in this composite layer.

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1-4</td>
<td>A4</td>
<td>IDRAW</td>
<td>DRAW to print drawdown</td>
</tr>
<tr>
<td></td>
<td>6-9</td>
<td>A4</td>
<td>IHEAD</td>
<td>HEAD to print hydraulic head</td>
</tr>
<tr>
<td></td>
<td>11-14</td>
<td>A4</td>
<td>IFLOW</td>
<td>MASS to compute a mass balance</td>
</tr>
<tr>
<td></td>
<td>16-18</td>
<td>A3</td>
<td>IDK1</td>
<td>DK1 to read initial head, elapsed time, and mass balance parameters on unit 4 (disk)</td>
</tr>
<tr>
<td></td>
<td>21-23</td>
<td>A3</td>
<td>IDK2</td>
<td>DK2 to write computed head, elapsed time, and mass balance parameters on unit 4 (disk)</td>
</tr>
<tr>
<td></td>
<td>26-29</td>
<td>A4</td>
<td>IWATER</td>
<td>WATE if the upper hydrologic unit is unconfined</td>
</tr>
<tr>
<td></td>
<td>31-34</td>
<td>A4</td>
<td>IQRE</td>
<td>RECH for a constant recharge that may be a function of space</td>
</tr>
<tr>
<td></td>
<td>36-39</td>
<td>A4</td>
<td>IPU1</td>
<td>PUN1 to read initial head, elapsed time, and mass balance parameters from cards</td>
</tr>
<tr>
<td></td>
<td>41-44</td>
<td>A4</td>
<td>IPU2</td>
<td>PUN2 to punch computed head, elapsed time, and mass balance parameters on cards</td>
</tr>
<tr>
<td></td>
<td>46-49</td>
<td>A4</td>
<td>ITK</td>
<td>ITKR to read the value of TK(I,J,K) for simulations in which confining layers are not represented by layers of nodes. TK(I,J,K) = Kv/b.</td>
</tr>
<tr>
<td></td>
<td>51-54</td>
<td>A4</td>
<td>IEQN</td>
<td>EQN3 to solve Equation 3</td>
</tr>
</tbody>
</table>
APPENDIX 1

DATA DECK INSTRUCTIONS--Continued

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>56-59</td>
<td>A4</td>
<td>ITL</td>
<td>ITLR to read values of Rate ( \frac{\text{ITL}}{\text{K}} ) ( \text{I}(\text{J},\text{K}-1) ) ( \text{M}(\text{I},\text{J},\text{K}-1) ), ( \text{SS}(\text{K}-1) )</td>
<td></td>
</tr>
</tbody>
</table>

Note: For continuation of a simulation, if transient leakage option (ITLR) is specified variables RM, XI, and DELT are written, punched or read on disk or cards dependent on options specified (DK1, DK2, PUN1, PUN2).

Group IA: Modeled nodes array

<table>
<thead>
<tr>
<th>ARRAY</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-80</td>
<td>8I10</td>
<td>JDIML1(IO)</td>
<td>Location of first modeled node in each row in x-direction</td>
</tr>
<tr>
<td>2</td>
<td>1-80</td>
<td>8I10</td>
<td>JDIML2(IO)</td>
<td>Location of last modeled node in each row in x-direction</td>
</tr>
</tbody>
</table>

NOTE: The first modeled node may not appear in column 1 and may not appear after column 40.

Group II: Scalar parameters

The parameters required in every problem are underlined. The other parameters are required as noted; when not required, their location on the card can be left blank. The G format is used to read E, F and I format data. Minimize mistakes by always right-justifying data in the field. If F format data do not contain significant figures to the right of the decimal point, the decimal point can be omitted.

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-10</td>
<td>G10.0</td>
<td>NPER</td>
<td>Number of pumping periods for the simulation</td>
</tr>
<tr>
<td>11-20</td>
<td>G10.0</td>
<td>KTH</td>
<td>Number of time steps between printouts</td>
<td></td>
</tr>
</tbody>
</table>

NOTE: To print only the results for the final time step in a pumping period, make KTH greater than the expected number of time steps. The program always prints the results for the final time step.

| 21-30 | G10.0 | ERR    | Error criteria for closure (L) |
NOTE: When the head change at all nodes on subsequent iterations is less than this value (for example, 0.01 foot), the program has converged to a solution for the time step.

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>31-40</td>
<td>G10.0</td>
<td>LENGTH</td>
<td>Number of iteration parameters</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1-20</td>
<td>G20.10</td>
<td>SUM</td>
<td></td>
</tr>
<tr>
<td>21-40</td>
<td>G20.10</td>
<td>SUMP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>41-60</td>
<td>G20.10</td>
<td>PUMPT</td>
<td>Parameters in which elapsed time and cumulative volumes for mass balance are stored. For the start of a simulation insert three blank cards. For continuation of a previous run using cards as input, replace the three blank cards with the first three cards of punched output from the previous run. Using data from disk for input, leave the three blank cards in the data deck.</td>
<td></td>
</tr>
<tr>
<td>61-80</td>
<td>G20.10</td>
<td>CFLUXT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1-20</td>
<td>G20.10</td>
<td>QRET</td>
<td></td>
</tr>
<tr>
<td>21-40</td>
<td>G20.10</td>
<td>CHST</td>
<td></td>
<td></td>
</tr>
<tr>
<td>41-60</td>
<td>G20.10</td>
<td>CHDT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>61-80</td>
<td>G20.10</td>
<td>FLUXT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1-20</td>
<td>G20.10</td>
<td>STORT</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>21-40</td>
<td>G20.10</td>
<td>ETFLXT</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>41-60</td>
<td>G20.10</td>
<td>FLXNT</td>
<td></td>
</tr>
</tbody>
</table>

Group III: Array Data

Each of the following data sets (except data set 1) consists of a parameter card and, if the data set contains variable data, a set of data cards for each layer in the model. Each parameter card contains at least five variables. The final card in Group III is not a part of an array data set but rather a single card defining the maximum iteration parameter.
### APPENDIX 1

#### DATA DECK INSTRUCTIONS--Continued

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Every 1-10 Parameter Card</td>
<td>1-10</td>
<td>G10.0</td>
<td>FAC</td>
<td>If IVAR = 0, FAC is the value assigned to every element of the matrix for this layer. If IVAR = 1, FAC is the multiplication factor for the following set of data cards for this layer.</td>
</tr>
<tr>
<td>11-20</td>
<td>G10.0</td>
<td>IVAR</td>
<td>= 0 if no data cards are to be read for this layer. = 1 if data cards for this layer follow.</td>
<td></td>
</tr>
<tr>
<td>21-30</td>
<td>G10.0</td>
<td>IPRN</td>
<td>= 0 if input data for this layer are to be printed. = 1 if input data for the layer are not to be printed.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transmissivity 31-40 Parameters Cards also have these Variables</th>
<th>G10.0</th>
<th>FACT(K,1)</th>
<th>multiplication factor for transmissivity in x-direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>41-50</td>
<td>G10.0</td>
<td>FACT(K,2)</td>
<td>multiplication factor for transmissivity in the y-direction</td>
</tr>
<tr>
<td>51-60</td>
<td>G10.0</td>
<td>FACT(K,3)</td>
<td>multiplication factor for hydraulic conductivity in the z-direction. (Not used when confining bed nodes are eliminated and TK values are read)</td>
</tr>
</tbody>
</table>
APPENDIX 1

DATA DECK INSTRUCTIONS--Continued

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>61-70</td>
<td>G10.0</td>
<td>IRECS</td>
<td>= 0 if the matrix is being read from cards or if each element is being set equal to FAC.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>= 1 if the matrix is to be read from disk (unit 2)</td>
</tr>
<tr>
<td></td>
<td>71-80</td>
<td>G10.0</td>
<td>IRECD</td>
<td>= 0 if the matrix is not to be stored on disk.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>= 1 if the matrix being read from cards or set equal to FAC is to be stored on disk (unit 2) for later retrieval.</td>
</tr>
</tbody>
</table>

When data cards are included, start each row on a new card.

To prepare a set of data cards for an array that is a function of space, the general procedure is to overlay the finite-difference grid on a contoured map of the parameter and record the average value of the parameter for each finite-difference block on coding forms according to the appropriate format. In general, record only significant digits and no decimal points (except for data set 2); use the multiplication factor to convert the data to their appropriate values. For example, if DELX ranges from 1,000 to 15,000 feet, coded values should range from 1-15; the multiplication factor (FAC) would be 1,000.

<table>
<thead>
<tr>
<th>DATA SET</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-80</td>
<td>8F10.4</td>
<td>PHI(IJ,K)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Head values for continuation of a previous run (L)</td>
</tr>
</tbody>
</table>

NOTE: For a new simulation this data set is omitted. Do not include a parameter card with this data set.

| 2        | 1-80    | 8F10.4 | START(IJ,K) |
|          |         |        |             |
| 3        | 1-80    | 20F4.0 | S (IJ,K)    |
|          |         |        |             |

NOTE: This matrix is also used to locate constant head boundaries by coding a negative number at constant head nodes.
### APPENDIX 1

#### DATA DECK INSTRUCTIONS—Continued

<table>
<thead>
<tr>
<th>DATA SET</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1-80</td>
<td>20F4.0</td>
<td>T(IJ,K)</td>
<td>Transmissivity (L$^2$/t)</td>
</tr>
</tbody>
</table>

**NOTE:**
1) See the previous page for the additional requirements on the parameter cards for this data set.

2) If the upper modeled layer is unconfined and PERM and BOTTOM are to be read for this layer, insert a parameter card for this layer with only the values for FACT on it.

3) Zero values may be used to represent aquifer pinchouts. Finite values of TK will allow flow directly from confining bed to confining bed in these areas.

| 5        | 1-80    | 8F10.4 | TK(IJ,K)     | $K_y/b$ leakance (/T).                          |

**NOTE:** This data set is read for both steady leakage(ITKR) and transient leakage(ITLR) options. If the Transient Leakage Option is specified, zero values must be specified for TK. The number of layers of TK values is KO-1.

| 6        | 1-80    | 20F4.0 | PERM(IJ)     | Hydraulic conductivity (L/T)                   |
| 7        | 1-80    | 20F4.0 | BOTTOM(IJ)   | Elevation of bottom of water-table unit (L)     |

**NOTE:** Data set 6 and 7 are required only for simulating unconfined conditions in the upper hydrologic unit.

| 8        | 1-80    | 20F4.0 | QRE(IJ)      | Recharge rate (L/T)                            |

**NOTE:** Omit if not used

| 9        | 1-80    | 20F4.0 | RATE(IJ,K)   | Vertical hydraulic conductivity of confining bed ($K'$) (L/T) |
| 10       | 1-80    | 20F4.0 | ZCB(IJ,K)    | Thickness of confining bed (L)                  |
DATA DECK INSTRUCTIONS--Continued

<table>
<thead>
<tr>
<th>SET</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>1-80</td>
<td>8G10.0</td>
<td>SS(K)</td>
<td>Specific storage of confining bed (/L)</td>
</tr>
</tbody>
</table>

NOTE: Data sets 9, 10, and 11 are required for simulating transient leakage from confining beds. If steady leakage is required for a particular confining bed in the simulation, specific storage for the particular confining bed can be specified as zero, or rate can be specified as zero and a finite value of TK for the confining bed can be used.

<table>
<thead>
<tr>
<th>SET</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>1-80</td>
<td>8G10.0</td>
<td>DELX(J)</td>
<td>Grid spacing in x direction (L)</td>
</tr>
<tr>
<td>13</td>
<td>1-80</td>
<td>8G10.0</td>
<td>DELY(I)</td>
<td>Grid spacing in y direction (L)</td>
</tr>
<tr>
<td>14</td>
<td>1-80</td>
<td>8G10.0</td>
<td>DELZ(K)</td>
<td>Grid spacing in z direction (L)</td>
</tr>
<tr>
<td>1-10</td>
<td>F10.7</td>
<td></td>
<td>WMAX</td>
<td>Maximum iteration parameter (optimum value can be determined by trial and error for each problem, 0.99863 is a good first guess)</td>
</tr>
</tbody>
</table>

Group IV: Parameters that change with the pumping period

The program has two options for the simulation period:

1. To simulate a given number of time steps, set TMAX to a value larger than the expected simulation period. The program will use NUMT, CDLT, and DELT as coded.

