Techniques of Water-Resources Investigations of the United States Geological Survey

Chapter A1

A MODULAR THREE-DIMENSIONAL FINITE-DIFFERENCE GROUND-WATER FLOW MODEL

By Michael G. McDonald and Arlen W. Harbaugh

This chapter supersedes U.S. Geological Survey Open-File Report 83-875

Book 6

MODELING TECHNIQUES
A copy of the source program of the Modular Model is available at cost, from:
Chief, Office of Ground Water, MS 411, National Center,
U.S. Geological Survey, Reston, Virginia 22092
PREFACE

The series of manuals on techniques describes procedures for planning and executing specialized work in water-resources investigations. The material is grouped under major subject headings called books and further subdivided into sections and chapters. Section A of Book 6 is on ground-water modeling.

Provisional drafts of chapters are distributed to field offices of the U.S. Geological Survey for their use. These drafts are subject to revision because of experience in use or because of advancement in knowledge, techniques, or equipment. After the technique described in a chapter is sufficiently developed, the chapter is published and is for sale by the Books and Open-File Reports Section, U.S. Geological Survey, Federal Center, Box 25425, Denver, Colorado 80225.

Reference to trade names, commercial products, manufacturers, or distributors in this manual constitutes neither endorsement by the U.S. Geological Survey nor recommendation for use.
The U.S. Geological Survey publishes a series of manuals describing procedures for planning and conducting specialized work in water-resources investigations. The manuals published to date are listed below and may be ordered by mail from the U.S. Geological Survey, Books and Open-File Reports, Federal Center, Box 25425, Denver, Colorado 80222 (an authorized agent of the Superintendent of Documents, Government Printing Office).

Prepayment is required. Remittance should be sent by check or money order payable to U.S. Geological Survey. Prices are not included in the listing below as they are subject to change. Current prices can be obtained by writing to the USGS, Books and Open File Reports. Prices include cost of domestic surface transportation. For transmittal outside the U.S.A. (except to Canada and Mexico) a surcharge of 25 percent of the net bill should be included to cover surface transportation. When ordering any of these publications, please give the title, book number, chapter number, and “U.S. Geological Survey Techniques of Water-Resources Investigations.”


# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>1- 1</td>
</tr>
<tr>
<td>Chapter 1. Introduction</td>
<td>1- 2</td>
</tr>
<tr>
<td>Purpose</td>
<td>1- 2</td>
</tr>
<tr>
<td>Organization of This Report</td>
<td>1- 3</td>
</tr>
<tr>
<td>Acknowledgement</td>
<td>1- 7</td>
</tr>
<tr>
<td>Chapter 2. Derivation of the Finite-Difference Equation</td>
<td>2- 1</td>
</tr>
<tr>
<td>Mathematical Model</td>
<td>2- 1</td>
</tr>
<tr>
<td>Discretization Convention</td>
<td>2- 2</td>
</tr>
<tr>
<td>Finite-Difference Equation</td>
<td>2- 5</td>
</tr>
<tr>
<td>Iteration</td>
<td>2-20</td>
</tr>
<tr>
<td>Formulation of equations for solution</td>
<td>2-25</td>
</tr>
<tr>
<td>Types of Model Cell and Simulation of Boundaries</td>
<td>2-27</td>
</tr>
<tr>
<td>Conceptual Aspects of Vertical Discretization</td>
<td>2-29</td>
</tr>
<tr>
<td>Chapter 3. Program Design</td>
<td>3- 1</td>
</tr>
<tr>
<td>Overall Structure</td>
<td>3- 1</td>
</tr>
<tr>
<td>Array Boundaries and Aquifer Boundaries</td>
<td>3-14</td>
</tr>
<tr>
<td>Volumetric Budget</td>
<td>3-16</td>
</tr>
<tr>
<td>Space Allocation</td>
<td>3-22</td>
</tr>
<tr>
<td>Three-Dimensional Subscripts for Model Arrays</td>
<td>3-23</td>
</tr>
<tr>
<td>Input Structure</td>
<td>3-24</td>
</tr>
<tr>
<td>Output Structure</td>
<td>3-28</td>
</tr>
<tr>
<td>Main Program</td>
<td>3-29</td>
</tr>
<tr>
<td>FORTRAN Listing of the Main Program</td>
<td>3-32</td>
</tr>
<tr>
<td>Chapter 4. Basic Package</td>
<td>4- 1</td>
</tr>
<tr>
<td>Conceptualization and Implementation</td>
<td>4- 1</td>
</tr>
<tr>
<td>Selection of Major Options and Designation of Input Files</td>
<td>4- 2</td>
</tr>
<tr>
<td>The IBOUND Array</td>
<td>4- 2</td>
</tr>
<tr>
<td>Initial Conditions</td>
<td>4- 5</td>
</tr>
<tr>
<td>Discretization of Time</td>
<td>4- 5</td>
</tr>
<tr>
<td>Output</td>
<td>4- 5</td>
</tr>
<tr>
<td>Budget Calculations in the Basic Package</td>
<td>4- 8</td>
</tr>
<tr>
<td>Input Instructions</td>
<td>4- 9</td>
</tr>
<tr>
<td>Sample Input</td>
<td>4-13</td>
</tr>
<tr>
<td>Input Instructions for Output Control</td>
<td>4-14</td>
</tr>
<tr>
<td>Sample Input for Output Control</td>
<td>4-17</td>
</tr>
<tr>
<td>Module Documentation</td>
<td>4-18</td>
</tr>
<tr>
<td>BASIDF</td>
<td>4-19</td>
</tr>
<tr>
<td>BASIAL</td>
<td>4-23</td>
</tr>
<tr>
<td>BASIRP</td>
<td>4-27</td>
</tr>
<tr>
<td>BASIST</td>
<td>4-31</td>
</tr>
<tr>
<td>BASIAD</td>
<td>4-35</td>
</tr>
<tr>
<td>BASIFM</td>
<td>4-39</td>
</tr>
<tr>
<td>BASIOC</td>
<td>4-42</td>
</tr>
<tr>
<td>BASIOT</td>
<td>4-46</td>
</tr>
<tr>
<td>SBASID</td>
<td>4-51</td>
</tr>
<tr>
<td>SBASIH</td>
<td>4-55</td>
</tr>
<tr>
<td>SADSII</td>
<td>4-59</td>
</tr>
<tr>
<td>Chapter</td>
<td>Package Name</td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
</tr>
<tr>
<td>Chapter 9</td>
<td>Drain Package</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Chapter 10</td>
<td>Evapotranspiration Package</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Chapter 11</td>
<td>General-Head Boundary Package</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Chapter 12</td>
<td>Strongly Implicit Procedure Package</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Chapter 13</td>
<td>Slice-Successive Overrelaxation Package</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Chapter 14</td>
<td>Utility Modules</td>
</tr>
</tbody>
</table>
ILLUSTRATIONS

Figure 1. A discretized hypothetical aquifer system......................2- 3
2. Grids showing the difference between block-centered and point-centered formulations.................................2- 6
3. Cell i,j,k and the six adjacent cells............................2- 8
4. Flow into cell i,j,k from cell i,j-1,k............................2- 9
5. Conceptual representation of leakage through a riverbed into a cell......................................................2-14
6. Hydrograph for cell i,j,k...........................................2-17
7. Iterative calculation of a head distribution............................2-22
8. Discretized aquifer showing boundaries and cell designations..............................................2-28
9. Schemes of vertical discretization................................2-30
10. Possible pattern of flow in a cross section consisting of two high conductivity units separated by a low conductivity unit..............2-32
11. A cross section in which a low conductivity unit is represented by six model layers....................................2-33
12. A cross section in which a low conductivity unit is represented by the conductance between model layers..............................2-35
13. Overall program structure..............................................3- 2
14. Organization of modules by procedures and packages.................3- 6
15. Primary modules organized by procedure and package.................3- 8
16. Overall program structure showing all primary modules................3-12
17. Specification of major options using the IUNIT array................3-26
18. Sample input data showing role of the IUNIT array.....................3-27
19. Example of the boundary array (IBOUND) for a single layer..............4- 3
20. Flow of head distributions during a simulation......................4- 4
21. Division of simulation time into stress periods and time steps...............................................................4- 6
22. Sample overall volumetric water budget................................4- 7
23. Prism of porous material illustrating Darcy's law.........................5- 3
24. Calculation of conductance through several prisms in series..................5- 5
25. Calculation of conductance between nodes using transmissivities and dimensions of cells.......................................5- 7
26. Diagram for calculation of vertical leakance, $V_{cont}$, between two nodes which fall within a single geohydrologic unit..........................5-14
27. Diagram for calculation of vertical leakance, $V_{cont}$, between two nodes located at the midpoints of vertically adjacent geohydrologic units........................5-15
28. Diagram for calculation of vertical leakance, $V_{cont}$, between two nodes located at the midpoints of aquifers which are separated by a semiconfining unit.........................5-17
29. Situation in which a correction is required to limit the downward flow into cell i,j,k+1 as a result of partial desaturation of the cell......................................................5-20
30. A model cell which uses two storage factors during one iteration...........................................................5-29
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>32.</td>
<td>Discretization of a stream into reaches</td>
<td>6-2</td>
</tr>
<tr>
<td>33.</td>
<td>(a) Cross section of an aquifer containing a stream, and (b) conceptual</td>
<td>6-3</td>
</tr>
<tr>
<td></td>
<td>representation of stream-aquifer interconnection in simulation</td>
<td></td>
</tr>
<tr>
<td>34.</td>
<td>Idealization of streambed conductance in an individual cell</td>
<td>6-4</td>
</tr>
<tr>
<td>35.</td>
<td>Cross sections showing the relation between head at the base of the</td>
<td>6-7</td>
</tr>
<tr>
<td></td>
<td>streambed layer and head in the cell</td>
<td></td>
</tr>
<tr>
<td>36.</td>
<td>Flow between stream and node i,j,k as a function of head in the aquifer,</td>
<td>6-9</td>
</tr>
<tr>
<td></td>
<td>( h_{i,j,k} )</td>
<td></td>
</tr>
<tr>
<td>37.</td>
<td>Limiting seepage from a stream at unit hydraulic gradient</td>
<td>6-11</td>
</tr>
<tr>
<td>38.</td>
<td>Hypothetical problem showing which cells receive recharge under the three</td>
<td>7-3</td>
</tr>
<tr>
<td></td>
<td>options available in the Recharge Package</td>
<td></td>
</tr>
<tr>
<td>39.</td>
<td>Cross section through cell i,j,k illustrating head loss in convergent</td>
<td>9-2</td>
</tr>
<tr>
<td></td>
<td>flow into drain</td>
<td></td>
</tr>
<tr>
<td>40.</td>
<td>Factors affecting head loss immediately around a drain: (a) buried drain</td>
<td>9-4</td>
</tr>
<tr>
<td></td>
<td>pipe in backfilled ditch; (b) open drain</td>
<td></td>
</tr>
<tr>
<td>41.</td>
<td>Plot of flow into drain, ( Q_i^D ), vs. head in cell i,j,k using</td>
<td>9-6</td>
</tr>
<tr>
<td></td>
<td>equations (69-a) and (69-b)</td>
<td></td>
</tr>
<tr>
<td>42.</td>
<td>Volumetric evapotranspiration from cell i,j,k as a function of head (</td>
<td>10-3</td>
</tr>
<tr>
<td></td>
<td>( h_{i,j,k} )</td>
<td></td>
</tr>
<tr>
<td>43.</td>
<td>Hypothetical problem showing cells from which ET will be abstracted under</td>
<td>10-6</td>
</tr>
<tr>
<td></td>
<td>the two options available in the ET Package</td>
<td></td>
</tr>
<tr>
<td>44.</td>
<td>Schematic diagram illustrating principle of General-head boundary</td>
<td>11-2</td>
</tr>
<tr>
<td></td>
<td>package</td>
<td></td>
</tr>
<tr>
<td>45.</td>
<td>Graph of flow from source into cell i,j,k vs. head in the cell, as</td>
<td>11-3</td>
</tr>
<tr>
<td></td>
<td>computed by the General-head boundary package using equation (78)</td>
<td></td>
</tr>
<tr>
<td>46.</td>
<td>Correspondence between the finite-difference equations and the matrix</td>
<td>12-3</td>
</tr>
<tr>
<td></td>
<td>equation for a grid of three rows, four columns, and two layers</td>
<td></td>
</tr>
<tr>
<td>47.</td>
<td>Structure of coefficient matrix showing nonzero diagonals</td>
<td>12-4</td>
</tr>
<tr>
<td>48.</td>
<td>Symmetric coefficient matrix for a grid containing two rows, three</td>
<td>12-6</td>
</tr>
<tr>
<td></td>
<td>columns, and two layers</td>
<td></td>
</tr>
<tr>
<td>49.</td>
<td>Decomposition of a coefficient matrix into lower and upper triangular</td>
<td>12-7</td>
</tr>
<tr>
<td></td>
<td>matrices</td>
<td></td>
</tr>
<tr>
<td>50.</td>
<td>Structure of matrix ([A+B]) showing nonzero diagonals</td>
<td>12-10</td>
</tr>
<tr>
<td>51.</td>
<td>Structure, showing nonzero diagonals, of (a) the lower triangular factor</td>
<td>12-11</td>
</tr>
<tr>
<td></td>
<td>([L]) of ([A+B]), and (b) the upper triangular factor ([U]) of</td>
<td></td>
</tr>
<tr>
<td></td>
<td>([A+B])</td>
<td></td>
</tr>
<tr>
<td>52.</td>
<td>Estimation of a function at one corner of a rectangle in terms of the</td>
<td>12-14</td>
</tr>
<tr>
<td></td>
<td>values of the function at the other three corners</td>
<td></td>
</tr>
</tbody>
</table>
Figure 53. Cell numbering schemes for a grid using three indices and using one index ........................................12-18

54. Division of the three-dimensional model array into vertical slices for processing in the SSOR package ....13-2

55. Coefficient matrix for slice equations and corresponding computer storage array ......................13-9

56. Illustration of wrap and strip forms of printed output for a layer containing 7 rows and 17 columns ...14-2

TABLES

1. List of packages .............................................3-10

2. Print-format codes for utility modules
   ULAPRS and ULAPRW ......................................14-3
A MODULAR THREE-DIMENSIONAL FINITE-DIFFERENCE GROUND-WATER FLOW MODEL

By Michael G. McDonald and Arlen W. Harbaugh

ABSTRACT

This report presents a finite-difference model and its associated modular computer program. The model simulates flow in three dimensions. The report includes detailed explanations of physical and mathematical concepts on which the model is based and an explanation of how those concepts are incorporated in the modular structure of the computer program. The modular structure consists of a Main Program and a series of highly independent subroutines called "modules." The modules are grouped into "packages." Each package deals with a specific feature of the hydrologic system which is to be simulated, such as flow from rivers or flow into drains, or with a specific method of solving linear equations which describe the flow system, such as the Strongly Implicit Procedure or Slice-Successive Overrelaxation.

The division of the program into modules permits the user to examine specific hydrologic features of the model independently. This also facilitates development of additional capabilities because new packages can be added to the program without modifying the existing packages. The input and output systems of the computer program are also designed to permit maximum flexibility.

Ground-water flow within the aquifer is simulated using a block-centered finite-difference approach. Layers can be simulated as confined, unconfined, or a combination of confined and unconfined. Flow associated with external stresses, such as wells, areal recharge, evapotranspiration, drains, and streams, can also be simulated. The finite-difference equations can be solved using either the Strongly Implicit Procedure or Slice-Successive Overrelaxation.

The program is written in FORTRAN 77 and will run without modification on most computers that have a FORTRAN 77 compiler. For each program module, this report includes a narrative description, a flow chart, a list of variables, and a module listing.
Since their inception, the two- and three-dimensional finite-difference models described by Trescott (1975), Trescott and Larson (1976), and Trescott, Pinder, and Larson (1976) have been used extensively by the U.S. Geological Survey and others for the computer simulation of ground-water flow. The basic concepts embodied in those models have been incorporated in the model presented here. The primary objectives in designing a new ground-water flow model were to produce a program that could be readily modified, was simple to use and maintain, could be executed on a variety of computers with minimal changes, and was relatively efficient with respect to computer memory and execution time.

The model program documented in this report uses a modular structure wherein similar program functions are grouped together, and specific computational and hydrologic options are constructed in such a manner that each option is independent of other options. Because of this structure, new options can be added without the necessity of changing existing subroutines. In addition, subroutines pertaining to options that are not being used can be deleted, thereby reducing the size of the program. The model may be used for either two- or three-dimensional applications. Input procedures have been generalized so that each type of model input data may be stored and read from separate external files. Variable formatting allows input data arrays to be read in any format without modification to the program. The type of output that is available has also been generalized so that the user may select various model output options to suit a particular
The program was originally written using FORTRAN 66 (McDonald and Harbaugh, 1984). It has subsequently been modified to use FORTRAN 77. This report documents the FORTRAN 77 version. The program is highly portable; it will run, without modification, on most computers. On some computers, minor modification may be necessary or desirable. A discussion about program portability is contained in Appendix A.

The major options that are presently available include procedures to simulate the effects of wells, recharge, rivers, drains, evapotranspiration, and "general-head boundaries". The solution algorithms available include two iteration techniques, the Strongly Implicit Procedure (SIP) and the Slice-Successive Overrelaxation method (SSOR).

Organization of This Report

The purpose of this report is to describe the mathematical concepts used in this program, the design of the program, and the input needed to use the program. The program has been divided into a main program and a series of highly independent subroutines called modules. The modules, in turn, have been grouped into "packages." A package is a group of modules that deals with a single aspect of the simulation. For example, the Well Package simulates the effect of wells, the River Package simulates the effect of rivers, and the SIP Package solves a system of equations using the Strongly Implicit Procedure. Many of the packages represent options which the user may or may not have occasion to use. Each of the packages is described in a separate chapter of this report. Two preliminary chapters
describe topics relating to the overall program; Chapter 2 derives the
finite-difference equation that is used in the model and Chapter 3 describes
the overall design of the program. Chapter 14 describes utility modules
that are used by various packages to perform special tasks. Appendices A-E
cover topics relating to the operation of the model.

Chapters 4 through 13 describe individual packages. The description
of each package consists of (1) a section entitled "Conceptualization and
Implementation," (2) input instructions for the package, and (3) documenta-
tion of the individual modules contained in the package. The Conceptualiza-
tion and Implementation section describes the physical and mathematical
concepts used to build the package. For example, in the chapter describing
the River Package, an equation is derived which approximates flow through a
riverbed, and a discussion is provided to show how that equation can be
incorporated into the finite-difference equation. Chapters 12 and 13 des-
cribe the solution procedures currently available in the model.

The input instructions in Chapters 4 through 13 are presented in terms of
input "items." An item of input may be a single record or a collection of
similar records, or it may be an array or a collection of arrays.(In the model
described herein, three-dimensional arrays are always read as a collection of
two-dimensional arrays, one associated with each model layer.) The input
section in each chapter presents a list of the input items associated with
the package described in that chapter; the entries in this list are numbered,
and generally consist of two lines (sometimes followed by a note or comment).
For items which consist of a single record or a group of similar records,
the first line in the entry gives the names of the fields comprising the
records, while the second line shows the format of those fields, in standard
FORTRAN notation. For an input item which consists of an array, the first
line of the entry gives the name of the array, while the second line gives
the name of the utility module which reads the array. Further details
concerning utility modules are provided in Chapter 14.

For most of the packages, the list of input items is subdivided into
two major sections. One of these falls under the heading "FOR EACH SIMULATION"
and includes all items for which only one entry is needed in each simulation;
the other falls under the heading "FOR EACH STRESS PERIOD", and includes
those items for which several entries may be needed in each simulation (for
example, pumping rate, which may change with time during the period repre-
sented in a simulation). These major sections of the input list are further
subdivided by headings which indicate the modules (subroutines) which read
the item, or, in the case of an array, which call a utility subroutine to
read the array. Input items that are printed entirely in capital letters
are used as FORTRAN variables or arrays in the model program; input items
which appear in mixed upper and lower case print are terms used in the
instructions to describe the input fields or procedures, and do not appear
in the model itself as FORTRAN variables. Chapter 4, which describes the
Basic Package, includes two lists of input items; one of these describes
input which is always required, while the other describes input associated
with the optional "output control" section of the Basic Package.

An explanation of input fields is presented following the list of in-
put items in Chapters 4 through 13. This explanation is followed in most
cases by a sample input for the package under consideration. In Chapter 4,
again, the input items associated with the output control option are treated
separately; thus an independent explanation of fields and sample input are
In each simulation, the user must designate which of the options of the program are to be utilized, and must indicate the file from which the input for each option is to be read. This is done through a one-dimensional array, IUNIT; the entries in this array are the unit numbers associated with the required files by the computer operating system. A location in the IUNIT array is given at the beginning of the input sections in Chapters 5 through 13, and at the beginning of the input discussion for "output control" in Chapter 4. If the option is to be utilized, the user must enter, in the designated IUNIT array location, the unit number of the file or channel through which input for the option is to be read; if the option is not required a zero is entered in this location. Further discussion of the IUNIT array is provided in Chapters 3 and 4.

Following the input section in Chapters 4 through 13, each chapter provides a documentation of the modules making up the associated package. This documentation consists of a list of the modules in the package, followed by detailed descriptions of each of the modules. The detailed description of a module generally contains four documents: (1) a narrative description of the module, (2) a flow chart of the module, (3) a FORTRAN listing of the module, and (4) a list of the variable names which are used in the module. For very simple modules, the flow chart is omitted. The narrative description is a numbered list of the functions performed by the module showing the order in which they are performed. The flow chart is a graphic equivalent of the narrative. The blocks in the flow chart are numbered with the same numbers used in the narrative so that the two documents can be cross referenced. An explanation of terms used in the flow chart is contained on the sheet
with the flow chart. The program listing contains comments with numbers corresponding to those used in the flow charts and the narratives. The fourth record of the listing contains a comment showing the time and day that the module was last modified. The list of variables shows the name, range, and definition of every variable used in the module. If the variable is used only in that module, its range is given as "Module"; if it is used in other modules of the package, but not outside the package, its range is given as "Package"; if it is used in the modules of more than one package, its range is given as "Global."

To summarize the organization of this report, Chapters 2 and 3, and the "Conceptualization and Implementation" section of Chapter 4, provide discussions relevant to the overall design and functioning of the program; the formulation of coefficients representing flow within the aquifer is discussed under "Conceptualization and Implementation" in Chapter 5; Chapters 6 through 11 provide discussions of particular external sources or sinks and their representation in the model; and Chapters 12 and 13 discuss the operation of particular solvers for the systems of finite difference equations generated in the model. Input instructions for each package are provided in the relevant chapter; a discussion of input for utility modules is provided in Chapter 14. The appendices provide a sample problem, abbreviated input instructions, and discussions of certain computer-related topics.

Acknowledgement

The authors wish to extend special thanks to Gordon Bennett. In addition to providing the administrative support for the model development, he provided encouragement and guidance along the way. His critical review of the report greatly improved its clarity.
The three-dimensional movement of ground water of constant density through porous earth material may be described by the partial-differential equation
\[
\frac{\partial}{\partial x} (K_{xx} \frac{\partial h}{\partial x}) + \frac{\partial}{\partial y} (K_{yy} \frac{\partial h}{\partial y}) + \frac{\partial}{\partial z} (K_{zz} \frac{\partial h}{\partial z}) - W = S_s \frac{\partial h}{\partial t}
\]
(1)

where

- \(K_{xx}, K_{yy}, K_{zz}\) are values of hydraulic conductivity along the x, y, and z coordinate axes, which are assumed to be parallel to the major axes of hydraulic conductivity (L·t⁻¹);
- \(h\) is the potentiometric head (L);
- \(W\) is a volumetric flux per unit volume and represents sources and/or sinks of water (t⁻¹);
- \(S_s\) is the specific storage of the porous material (L⁻¹); and
- \(t\) is time (t).

For a derivation of equation (1) see for example Rushton and Redshaw (1979). In general, \(S_s, K_{xx}, K_{yy}, K_{zz}\) may be functions of space \((S_s = S_s(x,y,z), K_{xx} = K_{xx}(x,y,z), \text{etc.})\) and \(W\) may be a function of space and time \((W = W(x,y,z,t))\); equation (1) describes ground-water flow under nonequilibrium conditions in a heterogeneous and anisotropic medium, provided the principal axes of hydraulic conductivity are aligned with the coordinate directions.

Equation (1), together with specification of flow and/or head conditions at the boundaries of an aquifer system and specification of initial-head conditions, constitutes a mathematical representation of a ground-water flow system. A solution of equation (1), in an analytical sense, is an algebraic expression giving \(h(x,y,z,t)\) such that, when the derivatives of \(h\) with
respect to space and time are substituted into equation (1), the equation and its initial and boundary conditions are satisfied. A time-varying head distribution of this nature characterizes the flow system, in that it measures both the energy of flow and the volume of water in storage, and can be used to calculate directions and rates of movement.

Except for very simple systems, analytical solutions of equation (1) are rarely possible, so various numerical methods must be employed to obtain approximate solutions. One such approach is the finite-difference method, wherein the continuous system described by equation (1) is replaced by a finite set of discrete points in space and time, and the partial derivatives are replaced by terms calculated from the differences in head values at these points. The process leads to systems of simultaneous linear algebraic difference equations; their solution yields values of head at specific points and times. These values constitute an approximation to the time-varying head distribution that would be given by an analytical solution of the partial-differential equation of flow.

The finite-difference analog of equation (1) may be derived by applying the rules of difference calculus; however, in the discussion presented here, an alternative approach is used with the aim of simplifying the mathematical treatment and explaining the computational procedure in terms of familiar physical concepts regarding the flow system.

**Discretization Convention**

Figure 1 shows a spatial discretization of an aquifer system with a mesh of blocks called cells, the locations of which are described in terms of rows, columns, and layers. An i,j,k indexing system is used. For a system...
Figure 1.—A discretized hypothetical aquifer system.
consisting of "nrow" rows, "ncol" columns, and "nlay" layers. i is the row index, \( i = 1, 2, \ldots \) nrow; j is the column index, \( j = 1, 2, \ldots \) ncol; and k is the layer index, \( k = 1, 2, \ldots \) nlay. For example, figure 1 shows a system with nrow = 5, ncol = 9, and nlay = 5. In formulating the equations of the model, an assumption was made that layers would generally correspond to horizontal geohydrologic units or intervals. Thus in terms of Cartesian coordinates, the k index denotes changes along the vertical, z; because the convention followed in this model is to number layers from the top down, an increment in the k index corresponds to a decrease in elevation. Similarly rows would be considered parallel to the x axis, so that increments in the row index, i, would correspond to decreases in y; and columns would be considered parallel to the y axis, so that increments in the column index, j, would correspond to increases in x. These conventions were followed in constructing figure 1; however, applications of the model requires only that rows and columns fall along consistent orthogonal directions within the layers, and does not require the designation of x, y, or z coordinate axes.

Following the conventions used in figure 1, the width of cells in the row direction, at a given column, j, is designated \( \Delta r_j \); the width of cells in the column direction at a given row, i, is designated \( \Delta c_i \); and the thickness of cells in a given layer, k, is designated \( \Delta v_k \). Thus a cell with coordinates \((i, j, k) = (4, 8, 3)\) has a volume of \( \Delta r_4 \Delta c_8 \Delta v_3 \).
Within each cell there is a point called a "node" at which head is to be calculated. Figure 2 illustrates, in two dimensions, two conventions for defining the configuration of cells with respect to the location of nodes—the block-centered formulation and the point-centered formulation. Both systems start by dividing the aquifer with two sets of parallel lines which are orthogonal. In the block-centered formulation, the blocks formed by the sets of parallel lines are the cells; the nodes are at the center of the cells. In the point-centered formulation, the nodes are at the intersection points of the sets of parallel lines, and cells are drawn around the nodes with faces halfway between nodes. In either case, spacing of nodes should be chosen so that the hydraulic properties of the system are, in fact, generally uniform over the extent of a cell. The finite-difference equation developed in the following section holds for either formulation; however, only the block-centered formulation is presently used in the model.

In equation (1), the head, h, is a function of time as well as space so that, in the finite-difference formulation, discretization of the continuous time domain is also required.

**Finite-Difference Equation**

Development of the ground-water flow equation in finite-difference form follows from the application of the continuity equation: the sum of all flows into and out of the cell must be equal to the rate of change in storage within the cell. Under the assumption that the density of ground water is constant, the continuity equation expressing the balance of flow for a cell is

\[ \sum Q_i = \frac{\Delta h}{
\]
Figure 2.—Grids showing the difference between block-centered and point-centered grids.
where

\[ Q_i \] is a flow rate into the cell (L^3t^{-1});

SS has been introduced as the notation for specific storage in the finite-difference formulation; its definition is equivalent to that of \( S_s \) in equation (1)---i.e., it is the volume of water which can be injected per unit volume of aquifer material per unit change in head (L^{-1});

\[ \Delta V \] is the volume of the cell (L^3); and

\[ \Delta h \] is the change in head over a time interval of length \( \Delta t \).

The term on the right hand side is equivalent to the volume of water taken into storage over a time interval \( \Delta t \) given a change in head of \( \Delta h \). Equation (2) is stated in terms of inflow and storage gain. Outflow and loss are represented by defining outflow as negative inflow and loss as negative gain.

Figure 3 depicts a cell \( i,j,k \) and six adjacent aquifer cells \( i-1,j,k \); \( i+1,j,k \); \( i,j-1,k \); \( i,j+1,k \); \( i,j,k-1 \); and \( i,j,k+1 \). To simplify the following development, flows are considered positive if they are entering cell \( i,j,k \); and the negative sign usually incorporated in Darcy's law has been dropped from all terms. Following these conventions, flow into cell \( i,j,k \) in the row direction from cell \( i,j-1,k \) (figure 4), is given by Darcy's law as

\[ q_{i,j-1/2,k} = K R_{i,j-1/2,k} \Delta C_{i,j} \Delta V_k \frac{(h_{i,j-1,k} - h_{i,j,k})}{\Delta r_{j-1/2}} \]  

(3)

where

\[ h_{i,j,k} \] is the head at node \( i,j,k \), and \( h_{i,j-1,k} \) that at node \( i,j-1,k \);

\[ q_{i,j-1/2,k} \] is the volumetric fluid discharge through the face between cells \( i,j,k \) and \( i,j-1,k \) (L^3t^{-1});
Figure 3.—Cell i,j,k and indices for the six adjacent cells.
Figure 4.—Flow into cell $i,j,k$ from cell $i,j-l,k$. 
KR_{i,j-1/2,k} is the hydraulic conductivity along the row between nodes i,j,k and i,j-1,k (Lt^{-1});  
\Delta c_{i} \Delta v_{k} is the area of the cell faces normal to the row direction; and  
\Delta r_{j-1/2} is the distance between nodes i,j,k and i,j-1,k (L).

Although the discussion is phrased in terms of flow into the central cell, it can be misleading to associate the subscript j-1/2 of equation (3) with a specific point between the nodes. Rather, the term KR_{i,j-1/2,k} of equation (3) is the effective hydraulic conductivity for the entire region between the nodes, normally calculated as a harmonic mean in the sense described by, for example, Collins (1961). If this is done, equation (3) gives the exact flow, for a one-dimensional steady-state case, through a block of aquifer extending from node i,j-1,k to node i,j,k and having a cross sectional area \Delta c_{i} \Delta v_{k}.

Similar expressions can be written approximating the flow into the cell through the remaining five faces, i.e., for flow in the row direction through the face between cells i,j,k and i,j+1,k,

\[ q_{i,j+1/2,k} = \frac{(h_{i+1,j,k} - h_{i,j,k})}{\Delta r_{j+1/2}} \Delta c_{i} \Delta v_{k} \]  

(4)

while for the column direction, flow into the block through the forward face is

\[ q_{i+1/2,j,k} = \frac{(h_{i+1,j,k} - h_{i,j,k})}{\Delta c_{i+1/2}} \Delta r_{j} \Delta v_{k} \]  

(5)

and flow into the block through the rear face is

\[ q_{i-1/2,j,k} = \frac{(h_{i-1,j,k} - h_{i,j,k})}{\Delta r_{i-1/2}} \Delta c_{i} \Delta v_{k} \]  

(6)
For the vertical direction, inflow through the bottom face is

$$q_{i,j,k+1/2} = KV_{i,j,k+1/2} \frac{(h_{i,j,k+1} - h_{i,j,k})}{\Delta v_{k+1/2}}$$

while inflow through the upper face is given by

$$q_{i,j,k-1/2} = KV_{i,j,k-1/2} \frac{(h_{i,j,k-1} - h_{i,j,k})}{\Delta v_{k-1/2}}$$

Each of equations (3)-(8) expresses inflow through a face of cell $i,j,k$ in terms of heads, grid dimensions, and hydraulic conductivity. The notation can be simplified by combining grid dimensions and hydraulic conductivity into a single constant, the "hydraulic conductance" or, more simply, the "conductance." For example,

$$CR_{i,j-1/2,k} = K R_{i,j-1/2,k} \frac{\Delta c_i \Delta v_k}{\Delta r_{j-1/2}}$$

where

$$CR_{i,j-1/2,k}$$

is the conductance in row $i$ and layer $k$ between nodes $i,j-1,k$ and $i,j,k$ ($L^2t^{-1}$).

Conductance is thus the product of hydraulic conductivity and cross-sectional area of flow divided by the length of the flow path (in this case, the distance between the nodes.)
Substituting conductance from equation (9) into equation (3) yields

$$q_{i,j-1/2,k} = CR_{i,j-1/2,k}(h_{i,j-1,k} - h_{i,j,k}) \quad (10)$$

Similarly, equations (4)-(8) can be rewritten to yield

$$q_{i,j+1/2,k} = CR_{i,j+1/2,k}(h_{i,j+1,k} - h_{i,j,k}) \quad (11)$$
$$q_{i-1/2,j,k} = CC_{i-1/2,j,k}(h_{i-1,j,k} - h_{i,j,k}) \quad (12)$$
$$q_{i+1/2,j,k} = CC_{i+1/2,j,k}(h_{i+1,j,k} - h_{i,j,k}) \quad (13)$$
$$q_{i,j,k-1/2} = CV_{i,j,k-1/2}(h_{i,j,k-1} - h_{i,j,k}) \quad (14)$$
$$q_{i,j,k+1/2} = CV_{i,j,k+1/2}(h_{i,j,k+1} - h_{i,j,k}) \quad (15)$$

where the conductances are defined analogously to $CR_{i,j-1/2,k}$ in equation (9).

Equations (10)-(15) account for the flow into cell $i,j,k$ from the six adjacent cells. To account for flows into the cell from features or processes external to the aquifer, such as streams, drains, areal recharge, evapotranspiration or wells, additional terms are required. These flows may be dependent on the head in the receiving cell but independent of all other heads in the aquifer, or they may be entirely independent of head in the receiving cell. Flow from outside the aquifer may be represented by the expression

$$a_{i,j,k,n} = p_{i,j,k,n}h_{i,j,k} + q_{i,j,k,n} \quad (16)$$

where

$$a_{i,j,k,n}$$ represents flow from the $n$th external source into cell $i,j,k$ ($L^2 t^{-1}$), and $p_{i,j,k,n}$ and $q_{i,j,k,n}$ are constants ($L^2 t^{-1}$ and $L^3 t^{-1}$, respectively).

For example, suppose a cell is receiving flow from two sources, recharge from a well and seepage through a riverbed. For the first source ($n=1$),
since the flow from the well is assumed to be independent of head. \( p_{i,j,k,1} \) is zero and \( q_{i,j,k,1} \) is the recharge rate for the well. In this case,

\[
a_{i,j,k,1} = q_{i,j,k,1}
\]

For the second source (n=2), the assumption is made that the stream-aquifer interconnection can be treated as a simple conductance, so that the seepage is proportional to the head difference between the river stage and the head in cell \( i,j,k \) (figure 5); thus we have

\[
a_{i,j,k,2} = CRIV_{i,j,k,2}(R_{i,j,k} - h_{i,j,k})
\]

where \( R_{i,j,k} \) is the head in the river (L) and \( CRIV_{i,j,k,2} \) is a conductance \((L^2 t^{-1})\) controlling flow from the river into cell \( i,j,k \). For example, in the situation shown schematically in figure 5, \( CRIV \) would be given as the product of the vertical hydraulic conductivity of the riverbed material and the area of the streambed as it crosses the cell, divided by the thickness of the streambed material. Equation (18) can be rewritten as

\[
a_{i,j,k,2} = -CRIV_{i,j,k,2}h_{i,j,k} + CRIV_{i,j,k,2}R_{i,j,k}
\]

The negative conductance term, \(-CRIV_{i,j,k,2}\) corresponds to \( p_{i,j,k,2} \) of equation 16, while the term \( CRIV_{i,j,k,2}R_{i,j,k} \) corresponds to \( q_{i,j,k,2} \). Similarly, all other external sources or stresses can be represented by an expression of the form of equation 16. In general, if there are \( N \) external sources or stresses affecting a single cell, the combined flow is expressed by

\[
Q_{S_{i,j,k}} = \sum_{n=1}^{N} a_{i,j,k,n} = \sum_{n=1}^{N} p_{i,j,k,n} h_{i,j,k} + \sum_{n=1}^{N} q_{i,j,k,n}
\]

Defining \( P_{i,j,k} \) and \( Q_{i,j,k} \) by the expressions

2-13
Figure 5.—Conceptual representation of leakage through a riverbed into a cell.
the general external flow term for cell $i,j,k$ is

$$Q_{Si,j,k} = P_{i,j,k} h_{i,j,k} + Q_{i,j,k}.$$  

(21)

Applying the continuity equation (2) to cell $i,j,k$, taking into account the flows from the six adjacent cells, as well as the external flow rate, $Q_S$, yields

$$q_{i,j-1/2,k} + q_{i,j+1/2,k} + q_{i-1/2,j,k} + q_{i+1/2,j,k} + q_{i,j,k-1/2} + q_{i,j,k+1/2} + Q_{Si,j,k} = SS_{i,j,k} \Delta h_{i,j,k} \Delta t$$

(22)

where

$$\frac{\Delta h_{i,j,k}}{\Delta t}$$

is a finite-difference approximation for the derivative of head with respect to time ($L^{-1}$);

$SS_{i,j,k}$ represents the specific storage of cell $i,j,k$ ($L^{-1}$); and

$\Delta r_j \Delta c_i \Delta v_k$ is the volume of cell $i,j,k$ ($L^3$).