2. To simulate a given pumping period, set NUMT larger than the number required for the simulation period (for example, 100). The program will compute the exact DELT (which will be < DELT coded) and NUMT to arrive exactly at TMAX on the last time step.
APPENDIX 1

DATA DECK INSTRUCTIONS--Continued

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-10</td>
<td>G10.0</td>
<td>KP</td>
<td>Number of the pumping period</td>
</tr>
<tr>
<td>11-20</td>
<td></td>
<td>G10.0</td>
<td>KPM1</td>
<td>Number of the previous pumping period</td>
</tr>
</tbody>
</table>

NOTE: KPM1 is currently not used

| 21-30|         | G10.0  | NWEL     | Number of wells for this pumping period         |
| 31-40|         | G10.0  | TMAX     | Number of days in this pumping period           |
| 41-50|         | G10.0  | NUMT     | Number of time steps                            |
| 51-60|         | G10.0  | CDLT     | Multiplying factor for DELT                     |

NOTE: 1.5 is commonly used

| 61-70|         | G10.0  | DELT     | Initial time step in hours                     |

if NWEL = 0 the following set of cards is omitted

DATA SET 1 (NWEL cards)

<table>
<thead>
<tr>
<th>CARD</th>
<th>COLUMNS</th>
<th>FORMAT</th>
<th>VARIABLE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-10</td>
<td></td>
<td>G10.0</td>
<td>K</td>
<td>Layer in which well is located</td>
</tr>
<tr>
<td>11-20</td>
<td></td>
<td>G10.0</td>
<td>I</td>
<td>Row location of well</td>
</tr>
<tr>
<td>21-30</td>
<td></td>
<td>G10.0</td>
<td>J</td>
<td>Column location of well</td>
</tr>
<tr>
<td>31-40</td>
<td></td>
<td>G10.0</td>
<td>WELL(IJ,K)</td>
<td>Pumping rate (L^3/t), negative for a pumping well.</td>
</tr>
</tbody>
</table>

For each additional pumping period, another set of group IV cards is required (that is, NPER sets of group IV cards are required).
MODIFIED FLOW MODEL TO REDUCE CORE STORAGE, INCLUDES TRANSIENT LEAKAGE
JUNE 1982

SPECIFICATIONS:
REAL *8YSTR
DIMENSION Y(20000), L(34), HEADNG(33), NAME(54), DUM(3)
1, LL(2), JD(100)

EQUIVALENCE (YSTR,Y(1))

COMMON /INTEGR/ 10,JO,KO,ITMAX,LENGTH,KP,NWEL,
1NUNT,IFINAL,IT,KT,IHEAD,IDRAW,IFLO,IERR,I2,J2,K2,IMAX,ITMX1,NCH,
2IDK1,IDK2,IWATER,IQRE,IPJP,IQJQ,IKJK,K5,IPU1,IPU2,ITK,IEQN,MODE
.,ILT,IOJ0,K6,K7,ILJL
COMMON /SPARAM/ TMAX,CDLT,DELT,ERR,TEST,SUM,SUMP,QR,DELM1

DATA NAME/2*4H , 4H S,4HTART,4HING ,4HHEAD,4H , 4H STO,4HRAG000000070
1E,4H COE,4HFFIC,4HIENT,2*4H ,4H TR,4HANSM,4HISSI,4HVITY,5*4H
2 ,4H TK,4H HY,4HDRAU,4HLIC ,4HCOND,4HUCTI,4HVITY,2*4H ,4HBO
3T,4HOM E,4HLEVA,4HTION,2*4H ,4H R,4HECHA,4HRGE ,4HRATE,4H A
4,4HQUIT,4HARD ,4HCOND,4HUCTI,4HVITY,4H ,4H AQ,4HUITA,4HRD T,4H
5HICK,4HNESS/

DEFINE FILE 2(8,1520,U,KKK)
ILENGY=20000
JLENG=100

---READ TITLE, PROGRAM SIZE AND OPTIONS---
READ (5,200) HEADNG
WRITE (6,190) HEADNG
READ (5,160) 10,JO,KO,ITMAX,NCH,MODE,IOJ0
WRITE (6,180) 10,JO,KO,ITMAX,NCH,MODE,IOJ0
READ (5,210) IDRAW,IHEAD,IFLO,IDK1,IDK2,IWATER,IQRE,IPU1,IPU2,ITK
1,IEQN,ITL
WRITE (6,220) IDRAW,IHEAD,IFLO,IDK1,IDK2,IWATER,IQRE,IPU1,IPU2,ITK
1,IEQN,ITL
IERR=0

---COMPUTE DIMENSIONS FOR ARRAYS---
J1=J0-1
I1=I0-1
K1=K0-1
APPENDIX 2
LISTING OF SOURCE CODE--Continued

\begin{verbatim}
I2=I0-2
J2=J0-2
K2=K0-2
IMAX=MAXO(I0,J0)
NCD=MAXO(1,NCH)
ITMX1=ITMAX+1
ISIZ=I0J0*K0
IK1=I0J0
IK2=MAXO(IK1*K1,1)
ISUM=2*ISIZ+1
L(1)=1
DO 30 I=2,14
   IF (I.NE.8) GO TO 20
   L(8)=ISUM
   ISUM=ISUM+IK2
   IF (IK2.EQ.1) GO TO 10
   IKJK=IK1
   K5=K1
   GO TO 30
10 IKJK=1
   K5=1
   GO TO 30
20 L(I)=ISUM
   ISUM=ISUM+ISIZ
30 CONTINUE
   L(15)=ISUM
   ISUM=ISUM+J0
   L(16)=ISUM
   ISUM=ISUM+I0
   L(17)=ISUM
   ISUM=ISUM+K0
   L(18)=ISUM
   ISUM=ISUM+IMAX
   L(19)=ISUM
   ISUM=ISUM+K0*3
   L(20)=ISUM
   ISUM=ISUM+ITMX1
   L(21)=ISUM
   ISUM=ISUM+3*NCD
   L(22)=ISUM
   ISUM=ISUM+NCD
   L(23)=ISUM
   IF (IWATER.NE.ICHK(6)) GO TO 40
   ISUM=ISUM+IK1
   L(24)=ISUM
   ISUM=ISUM+IK1
   IPJP=IK1
   GO TO 50
40 ISUM=ISUM+1
   L(24)=ISUM
   ISUM=ISUM+1
\end{verbatim}
APPENDIX 2
LISTING OF SOURCE CODE--Continued

IPJP=1
50 L(25)=ISUM
   IF (IQRE.NE.ICHK(7)) GO TO 60
   ISUM=ISUM+IK1
   IQJQ=IK1
   GO TO 70
60 ISUM=ISUM+1
   IQJQ=1
   L(26) = ISUM
   IF ( ITL.NE.ICHK(12) ) GO TO 75
   ILJL=IK1
   K6=K0
   K7=K1
   ISUM = ISUM + IK2
   L(27) = ISUM
   ISUM = ISUM + IK2
   L(28) = ISUM
   ISUM = ISUM + K1
   L(29) = ISUM
   ISUM = ISUM + ISIZ
   L(30) = ISUM
   ISUM = ISUM + ISIZ
   L(31) = ISUM
   ISUM = ISUM + ISIZ
   L(32) = ISUM
   ISUM = ISUM + 2*MODE*IK2
   GO TO 77
70 ILJL=1
   MODE=1
   K6=1
   K7=1
   ISUM=ISUM+1
   L(27)=ISUM
   ISUM=ISUM+1
   L(28)=ISUM
   ISUM=ISUM+1
   L(29)=ISUM
   ISUM=ISUM+1
   L(30)=ISUM
   ISUM=ISUM+1
   L(31)=ISUM
   ISUM=ISUM+1
   L(32)=ISUM
   ISUM=ISUM+2
   JSUM = IO
   LL(1)=1
   LL(2)=JSUM+1
   JSUM = JSUM + IO
   WRITE (6,170) ISUM,ILENGY
   WRITE (6,175) JSUM,JLENG
   IF (ISUM.GT.ILENGY) GO TO 155

25
---PASS INITIAL ADDRESSES OF ARRAYS TO SUBROUTINES---
CALL DATAI(Y(L(1)), Y(L(2)), Y(L(3)), Y(L(4)), Y(L(5)), Y(L(6)), Y(L(7)), Y(L(8)), Y(L(9)), Y(L(10)), Y(L(11)), Y(L(12)), Y(L(13)), Y(L(14)), Y(L(15)), Y(L(16)), Y(L(17)), Y(L(18)), Y(L(19)), Y(L(20)), Y(L(21)), Y(L(22)), Y(L(23)), Y(L(24)), Y(L(25)), Y(L(26)), Y(L(27)), Y(L(28)), Y(L(29)), Y(L(30)), Y(L(31)), Y(L(32)), JD(LL(1)), JD(LL(2)))

CALL STEP(Y(L(1)), Y(L(2)), Y(L(3)), Y(L(4)), Y(L(5)), Y(L(6)), Y(L(7)), Y(L(8)), Y(L(9)), Y(L(10)), Y(L(11)), Y(L(12)), Y(L(13)), Y(L(14)), Y(L(15)), Y(L(16)), Y(L(17)), Y(L(18)), Y(L(19)), Y(L(20)), Y(L(21)), Y(L(22)), Y(L(23)), JD(LL(1)), JD(LL(2)))

CALL SOLVE(Y(L(1)), Y(L(2)), Y(L(3)), Y(L(4)), Y(L(5)), Y(L(6)), Y(L(7)), Y(L(8)), Y(L(9)), Y(L(10)), Y(L(11)), Y(L(12)), Y(L(13)), Y(L(14)), Y(L(15)), Y(L(16)), Y(L(17)), Y(L(18)), Y(L(19)), Y(L(20)), Y(L(21)), Y(L(22)), Y(L(23)), JD(LL(1)), JD(LL(2)))

CALL COEF(Y(L(1)), Y(L(2)), Y(L(3)), Y(L(4)), Y(L(5)), Y(L(6)), Y(L(7)), Y(L(8)), Y(L(9)), Y(L(10)), Y(L(11)), Y(L(12)), Y(L(13)), Y(L(14)), Y(L(15)), Y(L(16)), Y(L(17)), Y(L(18)), Y(L(19)), Y(L(20)), Y(L(21)), Y(L(22)), Y(L(23)), JD(LL(1)), JD(LL(2)))

CALL CHECKI(Y(L(1)), Y(L(2)), Y(L(3)), Y(L(4)), Y(L(5)), Y(L(6)), Y(L(7)), Y(L(8)), Y(L(9)), Y(L(10)), Y(L(11)), Y(L(12)), Y(L(13)), Y(L(14)), Y(L(15)), Y(L(16)), Y(L(17)), Y(L(18)), Y(L(19)), Y(L(20)), Y(L(21)), Y(L(22)), Y(L(23)), JD(LL(1)), JD(LL(2)))

---START COMPUTATIONS---
GROUPS II AND III
---READ AND WRITE DATA FOR GROUPS II AND III---
CALL DATAB
IRN=1
NIJ=10
DO 80 K=1,K0
LOC=L(2)+(K-1)*NIJ
CALL ARRAY(Y(LOC),2,1,NAME(1),IRN,DUM,K)
DO 90 K=1,K0
LOC=L(5)+(K-1)*NIJ
CALL ARRAY(Y(LOC),1,2,NAME(7),IRN,DUM,K)
DO 100 K=1,K0
LOC=L(4)+(K-1)*NIJ
L1=L(19)+K-1
L2=L(19)+K0+K-1
L3=L(19)+2*K0+K-1
CALL ARRAY(Y(LOC),1,2,NAME(13),IRN,DUM,K)
Y(L1)=DUM(1)
Y(L2)=DUM(2)
Y(L3)=DUM(3)
100 WRITE (6,230) K,Y(L1),Y(L2),Y(L3)
IF (ITK.NE.ICHK(10)) GO TO 120
DO 110 K=1,K1
LOC=L(8)+(K-1)*NIJ
110 CALL ARRAY(Y(LOC),2,3,NAME(19),IRN,DUM,K)
120 IF (IWATER.NE.ICHK(6)) GO TO 130
K=K0
CALL ARRAY(Y(L(23)),1,4,NAME(25),IRN,DUM,K)
CALL ARRAY(Y(L(24)),1,1,NAME(31),IRN,DUM,K)
APPENDIX 2
LISTING OF SOURCE CODE--Continued

130 IF (IQRE.EQ.ICHK(7)) CALL ARRAY(Y(L(25)),1,4,NAME(37),IRN,DUM,K)
   IF (ITL.NE.ICHK(12)) GO TO 135
   DO 131 K=1,K1
    LOC = L(26) + (K-1)*NIJ
   131 CALL ARRAY (Y(LOC),1,3,NAME(43),IRN,DUM,K)
   DO 132 K=1,K1
    LOG = L(27) + (K-1)*NIJ
   132 CALL ARRAY (Y(LOG),1,1,NAME(49),IRN,DUM,K)
   READ (5,240) FAC,IVAR,IPRN
   LOC = L(28)
   IF (IVAR.EQ.1) READ (5,240)(Y(LOG+K-1),K=1,K1)
   DO 133 K=1,K1
    IF (IVAR.NE.1) GO TO 134
    Y(LOG+K-1) = Y(LOG+K-1)*FAC
   133 CONTINUE
   134 Y(LOG+K-1) = FAC
   135 CONTINUE
   CALL MDAT

C ---COMPUTE TRANSMISSIVITY FOR UNCONFINED LAYER---
   IF (IWRITE.EQ.ICHK(6)) CALL TRANS(1)

C ---COMPUTE T COEFFICIENTS---
   CALL TCOF

C ---COMPUTE ITERATION PARAMETERS---
   CALL ITER

C ---READ TIME PARAMETERS AND PUMPING DATA FOR A NEW PUMPING PERIOD---
   CALL NEWPER
   KT=0
   IFINAL=0

C ---START NEW TIME STEP COMPUTATIONS---
   CALL NEWSTP

C ---START NEW ITERATION IF MAXIMUM NO. ITERATIONS NOT EXCEEDED---
   CALL NEWITA

C ---PRINT OUTPUT AT DESIGNATED TIME STEPS---
   CALL OUTPUT

C ---LAST TIME STEP IN PUMPING PERIOD---
   IF (IFINAL.NE.1) GO TO 150

C ---CHECK FOR NEW PUMPING PERIOD---
   IF (KP.LT.NPER) GO TO 140

27
APPENDIX 2
LISTING OF SOURCE CODE--Continued

155 STOP

---FORMATS---

160 FORMAT (8I10)                        00001870
170 FORMAT ('0',54X,'WORDS OF VECTOR Y USED =',I7,8X,' OF Y = ',I7) 00001890
175 FORMAT ('0',54X,'WORDS OF VECTOR JD USED =',I6,8X,' OF JD = ',I6) 0001900

190 FORMAT ('1',33A4)                    00001920
200 FORMAT (20A4)                       00001930
210 FORMAT (16(A4,1X))                  00001940
220 FORMAT ('SIMULATION OPTIONS: ',12(A4,4X))  
230 FORMAT (1H,44X,'DIRECTIONAL TRANSMISSIVITY MULTIPLICATION FACTORS:1 FOR LAYER',I3,/,46X,'Y =','1P1G15.7/76X,'Z =','1P1G15.7)

240 FORMAT (8G10.0)                     00001956
250 FORMAT (1H ,46X,'SPECIFIC STORAGE'/47X,40('-')//('0',1P1 8G1 5. 7)) 00001980
260 FORMAT ('0','55X,'ALL SPECIFIC STORAGES = ','1P1G15.7)             00001990
270 FORMAT (1H,46X,'POSITION OF FIRST ACTIVE NODE IN X DIRECTION'/  
47X,44('-')//('0',12I10))                   00002000
280 FORMAT (1H,46X,'POSITION OF LAST ACTIVE NODE IN X DIRECTION'/  
47X,43('-')//('0',12I10))                  00002010

END

SUBROUTINE DATA(PHI,STRT,OLD,T,S,TR,TC,TK,WELL,DELX,DELY,DELZ,FACT,KPHI,STRT,OLD,T,S,TR,TC,TK,WELL,DELX,DELY,DELZ,FACT,PERM,BOTTOM,QRE,XI,RM,JDIML1,JDIML2)