Equations (10) through (15) and (21) may be substituted into equation (22) to give the finite-difference approximation for cell $i,j,k$ as

$$CR_{i,j-1/2,k} (h_{i,j-1,k} - h_{i,j,k}) + CR_{i,j+1/2,k} (h_{i,j+1,k} - h_{i,j,k}) + CC_{i-1/2,j,k} (h_{i-1,j,k} - h_{i,j,k}) + CC_{i+1/2,j,k} (h_{i+1,j,k} - h_{i,j,k}) + CV_{i,j,k-1/2} (h_{i,j,k-1} - h_{i,j,k}) + CV_{i,j,k+1/2} (h_{i,j,k+1} - h_{i,j,k}) + P_{i,j,k} h_{i,j,k} + Q_{i,j,k} = SS_{i,j,k} \Delta h_{i,j,k} \Delta t.$$  

(23)
The finite-difference approximation for the time derivative of head,

\[ \frac{\Delta h_{i,j,k}}{\Delta t} \]

must next be expressed in terms of specific heads and times. Figure 6 shows a hydrograph of head values at node \( i,j,k \). Two values of time are shown on the horizontal axis: \( t_m \), which is the time at which the flow terms of equation (23) are evaluated; and \( t_{m-1} \), a time which precedes \( t_m \). The head values at node \( i,j,k \) associated with these times are designated by superscript as \( h_{i,j,k}^m \) and \( h_{i,j,k}^{m-1} \), respectively. An approximation to the time derivative of head at time \( t_m \) is obtained by dividing the head difference \( h_{i,j,k}^m - h_{i,j,k}^{m-1} \) by the time interval \( t_m - t_{m-1} \); that is,

\[ (\frac{\Delta h_{i,j,k}}{\Delta t})_m = \frac{h_{i,j,k}^m - h_{i,j,k}^{m-1}}{t_m - t_{m-1}} \]

Thus the hydrograph slope, or time derivative, is approximated using the change in head at the node over a time interval which precedes, and ends with, the time at which flow is evaluated. This is termed a backward-difference approach, in that \( \frac{\Delta h}{\Delta t} \) is approximated over a time interval which extends backward in time from \( t_m \), the time at which the flow terms are calculated. There are other ways in which \( \frac{\Delta h}{\Delta t} \) could be approximated; for example, we could approximate it over a time interval which begins at the time of flow evaluation and extends to some later time; or over a time interval which is centered at the time of flow evaluation, extending both forward and backward from it. These alternatives, however, may cause numerical instability—that is, the growth or propagation of error during the calculation of heads at successive times in a simulation.
Head

$t_{m-1}$ $t_m$ $t_{m+1}$

$h_{i,j,k}^m$ head at node $i,j,k$ at time $t_m$

$h_{i,j,k}^{m+1}$ head at end of time step $m$

$\Delta h_{i,j,k}^m$ Backward difference approximation to slope of hydrograph at time $t_m$

Figure 6.—Hydrograph for cell $i,j,k$. 
In an unstable situation, errors which enter the calculation for any reason at a particular time will increase at each succeeding time as the calculation progresses, until finally they completely dominate the result. By contrast, the backward-difference approach is always numerically stable—that is, errors introduced at any time diminish progressively at succeeding times. For this reason, the backward-difference approach is preferred even though it leads to large systems of equations which must be solved simultaneously for each time at which heads are to be computed.

Equation (23) can be rewritten in backward-difference form by specifying flow terms at $t_m$, the end of the time interval, and approximating the time derivative of head over the interval $t_{m-1}$ to $t_m$; that is:

$$\begin{align*}
&CR_{i,j-1/2,k}(h_{i,j-1,k} - h_{i,j,k}) + CR_{i,j+1/2,k}(h_{i,j+1,k} - h_{i,j,k}) \\
&+ CC_{1-1/2,j,k}(h_{i-1,j,k} - h_{i,j,k}) + CC_{1+1/2,j,k}(h_{i+1,j,k} - h_{i,j,k}) \\
&+ CV_{i,j-1/2}(h_{i,j,k-1} - h_{i,j,k}) + CV_{i,j,k+1/2}(h_{i,j,k+1} - h_{i,j,k}) \\
&+ p_{i,j,k} h_{i,j,k} + Q_{i,j,k} = SS_{i,j,k}(\Delta r_{j} \Delta c_{i} \Delta v_{k}) \frac{(h_{i,j,k} - h_{i,j,k})}{t_{m} - t_{m-1}}.
\end{align*}$$

Equation (24) is a backward-difference equation which can be used as the basis for a simulation of the partial differential equation of ground water flow, equation (1). Like the term $Q_{i,j,k}$, the coefficients of the various head terms in equation (24) are all known, as is the head at the beginning of the time step, $h_{i,j,k}$. The seven heads at time $t_m$, the end of the time step, are unknown; that is, they are part of the head distribution to be predicted. Thus equation (24) cannot be solved independently, since it represents a single equation in seven unknowns. However, an equation of this type can be written for each active cell in the mesh; and, since there is only one unknown head for each cell, we are left with a system of "n" equations in "n" unknowns. Such a system can be solved simultaneously.
The objective of transient simulation is generally to predict head distributions at successive times, given the initial head distribution, the boundary conditions, the hydraulic parameters and the external stresses. The initial-head distribution provides a value of $h_{i,j,k}^1$ at each point in the mesh—that is, it provides the values of head at the beginning of the first of the discrete time steps into which the time axis is divided in the finite-difference process. The first step in the solution process is to calculate values of $h_{i,j,k}^2$—that is, heads at time $t_2$, which marks the end of the first time step. In equation (25), therefore, the head superscript $m$ is taken as 2, while the superscript $m-1$, which appears in only one head term, is taken as 1. The equation therefore becomes

$$
\begin{align*}
&CR_{i,j-1/2,k}(h_{i,j-1,k}^2 - h_{i,j,k}^2) + CR_{i,j+1/2,k}(h_{i,j+1,k}^2 - h_{i,j,k}^2) \\
&+ CC_{i-1/2,j,k}(h_{i-1,j,k}^2 - h_{i,j,k}^2) + CC_{i+1/2,j,k}(h_{i+1,j,k}^2 - h_{i,j,k}^2) \\
&+ CV_{i,j,k-1/2}(h_{i,j,k-1}^2 - h_{i,j,k}^2) + CV_{i,j,k+1/2}(h_{i,j,k+1}^2 - h_{i,j,k}^2) \\
&+ p_{i,j,k}h_{i,j,k}^1 + Q_{i,j,k} \\
&= SS_{i,j,k} \frac{(\Delta r_j \Delta c_i \Delta v_k)(h_{i,j,k}^2 - h_{i,j,k}^1)}{t_2 - t_1}
\end{align*}
$$

where again the superscripts 1 and 2 refer to the time at which the heads are taken and should not be interpreted as exponents.

2-19
An equation of this form is written for every cell in the mesh in which head is free to vary with time (variable-head cells), and the system of equations is solved simultaneously for the heads at time $t_2$. When these have been obtained, the process is repeated to obtain heads at time $t_3$, the end of the second time step. To do this, equation (25) is reapplied, now using 2 as time subscript $m-1$ and 3 as time subscript $m$. Again, a system of equations is formulated, where the unknowns are now the heads at time $t_3$; and this set of equations is solved simultaneously to obtain the head distribution at time $t_3$. This process is continued for as many time steps as necessary to cover the time range of interest.

It is important to note that the set of finite-difference equations is reformulated at each time step; that is, at each step there is a new system of simultaneous equations to be solved. The heads at the end of the time step make up the unknowns for which this system must be solved; the heads at the beginning of the step are among the known terms in the equations. The solution process is repeated at each time step yielding a new array of heads for the end of the step.

**Iteration**

The model described in this report utilizes iterative methods to obtain the solution to the system of finite-difference equations for each time step. In these methods, the calculation of head values for the end of a given time step is started by arbitrarily assigning a trial value, or estimate, for the head at each node at the end of that step. A procedure of calculation is then initiated which alters these estimated values, producing a new set of head values which are in closer agreement with the system of equations. These new, or interim, head values then take the place of the initially assumed heads, and the procedure of calculation is repeated, producing
a third set of head values. This procedure is repeated successively, at each stage producing a new set of interim heads which more nearly satisfies the system of equations. Each repetition of the calculation is termed an "iteration." Ultimately, as the interim heads approach values which would exactly satisfy the set of equations, the changes produced by succeeding stages of calculation become very small. This behaviour is utilized in determining when to stop iteration, as discussed in a subsequent paragraph.

Thus, during the calculations for a time step, arrays of interim head values are generated in succession, each array containing one interim head value for each active node in the mesh. In figure 7, these arrays are represented as three-dimensional lattices, each identified by an array symbol, $\bar{h}$, bearing two superscripts. The first superscript indicates the time step for which the heads in the array are calculated, while the second indicates the number, or level, of the iteration which produced the head array. Thus $\bar{h}^{m,2}$ represents the array of values computed in the first iteration for the end of step $m$; $\bar{h}^{m,2}$ would represent the array of values computed in the second iteration; and so on. The head values which were initially assumed for the end of time step $m$, to begin the process of iteration, appear in the array designated $\bar{h}^{m,0}$. In the example of figure 7, a total of $n$ iterations is required to achieve closure for the heads at the end of time step $m$; thus the array of final head values for the time step is designated $\bar{h}^{m,n}$. Figure 7 also shows the array of final head values for the end of the preceding time step $\bar{h}^{m-1,n}$ (where again it is assumed that $n$ iterations were required for closure). The head values in this array appear in the storage term of equation (24)—i.e., they are used in the term $h_{i,j,k}^{m-1}$ on the right side of equation (24)—in the calculation of heads for time step $m$. Because they represent heads for the preceding time step, for which computations have
The Criteria for Terminating Iteration for Time Step, $m$ is

$$\left| \bar{h}_{m,n} - \bar{h}_{m,n-1} \right| \leq \text{Arbitrarily Established Closure Criterion}$$

Initial Trial Head Values for Time Step, $m+1$ (Set $\bar{h}_{m+1,0} = \bar{h}_m$)

Final Head Values for Time Step, $m$, After $n$ iterations

Interim Head Values for Time Step, $m$, after $(n-1)$ iterations

Interim Head Values for Time Step, $m$ After One Iteration

Figure 7.—Iterative calculation of a head distribution.
already been completed, they appear as predetermined constants in the equation for time step m; thus they retain the same value in each iteration of the time step. Similarly, the final values of head for time step \( m \) are used as constants in the storage term during calculations for time step \( m+1 \).

Ideally, one would like to specify that iteration stop when the calculated heads are suitably close to the exact solution. However, because the actual solution is unknown, an indirect method of specifying when to stop iterating must be used. The method most commonly employed is to specify that the changes in computed heads occurring from one iteration level to the next must be less than a certain quantity, termed the "closure criterion" or "convergence criterion," which is specified by the user. After each iteration, absolute values of computed head change in that iteration are examined for all nodes in the mesh. The largest of these absolute head change values is compared with the closure criterion. If this largest value exceeds the closure criterion, iteration continues; if it is less than the closure criterion, iteration is said to have "closed" or "converged," and the process is terminated for that time step. Normally, this method of determining when to stop iteration is adequate. Note that the closure criterion refers to change in computed head, and that values of head are not themselves necessarily calculated to a level of accuracy comparable to the closure criterion. As a rule of thumb, it is wise to use a value of closure criterion that is an order of magnitude smaller than the level of accuracy desired in the head results.

The program described herein also incorporates a maximum permissible number of iterations per time step. If closure is not achieved within this maximum number of iterations, the iterative process will be terminated and a
corresponding message printed in the output. The closure criterion is designated HCLOSE in the model input, while the maximum number of iterations per time step is designated MXITER.

The initial estimates of head for the end of time step \( m \), in array \( \bar{h}_{m,0}^{n} \) of figure 7, could be assigned arbitrarily, or they could be chosen according to a number of different conventions. Theoretically, the iterative process would eventually converge to the same result regardless of the choice of initial head values, although the work required would be much greater for some choices than for others. In the model described in this report, the heads computed for the end of each time step are used as the initial trial values of head for the end of the succeeding time step. Thus in figure 7, the array \( \bar{h}_{m-1,n}^{n} \) contains the final estimates of head, obtained after \( n \) iterations, for the end of time step \( m-1 \). When the calculations for step \( m-1 \) are complete, these same values of head are transferred to the array \( \bar{h}_{m,0}^{n} \), and used as the initial estimates, or trial values, for the heads at the end of time step \( m \). Head values for the end of the first time step in the simulation are assumed initially to be equal to the heads specified by the user for the beginning of the simulation.

Discussions of the mathematical basis of various iterative methods may be found in many standard references, including Peaceman (1977), Crichlow (1977) and Remson, Hornberger and Molz (1971). It is suggested that the reader review one of these discussions, both to clarify general concepts and to provide an introduction to such topics as the use of matrix notation, the role of iteration parameters, and the influence of various factors on rate of convergence. In particular, such a review is recommended prior to reading Chapters 12 and 13 of this report.
An iterative procedure yields only an approximation to the solution of the system of finite-difference equations for each time step; the accuracy of this approximation depends upon several factors, including the closure criterion which is employed. However, it is important to note that even if exact solutions to the set of finite-difference equations were obtained at each step, these exact solutions would themselves be only an approximation to the solution of the differential equation of flow (equation (1)). The discrepancy between the head, $h_{i,j,k}^m$, given by the solution to the system of difference equations for a given node and time, and the head $h(x_i,y_j,z_k,t_m)$ which would be given by the formal solution of the differential equation for the corresponding point and time, is termed the truncation error. In general, this error tends to become greater as the mesh spacing and time-step length are increased. Finally, it must be recognized that even if a formal solution of the differential equation could be obtained, it would normally be only an approximation to conditions in the field, in that hydraulic conductivity and specific storage are seldom known with accuracy, and uncertainties with regard to hydrologic boundaries are generally present.

**Formulation of Equations for Solution**

The model described in this report presently incorporates two different options for iterative solution of the set of finite-difference equations, and is organized so that alternative schemes of solution may be added without disruption of the program structure. Whatever scheme of solution is employed, it is convenient to rearrange equation (24) so that all terms containing heads at the end of the current time step are grouped on the left-hand side of the equation, and all terms that are independent of head at the end of the current time step are on the right-hand side. The resulting equation is
\[
CV_{i,j,k-1/2}^{m} + CC_{i-1/2,j,k}^{m} + CR_{i,j-1/2,k}^{m} + CR_{i,j+1/2,k}^{m} + (-CV_{i,j,k-1/2}^{m} - CC_{i-1/2,j,k}^{m} - CR_{i,j-1/2,k}^{m} - CR_{i,j+1/2,k}^{m} - CC_{i+1/2,j,k}^{m} - CV_{i,j,k+1/2}^{m})h_{i,j,k}^{m} + CR_{i,j+1/2,k}^{m}h_{i,j+1,k}^{m} + CC_{i+1/2,j,k}^{m}h_{i+1,j,k}^{m} + CV_{i,j,k+1/2}^{m}h_{i,j+1,k}^{m} = RHS_{i,j,k}^{m}
\]

where

\[
HCOF_{i,j,k} = P_{i,j,k} - SC_{i,j,k}^{1}/(t_{m} - t_{m-1});
\]

\[
RHS_{i,j,k} = - Q_{i,j,k} - SC_{i,j,k}^{1}/(t_{m} - t_{m-1});
\]

and

\[
SC_{i,j,k} = SS_{i,j,k}Ar_{j}Ac_{i}Av_{k}.
\]

The entire system of equations of the form of (26), which includes one equation for each variable-head cell in the mesh, may be written in matrix form as

\[
[A] \{h\} = \{q\}
\]

where \([A]\) is a matrix of the coefficients of head, from the left side of equation (26), for all active nodes in the mesh; \(\{h\}\) is a vector of head values at the end of time step \(m\) for all nodes in the mesh; and \(\{q\}\) is a vector of the constant terms, \(RHS\), for all nodes of the mesh. The model described in this report assembles the vector \(\{q\}\) and the terms that comprise \([A]\) through a series of subroutines, or "modules". The vector \(\{q\}\) and the terms comprising \([A]\) are then transferred to modules which actually solve the matrix equations for the vector \(\{h\}\).
Types of Model Cell and Simulation of Boundaries

In practice, it is generally unnecessary to formulate an equation of the form of (24) for every cell in a model mesh, as the status of certain cells is specified in advance in order to simulate the boundary conditions of the problem. In the model described in this report, cells of this type are grouped into two categories—"constant-head" cells and "inactive" (or "no-flow") cells. Constant-head cells are those for which the head is specified in advance, and is held at this specified value through all time steps of the simulation. Inactive or no-flow cells are those for which no flow into or out of the cell is permitted, in any time step of the simulation. The remaining cells of the mesh, termed "variable-head" cells in this report, are characterized by heads which are unspecified and free to vary with time. An equation of the form of (24) must be formulated for each variable-head cell in the mesh, and the resulting system of equations must be solved simultaneously for each time step in the simulation.

Constant-head and no flow cells are used in the model described herein to represent conditions along various hydrologic boundaries. For example, figure 8 shows the map of an aquifer boundary superimposed on an array of cells generated for the model. The aquifer is of irregular shape, whereas the model array is always rectangular in outline; no-flow cells have therefore been used to delete the portion of the array beyond the aquifer boundary. The figure also shows constant-head cells along one section of the boundary; these may be used, for example, where the aquifer is in direct contact with major surface water features. Other boundary conditions, such as areas of constant inflow or areas where inflow varies with head, can be simulated through the use of external source terms or through a combination of no-flow cells and external source terms.
Figure 8.—Discretized aquifer showing boundaries and cell designations.
Conceptual Aspects of Vertical Discretization

The model described in this document handles discretization of space in the horizontal direction by reading the number of rows, the number of columns and the width of each row and column (that is, the width of the cells in the direction transverse to the row or column). Discretization of space in the vertical direction is handled in the model by specifying the number of layers to be used, and by specifying hydraulic parameters which contain or embody the layer thickness. This approach is followed in preference to explicit reading of layer thickness in order to accommodate two different ways of viewing vertical discretization.

At one extreme, vertical discretization can be visualized simply as an extension of areal discretization—a more or less arbitrary process of dividing the flow system into segments along the vertical, governed in part by the vertical resolution desired in the results. At the opposite extreme, vertical discretization can be viewed as an effort to represent individual aquifers or permeable zones by individual layers of the model. Figure 9-a shows a typical geohydrologic sequence which has been discretized according to both interpretations—in 9-b according to the first viewpoint, and in 9-c according to the second. The first viewpoint leads to rigid superposition of an orthogonal three-dimensional mesh on the geohydrologic system; while there may be a general correspondence between geohydrologic layers and model layers, no attempt is made to make the mesh conform to stratigraphic irregularities. Under the second viewpoint, model layer thickness is considered variable, to simulate the varying thickness of geohydrologic units; this leads, in effect, to a deformed mesh.
Figure 9.—Schemes of vertical discretization.
Each of these methods of viewing the vertical discretization process has advantages, and each presents difficulties. The model equations are based on the assumption that hydraulic properties are uniform within individual cells, or at least that meaningful average or integrated parameters can be specified for each cell; these conditions are more likely to be met when model layers conform to geohydrologic units as in figure 9-c. Moreover, greater accuracy can be expected if model layers correspond to intervals within which vertical head loss is negligible, and this is also more likely under the configuration of 9-c. On the other hand, the deformed mesh of 9-c fails to conform to many of the assumptions upon which the model equations are based; for example, individual cells may no longer have rectangular faces, and the major axes of hydraulic conductivity may not be aligned with the model axis. Some error is always introduced by these departures from assumed conditions.

In practice many vertical discretization schemes turn out to be a combination of the viewpoints illustrated in figures 9-b and 9-c. For example, even where layer boundaries conform to geohydrologic contacts, it may be necessary to use more than one layer to simulate a single geohydrologic unit, simply to achieve the resolution required in the results. Figure 10 shows a system consisting of two sand units separated by a clay; the units are of uniform thickness, and each could be represented by a single layer without deformation of the mesh. However, flow is neither fully horizontal nor fully vertical in any of the layers; if information on the direction of flow within each unit is required, several layers must be used to represent each unit. Similarly, figure 11 shows a sand-clay system in which pumpage from the sands is sustained partially by vertical flow of water released from storage in the clay. If the objective of analysis is to determine the pattern of storage release in the clay, several model layers would be
Figure 10.—Possible pattern of flow in a cross section consisting of two high conductivity units separated by a low conductivity unit.
Figure 11.—A cross section in which a low conductivity unit is represented by six model layers.
required to represent that unit, as shown in the figure. On the other hand, figure 12 shows a sand-clay system in which storage release occurs only in the sands, flow in the sand is essentially horizontal, and flow in the clay is essentially vertical. In this case a single model layer may be used to represent each sand, while the clay may be represented simply by the vertical conductance between layers. This approach to vertical discretization has sometimes been termed the "quasi three-dimensional" approach.

The approaches to vertical discretization described above all lead to a set of equations of the form of (26), which must be solved simultaneously at each time step. The differences among these approaches arise in the way the various conductances and storage terms are formulated and, in general, in the number of equations to be solved, the resolution of the results, and the accuracy of the results. The model described in this document is capable of implementing any of these approaches to vertical discretization in that, as noted above, the thickness of individual layers (Δv_k of figure 1 and equation (24)) is never read explicitly by the program; rather, this thickness is embedded in various hydraulic coefficients specified by the user. For example, in confined layers transmissivity, which is the product of hydraulic conductivity and layer thickness, is specified; and storage coefficient, the product of specific storage and layer thickness, is also used. For an unconfined layer, aquifer bottom elevation and hydraulic conductivity are input for each cell. Saturated thickness is calculated as head minus bottom elevation, and transmissivity is then calculated as hydraulic conductivity times saturated thickness. Thus, layer thickness can vary from cell to cell depending on bottom elevation and head. Chapter 5, which describes the Block Centered Flow Package, contains a discussion of the formulation of conductance and storage terms corresponding to the various ways of conceptualizing the vertical discretization.
Heads in this layer are not calculated. Resistance to flow in this layer is included in the conductance terms between layers 1 & 2.

Figure 12.—A cross section in which a low conductivity unit is represented by the conductance between model layers.
CHAPTER 3
PROGRAM DESIGN

Overall Structure

This chapter describes the overall design of the model program. The program consists of a main program (MAIN) and a large number of highly independent subroutines called modules. This chapter will explain the functions of MAIN and explain how the modules can be grouped into "packages" and "procedures".

The functions which must be performed for a typical simulation are shown in figure 13. The period of simulation is divided into a series of "stress periods" within which specified stress parameters are constant. Each stress period, in turn, is divided into a series of time steps. The system of finite-difference equations of the form of equation (27) is formulated and solved to yield the head at each node at the end of each time step. Iterative solution methods are used to solve for the heads for each time step. Thus within a simulation, there are three nested loops: a stress-period loop, within which there is a time-step loop, which in turn contains an iteration loop.

Each rectangle in figure 13 is termed a "procedure". For example, prior to entering the stress loop, the program executes three procedures which pertain to the simulation as a whole. In the Define Procedure, the problem to be simulated is defined: the size of the model, the type of simulation (transient or steady-state), the number of stress periods, the hydrologic options, and the solution scheme to be used are specified. In the Allocate Procedure, memory space required by the program is allocated. In the Read and Prepare Procedure, all data that are not functions of time
DEFINE — Read data specifying number of rows, columns, layers, stress periods, and major program options.

ALLOCATE — Allocate space in the computer to store data.

READ AND PREPARE — Read data which is constant throughout the simulation. Prepare the data by performing whatever calculations can be made at this stage.

STRESS — Determine the length of a stress period and calculate terms to divide stress periods into time steps.

READ AND PREPARE — Read data which changes from one stress period to the next. Prepare the data by performing whatever calculations can be made at this stage.

ADVANCE — Calculate length of time step and set heads at beginning of a new time step equal to heads calculated for the end of the previous time step.

FORMULATE — Calculate the coefficients of the finite difference equations for each cell.

APPROXIMATE — Make one cut at approximating a solution to the system of finite difference equations.

OUTPUT CONTROL — Determine whether results should be written or saved on disk for this time step. Send signals to the BUDGET and OUTPUT procedures to indicate exactly what information should be put out.

BUDGET — Calculate terms for the overall volumetric budget and calculate and save cell-by-cell flow terms for each component of flow.

OUTPUT — Print and save heads, drawdown and overall volumetric budgets in accordance with signals from OUTPUT CONTROL procedure.

Figure 13.— Overall program structure.
are read. These data may include all or some of the following: boundary conditions, initial heads (starting heads), transmissivity, hydraulic conductivity, specific yield, storage coefficients, elevations of layer tops and bottoms, and parameters required by the specified solution scheme. Certain preliminary calculations are also made in this procedure to prepare data for further processing.

Within the stress period loop the first procedure is termed the Stress Procedure. In this procedure the number of time steps (NSTP) in the stress period and certain information to calculate the length of each time step are read. In a second Read and Prepare Procedure, all data that pertain to a stress period, such as pumping rates and areal recharge, are read and processed. The time-step loop is then entered (figure 13); in the Advance Procedure, the length of the time step is calculated and the heads for the start of the time step are initialized. The iteration loop contains the Formulate Procedure which determines the conductances and coefficients for each node as required by equation (27), and the Approximate Procedure which approximates a solution to the system of linear equations for head. Iteration proceeds until closure is achieved or until a specified maximum number of allowable iterations is reached. At the end of the iteration loop, the Output Control Procedure determines the disposition of the computed heads, budget terms, and cell-by-cell flow terms. In the Budget Procedure, budget entries are calculated and cell-by-cell flow terms are printed or recorded, as explained in a subsequent section. In the Output Procedure, heads, drawdown, and the volumetric budget are printed or recorded.
As shown in the preceding discussion, figure 13 provides a flow chart for the overall program structure, a list of the various procedures, and an indication of the sequence in which those procedures are implemented; it also provides a flow chart for the main program of the model. The work within the procedures—i.e., within the rectangles of figure 13—is performed by individual subroutines, or modules, called by the main program. The main program itself is simply an organized sequence of call statements, most of which are coupled to "IF" tests which determine whether a module is required. Accordingly, the main program does not itself do the work of simulation; it merely calls the various modules in the proper sequence to do that work. Modules which are called directly by the main program are termed "primary" modules; those that are called by other modules are termed "secondary" modules.

Thus the various procedures indicated in figure 13 are implemented through individual modules; and the modules can accordingly be grouped according to the procedure which they help to perform. As noted in Chapter 1, modules can also be grouped by "packages", where a package (for example, the River Package, the Well Package, or the SIP Package) includes those modules required to incorporate a particular hydrologic process or solution algorithm into the simulation. In terms of understanding the operation of the model, these two methods of grouping modules are both useful. The package classification, for example, indicates which modules will be active in a given simulation. (Modules are called by the main program only if they are part of a package which is required in the simulation; and while some packages are required in all simulations, most are needed only when the hydrologic process or solution method embodied in the package is specified by the user.) The procedure classification, on the other hand, defines the
specific function of the module in relation to the functions of other modules of the package. For example, several modules whose function is to allocate space are grouped under the Allocate Procedure; each of these modules allocates the space required for the arrays used in a single package. If few options or features are specified, relatively few packages are involved in the simulation, and the Allocate Procedure is handled by a relatively small number of modules. As the options specified by the user increase, more packages enter the simulation, and more modules are called to complete the space allocation task.

Figure 14 illustrates the classification of modules by procedure and by package in terms of a matrix of primary modules (i.e., modules called by the main program). The horizontal rows in figure 14 correspond to procedures, while the vertical columns correspond to packages. An "X" is entered in each block of the matrix for which a module exists; absence of an "X" indicates that the procedure in question is not required in the indicated package. Entries marked with a subscript "S" indicate primary modules which utilize submodules in accomplishing their function; submodules are secondary modules which are utilized only in a single package. Entries marked with the subscript "U" indicate primary modules which utilize utility modules to accomplish their tasks; utility modules are secondary modules which are available to many packages.
<table>
<thead>
<tr>
<th>Procedures</th>
<th>F</th>
<th>E</th>
<th>L</th>
<th>H</th>
<th>V</th>
<th>N</th>
<th>T</th>
<th>B</th>
<th>S</th>
<th>I</th>
<th>P</th>
<th>O</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define (DF)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allocate (AL)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read &amp; Prepare (RP)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stress (ST)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read &amp; Prepare (RP)</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Advance (AD)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Formulate (FM)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Approximate (AP)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Output Control (OC)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Output (OT)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 14.—Organization of modules by procedures and packages.
The primary modules are named according to a convention which indicates both the package and the procedure to which they belong. The first three characters designate the package, the fourth is a package version number, and the last two indicate the procedure. For example, in figure 14, a module is indicated for the Well Package and Allocate Procedure. This module is designated as WEL1AL; the first three letters, WEL, indicate that the module is part of the Well Package; the last two letters, AL, indicate that it performs the Allocate Procedure in that package. Thus this module is one of those that deals with the simulation of specified withdrawal or input, as through wells, and its particular function is to allocate the space in computer memory used to store well data. The number one appearing in the fourth place of the six-character module designation is a package version number. If the package is modified to effect improvements, a different integer would be used in this place to distinguish the modified package from the original or from other modified versions.

Figure 15 shows the names of the primary modules arranged in the same matrix format that was used in figure 14. As in figure 14, a subscript "S" indicates that submodules are utilized and "U" indicates that utility modules are utilized.

Submodules are designated by a six-character name in which the first character is always the letter "S". This is followed by three characters designating the package name, a numeral indicating the package version number, and a one-character mnemonic to distinguish the module from other submodules of the same package; for example, the secondary module "SBC1C" is a submodule in version one of the Block-Centered Flow Package. Utility
<table>
<thead>
<tr>
<th>Procedures</th>
<th>BAS</th>
<th>BCF</th>
<th>WEL</th>
<th>RCH</th>
<th>RIV</th>
<th>DRN</th>
<th>EVT</th>
<th>GHR</th>
<th>SLP</th>
<th>SOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define (DF)</td>
<td>BAS1UP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allocate (AL)</td>
<td>BAS1AL</td>
<td>BAS1AL</td>
<td>WEL1AL</td>
<td>RCH1AL</td>
<td>RIV1AL</td>
<td>DRN1AL</td>
<td>EVT1AL</td>
<td>GHB1AL</td>
<td>SIP1AL</td>
<td>SOR1AL</td>
</tr>
<tr>
<td>Read &amp; Prepare (RP)</td>
<td>BAS1RP</td>
<td>BAS1RP</td>
<td>WEL1RP</td>
<td>RCH1RP</td>
<td>RIV1RP</td>
<td>DRN1RP</td>
<td>EVT1RP</td>
<td>GHB1RP</td>
<td>SIP1RP</td>
<td>SOR1RP</td>
</tr>
<tr>
<td>Stress (ST)</td>
<td>BAS1ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read &amp; Prepare (RP)</td>
<td>BAS1FM</td>
<td>BAS1FM</td>
<td>WEL1FM</td>
<td>RCH1FM</td>
<td>RIV1FM</td>
<td>DRN1FM</td>
<td>EVT1FM</td>
<td>GHB1FM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Advance (AD)</td>
<td>BAS1AD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Formulate (FM)</td>
<td>BAS1FM</td>
<td>BAS1FM</td>
<td>WEL1FM</td>
<td>RCH1FM</td>
<td>RIV1FM</td>
<td>DRN1FM</td>
<td>EVT1FM</td>
<td>GHB1FM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Approximate (AP)</td>
<td></td>
<td>BAS1OC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Budget (BD)</td>
<td>BCF1RD</td>
<td>BCF1RD</td>
<td>WEL1RD</td>
<td>RCH1RD</td>
<td>RIV1RD</td>
<td>DRN1RD</td>
<td>EVT1RD</td>
<td>GHB1RD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output (OT)</td>
<td>BAS1OT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 15.—Primary modules organized by procedure and package.
modules are designated by the letter "U" followed by a five-character mnemonic. For example, the secondary module "U2DREL" is a utility module which reads two-dimensional real arrays.

Table 1 lists the various packages documented in this publication, gives the three-character abbreviation used in the module designation scheme, and provides a brief description of the package operation. Two major categories of package may be recognized—the flow component packages and the solver packages; within the category of flow component packages, a stress package subcategory may be recognized. The flow component packages are those which calculate the coefficients of the finite-difference equation for each cell. This category includes the Block-Centered Flow Package, which formulates the internal flow terms (describing flow between cells and flow to or from storage); and the subcategory of stress packages. Each of the stress packages formulates the coefficients describing a particular external or boundary flow; for example, the River Package calculates the coefficients describing flow between a cell and a surface stream. The solver packages are those which implement algorithms for solution of the systems of finite-difference equations. This documentation describes two packages in this category, one incorporating the Strongly Implicit Procedure of solution, and the other utilizing Slice-Successive Overrelaxation. The only package which does not fit into any of these categories is the Basic Package, which addresses a variety of tasks in support of the entire simulation.

The Block-Centered Flow Package is the only option described in this documentation for the formulation of internal flow terms in the equations. However, alternative packages, for example, utilizing a point centered approach, could certainly be developed and used in place of the Block-Centered Flow Package.
Table 1.—List of packages.

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Abbreviation</th>
<th>Package Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>BAS</td>
<td>Handles those tasks that are part of the model as a whole. Among those tasks are specification of boundaries, determination of time-step length, establishment of initial conditions, and printing of results.</td>
</tr>
<tr>
<td>Block-Centered Flow</td>
<td>BCF</td>
<td>Calculates terms of finite-difference equations which represent flow within porous medium; specifically, flow from cell to cell and flow into storage.</td>
</tr>
<tr>
<td>Well</td>
<td>WEL</td>
<td>Adds terms representing flow to wells to the finite-difference equations.</td>
</tr>
<tr>
<td>Recharge</td>
<td>RCH</td>
<td>Adds terms representing areally distributed recharge to the finite-difference equations.</td>
</tr>
<tr>
<td>River</td>
<td>RIV</td>
<td>Adds terms representing flow to rivers to the finite-difference equations.</td>
</tr>
<tr>
<td>Drain</td>
<td>DRN</td>
<td>Adds terms representing flow to drains to the finite-difference equations.</td>
</tr>
<tr>
<td>Evapotranspiration</td>
<td>EVT</td>
<td>Adds terms representing ET to the finite-difference equations.</td>
</tr>
<tr>
<td>General-Head</td>
<td>GHB</td>
<td>Adds terms representing general-head boundaries to the finite-difference equations.</td>
</tr>
<tr>
<td>Boundaries</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strongly Implicit</td>
<td>SIP</td>
<td>Iteratively solves the system of finite-difference equations using the Strongly Implicit Procedure.</td>
</tr>
<tr>
<td>Procedure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slice-Successive</td>
<td>SOR</td>
<td>Iteratively solves the system of finite-difference equations using Slice-Successive Overrelaxation.</td>
</tr>
<tr>
<td>Overrelaxation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Flow Component Packages

Stress Packages

Solver Packages
Similarly, additional solver packages, incorporating different solution algorithms, could be added, as could additional stress packages. Every simulation must include the Basic Package, the Block-Centered Flow Package (or a suitable replacement) and a solver package. Beyond this, the packages to be included in a simulation are at the option of the user, and will depend on the hydrologic processes influencing the problem. The individual modules in the program have been designed in such a way that the packages are totally independent; with the exception of the three required packages noted above, addition or removal of an individual package has no effect on other packages. If an entirely new package is desired, modules can be developed for each of the procedures involved (and the main program modified to call those modules in proper sequence) without affecting other packages of the program.

Figure 16 shows a detailed flow chart of the main program, indicating all of the primary modules together with the tests which determine whether or not each module is to be called. Figure 16 may be studied in conjunction with figures 13 and 15, and table 1, for an appreciation of the overall structure and operation of the model.

The overall design of the model is such that the conductance terms for cell-to-cell flow (CC, CR, and CV of equation (26)) are formulated at the beginning of the simulation, and are reformulated if necessary at each iteration during solution. Reformulation takes place only in unconfined situations, where the conductances depend upon saturated thickness, which may change at each iteration. At present, formulation of conductances is done only by the Block-Centered Flow Package, although again, a replacement package could readily be developed. The lateral conductance terms (CC and CR
Figure 16.—Overall program structure showing all primary modules.

3-12
in equation (26)) are computed as harmonic means for the intervals between nodes, using parameters specified by the user for individual cells; the vertical conductance (CV in equation (26)) is calculated using information which is specified directly for the vertical interval between nodes. The various conductance terms are stored in arrays which are ultimately passed to the solver package, where the matrix equations (equation (27)) are solved.

The coefficient $HCOFi_{i,j,k}$ and the term $RHS_{i,j,k}$ of equation (26) are formulated anew at each iteration, for all active nodes in the mesh. This formulation is done progressively, as each package calculates and adds terms for the particular process associated with that package. At the beginning of each iteration, the values of $HCOFi_{i,j,k}$ and $RHS_{i,j,k}$ are set to zero throughout the mesh. The Block-Centered Flow Package then adds the term 

$$-SS_{i,j,k}Ar_jAC_iAV_k/(t_m - t_{m-1})$$

at each node, and adds the term

$$-SS_{i,j,k}Ar_jAC_iAV_k\frac{h_m-1}{tm-tm-1}$$

at each node. For cells that are affected by flow from a stream, given by an expression of the form $P_{si,j,k}(h_S-h_{i,j,k})$, where $h_S$ is the (constant) stream head, the River package adds the term $-P_{si,j,k}$ to $HCOFi_{i,j,k}$, and adds the constant term $-P_{si,j,k}h_S$ to $RHS_{i,j,k}$. This process continues until each package specified by the user has added its contribution to $HCOF$ and $RHS$ at each indicated node of the mesh. The $HCOF$ and $RHS$ arrays are then transferred to the solver package, together with the three conductance arrays (CC, CR and CV), an array containing heads at the beginning of the time step, and the IBOUND array, which identifies constant head, no flow and active nodes. The solver package sums the six conductance terms and the value of $HCOF$ at each node to create a single coefficient of $h_{i,j,k}$.
corresponding to the term in brackets in equation (26)), and carries out one iteration of the solution procedure. The various arrays used in the solution procedure are actually stored as segments of a single one-dimensional array, the "X" array.

As noted in Chapter 1, Chapters 4 through 13 of this document discuss the program in terms of individual packages. Each of these chapters contains a detailed description of a particular package, including a listing and discussion of each module included in the package. The remainder of this chapter describes the way boundaries, water budget calculations, space allocation and input-output are handled in the model, and provides a brief description and listing of the main program.

Array Boundaries and Aquifer Boundaries

As noted in Chapter 2, the model may be visualized in terms of a three-dimensional assemblage of cells, each cell associated with a node of the model array. The size of the model array is specified by the user in terms of the number of rows (NROW), number of columns (NCOL) and number of layers (NLAY); these terms define a three-dimensional array of cells in the form of a rectangular box. In formulating the finite-difference equations, cell-to-cell conductance terms are omitted for the exterior of cells on the outer surface of this rectangular array. Thus considering flow along a row, a cell-to-cell conductance term is developed for the interval between column 1 and column 2, but not for the interval to the opposite side of column 1; similarly, a conductance term is developed for the interval between column (NCOL-1) and column (NCOL), but not for the interval beyond column (NCOL). Similar conventions are established in the other two directions, so that in
effect the array is bounded externally by planes across which no cell-to-cell flow occurs. If these boundaries of the model array, which are actually embedded in the program, coincide with impermeable boundaries in the aquifer, they can be relied upon to simulate the no-flow condition along those aquifer boundaries without further intervention by the user. In general, however, the aquifer boundaries will be irregular in form, or will not be of a simple impermeable character. In these cases, the aquifer boundary must be simulated by specifying certain cells within the array as no-flow or constant-head, by using external stress terms, or by using a combination of no-flow cells and external stress terms. This was discussed in Chapter 2, and is further discussed below. It should also be noted that while no cell-to-cell conductance terms are formulated for the interval above the uppermost layer of the model array, flow into this layer from above is frequently represented in the model through external stress terms—for example, terms representing evapotranspiration or stream seepage.

A finite-difference equation of the form of (26) is formulated for each variable-head cell in the mesh. For constant-head cells, no equation is formulated; however, the equation for each variable-head cell adjacent to a constant-head cell contains a term describing flow to and from the constant-head cell. For inactive no-flow cells, no equation is formulated, and no term appears in the equation of any adjacent cell for flow to or from the inactive cell; thus no flow is simulated across the interval between an inactive cell and any adjacent cell.

As pointed out above, the model array as initially generated always has the form of a rectangular box. Where the limits of an aquifer do not coincide with this rectangular shape, inactive cells may be used to delete portions of the array which fall outside the aquifer boundaries; this was
discussed through an example in Chapter 2. As noted in the same example, constant-head cells may be used to represent such features as surface water bodies of constant level which are in full contact with the aquifer. Boundaries which are characterized by a constant rate of flow into or out of the aquifer may be simulated using a no-flow boundary in conjunction with the Well Package, by assigning appropriate withdrawal or recharge rates to nodes just inside the boundary. Boundaries characterized by inflow which varies in proportion to head can be simulated using the General Head Boundary Package or the River Package, where these again are applied to nodes just interior to a no-flow boundary. Use of the River Package would involve specifying artificial streambed conductance and stream-head values at each cell along the boundary, where these values are deliberately chosen in such a way as to duplicate the required head-flow relationships.

Constant-head cells, inactive cells and variable-head cells are distinguished from one another in the model through the IBOUND array, which contains one element for each cell in the mesh. The entry in the IBOUND array for a given cell indicates the type of cell according to the following convention:

\[
\begin{align*}
\text{IBOUND} (I,J,K) < 0 & \quad \text{Cell } I,J,K \text{ is constant head} \\
\text{IBOUND} (I,J,K) = 0 & \quad \text{Cell } I,J,K \text{ is inactive} \\
\text{IBOUND} (I,J,K) > 0 & \quad \text{Cell } I,J,K \text{ is variable head}
\end{align*}
\]

The IBOUND codes are initially specified by the user. If necessary, the codes are adjusted so that they are consistent with other data specified by the user and with intermediate results. For example, cells which are specified as active but are given transmissivity and vertical-leakance values equal to zero are changed to inactive cells by the program.

**Volumetric Budget**

A summary of all inflows and outflows to a region is generally called a water budget. In this report, the water budget is termed a volumetric...
budget because it deals with volumes of water and volumetric flow rates; thus strictly speaking it is not a mass balance, although this term has been used in reference to volumetric budgets in other model reports. The model program calculates a water budget for the overall model as a check on the acceptability of the solution, and in order to provide summarized information on the flow system.

Numerical solution techniques for simultaneous equations do not always result in a correct answer; in particular, iterative solvers may stop iterating before a sufficiently close approximation to the solution is attained. A water budget provides an indication of the overall acceptability of the solution. The system of equations solved by the model actually consists of a flow continuity statement for each model cell. Continuity should also exist for the total flows into and out of the model—that is, the difference between total inflow and total outflow should equal the total change in storage. In the model program, the water budget is calculated independently of the equation solution process, and in this sense may provide independent evidence of a valid solution.

Each flow component package calculates its own contribution to the budget. The total budget as printed in the output does not include internal flows between model cells—only flows into or out of the model as a whole. For example, flow to or from rivers, flow to or from constant head cells, and flow to wells are all included in the overall budget terms. Flow into and out of storage is also considered part of the overall budget inasmuch as accumulation in storage effectively removes water from the flow system, and storage release effectively adds water to the flow—even though neither process, in itself, involves the transfer of water into or out of the ground water regime.
For every time step, the budget module of each flow component package calculates the rate of flow into and out of the system due to the process simulated by the package. The inflows and outflows for each component of flow are stored separately in the VBVL array. Most packages deal with only one such component of flow, but the Block-Centered Flow Package deals with two—flow to constant head cells and flow to storage. In addition to flow, the volumes of water entering and leaving the model during the time step are calculated as the product of flow rate and time step length. Cumulative volumes, from the beginning of the simulation, are then calculated and stored in array VBVL.

Module SBAS1V in the BAS Package uses the inflows, outflows and cumulative volumes in the VBVL array to print the budget at the times requested by the model user. When a budget is printed, the flow rates for the last time step and cumulative volumes from the beginning of simulation are printed for each component of flow. Inflows are printed separately from outflows; following the convention indicated above, water entering storage is treated as an outflow while water released from storage is treated as an inflow. In addition, total inflow and total outflow are printed, as well as the difference between total inflow and outflow. The difference is then printed as a percent error, calculated using the formula:

\[ D = \frac{100(IN-OUT)}{(IN+OUT)/2} \]

where \( IN \) is the total inflow to the system, \( OUT \) is the total outflow and \( D \) is the percent error term. If the model equations are correctly solved, the percent error should be small. In general, flow rates may be taken as an indication of solution validity for the time step to which they apply, while cumulative volumes are an indication of validity for the entire
simulation up to the time of the printout. The budget is printed at the end of each stress period whether requested or not.

There are situations in which it is useful to calculate flow terms for various subregions of the model. To facilitate such calculations, provision has been made to save flow terms for individual cells on disk so they can be used in computations external to the model itself. These individual cell flows are referred to here as "cell-by-cell" flow terms, and are of four general types: (1) cell-by-cell stress flows, or flows into or from an individual cell due to one of the external stresses represented in the model, such as evapotranspiration or recharge; (2) cell-by-cell storage terms, which give the rate of accumulation or depletion of storage in an individual cell; (3) cell-by-cell constant-head flow terms, which give the net flow to or from individual constant-head cells; and (4) internal cell-by-cell flows, which are actually the flows across individual cell faces—that is, between adjacent model cells. These four kinds of cell-by-cell term are further discussed in subsequent paragraphs. To save any of these cell-by-cell terms, two flags in the model input must be set. The input to the Output Control section of the Basic Package includes a flag, ICBCFL, which must be set for each time step for which any cell-by-cell terms are to be saved. In addition, each flow component package includes a flag which is set if the cell-by-cell terms computed by that package are to be saved. Thus if the appropriate flag in the Evapotranspiration Package input is set, cell-by-cell evapotranspiration terms will be saved for each time step for which the ICBCFL flag in the Basic Package input is also set. Three of the four types of cell-by-cell flow terms listed above—storage, constant-head cell and internal flows—are computed in the Block-Centered Flow Package, and thus fall under the control of a single flag, IBCFCB, in the input to that
package. Thus in general all three types are saved on disk if this flag is set, and ICBCFL is also set for the time step. Only flow values are saved in the cell-by-cell disk files; neither water volumes nor cumulative water volumes are included. The flow dimensions are volume per unit time, where volume and time are in the same units used for all model input data. The cell-by-cell flow values are stored in unformatted form to make the most efficient use of disk space; see the narrative for the UBUDSV module for information on how the data are written to disk.

Cell-by-cell stress flows are flow rates into or out of the model, at a particular cell, due to one particular external stress. For example, the cell-by-cell evapotranspiration term for cell i,j,k would give the flow out of the model by evapotranspiration from cell i,j,k. Cell-by-cell stress flows are considered positive if flow is into the cell, and negative if it is out of the cell. A cell-by-cell stress flow value is saved for every model cell, for each stress component for which the cell-by-cell flow is requested. That is, an array the size of the model grid is saved on disk for each requested component of flow. For many of the stress components, flow will be zero at most model cells. For example, when using the River Package, there will be nonzero cell-by-cell budget values only at those cells that are traversed by rivers. Thus the amount of disk space required for cell-by-cell flow terms can be large; a flow value is stored for each model cell even when that value is zero, and terms may be saved at many time steps.

The cell-by-cell storage term gives the net flow to or from storage in a variable-head cell. An array of these terms, one for each cell in the mesh is saved in transient simulations if the appropriate flags are
set. Withdrawal from storage in the cell is considered positive, whereas accumulation in storage is considered negative.

The cell-by-cell constant-head flow term gives the flow into or out of an individual constant-head cell. This term is always associated with the constant-head cell itself, rather than with the surrounding cells which contribute or receive the flow. A constant-head cell may be surrounded by as many as six adjacent variable-head cells. The cell-by-cell calculation provides a single flow value for each constant-head cell, representing the algebraic sum of the flows between that cell and all of the adjacent variable-head cells. A positive value indicates that the net flow is away from the constant-head cell (into the variable-head portion of the mesh); a negative value indicates that the net flow is into the constant-head cell.

The internal cell-by-cell flow values represent flows across the individual faces of a model cell. Three such terms are saved by the Block-Centered Flow Package for each variable-head cell and constant-head cell in the mesh, whenever the appropriate cell-by-cell flags are set. These three terms are flow across the front cell face (between cell \(i,j,k\) and \(i+1,j,k\)), flow across the right face (between cell \(i,j,k\) and \(i,j+1,k\)), and flow across the lower face (between cell \(i,j,k\) and \(i,j,k+1\)). Each of these represents flow between a given cell and a neighboring cell. (Although each cell has six neighbors, only three flow terms are required; flow across the other three sides is accounted for in the calculations of flow for cells adjacent to those sides.) Flows are considered positive if they are in the direction of increasing row number, increasing column number or increasing layer number, and are considered negative if in the opposite directions. These internal cell-by-cell flow values are useful
in calculations of the ground-water flow into various subregions of the model, or in constructing flow vectors.

In theory one could calculate a budget identical to the overall budget by using the cell-by-cell flow terms. This is not always true in practice because in some situations the budgets may be summed differently. The cell-by-cell value at a cell for a given stress or flow component is the net flow for that component, which could possibly include two or more flows of the same type, some negative and some positive. Only the net flow for the cell is saved in the cell-by-cell disk file. In the overall budget calculations as performed in the model, on the other hand, positive and negative flows are assembled separately, so that a negative flow at an individual cell would be added to the outflow term and a positive flow at the same cell would be added to the inflow term. Thus if inflow and outflow terms for the entire model are calculated by summing individual cell-by-cell values, they may differ from the corresponding terms as calculated by the model program in the overall budget. However, the difference between inflow and outflow should be the same for either calculation.

**Space Allocation**

Space in the central memory of the computer used by data arrays and lists is allocated at execution time in a one-dimensional array called the "X" array. The Allocate Procedure contains a module for each package of the model which allocates space needed by that package. The total number of words needed in the X array depends on the type and number of packages required in a simulation and generally will range from 10 to 20 times the number of cells in the grid. The main program contains two statements referring to the length of the X array, both of which appear in the first
part of the program listing. In the listing reproduced with this documentation these statements are COMMON X(30000) and LENX = 30000. The number 30000 in the statements refers to the length of the X array; this number must be increased if the storage requirements of the problem exceed 30000 elements.

Three-Dimensional Subscripts for Model Arrays

The conceptualization and implementation sections of this report designate cell locations by row, column, and layer indices in that order (usually designated as i,j,k), as is customary in scientific literature; however, this order of indices is not the most efficient order for array subscripts in the model program. Many model parameters are declared to be three dimensional arrays and accordingly have row, column and layer subscripts. The order of array subscripts in the FORTRAN language determines how data are stored in computer memory. The design of the program is such that the model array subscripts should be in column, row, and layer order for the most efficient memory access on most computers; this order has been used throughout the program. Typically in the program, J is used for the column subscript, I is used for row, and K for layer, but the order is J,I,K rather than I,J,K. It is important to bear in mind this difference in the subscript ordering when comparing the model program to the conceptualization and implementation sections of the report.
Input Structure

The input structure of the program is designed to permit input to be gathered, as it is needed, from many different files. It is based on an element of the FORTRAN language called the unit number, which identifies the file from which the input is to be read (or to which the output is to be written). The user must provide a link between the name of each input or output file and the corresponding unit number; this is generally done externally to the program, through operating system statements.

For input purposes, the program may be discussed in terms of "major options"; these are major segments of the program which are utilized only at the user's request. They correspond generally to the individual packages; in fact, all of the existing packages except the Basic Package are considered major options. Output Control, which is not an individual package but rather an optional segment of the Basic Package providing flexibility in program output, is also considered a major option. The balance of the Basic Package is not considered an option since it is always utilized and input for it must always be read. Block-Centered Flow has been treated here as an option, even though it is presently required in all simulations. This has been done to allow for the addition of replacement packages for Block-Centered Flow in the future.
One of the first steps in organizing input data is to specify which of the major options are to be used. This is done using the "IUNIT" array (figure 17) which is read in the Define Procedure by the Basic Package. An option is invoked by inserting an input unit number in the appropriate element of the IUNIT array; if an option is not desired, the value of the element is set to zero. Thus the IUNIT array serves as a flag to indicate whether an option is active, and also serves to specify the unit number containing input data required by the option. For example, if the Drain Package is not to be used, the third element of the IUNIT array (figure 18) is set to zero; if it is to be used, the third element of the array is set to the unit number of the file containing the input data for the package. In the main program, the value of IUNIT (3) is tested in several of the program procedures. If it is zero, the Drain module associated with the procedure is not called. If IUNIT (3) is greater than zero, the subroutine is called and input data is read from the file associated with the unit number.

As noted above, the Basic (BAS) Package, exclusive of the Output Control option, is used for every simulation; and input data for the Basic Package are always required. Basic Package data (figure 18) are read from unit number 1 as specified in the main program. If necessary, the unit number for BAS input can be changed to meet the requirements of a particular computer.

The first element of the IUNIT array must contain the unit number from which data for the Block-Centered Flow (BCF) Package are to be read. At present, because BCF is the only package available for the formulation of cell-to-cell flow terms, a non-zero entry in the first element of IUNIT is always required.
Assignment of Major Options to Elements in the IUNIT Array

Sample IUNIT Input Record

<table>
<thead>
<tr>
<th>Element Number</th>
<th>BCF</th>
<th>WEL</th>
<th>DRN</th>
<th>RIV</th>
<th>EVT</th>
<th>GHB</th>
<th>RCH</th>
<th>SIP</th>
<th>SOR</th>
<th>Output Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 17.—Specification of major options using the IUNIT array.
Options Are Specified in IUNIT. The Locations in IUNIT Are Assigned as Follows:

1. BCF package
2. WEL package
3. DRN package
4. RIV package
5. EVT package
6. GHB package
7. RCH package
8. SIP package
9. SOR package
10. Output control

This Is BAS Input Read from Unit 1

This Is the First Record on Which User Puts Heading for the Problem
This Is the Second Card of the Heading

These are IBOUND Array Values for Layer Two Read on Unit 24

This Is BCF Input Read from Unit 17.

This Is WEL Input Read on Unit 23.

This Is RCH Input Read on Unit 57.

This Is SIP Input Read on Unit 77.

Figure 18.—Sample input data showing role of the IUNIT array.
Most of the data submitted by the user will consist of one-dimensional and two-dimensional arrays. Those arrays are submitted as an "array control record" plus, optionally, a series of records containing the array elements. The array control record is read from the unit number specified for the major option which calls for the array. If all the elements of an array have the same value, the value is specified on the control record and it is not necessary to read the associated array. If the elements of the array vary, records containing the array values are read from the unit specified on the array control record according to a format which is also specified in the control record. The unit number may be the same as that from which the control record is read, or it may be different. Thus there is a great deal of flexibility regarding the organization of the input data for a simulation.

Any consistent length and time units may be used for model data. This gives a certain amount of freedom to the user, but care must be exercised to avoid any mixing of units. There is no way for the program to detect the use of inconsistent units. For example, if transmissivity is entered in units of ft²/day and pumpage as m³/s, the program will run, but the results will be meaningless.

Output Structure

The output structure is designed to control the amount, type, and frequency of information to be printed or written on disk. It controls the printing of head and drawdown by layer and time step, and the printing of the overall volumetric budget. It also controls disk output of head,
drawdown, and cell-by-cell flow terms for use in calculations external to
the model, or in user-supplied printing and plotting programs.

Output Control, which is a major option contained within the Basic
Package, receives instructions from the user to control the amount and
frequency of output. To utilize this option, the user must specify the
unit number of the file or channel from which the input data for the Output
Control option are to be read. This unit number must be entered as the
twelfth element of the IUNIT array (IUNIT 12); the input information is
then read, at each time step, from the file identified by this unit number.
If a zero is specified as the twelfth element of the IUNIT array, a default
output convention is invoked. This default output consists of head values
and budget terms printed for the end of each stress period. Every simulation
generates some printer output. All printer output goes to unit number 6 as
specified in the main program. This unit number can be changed to meet the
requirements of a particular computer.

The Main Program

The main program serves two major purposes: (1) it controls the order
in which the primary modules are executed, and (2) it serves as a switching
system for information. It does so with CALL statements which specify, by
name, a module to be executed and lists the names of data fields (subroutine
arguments) which are accessible by both the main program and the module.
The arrangement of CALL statements in the program reflects the order of procedures shown in the system flow chart (figure 13). Within a procedure, the calls to specific modules can be in any order with one exception: if a procedure has a CALL to a module in the Basic Package, that CALL must precede all other CALLS in that procedure. The main program calls modules to perform the following tasks, in order (the numbers in the following list correspond to the numbers of the comments in the main program listing).

1. Set the length of the "X" array (LENX) in which all data arrays and lists are stored. Note: LENX should be set equal to the dimension of the X array prior to compilation.

2. Assign the input for the Basic Package to unit 1; assign printed output to unit 6.

3. Define the problem in terms of number of rows, columns, layers, stress periods, and major options to be used.

4. Allocate space in the X array for individual data arrays and lists.

5. If the X array is not big enough for the problem, STOP. (Redimension X and redefine LENX.)

6. Read and prepare information which is constant throughout the simulation.

7. For each stress period:
   
   (a) Read stress-period timing information.

   (b) Read and prepare information that changes each stress period.

   (c) For each time step:
(1) Calculate the current time-step length and move "new" heads from the preceding time step to the array containing "old" heads of the current time step.

(2) Iteratively formulate and solve the system of equations:
   a. Formulate the finite-difference equations.
   b. Calculate an approximate solution to the system of equations.
   c. If convergence criterion has been met, stop iterating.

(3) Determine the type and amount of output needed for this time step.

(4) Calculate overall budget terms and, if specified, calculate and print or record cell-by-cell flow terms.

(5) Print and/or record heads and/or drawdown. Print the overall volumetric budget and timing summary.

(6) If iteration fails to meet convergence criterion, STOP.

8. END PROGRAM.
**MAIN CODE FOR MODULAR MODEL -- 9/1/87**

**BY MICHAEL G. MCDONALD AND ARLEN W. HARBAUGH**

**----VERSION 1638 24JUL1987 MAIN1**

**---------------------------------------------**

**SPECIFICATIONS:**

```
COMMON X(30000)
COMMON /FLWCON/LAYCON(80)
CHARACTER*4 HEADNGsVBNM
DIMENSION HEADNG(32),VBNM(4,20),VBYL(4,20),IUNIT(24)
DOUBLE PRECISION DUMMY
EQUIVALENCE (DUMMY,X(1))
```

**C1-------SET SIZE OF X ARRAY. REMEMBER TO REDIMENSION X.**
LENX=30000

**C2-------ASSIGN BASIC INPUT UNIT AND PRINTER UNIT.**
INBAS=1
IOUT=6

**C3-------DEFINE PROBLEM ROWS, COLUMNS, LAYERS, STRESS PERIODS, PACKAGES**
CALL BASIDP(ISUM,HEADNG,NPER,ITMUNI,TOTIM,NCOL,NROW,NLAY,1
  NONFS,TRNAS,IOUT,IUNIT)

**C4-------ALLOCATE SPACE IN "X" ARRAY.**
CALL BASIAL(ISUM,LENX,LCBOU,LCHELDR,LCDELN,LCSTRT,LCIOFL,
  1 LCCHOF,LCRHs,LCDLR,LCDELn,LCSTR,LCBUFF,LCIOFL,
  2 INBS,ISTR,NCOL,NROW,NLAY,IOUT)
IF(IUNIT(1).GT.0) CALL BCFIAL(ISUM,LENX,LCSCL,LOHY,
  1 LCBO,LCTOP,LSCSC,LCTRPY,IUNIT(1),1SS,
  2 NCOL,NROW,NLAY,IOUT,IBCFCB)
IF(IUNIT(2).GT.0) CALL WEILAI(ISUM,LENX,LCWELL,MWELL,NWELLS,
  1 1UNIT(2),IOUT,IEWLCB)
IF(IUNIT(3).GT.0) CALL DRNIAL(ISUM,LENX,LCDRAI,NDRAIN,MDRN,
  1 1UNIT(3),IOUT,1DRNCG)
IF(IUNIT(8).GT.0) CALL RCHIAL(ISUM,LENX,LCIRCH,LCRECH,NCIRCH,
  1 NCOL,NROW,IUNIT(8),IOUT,IRCHCB)
IF(IUNIT(9).GT.0) CALL EVTIAL(ISUM,LENX,LCEVT,LCV,LCEXP,
  1 LCSSURF,NCOL,NROW,NEVTOP,IUNIT(9),IOUT,IEVTCB)
IF(IUNIT(4).GT.0) CALL RIVIAL(ISUM,LENX,LCRIVR,MXRIVR,MRVER,
  1 1UNIT(4),IOUT,IRIVCB)
IF(IUNIT(7).GT.0) CALL GHBIAL(ISUM,LENX,LCBND,NBOND,MBND,
  1 1UNIT(7),IOUT,IGHBCB)
IF(IUNIT(9).GT.0) CALL SIPAL(ISUM,LENX,LCAL,LCALG,LCALR,LCSURF,
  1 LCHDG,LCRCH,LCRCH,LPARM,NCOL,NROW,NLAY,1UNIT(9),IOUT,
  2 LCIEG,蜡XI,NCOL,NLAY,NLC,MBW,1UNIT(11),IOUT)

**C5-------IF THE "X" ARRAY IS NOT BIG ENOUGH THEN STOP.**
IF(ISUM-1.GT.LENX) STOP

**C6-------READ AND PREPARE INFORMATION FOR ENTIRE SIMULATION.**
CALL BASIRP(X(LCBOU),X(LCNEW),X(LCSTR),X(LCHOLD),1
  1 ISTR,INBAS,HEADNG,NCOL,NROW,NLAY,NODES,VBVL,X(LCIOFL),
  2 IUNIT(12),IHEDCF,1DDCNP,1HEDUN,IHDUN,IOUT)
IF(IUNIT(1).GT.0) CALL BCIRP(X(LCBOU),X(LCHNEW),X(LCSTR),X(LCHOLD),
  1 X(LOHY),X(LCCR),X(LCCS),X(CLCY),X(LCDLR),
  2 X(LCDLR),X(LCTOP),X(LSCSC),X(LCTRPY),
  3 IUNIT(1),ISS,NCOL,NROW,NLAY,NODES,IOUT)
IF(IUNIT(9).GT.0) CALL SIPRP(NPARM,MXITER,ACCL,HCLOSE,X(LCW),
1     IUNIT(9),IPCALC,IPRSIP,IOUT)
IF(IUNIT(11).GT.0) CALL SORIP(MXITER,ACCL,HCLOSE,IUNIT(11),
1     IPRSOR,IOUT)
C
C7------SIMULATE EACH STRESS PERIOD.
DO 300 KPER=1,NPER
   KKPER=KPER
C
C7A------READ STRESS PERIOD TIMING INFORMATION.
   CALL BASIST(NSTP,DELT,TSMULT,PERTIM,KPER,INBAS,IOUT)
C
C7B------READ AND PREPARE INFORMATION FOR STRESS PERIOD.
   IF(IUNIT(2).GT.0) CALL WELLRP(X(LCWELL),NWELLS,MXWELL,IUNIT(2),
1     IOUT)
   IF(IUNIT(3).GT.0) CALL DRNPX(X(LCDRAI),NDRAIN,MXDRN,IUNIT(3),
1     IOUT)
   IF(IUNIT(8).GT.0) CALL RVIPRP(NRCHOP,X(LCIRCH),X(LCRECH),
1     X(LCDELR),X(LCDELC),NRCW,NCQL,IUNIT(8),IOUT)
   IF(IUNIT(5).GT.0) CALL ETVIRP(NEVTOP,X(LCEVT),X(LCEVTR),
1     X(LCEXDP),X(LCSURF),X(LCDEL),X(LCDELC),NCOL,NROW,
1     IUNIT(5),IOUT)
   IF(IUNIT(4).GT.0) CALL GHBRP(NBOUND,MXBOUND,IUNIT(7),
1     IOUT)
C
C7C------SIMULATE EACH TIME STEP.
   DO 200 KSTP=1,NSTP
      KKSTP=KSTP
C
C7C1------CALCULATE TIME STEP LENGTH. SET HOLD=NEW.
   CALL BASIAD(DELT,TSMULT,TOTIM,PERTIM,X(LCHNEW),X(LCHOLD),KKSTP,
1     NCOL,NROW,NLAY)
C
C7C2------ITERATIVELY FORMULATE AND SOLVE THE EQUATIONS.
   DO 100 KITER=1,MXITER
      KKITER=KITER
C
C7C2A------FORMULATE THE FINITE DIFFERENCE EQUATIONS.
   CALL BASIFM(X(LCHCOF),X(LCHNEW),NODES)
   IF(IUNIT(1).GT.0) CALL BCIFM(X(LCHCOF),X(LCHNEW),X(LCHOLD),
1     X(LCSCI),X(LCHNEW),X(LCIBoU),X(LCCR),X(LCCO),X(LCCV),
2     X(LCHY),X(LCTR),X(LCHBOT),X(LCHTOP),X(LCCS2),
3     X(LCDEL),X(LCDEL),DEL,T,ISS,KITER,KSPT,KKPER,NCOL,
4     NROW,NLAY,IOUT)
   IF(IUNIT(2).GT.0) CALL WELIFM(NWELLS,MXWELL,X(LCWELL),X(LCWELL),
1     X(LCHIBO),NCOL,NROW,NLAY)
   IF(IUNIT(3).GT.0) CALL DRNIFM(NDRAIN,MXDRN,X(LCDRAI),X(LCRECH),
1     X(LCHOF),X(LCRHS),X(LCIBoU),NCOL,NROW,NLAY)
   IF(IUNIT(8).GT.0) CALL RVIFM(NRCHOP,X(LCIRCH),X(LCRECH),
1     X(LCHNEW),X(LCIBoU),NCOL,NROW,NLAY)
   IF(IUNIT(5).GT.0) CALL ETVIFM(NEVTOP,X(LCEVT),X(LCEVTR),
1     X(LCEXDP),X(LCSURF),X(LCHOF),X(LCIBO),X(LCCO).
1     X(LCHNEW),NCOL,NROW,NLAY)
   IF(IUNIT(4).GT.0) CALL GHIFM(NBOUND,MXBOUND,X(LCBND),X(LCHNEW),
1     X(LCHOF),X(LCRHS),X(LCIBoU),NCOL,NROW,NLAY)
   IF(IUNIT(7).GT.0) CALL GHIJIFM(NBOUND,MXBOUND,X(LCBND),X(LCHOF),
1     X(LCRHS),X(LCIBoU),NCOL,NROW,NLAY)
C
C7C2B---MAKE ONE CUT AT AN APPROXIMATE SOLUTION.
    IF(IUNIT(9).GT.0) CALL SIPIAP(X(LCHNEW),X(LCIBOU),X(LCCR),X(LCOC),
1     X(LCCV),X(LCHCOF),X(LCRHS),X(LCGL),X(LCFL),X(LCC),X(LCH)),
2     X(LCM),X(LCHDGO),X(LLACH),NPARM,KKITER,HCLOSE,ACCL,ICNVG,
3     KKSTP,KKPER,IPCALC,IPRSIP,MXITER,NSTP,NCOL,NROW,NLAY,NODES,
4     IOUT)
    IF(IUNIT(11).GT.0) CALL SORLAP(X(LCHNEW),X(LCIBOU),X(LCCR),
1     X(LCCCI),X(LCCCV),X(LCHCOF),X(LCRHS),X(LCA),X(LCRE),X(LCEIOP),
2     X(LCHDGO),X(LLARCH),KKITER,HCLOSE,ACCL,ICNVG,KKSTP,KKPER,
3     IPRSOR,MXITER,NSTP,NCOL,NROW,NLAY,SLICE,NMEW,IOUT)
C
C7C2C---IF CONVERGENCE CRITERION HAS BEEN MET STOP ITERATING.
    IF(ICNVG.EQ.1) GO TO 110
100 CONTINUE
    KITER=MXITER
110 CONTINUE
C
C7C3----Determine which output is needed.
    CALL BASIOC(NSTP,KKSTP,ICNVG,X(LCIOFL),NLAY,
1     1 IBUDFL,ICBCFL,IHDDFL,IUNIT(12),IOUT)
C
C7C4----Calculate budget terms. Save cell-by-cell flow terms.
    MSUM=1
    IF(IUNIT(1).GT.0) CALL BCF1BD(VBNM,VBVL,MSUM,X(LCHNEW),
1     1 X(LCIBOU),X(LCHOLD),X(LCSCI),X(LCCR),X(LCC),X(LCCCV),
2     X(LCTOP),X(LCCCSI),DELT,ISS,NCOL,NROW,NLAY,KKSTP,KKPER,
3     1 ICBCFL,X(LCBUFF),IOUT)
    IF(IUNIT(2).GT.0) CALL WEL1BD(NWELLS,MXWELL,VBNM,VBVL,MSUM,
1     1 X(LCWELL),X(LCWEL),DEL T,ISS,NCOL,NROW,NLAY,KKSTP,KKPER,
2     1 ICBCFL,X(LCBUFF),IOUT)
    IF(IUNIT(3).GT.0) CALL DRN1BD(NDRAIN,MXDRAIN,VBNM,VBVL,MSUM,
1     1 X(LCDRAIN),DEL T,X(LCHNEW),NCOL,NROW,NLAY,X(LCIBOU),KKSTP,
2     1 KKPER,IRNHRD,ICBCFL,X(LCBUFF),IOUT)
    IF(IUNIT(4).GT.0) CALL RCH1BD(NRCHE,RLYRCH),X(LCRECH),X(LCHNEW),
1     1 X(LCIBOU),X(LCIBOU),NCOL,NROW,NCOL,DEL T,VBVL,VBNM,MSUM,KKSTP,KKPER,
2     1 ICBCFL,X(LCBUFF),IOUT)
    IF(IUNIT(5).GT.0) CALL EVT1BD(NEVTOP,X(LCEVT),X(LCEVTR),
1     1 X(LCEXTP),X(LCEXTR),X(LCHNEW),NCOL,NROW,NCOL,DEL T,VBVL,VBNM,MSUM,
2     1 DELT,VBVL,VBNM,MSUM,KKSTP,KKPER,ISTRT,ICBCFL,X(LCBUFF),IOUT)
    IF(IUNIT(6).GT.0) CALL RIV1BD(NRIVER,MRIVER,X(LCIRV),X(LCIBOU),
1     1 X(LCIBOU),NCOL,NROW,NCOL,NSVAL,VBVL,VBNM,MSUM,
2     1 DELT,VBVL,VBNM,MSUM,KKSTP,KKPER,IRVCH,ICBCFL,X(LCBUFF),IOUT)
    IF(IUNIT(7).GT.0) CALL GHB1BD(NBOUND,NMBND,VBNM,VBVL,MSUM,
1     1 X(LCBND),X(LCHNEW),NCOL,NROW,NLAY,X(LCIBOU),KKSTP,
2     1 KKPER,IGHBC,ICBCFL,X(LCBUFF),IOUT)
C
C7C5----Print and or save heads and drawdowns. Print overall budget.
    CALL BASIOT(X(LCHNEW),X(LCSTRP),X(LCIOFL),X(LCBUFF),X(LCIOF),
1     1 MSUM,X(LCIBOU),VBVM,VBVL,KKSTP,KKPER,DELT,
2     1 PERTIM,TOTIM,ITMUN,NCOL,NROW,NCOL,ICNVG,
3     1 IHDDFL,IBUDFL,IBUDPM,IBUDSP,IBUDPN,IBUDNM,IOUT)
C
C7C6----IF ITERATION FAILED TO CONVERGE THEN STOP.
    IF(ICNVG.EQ.0) STOP
200 CONTINUE
300 CONTINUE
C
C8-------END PROGRAM
STOP
C
END
Conceptualization and Implementation

The Basic Package handles a number of administrative tasks for the model. It reads data on the number of rows, columns, layers, and stress periods, on the major options to be used, and on the location of input data for those options. It allocates space in computer memory for model arrays; it reads data specifying initial and boundary conditions; it reads and implements data establishing the discretization of time; it sets up the starting head arrays for each time step; it calculates an overall water budget; and it controls model output according to user specification.

Selection of Major Options and Designation of Input Files

The selection of major options and the designation of their input unit numbers were discussed in the preceding chapter. The primary role of the Basic Package in these operations is to read the IUNIT array; as noted in Chapter 3, the entries in this array determine (a) whether or not a major option is to be used and (b) the unit number from which data for the option is to be read. Whenever a new major option is added to the program, an element corresponding to that option must be added to the IUNIT array.
Recall that the finite-difference equation for a cell has the form

\[
\begin{align*}
CR_{i,j-1/2,k} (h_{i,j-1,k}^m - h_{i,j,k}^m) + CR_{i,j+1/2,k} (h_{i,j+1,k}^m - h_{i,j,k}^m) \\
+ CC_{i-1/2,j,k} (h_{i-1,j,k}^m - h_{i,j,k}^m) + CC_{i+1/2,j,k} (h_{i+1,j,k}^m - h_{i,j,k}^m) \\
+ CV_{i,j,k-1/2} (h_{i,j,k-1}^m - h_{i,j,k}^m) + CV_{i,j,k+1/2} (h_{i,j,k+1}^m - h_{i,j,k}^m) \\
+ P_{i,j,k} h_{i,j,k}^m = SC_{i,j,k} (h_{i,j,k}^m - h_{i,j,k}^m)/\Delta t
\end{align*}
\]

One equation of this form is written for each variable-head cell in the grid. The IBOUND array, which is specified by the user and read by the Basic Package, contains a code for each cell which indicates whether (1) the head varies with time (variable-head cell), (2) the head is constant (constant-head cell), or (3) no flow takes place within the cell (no-flow or inactive cell). The IBOUND array can be modified by other packages if the state of a cell changes. Figure 19 illustrates the distribution of IBOUND code entries for a typical model layer.

**Initial Conditions**

Because equation (28) is in backward-difference form, a head distribution at the beginning of each time step is required to calculate the head distribution at the end of that time step (figure 20). For each time step after the first, the head distribution at the start of one time step is set equal to the head distribution at the end of the previous time step. For the first time step, "starting heads" are specified by the user. These specified initial heads are used for head calculation only in the first time step; however, they may also be saved, in the array STRT, and used to
Figure 19.—Example of the boundary array (IBOUND) for a single layer.

Area Where Heads Vary With Time

Aquifer Boundary

Area of Constant Head

| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |

IBOUND Codes

< 0 Constant Head
= 0 No Flow
> 0 Variable Head
Starting heads (STRT) are the heads at the beginning of the simulation.

New Heads (HNEW) are the latest estimate of the heads at the end of the current time step. Each iteration produces a new estimate.

Old Heads (HOLD) are the heads at the beginning of the current time step. They are, therefore, equal to the heads at the end of the previous time step.

Figure 20.—Flow of head distributions during a simulation.
calculate drawdown, the difference between the starting head distribution and some later head distribution.

**Discretization of Time**

Simulation time is divided into stress periods--time intervals during which all external stresses are constant--which are, in turn, divided into time steps as shown in figure 21. Within each stress period, the time steps form a geometric progression. The user specifies the length of the stress period, the number of time steps into which it is to be divided, and the time step multiplier, or ratio of the length of each time step to that of the preceding time step. Using these terms, the program calculates the length of each time step in the stress period.