READ AND WRITE DATA

SPECIFICATIONS:
REAL *8PHI
REAL*4 IOFT1(12),INFT1(12)

DIMENSION PHI(IOJO,K0), STRT(IOJO,K0), OLD(IOJO,K0), T(IOJO,K0), S(IOJO,K0), TR(IOJO,K0), TC(IOJO,K0), TK(IKJK,K5), WELL(IOJO,K0), DELX(IO), DELY(IO), DELZ(K0), FACT(K0,3), PERM(IPJP), BOTTOM(IPJP), QRE(IQJQ), TF(3), A(IOJO), IN(6), XI(IOJO,K0), RM(2,MODE,4ILJL,K7), JDIML1(IO), JDIML2(IO)

COMMON /INTEGR/ IO,K0,I1,J1,K1,NPER,KTH,ITMAX,LENGTH,KP,NWEL,1NUMT,IFINAL,IT,KT,IHEAD,IDRAW,IFLO,IERR,I2,J2,K2,IMAX,ITMX1,NCH,2DK1,DK2,IWATER,IQRE,IPJP,IQJQ,IKJK,K5,IPU1,IPU2,ITK,IEQN,MODE.,ITL,IOJO,K6,K7,ILJL
COMMON /SPARAM/ TMAX,CDLT,DELT,ERR,TEST,SUM,SUMP,QR,DELM1
APPENDIX 2
LISTING OF SOURCE CODE--Continued

COMMON /SARRAY/ ICHK(13) 00002090
COMMON /CK/ ETFLXT,STORT,QRET,CHST,CHDT,FLUXT,PUMPT,CFLUXT,FLXNT 00002100
COMMON /PR/ DIGIT(129),VF4(12),VF5(12),VF6(12),VF7(12),VF8(12),VF9(12),VF10(12) 00002110
RETURN 00002120

********************
ENTRY DATAIN 00002130
********************

---READ AND WRITE LIMIT OF ACTIVE NODES IN X DIRECTION---
READ (5,330) (JDIML1(I),I=1,KO) 00002140
READ (5,330) (JDIML2(I),I=1,KO) 00002150
WRITE (6,460) (JDIML1(I),I=1,KO) 00002160
WRITE (6,470) (JDIML2(I),I=1,KO) 00002170

---READ AND WRITE SCALAR PARAMETERS---
READ (5,330) NPER,KTH,ERR,LENGTH 00002180
WRITE (6,340) NPER,KTH,ERR 00002190

---READ CUMULATIVE MASS BALANCE PARAMETERS---
READ (5,450) SUM,SUMP,PUMPT,CFLUXT,STORT,ETFL00002200
1XT,FLXNT,DELT
IF (IDK1.EQ.ICHK(4)) GO TO 20 00002210
IF (IPU1.NE.ICHK(8)) GO TO 50 00002220

---READ INITIAL HEAD VALUES FROM CARDS---
DO 15 K=1,KO 00002230
J4=1 00002240
DO 10 I=1,I0 00002250
JJ=J0-JDIML2(I) 00002260
VF4(4)=DIGIT(129) 00002270
ND=(JDIML1(I)-1)*10 00002280
ND1=ND/80 + 123 00002290
ND=ND-80*(ND1-123) 00002300
J5=JDIML2(I)-JDIML1(I)+J4 00002310
IF(NI.1.EQ.123) ND1=128 00002320
VF4(2)=DIGIT(ND1) 00002330
IF(NI.NE.0) GO TO 12 00002340
VF4(3)=DIGIT(128) 00002350
VF4(4)=DIGIT(128) 00002360
GO TO 11 00002370
12 VF4(3)=DIGIT(ND) 00002380
11 NE=(80-ND)/10 00002390
VF4(5)=DIGIT(NE) 00002400
READ (5,VF4) (PHI(IJ,K),IJ=J4,J5),(DELX(J),J=1, jj) 00002410
10 J4=J5+1 00002420
15 CONTINUE 00002430
IF(ITL.NE.ICHK(12)) GO TO 16 00002440
DO 17 K=1,KO 00002450
APPENDIX 2
LISTING OF SOURCE CODE—Continued

READ(5,330) (XI(IJ,K),IJ=1,IOJO) 00002460
17 CONTINUE 00002470
READ(5,480) (((RM(N,M,IJ,K),N=1,2),M=1,MODE),IJ=1,IILJL),K=1,K7) 00002480
16 CONTINUE 00002490
GO TO 30 00002500

C
---READ INITIAL HEAD AND MASS BALANCE PARAMETERS FROM DISK---
20 READ (4) PHI,SUM,SUMP,PUMPT,CFLUXT,QRET,CHST,CHDT,FLUXT,STORT,ETFLO00002510
1XT,FLXNT,DELT,XI,RM
REWIND 4 00002520
30 WRITE (6,430) SUM 00002530
DO 45 K=1,K0 00002540
J4=1 00002550
WRITE (6,440) K 00002560
DO 40 I=1,10 00002570
ND=(JDIML1(I)-1)*6 00002580
ND1=ND/120 + 123 00002590
VF5(2)=DIGIT(ND1) 00002600
ND=ND-120*(ND1-123) 00002610
NE=(120-ND)/6 00002620
J5=JDIML2(I)-JDIML1(I)+J4 00002630
IF(ND1,NE.123) ND=3+ND 00002640
ND=ND+2 00002650
VF5(3)=DIGIT(ND) 00002660
VF5(5)=DIGIT(NE) 00002670
WRITE (6,VF5) I,(PHI(IJ,K),IJ=J4,J5) 00002680
40 J4=J5+1 00002690
45 CONTINUE 00002700

C
50 DO 60 K=1,K0 00002710
DO 60 IJ=1,IOJO 00002720
WELL(IJ,K)=0. 00002730
TR(IJ,K)=0. 00002740
TC(IJ,K)=0. 00002750
IF (K.NE.KO) TK(IJ,K)=0. 00002760
60 CONTINUE 00002770
RETURN 00002780

C
ENTRY ARRAY(A,INFT,IOFT,IN,IRN,TF,K) 00002790

C
READ (5,330) FAC,IVAR,IPRN,TF,IRECS,IRECD 00002800
IC=4*IRECS+2*IVAR+IPRN+1 00002810
GO TO (70,70,90,90,120,120), IC 00002820
70 DO 80 IJ=1,IOJO 00002830
80 A(IJ)=FAC 00002840
WRITE (6,280) IN,FAC,K 00002850
GO TO 140 00002860
90 IF (IC.EQ.3) WRITE (6,290) IN,K 00002870
J4=1 00002880
DO 111 I=1,IO 00002890
JJ=JO-JDIML2(I) 00002900
APPENDIX 2
LISTING OF SOURCE CODE--Continued

IF (INFT.EQ.1) GO TO 91
VF4(4)=DIGIT(129)
ND=(JDIML1(I)-1)*10
ND1=ND/80 + 123
ND=ND-80*(ND1-123)
J5=JDIML2(I)-JDIML1(I)+J4
IF(ND1.EQ.123) ND1=128
VF4(2)=DIGIT(ND1)
IF(ND.NE.0) GO TO 71
VF4(3)=DIGIT(128)
VF4(4)=DIGIT(128)
GO TO 72
71 VF4(3)=DIGIT(ND)
72 NE=(80-ND)/10
VF6(5)=DIGIT(NE)
DO 89 KVF=1,12
INFT1(KVF)=VF4(KVF)
GO TO 92
91 ND=(JDIML1(I)-1)*4
VF6(4)=DIGIT(129)
ND1=ND/80 + 123
ND=ND-80*(ND1-123)
J5=JDIML2(I)-JDIML1(I)+J4
IF(ND1.EQ.123) ND1=128
VF6(2)=DIGIT(ND1)
IF(ND.NE.0) GO TO 73
VF6(3)=DIGIT(128)
VF6(4)=DIGIT(128)
GO TO 74
73 VF6(3)=DIGIT(ND)
74 NE=(80-ND)/4
IF(J0.LE.20) NE=NE+JJ
VF6(5)=DIGIT(NE)
DO 88 KVF=1,12
INFT1(KVF)=VF6(KVF)
92 READ (5,INFT1) (A(IJ),IJ=J4,J5),(DELX(J),J=1,JJ)
DO 100 IJ=J4,J5
A(IJ)=A(IJ)*FAC
GO TO (101,102,103,104),IOFT
100 A(IJ)=A(IJ)*FAC
101 ND=(JDIML1(I)-1)*6
ND1=ND/120 + 123
VF5(2)=DIGIT(ND1)
ND=ND-120*(ND1-123)
NE=(120-ND)/6
IF(ND1.NE.123) ND=3+ND
ND=ND+2
VF5(3)=DIGIT(ND)
VF5(5)=DIGIT(NE)
DO 87 KVF=1,12
INFT1(KVF)=VF5(KVF)
GO TO 110
87
APPENDIX 2
LISTING OF SOURCE CODE--Continued

102 ND=(JDIML1(I)-1)*9
    ND1=ND/126 + 123
    VF7(2)=DIGIT(ND1)
    ND=ND - 126*(ND1 - 123)
    NE=(126-ND)/9
    IF(ND1.NE.123) ND=6+ND
    VF7(3)=DIGIT(ND)
    VF7(5)=DIGIT(NE)
    DO 86 KVF=1,12
86 IOFT1(KVF)=VF7(KVF)
    GO TO 110
103 ND=(JDIML1(I)-1)*12
    ND1=ND/120 + 123
    VF8(2)=DIGIT(ND1)
    ND=ND - 120*(ND1 - 123)
    NE=(120-ND)/12
    IF(ND1.NE.123) ND=6+ND
    VF8(3)=DIGIT(ND)
    VF8(5)=DIGIT(NE)
    DO 85 KVF=1,12
85 IOFT1(KVF)=VF8(KVF)
    GO TO 110
104 ND=(JDIML1(I)-1)*11
    ND=ND/110 + 123
    VF9(2)=DIGIT(ND1)
    ND=ND - 110*(ND1 - 123)
    NE=(110-ND)/11
    IF(ND1.NE.123) ND=6+ND
    VF9(3)=DIGIT(ND)
    VF9(5)=DIGIT(NE)
    DO 105 KVF=1,12
105 IOFT1(KVF)=VF9(KVF)
    GO TO 110
110 IF (IC.EQ.3) WRITE (6,IOFT1) I,(A(I,J),IJ=J4,J5)
111 J4=J5+1
    GO TO 140
120 READ (2'IRN) A
    IF (IC.EQ.6) GO TO 140
    WRITE (6,290) IN,K
    J4=1
    DO 131 I=1,I0
    GO TO (121,122,123,124),IOFT
121 ND=(JDIML1(I)-1)*6
    ND1=ND/120 + 123
    VF5(2)=DIGIT(ND1)
    ND=ND - 120*(ND1 - 123)
    NE=(120-ND)/6
    IF(ND1.NE.123) ND=3+ND
    ND=ND+2
    VF5(3)=DIGIT(ND)
    VF5(5)=DIGIT(NE)
    DO 119 KVF=1,12
119 IF (IC.EQ.3) WRITE (6,IOFT1) I,(A(I,J),IJ=J4,J5)
120 READ (2'IRN) A
131 END
APPENDIX 2
LISTING OF SOURCE CODE--Continued

119 IOFT1(KVF)=VF5(KVF)
   GO TO 130
122 ND=(JDIML1(I)-1)*9
    ND1=ND/126 + 123
    VF7(2)=DIGIT(ND1)
    ND=ND-126*(ND1-123)
    NE=(126-ND)/9
    IF(ND1.NE.123) ND=6+ND
    VF7(3)=DIGIT(ND)
    VF7(5)=DIGIT(NE)
    DO 118 KVF=1,12
118 IOFT1(KVF)=VF7(KVF)
   GO TO 130
123 ND=(JDIML1(I)-1)*12
    ND1=ND/120 + 123
    VF8(2)=DIGIT(ND1)
    ND=ND-120*(ND1-123)
    NE=(120-ND)/12
    IF(ND1.NE.123) ND=6+ND
    VF8(3)=DIGIT(ND)
    VF8(5)=DIGIT(NE)
    DO 117 KVF=1,12
117 IOFT1(KVF)=VF8(KVF)
   GO TO 130
124 ND=(JDIML1(I)-1)*11
    ND1=ND/110 + 123
    VF9(2)=DIGIT(ND1)
    ND=ND-110*(ND1-123)
    NE=(110-ND)/11
    IF(ND1.NE.123) ND=6+ND
    VF9(3)=DIGIT(ND)
    VF9(5)=DIGIT(NE)
    DO 125 KVF=1,12
125 IOFT1(KVF)=VF9(KVF)
130 WRITE (6,IOFT1) I,(A(IJ),IJ=J4,J5)
131 J4=J5+1
140 IF (IRECD.EQ.1) WRITE (2'IRN) A
    IRN=IRN+1
    RETURN
C
************MDAT************
C
ENTRY MDAT
************MDAT************
C
DO 155 K=1,K0
   J4=1
   DO 152 I=1,I0
   J5=JDIML2(I)-JDIML1(I)+J4
   DO 150 IJ=J4,J5
      IF (I.EQ.1.OR.I.EQ.IO) T(IJ,K)=0.
      IF (IDK1.NE.ICHK(4).AND.IP11.NE.ICHK(8)) PHI(IJ,K)=STRT(IJ,K)
      IF (K.NE.K0.OR.IWATER.NE.ICHK(6)) GO TO 150
   IF (I.EQ.1.OR.I.EQ.IO) PERM(IJ)=0.
   C   C   C   C   C   C
APPENDIX 2
LISTING OF SOURCE CODE--Continued

150 CONTINUE
152 J4=J5+1
155 CONTINUE
C

153

DO 153 J=1,JO
153 DELX(J)=0.
READ (5,330) FAC,IVAR,IPRN
IF (IVAR.EQ.1) READ (5,330) (DELX(J),J=1,JO)
DO 170 J=1,JO
IF (IVAR.NE.1) GO TO 160
DELX(J)=DELX(J)*FAC
GO TO 170
160 DELX(J)=FAC
170 CONTINUE