**Output**

The primary output of the program is head distribution. The user may control the frequency at which heads are printed or saved on disk through the "Output Control" option, a major option contained in the Basic Package. Other output items include drawdowns and volumetric budget terms; the Output Control option also provides for storage or printing of these terms. If Output Control is not utilized, a default output option is invoked--the head distribution and the overall volumetric budget are printed at the end of each stress period, and drawdowns are also printed if starting heads were saved. Figure 22 shows an example of a volumetric budget printout for the end of a stress period.
**Figure 21.**——Division of simulation time into stress periods and time steps.
<table>
<thead>
<tr>
<th>CUMULATIVE VOLUMES</th>
<th>RATES FOR THIS TIME STEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>STORAGE = 0.0</td>
<td>STORAGE = 0.0</td>
</tr>
<tr>
<td>CONSTANT HEAD = 0.0</td>
<td>CONSTANT HEAD = 0.0</td>
</tr>
<tr>
<td>WELLS = 0.0</td>
<td>WELLS = 0.0</td>
</tr>
<tr>
<td>DRAINS = 0.0</td>
<td>DRAINS = 0.0</td>
</tr>
<tr>
<td>RECHARGE = 13608E+08</td>
<td>RECHARGE = 157.50</td>
</tr>
<tr>
<td>TOTAL IN = 13608E+08</td>
<td>TOTAL IN = 157.50</td>
</tr>
<tr>
<td>OUTPUT:</td>
<td>OUTPUT:</td>
</tr>
<tr>
<td>STORAGE = 0.0</td>
<td>STORAGE = 0.0</td>
</tr>
<tr>
<td>CONSTANT HEAD = 43265E+07</td>
<td>CONSTANT HEAD = 50.075</td>
</tr>
<tr>
<td>WELLS = 64800E+07</td>
<td>WELLS = 75.000</td>
</tr>
<tr>
<td>DRAINS = 28010E+07</td>
<td>DRAINS = 32.419</td>
</tr>
<tr>
<td>RECHARGE = 0.0</td>
<td>RECHARGE = 0.0</td>
</tr>
<tr>
<td>TOTAL OUT = 13607E+08</td>
<td>TOTAL OUT = 157.49</td>
</tr>
<tr>
<td>IN - OUT = 303.00</td>
<td>IN - OUT = 34943E-02</td>
</tr>
</tbody>
</table>

PERCENT DISCREPANCY = 0.00
PERCENT DISCREPANCY = 0.00

Figure 22.--Sample overall volumetric water budget.
The calculation of the volumetric budget is carried out in two parts, the calculation of budget entries and the summation of those entries. As explained in Chapter 3 the entries, which correspond to individual components of flow, are calculated in the flow component packages and stored in the one-dimensional array VBVL. The array VBVL is passed to the Basic Package which sums and prints the budget entries.
Input for the Basic (BAS) Package except for output control is read from unit 1 as specified in the main program. If necessary, the unit number for BAS input can be changed to meet the requirements of a particular computer. Input for the output control option is read from the unit number specified in IUNIT(12).

Information for the Basic Package must be submitted in the following order:

**FOR EACH SIMULATION**

### BASIDF

1. Data: HEADNG(32)
   Format: 20A4

2. Data: HEADNG (continued)
   Format: 12A4

3. Data: NLAY NROW NCOL NPER ITMUNI
   Format: I10 I10 I10 I10 I10

4. Data: IUNIT(24)
   Format: 2413
   (BCF WEL DRN RIV EVT XXX GHB RCH SIP XXX SOR OC)

### BASIAL

5. Data: IAPART ISTRT
   Format: I10 I10

### BASIRP

6. Data: IBOUND(NCOL,NROW)
   Module: U2DINT
   (One array for each layer in the grid)

7. Data: HNOFLO
   Format: F10.0

8. Data: Shead(NCOL,NROW)
   Module: U2DREL
   (One array for each layer in the grid)

**NOTE:** IBOUND and Shead are treated as three-dimensional arrays in the program. However, the input to each of these arrays is handled as a series of two-dimensional arrays, one for each layer in the grid.
FOR EACH STRESS PERIOD

BAS1ST

9. Data: PERLEN NSTP TSMULT
   Format: F10.0 I10 F10.0

Explanation of Fields Used in Input Instructions

HEADNG—is the simulation title that is printed on the printout. It may be up to 132 characters long; 80 in the first record and 52 in the second. Both records must be included even if they are blank.

NLAY—is the number of model layers.

NROW—is the number of model rows.

NCOL—is the number of model columns.

NPER—is the number of stress periods in the simulation.

ITMUNI—indicates the time unit of model data. (It is used only for printout of elapsed simulation time. It does not affect model calculations.)

0 - undefined  3 - hours
1 - seconds    4 - days
2 - minutes    5 - years

The unit of time must be consistent for all data values that involve time. For example, if years is the chosen time unit, stress-period length, time-step length, transmissivity, etc., must all be expressed using years for their time units. Likewise, the length unit must also be consistent.
IUNIT--is a 24-element table of input units for use by all major options. Only 10 elements (1-5, 7-9, 11, and 12) are being used. Element 6 has been reserved for a transient leakage package, while element 10 has been reserved for an additional solver, both on the assumption that such packages will be added to the model in the future. Elements 13-24 are reserved for future major options.

<table>
<thead>
<tr>
<th>IUNIT LOCATION</th>
<th>MAJOR OPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Block-Centered Flow Package</td>
</tr>
<tr>
<td>2</td>
<td>Well Package</td>
</tr>
<tr>
<td>3</td>
<td>Drain Package</td>
</tr>
<tr>
<td>4</td>
<td>River Package</td>
</tr>
<tr>
<td>5</td>
<td>Evapotranspiration Package</td>
</tr>
<tr>
<td>6</td>
<td>Reserved for Transient Leakage Package</td>
</tr>
<tr>
<td>7</td>
<td>General-Head Boundary Package</td>
</tr>
<tr>
<td>8</td>
<td>Recharge Package</td>
</tr>
<tr>
<td>9</td>
<td>SIP Package</td>
</tr>
<tr>
<td>10</td>
<td>Reserved for additional solver</td>
</tr>
<tr>
<td>11</td>
<td>SSOR Package</td>
</tr>
<tr>
<td>12</td>
<td>Output Control Option</td>
</tr>
</tbody>
</table>

If IUNIT(n) ≤ 0, the corresponding major option is not being used.

If IUNIT(n) > 0, the corresponding major option is being used and data for that option will be read from the unit number contained in IUNIT(n). The unit numbers in IUNIT should be integers from 1 to 99. Although the same number may be used for all or some of the major options, it is recommended that a different number be used for each major option. Printer output is assigned to unit 6 (unless it is changed to meet computer requirements). That unit number should not be used for any other input or output. The user is also permitted to assign unit numbers for output. Those numbers should be different from those assigned to input. The Basic Package reads from unit 1 (unless it is changed to meet computer requirements). It is permissible but unwise to use that unit for other major options.

IAPART--indicates whether array BUFF is separate from array RHS.

If IAPART = 0, the arrays BUFF and RHS occupy the same space. This option conserves space. This option should be used unless some other package explicitly says otherwise.

If IAPART ≠ 0, the arrays BUFF and RHS occupy different space. This option is not needed in the program as documented in this publication. It may be needed for packages yet to be written.
**ISTRT**—indicates whether starting heads are to be saved. If they are saved, they will be stored in array STRT. They must be saved if drawdown is calculated.

If **ISTRT** = 0, starting heads are not saved.

If **ISTRT** ≠ 0, starting heads are saved.

**IBOUND**—is the boundary array.

- If **IBOUND**(I,J,K) < 0, cell I,J,K has a constant head.
- If **IBOUND**(I,J,K) = 0, cell I,J,K is inactive (no-flow).
- If **IBOUND**(I,J,K) > 0, cell I,J,K is variable-head.

**HNOFLO**—is the value of head to be assigned to all inactive cells (**IBOUND** = 0) throughout the simulation. Since heads at inactive cells are unused, this does not affect model results but serves to identify inactive cells when head is printed. This value is also used as drawdown at inactive cells if the drawdown option is used. Even if the user does not anticipate having inactive cells, a value for **HNOFLO** must be submitted.

**Shead**—is head at the start of the simulation. Regardless of whether starting head is saved, these values must be input to initialize the solution.

**PERLEN**—is the length of a stress period. It is specified for each stress period.

**NSTP**—is the number of time steps in a stress period.

**TSMULT**—is the multiplier for the length of successive time steps. The length of the first time step **DELT**(1) is related to **PERLEN**, **NSTP** and **TSMULT** by the relation

\[ \text{DELT}(1) = \frac{\text{PERLEN}(1-\text{TSMULT})}{(1-\text{TSMULT}^\text{NSTP})}. \]
### DATA ITEM

<table>
<thead>
<tr>
<th>ITEM</th>
<th>EXPLANATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{HEADING}</td>
</tr>
<tr>
<td>2</td>
<td>{HEADING} CONTINUED</td>
</tr>
<tr>
<td>3</td>
<td>{NLAY, NR, W, NCOL, NPER, ITMUNI}</td>
</tr>
<tr>
<td>4</td>
<td>{UNIT, TAPE}</td>
</tr>
<tr>
<td>5</td>
<td>{RSTRT, RNR, RE}</td>
</tr>
<tr>
<td>6</td>
<td>CONTROL RECORD FOR IBOUND ARRAY LAYER 1</td>
</tr>
<tr>
<td>7</td>
<td>IBOUND VALUES FOR LAYER 1</td>
</tr>
<tr>
<td>8</td>
<td>CONTROL RECORD FOR IBOUND ARRAY LAYER 2</td>
</tr>
<tr>
<td>9</td>
<td>IBOUND VALUES FOR LAYER 2</td>
</tr>
<tr>
<td>10</td>
<td>CONTROL RECORD FOR IBOUND ARRAY LAYER 3</td>
</tr>
<tr>
<td>11</td>
<td>IBOUND VALUES FOR LAYER 3</td>
</tr>
<tr>
<td>12</td>
<td>CONTROL RECORD FOR STARTING HEAD ARRAY FOR LAYER 1</td>
</tr>
<tr>
<td>13</td>
<td>STARTING HEAD VALUES FOR LAYER 1</td>
</tr>
<tr>
<td>14</td>
<td>CONTROL RECORD FOR STARTING HEAD ARRAY FOR LAYER 2</td>
</tr>
<tr>
<td>15</td>
<td>STRESS PERIOD 1----- (PERLEN, NSTP, TSMULT)</td>
</tr>
<tr>
<td>16</td>
<td>STRESS PERIOD 2----- (PERLEN, NSTP, TSMULT)</td>
</tr>
</tbody>
</table>

### INPUT RECORDS

<table>
<thead>
<tr>
<th>ITEM</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3 10 12 2 1</td>
</tr>
<tr>
<td>2</td>
<td>-1 1 1 1 1 1</td>
</tr>
<tr>
<td>3</td>
<td>1 1 (1212)</td>
</tr>
<tr>
<td>4</td>
<td>-2 2 2 2 2 2</td>
</tr>
<tr>
<td>5</td>
<td>-2 2 2 2 2 2</td>
</tr>
<tr>
<td>6</td>
<td>1 1 (1213)</td>
</tr>
<tr>
<td>7</td>
<td>-2 2 2 2 2 2</td>
</tr>
<tr>
<td>8</td>
<td>1 1 1 1 1 1</td>
</tr>
<tr>
<td>9</td>
<td>-2 2 2 2 2 2</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>-2 2 2 2 2 2</td>
</tr>
<tr>
<td>12</td>
<td>999.99</td>
</tr>
<tr>
<td>13</td>
<td>1 1 1 (1256.2)</td>
</tr>
<tr>
<td>14</td>
<td>10. 17. 24 31. 38. 45. 52. 59. 66. 73. 80. 87.</td>
</tr>
<tr>
<td>15</td>
<td>11. 19. 27. 35. 43. 51. 59. 67. 75. 83. 91. 99.</td>
</tr>
<tr>
<td>16</td>
<td>13. 23. 33. 43. 53. 63. 73. 83. 93. 103. 113. 123.</td>
</tr>
<tr>
<td>17</td>
<td>16. 29. 42. 55. 68. 81. 94. 107. 120. 133. 146. 159.</td>
</tr>
<tr>
<td>18</td>
<td>20. 37. 54. 71. 88. 105. 122. 139. 156. 173. 190. 207.</td>
</tr>
<tr>
<td>19</td>
<td>26. 37. 54. 71. 88. 105. 122. 139. 156. 173. 190. 207.</td>
</tr>
<tr>
<td>20</td>
<td>16. 29. 42. 55. 68. 81. 94. 107. 120. 133. 146. 159.</td>
</tr>
<tr>
<td>21</td>
<td>13. 23. 33. 43. 53. 63. 73. 83. 93. 103. 113. 123.</td>
</tr>
<tr>
<td>22</td>
<td>11. 19. 27. 35. 43. 51. 59. 67. 75. 83. 91. 99.</td>
</tr>
<tr>
<td>23</td>
<td>10. 17. 24. 31. 38. 45. 52. 59. 66. 73. 80. 87.</td>
</tr>
</tbody>
</table>

### FIELDS IN ARRAY CONTROL RECORDS ARE

- LOCAT
- CONST
- FMTIN
- IPRN
Output Control is a major option separate from the rest of the Basic Package. Input to Output Control is read from the unit specified in IUNIT(12). If IUNIT(12) is zero, no output control data are read, and default output control is used. Under the default, head and total budget are printed at the end of every stress period. Additionally, if starting heads are saved (ISTRT is not 0), drawdown is printed at the end of every stress period. The default printout format for head and drawdown is 10G11.4. All printer output goes to unit 6 as specified in the main program. If necessary, the unit number for printer output can be changed to meet the requirements of a particular computer.

FOR EACH SIMULATION

BASIRP

1. Data: IHEDFM  IDDNFM  IHEDUN  IDDNUN
   Format: I10  I10  I10  I10

FOR EACH TIME STEP

BASIOC

2. Data: INCODE  IHDDFL  IBUDFL  ICBCFL
   Format: I10  I10  I10  I10

3. Data: Hdpr  Ddpr  Hdsv  Ddsv
   Format: I10  I10  I10  I10

(Record 3 is read 0, 1, or NLAY times, depending on the value of INCODE.)

Explanation of Fields Used in Input Instructions

IHEDFM--is a code for the format in which heads will be printed. Format codes have the same meaning for both head and drawdown. A positive format code indicates that each row of data is printed completely before starting the next row. This means that when there are more columns in a row than will fit on one line, additional lines are used as required to complete the row. This format is called the wrap format. A negative format code indicates that the printout is broken into strips where only that number of columns that will fit across one line are printed in a strip. As many strips are used as are required to print the entire model width. This format is called the strip format. The absolute value of the format code specifies the printout format as follows.

4-14
IHEDUN--is the unit number to which heads will be written if they are saved on disk.

IDDNUN--is the unit number to which drawdowns will be written if they are saved on disk.

INCODE--is the head/drawdown output code. It determines the number of records in input item 3.

If INCODE < 0, layer-by-layer specifications from the last time steps are used. Input item 3 is not read.

If INCODE = 0, all layers are treated the same way. Input item 3 will consist of one record.

If INCODE > 0, input item 3 will consist of one record for each layer.

IHDDFL--is a head and drawdown output flag.

If IHDDFL = 0, neither heads nor drawdowns will be printed or saved on disk.

If IHDDFL ≠ 0, heads and drawdowns will be printed or saved according to the flags for each layer specified in input item 3.

IBUDFL--is a budget print flag.

If IBUDFL = 0, overall volumetric budget will not be printed.

If IBUDFL ≠ 0, overall volumetric budget will be printed.

(Note that the overall volumetric budget will always be printed at the end of a stress period, even if the value of IBUDFL is zero.)

ICBCFL--is a cell-by-cell flow-term flag.

If ICBCFL = 0, cell-by-cell flow terms are not saved or printed.

If ICBCFL ≠ 0, cell-by-cell flow terms are printed or recorded on disk depending on flags set in the component of flow packages, i.e., IWELCB, IRCHCB, etc.
Hdpr--is the output flag for head printout.
   If Hdpr = 0, head is not printed for the corresponding layer.
   If Hdpr ≠ 0, head is printed for the corresponding layer.

Ddpr--is the output flag for drawdown printout.
   If Ddpr = 0, drawdown is not printed for the corresponding layer.
   If Ddpr ≠ 0, drawdown is printed for the corresponding layer.

Hdsv--is the output flag for head save.
   If Hdsv = 0, head is not saved for the corresponding layer.
   If Hdsv ≠ 0, head is saved for the corresponding layer.

Ddsv--is the output flag for drawdown save.
   If Ddsv = 0, drawdown is not saved for the corresponding layer.
   If Ddsv ≠ 0, drawdown is saved for the corresponding layer.
### SAMPLE INPUT TO THE OUTPUT CONTROL OPTION

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>4 6 75 77</td>
</tr>
<tr>
<td>1</td>
<td>[IH6DFM, IDDFFM, IHEDUN, IDENUM]</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 1--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>1 1 0 0</td>
</tr>
<tr>
<td>3</td>
<td>LAYER 1--[HDPR, DBPR, IDSV, DDSV]</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>LAYER 2--{HDPR, DBPR, IDSV, DDSV}</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>LAYER 3--{HDPR, DBPR, IDSV, DDSV}</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 2--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 0 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 3--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 1 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 4--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 0 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 5--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 1 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 6--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>0 1 1 0</td>
</tr>
<tr>
<td>2</td>
<td>ALL LAYERS--{HDPR, DDFF, HDSV, DDSV}</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 7--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 0 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 8--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 1 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 9--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 0 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 10--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 1 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 11--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 0 1 0</td>
</tr>
<tr>
<td>2</td>
<td>TIME STEP 12--{INCODE, IHDDFL, IMUDFL, ICBCFL}</td>
<td>-1 1 1 1</td>
</tr>
</tbody>
</table>
The Basic Package (BAS1) consists of eight primary modules and five submodules. The modules are:

### Primary Modules

- **BAS1DF**: Defines and sets key model parameters.
- **BAS1AL**: Allocates space for data arrays used by the Basic Package.
- **BAS1RP**: Reads and prepares data for the Basic Package.
- **BAS1ST**: Reads timing information and initializes variables needed to calculate the length of time steps.
- **BAS1AD**: Calculates the length of time steps, accumulates elapsed time, and initializes heads at the beginning of each time step.
- **BAS1FM**: Clears accumulators RHS and HCOF.
- **BAS1OC**: Sets flags which indicate when data should be printed or recorded on disk.
- **BAS1OT**: Prints and records heads, drawdowns, and overall volumetric budget.

### Submodules

- **SBAS1D**: Calculates, writes, and records drawdown distribution.
- **SBAS1H**: Writes and records head distribution.
- **SBAS1I**: Initializes the Output Control System.
- **SBAS1T**: Prints a time summary.
- **SBAS1V**: Calculates and prints the overall volumetric budget.
The BAS1DF module defines and sets key model parameters. It does so in the following order:

1. Print the name of the program.

2. Read and print a heading.

3. Read the number of layers, rows, columns, stress periods, and units of time code ITMUNI. ITMUNI is a code which indicates the time units of model data. It does not affect model calculations but is used when printing the amount of elapsed time (see the input instructions for the codes).

4. Print the number of layers, rows, columns, and stress periods.

5. Select and print a message showing the time units.

6. Read and print the input unit numbers IUNIT for all major options. IUNIT is a 24-element table. Each entry has been assigned to a particular major option. The user specifies that a certain major option is to be used by putting a positive integer into the IUNIT entry corresponding to that major option. The integer is the unit number from which input to the major option will be read. If a major option is not going to be used, the corresponding IUNIT element is set equal to zero.

7. Initialize the total-elapsed time counter (TOTIM) and the storage-array counter (ISUM) and calculate the total number of cells.

8. RETURN.
ITMUNI is a code which indicates units of time used in the input data. This code is only used to print a table showing elapsed time in seconds, minutes, hours, days, and years. It is not used in formulating or solving the finite-difference equation.

IUNIT is a table that indicates which major options are to be used and the unit numbers from which input is to be read.

TOTIM is an accumulator in which total simulation time is stored. It is incremented at each time step.

ISUM is a location counter for the first unallocated space in the X array. It is incremented by each module in the Allocate Procedure.
SUBROUTINE BASIDF(ISUMIHEADNG,NPERIITMUMI,TOTIM,NCOL,NROW,
1 NLAY,NODES,INBAS,IUNIT)
C
C----VERSION 1513 12MAY1987 BASIDF
C
C DEFINE KEY MODEL PARAMETERS
C
C SPECIFICATIONS:
C
CHARACTER*4 HEADNG
DIMENSION HEADING(32),IUNIT(24)
C
C1------PRINT THE NAME OF THE PROGRAM.
     WRITE(IOUT,1)  
1 FORMAT(1H1,20X,'U.S. GEOLOGICAL SURVEY MODULAR',  
1 ' FINITE-DIFFERENCE GROUND-WATER MODEL')
C
C2------READ AND PRINT A HEADING.
     READ(INBAS,2) HEADNG
     WRITE(IOUT,3) HEADNG
3 FORMAT(1H0,N4)
C
C3-------READ NUMBER OF LAYERS, ROWS, COLUMNS, STRESS PERIODS AND
C4------UNITS OF TIME CODE.
     READ(INBAS,4) NLAY,NROW,NCOL,NPER,ITMUNI
4 FORMAT(8I10)
C
C5-------PRINT # OF LAYERS, ROWS, COLUMNS AND STRESS PERIODS.
     WRITE(IOUT,5) NLAY,NROW,NCOL
5 FORMAT(1X,I4,' LAYERS',10,' ROWS',10,' COLUMNS')
     WRITE(IOUT,6) NPER
6 FORMAT(1X,I3,' STRESS PERIOD(S) IN SIMULATION')
C
C6------SELECT AND PRINT A MESSAGE SHOWING TIME UNITS.
     IF(ITMUNI.LE.0 .OR. ITMUNI.GT.5) ITMUNI=0
     GO TO (10,20,30,40,50) rITMUNI
     WRITE(IOUT,9)
9 FORMAT(1X,'MODEL TIME UNITS ARE UNDEFINED')
     GO TO 100
10 WRITE(IOUT,11)
11 FORMAT(1X,'MODEL TIME UNIT IS SECONDS')
     GO TO 100
20 WRITE(IOUT,21)
21 FORMAT(1X,'MODEL TIME UNIT IS MINUTES')
     GO TO 100
30 WRITE(IOUT,31)
31 FORMAT(1X,'MODEL TIME UNIT IS HOURS')
     GO TO 100
40 WRITE(IOUT,41)
41 FORMAT(1X,'MODEL TIME UNIT IS DAYS')
     GO TO 100
50 WRITE(IOUT,51)
51 FORMAT(1X,'MODEL TIME UNIT IS YEARS')
C
C7-------READ & PRINT INPUT UNIT NUMBERS (IUNIT) FOR MAJOR OPTIONS.
     100 READ(INBAS,101) IUNIT
101 FORMAT(24I3)  
     WRITE(IOUT,102) (I,1=1,24),IUNIT
102 FORMAT(1H0,'I/O UNITS:'1/X,1/IX,'ELEMENT OF IUNIT:'1,24I3,  
1 /IX,' I/O UNIT:'1,24I3)
C
C8------INITIALIZE TOTAL ELAPSED TIME COUNTER STORAGE ARRAY COUNTER
C9------AND CALCULATE NUMBER OF CELLS.
     TOTIM=0.
     ISUMI=1
     NODES=NROW*NROW*NLAY
C
C9------RETURN
     RETURN
     END
## List of Variables for Module BAS1DF

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>INBAS</td>
<td>Package</td>
<td>Primary unit number from which input to the BAS1 Package will be read. INBAS = 1.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.</td>
</tr>
<tr>
<td>ITMUNI</td>
<td>Package</td>
<td>Code for time units for this problem:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - years</td>
</tr>
<tr>
<td>IUNIT</td>
<td>Module</td>
<td>DIMENSION (24), Primary input units for each of the major options.</td>
</tr>
<tr>
<td>HEADNG</td>
<td>Package</td>
<td>DIMENSION (32), Heading printed on output to identify the problem.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NODES</td>
<td>Global</td>
<td>Number of cells (nodes) in the finite-difference grid.</td>
</tr>
<tr>
<td>NPER</td>
<td>Global</td>
<td>Number of stress periods.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>TOTIM</td>
<td>Package</td>
<td>Elapsed time in the simulation.</td>
</tr>
</tbody>
</table>
Module BASIAL allocates space for data arrays used by the BAS Package. Space is allocated for HNEW, HOLD, IBOUND, CR, CC, CV, HCOF, RHS, DELR, DELC, and IOFLG. Space is also allocated for an array called BUFFER, which is used to accumulate various data arrays such as drawdown and cell-by-cell flow terms when they are being calculated prior to output. To conserve space, the user may specify that arrays BUFFER and RHS should occupy the same space.

The number of spaces allocated for each of the arrays—HOLD, IBOUND, CR, CC, CV, HCOF, RHS, STRT, and BUFFER—is equal to the number of cells in the grid. Twice that number of spaces is reserved for HNEW because it is double precision. DELR and DELC are allocated a number of spaces equal to the number of rows and columns, respectively. IOFLG (an array of flags used by Output Control) is allocated a number of spaces equal to four times the number of layers.

Module BASIAL performs its functions in the following order:

1. Print a message identifying the package.

2. Read and print flags IAPART and ISTRT which indicate whether the BUFFER and RHS arrays should occupy the same space and whether the start array (STRT) should be saved.

3. Store in ISOLD the location in the X array of the first unallocated space. Calculate the number of cells in the grid.

4. Allocate space for HNEW, HOLD, IBOUND, CR, CC, CV, HCOF, RHS, DELR, DELC, and IOFLG.

5. If the user specified that BUFFER and RHS should share space (IAPART equal to zero), set the address of the BUFFER (LCBUFF) equal to the address of RHS (LCRHS); otherwise, allocate separate space for BUFFER.

6. If the user specified that the starting array must be saved, allocate space for STRT.

7. Print the amount of space used by the BAS Package.

8. RETURN.
IAPART is a flag specified by the user which, if equal to zero, indicates that the arrays BUFFER and RHS should overlay each other.

BUFFER is an array in which data is temporarily stored while it is being gathered for printing.

RHS is an array which contains the right hand side of each finite-difference equation.

ISTRT is a flag specified by the user. If it is not equal to zero, starting heads are to be saved.

ISOLD marks the location of ISUM before any space was allocated by this module. After all space is allocated, ISOLD is subtracted from ISUM to calculate the amount of space allocated by this module.

ISUM is a counter which contains the location of the first unallocated element in the X array. Each time space is allocated for an array, the value in ISUM is incremented by the size of the array.
SUBROUTINE BASAL(ISUM,LENX,LOHNEW,LOHOLD,LCIDOU,LCOR,LCGC,LCOV,
  1 LHCOF,LCRHS,LCOCLR,LCDFLC,LCSTRT,LOBUFF,LCIOFL,LIBAS,
  2 ISTRT,MODL,NROW,NLAY,IOUT)
C---VERSION 1515 10/MAY/1987 BASAL
C*******************************************************************************
C ALLOCATE SPACE FOR BASIC MODEL ARRAYS
C*******************************************************************************
C SPECIFICATIONS:
C*******************************************************************************
C
C1-------PRINT A MESSAGE IDENTIFYING THE PACKAGE.
WRITE(IOUT,1) INBAS
  1 FORMAT('BASI -- BASIC MODEL PACKAGE, VERSION 1, 9/1/87',
        ' INPUT READ FROM UNIT',I3)
C
C2-------READ & PRINT FLAG IAPART (RHS & BUFFER SHARE SPACE) AND
C2-------FLAG ISTRT (SHOULD STARTING HEADS BE SAVED FOR DRAWDOWN)
READ(INBAS,2) IAPART,ISTRT
  2 FORMAT(2I10)
     IF(IAPART.EQ.0) WRITE(IOUT,3)
  3 FORMAT('ARRAYS FOR I AND RBUFF WILL SHARE MEMORY,')
     IF(ISTRT.NE.0) WRITE(IOUT,4)
  4 FORMAT('START HEAD WILL BE SAVED')
     IF(ISTRT.EQ.0) WRITE(IOUT,5)
  5 FORMAT('START HEAD WILL NOT BE SAVED',
                '-- DRAWDOWN CANNOT BE CALCULATED')
C
C3-------STORE, IN ISOLD, LOCATION OF FIRST UNALLOCATED SPACE IN X.
ISOLD=ISUM
         NROW=NROW*NCOL*NLAY
C
C4-------ALLOCATE SPACE FOR ARRAYS.
   LOHNEW=ISUM
   ISUM=ISUM+2*NRCL
   LOHOLD=ISUM
   ISUM=ISUM+NRCL
   LCIDOU=ISUM
   ISUM=ISUM+NRCL
   LCOR=ISUM
   ISUM=ISUM+NRCL
   LCGC=ISUM
   ISUM=ISUM+NRCL
   LCOV=ISUM
   ISUM=ISUM+NRCL
   LCFLO=ISUM
   ISUM=ISUM+NLAY*4
C
C5-------IF BUFFER AND RHS SHARE SPACE THEN LOBUFF=LORHS.
   LOBUFF=LORHS
   IF(IAPART.EQ.0) GO TO 50
   ISUM=ISUM+NRCL
C
C6-------IF STRT WILL BESaved THEN ALLOCATE SPACE.
50    LCSTRT=ISUM
   IF(ISTRT.NE.0) ISUM=ISUM+NRCL
   ISP=ISUM-ISOLD
C
C7-------PRINT AMOUNT OF SPACE USED.
   WRITE(IOUT,6) ISP
  6 FORMAT(1A18,1B,1A18,1I8,1A20,1A18)
   ISP=ISP-LENX
   IF(ISUM.GT.LENX) WRITE(IOUT,8)
  8 FORMAT(1X,'***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C8-------RETURN
   RETURN
C
END
### List of Variables for Module BASlAL

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
</table>
| IAPART   | Module| Flag set by user.  
           |       | $= 0$, arrays RHS and BUFFER will share space in the X array.  
           |       | $\neq 0$, arrays RHS and BUFFER will not share space in the X array. |
| INBAS    | Package| Primary unit number from which input to the BASl Package will be read. INBAS = 1. |
| IOUT     | Global| Primary unit number for all printed output. IOUT = 6. |
| ISOLD    | Package| Before this module allocates space, ISOLD is set equal to ISUM. After allocation, ISOLD is subtracted from ISUM to get ISP, the amount of space in the X array allocated by this module. |
| ISP      | Module| Number of words in the X array allocated by this module. Flag.  
           |       | $= 0$, starting heads will be saved so that drawdown can be calculated.  
           |       | $\neq 0$, starting heads will not be saved. |
| LSUM     | Global| Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM. |
| ISTRT    | Package| Index number of the last element of the X array allocated by this module. |
| LCBUFF   | Package| Location in the X array of the first element of array BUFF. |
| LCCC     | Package| Location in the X array of the first element of array CC. |
| LCCR     | Package| Location in the X array of the first element of array CR. |
| LCCV     | Package| Location in the X array of the first element of array CV. |
| LCDEL    | Package| Location in the X array of the first element of array DELC. |
| LCDELR   | Package| Location in the X array of the first element of array DELR. |
| LCHCOF   | Package| Location in the X array of the first element of array HCOF. |
| LCHNEW   | Package| Location in the X array of the first element of array HNEW. |
| LCHOLD   | Package| Location in the X array of the first element of array HOLD. |
| LCIBOU   | Package| Location in the X array of the first element of array IBOUND. |
| LCIOFL   | Package| Location in the X array of the first element of array IOFLG. |
| LCRHS    | Package| Location in the X array of the first element of array RHS. |
| LCSTRT   | Package| Location in the X array of the first element of array STRT. |
| LENX     | Global| Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program. |
| NCOL     | Global| Number of columns in the grid. |
| NLAY     | Global| Number of layers in the grid. |
| NRCL     | Module| Number of cells in the grid. |
| NROW     | Global| Number of rows in the grid. |
This module reads and prepares data for the BAS Package. It reads the boundary array (IBOUND) and the starting-head array (HNEW), sets the heads in no-flow cells to a user-supplied value (for printout convenience), initializes the starting-head array (STRT) and the volumetric-budget accumulators (VBVL), and sets up the Output Control System. The IBOUND codes are as follows:

<table>
<thead>
<tr>
<th>Code</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>negative</td>
<td>constant head</td>
</tr>
<tr>
<td>zero</td>
<td>inactive (no-flow)</td>
</tr>
<tr>
<td>positive</td>
<td>variable head</td>
</tr>
</tbody>
</table>

The user must specify a head value HNOFLO that he wants printed for no-flow (inactive) cells. That value is only used during printing and makes inactive cells stand out on the listing (e.g., 0.0 and 9999.99).

Recall that initial heads are needed for each time step; however, they must be read for only the first time step, at which time they are called the starting heads. For subsequent time steps, the ending heads of the preceding time step will be used as the initial heads of the current time step. The starting heads are read in single precision into the array HOLD and converted to double precision as they are moved into HNEW.

Module BAS1RP performs its functions in the following order:

1. Print the simulation title and calculate the number of cells in a layer.
2. Read the boundary array (IBOUND).
3. Read and print the head value to be printed for no-flow cells (HNOFLO).
4. Read the starting heads into array HOLD.
5. Copy the starting heads (and convert to double precision) from HOLD into HNEW.
6. If the starting heads must be saved, copy them from HOLD to STRT.
7. Initialize volumetric-budget accumulators.
8. Call submodule SBAS1I to initialize the Output Control System.
9. RETURN.
HNOFLO is a value assigned to head in inactive (no-flow) cells. It makes those cells stand out in listings of heads.

HNEW is an array containing the latest estimates of heads. It starts each time step with heads calculated for the end of the previous time step. It is changed at each iteration until the last iteration when it contains the heads at the end of the time step.

HOLD is an array containing heads at the beginning of the current time step. At the beginning of a time step, HOLD AND HNEW contain identical values. HNEW changes from one iteration to the next; HOLD does not.

OUTPUT CONTROL is part of the Basic Package which gives the user the ability to control the kind and amount of information that is printed by the program.

U2DINT is a utility module which reads two-dimensional integer arrays.
SUBROUTINE BASlRP(IBOUND,HNEW,STRT,HOLD,ISTRT,INBAS,
  1 HEADING,ANAME,NNAME,NROW,NLAY,NODES,VBVL,IOFLG,INOC,IHEDFM,
  2 IDDNFM,IHEDUN,IDDNUN,IOUT)
C----------VERSION 1628 15MAY1987 BASlRP
C-----------------------------------------------------------
C READ AND INITIALIZE BASIC MODEL ARRAYS
C-----------------------------------------------------------

C
C SPECIFICATIONS:
C-----------------------------------------------------------
CHARACTER*4 HEADING,ANAME
DOUBLE PRECISION HNEW,HNOFLO
C
DIMENSION HNEW(NODES),IBOUND(NODES),STRT(NODES),HOLD(NODES),
  1 ANAME(6,2),VBVL(4,20),IOFLG(NLAY,4),HEADNG(32)
C
DATA ANAME(1,1),ANAME(2,1),ANAME(3,1),ANAME(4,1),ANAME(5,1),
  1 ANAME(6,1) /' 1,' ',' BD',' A',' RRAY' /
DATA ANAME(1,2),ANAME(2,2),ANAME(3,2),ANAME(4,2),ANAME(5,2),
  1 ANAME(6,2) /' 1,' ',' INIT',' IAL ',' HEAD' /
C
C1------PRINT SIMULATION TITLE, CALCULATE # OF CELLS IN A LAYER.
  WRITE(IOUT,1) HEADNG
  1 FORMAT(1H1,32A4)
  NCR=NCOL*NROW
C
C2------READ BOUNDARY ARRAY(IBOUND) ONE LAYER AT A TIME.
  DO 100 K=1,NLAY
  KK=K
  LOC=1+(K-1)*NCR
  CALL U2DINT(IBOUND(LOC),ANAME(1,1),NROW,NCOL,KK,INBAS,IOUT)
  100 CONTINUE
C
C3------READ AND PRINT HEAD VALUE TO BE PRINTED FOR NO-FLOW CELLS.
  READ(INBAS,Z) TMP
  2 FORMAT(FlO.0)
  HNOFLO=TMP
  WRITE(IOUT,3) TMP
  3 FORMAT(lHOf' AQUIFER HEAD WILL BE SET TO '1PG11.5,
  1 ' AT ALL NO-FLOW NODES (IBOUND=O).')
C
C4------READ STARTING HEADS.
  DO 300 K=1,NLAY
  KK=K
  LOC=1+(K-1)*NCR
  CALL U2DREL(HOLD(LOC),ANAME(1,2),NROW,NCOL,KK,INBAS,IOUT)
  300 CONTINUE
C
C5------COPY INITIAL HEADS FROM HOLD TO HNEW.
  DO 400 I=1,NODES
  HNEW(I)=HOLD(I)
  IF(IBOUND(I).EQ.O) HNEW(I)=HNOFLO
  400 CONTINUE
C
C6------IF STARTING HEADS ARE TO BE SAVED THEN COPY HOLD TO STRT.
  IF(ISTRT.EQ.0) GO TO 590
  DO 500 I=1,NODES
  STRT(I)=HOLD(I)
  500 CONTINUE
C
C7------INITIALIZE VOLUMETRIC BUDGET ACCUMULATORS TO ZERO.
  590 DO 600 I=1,20
  DO 600 J=1,4
  VBVL(J,I)=0.
  600 CONTINUE
C
C8------SET UP OUTPUT CONTROL.
  CALL SBASIII(NLAY,ISTRT,IOFLG,INOC,IOUT,IHEDFM,
  1 IDDNFM,IHEDUN,IDDNUN)
C
C9------RETURN
  1000 RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANAME</td>
<td>Module</td>
<td>DIMENSION (32), Label for printout of input array.</td>
</tr>
<tr>
<td>HEADING</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NLAY), Heading printed on output to identify problem.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HNOFLO</td>
<td>Module</td>
<td>User specified value for head in cells which are inactive at the start of simulation.</td>
</tr>
<tr>
<td>HOLD</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Head at the start of the current time step.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IDDNM</td>
<td>Package</td>
<td>Code for format in which drawdown should be printed.</td>
</tr>
<tr>
<td>IDDUN</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing drawdown should be recorded.</td>
</tr>
<tr>
<td>IHEDFM</td>
<td>Package</td>
<td>Code for format in which head should be recorded.</td>
</tr>
<tr>
<td>IHEDUN</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing head should be recorded.</td>
</tr>
<tr>
<td>INBAS</td>
<td>Package</td>
<td>Primary unit number from which input to BAS1 Package will be read. INBAS = 1.</td>
</tr>
<tr>
<td>INOC</td>
<td>Package</td>
<td>Unit number from which input to output control option will be read.</td>
</tr>
<tr>
<td>IOFLG</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NLAY), Flags to control printing and recording of head and drawdown for each layer.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISTRT</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, starting heads will be saved so that drawdown can be calculated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, starting heads will not be saved.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Temporary variable set equal to K. KK is used as an actual argument in subroutine calls to avoid using the DO loop variable K as an argument, which causes problems with some compilers.</td>
</tr>
<tr>
<td>KK</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>LOC</td>
<td>Module</td>
<td>Pointer to location in an array for a specific layer.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NCR</td>
<td>Module</td>
<td>Number of cells in a layer.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NODES</td>
<td>Global</td>
<td>Number of cells (nodes) in the finite-difference grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>STRT</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NLAY), Starting head.</td>
</tr>
<tr>
<td>TMP</td>
<td>Module</td>
<td>Single-precision temporary storage place for HNOFLO.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For flow component N, the values in VBVL are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1,N) Rate for current time step into the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(2,N) Rate for current time step out of the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,N) Volume into the flow field during simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4,N) Volume out of the flow field during simulation.</td>
</tr>
</tbody>
</table>
Module BAS1ST reads timing information for a stress period and initializes variables used to calculate the length of time steps and elapsed time. Each stress period is divided into time steps which form a geometric progression (for a stress period, there is a multiplier TSMULT such that the length of a time step is equal to TSMULT times the length of the previous time step). If the length of the stress period (PERLEN) and the number of time steps (NSTP) is known, the length of the first time step DELT can be calculated with the equation

\[ \text{DELT} = \frac{(1-\text{TSMULT}) \times \text{PERLEN}}{(1-\text{TSMULT}^\text{NSTP})}. \]

Note: When TSMULT is equal to one, all the time steps are the same length. In that case, the time-step length is the length of the stress period (PERLEN) divided by the number of time steps (NSTP).