IF (IVAR.EQ.1.AND.IPRN.NE.1) WRITE (6,370) (DELX(J),J=1,JO)
IF (IVAR.EQ.0) WRITE (6,300) FAC
C

READ (5,330) FAC,IVAR,IPRN
IF (IVAR.EQ.1) READ (5,330) (DELY(I),I=1,IO)
DO 190 I=1,IO
IF (IVAR.NE.1) GO TO 180
DELY(I)=DELY(I)*FAC
GO TO 190
180 DELY(I)=FAC
190 CONTINUE

IF (IVAR.EQ.1.AND.IPRN.NE.1) WRITE (6,380) (DELY(I),I=1,IO)
IF (IVAR.EQ.0) WRITE (6,310) FAC
C

READ (5,330) FAC,IVAR,IPRN
IF (IVAR.EQ.1) READ (5,330) (DELZ(K),K=1,KO)
DO 210 K=1,KO
IF (IVAR.NE.1) GO TO 200
DELZ(K)=DELZ(K)*FAC
GO TO 210
200 DELZ(K)=FAC
210 CONTINUE

IF (IVAR.EQ.1.AND.IPRN.NE.1) WRITE (6,390) (DELZ(K),K=1,KO)
IF (IVAR.EQ.0) WRITE (6,320) FAC
C

---INITIALIZE VARIABLES---
B=0.
D=0.
F=0.
H=0.
SU=0.
Z=0.
RETURN
C

---READ TIME PARAMETERS AND PUMPING DATA FOR A NEW PUMPING PERIOD---
ENTRY NEWPER

***************

DELTM1 = DELT
READ (5,330) KP,KPM1,NWEL,TMAX,NUMT,CDLT,DELT
IF (KP.GT.1) GO TO 225
IF(ITL.EQ.ICHK(12).AND.(IDK1.EQ.ICHK(4).OR.IPU1.EQ.ICHK(8))) GO TO 225
DELTM1 = 1.0
DO 215 K=1,KO
DO 215 IJ=1,IOJO
215 XI(IJ,K) = 0.0

---COMPUTE ACTUAL DELT AND NUMT---
225 DT=DELT/24.
TM=0.0
DO 220 I=1,NUMT
DT=CDLT*DT
TM=TM+DT
IF (TM.GE.TMAX) GO TO 230
220 CONTINUE
GO TO 240
230 DELT=TMAX/TM*DELT
NUMT=I
240 WRITE (6,400) KP,TMAX,NUMT,DELT,CDLT
DELT=DELT*3600.
TMAX=TMAX*86400.
SUMP=0.0

---READ AND WRITE WELL PUMPING RATES---
WRITE (6,410) NWEL
IF (NWEL.EQ.0) GO TO 260
DO 245 K=1,KO
DO 245 IJ=1,IOJO
245 WELL(IJ,K)=0.0
DO 250 I=1,NWEL
READ (5,330) K,I,J,TWELL
IJ=0
J4=1
IV=I-1
DO 246 III=1,IV
J5=JDIML2(III)-JDIML1(III)+J4
246 J4=J5+1
IJ=J4+J-JDIML1(I)
WRITE (6,420) K,I,J,TWELL
250 WELL(IJ,K)=TWELL/(DELX(J)*DELY(I))
260 RETURN

---FORMATS---
APPENDIX 2
LISTING OF SOURCE CODE—Continued

280 FORMAT ('1H0,52X,6A4,','=',1P1G15.7,',' FOR LAYER',I3) 00005240
290 FORMAT ('1H1,45X,6A4,',' MATRIX, LAYER',I3/46X,41('-')) 00005250
300 FORMAT ('0',72X,'DELX =',1P1G15.7) 00005260
310 FORMAT ('0',72X,'DELY =',1P1G15.7) 00005270
320 FORMAT ('0',72X,'DELZ =',1P1G15.7) 00005280
330 FORMAT (8G10.0) 00005290
340 FORMAT ('0',51X,'NUMBER OF PUMPING PERIODS =',I5/49X,'TIME STEPS BETWEEN PRINTOUTS =',I5//51X,'ERROR CRITERIA FOR CLOSURE =',1P1G15.7/)
370 FORMAT ('1H1,46X,4OHGRID SPACING IN PROTOTYPE IN X DIRECTION/47X,4000005310 1(''-')/('0',12F10.0)) 00005360
380 FORMAT ('1H1,46X,4OHGRID SPACING IN PROTOTYPE IN Y DIRECTION/47X,4000005320 1('''-')/('0',12F10.0)) 00005390
390 FORMAT ('1H1,46X,4OHGRID SPACING IN PROTOTYPE IN Z DIRECTION/47X,4000005330 1(''-')/('0',12F10.0)) 00005390
400 FORMAT ('1H1,50X,'PUMPING PERIOD NO.',I4,':',F10.2,' DAYS'/51X,38('00005401-')/53X,'NUMBER OF TIME STEPS=',I6//59X,'DELT IN HOURS =',F10.3// 253X,'MULTIPLIER FOR DELT =',F10.3)
410 FORMAT ('1H1,63X,I4,' WELLS=65X,9('-')/5X,5X,'K ','9X,' I ',9X,' J PUMPING RATE'/) 00005350
420 FORMAT (41X,3I10,2F13.2) 00005360
430 FORMAT ('1H1,40X,' CONTINUATION - HEAD AFTER ',G20.7,' SEC PUMPING 00005370 1(''-')/('0',42X,58('-'))) 00005370
440 FORMAT ('1H1,55X,'INITIAL HEAD MATRIX, LAYER',I3/56X,30('-')) 00005530
450 FORMAT (4G20.10) 00005390
460 FORMAT (1H1,46X,'POSITION OF FIRST ACTIVE NODE IN THE X DIRECTION'/47X,4000005400 1(''-')/('0',12I10)) 00005400
470 FORMAT (1H0,46X,'POSITION OF LAST ACTIVE NODE IN THE X DIRECTION'/ 47X,43('''-')/('0',12I10)) 00005410
480 FORMAT (6E13.6) 00005420
END
SUBROUTINE STEP(PHI,STRT,OLD,T,S,TR,TC,TK,WELL,DELX,DELY,DELZ,FACT,DDN,TEST3,XI,RM,JDIML1,JDIML2)

INITIALIZE DATA FOR A NEW TIME STEP AND PRINT RESULTS

SPECIFICATIONS:
REAL *8PHI
DIMENSION PHI(IOJO,KO), STRT(IOJO,KO), OLD(IOJO,KO), T(IOJO,KO), S(IOJO,KO), TR(IOJO,KO), TC(IOJO,KO), TK(IKJK,K5), WELL(IOJO 2,KO), DELX(JO), DELY(IO), DELZ(KO), FACT(KO,3), DDN(IMAX), TEST3 3(ITMX1), ITTO(50)
4, XI(IOJO,KO), RM(2,MODE,ILJL,K7), JDIML1(IO), JDIML2(IO)

COMMON /INTEGR/ IO,J0,K0,I1,J1,K1,NPER,KTH,ITMAX,LENGTH,KP,NWEL, 1NUMT,IFINAL,IT,KI,HEAD,IDRAW,IFLO,IERR,I2,J2,K2,IMAX,ITMX1,NCH, 2IDK1,IKD2,IFWATER,IFRE,IPJP,IQJQ,IKJK,K5,IPU1,IPU2,ITK,IEQN,MODE

36
APPENDIX 2
LISTING OF SOURCE CODE--Continued

.,ITL,I0J0,K6,K7,ILJL
COMMON /SPARAM/ TMAX,CDLT,DELT,ERR,TEST,SUM,SUMP,QR,DELM1
COMMON /SARRAY/ ICHK(13)
COMMON /CK/ ETFLXT,STORT,QRET,CHST,CHDT,FLUXT,PUMPT,CFLUXT,FLXNT
COMMON /PR/ DIGIT(129),VF4(12),VF5(12),VF6(12),VF7(12),VF8(12),VF9(12),VF10(12)
RETURN

ENTRY NEWSTP
KT=KT+1
IT=0
IF (KT.GT.1) DELTM1 = DELT
DELT=CDLT*DELT
SUM=SUM+DELT
SUMP=SUMP+DELT
DAYSP=SUMP/86400.
YRSP=DAYSP/365.
HRS=SUM/3600.
SMIN=HRS*60.
DAYS=HRS/24.
YRS=DAYS/365.

**COMPUTE TRANSIENT LEAKAGE COEFFICIENTS**
IF (ITL.NE.ICHKC12)) GO TO 11
CALL CLAY
DO 10 K=1,KO
DO 10 IJ=1,IOJO
XI(IJ,K) = 0.0
OLD(IJ,K)=PHI(IJ,K)
RETURN

**PRINT OUTPUT AT DESIGNATED TIME STEPS**
ENTRY OUTPUT
IF (KT.EQ.NUMT) IFINAL=1
ITTO(KT)=IT
IF (IT.LE.ITALM) GO TO 20
IT=IT-1
ITTO(KT)=IT
IERR=2
IF MAXIMUM ITERATIONS EXCEEDED,WRITE RESULTS ON DISK OR CARDS--
IF (IDK2.EQ.ICHK(5)) WRITE (4) PHI,SUM,SUMP,PUMPT,CFLUXT,QRET,CHST00005800
1,CHDT,FLUXT,STORT,ETFLXT,FLXNT,DEL,T,XI,RM
IF (IPU2.EQ.ICHK(9)) WRITE (7,230) SUM,SUMP,PUMPT,CFLUXT,QRET,CHST00005810
1,CHDT,FLUXT,STORT,ETFLXT,FLXNT,DEL,T
20 IF (IFLO.EQ.ICHKC3)) CALL CHECK
IF (IERR.EQ.2) GO TO 30

20 IF (IFLO.EQ.ICHKC3)) CALL CHECK
IF (IERR.EQ.2) GO TO 30
APPENDIX 2
LISTING OF SOURCE CODE--Continued

IF (MOD(KT,KTH).NE.0.AND.IFINAL.NE.1) RETURN
30 WRITE (6,210) KT,DELT,SUM,SMIN,HRS,DAYS,YRS,DAYSP,YRSP
IF (IFLO.EQ.ICHK(3)) CALL CWRITE
IT=IT+1
WRITE (6,180) (TEST3(J),J=1,IT)
I3=1
I5=0
352 I5=I5+40
I4=MINO(KT,I5)
WRITE (6,240) (I,I=I3,I4)
WRITE (6,260)
WRITE (6,250) (ITTO(I),I=I3,I4)
WRITE (6,260)
IF(KT.LE.I5) GO TO 353
I3=I3+40
GO TO 352
C
353 IF (IDRAW.NE.ICHK(1)) GO TO 100
C
---PRINT DRAWDOWN---
DO 90 K=1,K0
J4=1
WRITE (6,200) K
DO 90 I=1,I0
J=0
J5=JDIML2(I)-JDIML1(I)+J4
ND=(JDIML1(I)-1)*7
ND1=ND/126 + 123
VF10(2)=DIGIT(ND1)
ND=ND-126*(ND1-123)
NE=(126-ND)/7
IF(ND1.NE.123) ND=5+ND
VF10(3)=DIGIT(ND)
VF10(5)=DIGIT(NE)
DO 80 IJ=J4,J5
J=J+1
80 DDN(J)=STRT(IJ,K)-PHI(IJ,K)
J4=J5+1
90 WRITE (6,VF10) I,(DDN(II),II=1,J)
100 IF (IHEAD.NE.ICHK(2)) GO TO 120
C
---PRINT HEAD MATRIX---
DO 115 K=1,K0
J4=1
WRITE (6,190) K
DO 110 I=1,I0
J5=JDIML2(I)-JDIML1(I)+J4
ND=(JDIML1(I)-1)*7
ND1=ND/126 + 123
VF10(2)=DIGIT(ND1)
ND=ND-126*(ND1-123)
APPENDIX 2
LISTING OF SOURCE CODE--Continued

NE=(126-ND)/7
IF(ND1.NE.123) ND=5+ND
VF10(3)=DIGIT(ND)
VF10(5)=DIGIT(NE)
WRITE (6,VF10) I,(PHI(IJ,K),IJ=J4,J5)
110 J4=J5+1
115 CONTINUE

C

---WRITE ON DISK---
120 IF (IERR.EQ.2) GO TO 130
IF (KP.LT.NPER.OR.IFINAL.NE.1) RETURN
IF (IDK2.EQ.ICHK(5)) WRITE (4) PHI, SUM, SUMP, PUMPT, CFLUXT, QRET, CHST00006390
1, CHDT, FLUXT, STORT, ETLFLXT, FLXNT, DELT, XI, RM

C

---PUNCH OUTPUT---
130 IF (IPU2.NE.ICHK(9)) GO TO 160
IF (IERR.EQ.2) GO TO 140
WRITE (7,230) SUM, SUMP, PUMPT, CFLUXT, QRET, CHST, CHDT, FLUXT, STORT, ETLF00006420
00X, FLXNT, DELT
140 DO 135 J=1,J0
135 DELX(J)=0.0
DO 155 K=1,K0
J4=1
DO 150 1=1,10
JJ=J0-JDIML2(I)
VF4(4)=DIGIT(129)
J5=JDIML2(I)-JDIML1(I)+J4
ND=(JDIML1(I)-1)*10
ND1=ND/80 + 123
ND=ND-80*(ND1-123)
IFCND1.EQ.123) ND1=128
VF4(2)=DIGIT(ND1)
IF(ND.NE.O) GO TO 141
VF4(4)=DIGIT(128)
VF4(3)=DIGIT(128)
GO TO 142
141 VF4(3)=DIGIT(ND)
142 NE=(80-ND)/10
VF4(5)=DIGIT(NE)
WRITE (7,VF4) (PHI(IJ,K),IJ=J4,J5),(DELX(J),J=1,JJ)
150 J4=J5+1
155 CONTINUE
IF (ITL.NE.ICHK(12)) GO TO 160
DO 156 K=1,K0
WRITE (7,270) (XI(IJ,K),IJ=1,IOJ0)
156 CONTINUE
WRITE (7,280) (((RM(N,M,IJ,K),N=1,2),M=1,MODE),IJ=1,ILJL),K=1,K7)
160 IF (IERR.EQ.2) STOP
RETURN
C