Module BAS1ST performs its functions in the following order:

1. Read the length of the stress period (PERLEN), the number of time steps in the stress period (NSTP), and the time-step multiplier (TSMULT).

2. Calculate the length of the first time step.
   (a) Assume the time-step multiplier is equal to one.
   (b) If the time-step multiplier (TSMULT) is not equal to one, calculate the first term of the geometric progression.

3. Print the timing information.

4. Initialize the variable PERTIM which keeps track of elapsed time within a stress period.

5. RETURN.
PERLEN is the length of a stress period.

NSTP is the number of time steps in a stress period.

TSMULT is a constant which, when multiplied by the length of a time step, gives the length of the next time step.

DELT is the length of the first time step. Since the time steps form a geometric progression, the formula for calculating DELT is:

\[ \text{DELT} = \frac{(1 - \text{TSMULT}) \times \text{PERLEN}}{1 - \text{TSMULT}^{\text{NSTP}}} \]

PERTIM is a field in which elapsed time during a stress period is accumulated. During each time step, the length of the time step is added to PERTIM.
SUBROUTINE BAS1ST(NSTP,DEL T,TSMULT,PERTIM,KPER,INBAS,IOUT)

C-----VERSION 1614 08SEP1982 BAS1ST
C
C SETUP TIME PARAMETERS FOR NEW TIME PERIOD
C
C
C SPECIFICATIONS:
C
C
C
C1------READ LENGTH OF STRESS PERIOD, NUMBER OF TIME STEPS AND.
C1------TIME STEP MULTIPLIER.
READ (INBAS,1) PERLEN,NSTP,TSMULT
1 FORMAT(F10.0,I10,F10.0)

C
C2------CALCULATE THE LENGTH OF THE FIRST TIME STEP.
C
C2A------ASSUME TIME STEP MULTIPLIER IS EQUAL TO ONE.
DELT=PERLEN/FLOAT(NSTP)

C
C2B------IF TIME STEP MULTIPLIER IS NOT ONE THEN CALCULATE FIRST
C2B------TERM OF GEOMETRIC PROGRESSION.
IF(TSMULT.NE.1.) DELT=PERLEN*(1.-TSMULT)/(1.-TSMULT**NSTP)

C
C3------PRINT TIMING INFORMATION.
WRITE (IOUT,2) KPER,PERLEN,NSTP,TSMULT,DEL T
2 FORMAT(1HL,51X,'STRESS PERIOD NO.',I4,', LENGTH =',G15.7/52X
1,46('('-')//52X,'NUMBER OF TIME STEPS =',I6
2//53X,'MULTIPLIER FOR DELT =',F10.3
3//50X,'INITIAL TIME STEP SIZE =',G15.7)

C
C4------INITIALIZE PERTIM (ELAPSED TIME WITHIN STRESS PERIOD).
PERTIM=0.

C
C5------RETURN
RETURN
END
### List of Variables for Module BASI1ST

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>INBAS</td>
<td>Package</td>
<td>Primary unit number from which input to the BASI1 package will be read. INBAS = 1.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>NSTP</td>
<td>Global</td>
<td>Number of time steps in the current stress period.</td>
</tr>
<tr>
<td>PERLEN</td>
<td>Module</td>
<td>Length of the stress period.</td>
</tr>
<tr>
<td>PERTIM</td>
<td>Package</td>
<td>Elapsed time during the current stress period.</td>
</tr>
<tr>
<td>TSMULT</td>
<td>Package</td>
<td>Multiplier to get from one time step length to the next.</td>
</tr>
</tbody>
</table>
Module BAS1AD calculates the length of the time step, accumulates the elapsed time for the stress period and the total simulation period, and sets the old head values equal to the new head values.

Within a stress period, the length of the time steps form a geometric progression—the length of each time step is a constant (TSMULT) times the length of the previous time step. The length of the first time step is calculated in module BAS1ST.

The array HNEW contains the heads calculated for the end of the last time step. Those heads which are also the heads at the beginning of the current time step are copied into HOLD.

Module BAS1AD performs its functions in the following order:

1. If this is not the first time step in the stress period, calculate the length of the time step (DELT). Note: The length of the first time step is calculated by BAS1ST.

2. Accumulate the elapsed time since the beginning of the simulation period (TOTIM) and the beginning of the stress period (PERTIM).

3. Set the heads at the beginning of this time step (HOLD) equal to the heads at the end of the previous time step (HNEW).

4. RETURN.
TOTIM is an accumulator in which the total elapsed time since the beginning of the simulation is stored.

PERTIM is an accumulator in which the total elapsed time during the current stress period is stored.

HOLD is the head distribution at the beginning of a time step.

HNEW is the head distribution at the end of a time step.
SUBROUTINE BAS1AD(DEL,TSMULT,TOTIM,PERTIM,HNEW,HOLD,KSTP,
    NCOL,NROW,NLAY)

C-----VERSION 1412 22FEB1982 BAS1AD
C
C ******************************************************************
C ADVANCE TO NEXT TIME STEP
C ******************************************************************
C
C SPECIFICATIONS:
C ***************************************************************
DOUBLE PRECISION HNEW
C
DIMENSION HNEW(NCOL,NROW,NLAY), HOLD(NCOL,NROW,NLAY)
C ***************************************************************
C
C1------IF NOT FIRST TIME STEP THEN CALCULATE TIME STEP LENGTH.
    IF(KSTP.NE.1) DELT=TSMULT*DEL T
C
C2------ACCUMULATE ELAPSED TIME IN SIMULATION(TOTIM) AND IN THIS
C2------STRESS PERIOD(PERTIM).
    TOTIM=TOTIM+DEL T
    PERTIM=PERTIM+DEL T
C
C3------COPY HNEW TO HOLD.
    DO 10 K=1,NLAY
    DO 10 I=1,NROW
    DO 10 J=1,NCOL
    10 HOLD(J,I,K)=HNEW(J,I,K)
C
C4------RETURN
    RETURN
    END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HOLD</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Head at the start of the current time step.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Row index.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Column index.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Layer index.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>PERTIM</td>
<td>Package</td>
<td>Elapsed time during the current stress period.</td>
</tr>
<tr>
<td>TOTIM</td>
<td>Package</td>
<td>Elapsed time in the simulation.</td>
</tr>
<tr>
<td>TSMULT</td>
<td>Package</td>
<td>Multiplier to get from one time step length to the next.</td>
</tr>
</tbody>
</table>
Narrative for Module BAS1FM

This module initializes the arrays in which the right hand side (RHS) and the h-coefficient (HCOF) are accumulated.

Recall that the equation for cell $i,j,k$ contains a term $RHS_{i,j,k}$ on the right hand side and a coefficient $HCOF_{i,j,k}$ (h-coefficient) which multiplies $h_{i,j,k}$ on the left hand side of the equation. The right-hand-side term and the h-coefficient are the sum of terms related to many of the flow components. They are calculated every time the equations are formulated.

Module BAS1FM performs its functions in the following order:

1. For each cell, initialize (set equal to zero) the HCOF and RHS accumulators.

2. RETURN.
SUBROUTINE BAS1FM(HCOF,RHS,NODES)

C

C

C

C-----VERSION 1632 24JUL1987 BAS1FM
C

C

C

C

********i*********************************************************

C

C SPECIFICATIONS:
C ------------------------------------------------------------------

DIMENSION HCOF(NODES),RHS(NODES)
C ------------------------------------------------------------------

C

C1------FOR EACH CELL INITIALIZE HCOF AND RHS ACCUMULATORS.
DO 100 I=1,NODES
   HCOF(I)=0.
   RHS(I)=0.
100 CONTINUE
C

C2------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NODES). Coefficient of head in cell (J,I,K) in the finite-difference equation.</td>
</tr>
<tr>
<td>NODES</td>
<td>Global</td>
<td>Number of cells (nodes) in the finite-difference grid.</td>
</tr>
<tr>
<td>RHS</td>
<td>Global</td>
<td>DIMENSION (NODES), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.</td>
</tr>
</tbody>
</table>
Narrative for Module BAS1OC

Module BAS1OC sets flags used by the budget and output procedures to determine what data should be printed or recorded on disk. There are three individual flags and one table of flags. The individual flags are IHDDFL which indicates that head or drawdown is to be printed or recorded, IBUDFL which indicates that the overall budget should be printed, and ICBCFL which indicates that cell-by-cell flow terms should be calculated and printed or recorded. The table of flags called IOFLG has four flags for each layer. They correspond to the four options: print heads, print drawdown, save heads, and save drawdown. The flags in IOFLG are used in conjunction with the flag IHDDFL. If IHDDFL is set, IOFLG is used to determine head and drawdown on a layer-by-layer basis. If IHDDFL is not set, heads and drawdown are not printed or saved and IOFLG is ignored.

If the user is controlling output, the flags are read at each time step; if not, IOFLG is set at the start of the simulation and the individual flags are set at each time step.

Module BAS1OC performs its functions in the following order:

1. Determine if the user has specified that he will control output. He does so by coding a positive integer in the twelfth element of the IUNIT table. That integer is read by module BAS1DF and is passed to this module (BAS1OC) under the name INOC. Go to either 2 or 3.

2. The user is not controlling output. Set flags for default-output and then return. Flags IHDDFL and IBUDFL are set only at the last time step in each stress period or when the iterative procedure fails to converge. RETURN.

3. The user has chosen to control output. Read and print the code INCODE and flags IHDDFL, IBUDFL, and ICBCFL. The code INCODE gives the user several options for specifying the flag table IOFLG.

4. Determine whether INCODE is less than zero, equal to zero, or greater than zero. Go to 5, 6, or 7.

5. INCODE is less than zero. Use the IOFLG flags used in the previous time step and print a message to that effect. Go to 8.

6. INCODE is equal to zero. Read IOFLG for layer 1 and then set flags in all other layers equal to those in layer 1. Go to 8.

7. INCODE is greater than zero. Read IOFLG array. Go to 8.

8. Regardless of what the user has specified, set the flag IBUDFL if the iterative procedure failed to converge or if the current time step is the last time step in the stress period.

9. RETURN.
INOC is the input unit for Output Control specifications. It is specified by the user as the twelfth element of the IUNIT array. If it is less than or equal to zero, the user has chosen the default output. If it is greater than zero, the user has chosen to control output.

INCODE provides the user with options for filling the IOFLG array.

If INCODE < 0, IOFLG from the last time step is reused.

If INCODE = 0, IOFLG for layer 1 is read and all other layers are set equal to layer 1.

If INCODE > 0, IOFLG is read.

IOFLG is a table of flags with one entry for each layer. Each entry has four flags:

1---head print
2---drawdown print
3---head save
4---drawdown save

If a flag is set (equal to 1), head or drawdown for the corresponding layer is either printed or saved on disk.

IHDDFL is the head/drawdown flag. If it is set, heads and drawdowns will be written in accordance with the flags in IOFLG.

IBUDFL is the budget print flag. If it is set, the overall budget will be printed.

ICBCFL is the cell-by-cell flow term flag. If it is set, cell-by-cell flow terms will be printed or recorded on disk for those components of flow for which the CBC flag (IWELCB, IRCHCB, IDRNCB, etc.) is set.
SUBROUTINE BASLCC(NSTP,KSTP,ICNVG,IOFLG,NLAY, 
   IBUDFL,ISCBCFL,IHDDFL,INOC,IOUT)
C
C-------VERSION 1632 24JUL1987 BASLCC
C
C OUTPUT CONTROLLER FOR HEAD, DRAWDOWN, AND BUDGET
C
C SPECIFICATIONS:
C
C DIMENSION IOFLG(NLAY,4)
C
C
C C1------TEST UNIT NUMBER (INOC = IUNIT(12)) TO SEE IF
C C CONTROL IS ACTIVE.
C IF(INOC .EQ. 0) GO TO 500
C
C C2------IF OUTPUT CONTROL IS INACTIVE THEN SET DEFAULTS AND RETURN.
C IF(INOC .EQ. 0) .OR. (KSTP .EQ. NSTP) IHDDFL = 1
C IF(INOC .EQ. 0) .OR. (KSTP .EQ. NSTP) IBUDFL = 1
C IF(INOC .EQ. 0) .OR. (KSTP .EQ. NSTP) ISCBCFL = 1
C GO TO 1000
C
C C3------READ AND PRINT OUTPUT FLAGS AND CODE FOR DEFINING IOFLG.
C 500 READ(INOC,1) INCODE,IHDDFL,IBUDFL,ISCBCFL
C 1 FORMAT(4110)
C WRITE(IOUT,3) IHDDFL,IBUDFL,ISCBCFL
C 3 FORMAT(1H0,'HEAD/DRAWDOWN PRINTOUT FLAG =',I2,1X,'CELL-BY-CELL FLOW TERM FLAG =',I2)
C
C C4------DECODE INCODE TO DETERMINE HOW TO SET FLAGS IN IOFLG.
C IF(INCODE) 100,200,300
C
C C5------USE IOFLG FROM LAST TIME STEP.
C 100 WRITE(IOUT,101)
C 101 FORMAT(1H0,'REUSING PREVIOUS VALUES OF IOFLG')
C GO TO 600
C
C C6------READ IOFLG FOR LAYER 1 AND ASSIGN SAME TO ALL LAYERS
C 200 READ(INOC,201) (IOFLG(K,1),K=1,NLAY)
C 201 FORMAT(4110)
C DO 210 K=1,NLAY
C IOFLG(K,1)=IOFLG(1,1)
C IOFLG(K,2)=IOFLG(1,2)
C IOFLG(K,3)=IOFLG(1,3)
C IOFLG(K,4)=IOFLG(1,4)
C 210 CONTINUE
C WRITE(IOUT,211) (IOFLG(K,1),K=1,NLAY)
C 211 FORMAT(1H0,'OUTPUT FLAGS FOR ALL LAYERS ARE THE SAME:/',1X,12X,'HEAD',12X,'DRAWDOWN',12X,'PRINTOUT',12X,'SAVE')
C 7X,12X,'HEADDRAWDOWN',12X,'PRINTOUT',12X,'SAVE')
C 2 DATA 1,1,1,1
C GO TO 600
C
C C7------READ IOFLG IN ENTIRETY
C 300 READ(INOC,301) ((IOFLG(K,I),I=1,4),K=1,NLAY)
C 301 FORMAT(4110)
C WRITE(IOUT,302)
C 302 FORMAT(1H0,'OUTPUT FLAGS FOR EACH LAYER:',/1X,4X,'LAYER',/1X,12X,'PRINTOUT',/1X,12X,'SAVE')
C 3 DATA 1,1,1,1
C WRITE(IOUT,303) (K,(IOFLG(K,I),I=1,4),K=1,NLAY)
C 303 FORMAT(1X,14,10,110,10,18)
C
C C8------THE LAST STEP IN A STRESS PERIOD AND STEPS WHERE ITERATIVE
C PROCEDURE FAILED TO CONVERGE GET A VOLUMETRIC BUDGET.
C 600 IF(INOC .EQ. 0) .OR. (KSTP .EQ. NSTP) IBUDFL = 1
C
C C9------RETURN
C 1000 RETURN
END
List of Variables for Module BAS10C

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>IBDUFL</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, volumetric budget will not be printed for the current time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, volumetric budget should be printed for the current time step.</td>
</tr>
<tr>
<td>ICBCFL</td>
<td>Global</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be recorded or printed for the current time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, cell-by-cell flow terms will be recorded or printed for the current time step.</td>
</tr>
<tr>
<td>ICNVG</td>
<td>Global</td>
<td>Flag is set equal to 1 when the iteration procedure has converged.</td>
</tr>
<tr>
<td>IHDDFL</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, neither head nor drawdown will be printed at this time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, head and drawdown may be printed at the end of the current time step.</td>
</tr>
<tr>
<td>INCODE</td>
<td>Module</td>
<td>Code specified by user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, reuse contents of IOFLG from the last time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, read IOFLG for layer 1 and set all other layers to the same thing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, read IOFLG contents for each layer.</td>
</tr>
<tr>
<td>INOC</td>
<td>Package</td>
<td>Unit number from which input to output control option will be read.</td>
</tr>
<tr>
<td>IOFLG</td>
<td>Package</td>
<td>DIMENSION (NLAY,4), Flags to control printing and recording of head and drawdown for each layer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,1) ≠ 0, heads will be printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,2) ≠ 0, drawdown will be printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,3) ≠ 0, heads will be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,4) ≠ 0, drawdown will be recorded.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Layer index.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>M</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NSTP</td>
<td>Global</td>
<td>Number of time steps in the current stress period.</td>
</tr>
</tbody>
</table>
Module BASIOT invokes submodules which write results of the simulation. Those results include head, drawdown, overall volumetric budget, and a time summary. Results are printed according to flags IHDDFL, IOFLG, and IBUDFL which are set by module BAS1OC (Output Control). If flag IHDDFL is set, a table of flags named IOFLG is used to determine which heads and drawdown should be written (printer or disk) and for which layers it should be written. This module (BASIOT) calls submodules SBASIH and SBASID to write heads and drawdowns respectively. If flag IBUDFL is set, submodule SBASIV is invoked to calculate and print the overall volumetric budget. After every time step during which results have been printed, a time summary is printed.

Module BASIOT performs its functions in the following order:

1. Clear flag IPFLG. This flag is set later in this module if any results are printed. It controls the printing of a time summary.

2. If the iterative procedure failed to converge, print a message to that effect.

3. If the head and drawdown flag (IHDDFL) are set, call submodules SBASIH and SBASID to write heads and drawdowns in accordance with the flags in the table IOFLG.

4. If the budget flag (IBUDFL) is set, call submodule SBASIV to calculate and print the volumetric budget.

5. If the printout flag (IPFLG) is set, call submodule SBASIT to print a time summary.

6. RETURN.
IPFLG is the printout flag. It is set when any results are printed. If it is set, a time summary is printed.

SBAS1H is a submodule which writes heads.

SBAS1D is a submodule which writes drawdown.

SBAS1V is a submodule which prints the volumetric budget.

SBAS1T is a submodule which prints a time summary.

IHDDFL is the head/drawdown flag. If it is set, heads and drawdown will be written in accordance with flag settings in IOFLG.

IBUDFL is the budget print flag. If it is set, volumetric budget will be printed.

IOFLG is a table of flags with one entry for each layer. Each entry has four flags:

1 --- head print
2 --- drawdown print
3 --- head save
4 --- drawdown save

IF IPFLG IS SET, CALL SBAS1H TO WRITE HEAD AND DRAWDOWN.

IF IPFLG IS SET, CALL SBAS1D TO WRITE HEAD AND DRAWDOWN.

IF IPFLG IS SET, CALL SBAS1V TO PRINT BUDGET.

IF IPFLG IS SET, CALL SBAS1T TO PRINT TIME SUMMARY.

RETURN
SUBROUTINE BASLOT(HNEW, STRT, ISTRT, BUFF, IOFLG, MSUM, IBOUND, VBNM,
                1 VBLV, KSTP, KPER, DELT, PERTIM, TOTIM, ITMUNI, NCOL, NROW, NLAY, ICNVG,
                2 IHDDFL, IBUDFL, IHEDFM, IHEDUN, IDDNM, IDDNU, IOUT)

C-----VERSION 1522 12MAY1987 BASLOT
C
C OUTPUT TIME, VOLUMETRIC BUDGET, HEAD, AND DRAWDOWN
C
C SPECIFICATIONS:
C -------------------------- ----------------------------------------
CHARACTER*4 VBNM
DOUBLE PRECISION HNEW

DIMENSION HNEW(NCOL, NROW, NLAY), STRT(NCOL, NROW, NLAY),
1 VBVM(4,20), VBVL(4,20), IOFLG(NLAY,4),
2 IBOUND(NCOL, NROW, NLAY), BUFF(NCOL, NROW, NLAY)

------------------------------------------------------------------
C
C1------CLEAR PRINTOUT FLAG (IPFLG)
   IPFLG=0

C2------IF ITERATIVE PROCEDURE FAILED TO CONVERGE PRINT MESSAGE
   IF(ICNVG.EQ.0) WRITE(IOUT,1) KSTP, KPER
      1 FORMAT(1HO,10X,'****FAILED TO CONVERGE IN TIME STEP',I3,
               1 ' OF STRESS PERIOD',I3,'****')

C3------IF HEAD AND DRAWDOWN FLAG (IHDDFL) IS SET WRITE HEAD AND
C3------DRAWDOWN IN ACCORDANCE WITH FLAGS IN IOFLG.
      IF(IHDDFL.EQ.0) GO TO 100

C CALL SBASIH(HNEW, BUFF, IOFLG, KSTP, KPER, NCOL, NROW,
1       NLAY, IOUT, IHEDFM, IHEDUN, IPFLG, PERTIM, TOTIM)
CALL SBASID(HNEW, BUFF, IOFLG, KSTP, KPER, NCOL, NROW, NLAY, IOUT,
1       IDDNM, IDDNU, ISTRT, IBOUND, IPFLG, PERTIM, TOTIM)

C4------PRINT TOTAL BUDGET IF REQUESTED
      100 IF(IBOUND.EQ.0) GO TO 120
      CALL SBASIV(MSUM, VBNM, VBVL, KSTP, KPER, IOUT)
      IPFLG=1

C5------END PRINTOUT WITH TIME SUMMARY AND FORM FEED IF ANY PRINTOUT
C5------WILL BE PRODUCED.
      120 IF(IPFLG.EQ.0) RETURN
      CALL SBASIT(KSTP, KPER, DELT, PERTIM, TOTIM, ITMUNI, IOUT)
      WRITE(IOUT,101)
      101 FORMAT(1H1)

C6------RETURN
   RETURN
END
### List of Variables for Module BAS1OT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IBUDFL</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, volumetric budget will not be printed for the current time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, volumetric budget should be printed for the current time step.</td>
</tr>
<tr>
<td>ICNVG</td>
<td>Global</td>
<td>Flag is set equal to one when the iteration procedure has converged.</td>
</tr>
<tr>
<td>IDDNUM</td>
<td>Package</td>
<td>Code for format in which drawdown should be printed.</td>
</tr>
<tr>
<td>IDDNUM</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing drawdown should be recorded.</td>
</tr>
<tr>
<td>IHDDFL</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, neither head nor drawdown will be printed at this time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, head and drawdown may be printed at the end of the current time step.</td>
</tr>
<tr>
<td>IHEDFM</td>
<td>Package</td>
<td>Code for the format in which head should be printed.</td>
</tr>
<tr>
<td>IHEDUN</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing head should be recorded.</td>
</tr>
<tr>
<td>IOFLG</td>
<td>Package</td>
<td>DIMENSION (NLAY,4), Flags to control printing and recording of head and drawdown for each layer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,1) ≠ 0, heads will be printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,2) ≠ 0, drawdown will be printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,3) ≠ 0, heads will be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,4) ≠ 0, drawdown will be recorded.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPFLG</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, starting heads will be saved so that drawdown can be calculated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, starting heads will not be saved.</td>
</tr>
<tr>
<td>ISTRT</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, starting heads will be saved so that drawdown can be calculated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, starting heads will not be saved.</td>
</tr>
<tr>
<td>ITMUNI</td>
<td>Package</td>
<td>Code for time units for this problem:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - years</td>
</tr>
</tbody>
</table>

4-49
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>PERTIM</td>
<td>Package</td>
<td>Elapsed time during the current stress period.</td>
</tr>
<tr>
<td>STRT</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NLAY), Starting head.</td>
</tr>
<tr>
<td>TOTIM</td>
<td>Package</td>
<td>Elapsed time in the simulation.</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For flow component N, the values in VBVL are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1,N) Rate for the current time step into the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(2,N) Rate for the current time step out of the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,N) Volume into the flow field during simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4,N) Volume out of the flow field during simulation.</td>
</tr>
</tbody>
</table>
Narrative for Module SBAS1D

Module SBAS1D is called by module BAS1OT to calculate and write drawdown for every cell in certain layers in the grid. The module is called at the end of each time step if the head and drawdown flag (IHDDFL) is set. It calculates drawdown only if the user has specified that starting heads should be saved.

The layers for which drawdown is to be written are determined by the settings of flags in the table named IOFLG. In IOFLG, there are four flags for each layer. The second flag, if it is set, causes drawdown to be printed. The fourth flag, if it is set, causes drawdown to be recorded.

Module SBAS1D performs its functions in the following order:

1. For each layer, do steps 2-5.
2. If flags indicate that drawdown is not needed for this layer, go on to the next layer.
3. Test flag ISTRT to see if starting heads were saved. Go to either 4 or 5.
4. Starting heads were not saved. Write a message to that effect and STOP.
5. Starting heads were saved. Calculate drawdown for this layer.
6. For each layer, if drawdown is to be printed, call module ULAPRS or ULAPRW, depending on the format requested (IDDNFM), to print drawdown.
7. For each layer, if drawdown is to be recorded, call module ULASAV to write the drawdown to the unit specified in IDDNUN.
8. RETURN.
IOFLG is a table containing one entry for each layer. Each element consists of four flags which, when set, cause (1) head to be printed, (2) drawdown to be printed, (3) head to be recorded, and (4) drawdown to be recorded.

ULAPRW is a utility module which prints a value for each cell in a wrap format. In the wrap format, all values for one row are printed before any values for the next row.

ULAPRS is a utility module which prints a value for each cell in the layer in a strip format. In the strip format, all values in a group of N columns are printed before any values in the next N columns are printed.

ULASAV is a utility module which records a value for each cell in a layer.

IDDNUN is a unit number, specified by the user, on which drawdown will be recorded.
SUBROUTINE SBASID(HNEW,BUFF,IOFLG,KSTF',KPERINCOL,NRCW,
1 NLAY,IOUT,IDDNM,NLAY,STRT,ISTRT,IPFLG,
2 PERTIM,TOTIM)
C-----VERSION 1630 1SMAY1987 SBASID
C ***************************************************
C CALCULATE PRINT AND RECORD DRAWDOWNS
C ***************************************************
C SPECIFICATIONS
C -------------------------------------------------------
CHARACTER*4 TEXT
DOUBLE PRECISION HMW
C DIMENSION HNEW(NCOL,NROW,NLAY),IOFLG(NLAY,4),TEXT(4),
1 BUFF(NCOL,NROW,NLAY),STRT(NCOL,NROW,NLAY),
2 IBOUND(NCOL,NROW,NLAY)
C DATA TEXT(1),TEXT(2),TEXT(3),TEXT(4) /' ',' ','DRAW',
1 'DOWN'/
C -------------------------------------------------------
C----FOR EACH LAYER CALCULATE DRAWDOWN IF PRINT OR RECORD
C1------IS REQUESTED
DO 59 K=1,NLAY
C2------IS DRAWDOWN NEEDED FOR THIS LAYER?
IF(IOFLG(K,2).EQ.0 .AND. IOFLG(K,4).EQ.0) GO TO 59
C3------DRAWDOWN IS NEEDED. WERE STARTING HEADS SAVED?
IF(ISTRT.NE.0) GO TO 53
C4------STARTING HEADS WERE NOT SAVED. PRINT MESSAGE AND STOP.
WRITE(IOUT,52)
52 FORMAT(10D,H,'CANNOT CALCULATE DRAWDOWN BECAUSE ST',
1 'ART HEADS WERE NOT SAVED')
STOP
C5------CALCULATE DRAWDOWN FOR THE LAYER.
53 DO 58 I=1,NROW
DO 58 J=1,NCOL
HSING=HNEW(J,I,K)
BUFF(J,I,K)=HSING
IF(IBEUN(J,I,K).NE.0) BUFF(J,I,K)=STRT(J,I,K)-HSINC
58 CONTINUE
59 CONTINUE
C6------FOR EACH LAYER: DETERMINE IF DRAWDOWN SHOULD BE PRINTED.
C7------IF SO THEN CALL ULAPRS OR ULAPRW TO PRINT DRAWDOWN.
DO 69 K=1,NLAY
KK=K
IF(IOFLG(K,2).EQ.0) GO TO 69
IF(KSTP,LT.0) CALL ULAPRS(BUFF(1,1,K),TEXT(1),KSTP,KPER,
1 NCOL,NROW,KK,-IDDNM,IOUT)
IF(KSTP.GE.0) CALL ULAPRW(BUFF(1,1,K),TEXT(1),KSTP,KPER,
1 NCOL,NROW,KK,IDDNM,IOUT)
IPFLG=1
69 CONTINUE
C8------FOR EACH LAYER: DETERMINE IF DRAWDOWN SHOULD BE RECORDED.
C----IF SO THEN CALL ULASAV TO RECORD DRAWDOWN.
79 CONTINUE
C9------RETURN
80 RETURN
END
### List of Variables for Module SBAS1D

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HSING</td>
<td>Module</td>
<td>Single-precision temporary field for HNEW (J,I,K).</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$&lt; 0$, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= 0$, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$&gt; 0$, variable-head cell</td>
</tr>
<tr>
<td>IDDNFM</td>
<td>Package</td>
<td>Code for format in which drawdown should be printed.</td>
</tr>
<tr>
<td>IDDNUM</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing drawdown should be recorded.</td>
</tr>
<tr>
<td>IFIRST</td>
<td>Module</td>
<td>Flag to indicate that a notice should be printed when drawdown is recorded.</td>
</tr>
<tr>
<td>IOFLG</td>
<td>Package</td>
<td>DIMENSION (NLAY,4), Flags to control printing and recording of head and drawdown for each layer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,1) $\neq 0$, heads will be printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,2) $\neq 0$, drawdown will be printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,3) $\neq 0$, heads will be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NLAY,4) $\neq 0$, drawdown will be recorded.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPFLG</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= 0$ means nothing has been printed this time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\neq 0$ means something has been printed this time step; therefore, a time summary must be printed.</td>
</tr>
<tr>
<td>ISRT</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\neq 0$, starting heads will be saved so that drawdown can be calculated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= 0$, starting heads will not be saved.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KK</td>
<td>Module</td>
<td>Temporary variable set equal to K. KK is used as an actual argument in subroutine calls to avoid using the DO loop variable K as an argument, which causes problems with some compilers.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>PERTIM</td>
<td>Package</td>
<td>Elapsed time during the current stress period.</td>
</tr>
<tr>
<td>STRT</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NLAY), Starting head.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with array data.</td>
</tr>
<tr>
<td>TOTIM</td>
<td>Package</td>
<td>Elapsed time in the simulation.</td>
</tr>
</tbody>
</table>
Narrative for Module SBAS1H

Module SBAS1H prints and records head for every cell in certain layers in the grid. It is called by module BAS1OT at the end of each time step if the head and drawdown flag (IHDDFL) is set. The layers for which head is written is controlled by the settings of flags in the table named IOFLG. In IOFLG, there are four flags for each layer. The first flag, if it is set, causes head for the corresponding layer to be printed. The third flag, if it is set, causes head to be recorded.

Module SBAS1H performs its functions in the following order:

1. For each layer, DO STEPS 2-4.

2. Test the flag table (IOFLG) to see if heads should be printed for this layer. If so, DO STEPS 3 AND 4.

3. Copy heads for this layer (which are contained in the double-precision array (HNEW)) into the single-precision buffer array (BUFF).

4. Depending on the print-format code, call either module ULAPRW or ULAPRS to print the contents of the buffer array.

5. Test the unit number for recording heads (IHEDUN) to see if it is positive. If it is not positive, heads will not be recorded (SKIP STEPS 6-9). If it is positive, heads may be recorded in accordance with the setting of flags in the IOFLG array. DO STEPS 6-9.

6. For each layer, DO STEPS 7-9.

7. If flags in IOFLG indicate that heads are not to be recorded for this layer, move on to the next layer.

8. Copy heads from the HNEW array (double-precision) to the BUFF array (single-precision).

9. Call module ULASAV to record the heads on unit IHEDUN.

10. RETURN.
Flow Chart for Module SBAS1H

IOFLG is a table containing one entry for each layer. Each element consists of four flags which, when set, cause (1) head to be printed, (2) drawdown to be printed, (3) head to be recorded, and (4) drawdown to be recorded.

ULAPRW is a utility module which prints a value for each cell in the layer in a wrap format. In the wrap format, all values for one row are printed before any values for the next row.

ULAPRS is a utility module which prints a value for each cell in the layer in a strip format. In the strip format, all values in a group of N columns are printed before any values in the next N columns are printed.

ULASAV is a utility module which records a value for each cell in a layer.

IHEDUN is a unit number, specified by the user, on which heads will be recorded. If IHEDUN is less than or equal to zero, heads will not be recorded.
SUBROUTINE SBASIH(HNEW, BUFFER, IOFLG, KSTP, KPER, NCOL, NROW,
1 NLAY, IOUT, IHEDFM, IHEDUN, IPFLG, PERTIM, TOTIM)

C-------------------------------------------------------------
C               VERSION 1563 15MAY1987 SBASIH
C-------------------------------------------------------------
C PRINT AND RECORD HEADS
C-------------------------------------------------------------
C SPECIFICATIONS
C-------------------------------------------------------------
CHARACTER*4 TEXT
DOUBLE PRECISION HNEW
C DIMENSION HNEW(NCOL, NROW, NLAY), IOFLG(NLAY, 4), TEXT(4),
1 BUFFER(NCOL, NROW, NLAY)
C DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /' ', ', ', '
1 'HEAD'/
C-------------------------------------------------------------
C1-------FOR EACH LAYER: PRINT HEAD IF REQUESTED.
DO 39 K=1, NLAY
   KK=K
C C2-------TEST IOFLG TO SEE IF HEAD SHOULD BE PRINTED.
   IF(IOFLG(K,1).EQ.0) GO TO 39
      IPFLG=1
C C3-------COPY HEADS FOR THIS LAYER INTO BUFFER.
   DO 32 I=1, NROW
      DO 32 J=1, NCOL
         BUFFER(J, I, K) = HNEW(J, I, KK)
32 CONTINUE
C C4-------CALL UTILITY MODULE TO PRINT CONTENTS OF BUFFER.
   IF(IHEDFM.LT.0) CALL ULAPRS(BUFFER, TEXT(1), KSTP, KPER, NCOL, NROW, KK,
1 -IHEDFM, IOUT)
   IF(IHEDFM.GE.0) CALL ULAPRW(BUFFER, TEXT(1), KSTP, KPER, NCOL, NROW, KK,
1 IHEDFM, IOUT)
39 CONTINUE
C C5-------IF UNIT FOR RECORDING HEADS <= 0: THEN RETURN.
   IF(IHEDUN.LE.0) GO TO 50
      IFIRST=1
C C6-------FOR EACH LAYER: RECORD HEAD IF REQUESTED.
   DO 49 K=1, NLAY
      KK=K
C C7-------CHECK IOFLG TO SEE IF HEAD FOR THIS LAYER SHOULD BE RECORDED.
   IF(IOFLG(K,3).LE.0) GO TO 49
   IF(IFIRST.EQ.1) WRITE(IOUT,41) IHEDUN, KSTP, KPER
41 FORMAT('HEAD WILL BE SAVED ON UNIT', I3, ' AT END OF TIME STEP',
1 's STRESS PERIOD', I3)
      IFIRST=0
C C8-------COPY HEADS FOR THIS LAYER INTO BUFFER.
   DO 44 I=1, NROW
      DO 44 J=1, NCOL
         BUFFER(J, I, K) = HNEW(J, I, KK)
44 CONTINUE
C C9-------RECORD CONTENTS OF BUFFER ON UNIT=IHEDUN
   CALL ULASAV(BUFFER, TEXT(1), KSTP, KPER, PERTIM, TOTIM, NCOL, NROW, KK,
1 IHEDUN)
49 CONTINUE
C C10-------RETURN
50 RETURN
END
## List of Variables for Module SBASIH

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IFIRST</td>
<td>Module</td>
<td>Flag which, if set (equal to 1), indicates that a notice should be printed when head is recorded.</td>
</tr>
<tr>
<td>IHEDFM</td>
<td>Package</td>
<td>Code for format in which head should be printed.</td>
</tr>
<tr>
<td>IHEDUN</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing head should be recorded.</td>
</tr>
<tr>
<td>IOFLG</td>
<td>Package</td>
<td>DIMENSION (NLAY,4), Flags to control printing and recording of head and drawdown for each layer. (NLAY,1) ≠ 0, heads will be printed. (NLAY,2) ≠ 0, drawdown will be printed. (NLAY,3) ≠ 0, heads will be recorded. (NLAY,4) ≠ 0, drawdown will be recorded.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPFLG</td>
<td>Package</td>
<td>Flag. = 0 means nothing has been printed this time step. ≠ 0 means something has been printed this time step; therefore, a time summary must be printed.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KK</td>
<td>Module</td>
<td>Temporary variable set equal to K. KK is used as an actual argument in subroutine calls to avoid using the DO loop variable K as an argument, which causes problems with some compilers.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>PFRTIM</td>
<td>Package</td>
<td>Flapsed time during the current stress period.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with array data.</td>
</tr>
<tr>
<td>TOTIM</td>
<td>Package</td>
<td>Elapsed time in the simulation.</td>
</tr>
</tbody>
</table>
Module SBASII initializes the Output Control System. If the user does not opt to control output, the formats for printing head and drawdown are set to the default format and flags are set so that, whenever heads or drawdowns are printed, they are printed for all layers. If the user does opt to control output, the formats for printing and the unit numbers for recording head and drawdown are read.

A table named IOFLG contains one entry for each layer in the grid. Each entry consists of four flags corresponding to four operations: (1) head print, (2) drawdown print, (3) head record, and (4) drawdown record. The module BASIOT examines the table and, for each layer, performs only the operations for which the corresponding flags are set (equal to one). This module (SBASII) sets the head-print flag if the user opts for default output. If starting heads are saved, it also sets the drawdown-print flag. If the user opts to control output, the flags in IOFLG are read at each time step.

Module SBASII performs its functions in the following order:

1. Test the unit number for Output Control (IUNIT (12)), which is known in this module by the name INOC, to see if it is positive. If it is positive, the Output Control option is active and output specification will be read from the unit number contained in INOC. If it is not positive, the Output Control option is not active and flags are set to defaults. GO TO 2 OR 3.

2. Output Control is active. Read and print the head-print format code (IHEDFM), the drawdown-print format code (IDDNFM), the unit number to record heads (IHEDUN), and the unit number to record drawdown (IDDNUN). GO TO 6. Note: The formats and associated codes are listed in the Input Instructions for Output Control.

3. Output Control is inactive. Print a message listing the defaults.

4. Set the print-format codes (IHEDFM and IDDNNFM) equal to zero to get the default format.

5. Set the flags in IOFLG so that head and drawdown are printed for all layers.

6. RETURN.
INOC is the input unit for Output Control. It is the same as element 12 in the IUNIT table. When it is greater than zero, Output Control is active—the user will provide output specifications. When it is less than or equal to zero, Output Control is inactive—output will be controlled by default.

IOFLG is a table containing one entry for each layer. Each entry consists of four flags which, when set, cause (1) head to be printed, (2) drawdown to be printed, (3) head to be recorded, and (4) drawdown to be recorded.
SUBROUTINE SBAS1I(NLAY,ISTRT,IOFLG,INOC,IOUT,IHEDFM, 
1 IDDNFM,IHEDMUN,IDDNUN)
C
C------VERSION 1531 12MAY1987 SBAS1J
C
C  SET UP OUTPUT CONTROL
C  ***********************************************************************
C
C  SPECIFICATIONS:
C  _________________________________________________________________
DIMENSION IOFLG(NLAY,4)
C  __________________________________________________________________
C
C1------TEST UNIT NUMBER FROM IUNIT (INOC) TO SEE IF OUTPUT
C1------CONTROL IS ACTIVE.
     IF(INOC.LE.0) GO TO 600
C
C2------READ AND PRINT FORMATS FOR PRINTING AND UNIT NUMBERS FOR
C2------RECORDING HEADS AND DRAWDOWN. THEN RETURN.
   500 READ (INOC,1)IHEDFM,IDDNFM,IHEDMUN,IDDNUN
      1 FORMAT (4110)
      WRITE (IOUT,3)IHEDFM,IDDNFM
     3 FORMAT (1HO,'HEAD PRINT FORMAT IS FORMAT NUMBER',14,
      1 ' DRAWDOWN PRINT FORMAT IS FORMAT NUMBER',14)
      WRITE (IOUT,4)IHEDMUN,IDDNUN
     4 FORMAT (1HO,'HEADS WILL BE SAVED ON UNIT',13,
      1 ' DRAWDOWNS WILL BE SAVED ON UNIT',13)
      WRITE(IOUT,561)
      561 FORMAT(1HO,'OUTPUT CONTROL IS SPECIFIED EVERY TIME STEP')
      GO TO 1000
C
C3------OUTPUT CONTROL IS INACTIVE. PRINT A MESSAGE LISTING DEFAULTS.
   600 WRITE(IOUT,641)
      641 FORMAT(1HO,'DEFAULT OUTPUT CONTROL -- THE FOLLOWING OUTPUT',
      1 ' COMES AT THE END OF EACH STRESS PERIOD: ')
      WRITE(IOUT,642)
      642 FORMAT(1X,'TOTAL VOLUMETRIC BUDGET')
      WRITE(IOUT,643)
      643 FORMAT(1X,10X,'HEAD')
      IF(ISTRT.NE.0) WRITE(IOUT,644)
      644 FORMAT(1X,10X,'DRAWDOWN')
C
C4------SET THE FORMAT CODES EQUAL TO THE DEFAULT FORMAT.
   IHEDFM=0
   IDDNFM=0
C
C5------SET DEFAULT FLAGS IN IOFLG SO THAT HEAD (AND DRAWDOWN) IS
C5------PRINTED FOR EVERY LAYER.
   ID=0
   IF(ISTRT.NE.0) ID=1
   670 DO 680 K=1,NLAY
      IOFLG(K,1)=1
      IOFLG(K,2)=ID
      IOFLG(K,3)=0
      IOFLG(K,4)=0
   680 CONTINUE
   GO TO 1000
C
C6------RETURN
   1000 RETURN
END
## List of Variables for Module SBASII

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Module</td>
<td>Flag to show if STRT was saved (one means yes; zero means no).</td>
</tr>
<tr>
<td>IDDNUM</td>
<td>Package</td>
<td>Code for format in which drawdown should be printed.</td>
</tr>
<tr>
<td>IDDNUM</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing drawdown should be recorded.</td>
</tr>
<tr>
<td>IHEDFM</td>
<td>Package</td>
<td>Code for format in which head should be printed.</td>
</tr>
<tr>
<td>IHEDUN</td>
<td>Package</td>
<td>Unit number on which an unformatted record containing head should be recorded.</td>
</tr>
<tr>
<td>INOC</td>
<td>Package</td>
<td>Unit number from which input to output control option will be read.</td>
</tr>
</tbody>
</table>
| IOFLAG   | Package | DIMENSION (NLAY,4), Flags to control printing and recording of head and drawdown for each layer.  
            |                     | (NLAY,1) ≠ 0, heads will be printed.                                         |
            |                     | (NLAY,2) ≠ 0, drawdown will be printed.                                     |
            |                     | (NLAY,3) ≠ 0, heads will be recorded.                                      |
            |                     | (NLAY,4) ≠ 0, drawdown will be recorded.                                    |
| IOUT     | Global  | Primary unit number for all printed output. IOUT = 6.                      |
| ISTRT    | Package | Flag.  
            |                     | ≠ 0, starting heads will be saved so that drawdown can be calculated.        |
            |                     | = 0, starting heads will not be saved.                                     |
| K        | Module  | Index for layers                                                          |
| NLAY     | Global  | Number of layers in the grid.                                             |
Submodule SBAS1T prints a time summary which consists of the time-step length and the elapsed time in seconds, minutes, hours, days, and years. The program can use any consistent set of time units. However, the user is given the option to specify the time units that he is using and the program converts those units to all other convenient units. The user specifies time units (ITMUNI) in module BAS1DF.

1. Use the time-unit indicator (ITMUNI) to determine the conversion factor (CNV) needed to convert time to seconds.

2. If the conversion factor is equal to zero, nonstandard time units are being used.
   
   (a) Print the time-step length and the elapsed time in the nonstandard units.
   
   (b) RETURN.

3. Calculate the length of the time step and the elapsed times in seconds.

4. Calculate the time-step length and the elapsed times in minutes, hours, days, and years.

5. Print the time-step length and the elapsed times in all time units.

6. RETURN.
SUBROUTINE SBASIT(KSTP,KPER,DELT,PERTIM,TOTIM,ITMUNI,IOUT)

C
C
C-----VERSION 0837 09APR1982 SBASIT
C ******************************************************************
C SPECIAL SPECIFICATIONS:
C ------------------------------------------------------------------
C ***-----------------------------------------------------------------
C WRITE(IOUT,199) KSTP,KPER
199 FORMAT(110,//10X,'TIME SUMMARY AT END OF TIME STEP',I3,
1   ' IN STRESS PERIOD',I3)
C
C1-----USE TIME UNIT INDICATOR TO GET FACTOR TO CONVERT TO SECONDS.
CNV=0.
   IF(ITMUNI.EQ.1) CNV=1.  
   IF(ITMUNI.EQ.2) CNV=60.  
   IF(ITMUNI.EQ.3) CNV=3600.  
   IF(ITMUNI.EQ.4) CNV=86400.  
   IF(ITMUNI.EQ.5) CNV=31557600.
C
C2-----IF FACTOR=0 THEN TIME UNITS ARE NON-STANDARD.
   IF(CNV.NE.0.) GO TO 100
C
C2A-----PRINT TIMES IN NON-STANDARD TIME UNITS.
   WRITE(IOUT,301) DELT,PERTIM,TOTIM
301 FORMAT(21X,' TIME STEP LENGTH=',G15.6/
1   21X,' STRESS PERIOD TIME=',G15.6/
2   21X,'TOTAL SIMULATION TIME=',G15.6)
C
C2B-----RETURN
   RETURN
C
C3-----CALCULATE LENGTH OF TIME STEP & ELAPSED TIMES IN SECONDS.
   DELSEC=CNV*DELT
   TOTSEC=CNV*TOTIM
   PERSEC=CNV*PERTIM
C
C4-----CALCULATE TIMES IN MINUTES,HOURS,DAYS AND YEARS.
   DELMN=DELSEC/60.
   DELHR=DELMN/60.
   DELDY=DELHR/24.
   DELYR=DELDY/365.25
   TOTMN=TOTSEC/60.
   TOTHR=TOTMN/60.
   TOTDY=TOTHr/24.
   TOTYR=TOTDY/365.25
   PERMN=PERSEC/60.
   PERHR=PERMN/60.
   PERDY=PERHR/24.
   PERYR=PERDY/365.25
C
C5-----PRINT TIME STEP LENGTH AND ELAPSED TIMES IN ALL TIME UNITS.
   WRITE(IOUT,200)
200 FORMAT(72X,' SECONDS MINUTES HOURS',10X,
1   'DAYS YEARS'/72X,5X,5G15.6)
   WRITE(IOUT,201) DELSEC,DELMN,DELHR,DELDY,DELYR
201 FORMAT(1X,' TIME STEP LENGTH',5X,5G15.6)
   WRITE(IOUT,202) PERSEC,PERMN,PERHR,PERDY,PERYR
202 FORMAT(1X,' STRESS PERIOD TIME',5X,5G15.6)
   WRITE(IOUT,203) TOTSEC,TOTMN,TOTHR,TOTDY,TOTYR
203 FORMAT(1X,'TOTAL SIMULATION TIME',5X,5G15.6)
C
C6-----RETURN
   RETURN
END
### List of Variables for Module SBASLT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNV</td>
<td>Module</td>
<td>Factor to convert elapsed time from units, specified by the user, to seconds.</td>
</tr>
<tr>
<td>DELDY</td>
<td>Module</td>
<td>Length of the time step in days.</td>
</tr>
<tr>
<td>DELHR</td>
<td>Module</td>
<td>Length of the time step in hours.</td>
</tr>
<tr>
<td>DELMN</td>
<td>Module</td>
<td>Length of the time step in minutes.</td>
</tr>
<tr>
<td>DELSEC</td>
<td>Module</td>
<td>Length of the time step in seconds.</td>
</tr>
<tr>
<td>DELT</td>
<td>Module</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>DELYR</td>
<td>Module</td>
<td>Length of the time step in years.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ITMUNI</td>
<td>Package</td>
<td>Code for time units for this problem:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - undefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - years</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>PERDY</td>
<td>Module</td>
<td>Elapsed time in the stress period in days.</td>
</tr>
<tr>
<td>PERHR</td>
<td>Module</td>
<td>Elapsed time in the stress period in hours.</td>
</tr>
<tr>
<td>PERMN</td>
<td>Module</td>
<td>Elapsed time in the stress period in minutes.</td>
</tr>
<tr>
<td>PERSEC</td>
<td>Module</td>
<td>Elapsed time in the stress period in seconds.</td>
</tr>
<tr>
<td>PERTIM</td>
<td>Package</td>
<td>Elapsed time during the current stress period.</td>
</tr>
<tr>
<td>PERYR</td>
<td>Module</td>
<td>Elapsed time in the stress period in years.</td>
</tr>
<tr>
<td>TOTDY</td>
<td>Module</td>
<td>Elapsed time in the simulation in days.</td>
</tr>
<tr>
<td>TOTHR</td>
<td>Module</td>
<td>Elapsed time in the simulation in hours.</td>
</tr>
<tr>
<td>TOTIM</td>
<td>Package</td>
<td>Elapsed time in the simulation.</td>
</tr>
<tr>
<td>TOTMN</td>
<td>Module</td>
<td>Elapsed time in the simulation in minutes.</td>
</tr>
<tr>
<td>TOTSEC</td>
<td>Module</td>
<td>Elapsed time in the simulation in seconds.</td>
</tr>
<tr>
<td>TOTYR</td>
<td>Module</td>
<td>Elapsed time in the simulation in years.</td>
</tr>
</tbody>
</table>
Module SBAS1V calculates and prints the overall volumetric budget. The individual entries for the budget, which are calculated by the budget modules in each of the component-of-flow packages, are passed to this module in a table named VBVL.

Each entry in VBVL corresponds to a component-of-flow. It consists of four values: rate of inflow for the current time step, rate of outflow for the current time step, accumulated volume of inflow since the beginning of the simulation, and accumulated volume of outflow since the beginning of the simulation. In this module, the total of all inflow rates (TOTRIN), outflow rates (TOTROT), inflow-accumulated volumes (TOTVIN), and outflow-accumulated volumes (TOTVOT) are calculated. The percent differences between those totals are also calculated and printed. The labels for the entries are supplied by the budget modules in the component-of-flow packages and passed in the table VBNM.

Module SBAS1V performs its functions in the following order:

1. Use the counter MSUM to determine the number of individual budget terms (MSUM1).

2. Clear the four accumulators for rates and volumes. The accumulators are total rate into the system (TOTRIN), total rate out of the system (TOTROT), accumulated volume into the system (TOTVIN), and accumulated volume out of the system (TOTVOT).

3. For each source or sink, add the budget entries (rates and volumes), calculated by the budget modules, to the accumulators.

4. Print the number of the time step and stress period.

5. Print the individual input rates and volumes and their totals.

6. Print the individual output rates and volumes and their totals.

7. Calculate the difference between flow into and out of the simulated-flow system. Calculate the percent difference between input and output rates (100*(TOTRIN-TOTROT)/((TOTRIN+TOTROT)/2)). Calculate the percent difference between input and output accumulated volumes (100*(TOTVIN-TOTVOT)/((TOTVIN+TOTVOT)/2)).

8. Print the differences and percent differences between input and output rates and volumes.

9. RETURN.
The inflow and outflow rates for the current time step and accumulated volumes since the beginning of the simulation for each budget entry (component-of-flow) are added to the four accumulators to obtain the total inflow and outflow rates for the current time step and the total accumulated volume of flow in and volume of flow out since the start of the simulation.
SUBROUTINE SBASlV(MSUM, VBNM, VBVL, KSTP, KPER, IOUT)

C
C******VERSION 1531 12MAY1987 SBASlV
C
C *********************************************************
C PRINT VOLUMETRIC BUDGET
C *********************************************************
C
C SPECIFICATIONS:
CHARACTER*4 VBNM
DIMENSION VBNM(4,20), VBVL(4,20)
C
C1------DETERMINE NUMBER OF INDIVIDUAL BUDGET ENTRIES.
MSUM1=MSUM-1
IF(MSUM1.LE.0) RETURN
C
C2------CLEAR RATE AND VOLUME ACCUMULATORS.
TOTRIN=0.
TOTROT=0.
TOTVIN=0.
TOTVOT=0.
C
C3------ADD RATES AND VOLUMES (IN AND OUT) TO ACCUMULATORS.
DO 100 L=1,MSUM1
  TOTRIN=TOTRIN+VBVL(3,L)
  TOTROT=TOTROT+VBVL(4,L)
  TOTVIN=TOTVIN+VBVL(1,L)
  TOTVOT=TOTVOT+VBVL(2,L)
100 CONTINUE
C
C4------PRINT TIME STEP NUMBER AND STRESS PERIOD NUMBER.
WRITE(IOUT,260) KSTP, KPER
WRITE(IOUT,265)
C
C5------PRINT INDIVIDUAL INFLOW RATES AND VOLUMES AND THEIR TOTALS.
DO 200 L=1,MSUM1
  WRITE(IOUT,275) (VBNM(I,L),I=1,4), VBVL(1,L), (VBNM(I,L),I=1,4),
  VBVL(3,L)
200 CONTINUE
WRITE(IOUT,286) TOTVIN, TOTRIN
C
C6------PRINT INDIVIDUAL OUTFLOW RATES AND VOLUMES AND THEIR TOTALS.
WRITE(IOUT,287)
DO 250 L=1,MSUM1

4-68
WRITE(IOUT,275) (VBNM(I,L),I=1,4),VBVL(2,L),(VBNM(I,L),I=1,4)
1,VDVL(4,L)
250 CONTINUE
WRITE(IOUT,298) TOTVOT,TOTROT
C
C7------CALCULATE THE DIFFERENCE BETWEEN INFLOW AND OUTFLOW.
C
C7A------CALCULATE DIFFERENCE BETWEEN RATE IN AND RATE OUT.
DIFFR=TOTRIN-TOTROT
C
C7B------CALCULATE PERCENT DIFFERENCE BETWEEN RATE IN AND RATE OUT.
PDIFFR=100.*DIFFR/((TOTRIN+TOTROT)/2)
C
C7C------CALCULATE DIFFERENCE BETWEEN VOLUME IN AND VOLUME OUT.
DIFFV=TOTVIN-TOTVOT
C
C7D------GET PERCENT DIFFERENCE BETWEEN VOLUME IN AND VOLUME OUT.
PDIFFV=100.*DIFFV/((TOTVIN+TOTVOT)/2)
C
C8------PRINT DIFFERENCES AND PERCENT DIFFERENCES BETWEEN INPUT
C8------AND OUTPUT RATES AND VOLUMES.
WRITE(IOUT,299) DIFFV,DIFFR
WRITE(IOUT,300) PDIFFV,PDIFFR
C
C9------RETURN
RETURN
C
C7-----FORMATS
C
260 FORMAT(1H0,,/'30X,'VOLUMETRIC BUDGET FOR ENTIRE MODEL AT END OF'
1,' TIME STEP',I3,' IN STRESS PERIOD',I3/30X,77(/'-'))
265 FORMAT(1H0,19X,'CUMULATIVE VOLUMES',6X,'L**3',37X
1,'RATES FOR THIS TIME STEP',6X,'L**3/T',18('=',),47X,24('=',)
2//26X,'IN:',68X,'IN:',26X,18('=',),68X,18('=',)
275 FORMAT(1X,18X,14A4,'=',G14.5,39X,14A4,'=',G14.5)
286 FORMAT(1H0,26X,'TOTAL IN =',G14.5,47X,'TOTAL IN ='
1,G14.5)
287 FORMAT(1H0,24X,'OUT:',1,67X,'OUT:',4('=',),67X,18('=',)
298 FORMAT(1H0,25X,'TOTAL OUT =',G14.5,46X,'TOTAL OUT ='
1,G14.5)
299 FORMAT(1H0,26X,'IN - OUT =',G14.5,47X,'IN - OUT =',G14.5)
300 FORMAT(1H0,15X,1PERCENT DISCREPANCY =',F20.2
1,30X,1PERCENT DISCREPANCY =',F20.2,///)
C
END
### List of Variables for Module SBASIV

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIFFR</td>
<td>Module</td>
<td>Sum of all inflow rates minus sum of all outflow rates (TOTRIN - TOTROT).</td>
</tr>
<tr>
<td>DIFFV</td>
<td>Module</td>
<td>Sum of all inflow volumes minus sum of all outflow volumes (TOTVIN - TOTVOT).</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for individual flows.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for the budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>MSUM1</td>
<td>Module</td>
<td>MSUM - 1.</td>
</tr>
<tr>
<td>PDIFFR</td>
<td>Module</td>
<td>Percent difference between the rate in and rate out.</td>
</tr>
<tr>
<td>PDIFFV</td>
<td>Module</td>
<td>Percent difference between the volume in and volume out.</td>
</tr>
<tr>
<td>TOTRIN</td>
<td>Module</td>
<td>Accumulator for the total of all inflow rates.</td>
</tr>
<tr>
<td>TOTROT</td>
<td>Module</td>
<td>Accumulator for the total of all outflow rates.</td>
</tr>
<tr>
<td>TOTVIN</td>
<td>Module</td>
<td>Accumulator for the total of all inflow volumes.</td>
</tr>
<tr>
<td>TOTVOT</td>
<td>Module</td>
<td>Accumulator for the total of all outflow volumes.</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For flow component N, the values in VBVL are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1,N) Rate for the current time step into the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(2,N) Rate for the current time step out of the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,N) Volume into the flow field during simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4,N) Volume out of the flow field during simulation.</td>
</tr>
</tbody>
</table>
The Block-Centered Flow (BCF) Package computes the conductance components of the finite-difference equation which determine flow between adjacent cells. It also computes the terms that determine the rate of movement of water to and from storage. To make the required calculations, it is assumed that a node is located at the center of each model cell; thus the name Block-Centered Flow is given to the package.

In Chapter 2, the equation of flow for each cell in the model was developed as

\[
CV_{i,j,k-1/2}h_{i,j,k-1/2} + CC_{i-1/2,j,k}h_{i-1,j,k} + CR_{i,j-1/2,k}h_{i,j-1,k} + RHS_{i,j,k} = 0
\]

The CV, CR, and CC coefficients are conductances between nodes—sometimes called "branch conductances." The HCOF and RHS coefficients are composed of external source terms and storage terms. Besides calculating the conductances and storage terms, the BCF Package calculates flow-correction terms that are added to HCOF and RHS when an underlying aquifer becomes partially unsaturated. Under these conditions the flow to the underlying aquifer no longer increases in proportion to the head difference between aquifers, but rather reaches a constant limiting value. The additional terms correct the flow equations, in effect reducing the expressions for downward flow to correspond to this limiting value.
The following discussion of the conceptualization and implementation of the BCF package is divided into nine sections: Basic Conductance Equations, Horizontal Conductance Under Confined Conditions, Horizontal Conductance Under Water Table Conditions, Vertical Conductance Formulation, Vertical Flow Calculation Under Desaturating Conditions, Storage Formulation, Storage Term Conversion, Applicability and Limitations of Optional Formulations and Data Requirements.

Basic Conductance Equations

The concept of hydraulic conductance was introduced in Chapter 2 (equation (9)). It is reviewed here and extended to cover the calculation of equivalent conductance for elements arranged in series.

Conductance is a combination of several parameters used in Darcy's law. Darcy's law defines one-dimensional flow in a prism of porous material (figure 23) as

\[ Q = KA(h_2 - h_1)/L \]  \hspace{1cm} (30)

where

- \( Q \) is the flow (L³t⁻¹);
- \( K \) is the hydraulic conductivity of the material in the direction of flow (Lt⁻¹);
- \( A \) is the cross-sectional area perpendicular to the flow (L²);
- \( h_2 - h_1 \) is the head differences across the prism parallel to flow (L); and
- \( L \) is the length of the flow path (L).

Conductance, \( C \), is defined as

\[ C = KA/L. \]  \hspace{1cm} (31)
Explanation

K is Hydraulic Conductivity

\( h_2 \) is the Head at the Left End of the Prism

\( h_1 \) is the Head at the Right End of the Prism

Q is the Flow Rate from the Left End to the Right End

L is the Length of the Flow Path

A is the Cross Sectional Area Perpendicular to the Direction of Flow

Figure 23.—Prism of porous material illustrating Darcy's law.
Therefore, Darcy's law can be written as

\[ Q = C(h_2 - h_1). \]  \hspace{1cm} (32)

Another form of the conductance definition for horizontal flow in a prism is

\[ C = \frac{TW}{L} \]  \hspace{1cm} (33)

where

- \( T \) is transmissivity (K times thickness of the prism) in the direction of flow (L^2t^{-1}); and
- \( W \) is the width of the prism (L).

Conductance is defined for a particular prism of material and for a particular direction. In an anisotropic medium characterized by three principal directions of hydraulic conductivity, the conductances of a prism in these three principal directions will generally differ.

If a prism of porous material consists of two or more subprisms in series—that is, aligned sequentially in the direction of flow, as shown in figure 24—and the conductance of each subprism is known, a conductance representing the entire prism can be calculated. The equivalent conductance for the entire prism is the rate of flow in the prism divided by the head change across the prism.

\[ C = \frac{Q}{(h_A - h_B)} \]  \hspace{1cm} (34)

Assuming continuity of head across each section in series gives the identity

\[ \sum_{i=1}^{n} \Delta h_i = h_A - h_B. \]  \hspace{1cm} (35)

Substituting for head change across each section using Darcy's law (equation (32)) gives

\[ \sum_{i=1}^{n} \frac{q_i}{C_i} = h_A - h_B. \]  \hspace{1cm} (36)
\[ \frac{1}{C} = \frac{1}{C_1} + \frac{1}{C_2} + \frac{1}{C_3} + \cdots + \frac{1}{C_n} \]

Explanation

- \( Q \) is the Flow Rate
- \( C_m \) is Conductance of Prism \( m \)
- \( h_m \) is Head at the Right Side of Prism \( m \)
- \( \Delta h_m \) is the Head Change Across Prism \( m \)
- \( C \) is the Conductance of the Entire Prism

Figure 24.—Calculation of conductance through several prisms in series.
Since flow is one-dimensional and we are assuming no accumulation or depletion in storage, all \( q_i \) are equal to the total flow \( Q \); therefore,

\[
Q \sum_{i=1}^{n} \frac{1}{C_i} = h_A - h_B \quad \text{and} \quad \sum_{i=1}^{n} \frac{1}{C_i} = Q \sum_{i=1}^{n} \frac{1}{C_i}.
\]  

(37)

By comparison with equation (34), it can be seen that

\[
\frac{1}{C} = \sum_{i=1}^{n} \frac{1}{C_i}.
\]

(38)

Thus for a set of conductances arranged in series, the inverse of the equivalent conductance equals the sum of the inverses of the individual conductances. When there are only two sections, the equivalent conductance reduces to

\[
C = \frac{C_1 C_2}{C_1 + C_2}.
\]

(39)

Horizontal Conductance Under Confined Conditions

The finite-difference equations presented in this report use equivalent conductances between nodes of adjacent cells—i.e., "branch conductances,"—rather than conductances defined within individual cells. The horizontal conductance terms, \( CR \) and \( CC \) of equation (29), are calculated between adjacent horizontal nodes. \( CR \) terms are oriented along rows and thus specify conductance between two nodes in the same row. Similarly, \( CC \) terms specify conductance between two nodes in the same column. To designate conductance between nodes, as opposed to conductance within a cell, the subscript notation "1/2" is used. For example, \( CR_{i,j+1/2,k} \) represents the conductance between nodes \( i,j,k \) and \( i,j+1,k \).

Figure 25 illustrates two cells along a row, and the parameters used to calculate the conductance between nodes. Two assumptions are made: (1)
\[
\frac{1}{CR_{i,j+\frac{1}{2},k}} = \frac{1}{TR_{i,j,k} \cdot DELC_i} \left( \frac{DELR_1}{2} \right) + \frac{1}{TR_{i,j+1,k} \cdot DELC_i} \left( \frac{DELR_{i+1}}{2} \right)
\]

\[
CR_{i,j+\frac{1}{2},k} = 2 \cdot DELC_i \times \frac{TR_{i,j,k} \cdot TR_{i,j+1,k}}{TR_{i,j,k} \cdot DELR_{j+1} + TR_{i,j+1,k} \cdot DELR_j}
\]

**Explanation**

- \(TR_{i,j,k}\) is Transmissivity in the Row Direction in Cell \(i,j,k\)
- \(CR_{i,j+\frac{1}{2},k}\) is Conductance in the Row Direction Between Nodes \(i,j,k\) and \(i,j+1,k\)

Figure 25.—Calculation of conductance between nodes using transmissivity and dimensions of cells.
the nodes are in the center of the cells and (2) the transmissivity is uniform over each cell. Thus the conductance between the nodes is the equivalent conductance of two half cells in series \((C_1\) and \(C_2\)). Applying equation (39) gives

\[
\omega_{i,j+1/2,k} = \frac{C_1C_2}{(C_1 + C_2)}.
\]

Substituting the conductance for each half cell from equation (33) gives

\[
\omega_{i,j+1/2,k} = \frac{\omega_{i,j,k} \Delta L_{C_i}}{1/2 \Delta L_{R_j}} \frac{\omega_{i,j+1,k} \Delta L_{C_i}}{1/2 \Delta L_{R_{j+1}}}
\]

where

\[
\triangle R_i\text{ is transmissivity in the row direction (L}^2t^{-1})\text{;}
\]

\[
\Delta L_{R_i}\text{ is the grid width along a row (L); and}
\]

\[
\Delta L_{C_i}\text{ is the grid width along a column (L).}
\]

\(\Delta R_i\) and \(\Delta L_{C_i}\) are identical to the terms \(\Delta r\) and \(\Delta c\), respectively, which were introduced in figure 4 and equation (3), Chapter 2. The new notation is introduced here to conform to the input of the Block-Centered Flow Package.

Simplification of the above expression gives the final equation

\[
\omega_{i,j+1/2,k} = 2 \frac{\omega_{i,j,k} \Delta L_{C_i}}{\Delta R_i} \frac{\omega_{i,j+1,k} \Delta L_{C_i}}{\Delta R_{j+1}} + \omega_{i,j+1,k} \Delta L_{R_j}.
\]
The same process can be applied to the calculation of \( \text{CCI}_{i+1/2, j, k} \) giving

\[
\text{CCI}_{i+1/2, j, k} = 2 \frac{\text{DELR}_j \text{CCI}_{i, j, k} \text{CCI}_{i+1, j, k}}{\text{TC}_{i, j, k} \text{DELC}_{i+1} + \text{TC}_{i+1, j, k} \text{DELC}_i}
\]

(42)

where \( \text{TC} \) is the transmissivity in the column direction (\( L^2 L^{-1} \)). Equations (41) and (42) are used in the BCF Package to calculate the horizontal conductances between nodes within each layer of the model. However, where the transmissivity of both cells is zero, the conductance between the nodes in the cells is set equal to zero without invoking the equations.

Horizontal Conductance Under Water Table Conditions

In a model layer which is confined, horizontal conductance will be constant for the simulation. If a layer is unconfined or potentially unconfined, new values of horizontal conductance must be calculated as the head fluctuates. This is done at the start of each iteration. First, transmissivity is calculated as the product of hydraulic conductivity and saturated thickness; then conductance is calculated from transmissivity and cell dimensions using equations (41) and (42).

Transmissivity within a cell in the row direction is calculated using one of the following three equations

\[
\text{if } H_{\text{NEW}}_{i, j, k} \geq \text{TOP}_{i, j, k}, \\
\quad \text{then } \text{TR}_{i, j, k} = (\text{TOP}_{i, j, k} - \text{BOT}_{i, j, k}) \text{HYR}_{i, j, k}; \\
\text{if } \text{TOP}_{i, j, k} > H_{\text{NEW}}_{i, j, k} > \text{BOT}_{i, j, k}, \\
\quad \text{then } \text{TR}_{i, j, k} = (H_{\text{NEW}}_{i, j, k} - \text{BOT}_{i, j, k}) \text{HYR}_{i, j, k}; \\
\text{if } \text{TOP}_{i, j, k} < H_{\text{NEW}}_{i, j, k}, \\
\quad \text{then } \text{TR}_{i, j, k} = (\text{TOP}_{i, j, k} - H_{\text{NEW}}_{i, j, k}) \text{HYR}_{i, j, k};
\]

(43) \quad (44)
if \( HNEW_{i,j,k} \leq BOT_{i,j,k} \),
\[
    \text{then } TR_{i,j,k} = 0
\]

where

\[ HYR_{i,j,k} \] is the hydraulic conductivity of cell \( i,j,k \) in the row direction (Lt-l); (this notation is introduced here to conform to the input of the Block-Centered Flow Package);

\[ TOP_{i,j,k} \] is the elevation of the top of cell \( i,j,k \) (L); and

\[ BOT_{i,j,k} \] is the elevation of the bottom of cell \( i,j,k \) (L).

Transmissivity in the column direction is the product of transmissivity in the row direction and a horizontal anisotropy factor specified by the user; the horizontal anisotropy factor is a constant for each layer. Conductances in each direction are calculated from transmissivity and cell dimensions. When head drops below the aquifer bottom (equation (45)), the cell is considered to be dewatered, and is permanently set to no flow; the model has no provision for the resaturation of a dewatered cell. Thus errors may arise in attempts to simulate situations in which actual reversals in water-level occur. Errors can also arise if oscillations of computed heads occur during iteration; if such computational oscillations cause head to drop erroneously below the bottom of the cell, the cell will change to no flow for all succeeding iterations and time steps. As a means of controlling this problem, the iterative solvers contain provisions for slowing the rate of convergence.

In the program described herein a layer-type flag, LAYCON, is used to specify whether or not the simulation of water table conditions through equations (43)-(45) is to be invoked. This is discussed more fully in the section on data requirements.
Vertical Conductance Formulation

Vertical conductance terms are calculated within the model using data from an input array which incorporates both thickness and vertical hydraulic conductivity in a single term, and using horizontal (or map) areas calculated from cell dimensions. In general, the vertical interval between two nodes, $i,j,k$ and $i,j,k+1$, may be considered to contain $n$ geohydrologic layers or units, having vertical hydraulic conductivities $K_1, K_2 \ldots K_n$ and thicknesses $\Delta z_1, \Delta z_2 \ldots \Delta z_n$. The map area of the cells around nodes $i,j,k$ and $i,j,k+1$ is $\Delta R_j \ast \Delta L_i$; the vertical conductance of an individual geohydrologic layer, $g$, in this area is given by

$$C_g = \frac{K_g \Delta R_j \ast \Delta L_i}{\Delta z_g} \quad (46)$$

The equivalent vertical conductance, $C_{i,j,k+1/2}$, for the full vertical interval between nodes $i,j,k$ and $i,j,k+1$ is found by treating the $n$ individual geohydrologic layers as conductances in series; this yields

$$\frac{1}{C_{i,j,k+1/2}} = \sum_{g=1}^{n} \frac{1}{C_g} = \sum_{g=1}^{n} \frac{1}{K_g \Delta R_j \ast \Delta L_i / \Delta z_g}$$

$$\frac{1}{C_{i,j,k+1/2}} = \frac{1}{\Delta R_j \ast \Delta L_i} \sum_{g=1}^{n} \frac{\Delta z_g}{K_g} \quad (47)$$

rearranging equation (47)

$$\frac{C_{i,j,k+1/2}}{\Delta R_j \ast \Delta L_i} = \frac{1}{\sum_{g=1}^{n} \frac{\Delta z_g}{K_g}} \quad (48)$$
The quantity \( \frac{C_{i,j,k+1/2}}{D_{i,j,k+1/2}} \) has been termed the "vertical leakance" and is designated \( V_{i,j,k+1/2} \) in this report; thus we have

\[
V_{i,j,k+1/2} = \frac{1}{\sum_{g=1}^{n} \frac{\Delta z_g}{k_g}}
\]  

(49)

\( V_{i,j,k+1/2} \) is the term actually used as input in the model described herein. That is, rather than specifying a total thickness and an equivalent (or harmonic mean) vertical hydraulic conductivity for the interval between node \( i,j,k \) and node \( i,j,k+1 \), the user specifies the term \( V_{i,j,k+1/2} \), which is actually the conductance of the interval divided by the cell area, and as such incorporates both hydraulic conductivity and thickness. The program multiplies \( V_{i,j,k+1/2} \) by cell area to obtain vertical conductance. The values of \( V_{i,j,k+1/2} \) must be calculated or determined externally to the program; this is generally done through an application of equation (49). The \( V_{i,j,k+1/2} \) values are actually read as the elements of a two-dimensional input array, \( V_{i,j} \), for each layer. Each value of \( V_{i,j,k+1/2} \) is the vertical leakance for the interval between cell \( i,j,k \) and cell \( i,j,k+1 \)—that is, for the interval between the layer for which the array is read, and the layer below it. It follows that the \( V_{i,j} \) array is not read for the lowermost layer in the model. Although values of \( V_{i,j,k+1/2} \) are thus read into the model through a series of two-dimensional input arrays, the discussion in this section will continue to be given in terms of three-dimensional array notation, \( V_{i,j,k+1/2} \), to emphasize the fact that the \( V_{i,j,k+1/2} \) values refer to the intervals between layers.
Figure 26 shows a situation in which nodes $i,j,k$ and $i,j,k+1$ both fall within a single hydrogeologic unit, having a vertical hydraulic conductivity $K_z i,j$ which is uniform at least within the cell area. For this case, application of equation (49) yields

$$V_{\text{cont}1,i,j,k+1/2} = \frac{K_z i,j}{\Delta z_{k+1/2}}$$  \hspace{1cm} (50)$$

where $\Delta z_{k+1/2}$ is the vertical distance between nodes, is the sum of $\frac{\Delta v_k}{2}$ and $\Delta v_{k+1}$, in which $\Delta v$ represents layer thickness as in figure 1. This situation might be found, for example, where several model layers are used to represent a single geohydrologic unit in order to provide greater vertical resolution.

Figure 27 shows a case in which two adjacent model layers are used to represent two vertically adjacent hydrogeologic units, so that nodes $i,j,k$ and $i,j,k+1$ fall at the midpoints of these geohydrologic layers. Each layer is characterized by its own value of vertical hydraulic conductivity, which is again assumed to be uniform at least over the cell area. The expression for $V_{\text{cont}}$ in this case becomes

$$V_{\text{cont}2,i,j,k+1/2} = \frac{1}{K_z i,j,k} \frac{(\Delta v_k)^2}{2} + \frac{1}{K_z i,j,k+1} \frac{(\Delta v_{k+1})^2}{2} \hspace{1cm} (51)$$

where $\Delta v_k$ is the thickness of model layer $k$ and $\Delta v_{k+1}$ is the thickness of model layer $k+1$.

$K_z i,j,k$ is the vertical hydraulic conductivity of the upper layer in cell $i,j,k$.

$K_z i,j,k+1$ is the vertical hydraulic conductivity of the lower layer in cell $i,j,k+1$. 

5-13
Figure 26.—Diagram for calculation of vertical leakance, $V_{cont}$, between two nodes which fall within a single geohydrologic unit.
Boundary between model layers (Coincident with Geohydrologic Boundary)

Figure 27.—Diagram for calculation of vertical leakance, $V_{cont}$, between two nodes located at the midpoints of vertically adjacent geohydrologic units.
If one value of $K_z$ is much smaller than the other, the term containing the larger $K_z$ value will be negligible in equation (51). Thus for this condition, only the term involving the smaller $K_z$ value need be retained in the denominator of (51).