---FORMATS---

39
APPENDIX 2
LISTING OF SOURCE CODE--Continued

180 FORMAT ('OMAXIMUM HEAD CHANGE FOR EACH ITERATION:/' ,39('-')/('0000067301',10F12.4))
190 FORMAT ('1',55X,'HEAD MATRIX, LAYER',I3/56X,21('-'))
200 FORMAT ('1',55X,' DRAWDOWN, LAYER',I3/59X,18('-'))
210 FORMAT (1H1,44X,57('-')/45X,14X,'TIME STEP NUMBER =',19,14X,'±000067601/45X,57('-')//50X,29HSIZE OF TIME STEP IN SECONDS=',F14.2//55X,'TO
2TAL SIMULATION TIME IN SECONDS=',F14.2/80X,8HMINUTES=',F14.2/82X,6H
3OURS=',F14.2/83X,'YEARS=',F14.2//)
230 FORMAT (1P4G20.10)
240 FORMAT ('OTIME STEP :',40I3)
250 FORMAT ('OITERATIONS:',40I3)
260 FORMAT (' ',10( '-'))
270 FORMAT(8F10.6)
280 FORMAT(6E13.6)
END


SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE

SPECIFICATIONS:
REAL *8PHI,RHO,B,D,F,H,Z,SU,RHOP,W,WMIN,RH01,RH02,RH03,XPART,YPART000067801,ZPART,DMIN1,WMAX,XT,YT,ZT,DABS,DMAX1,DEN,TXM,TYM,TZM
REAL *8E,AL,BL,CL,A,C,G,WU,TU,U,DL,RES,SUPH,GLXI,ZPHI
DIMENSION PHI(1), STRT(1), OLD(1), T(1), S(1), TR(1), TC(1), TK(1), WELL(1), DELX(1), DELY(1), DELZ(1), FACT(1), RHOP(1), TEST3(1), EL(1), FL(1), GL(1), V(1), XI(1), QRE(1), RATE(1), ZCB(1), SS(1), TL(1), TK(1), SL(1), RM(1), JDIML1(1), JDIML2(1)

COMMON /INTEGR/ IO,J0,K0,I1,J1,K1,NPER,KTH,ITMAX,LENGTH,KP,NWEL,1NUMT,IFINAL,IT,KT,IHEAD,IDRAW,IFLO,IERR,I2,J2,K2,IMAX,ITMX1,NCH,2DK1,IK1,IV,IPJ,IPJQ,IKJ,IKJQ,K5,IPU1,IPU2,ITK,IEN,MODE
.ITL,IOJ0,K6,K7,ILJL
COMMON /SARRAY/ ICHK(13)
RETURN
ENTRY ITER

WRITE (6,240)
P2=LENGTH-1
NT=IOJ0*K0
NIJ=IOJ0
APPENDIX 2
LISTING OF SOURCE CODE--Continued

READ 888,WMAX
888 FORMAT(F10.0)
PRINT 889,WMAX
889 FORMAT(5X,6HWMAX =,F10.7)
PJ=-1.
DO 50 I=1,LENGTH
PJ=PJ+1.
50 RHOP(I)=1.DO-(1.DO-WMAX)**(PJ/P2)
WRITE (6,230) LENGTH ,( RHOP(J ), J=1 , LENGTH )
RETURN

C ...........................................................................
C
C  ***INITIALIZE DATA FOR A NEW ITERATION***
C 60 IT=IT+1
C IF (IT.LE.ITMAX) GO TO 70
C WRITE (6,220)
C CALL OUTPUT
C 70 IF (MOD(IT,LENGTH)) 80,80,90
C ENTRY NEWITA
C 80 NTH=0
C 90 NTH=NTH+1
C W=RHOP(NTH)
C TEST3(IT+1)=0.
C TEST=0.0
C BIG=0.
C DO 100 I=1,NT
C EL(I)=0.
C FL(I)=0.
C GL(I)=0.
C 100 V(I)=0.
C
C
C  ---COMPUTE TRANSMISSIVITY AND T COEFFICIENTS FOR UPPER
C HYDROLOGIC UNIT WHEN IT IS UNCONFINED---
C IF (IWATER.NE.ICHK(6)) GO TO 110
C CALL TRANS(0)
C
C  ---CHOOSE SIP NORMAL OR REVERSE ALGORITHM---
C 110 IF (MOD(IT,2)) 120,120,170
C 120 J9=1
C DO 155 K=1,K0
C J4=JDIML2(1)-JDIML1(1)+2
C KN=(K-1)*NIJ
C DO 152 I=2,I1
C J5=JDIML2(I)-JDIML1(I)+J4
C J6=J4+KN
C J7=J5+KN
C J8=J7+JDIML2(I+1)-JDIML1(I+1)+1
C DO 150 IJ=J4,J5
C N=IJ+KN
C
APPENDIX 2
LISTING OF SOURCE CODE—Continued

---SKIP COMPUTATIONS IF NODE CONSTANT HEAD---

IF(S(N).LT.0.) GO TO 150
NIB=N-JDIML2(I-1)+JDIML1(I)-1
NIA=N+JDIML2(I)-JDIML1(I+1)+1
NJA=N+1
NJB=N-1
IF(N.EQ.J6) NJB=1
IF(N.EQ.J7) NJA=1
IF(NIB.GE.J6) NIB=1
IF(NIB.LT.J9) NIB=1
IF(NIA.GT.J8) NIA=1
IF(NIA.LE.J7) NIA=1
IF(I.EQ.2) NIB=1
IF(I.EQ.H) NIA=1
NKA=N+NIJ
NKB=N-NIJ

IF (ITL.NE.ICHK(12)) GO TO 121
L=N
LKB=NKB
GO TO 122
121 L=1
LKB=1
TLK(LKB)=0.
TLK(L)=0.
TL(L)=0.
SL(L)=0.

---SKIP COMPUTATIONS IF NODE OUTSIDE FLOW SYSTEM---

122 IF(K.EQ.1) GO TO 111
IF(K.EQ.KO) GO TO 113
1 .AND.TLK(LKB).EQ.O. ) GO TO 150
GO TO 112
111 IF(T(N).EQ.O..AND.TK(N).EQ.O..AND.TL(L).EQ.O. ) GO TO 150
GO TO 112
113 IF(T(N).EQ.O..AND.TK(NKB).EQ.O..AND.TLK(LKB).EQ.O. ) GO TO 150
---COMPUTE COEFFICIENTS---

112 J=N-J4-KN+JDIML1(I)
D=TR(NJB)/DELX(J)
F=TR(N)/DELX(J)
B=TC(NIB)/DELY(I)
H=TC(N)/DELY(I)
SU=0.DO
Z=0.DO
IF(K.EQ.1) GO TO 124
APPENDIX 2
LISTING OF SOURCE CODE—Continued

Z=TK(NKB) + TLK(LKB)
IF(IEQN.EQ.ICHK(11)) Z=Z/DELTZ(K)
124 IF(K.EQ.KO) GO TO 125
SU=TK(N) + TLK(L)
IF(IEQN.EQ.ICHK(11)) SU=SU/DELTZ(K)
125 RHO=S(N)/DELT
QR=0.
IF(K.EQ.KO.AND.IQRE.EQ.ICHK(7)) QR=QRE(IJ)

---SIP NORMAL ALGORITHM---
---FORWARD SUBSTITUTE, COMPUTING INTERMEDIATE VECTOR V---
IF(K.EQ.1) GO TO 131
IF(K.EQ.KO) GO TO 133
GO TO 130
131 E=-B-D-F-H-SU-Z-RHO-TL(L)+TLK(L)
GO TO 132
133 E=-B-D-F-H-SU-Z+TLK(LKB)-TL(L)
GO TO 132
130 E=-B-D-F-H-SU-Z-RHO-HTLK(LKB)-HTLK(L)-TL(L)
GO TO 132
CONTINUE
BL=B/(1.+W*(EL(NIB)+GL(NIB)))
CL=D/(1.+W*(FL(NJB)+GL(NJB)))
C=BL*EL(NIB)
G=CL*FL(NJB)
WU=CL*GL(NJB)
U=BL*GL(NIB)
IF(K.EQ.1) GO TO 140
AL=Z/(1.+W*(EL(NKB)+FL(NKB)))
A=AL*EL(NKB)
TU=AL*FL(NKB)
DL=E+H*(C+G+WU+TU+U)-CL*EL(NJB)-BL*FL(NIB)-AL*GL(NKB)
EL(N)=(F-W*(A+C))/DL
FL(N)=(H-W*(G+TU))/DL
GL(N)=(SU-W*(WU+U))/DL
SUPH=0.DO
IF(K.NE.KO) SUPH=SUPH*PHI(NKA)
RES=B*PHI(NIB)-D*PHI(NJB)-E*PHI(N)-F*PHI(NJA)-H*PHI(NIA)-SUPH-Z*PHI(NKB)
HI(NKB)=WELL(N)-RHO*OLD(N)-QR-SL(L)
V(N)=(RES-AL*V(NKB)-BL*V(NIB)-CL*V(NJB))/DL
GO TO 150
140 DL=E+H*(C+G+SU+WU+U)-CL*EL(NJB)-BL*FL(NIB)
EL(N)=(F-W*C)/DL
FL(N)=(H-W*G)/DL
GL(N)=(SU-W*(WU+U))/DL
SUPH=0.DO
IF(K.NE.KO) SUPH=SUPH*PHI(NKA)
RES=B*PHI(NIB)-D*PHI(NJB)-E*PHI(N)-F*PHI(NJA)-H*PHI(NIA)-SUPH+WELL(N)
1L(N)-RHO*OLD(N)-QR-SL(L)
V(N)=(RES-BL*V(NIB)-CL*V(NJB))/DL
CONTINUE
J9=J6
APPENDIX 2
LISTING OF SOURCE CODE--Continued

152 J4=J5+1
155 CONTINUE

C
---BACK SUBSTITUTE FOR VECTOR XI---
DO 165 K=1,K0
   K3=K0-K+1
      J4=JDIML2(I0)-JDIML1(I0)+2
      KN=(K3-1)*NIJ
   DO 162 I=1,I2
      I3=I0-I
         J5=JDIML2(I3)-JDIML1(I3)+J4
         J6=NIJ-J4+1+KN
         J7=J6+JDIML2(I3+1)-JDIML1(I3+1)+1
   DO 160 IJ=J4,J5
      N = NIJ-IJ+UKN
      IF(S(N).LT.O.) GO TO 160
      IF(ITL.NE.ICHK(12)) GO TO 157
      L=N
      LKB=N-NIJ
      GO TO 158
   157 L=1
      LKB=1
   158 IF(K3.EQ.K0) GO TO 151
      IF(K3.EQ.1) GO TO 159
         1 O..AND.TLK(LKB).EQ.O.) GO TO 160
      GO TO 156
   151 IF(T(N).EQ.O..AND.TK(N-NIJ).EQ.O..AND.TLK(LKB).EQ.O.) GO TO 160
      GO TO 156
   159 IF(T(N).EQ.O..AND.TK(N).EQ.O..AND.TLK(L).EQ.O.) GO TO 160
   156 GLXI=O.DO
      NJA=N+1
      NIA=N+JDIML2(I3)-JDIML1(I3+1)+1
      IF(N.EQ.J6) NJA=1
      IF(NIA.GT.J7)NIA=1
      IF(NIA.LE.J6) NIA=1
      IF(I.EQ.1) NIA=1
      IF(K3.NE.K0) GLXI=GL(N)*V (N+NIJ)
      V (N)=V (N)-EL(N)*V (NJA)-FL(N)*V (NIA)-GLXI
   160 CONTINUE
   162 J4=J5+1
   165 CONTINUE
      IF(BIG.GT.ERR) TEST=1.
      TEST3(IT+1)=BIG
      IF (TEST.EQ.O.) RETURN

C
---COMPARE MAGNITUDE OF CHANGE WITH CLOSURE CRITERIA---
TCHK=ABS( V(N))
   IF(TCHK.GT.BIG) BIG=TCHK
   PHI(N)=PHI(N)+ V(N)
   XI(N)=XI(N)+ V(N)
160 CONTINUE
GO TO 60

170 J8=1
DO 205 KK=1,K0
    K=K0-KK+1
    J4=JDIML2(I0)-JDIML1(I0)+2
    KN=(K-1)*NIJ
DO 202 II=1,I2
    I=IO-II
    J5=JDIML2(I)-JDIML1(I)+J4
    J6=NIJ-J5+1+KN
    J7=NIJ-J4+1+KN
    J9=J6-JDIML2(I-1)+JDIML1(I-1)-1
DO 200 IJ=J4,J5
    N=NIJ-J5-JM+IJ+1+KN
    IF (S(N).LT.O.) GO TO 200
    NIB=N-JDIML2(I-1)+JDIML1(I)-1
    NIA=N+JDIML2(I)-JDIML1(I+1)+1
    NJA=N+1
    NJB=N-1
    IF (N.EQ.J6) NJB=1
    IF (N.EQ.J7) NJA=1
    IF (NIB.GE.J6) NIB=1
    IF (NIB.LT.J9) NIB=1
    IF (NIA.GT.J8) NIA=1
    IF (NIA.LE.J7) NIA=1
    IF (I.EQ.2) NIB=1
    IF (I.EQ.I1) NIA=1
    NKA=N+NIJ
    NKB=N-NIJ
    IF (ITL.NE.ICHK(12)) GO TO 171
    L=N
    LKB=NKB
GO TO 172
171 L=1
    TLK(LKB)=0.
    TLK(L)=0.
    TL(L)=0.
    SL(L)=0.
---SKIP COMPUTATIONS IF NODE OUTSIDE FLOW SYSTEM---
172 IF(K.EQ.1) GO TO 164
    IF(K.EQ.K0) GO TO 163
    IF (T(N).EQ.0..AND.TK(N).EQ.0..AND.TK(NKB).EQ.0..AND.TLK(L).EQ.0.
1 .AND.TLK(LKB).EQ.0.) GO TO 200
    GO TO 161
164 IF(T(N).EQ.0..AND.TK(N).EQ.0..AND.TLK(L).EQ.0.) GO TO 200
    GO TO 161
APPENDIX 2
LISTING OF SOURCE CODE--Continued

163 IF(T(N).EQ.O..AND.TK(NKB).EQ.O..AND.TLK(LKB).EQ.O.) GO TO 200

---COMPUTE COEFFICIENTS---
161 J=N-J6+JDIM1(I) 0000916
D=TR(NJB)/DELX(J) 0000917
F=TR(N)/DELX(J) 0000918
B=TC(NIB)/DELY(I) 0000919
H=TC(N)/DELY(I) 0000920
SU=O.DO 0000921
Z=O.DO 0000922

IF(K.EQ.1) GO TO 174 0000923
Z=TK(NKB) + TLK(LKB) 0000924
IF(IEQN.EQ.ICHK(11)) Z=Z/DELZ(K) 0000925
174 IF(K.EQ.KO) GO TO 175 0000926
SU=TK(N) + TLK(L) 0000927
IF(IEQN.EQ.ICHK(11)) SU=SU/DELZ(K) 0000928
175 RHO=S(N)/DELT 0000929
QR=O. 0000930
IF(K.EQ.KO.AND.IQRE.EQ.ICHK(7)) QR=QRE(N-KN) 0000931

---SIP REVERSE ALGORITHM---
---FORWARD SUBSTITUTE, COMPUTING INTERMEDIATE VECTOR V---
IF (K.EQ.1) GO TO 181 0000932
IF (K.EQ.KO) GO TO 183 0000933
GO TO 180 0000934
181 E=-B-D-F-H-SU-Z-RHO-TL(L)+TLK(L) 0000935
GO TO 182 0000936
183 E=-B-D-F-H-SU-Z-RHO+TLK(LKB)-TL(L) 0000937
GO TO 182 0000938
180 E=-B-D-F-H-SU-Z-RHO+TLK(LKB)+TLK(L)-TL(L) 0000939
182 CONTINUE 0000940
BL=H/(1.+W*(EL(NIA)+GL(NIA))) 0000941
CL=D/(1.+W*(FL(NJB)+GL(NJB))) 0000942
C=BL*EL(NIA) 0000943
G=CL*FL(NJB) 0000944
WU=CL*GL(NJB) 0000945
U=BL*GL(NIA) 0000946
IF (K.EQ.KO) GO TO 190 0000947
AL=SU/(1.+W*(EL(NKA)+FL(NKA))) 0000948
A=AL*EL(NKA) 0000949
TU=AL*FL(NKA) 0000950
DL=E+W*(C+G+A+WU+TU+U)-AL*GL(NKA)-BL*FL(NIA)-CL*EL(NJB) 0000951
EL(N)=(F-W*(C+A))/DL 0000952
FL(N)=(B-W*(G+TU))/DL 0000953
GL(N)=(Z-W*(WU+U))/DL 0000954
ZPHI=O.DO 0000955
IF (K.NE.1) ZPHI=Z*PHI(NKB) 0000956
RES=-B*PHI(NIB)-D*PHI(NJB)-E*PHI(N)-F*PHI(NJA)-H*PHI(NIA)-SU*PHI(N)0000957
1KA)-ZPHI-WELL(N)-RHO*OLD(N)-QR-SL(L) 0000958
V(N)=(RES-AL*V(NKA)-BL*V(NIA)-CL*V(NJB))/DL 0000959
GO TO 200 0000960
190 DL=E+W*(C+G+WU+U)-BL*FL(NIA)-CL*EL(NJB) 0000961

46
APPENDIX 2
LISTING OF SOURCE CODE--Continued

EL(N) = (F - W*C) / DL  
FL(N) = (B - W*G) / DL  
GL(N) = (Z - W*(W*U + U)) / DL  
ZPHI = 0.0D0  
IF (K.NE.1) ZPHI = Z*PHI(NKB)  
RES = -B*PHI(NIB) - D*PHI(NJB) - E*PHI(N) - F*PHI(NJA) - H*PHI(NIA) - ZPHI  
1L(N) = -RHO*OLD(N) - QR*SL(L)  
V(N) = (RES - BL*V(NIA) - CL*V(NJB)) / DL  
200 CONTINUE  
J8 = J7  
202 J4 = J5 + 1  
205 CONTINUE

C --- BACK SUBSTITUTE FOR VECTOR XI ---
J8 = 1  
DO 215 K = 1, K0  
J4 = JDIML2(1) - JDIML1(1) + 2  
KN = (K - 1) * NIJ  
DO 212 I = 2, I1  
J5 = JDIML2(I) - JDIML1(I) + J4  
J6 = J5 + KN  
J7 = J4 + KN  
DO 210 IJ = J4, J5  
N = J5 - IJ + J4 + KN  
IF (S(N).LT.0.) GO TO 210  
IF (ITL.NE.ICHK(12)) GO TO 204  
L = N  
LKB = N - NIJ  
GO TO 206  
204 L = 1  
LKB = 1  
206 IF (K.EQ.1) GO TO 201  
IF (K.EQ.K0) GO TO 207  
IF (T(N).EQ.0..AND.TK(N).EQ.0..AND.TK(N-NIJ).EQ.0..AND.TLK(L).EQ.0.) GO TO 210  
1 .AND.TLK(LKB).EQ.0.) GO TO 210  
GO TO 203  
201 IF (T(N).EQ.0..AND.TK(N).EQ.0..AND.TLK(L).EQ.0.) GO TO 210  
GO TO 203  
207 IF (T(N).EQ.0..AND.TK(N-NIJ).EQ.0..AND.TLK(LKB).EQ.0.) GO TO 210  
203 GLXI = 0.0D0  
NJA = N + 1  
NIB = N - JDIML2(I-1) + JDIML1(I-1)  
IF (NIB.GE.J7) NIB = 1  
IF (NIB.LT.J8) NIB = 1  
IF (I.EQ.2) NIB = 1  
IF (N.EQ.J6) NJA = 1  
IF (K.NE.1) GLXI = GL(N)*V(N-NIJ)  
V(N) = V(N) - EL(N)*V(NJA) - FL(N)*V(NIB) - GLXI

C --- COMPARE MAGNITUDE OF CHANGE WITH CLOSURE CRITERIA ---
TCHK = ABS(V(N))  

47
APPENDIX 2
LISTING OF SOURCE CODE--Continued

IF (TCHK.GT.BIG) BIG=TCHK
PHI(N)=PHI(N)+ V(N)
XI(N)= XI(N)+ V(N)

CONTINUE
J8=J7
J4=J5+1

CONTINUE
IF (BIG.GT.ERR) TEST=1.
TEST3CIT+1 )=BIG
IF (TEST.EQ.O.) RETURN
GO TO 60

-----------------------------------------------------------------------------
---FORMATS---

220 FORMAT ('OEXCEEDED PERMITTED NUMBER OF ITERATIONS'/' ',39('*'))
230 FORMAT ('///1HO,I5,22H ITERATION PARAMETERS:1P6E15.7(/28X,1P6E15.7/))
240 FORMAT ('-44X,'SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE'/'45X,00010190

END

SUBROUTINE CHECKI (PHI, STRT, OLD, T, S, TR, TC, TK, WELL, DELX, DELY, DELZ, FACT, JFLO, FLOW, QRE, XI, RATE, ZCB, SS, TL, TLK, SL, RM, JDIML1, JDIML2)

---------------------------------------------------------------
COMPUTE A VOLUMETRIC BALANCE
---------------------------------------------------------------

REAL *8PHI

DIMENSION PHI(IOJO,KO), STRT(IOJO,KO), OLDUOJO, KO), TdOJO, KO 1), S(IOJO,KO), TR(IOJO,KO), TC(IOJO,KO), TK(1KJK,K5), WELL(IO 2JO,KO), DELX(JO), DELY(IO), DELZ(KO), FACT(KO,3), JFLO(NCH,3), FLO 3W(NCH), QRE(IQJQ)
4,XI(IOJO,KO),RATE(1),ZCB(1),SS(1),TL(1),TLK(1),SL(1),RM(1)
5,JDIML1(IO),JDIML2(IO)

COMMON /INTEGR/ I0,K0,IT,7,K1,NPER,KTH,ITMAX,LENGTH,KP,NWEL, 1NUMT,IFINAL,IT,7,K1,HEAD, redraw, IFLO, IERR, I2, J2,K2,IMAX,ITMX1,NCH, 2DK1,DK2,IWATER,ICRE,IPJP, IQJQ, IKJK,K5, IPU1, IPU2, ITK, IEQN, MODE .,ITL,IOJO,K6,K7,ILJL
COMMON /SPARAM/ TMAX, CDLT, DELT, ERR, TEST, SUM, SUMP, QR, DELTM1
COMMON /SARRAY/ ICHK(13)
COMMON /CK/ ETFLX,TXT, STORT, QRET, CHST, CHDT, FLXNT, PUMPT, CFLXNT, FLXNT
RETURN

ENTRY CHECK
---------------------------------------------------------------


---INITIALIZE VARIABLES---
NIJ = IOJO
PUMP=0.
STOR=0.
FLUXS=0.0
CHD1=0.0
CHD2=0.0
QREFLX=0.
CFLUX=0.
FLUX=0.
ETFLUX=0.
FLXN=0.0
II=0
ACHD1=0.
ACHD2=0.

---COMPUTE RATES, STORAGE AND PUMPAGE FOR THIS STEP---
DO 225 K=1,K0
   J7=1
   NK = IOJO*(K-1)
   J4=JDIML2(1)-JDIML1(1) + 2
   DO 222 I=2,11
      J5 = JDIML2(I)-JDIML1(I)+J4
      J6=J5+JDIML2(I+1)-JDIML1(I+1) + 1
   DO 220 IJ=J4,J5
      NI = IJ + NK
      NIJB=NI-NIJ
      IF(ITL.EQ.ICHK(1)) GO TO 4
      NI=1
      NIJB=1
      IFCK.NE.1) GO TO 5
   IF(T(IJ,K).EQ.O. .AND.TK(IJ,K).EQ.O. .AND. TLK(NI ).EQ. 0. ) GO TO 7
   IF(T(IJ-1,K).LT.O. .OR.T(IJ-1,K).EQ.O.) GO TO 30
      X=(PHI(IJ,K)-PHI(IJ-1,K))*TR(IJ-1,K)*DELY(I)
      IF(IEQN.EQ.ICHK(1)) X=X*DELZ(K)
   7 J=IJ-J4+JDIML1(I)
      AREA=DELX(J)*DELY(I)
      VOLUME=AREA*DELZ(K)
      IF (S(IJ,K).GE.O.) GO TO 180
   220 CONTINUE

---COMPUTE FLOW RATES TO AND FROM CONSTANT HEAD BOUNDARIES---
II=II+1
FLOW(II)=0.
JFLO(II,1)=K
JFLO(II,2)=I
JFLO(II,3)=J
IF(IJ.EQ.J4) GO TO 30
IF (S(IJ-1,K).LT.O. .OR.T(IJ-1,K).EQ.O.) GO TO 30
X=(PHI(IJ,K)-PHI(IJ-1,K))*TR(IJ-1,K)*DELY(I)
IF(IEQN.EQ.ICHK(11)) X=X*DELZ(K)
APPENDIX 2
LISTING OF SOURCE CODE—Continued

FLOW(II)=FLOW(II)+X
IF (X) 10, 30, 20
10 CHD1=CHD1+X
GO TO 30
20 CHD2=CHD2+X
30 IF (I,J.EQ.J5) GO TO 60
   IF (S(IJ+1,K).LT.0..OR.T(IJ+1,K).EQ.0.) GO TO 60
   X=(PHI(I,J,K)-PHI(I,J+1,K))*DELY(I)*TR(IJ,K)
   IF (IEQN.EQ.ICHK(11)) X=X*DELZ(K)
   FLOW(II)=FLOW(II)+X
   IF (X) 40, 60, 50
40 CHD1=CHDUX
GO TO 60
50 CHD2=CHD2+X
60 IF (K.EQ.1) GO TO 90
   IF (S(IJ,K-1).LT.0..OR.TK(IJ,K-1).EQ.0.) GO TO 90
   X=(PHI(I,J,K)-PHI(I,J,K-1))*TK(IJ,K-1)*AREA
   FLOW(II)=FLOW(II)+X
   IF (X) 70, 90, 80
70 CHD1=CHDUX
GO TO 90
80 CHD2=CHD2+X
90 IF (K.EQ.KO) GO TO 120
   IF (S(IJ,K+1).LT.0..OR.TK(IJ,K).EQ.0.) GO TO 120
   X=(PHI(I,J,K)-PHI(I,J,K+1))*TK(IJ,K)*AREA
   FLOW(II)=FLOW(II)+X
   IF (X) 100, 120, 110
100 CHD1=CHDUX
GO TO 120
110 CHD2=CHD2+X
120 NIB=IJ-JDIML2(I-1)+JDIML1(I)-1
   IF (NIB.GE.J4) NIB=1
   IF (NIB.LT.J7) NIB=1
   IF (I.EQ.2) NIB=1
   IF (S(NIB,K).LT.0..OR.T(NIB,K).EQ.0.) GO TO 150
   X=(PHI(I,J,K)-PHI(NIB,K))*TC(NIB,K)*DELX(J)
   IF (IEQN.EQ.ICHK(11)) X=X*DELZ(K)
   FLOW(II)=FLOW(II)+X
   IF (X) 130, 150, 140
130 CHD1=CHDUX
GO TO 150
140 CHD2=CHD2+X
150 NIA=IJ+JDIML2(I)-JDIML1(I+1)+1
   IF (NIA.GT.J6) NIA=1
   IF (NIA.LE.J5) NIA=1
   IF (I.EQ.I1) NIA=1
   IF (S(NIA,K).LT.0..OR.T(NIA,K).EQ.0.) GO TO 171
   X=(PHI(I,J,K)-PHI(NIA,K))*TC(IJ,K)*DELX(J)
   IF (IEQN.EQ.ICHK(11)) X=X*DELZ(K)
   FLOW(II)=FLOW(II)+X
   IF (X) 160, 171, 170
50
APPENDIX 2
LISTING OF SOURCE CODE—Continued

160 CHD1=CHD1+X
GO TO 171
170 CHD2=CHD2+X
171 IF(FLOW(I)) 172,220,174
172 ACHD1=ACHD1+AREA
GO TO 220
174 ACHD2=ACHD2+AREA
GO TO 220