Figure 28 shows a third situation, in which node $i,j,k$ and node $i,j,k+1$ are taken within (i.e., at the median depths of) two aquifers which are separated by a semiconfining unit. In this case, three intervals must be represented in the summation of equation (49)--the lower half of the upper aquifer, the semiconfining unit, and the upper half of the lower aquifer. The resulting expression for $V_{cont}$ is

$$V_{cont, j, k+1/2} = \frac{1}{\Delta Z_U/2 + \Delta Z_C + \Delta Z_L/2} \frac{\Delta Z_U/2}{K_{zu}} + \frac{\Delta Z_C}{K_{zc}} + \frac{\Delta Z_L/2}{K_{zL}}$$

(52)

where $\Delta Z_U$ is the thickness of the upper aquifer

$\Delta Z_C$ is the thickness of the confining bed

$\Delta Z_L$ is the thickness of the lower aquifer

$K_{zu}$ is the vertical hydraulic conductivity of the upper aquifer

$K_{zc}$ is the vertical hydraulic conductivity of the semiconfining unit

$K_{zL}$ is the vertical hydraulic conductivity of the lower aquifer; and each of these terms must in general be considered to vary with the map location $(i,j)$ of the nodes. In many applications it turns out that $K_{zc}$ is much smaller than either $K_{zu}$ or $K_{zL}$; in these situations the terms involving $K_{zu}$ and $K_{zL}$ are negligible in equation (52) so that the expression for $V_{cont}$ becomes

$$V_{cont, j, k+1/2} = \frac{K_{zc}}{\Delta Z_C}$$

(53)
Figure 28.—Diagram for calculation of vertical leakance, \( V_{\text{cont}} \), between two nodes located at the midpoints of aquifers which are separated by a semiconfining unit.

\[
\frac{1}{C_{\text{eq}}} = \frac{1}{C_u} + \frac{1}{C_c} + \frac{1}{C_L} = \frac{1}{\text{DELC}_i \times \text{DELR}_j} \left\{ \frac{\Delta Z_u/2}{K_{zu}} + \frac{\Delta Z_c}{K_{zc}} + \frac{\Delta Z_L/2}{K_{zl}} \right\}
\]

\[
V_{\text{cont}}^{i,j,k+1/2} = \frac{1}{K_{zu} + \frac{\Delta Z_c}{K_{zc}} + \frac{\Delta Z_L/2}{K_{zl}}}
\]
If the formulation of equation (53) is applied to the situation shown in figure 28, and if the further assumptions are made that the confining bed makes no measureable contribution to the horizontal conductance or the storage capacity of either model layer, then in effect model layer k represents the upper aquifer, model layer k+1 represents the lower aquifer, and the confining bed is treated simply as the vertical conductance between the two model layers. This formulation is equivalent to that of figure 12, and is frequently referred to as the "quasi-three-dimensional" approach.

In summary, the model described herein utilizes a single input array, Vcont, which incorporates both vertical hydraulic conductivity and thickness, rather than independent inputs for thickness and conductivity. The program multiplies Vcont by cell area to obtain vertical conductance. This requires the user to calculate Vcont values externally to the program, using equation (49) in the general case (where n hydrogeologic layers occur in the vertical interval between nodes) or equations (50), (51), (52) or (53) in the situations shown in figures 26-28. While this approach involves some preprocessing of input data, it actually increases the flexibility of model application. Because layer transmissivity (or hydraulic conductivity and bottom elevation if unconfined) and layer storage coefficient are also used as input terms, the model never actually reads vertical grid spacing data. Thus the model can implement either the orthogonal mesh of figure 9-b or a deformed mesh such as that of figure 9-c, and can similarly be adapted to either a direct three-dimensional simulation or to the quasi-three-dimensional formulation, without modification of the program.
The basic finite difference equation for cell $i,j,k$ (equation (24)) was given as

$$
\begin{align*}
&CR_{i,j-1/2,k}(h_{i,j-1,k}^{m} - h_{i,j,k}^{m}) + CR_{i,j+1/2,k}(h_{i,j+1,k}^{m} - h_{i,j,k}^{m}) + \\
&CC_{i-1/2,j,k}(h_{i-1,j,k}^{m} - h_{i,j,k}^{m}) + CC_{i+1/2,j,k}(h_{i+1,j,k}^{m} - h_{i,j,k}^{m}) + \\
&CV_{i,j,k-1/2}(h_{i,j,k-1}^{m} - h_{i,j,k}^{m}) + CV_{i,j,k+1/2}(h_{i,j,k+1}^{m} - h_{i,j,k}^{m}) + \\
&P_{i,j,k} - \frac{Q_{i,j,k}}{h_{i,j,k}^{m-1}} = SS_{i,j,k} \left( \frac{\Delta r_{j} \Delta c_{l}}{\Delta v_{k}} \right)
\end{align*}
$$

In this equation the term $CV_{i,j,k+1/2}(h_{i,j,k+1}^{m} - h_{i,j,k}^{m})$ gives the flow into cell $i,j,k$ through its lower face, i.e.

$$
q_{i,j,k+1/2} = CV_{i,j,k+1/2} \left( h_{i,j,k+1}^{m} - h_{i,j,k}^{m} \right)
$$

where following the convention of equation (24), a positive value of $q_{i,j,k+1/2}$ indicates flow into cell $i,j,k$ and a negative value indicates flow out of the cell. Equations (54) and (55) are based on the assumption that cells $i,j,k$ and $i,j,k+1$ are fully saturated - i.e., that the water level in each cell stands higher than the elevation of the top of the cell. There are, however, situations in which a portion of a confined aquifer may become unsaturated--for example, when drawdown due to pumpage causes water levels to fall, at least locally, below the top of the aquifer. In terms of simulation, this condition is shown in figure 29. Two aquifers separated by a confining bed are simulated using the quasi-three-dimensional approach, in which the upper aquifer is represented by cell $i,j,k$, the underlying aquifer by cell $i,j,k+1$, and the confining bed by the vertical conductance between the two layers, $CV_{i,j,k+1/2}$. Pumping from the lower layer has...
Figure 29.—Situation in which a correction is required to limit the downward flow into cell $i,j,k+1$, as a result of partial desaturation of the cell.
lowered the water level in cell $i,j,k+1$ below the elevation of the top of the cell, so that the aquifer is effectively unconfined within the cell area. An assumption is made that the confining layer remains fully saturated from top to bottom, and we consider the head difference across this confining unit. At the upper surface of the confining unit the head is simply $h_{i,j,k}$, just below the lower surface of the confining unit, however, unsaturated conditions prevail, so that the pressure sensed on the lower surface of the confining unit is atmospheric--taken as zero in the model formulation. Thus the head at the base of the confining unit is simply the elevation at that point--i.e., the elevation of the top of the lower cell. If this elevation is designated $\text{TOP}_{i,j,k+1}$, the flow through the confining bed is obtained by substituting $\text{TOP}_{i,j,k+1}$ for $h_{i,j,k+1}$ in equation (55),

$$q_{i,j,k+1/2} = CV_{i,j,k+1/2}(\text{TOP}_{i,j,k+1} - h_{i,j,k}) \quad (56)$$

Thus the flow will be downward, from cell $i,j,k$ to cell $i,j,k+1$ (i.e., following the convention of equation (26), $q_{i,j,k+1/2}$ will be negative); but under this condition the flow will no longer be dependent on the water level, $h_{i,j,k+1}$, in the lower cell. The simplest approach to this problem in formulating the equation for cell $i,j,k$ would be to substitute the flow expression of equation (56) into equation (54), in place of the expression given in (55). However, if we consider the matrix of coefficients of the entire system of finite difference equations (matrix $[A]$ of equation (27)), direct substitution of the expression in (56) into the equation for node $i,j,k$ would render this matrix unsymmetric, generating problems in the solution process. To avoid this condition, an alternative approach is used. The flow term of equation (55) is allowed to remain on the left side of equation (54). The flow into cell $i,j,k$ as computed by this term, is
\begin{align*}
CV_{i,j,k+1/2}(h_{i,j,k+1} - h_{i,j,k}) \\
\text{(where in this case, since } h_{i,j,k} > h_{i,j,k+1}, \text{ the computed flow is negative, indicating movement out of cell } i,j,k. \text{)} \text{ The "actual" flow into cell } i,j,k \text{ is given by equation } (56) \text{ as } \frac{m}{CV_{i,j,k+1/2}(TOP_{i,j,k+1} - h_{i,j,k})} \text{ (where again } h_{i,j,k} > TOP_{i,j,k+1} \text{ indicating movement out of the cell). A correction term, } q_c, \text{ can be obtained by subtracting equation } (56) \text{ from equation } (55), \text{ i.e.}
\end{align*}

\begin{align*}
q_c = (\text{computed flow into cell } i,j,k) \\
- (\text{"actual" flow into all } i,j,k) = CV_{i,j,k+1/2}(h_{i,j,k+1} - TOP_{i,j,k+1}) \\
\text{(57)}
\end{align*}

To compensate for allowing the computed flow to remain on the left side of equation (54), the term \(q_c\) is added to the right side of equation (54). In the operation of the model, equation (54), which is identical to equation (24), is rearranged to the form of equation (26); and in practice, the term \(q_c\) is added to the right side, RHS, of equation (26). This immediately introduces a difficulty, since \(q_c\) contains the term \(h_{i,j,k+1}^m\), and all terms involving unknown heads must be kept on the left side of equation (26). To circumvent this difficulty, \(q_c\) is actually computed using the value of \(h_{i,j,k+1}^m\) from the preceding iteration, rather than that from the current iteration, i.e.

\begin{align*}
q_{c,n} = CV_{i,j,k+1/2}(h_{i,j,k+1}^{m,n-1} - TOP_{i,j,k+1}) \\
\text{(58)}
\end{align*}

where \(q_{c,n}\) is the value of \(q_c\) to be added to RHS in the \(n^{th}\) iteration, and \(h_{i,j,k+1}^{m,n-1}\) is the value of \(h_{i,j,k+1}^m\) from the preceding iteration, \(n-1\). As convergence is approached the difference between \(h_{i,j,k+1}^m\) and \(h_{i,j,k+1}^m\) becomes progressively smaller, and the approximation involved in (58) thus becomes
more accurate. In the first iteration of each time step, the initial trial value of $h_{i,j,k+1}$ is used in computing $q_c$.

The process described above is used in formulating the equations for cell $i,j,k$ when the underlying cell, $i,j,k+1$, has "dewatered"—i.e., when the water level in $i,j,k+1$ has fallen below the top of the cell. A correction must also be applied in formulating the equations for the dewatered cell itself. To examine this correction, we now take cell $i,j,k$ to be the dewatered cell, and we consider flow into $i,j,k$ from the overlying cell, $i,j,k-1$. For this case, the computed flow into cell $i,j,k$ from above is $CV_{i,j,k-1/2}(h_{i,j,k-1} - h_{i,j,k})$ whereas the "actual" flow into the cell is $CV_{i,j,k-1/2}(m_{TOP_{i,j,k}} - h_{i,j,k})$. The difference, computed minus "actual" flow, is thus $q'_c = CV_{i,j,k-1/2}(m_{TOP_{i,j,k}} - h_{i,j,k})$ where $q'_c$ should be added to the right hand side of equation (54) or (26). From a programming point of view, the most efficient way to handle this correction is to add the term $CV_{i,j,k-1/2}$ to HCOF on the left side of equation (26), while adding the term $(CV_{i,j,k-1/2} \cdot m_{TOP_{i,j,k}})$ to the RHS term. Because HCOF forms part of the coefficient of $h_{i,j,k}$, which falls on the main diagonal of the coefficient matrix, this correction does not affect the symmetry of the coefficient matrix; at the same time, the problems entailed in placing an unknown head value on the right side of the equation are avoided.

In summary, whenever dewatering of a cell occurs, two corrections must be made—one in formulating equation (26) as it applies to the overlying cell, and one in formulating equation (26) as it applies to the dewatered cell.
cell itself. These two corrections are discussed separately above, in each case using the designation \( i,j,k \) to represent the cell for which equation (26) is formulated. It is important to keep in mind, however, that both corrections are applied in any dewatering event, and that the form of the corrections has been developed to preserve the symmetry of the coefficient matrix \([A]\) of equation (27), and to maximize program efficiency.

In the program described herein, the user specifies whether or not the procedure for limiting vertical flow under dewatered conditions is to be implemented. This is done through the layer type-flag, \( \text{LAYCON} \), as discussed in the section on data requirements.

**Storage Formulation**

In the formulation of storage terms, the program described herein distinguishes between layers in which storage coefficient values remain constant throughout the simulation, and those in which the storage coefficient may "convert" from a confined value to a water table value, or vice-versa, as the water level in a cell falls below or rises above the top of the cell. This distinction is made through the use of the layer flag, \( \text{LAYCON} \), as described in the section on data requirements.

For a layer in which storage coefficient is to remain constant during the simulation, the storage formulation is based upon a direct application of the storage expression in equation (24) or (54). This expression, which applies to an individual cell, \( i,j,k \), has the form

\[
\Delta V = SS_{i,j,k} \left( \Delta r_j \Delta c_i \Delta v_k \right) \frac{h_{i,j,k} - h_{i,j,k}}{t_m - t_{m-1}} \tag{60}
\]
where $\frac{\Delta V}{\Delta t}$ is the rate of accumulation of water in the cell, and as such must appear on the right side of equation (24) or (54); $SS_{i,j,k}$ is the specific storage of the material in cell $i,j,k$; $\Delta r_i$, $\Delta c_i$ and $\Delta v_k$ are the cell dimensions; $h_{i,j,k}^m$ is the head in cell $i,j,k$ at the end of time step $m$; $h_{i,j,k}^{m-1}$ is the head in cell $i,j,k$ at the end of time step $m-1$; $t_m$ is the time at the end of time step $m$; and $t_{m-1}$ is the time at the end of time step $m-1$.

In equation (26) the notation $SCI_{i,j,k}$ was introduced, where $SCI_{i,j,k} = SS_{i,j,k}Ar_{j}Ac_{i}Av_{k}$. In this report the term $SCI_{i,j,k}$ is termed the "storage capacity" or the "primary storage capacity" of cell $i,j,k$; the "primary" designation is used to distinguish $SCI_{i,j,k}$ from a secondary storage capacity which is used when storage term conversion is invoked, as explained in the following section. Using the concept of storage capacity, the expression for rate of accumulation in storage in cell $i,j,k$ can be written

$$SCI_{i,j,k} (h_{i,j,k}^m - h_{i,j,k}^{m-1}) / (t_m - t_{m-1}) .$$

This expression is separated into two terms in equation (26), $SCI_{i,j,k} h_{i,j,k}^m / (t_m - t_{m-1})$, which is incorporated in the left side of (26) through the term $HCOF_{i,j,k}$, and $SCI_{i,j,k} h_{i,j,k}^{m-1} / (t_m - t_{m-1})$, which is included in the term $RHS_{i,j,k}$ on the right side of (26).

The input to the Block-Centered Flow Package requires specification of dimensionless storage coefficient values in each layer of the model; for a confined layer these storage coefficient values are given by the specific storage of the cell material multiplied by layer thickness in the cell, $SS_{i,j,k} Av_k$; for an unconfined layer they are equal to the specific yield of the material in the cell. The incorporation of layer thickness into the confined
storage term maintains the flexibility of the program to represent layers of varying thickness, and to implement either the direct three-dimensional or "quasi-three-dimensional" conceptualizations of vertical discretization. The storage coefficient values are read layer by layer; they are designated as array sfl in the input instructions. These values are then multiplied by the cell areas, $\Delta r_j \Delta x_i$, to create storage capacity values, and they are stored in the SC1 array.

### Storage Term Conversion

The primary storage capacity described above, $SC_{l,i,j,k}$ is adequate for simulations in which the water level in each individual cell remains either above the top of the cell or below the top of the cell throughout the course of the simulation. If the water level crosses the top of a cell during a simulation—i.e., if the water level in a confined (fully saturated) cell falls below the top of the cell as a result of simulated pumpage, or if the water level in an unconfined cell rises above the top of the cell—then in effect the system "converts" from confined to water table conditions, or vice versa, during the simulation. Where these conditions appear to be possible, the user may invoke storage term conversion for the entire layer through use of the layer-type flag. When this is done, the primary storage capacity, $SC_{l,i,j,k}$ for any cell in the layer will represent the confined storage coefficient multiplied by cell area; a secondary storage capacity, $SC_{z,i,j,k}$ is used to represent specific yield multiplied by cell area. Values of confined storage coefficient for each cell in the layer are read through the two-dimensional input array sfl. These confined storage
coefficient values are multiplied by cell areas to obtain confined storage capacities, which are stored in the array $S_{Cl}$. Values of specific yield for each cell in the layer are read through the two-dimensional input array $sf_2$. These specific yield values are multiplied by cell areas to obtain unconfined storage capacities, which are stored in array $SC_{u}$.

In a layer which has been designated for storage term conversion, the expression for rate of accumulation in storage in cell $i,j,k$ is formulated as follows

$$\frac{\Delta V}{\Delta t} = \frac{SC_{B} \left( h_{i,j,k}^m - TOP_{i,j,k} \right) + SCA \left( TOP_{i,j,k} - h_{i,j,k}^{m-1} \right)}{t_m - t_{m-1}} \quad (61)$$

where again $\frac{\Delta V}{\Delta t}$ is rate of accumulation of water in storage in cell $i,j,k$ and as such must appear on the right side of equation (24) or (54); $SCA$ is the storage capacity in effect in cell $i,j,k$ at the start of the time step; and $SC_{B}$ is the "current" storage capacity—that is, the storage capacity in effect during the iteration in process. Consider a case in which the head in cell $i,j,k$ at the beginning of time step $m$ ($h_{i,j,k}^m$) is above the top of the cell. Since there is no free surface in the cell at the start of the time step, the storage capacity at that time is taken as the confined storage capacity—that is, $SCA$ is set equal to $S_{Cl_{i,j,k}}$. If, during a given iteration for time step $m$, the computed value of head for the end of the time step ($h_{i,j,k}^m$) is found to be above the top of the cell, $SC_{B}$ for the following iteration is also set equal to $S_{Cl_{i,j,k}}$; equation (61) for that
iteration then reverts to the form of equation (60). However, if the computed value of \( h_{i,j,k} \) in a given iteration turns out to be below the top of the cell, as shown in figure 30, the value of SCB for the following iteration is set equal to SC2, the unconfined storage capacity. In this case the computed rate of release of water from storage in the time step has two components:

\[
SC1_{i,j,k} \left( \frac{\text{TOP}_{i,j,k} - h_{i,j,k}}{t_m - t_{m-1}} \right) \quad \text{the rate of release from confined or compressive storage; and}
\]

\[
SC2_{i,j,k} \left( \frac{h_{i,j,k} - \text{TOP}_{i,j,k}}{t_m - t_{m-1}} \right) \quad \text{the rate of release from water table storage.}
\]

If the head at the beginning of the time step, \( h_{i,j,k} \), is below the top of cell \( i,j,k \), so that a free surface exists within the cell, SCA in equation (61) is set equal to SC2\(_{i,j,k}\). If, during an iteration for time step \( m \), the computed value of head for the end of the time step turns out to be below the top of the cell, SCB in the subsequent iteration is also set equal to SC2\(_{i,j,k}\) and equation (61) again reverts to the form of equation (60). However, if the computed head for the end of the time step turns out to be above the top of the cell, SCB in the subsequent iteration is set equal to SC1\(_{i,j,k}\), the confined storage capacity. This situation occurs during intervals of rising water level, and again two components are computed for the rate of accumulation of water in storage—one corresponding to unconfined or water table storage and one corresponding to confined or compressive storage.

Equation (61) can be rearranged as follows

\[
\Delta V = \frac{SCB}{\Delta t} = \frac{mtm - tm - 1}{tm - tm - 1} \left( \frac{h_{i,j,k}}{tm - tm - 1} \right) - \frac{SCB \times \text{TOP}_{i,j,k}}{tm - tm - 1} + \frac{SCA \times \left( \text{TOP}_{i,j,k} - h_{i,j,k} \right)}{tm - tm - 1} \quad (62)
\]
Figure 30.—A model cell which uses two storage factors during one iteration.
\[ \frac{\Delta V}{\Delta t} \] represents rate of accumulation in storage and as such would appear on the right in equation (24) or (54). In the formulation of equation (26), therefore, the term \[ \frac{SCB}{t_m - t_{m-1}} \] is subtracted from \( HCOF_{i,j,k} \) on the left hand side, while the term \[ \frac{SCA(TOP_{i,j,k} - h_{i,j,k}) - SCB*TOP_{i,j,k}}{t_m - t_{m-1}} \] is added to \( \text{RHS}_{i,j,k} \) on the right.

**Applicability and Limitations of Optional Formulations**

The options for calculation of horizontal conductance under water table conditions, limitation of vertical flow under desaturating conditions, and storage term conversion were all developed on the assumption that each model layer corresponds to a distinct aquifer or permeable horizon, and that these horizons are separated by distinct units of low permeability. Use of these options where these conditions are not satisfied may lead to a variety of problems and inaccuracies in simulation. For example, if the option for horizontal conductance calculation under water table conditions is used where a water table aquifer is represented by several model layers, and the water table is expected to traverse more than one layer during simulation, incorrect (and irreversible) conversion of cells to a no-flow condition may occur as iterations are carried out. Thus care should be exercised in the decision to use any of the three options noted above.

**Data Requirements**

The fundamental variables controlling cell-to-cell flow and storage in the model are entered through the Block-Centered-Flow Package input. These variables, depending on the options which are invoked, may include...
transmissivity, hydraulic conductivity, specific yield, confined storage
coefficient, vertical leakance, aquifer bottom elevation and aquifer top
elevation. Each of these variables is entered using the utility array-
reader module U2DREL, which is described in Chapter 14. This module
either reads a two-dimensional array of data for a single layer, or
accepts a single value provided by the user and applies that value
throughout the array, for all cells in the layer.

The model utilizes a layer-type code to classify layers according to
the simulation options that are used. In particular, the layer-type code
indicates whether specified transmissivity values are to be used, or
transmissivities are to be calculated at each iteration as the product of
hydraulic conductivity and saturated thickness; whether storage term
conversion is to be used; and whether limitation of vertical flow from
above is to be invoked under dewatered conditions. Because the layer-type
code identifies the options to be employed in a given layer, it indicates
the kinds of data required for the layer, and thus identifies the data
arrays to be read. The data are entered layer by layer; for each layer a
set of two-dimensional arrays, one array for each required parameter, is
read in turn. That is, all of the required arrays for layer 1 are read
initially, in sequence, then all of the arrays for layer 2, and so on
until all layers have been covered. This method of data organization
provides a simpler input process than would be possible using the alternative
of a series of three-dimensional arrays corresponding to the various
parameters.

Within each layer the required parameters should be specified for
every cell, including constant-head and no-flow cells. For no-flow cells,
the entered values are never used in calculation, and thus any values may be specified; for constant head cells, the storage terms are not used but the other parameters are, and realistic values for those parameters must be entered.

Two parameters, transmissivity and hydraulic conductivity, each require the designation of two values at each cell—one in the row direction and one in the column direction. To reduce input effort, only a single array is read for each of these parameters, giving only the values in the row direction; these row-direction values are subsequently multiplied by an anisotropy factor to obtain the corresponding column-direction values. A single value of the anisotropy factor is specified by the user for each layer, through the one-dimensional array TRPY (NLAY).

Vertical leakance terms (Vcont, or $K_z/A_z$) are associated with each layer except the lowermost; the values associated with a given layer actually apply to the interval between that layer and the next lower layer. For example, the array of Vcont values entered during the input sequence for layer 1 actually applies to the interval between the midpoint of layer 1 and the midpoint of layer 2.

In addition to the terms mentioned above, the Block-Centered Flow Package input includes cell dimensions (DELR and DELC), a flag to indicate whether the simulation is transient or steady state (ISS), and a flag to indicate whether cell-by-cell flow terms are to be saved (IBCFCB). If the ISS flag is set for steady-state conditions (ISS $\neq$ 0), no space is allocated for storage coefficient or specific yield and storage calculations are skipped. Thus for steady-state runs, arrays of storage coefficients or
specific yields must not be included in the input data; if they are included, the data sequence will be misread. Note that erroneous specification of ISS or of a LAYCON value will also cause misreading of the data array sequence.

Four types of layer are recognized by the model, incorporating various combinations of the options provided by the Block-Centered-Flow Package. These four layer types are identified by their layer-type codes, which are stored in the one-dimensional array LAYCON (NLAY). The code values and the corresponding layer characteristics are given below.

Layer-type O--In this category there is no provision for modification of transmissivity as water level varies, for storage term conversion, or for limitation of vertical flow from above if water level falls below the top of the cell. This layer type is normally used to simulate confined conditions, but could also be used to simulate a layer in which unconfined conditions will always prevail, provided drawdowns are expected to be a small fraction of layer thickness and flow from the overlying layer (if present) is expected to be negligible. If the simulation is transient, storage coefficient or specific yield values are entered in the input array sf1(NCOL, NROW); then row-direction transmissivities are entered in the input array Tran (NCOL, NROW); and following the transmissivities, unless the layer is the lowermost in the model, vertical leakance values are entered in the input array Vcont (NCOL, NROW). Again, parameter values may be specified by providing the entire array, or by providing a single default value which is applied to all cells of the layer. The parameter values assigned at the beginning of a simulation in this type of layer are retained without change throughout the simulation.
Layer-type 1--This layer type is utilized only in a single-layer model or in the uppermost layer of a model, and only where unconfined conditions are expected to persist in the layer throughout the entire period of simulation. No provision is made for storage term conversion, by virtue of the assumption that water table conditions will always prevail; and no provision is made for limiting flow from above under dewatered conditions, since layer-type 1 is used only for the uppermost layer of a model. However, transmissivities are computed at each iteration as the products of hydraulic conductivity and saturated thickness values within the layer. Thus the input data includes hydraulic conductivity and cell bottom elevation, rather than transmissivity. If the simulation is indicated as transient, specific yield values are entered in the input array sfi(NCOL, NROW). Row direction hydraulic conductivity values are then entered in the input array HY(NCOL, NROW) and cell bottom elevations are entered in the array BOT(NCOL, NROW). If the model contains more than one layer, vertical leakance values are entered in the input array Vcont(NCOL, NROW). Because use of this layer type would be inappropriate except in the uppermost layer, a check of the layer number is made whenever LAYCON is given a value of one; if the layer number is not also equal to one, indicating the uppermost model layer, an error message is printed.

Layer-type 2--This layer type is used where the situation may alternate between confined and unconfined conditions, so that storage term conversion and limitation of flow from above under dewatered conditions are both desirable; but where the saturated thickness is expected to remain everywhere a high fraction of the layer thickness throughout the period of simulation,
so that recalculation of transmissivity as the product of hydraulic conductivity and saturated thickness is not necessary. The storage term conversion option requires that both a confined storage coefficient and a specific yield value be specified for each cell, and that the top elevation be specified for each cell; the top elevation is also used in the option to limit flow from above under dewatered conditions. If the simulation is transient, confined storage coefficient values are entered in the input array $s_{f1}(N_{COL}, N_{ROW})$. Transmissivity values are then entered in the array $Tran(N_{COL}, N_{ROW})$. Unless the layer is the lowermost in the model, vertical leakance values are next entered in the array $V_{cont}(N_{COL}, N_{ROW})$. Specific yield values are then entered in the array $sf2(N_{COL}, N_{ROW})$ if the simulation is transient; and finally layer top elevations are entered in the array $TOP(N_{COL}, N_{ROW})$.

Layer-Type 3--This layer type incorporates all of the Block-Centered-Flow options associated with water table conditions. Transmissivities are recalculated at each iteration using hydraulic conductivities and layer bottom elevations, and both storage term conversion and limitation of flow from above under dewatered conditions are implemented. The required data thus includes hydraulic conductivities, layer bottom elevations, confined storage coefficients (if transient), specific yields (if transient), vertical leakances and layer top elevations. Confined storage coefficients are entered in the input array $s_{f1}(N_{COL}, N_{ROW})$; hydraulic conductivity values are then entered in the array $HY(N_{COL}, N_{ROW})$, and aquifer bottom elevations in $BOT(N_{COL}, N_{ROW})$. Unless the layer is the lowermost in the model, vertical leakance values are next entered in the array $V_{cont}(N_{COL}, N_{ROW})$. Specific yield values are then entered in the array...
sf2(NCOL,NROW); and finally aquifer top elevations are entered in the array TOP(NCOL, NROW).

The input sequence is outlined in the following section. Both of the utility modules which are used are described in Chapter 14, and the required formats are illustrated in the "Sample Input to the BCF Package" and in appendix D.
Input for the Block-Centered Flow (BCF) Package is read from the unit specified in IUNIT(1).

FOR EACH SIMULATION

BCF1AL

1. Data: ISS IBCFCB
   Format: 110 110

2. Data: LAYCON(NLAY) (Maximum of 80 layers)
   Format: 4012
   (If there are 40 or fewer layers, use one record; otherwise, use two records.)

BCF1RP

3. Data: TRPY(NLAY)
   Module: U1DREL

4. Data: DELR(NCOL)
   Module: U1DREL

5. Data: DELC(NROW)
   Module: U1DREL

A subset of the following two-dimensional arrays are used to describe each layer. The arrays needed for each layer depend on the layer type code (LAYCON) and whether the simulation is transient (ISS = 0) or steady state (ISS ≠ 0). If an array is not needed, it must be omitted. All of the arrays (items 6-12) for layer 1 are read first; then all of the arrays for layer 2, etc.

IF THE SIMULATION IS TRANSIENT

6. Data: sf1(NCOL,NROW)
   Module: U2DREL

IF THE LAYER TYPE CODE (LAYCON) IS ZERO OR TWO

7. Data: Tran(NCOL,NROW)
   Module: U2DREL

IF THE LAYER TYPE CODE (LAYCON) IS ONE OR THREE

8. Data: HY(NCOL,NROW)
   Module: U2DREL

9. Data: BOT(NCOL,NROW)
   Module: U2DREL
10. Data: Vcont(NCOL,NROW)  
    Module: U2DREL

11. Data: sf2(NCOL,NROW)  
    Module: U2DREL

12. Data: TOP(NCOL,NROW)  
    Module: U2DREL

Explanation of Fields Used in Input Instructions

ISS--is the steady-state flag.

If ISS ≠ 0, the simulation is steady state.

If ISS = 0, the simulation is transient.

IBCFCB--is a flag and a unit number.

If IBCFCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL (see Output Control) is set; the terms which are saved will include cell-by-cell storage terms, cell-by-cell constant head flows, and internal cell-by-cell flows.

If IBCFCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IBCFCB < 0, flow for each constant-head cell will be printed, rather than saved on disk, whenever ICBCFL is set; cell-by-cell storage terms and internal cell-by-cell flows will neither be saved nor printed.

LAYCON--is the layer type table. Each element holds the code for the respective layer. Read one value for each layer. There is a limit of 80 layers. Leave unused elements blank.

0 - confined--Transmissivity and storage coefficient of the layer are constant for the entire simulation.

1 - unconfined--Transmissivity of the layer varies. It is calculated from the saturated thickness and hydraulic conductivity. The storage coefficient is constant; valid only for layer 1.
2 - confined/unconfined—Transmissivity of the layer is constant. The storage coefficient may alternate between confined and unconfined values. Vertical leakage from above is limited if the layer desaturates.

3 - confined/unconfined—Transmissivity of the layer varies. It is calculated from the saturated thickness and hydraulic conductivity. The storage coefficient may alternate between confined and unconfined values. Vertical leakage from above is limited if the aquifer desaturates.

TRPY—is a one-dimensional array containing an anisotropy factor for each layer. It is the ratio of transmissivity or hydraulic conductivity (whichever is being used) along a column to transmissivity or hydraulic conductivity along a row. Read one value per layer. Set to 1.0 for isotropic conditions. NOTE: This is one array with one value for each layer.

DELR—is the cell width along rows. Read one value for each of the NCOL columns.

DELC—is the cell width along columns. Read one value for each of the NROW rows.

sfl—is the primary storage coefficient. Read only for a transient simulation (steady-state flag, ISS, is 0). Note that for Laycon=1, sfl will always be specific yield, while for Laycon=2 or 3, sfl will always be confined storage coefficient. For Laycon=0, sfl would normally be confined storage coefficient; however, layer-type 0 can also be used for simulation of water table conditions where drawdowns are expected to remain everywhere a small fraction of the saturated thickness, and where there is no layer above, or flow from the layer above is negligible; and in this case specific yield values would be entered in sfl.

Tran—is the transmissivity along rows. Tran is multiplied by TRPY to obtain transmissivity along columns. Read only for layers where LAYCON is zero or two.

HY—is the hydraulic conductivity along rows. HY is multiplied by TRPY to obtain the hydraulic conductivity along columns. Read only for layers where LAYCON is one or three.

BOT—is the elevation of the aquifer bottom. Read only for layers where LAYCON is one or three.

Vcont—is the vertical hydraulic conductivity divided by the thickness from a layer to the layer beneath it. Since there is not a layer beneath the bottom layer, Vcont cannot be specified for the bottom layer.
sf2--is the secondary storage coefficient. Read it only for layers where LAYCON is two or three and only if a transient simulation (steady-state flag, ISS, is zero). The secondary storage coefficient is always specific yield.

TOP--is the elevation of the aquifer top. Read only for layers where LAYCON is two or three.
SAMPLE INPUT TO THE BCF PACKAGE

DATA ITEM

EXPLANATION

3 CONTROL RECORD FOR TRPY ARRAY

TRPY VALUES FOR LAYERS 1, 2 AND 3

4 CONTROL RECORD FOR DELC ARRAY

5 CONTROL RECORD FOR DELC ARRAY

DELC VALUES FOR EACH ROW

6 CONTROL RECORD FOR PRIMARY STORAGE FACTOR LAYER 1

8 CONTROL RECORD FOR HYDRAULIC CONDUCTIVITY LAYER 1

9 CONTROL RECORD FOR BOTTOM LAYER 1

10 CONTROL RECORD FOR VCONT LAYER 1

6 CONTROL RECORD FOR PRIMARY STORAGE FACTOR LAYER 2

8 CONTROL RECORD FOR HYDRAULIC CONDUCTIVITY LAYER 2

9 CONTROL RECORD FOR BOTTOM LAYER 2

VALUES FOR BOTTOM LAYER 2

10 CONTROL RECORD FOR VCONT LAYER 2

11 CONTROL RECORD FOR SECONDARY STORAGE FACTOR LAYER 2

12 CONTROL RECORD FOR TOP LAYER 2

6 CONTROL RECORD FOR PRIMARY STORAGE FACTOR LAYER 3

7 CONTROL RECORD FOR TRANSMISSIVITY LAYER 3

VALUES FOR TRANSMISSIVITY LAYER 3

INPUT RECORDS

5.41

FIELDS IN ARRAY CONTROL RECORDS ARE-- (LOCAT, CONST, FMTIN, IPRN)
The Block-Centered Flow Package (BCFl) has four primary modules and three submodules. The relationship of the modules to MAIN and to each other is shown in figure 31. The flow of information used to calculate horizontal-hydraulic conductances (CC and CR) is shown for several of the modules. For example, BCFlRP passes transmissivity (T) and cell dimensions (DELR and DELC) to SBCFlN. Module SBCFlN then returns CC and CR to BCFlRP. The modules are:

**Primary Modules**

BCFlAL
- Allocates space for data arrays.

BCFlRP
- Reads all data needed by the package, invokes SBCFlN to reconcile input transmissive values with the IBOUND array, and calculates storage capacities and constant conductances.

BCFlFM
- Calculates all coefficients of the system of equations that are not constant and invokes SBCFlH to calculate horizontal-branch conductances in partially saturated layers.

BCFlBD
- Calculates flow rates and accumulated flow volumes into and out of storage and constant-head boundaries. When cell-by-cell flow is specified, flow across all sides of each cell is also calculated.

**Submodules**

SBCFlN
- Reconciles input transmissive values with the IBOUND array and calculates storage capacities and constant conductances. Invokes SBCFlC to calculate horizontal-branch conductances for layers where transmissivity is constant.

SBCFlH
- Calculates transmissivity for cells in layers where it depends on heads and invokes SBCFlC to calculate horizontal-branch conductances.

SBCFlC
- Calculates horizontal-branch conductance from cell transmissivity.

SBCFlB
- Calculates cell-by-cell flow terms across cell faces.

SBCFlF
- Calculates flow terms (both cell-by-cell and entries to overall budget) for flow to and from constant-head cells.
Figure 31.—Relationship among the modules in the Block-Centered Flow Package.

<table>
<thead>
<tr>
<th>CC</th>
<th>Conductance in the Column Direction</th>
<th>HY</th>
<th>Hydraulic Conductivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR</td>
<td>Conductance in the Row Direction</td>
<td>TOP</td>
<td>Elevation of the Top of a Layer</td>
</tr>
<tr>
<td>T</td>
<td>Transmissivity</td>
<td>BOT</td>
<td>Elevation of the Bottom of a Layer</td>
</tr>
<tr>
<td>DELR</td>
<td>Grid Spacing in the Row Direction</td>
<td>DELC</td>
<td>Grid Spacing in the Column Direction</td>
</tr>
</tbody>
</table>
Narrative for Module BCFlAL

This module allocates space for data arrays for the Block-Centered Flow Package. It is done in the following order:

1. Print the message identifying the package.

2. Read and print the steady-state flag ISS and the cell-by-cell flow-term unit and flag (IBCFCB). Cell-by-cell flow terms for the BCF Package are flow to the right, flow forward, flow down, increase in storage, and flow to constant heads.

3. Read and print the layer-type code and count the number of layers which need the TOP array and the BOTTOM array.

   (a) Read the layer-type codes.

   0 = confined
   1 = unconfined
   2 = confined/unconfined but transmissivity is constant
   3 = confined/unconfined but transmissivity depends on head

   (b) Initialize the counters KT and KB in which the numbers of layers needing the TOP and BOTTOM are accumulated.

   (c) For each layer, print the layer-type code and determine if TOP and/or BOTTOM arrays are needed.

      (1) Print the layer number and the layer-type code.

      (2) If a layer other than the top layer is unconfined (type = 1), print an error message and STOP.
(3) If the layer type is one or three, add one to the BOTTOM counter, KB.

(4) If the layer type is two or three, add one to the TOP counter, KT.