---CHECK FOR EQUATION BEING SOLVED---
180 IF(IEQN.EQ.ICHK(11)) GO TO 211

---EQUATION 4---
---RECHARGE AND WELLS---
IF (K.EQ.KO.AND.IQRE.EQ.ICHK(7)) QREFLX=QREFLX+QRE(IJ)*AREA
IF (WELL(IJ,K)) 190,210,200
190 PUMP=PUMP+WELL(IJ,K)*AREA
GO TO 210
200 CFLUX=CFLUX+WELL(IJ,K)*AREA

---COMPUTE VOLUME FROM STORAGE---
210 STOR=STOR-S(IJ,K)*XI(IJ,K)*AREA
GO TO 215

---EQUATION 3---
---RECHARGE AND WELLS---
211 IF (K.EQ.KO.AND.IQRE.EQ.ICHK(7)) QREFLX=QREFLX+QRE(IJ)*VOLUME
IF (WELL(IJ,K)) 212,214,213
212 PUMP=PUMP+WELL(IJ,K)*VOLUME
GO TO 214
213 CFLUX=CFLUX+WELL(IJ,K)*VOLUME

---COMPUTE VOLUME FROM STORAGE---
214 STOR=STOR-S(IJ,K)*XI(IJ,K)*VOLUME
215 IF (ITL.NE.ICHK(12)) GO TO 220
Z = 0.0
SU = 0.0
IF (K.NE.1) Z = TLK(NI-NIJ)*PHI(IJ,K-1)
IF (K.NE.KO) SU = TLK(NI)*PHI(IJ,K+1)
QLEAKN=-(TL(NI)*PHI(IJ,K)-SU-Z-SL(NI))*AREA
FLUXS = FLUXS + QLEAKN
IF (QLEAKN.LT.0) FLXN=FLXN-QLEAKN
CONTINUE
J7=J4
220 CONTINUE
J4=J5+1
225 CONTINUE

---COMPUTE CUMULATIVE VOLUMES, TOTALS, AND DIFFERENCES---
FLUXT = FLUXT + FLUXS*DELT
FLXNT=FLXN*DELT+FLXNT

51
APPENDIX 2
LISTING OF SOURCE CODE--Continue<

FLXPT = FLUXT + FLXNT
STORT=STORT+STOR
STOR=STOR/DELT
QRET=QRET+QREFLX*DELT
CHDT=CHDT-CHD1*DELT
CHST=CHST+CHD2*DELT
PUMPT=PUMPT-PUMP*DELT
CFLUXT=CFLUXT+CFLUX*DELT
TOTL1=STORT+QRET+CFLUXT+CHST+FLXPT
TOTL2=CHDT+PUMPT+ETFLXT+FLXNT
SUMR=QREFLX+CFLUX+CHD2+CHDUPUMP+ETFLUX+FLUXS+STOR
DIFF=TOTL2-TOTL1
PERCNT=0.0
IF (TOTL2.EQ.0. ) GO TO 230
PERCNT=DIFF/TOTL2*100.
230 RETURN

---PRINT RESULTS---
ENTRY CWRITE
WRITE (6,260) STOR, QREFLX, STORT, CFLUX, QRET, PUMP, CFLUXT, ETFLUX, CHST, CFLUXT, CHST, TOTL1, CHST, TOTL2, CHD2, TOTL1, CHD1, FLUX, FLUXS, ETFLUX, CHDT, SUMR, PUMPT, FLXNT, TOTL2, DIFF, PERCNT

---COMPUTE VERTICAL FLOW---
240 X=0.
Y = 0.
IF (KO.EQ.1) RETURN
J4=JDIML2(1)-JDIML1(1) + 2
DO 252 I=2,11
J5 = JDIML2(I)-JDIML1(I)+J4
DO 250 IJ=J4,J5
J=IJ-J4+JDIML1(I)
X=X+(PHI(IJ, 1)-PHI(IJ,2))*TK(IJ, 1)*DELX(J)*DELY(I)
Y=Y+(PHI(IJ,K1)-PHI(IJ,K0))*TK(IJ,K1)*DELX(J)*DELY(I)
250 CONTINUE
252 J4=J5+1
WRITE (6,290) Y,X
WRITE (6,300) ACHD1,ACHD2
RETURN

---FORMATS---

260 FORMAT ('0',10X,'CUMULATIVE MASS BALANCE: ',16X,'L**3',23X,'RATES F001194
APPENDIX 2
LISTING OF SOURCE CODE--Continued

10R THIS TIME STEP: ',16X,'L**3/T'/11X,24('-'),43X,25('-')/20X,'SOU
2RCES: ',69X,'STORAGE = ',F20.4/20X,8('-'),68X,'RECHARGE = ',F20.4/27X
3,'STORAGE = ',F20.2,35X,'CONSTANT FLUX = ',F20.4/26X,'RECHARGE = ',F2
40.2,41X,'PUMPING = ',F20.4/21X,'CONSTANT FLUX = ',F20.2,30X,'EVAPOTR
5ANSPIRATION = ',F20.4/21X,'CONSTANT HEAD = ',F20.2,34X,'CONSTANT HEA
6D: ',27X,'LEAKAGE = ',F20.4/20X,11('-'),68X,'TOTAL SOURCES = ',F
720.2,45X,'OUT = ',F20.4/96X,'LEAKAGE: ',F20.4/20X,'DISCHARGES: ',45X,'FROM
8PREVIOUS PUMPING PERIOD = ',F20.4/20X,11('-'),68X,'TOTAL = ',F20.4/1
96X,'EVAPOTRANSPIRATION = ',F20.2/21X,'CONSTANT HEAD = ',F20.2,36X,'S
APTURED RATE = ',F20.2/27X,'TOTAL DISCHARGE = ',F20.2/17X,'DISCHARGE-SOURCES = ',F20
n.2/15X,'PER CENT DIFFERENCE = ',F20.2//)
100 FORMAT ('OFLow TO TOP LAYER = ',G15.7, ' FLOW TO BOTTOM LAYER = ',G00011950
115.7, ' POSITIVE UPWARD')
200 FORMAT ('O'5X,'AREA OF CONSTANT HEAD DISCHARGE = ',G15.7,'L**2', 00011960
13X,'AREA OF CONSTANT HEAD SOURCE = ',G15.7,'L**2')
310 FORMAT('O'5X,'BOUNDARY FLUX IN =', G15.7,'L**3/S',3X,'BOUNDARY FLUXOUT =',G15.7,'L**3/S'/5X,'NOTE: THESE FLUXES ARE INCLUDED IN THE
2 BUDGET AS CONSTANT FLUX AND PUMPING')
END
SUBROUTINE COEF(PHI, STRT, OLD, T, S, TR, TC, TK, WELL, DELX, DELY, DELZ, FACT
1, PERM, BOTTOM, QRE, XI, RATE, ZCB, SS, TL, TLK, SL, RM, JDIML1, JDIML2)
---------------------------------
COMPUTE COEFFICIENTS
---------------------------------

SPECIFICATIONS:
REAL *8PHI
DIMENSION PHI(IOJO,KO), STRT(IOJO,KO), OLD(IOJO,KO), T(IOJO,KO)
1, S(IOJO,KO), TR(IOJO,KO), TC(IOJO,KO), TK(IKJK,K5), WELL(IO
2JO,KO), DELX(JO), DELY(IO), DELZ(KO), FACT(KO,3), PERM(IPJP), BOT
3TOM(IPJP), QRE(IQJQ)
., XI(IOJO,KO), RATE(ILJL,K7), ZCB(ILJL,K7), TL(ILJL,K6), SS(K7),
.TLK(ILJL,K6), SL(ILJL,K6), RM(2,MODE, ILJL,K7), JDIML1(IO), JDIML2(IO)
DIMENSION A(10,10), B(10), B1(10), BO(10)

COMMON /INTEGR/ 10, JO, KO, I1, J1, K1, NPER, KTH, ITMAX, LENGTH, KP, NWEL,
1NUMT, IFINAL, IT, KTH, IHED, IDRAW, IFLO, IERR, I2, J2, K2, IMAX, ITMX1, NCH,
2IDK1, IDK2, IWATER, IQRE, IPJP, IQJQ, IKJK, K5, IPU1, IPU2, ITK, IEQN, MODE
., ITL, IOJO, K6, K7, ILJL
COMMON /SPARAM/ TMAX, CDLT, DELT, ERR, TEST, SUM, SUMP, QR, DELTM1
COMMON /SARRAY/ ICHK(13)
RETURN

---COMPUTE TRANSMISSIVITY FOR UPPER HYDROLOGIC UNIT WHEN
IT IS UNCONFINED---

ENTRY TRANS(N3)

J6=1

53
APPENDIX 2
LISTING OF SOURCE CODE--Continued

J4 = JDIML2(1) - JDIML1(1) + 2
DO 12 I = 2, I1
   J5 = JDIML2(I) - JDIML1(I) + J4
DO 10 IJ = J4, J5
   IF (PERM(IJ).EQ.0.) GO TO 10
   T(IJ, KO) = PERM(IJ) * (PHI(IJ, KO) - BOTTOM(IJ))
   IF (T(IJ, KO).GT.0.) GO TO 10
   J = IJ - J4 + JDIML1(I)
   IF (WELL(IJ, KO).LT.0.) WRITE (6,600) I, J, KO
   IF (WELL(IJ, KO).GE.0.) WRITE (6,610) I, J, KO
   PERM(IJ) = 0.
   T(IJ, KO) = 0.
   NJB = IJ - 1
   IF (IJ.EQ.J4) NJB = 1
   TR(NJB, KO) = 0.
   TR(IJ, KO) = 0.
   TC(IJ, KO) = 0.
   NIB = IJ - JDIML2(I - 1) + JDIML1(I - 1)
   IF (I.EQ.2) NIB = 1
   IF (NIB.GE.J4) NIB = 1
   IF (NIB.LT.J6) NIB = 1
   TC(NIB, KO) = 0.
   IF (KO.NE.1) TK(IJ, K1) = 0.
   PHI(IJ, KO) = 1.D30
10 CONTINUE
   J6 = J4
12 J4 = J5 + 1
   IF (N3.EQ.1) RETURN
   N1 = KO
   N2 = KO
   N4 = K1
   GO TO 20

C .................................................................
C ---COMPUTE T COEFFICIENTS---
C .................................................................
C **********************
ENTRY TCOF
C **********************
N1 = 1
N2 = KO
N4 = 1
20 DO 45 K = N1, N2
   J4 = 1
   DO 42 I = 1, I1
      J5 = JDIML2(I) - JDIML1(I) + J4
      J6 = J5 + JDIML2(I + 1) - JDIML1(I + 1) + 1
   DO 39 IJ = J4, J5
      IF (T(IJ, K).EQ.0.) GO TO 39
      NJA = IJ + 1
      IF (NJA.EQ.J5 + 1) NJA = 1
      J = IJ - J4 + JDIML1(I)
   40 CONTINUE
   IFUJ.EQ.J4) NJA = 1
   TR(NJB, KO) = 0.
   TR(IJ, KO) = 0.
   TC(IJ, KO) = 0.
   NIB = IJ - JDIML2(I - 1) + JDIML1(I - 1)
   IF (I.EQ.2) NIB = 1
   IF (NIB.GE.J4) NIB = 1
   IF (NIB.LT.J6) NIB = 1
   TC(NIB, KO) = 0.
   IF (KO.NE.1) TK(IJ, K1) = 0.
   PHI(IJ, KO) = 1.D30
10 CONTINUE
   J6 = J4
12 J4 = J5 + 1
   IF (N3.EQ.1) RETURN
   N1 = KO
   N2 = KO
   N4 = K1
   GO TO 20

C .................................................................
C ---COMPUTE T COEFFICIENTS---
C .................................................................
C **********************
ENTRY TCOF
C **********************
N1 = 1
N2 = KO
N4 = 1
20 DO 45 K = N1, N2
   J4 = 1
   DO 42 I = 1, I1
      J5 = JDIML2(I) - JDIML1(I) + J4
      J6 = J5 + JDIML2(I + 1) - JDIML1(I + 1) + 1
   DO 39 IJ = J4, J5
      IF (T(IJ, K).EQ.0.) GO TO 39
      NJA = IJ + 1
      IF (NJA.EQ.J5 + 1) NJA = 1
      J = IJ - J4 + JDIML1(I)
   40 CONTINUE
   IFUJ.EQ.J4) NJA = 1
   TR(NJB, KO) = 0.
   TR(IJ, KO) = 0.
   TC(IJ, KO) = 0.
   NIB = IJ - JDIML2(I - 1) + JDIML1(I - 1)
   IF (I.EQ.2) NIB = 1
   IF (NIB.GE.J4) NIB = 1
   IF (NIB.LT.J6) NIB = 1
   TC(NIB, KO) = 0.
   IF (KO.NE.1) TK(IJ, K1) = 0.
   PHI(IJ, KO) = 1.D30
10 CONTINUE
   J6 = J4
12 J4 = J5 + 1
   IF (N3.EQ.1) RETURN
   N1 = KO
   N2 = KO
   N4 = K1
   GO TO 20

C .................................................................
C ---COMPUTE T COEFFICIENTS---
C .................................................................
C **********************
ENTRY TCOF
C **********************
N1 = 1
N2 = KO
N4 = 1
20 DO 45 K = N1, N2
   J4 = 1
   DO 42 I = 1, I1
      J5 = JDIML2(I) - JDIML1(I) + J4
      J6 = J5 + JDIML2(I + 1) - JDIML1(I + 1) + 1
   DO 39 IJ = J4, J5
      IF (T(IJ, K).EQ.0.) GO TO 39
      NJA = IJ + 1
      IF (NJA.EQ.J5 + 1) NJA = 1
      J = IJ - J4 + JDIML1(I)
   40 CONTINUE
   IFUJ.EQ.J4) NJA = 1
   TR(NJB, KO) = 0.
   TR(IJ, KO) = 0.
   TC(IJ, KO) = 0.
   NIB = IJ - JDIML2(I - 1) + JDIML1(I - 1)
   IF (I.EQ.2) NIB = 1
   IF (NIB.GE.J4) NIB = 1
   IF (NIB.LT.J6) NIB = 1
   TC(NIB, KO) = 0.
   IF (KO.NE.1) TK(IJ, K1) = 0.
   PHI(IJ, KO) = 1.D30
10 CONTINUE
   J6 = J4
12 J4 = J5 + 1
   IF (N3.EQ.1) RETURN
   N1 = KO
   N2 = KO
   N4 = K1
   GO TO 20

C .................................................................
C ---COMPUTE T COEFFICIENTS---
C .................................................................
C **********************
ENTRY TCOF
C **********************
N1 = 1
N2 = KO
N4 = 1
20 DO 45 K = N1, N2
   J4 = 1
   DO 42 I = 1, I1
      J5 = JDIML2(I) - JDIML1(I) + J4
      J6 = J5 + JDIML2(I + 1) - JDIML1(I + 1) + 1
   DO 39 IJ = J4, J5
      IF (T(IJ, K).EQ.0.) GO TO 39
      NJA = IJ + 1
      IF (NJA.EQ.J5 + 1) NJA = 1
      J = IJ - J4 + JDIML1(I)
   40 CONTINUE
   IFUJ.EQ.J4) NJA = 1
   TR(NJB, KO) = 0.
   TR(IJ, KO) = 0.
   TC(IJ, KO) = 0.
   NIB = IJ - JDIML2(I - 1) + JDIML1(I - 1)
   IF (I.EQ.2) NIB = 1
   IF (NIB.GE.J4) NIB = 1
   IF (NIB.LT.J6) NIB = 1
   TC(NIB, KO) = 0.
   IF (KO.NE.1) TK(IJ, K1) = 0.
   PHI(IJ, KO) = 1.D30
10 CONTINUE
   J6 = J4
12 J4 = J5 + 1
   IF (N3.EQ.1) RETURN
   N1 = KO
   N2 = KO
   N4 = K1
   GO TO 20

C .................................................................
C ---COMPUTE T COEFFICIENTS---
C .................................................................
C **********************
ENTRY TCOF
C **********************
N1 = 1
N2 = KO
N4 = 1
20 DO 45 K = N1, N2
   J4 = 1
   DO 42 I = 1, I1
      J5 = JDIML2(I) - JDIML1(I) + J4
      J6 = J5 + JDIML2(I + 1) - JDIML1(I + 1) + 1
   DO 39 IJ = J4, J5
      IF (T(IJ, K).EQ.0.) GO TO 39
      NJA = IJ + 1
      IF (NJA.EQ.J5 + 1) NJA = 1
      J = IJ - J4 + JDIML1(I)
   40 CONTINUE
   IFUJ.EQ.J4) NJA = 1
   TR(NJB, KO) = 0.
   TR(IJ, KO) = 0.
   TC(IJ, KO) = 0.
   NIB = IJ - JDIML2(I - 1) + JDIML1(I - 1)
   IF (I.EQ.2) NIB = 1
   IF (NIB.GE.J4) NIB = 1
   IF (NIB.LT.J6) NIB = 1
   TC(NIB, KO) = 0.
   IF (KO.NE.1) TK(IJ, K1) = 0.
   PHI(IJ, KO) = 1.D30
10 CONTINUE
   J6 = J4
12 J4 = J5 + 1
   IF (N3.EQ.1) RETURN
   N1 = KO
   N2 = KO
   N4 = K1
   GO TO 20

54
APPENDIX 2
LISTING OF SOURCE CODE--Continued

IF (T(NJA,K).EQ.O.) GO TO 30
TR(IJ,K)=(2.*T(NJA,K)*T(IJ,K))/(T(IJ,K)*DELCX(J+1)+T(NJA,K)*
1DELX(J))*FACT(K,1)
30 NIA=IJ+JDIML2(I)-JDIML1(I+1)+1
IF(NIA.GT.J6) NIA=1
IF(I.EQ.I1) NIA=1
IF (T(NIA,K).EQ.O.) GO TO 39
TC(IJ,K)=(2.*T(NIA,K)*T(IJ,K))/(T(IJ,K)*DELY(I+1)+T(NIA,K)*
1DELY(I))*FACT(K,2)
39 CONTINUE
42 J4=J5+1
45 CONTINUE
IF (KO.EQ.1.OR.ITK.EQ.ICHK(10).OR.N3.EQ.O) RETURN
53 K=N4,K1
J4=1
DO 52 I=2,I1
J5=JDIML2(I)-JDIML1(I)+J4
50 DO 50 IJ=1,IOJO
IF (ZCB(IJ,K).LE.O.) GO TO 55
T1=T(IJ,K)*FACT(K,3)
T2=T(IJ,K+1)*FACT(K+1,3)
TK(IJ,K)=(2.*T2*T1)/(T1*DELZ(K+1)+T2*DELZ(K))
50 CONTINUE
52 J4=J5+1
53 CONTINUE
RETURN

---COMPUTE COEFFICIENTS FOR TRANSIENT PART OF LEAKAGE TERM---
******************************************************************************
ENTRY CLAY
******************************************************************************
R24 = 1.0/24.0
P1 = 3.1415927
P2 = 9.8696044
P4 = 97.409091
F0 = P2/6.0
G0 = -P2/12.0
G1 = -7.0*P4/720.0
G2 = -0.5
DO 60 K=1,K0
DO 60 IJ=1,IOJO
TL(IJ,K) = 0.0
60 TLK(IJ,K) = 0.0
IF (SS(1).GT.O.) GO TO 65
DO 55 K=1,K1
KP1 = K + 1
DO 55 IJ=1,IOJO
IF (ZCB(IJ,K).LE.O.) GO TO 55

55
APPENDIX 2
LISTING OF SOURCE CODE--Continued

AFACT = RATE(IJ,K)/ZCB(IJ,K)
TLK(IJ,K) = AFACT
TL(IJ,KP1) = AFACT
TL(IJ,K) = TL(IJ,K) + AFACT
55 CONTINUE
RETURN
65 IF(KP.NE.1.OR.KT.NE.1) GO TO 160
IF (MODE.GT.0) GO TO 80
B(1) = GO
B1(1) = FO
GO TO 160
80 IF (MODE.GT.1) GO TO 90
B(1) = GO
GO TO 135
90 DO 100 J=1,MODE
B(J) = 0.0
RJ2 = J*J
DO 100 I=1,MODE
I3 = I - 3
100 A(I,J) = RJ2**I3
B(1) = G1
B(2) = G0
B(3) = G2
DO 110 I=2,MODE
RAIJ = 1.0/A(I-1,I-1)
DO 110 K=I,MODE
AA = A(K,I-1)*RAIJ
B(K) = B(K) - B(I-1)*AA
DO 110 J=I,MODE
B1(K) = 1.0 + CN
110 A(K,J) = A(K,J) - AA*A(I-1,J)
B(MODE) = B(MODE)*A(MODE,MODE)
DO 130 K=2,MODE
J = MODE + 2 - K
JJ = J - 1
BJJ = B(JJ)
DO 120 I=J,MODE
B1(J) = B1(J) - A(JJ,I)*B(I)
120 BJ = BJJ - A(JJ,I)*B(I)
130 B(JJ) = BJ/J/A(JJ,JJ)
135 SUMN2 = 0.
SUMN4 = 0.
DO 140 K=1,MODE
B1(K) = 1.0
L = MODE + 1 - K
B(L+1) = B(L)
SUMN4 = SUMN4 + 1.0/(K*K*K*K*1.0)
140 SUMN2 = SUMN2 + 1.0/(K*K*1.0)
CN = (P4/90.0-SUMN4)*MODE**4
B1(MODE+1) = 1.0 + CN
B1(1) = FO - SUMN2 - CN/(1.0*MODE*MODE)
B(1) = 0.0
DO 145 K=1,MODE
56
APPENDIX 2
LISTING OF SOURCE CODE--Continued

RK = 1.0/(1.0*K*K)
RK = RK*RK
B1(K+1) = RK*B1(K+1)

DO 150 K=1,K1
DO 150 IJ=2,IOJO
---SKIP COMPUTATIONS IF T, RATE OR M = 0, OR IF CONSTANT
HEAD BOUNDARY---
IF (T(IJ,K).LE.0..OR.T(IJ,K).EQ.0..OR.ZCB(IJ,K).EQ.0..OR.
T(IJ,K+1).EQ.0.) GO TO 155
RATE(IJ,K) = RATE(IJ,K)/ZCB(IJ,K)
ZCB(IJ,K) = RATE(IJ,K)*P2/(ZCB(IJ,K)*SS(K))
IF(IDK1.EQ.ICHK(4).OR.IPU1.EQ.ICHK(8)) GO TO 150
DO 151 M=1,MODE
RM(1,M,IJ,K) = 0.0
RM(2,M,IJ,K) = 0.0
GO TO 150

151 RM(2,M,IJ,K) = 0.0
GO TO 150

155 RATE(IJ,K) = - 1.0
150 CONTINUE
160 CONTINUE
DO 180 K=1,K1
DO 180 IJ=2,IOJO
---SKIP COMPUTATIONS IF T, RATE OR M = 0, OR IF CONSTANT
HEAD BOUNDARY---
IF (T(IJ,K).LT.0.0) GO TO 180
XLM = ZCB(IJ,K)
XLMT = XLM*DELT
RXLMT = 1.0/XLMT
XLMT1 = XLM*DELTM1
RXLMT1 = 1.0/XLMT1
TLN = RATE(IJ,K)
HIM1 = PHI(IJ,K)
HIM2 = PHI(IJ,K+1)
SLN = TLN*(HIM2-HIM1)
SLM = TLN*(HIM1-HIM2)
C1 = 2.*TLN*RXLMT
C2 = 2.*TLN*RXLMT1
TLN = 0.5*( TLN - C1*P2/6.0)
TLN = 0.5*( TLN + C1*P2/3.0)
IF (MODE.LT.1) GO TO 175
C1 = C1*RXLMT
C2 = C2*RXLMT1*DELTM1
RDELT = 1.0/DELT
OLDM1 = XI(IJ,K)
OLDM2 = XI(IJ,K+1)
DO 170 M=1,MODE
BM = B(M+1)
B1M = B1(M+1)
XN = M*M
XPP = XN*XLMT1
TEX = XPP*(24.+XPP*(-12.+XPP*(4.-XPP))) R24
APPENDIX 2
LISTING OF SOURCE CODE--Continued

IF (XPP.GT.1.0E-01) TEX = 1.0-EXP(-XPP)
0001396
XPN = XN*XLMT
0001397
XXN = XPN*(24.+XPN*(-12.+XPN*(4.-XPN)))*R24
0001398
IF (XPN.GT.1.0E-01) XXN = 1.0-EXP(-XPN)
0001399
DTEX = 1.0 - XXN
0001400
XXE = C1*XXN
0001401
TLN = TLN - XXE*B1M
0001402
TLM = TLM - XXE*BM
0001403
RM(1,M,IJ,K) = RM(1,M,IJ,K) - C2*(B1M*OLDM1-BM*OLDM2)*TEX
0001404
RM(2,M,IJ,K) = RM(2,M,IJ,K) - C2*(B1M*OLDM2-BM*OLDM1)*TEX
0001405
SLN = SLN + RM(1,M,IJ,K)*XXN*RDELT
0001406
SLM = SLM + RM(2,M,IJ,K)*XXN*RDELT
0001407
RM(1,M,IJ,K) = RM(1,M,IJ,K)*DTEX
0001408
RM(2,M,IJ,K) = RM(2,M,IJ,K)*DTEX
0001409
SLN = SLN - TLN*HIM1 - TLM*HIM2
0001410
SLM = SLM - TLN*HIM2 - TLM*HIM1
0001411
TL(IJ,K) = TL(IJ,K) + TLN
0001412
TL(IJ,K+1) = TLN
0001413
TLK(IJ,K) = TLM
0001414
SL(IJ,K) = SLM
0001415
SL(IJ,K+1) = SLM
0001416
180 CONTINUE
0001417
RETURN
0001418
600 FORMAT ('-',20('*'),'WELL',213,'
0001419
610 FORMAT ('-',20('*'),'NODE',213,'
0001420
END

BLOCK DATA
0001421

SPECIFICATIONS:
0001422
COMMON /SARRAY/ ICHK(13)
0001423
COMMON /PR/ DIGIT(129),VF4(12),VF5(12),
0001424
1VF6(12),VF7(12),VF8(12),VF9(12),VF10(12)
0001425

DATA ICHK/'DRAW','HEAD','MASS','DK1','DK2','WATE','RECH','PUN1','P0001426
1UN2','ITKR','EQN3','ITLR','MPTY'/
0001427
DATA DIGIT/'1','2','3','4','5','6','7','8','9','10','11','12','13'0001428
1,'14','15','16','17','18','19','20','21','22','23','24','25','26',
2,'27','28','29','30','31','32','33','34','35','36','37','38','39',
3,'40','41','42','43','44','45','46','47','48','49','50','51','52',
5,'53','54','55','56','57','58','59','60','61','62','63','64','65',
6,'66','67','68','69','70','71','72','73','74','75','76','77','78',
7,'79','80','81','82','83','84','85','86','87','88','89','90','91',
8,'92','93','94','95','96','97','98','99','100','101','102','103',
9,'104','105','106','107','108','109','110','111','112','113','114',
10,'115','116','117','118','119','120','121','122','123','124',
11,'125','126','127','128','129','130','131','132','133','134','135'
0001429

DATA VF4/('',' ',' ',' ',' ','F10.','.4(/8','F10.','.4)),3 0001430
1*/

58
APPENDIX 2
LISTING OF SOURCE CODE--Continued

DATA VF5/( 'I3', ' ', ' ', 'X', ' ', 'F6.1', '/(5X', 00014280
1', '20F', '6.1')', ')' 2* ' /
DATA VF6/( ' ', ' ', ' ', ' ', 'F4.0', '/(20', 'F4.0', ')'), 3 00014290
1*' '/
DATA VF7/( 'I6', ' ', ' ', 'X', ' ', 'F9.5', '/(1H', ' ', 5X', 00014300
1', '14F', '9.5')', ' ')
DATA VF8/( 'I6', ' ', ' ', 'X', '1P', ' ', 'E12', '5/(1', 'H', 5' 00014310
1', 'X', 'P', '10E1', '2.5'), ')'/
DATA VF9/( 'I6', ' ', ' ', 'X', '1P', ' ', 'E11', '3/(1', 'H', 5' 00014320
1', 'X', '1P', '10E1', '1.3'), ')'/
DATA VF10/( 'I5', ' ', ' ', 'X', '1P', ' ', 'F7.2', '/(5X', '18F' 00014330
1', '7.2', ')'), 2* ' /
******************************************************************************
END 00014340