4. Calculate the number of elements in the grid and in a layer.

5. Allocate space for the following arrays:

   SC1  Primary storage capacity;
   SC2  Secondary-storage capacity (layer type 2 or 3 only);
   TRPY Horizontal anisotropy factor;
   BOT  Bottom of layers (layer type 2 or 3 only);
   TOP  Top of layers (layer type 2 or 3 only); and
   HY   Hydraulic conductivity (layer type 1 or 3 only).

The following notes apply:

   If the simulation is transient (ISS = 0), storage coefficients are
   needed.

   The number of vertical conductance arrays is one less than the
   number of layers.

6. Print the amount of space used by the BCF Package.

7. RETURN.
ISS is the steady-state flag. If it is set (ISS = 1), the simulation is steady state (storage is not considered).

IBCFCB is a flag and a unit number.

If IBCFCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.

If IBCFCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IBCFCB < 0, flow from constant-head cells will be printed whenever ICBCFL is set.

LAYCON is a layer-type code (one for each layer).

0 - confined
1 - unconfined
2 - confined/unconfined but transmissivity is constant
3 - confined/unconfined

KT is a counter for the number of layers for which TOP is needed. (It is also the number of layers for which a secondary storage factor is needed.)

KB is a counter for the number of layers for which BOTTOM is needed. (It is also the number of layers for which hydraulic conductivity is needed.)
SUBROUTINE BCFlAL(ISUM, LENX, LCSC1, LCHY, LCBOT, LCTOP, LSCSZ, LCTRYP, IN, ISS, NCOL, NROW, NLAY, IOUT, IBCFCB)

C
C-----VERSION 1542 12MAY1987 BCFlAL
C
C ******************************************
C ALLOCATE ARRAY STORAGE FOR BLOCK-CENTERED FLOW PACKAGE
C ******************************************

C
C SPECIFICATIONS:
C ----------------------------------
COMMON /FLWCOM/LAYCON(80)
C
C
C1------IDENTIFY PACKAGE
C    WRITE(IOUT,1)IN
1 FORMAT(1HO,'BCFl -- BLOCK-CENTERED FLOW PACKAGE, VERSION 1',
     1', 9/187', ' INPUT READ FROM UNIT', I3)

C
C2------READ AND PRINT ISS (STEADY-STATE FLAG) AND IBCFCB (FLAG FOR
C2------PRINTING OR UNIT# FOR RECORDING CELL-BY-CELL FLOW TERMS)
C    READ(IN,2) ISS, IBCFCB
2 FORMAT(2110)
     IF(ISS.EQ.0) WRITE(IOUT,3)
3 FORMAT(1X,'TRANSIENT SIMULATION')
     IF(ISS.NE.0) WRITE(IOUT,4)
4 FORMAT(1X,'STEADY-STATE SIMULATION')
     IF(IBCFCB.GT.0) WRITE(IOUT,9) IBCFCB
9 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE RECORDED ON UNIT', I3)
     IF(IBCFCB.LT.0) WRITE(IOUT,88)
88 FORMAT(1X,'CONSTANT HEAD CELL-BY-CELL FLOWS WILL BE PRINTED')

C
C3------READ TYPE CODE FOR EACH LAYER AND COUNT TOPS AND BOTTOMS
C   IF(NLAY.LE.80) GO TO 50
C   WRITE(IOUT,11)
11 FORMAT(1HO,'YOU HAVE SPECIFIED MORE THAN 80 MODEL LAYERS'/IX,
   1 'SPACE IS RESERVED FOR A MAXIMUM OF 80 LAYERS IN ARRAY LAYCON')
   STOP

C
C3A------READ LAYER TYPE CODES.
C   50 READ(IN,51) (LAYCON(I), I=1, NLAY)
51 FORMAT(40I2)
C   BOTTOM IS READ FOR TYPES 1,3 TOP IS READ FOR TYPES 2,3
C   WRITE(IOUT,52)
52 FORMAT(1X,5X,'LAYER AQUIFER TYPE',/1X,5X,19('-'))

C
C3B------INITIALIZE TOP AND BOTTOM COUNTERS.
C   NBOT=0
C   NTOP=0

C
C3C------PRINT LAYER TYPE AND COUNT TOPS AND BOTTOMS NEEDED.
C   DO 100 I=1, NLAY
C3C1----PRINT LAYER NUMBER AND LAYER TYPE CODE.
    L=LAYCON(I)
    WRITE(IOUT,7) I,L
    7 FORMAT(1X,I9,110)

C3C2----ONLY THE TOP LAYER CAN BE UNCONFINED(LAYCON=1).
    IF(L.NE.1 .OR. L.EQ.1) GO TO 70
    WRITE(IOUT,8)
    8 FORMAT(1HO,'AQUIFER TYPE 1 IS ONLY ALLOWED IN TOP LAYER')
    STOP

C3C3----LAYER TYPES 1 AND 3 NEED A BOTTOM. ADD 1 TO KB.
    IF(L.EQ.1 .OR. L.EQ.3) NBOT=NBOT+1

C3C4----LAYER TYPES 2 AND 3 NEED A TOP. ADD 1 TO KT.
    IF(L.EQ.2 .OR. L.EQ.3) NTOP=NTOP+1
    100 CONTINUE

C

C4-------COMPUTE DIMENSIONS FOR ARRAYS.
    NRC= NROW*NCOL
    ISIZ=NRC*NLAY

C5-------ALLOCATE SPACE FOR ARRAYS. IF RUN IS TRANSIENT(ISS=0)
    THEN SPACE MUST BE ALLOCATED FOR STORAGE.
    ISOLD=ISUM
    LCSC1=ISUM
    IF(ISS.EQ.0) ISUM=ISUM+ISIZ
    LCSC2=ISUM
    IF(ISS.EQ.0) ISUM=ISUM+NRC*NTOP
    LCTRYP=ISUM
    ISUM=ISUM+NLAY
    LCROT=ISUM
    ISUM=ISUM+NRC*NBOT
    LCHY=ISUM
    ISUM=ISUM+NRC*NBOT
    LCTOP=ISUM
    ISUM=ISUM+NRC*NTOP

C6-------PRINT THE AMOUNT OF SPACE USED BY THE BCF PACKAGE.
    ISP=ISUM-ISOLD
    WRITE(IOUT,101) ISP
    101 FORMAT(1X,I8,' ELEMENTS IN X ARRAY ARE USED BY BCF')
    ISUM1=ISUM-1
    WRITE(IOUT,102) ISUM1,LENX
    102 FORMAT(1X,I8,' ELEMENTS OF X ARRAY USED OUT OF',181
    IF(ISUM1.GT.LENX) WRITE(IOUT,103)
    103 FORMAT(1X, '***X ARRAY MUST BE DIMENSIONED LARGER***')

C

C7-------RETURN
    RETURN
    END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Module</td>
<td>Index.</td>
</tr>
<tr>
<td>IBCFCB</td>
<td>Package</td>
<td>Flag and a unit number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will be not be printed or recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, flow from each constant-head cell will be printed whenever IBCFL is set.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISIZ</td>
<td>Module</td>
<td>Number of cells in the grid.</td>
</tr>
<tr>
<td>ISOLD</td>
<td>Package</td>
<td>Before this module allocates space, ISOLD is set equal to ISUM. After allocation, ISOLD is subtracted from ISUM to get ISP, the amount of space in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISP</td>
<td>Module</td>
<td>Number of words in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISS</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, simulation is transient.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, simulation is steady state.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.</td>
</tr>
<tr>
<td>ISUM1</td>
<td>Module</td>
<td>ISUM-1.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Temporary storage for LAYCON(I).</td>
</tr>
<tr>
<td>LAYCON</td>
<td>Package</td>
<td>DIMENSION (80) Layer type code:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - Layer strictly confined.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - Layer strictly unconfined.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - Layer confined/unconfined (transmissivity is constant).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - Layer confined/unconfined (transmissivity varies).</td>
</tr>
<tr>
<td>LCBOT</td>
<td>Package</td>
<td>Location in the X array of the first element of array BOT.</td>
</tr>
<tr>
<td>LCHY</td>
<td>Package</td>
<td>Location in the X array of the first element of array HY.</td>
</tr>
<tr>
<td>LCSC1</td>
<td>Package</td>
<td>Location in the X array of the first element of array SC1.</td>
</tr>
<tr>
<td>LCSC2</td>
<td>Package</td>
<td>Location in the X array of the first element of array SC2.</td>
</tr>
<tr>
<td>LCTOP</td>
<td>Package</td>
<td>Location in the X array of the first element of array TOP.</td>
</tr>
<tr>
<td>LCTRPY</td>
<td>Package</td>
<td>Location in the X array of the first element of array TRPY.</td>
</tr>
<tr>
<td>LENX</td>
<td>Global</td>
<td>Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.</td>
</tr>
<tr>
<td>NBOT</td>
<td>Module</td>
<td>Counter for the number of layers which need elevation of the bottom. Layers for which LAYCON = 1 or 3.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NRC</td>
<td>Module</td>
<td>Number of cells in a layer.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>NTOP</td>
<td>Module</td>
<td>Counter for the number of layers which need elevation of the top. LAYCON = 2 or 3.</td>
</tr>
</tbody>
</table>

5-49
Narrative for Module BCF1RP

This module reads transmissivity along rows, hydraulic conductivity along rows, storage coefficients, vertical conductance, elevation of top of layer, and elevation of bottom of layer. It also calls SBCFIN to calculate parameters which are constant throughout simulation. It does this in the following order:

1. Call utility module UIDREL to read DELR, DELC, and TRPY which have one value for each column, row, and layer, respectively. TRPY is the ratio of transmissivity along columns to transmissivity along rows for each layer.

2. For each layer, use utility module U2DREL to read the properties of the porous medium. The data requirements for each layer are determined by the layer-type code.
   
   (a) Find the address of the layer in the three-dimension arrays.
   
   (b) If the simulation is transient (ISS = 0), read the primary storage coefficient.
   
   (c) For constant transmissivity layers (LAYCON = 0 or 2), read the transmissivity.
   
   (d) For variable transmissivity layers (LAYCON = 1 or 3), read hydraulic conductivity and bottom.
   
   (e) Read vertical-hydraulic conductivity divided by thickness. These values will be multiplied in the program by cell areas to get vertical conductance. For each layer, the vertical conductance to the next lower layer is calculated. Therefore, no vertical conductance is calculated for the lowest layer in the mesh.
   
   (f) If the simulation is transient and the layer type is two or three, read the secondary storage coefficient (specific yield).
   
   (g) Read the top elevation if the layer type is two or three.

3. Call SBCFIN to calculate conductance and storage terms which are constant during the simulation and check to see that branch conductances agree with boundaries specified in the IBOUND array.

4. RETURN.
DELR is the grid spacing in the row direction.

DELC is the grid spacing in the column direction.

TRPY is the ratio of transmissivity in the column direction to transmissivity in the row direction.

LAYCON is a layer-type code (one for each layer).

0 - confined
1 - unconfined
2 - confined/unconfined but transmissivity is constant
3 - confined/unconfined

Secondary Storage coefficient is relevant only for convertible layers (LAYCON = 2 or 3); then it is equal to specific yield.
SUBROUTINE BCF1RP(IBOUND, HNEW, SC1, HY, CR, CC, CV, DELR, DELC,
1     DOT, TOP, SC2, TRPY, IN, ISS, NCOL, NROW, NLAY, NODES, IOUT)
C
C ------ VERSION 1636 15MAY1987 BCF1RP
C
C **********************************************************************
C READ AND INITIALIZE DATA FOR BLOCK-CENTERED FLOW PACKAGE
C **********************************************************************
C
C SPECIFICATIONS:
C ---------------------------------------------------------------
C CHARACTER*4 ANAME
C DOUBLE PRECISION HNEW
C
C DIMENSION HNEW(NODES), SC1(NODES), HY(NODES), CR(NODES), CC(NODES),
1     CV(NODES), ANAME(6,10), DELR(NCOL), DELC(NROW), BOT(NODES),
1     TOP(NODES), SC2(NODES), TRPY(NLAY), IBOUND(NODES)
C
C COMMON /FLWCOM/LAYCON(80)
C
C DATA ANAME(1,1), ANAME(2,1), ANAME(3,1), ANAME(4,1), ANAME(5,1),
1     ANAME(6,1) /' PRIM', 'ARY', 'STOR', 'AGE', 'COEF'/
C DATA ANAME(1,2), ANAME(2,2), ANAME(3,2), ANAME(4,2), ANAME(5,2),
1     ANAME(6,2) /' TRAN', 'SMIS', ' AL', 'ONG', 'ROWS'/
C DATA ANAME(1,3), ANAME(2,3), ANAME(3,3), ANAME(4,3), ANAME(5,3),
1     ANAME(6,3) /' HY', 'YD', 'COND', ' AL', 'ONG', 'ROWS'/
C DATA ANAME(1,4), ANAME(2,4), ANAME(3,4), ANAME(4,4), ANAME(5,4),
1     ANAME(6,4) /' VERT', 'HYD', 'COND', 'AL', 'ONG', 'ROWS'/
C DATA ANAME(1,5), ANAME(2,5), ANAME(3,5), ANAME(4,5), ANAME(5,5),
1     ANAME(6,5) /' ', ' ', ' ', ' ', 'BO', 'TTOM'/
C DATA ANAME(1,6), ANAME(2,6), ANAME(3,6), ANAME(4,6), ANAME(5,6),
1     ANAME(6,6) /' ', ' ', ' ', ' ', ' ', 'TOP'/
C DATA ANAME(1,7), ANAME(2,7), ANAME(3,7), ANAME(4,7), ANAME(5,7),
1     ANAME(6,7) /' SE', 'COND', 'ARY', 'STOR', 'AGE', 'COEF'/
C DATA ANAME(1,8), ANAME(2,8), ANAME(3,8), ANAME(4,8), ANAME(5,8),
1     ANAME(6,8) /' COLU', 'MN T', 'O RO', 'W AN', 'ISOT', 'ROPY'/
C DATA ANAME(1,9), ANAME(2,9), ANAME(3,9), ANAME(4,9), ANAME(5,9),
1     ANAME(6,9) /' ', ' ', ' ', ' ', ' ', 'DELRA'/
C DATA ANAME(1,10), ANAME(2,10), ANAME(3,10), ANAME(4,10), ANAME(5,10),
1     ANAME(6,10) /' ', ' ', ' ', ' ', ' ', 'DELRC'/
C ---------------------------------------------------------------
C
C CALL UIDREL(TRPY, ANAME(1,8), NLAY, IN, IOUT)
C CALL UIDREL(DELR, ANAME(1,9), NCOL, IN, IOUT)
C CALL UIDREL(DELC, ANAME(1,10), NROW, IN, IOUT)
C
C
C
READ ALL PARAMETERS FOR EACH LAYER

K_T=0
K_B=0
DO 200 K=1,NLAY
  KK=K

FIND ADDRESS OF EACH LAYER IN THREE DIMENSION ARRAYS.

IF(LAYCON(K).EQ.1 .OR. LAYCON(K).EQ.3) KB=KB+1
IF(LAYCON(K).EQ.2 .OR. LAYCON(K).EQ.3) KT=KT+1
LOC=1+(K-1)*NIJ
LOC_B=1+(KB-1)*NIJ
LOC_T=1+(KT-1)*NIJ

READ PRIMARY STORAGE COEFFICIENT INTO ARRAY SC1 IF TRANSIENT

IF(ISS.EQ.0)CALL U2DREL(SC1(LOC),ANAME(1,1),NROW,NCOL,KK,IN,IOUT)

READ TRANSMISSIVITY INTO ARRAY CC IF LAYER TYPE IS 0 OR 2

IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.1) GO TO 100
CALL U2DREL(CC(LOC),ANAME(1,2),NROW,NCOL,KK,IN,IOUT)
GO TO 110

READ HYDRAULIC CONDUCTIVITY(HY) AND BOTTOM ELEVATION(BOT)

READ VERTICAL HYCOND/THICK INTO ARRAY CV IF NOT BOTTOM LAYER

IF(K.EQ.NLAY) GO TO 120
CALL U2DREL(CV(LOC),ANAME(1,4),NROW,NCOL,KK,IN,IOUT)

READ SECONDARY STORAGE COEFFICIENT INTO ARRAY SC2 IF TRANSIENT

AND LAYER TYPE IS 2 OR 3

120 IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.2) GO TO 200
IF(ISS.EQ.0)CALL U2DREL(SC2(LOC),ANAME(1,7),NROW,NCOL,KK,IN,IOUT)

READ TOP ELEVATION(TOP) IF LAYER TYPE IS 2 OR 3

CALL U2DREL(TOP(LOC_T),ANAME(1,6),NROW,NCOL,KK,IN,IOUT)

CONTINUE

PREPARE AND CHECK BCF DATA

CALL SBCF1N(HNEW,IBOUND,SC1,SC2,CR,CC,CV,HY,TRPY,DELR,DELC,ISS,
               NCOL,NROW,NLAY,IOUT)

RETURN
END
### List of Variables for Module BCFlRP

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANAME</td>
<td>Module</td>
<td>Label for printout of input array.</td>
</tr>
<tr>
<td>BOT</td>
<td>Package</td>
<td>Elevation of the bottom of each layer. (NBOT is the number of layers for which LAYCON = 1 or 3.)</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>Conductance in the column direction. CC(J, I, K) contains conductance between nodes (J, I, K) and (J, I+1, K). This array is used to temporarily hold transmissivity.</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>Conductance in the row direction. CR(J, I, K) contains conductance between nodes (J, I, K) and (J, I, K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>Conductance in the vertical direction. CV(J, I, K) contains conductance between nodes (J, I, K) and (J, I, K+1). This array is used to temporarily hold Vcont.</td>
</tr>
<tr>
<td>DELC</td>
<td>Global</td>
<td>Cell dimension in the column direction. DELC(I) contains width of row I.</td>
</tr>
<tr>
<td>DELR</td>
<td>Global</td>
<td>Cell dimension in the row direction. DELR(J) contains the width of column J.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HY</td>
<td>Package</td>
<td>Hydraulic conductivity of a cell. (NBOT is the number of layers where LAYCON = 1 or 3.)</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>Status of each cell. &lt; 0, constant-head cell = 0, inactive cell &gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISS</td>
<td>Package</td>
<td>Flag. 0, simulation is transient. ≠ 0, simulation is steady state.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KB</td>
<td>Module</td>
<td>Counter for the number of layers for which the bottom elevation is needed (LAYCON = 1 or 3).</td>
</tr>
<tr>
<td>KK</td>
<td>Module</td>
<td>Temporary variable set equal to K. KK is used as an actual argument in subroutine calls to avoid using the DO loop variable K as an argument, which causes problems with some compilers.</td>
</tr>
<tr>
<td>KT</td>
<td>Module</td>
<td>Counter for the number of layers for which the top elevation is needed (LAYCON = 2 or 3).</td>
</tr>
<tr>
<td>LAYCON</td>
<td>Package</td>
<td>Layer type code: 0 - Layer strictly confined. 1 - Layer strictly unconfined. 2 - Layer confined/unconfined (transmissivity is constant). 3 - Layer confined/unconfined (transmissivity...</td>
</tr>
<tr>
<td>Variable</td>
<td>Range</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td>------------</td>
</tr>
<tr>
<td>LOC</td>
<td>Module</td>
<td>Pointer to parts of the conductance arrays corresponding to particular layers.</td>
</tr>
<tr>
<td>LCOB</td>
<td>Module</td>
<td>Pointer to parts of the BOT and HY arrays corresponding to particular layers.</td>
</tr>
<tr>
<td>LOCT</td>
<td>Module</td>
<td>Pointer to parts of the TOP and SC1 arrays corresponding to particular layers.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NIJ</td>
<td>Module</td>
<td>Number of cells in a layer.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NODES</td>
<td>Global</td>
<td>Number of cells (nodes) in the finite-difference grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>tSC1</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NLAY), Primary storage capacity of each cell (S<em>DELC</em>DELR).</td>
</tr>
<tr>
<td>tSC2</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NTOP), Secondary storage capacity of each cell in the grid. (NTOP is the number of layers for which LAYCON = 2 or 3.)</td>
</tr>
<tr>
<td>TOP</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NTOP), Elevation of the top of the layers. (NTOP is the number of layers for which LAYCON = 2 or 3.)</td>
</tr>
<tr>
<td>TRPY</td>
<td>Package</td>
<td>DIMENSION (NLAY), Ratio of transmissivity in the column direction to transmissivity in the row direction.</td>
</tr>
</tbody>
</table>

Initially, storage coefficient values are read into these arrays; these values are multiplied by cell areas in submodule SBCF1N to yield storage capacities.
Narrative for Module BCF1FM

This module calculates branch conductances which are not constant throughout the simulation, adds storage terms to the accumulators in which HCOF and RHS are formed, and adds terms to RHS and HCOF which correct for overestimation of flow down into partially saturated cells.

1. For each layer in which transmissivity varies with head (LAYCON = 1 or 3), call submodule SBCFlFM to calculate branch conductance.

2. If the simulation is transient, calculate storage terms (STEPS 3-5) for each layer. If the simulation is steady state, GO TO STEP 6.

3. Determine if there is one storage factor or two.

4. If there is only one storage factor (LAYCON = 0 or 1), use it to calculate storage terms and add them to the right hand side (RHS) and the h-coefficient (HCOF).

5. If there are two storage factors, then, using head at the beginning of the time step (HOLD), determine the storage factor at the beginning of the time step (SOLD) and use the latest estimate of head at the end of the time step (HNEW) to determine the storage factor at the end of the time step (SNEW). Use SOLD and SNEW to calculate the storage terms to add to RHS and HCOF.

6. For each layer, determine if correction terms are needed for flow down into a partially saturated layer (STEPS 7-8).

7. If the layer is partially saturated and there is flow from above, calculate correction terms and add to RHS and HCOF.

8. If this is not the bottom layer and the layer below is partially saturated, calculate the correction terms and add to RHS and HCOF.

9. RETURN.

The term storage factor, as used in Subroutine BCF1FM, refers to storage capacity divided by time step length. SOLD is thus equivalent here to SCA/(tm-tm,-1), in the notation of equations (61) and (62), while SNEW is equivalent to SCB/(tm-tm,-1).
LAYCON is a layer-type code (one for each layer).

0 - confined
1 - unconfined
2 - confined/unconfined but transmissivity is constant
3 - confined/unconfined

FOR EACH LAYER
IF T VARIES, CALCULATE HORIZONTAL CONDUCTANCES

TRANSIENT
YES

HOW MANY STORAGE FACTORS?
ONE
TWO

ADD STORAGE TERMS TO RHS AND HCOF

FOR EACH LAYER
IF LAYCON IS 2 OR 3, GET CORRECTION TERMS FOR FLOW FROM ABOVE
IF LAYCON FOR LAYER BELOW IS 2 OR 3, GET CORRECTION TERMS FOR FLOW DOWN

RETURN
SUBROUTINE BCF1FM(HCOF, RHS, HOLD, SCI, HNEW, IBOUND, CR, CC, CV, HY, TRPY, 
1      BOT, TOP, SC2, DELR, DELC, DELT, ISS, KITER, KSTP, KPER, 
2      NCOL, NROW, NLAY, IOUT)

C-----VERSION 1640 15MAY1987 BCF1FM
C
C ********************************************************************
C ADD LEAKAGE CORRECTION AND STORAGE TO HCOF AND RHS, AND CALCULATE
C CONDUCTANCE AS REQUIRED
C ********************************************************************
C
C SPECIFICATIONS:
C
DOUBLE PRECISION HNEW
C
DIMENSION HCOF(NCOL, NROW, NLAY), RHS(NCOL, NROW, NLAY), 
1      HOLD(NCOL, NROW, NLAY), SCI(NCOL, NROW, NLAY), HNEW(NCOL, NROW, NLAY), 
2      IBOUND(NCOL, NROW, NLAY), CR(NCOL, NROW, NLAY), 
3      CC(NCOL, NROW, NLAY), CV(NCOL, NROW, NLAY), HY(NCOL, NROW, NLAY), 
4      TRPY(NLAY), BOT(NCOL, NROW, NLAY), TOP(NCOL, NROW, NLAY), DELR(NCOL), 
5      DELC(NROW), SC2(NCOL, NROW, NLAY)
C
COMMON /FLWCON/LAYCON(BO)
C
-------------------------------------------------------------------

KB=0
KT=0
C
C1------FOR EACH LAYER: IF T VARIES CALCULATE HORIZONTAL CONDUCTANCES
DO 100 K=1,NLAY
  KK=K
  IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.2) KT=KT+1
C
C1A------IF LAYER TYPE IS NOT 1 OR 3 THEN SKIP THIS LAYER.
  IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.1) GO TO 100
  KB=KB+1
  ----FOR LAYER TYPES 1 & 3 CALL SBCFHI TO CALCULATE
C1B------HORIZONTAL CONDUCTANCES.
  CALL SBCFHI(HNEW, IBOUND, CR, CC, CV, HY, TRPY, DELR, DELC, BOT, TOP, 
1      KK, KB, KT, KITER, KSTP, KPER, NCOL, NROW, NLAY, IOUT)
100 CONTINUE
C
C2------IF THE SIMULATION IS TRANSIENT ADD STORAGE TO HCOF AND RHS
IF(ISS.NE.0) GO TO 201
  TLED=1./DELT
  KT=0
  DO 200 K=1,NLAY
C
C3------SEE IF THIS LAYER IS CONVERTIBLE OR NON-CONVERTIBLE.
  IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.2) GO TO 150
C4------NON-CONVERTIBLE LAYER, SO USE PRIMARY STORAGE
  DO 140 I=1,NROW
  DO 140 J=1,NCOL
    IF(IBOUND(J,I,K).LE.0) GO TO 140
    RHO=SCI(J,I,K)*TLED
    HCOF(J,I,K)=HCOF(J,I,K)-RHO 
    RHS(J,I,K)=RHS(J,I,K)-RHO*HOLD(J,I,K)
  140 CONTINUE
  GO TO 200
C
C5------A CONVERTIBLE LAYER, SO CHECK OLD AND NEW HEADS TO DETERMINE
C5A------WHEN TO USE PRIMARY AND SECONDARY STORAGE
  150 KT=KT+1
  DO 180 J=1,NCOL
    IF(IBOUND(J,I,K).LE.0) GO TO 180
    TP=TOP(J,I,K)
    RHO2=SC2(J,I,KT)*TLED
  180 CONTINUE
C
5-58
RHO1=SCI(J,I,K)*TLED

C
CSB------FIND STORAGE FACTOR AT START OF TIME STEP.
SOLD=RHO2
IF(HOLD(J,I,K).GT.TP) SOLD=RHO1

C
CSC------FIND STORAGE FACTOR AT END OF TIME STEP.
HTMP=HNEW(J,I,K)
SNEW=RHO2
IF(HTMP.GT.TP) SNEW=RHO1

C
CSD------ADD STORAGE TERMS TO RHS AND HCOF.
HCOF(J,I,K)=HCOF(J,I,K)-SNEW
RHS(J,I,K)=RHS(J,I,K) - SOLD*(HOLD(J,I,K)-TP) - SNEW*TP

C
180 CONTINUE

C
200 CONTINUE

C
C6------FOR EACH LAYER DETERMINE IF CORRECTION TERMS ARE NEEDED FOR
C6------FLOW DOWN INTO PARTIALLY SATURATED LAYERS.
201 KT=0
DO 300 K=1,NLAY

C
C7------SEE IF CORRECTION IS NEEDED FOR LEAKAGE FROM ABOVE.
IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.2) GO TO 250
KI=KI+1
IF(K.EQ.1) GO TO 250

C
C7A------FOR EACH CELL MAKE THE CORRECTION IF NEEDED.
DO 220 I=1,NROW
DO 220 J=1,NCOL

C
C7B------IF THE CELL IS EXTERNAL(IBOUND<=0) THEN SKIP IT.
IF(IBOUND(J,I,K).LE.0) GO TO 220
HTMP=HNEW(J,I,K)

C
C7C------IF HEAD IS ABOVE TOP THEN CORRECTION NOT NEEDED
IF(HTMP.GE.TOP(J,I,KT)) GO TO 220

C
C7D------WITH HEAD BELOW TOP ADD CORRECTION TERMS TO RHS AND HCOF.
RHS(J,I,K)=RHS(J,I,K) + CV(J,I,K-1)*TOP(J,I,KT)
HCOF(J,I,K)=HCOF(J,I,K) + CV(J,I,K-1)

220 CONTINUE

C
C8------FOR EACH CELL MAKE THE CORRECTION IF NEEDED.
DO 280 I=1,NROW
DO 280 J=1,NCOL

C
C8A------IF CELL IS EXTERNAL (IBOUND<=0) THEN SKIP IT.
IF(IBOUND(J,I,K).LE.0) GO TO 280

C
C8B------IF HEAD IN THE LOWER CELL IS LESS THAN TOP ADD CORRECTION
C8C------TERM TO RHS.
HTMP=HNEW(J,I,K+1)
IF(HTMP.LT.TOP(J,I,KTT)) RHS(J,I,K)=RHS(J,I,K)
1 - CV(J,I,K)*(TOP(J,I,KTT)-HTMP)

280 CONTINUE

300 CONTINUE

C
C9------RETURN
RETURN
END
## List of Variables for Module BCF1FM

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOT</td>
<td>Package</td>
<td>DIMENSION (NCOL, NROW, NBOT), Elevation of bottom of each layer. (NBOT is the number of layers for which LAYCON = 1 or 3.)</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL, NROW, NLAY), Conductance in the column direction. CC(J, I, K) contains conductance between nodes (J, I, K) and (J, I+1, K).</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL, NROW, NLAY), Conductance in the row direction. CR(J, I, K) contains conductance between nodes (J, I, K) and (J+1, I, K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL, NROW, NLAY-1), Conductance in the vertical direction. CV(J, I, K) contains conductance between nodes (J, I, K) and (J, I+1, K).</td>
</tr>
<tr>
<td>DELC</td>
<td>Global</td>
<td>DIMENSION (NROW), Cell dimension in the column direction. DELC(I) contains the width of row I.</td>
</tr>
<tr>
<td>DELR</td>
<td>Global</td>
<td>DIMENSION (NCOL), Cell dimension in the row direction. DELR(J) contains the width of column J.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NCOL, NROW, NLAY), Coefficient of head in cell (J, I, K) in the finite-difference equation.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL, NROW, NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HOLD</td>
<td>Global</td>
<td>DIMENSION (NCOL, NROW, NLAY), Head at the start of the current time step.</td>
</tr>
<tr>
<td>HTMP</td>
<td>Module</td>
<td>Temporary single precision HNEW(J, I, K).</td>
</tr>
<tr>
<td>HY</td>
<td>Package</td>
<td>DIMENSION (NCOL, NROW, NBOT), Hydraulic conductivity of a cell. (NBOT is the number of layers where LAYCON = 1 or 3.)</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL, NROW, NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISS</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, simulation is transient.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, simulation is steady state.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KB</td>
<td>Module</td>
<td>Counter for layers for which bottom elevation is needed.</td>
</tr>
<tr>
<td>KITER</td>
<td>Global</td>
<td>Iteration counter. Reset at the start of each time step.</td>
</tr>
<tr>
<td>KK</td>
<td>Module</td>
<td>Temporary variable set equal to K. KK is used as an actual argument in subroutine calls to avoid using the DO loop variable K as an argument, which causes problems with some compilers.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>KT</td>
<td>Module</td>
<td>Counter for layers for which top elevation is needed.</td>
</tr>
<tr>
<td>KTT</td>
<td>Module</td>
<td>Pointer to TOP array of layer immediately below layer K.</td>
</tr>
</tbody>
</table>
### List of Variables for Module BCF1FM (Continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
</table>
| LAYCON   | Package | DIMENSION (80) Layer type code:  
|          |        | 0 - Layer strictly confined.  
|          |        | 1 - Layer strictly unconfined.  
|          |        | 2 - Layer confined/unconfined (transmissivity is constant).  
|          |        | 3 - Layer confined/unconfined (transmissivity varies).  |
| NCOL     | Global | Number of columns in the grid.  |
| NLAY     | Global | Number of layers in the grid.  |
| NROW     | Global | Number of rows in the grid.  |
| RHO      | Module | Storage coefficient for strictly confined or strictly unconfined layers.  |
| tRHO1    | Module | Confined storage factor for convertible layers.  |
| tRHO2    | Module | Unconfined storage factor for convertible layers.  |
| RHS      | Global | DIMENSION (NCOL,NROW,NLAY), Right hand side of finite-difference equation. RHS is an accumulation of terms from several different packages.  |
| SC1      | Package | DIMENSION (NCOL,NROW,NLAY), Primary storage capacity of each cell (S*DELC*DELR).  |
| SC2      | Package | DIMENSION (NCOL,NROW,NTOP), Secondary storage capacity of each cell in the grid. (NTOP is the number of layers for which LAYCON = 2 or 3.)  |
| tSNEW    | Module | Storage factor at the end of the time step for convertible layers.  |
| tSOLD    | Module | Storage factor at the start of the time step for convertible layers.  |
| TLED     | Module | DIMENSION (NCOL,NROW,NTOP), Elevation of top of layers. (NTOP is the number of layers for which LAYCON = 2 or 3.)  |
| TOP      | Package | Temporary variable for TOP(J,I,K).  |
| TRPY     | Package | DIMENSION (NLAY), Ratio of transmissivity in the column direction to transmissivity in the row direction.  |

Storage factor, as used in Subroutine BCF1FM, refers to storage capacity divided by time step length.
Module BCF1BD calculates flow rates within the porous medium for use in the overall volumetric budget and calculates cell-by-cell flow terms for recording on disk. Flow rates to constant heads and from storage are accumulated and passed to the module BAS101 for inclusion in the budget. They are accumulated by sign so that flow into constant-head cells is separate from flow out of constant-head cells, and flow into storage is separate from flow out of storage. Flow rates to constant-head cells and from storage as well as flow across cell boundaries can be recorded on a cell-by-cell basis for use by other programs.

Flow from storage is calculated inside BCF1BD. Flow to constant-head cells and across cell boundaries is calculated in submodules SBCF1F and SBCF1B, respectively.

Module BCF1BD performs its tasks in the following order:

1. Clear the fields STOIN and STOUT in which flow out of and into storage, respectively, are accumulated.

2. If the user has specified that cell-by-cell flow terms should be recorded this time step (ICBCFL #0) and has specified a unit number (IBCFCB) for cell-by-cell flow terms for the BCF Package, set the cell-by-cell flag (IBD).

3. If this is steady-state simulation, skip all of the calculations for flow from storage.

4. If cell-by-cell flow terms are to be saved (i.e., if IBD was set in STEP 2), clear the buffer (BUFF) in which they will be accumulated prior to printing.

5. For each cell in the grid, calculate flow from storage and move to accumulator (STEPS 6 AND 7).

6. Calculate flow from storage in the cell.

7. If the cell-by-cell rates are being recorded, store flow rate from storage in the buffer. Depending on the sign, add the flow from storage to the accumulators STOIN or STOUT.

8. If the cell-by-cell flag (IBD) is set, record the contents of the buffer.

9. Store the accumulated rates and volumes of flow from storage in table VBVL for inclusion in the overall volumetric budget. Store an appropriate label in the corresponding location in the table VBNM.

10. Call submodule SBCF1F to calculate flow from constant-head cells.

11. If the cell-by-cell flag (IBD) is set, call submodule SBCF1B to calculate and record the flow across cell boundaries.

12. RETURN.
STOIN is an accumulator for flow terms having a positive sign (flow from storage into the flow system) for inclusion in the volumetric budget.

STOUT is an accumulator for flow terms having a negative sign (flow into storage and out of the flow system) for inclusion in the volumetric budget.

IBD is a flag which indicates that for this time step, BCF cell-by-cell flow terms should be recorded.

BUFF is a buffer where flow terms are gathered prior to recording them.

VBVL is a table of budget entries calculated by component-of-flow packages for use in calculating the volumetric budget.

VBNM is a table of labels for budget terms.
SUBROUTINE BCFlBD(VBNM, VBVL, MSUM, HNEW, IBOUND, HOLD, SC1, CR, CC, CV, 
1      TOP, SCZ, DELT, ISS, NCOL, NROW, NLAY, KSTEP, KPER, IBCFGB, 
2      IBCFL, BUFF, IOUT)

C-----VERSION 1546 12MAY1987 BCFlBD
C
C

COMPUTE BUDGET FLOW TERMS FOR BCF -- STORAGE, CONSTANT HEAD, AND 
FLOW ACROSS CELL WALLS

C
C SPECIFICATIONS:
C
------------------------------------------------------------------

CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW

DIMENSION HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY), 
1   HOLD(NCOL, NROW, NLAY), SC1(NCOL, NROW, NLAY), 
2   CR(NCOL, NROW, NLAY), CC(NCOL, NROW, NLAY), 
3   CV(NCOL, NROW, NLAY), VBNM(4, 20), VBVL(4, 20), 
4   SCZ(NCOL, NROW, NLAY), 
5   TOP(NCOL, NROW, NLAY), BUFF(NCOL, NROW, NLAY)

COMMON /FLWCOM/LAYCON(80)

DIMENSION TEXT(4)

DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /' ', ', ', 'STORAGE'/
------------------------------------------------------------------

C1-------INITIALIZE BUDGET ACCUMULATORS
STOIN=0.
STOUT=0.

C2-------IF CELL-BY-CELL FLOWS ARE NEEDED THEN SET FLAG IBD.
IBD=0
   IF(IBCFL, NE.0, AND. IBCFGB, GT.0) IBD=1

C3-------IF STEADY STATE THEN SKIP ALL STORAGE CALCULATIONS
   IF(ISS, NE.0) GO TO 305

C4-------IF CELL-BY-CELL FLOWS ARE NEEDED (IBD IS SET) CLEAR BUFFER
   IF(IBD, EQ.0) GO TO 220
   DO 210 K=1, NLAY
   DO 210 I=1, NROW
   DO 210 J=1, NCOL
       BUFF(J, I, K)=0.
   210 CONTINUE

C5-------RUN THROUGH EVERY CELL IN THE GRID
220 KT=0
   DO 300 K=1, NLAY
   DO 300 I=1, NROW
       LC=LAYCON(K)
       IF(LC, EQ.0, OR. LC, EQ.2) KT=KT+1
   DO 300 J=1, NCOL

C6-------CALCULATE FLOW FROM STORAGE (VARIABLE HEAD CELLS ONLY)
IF(IBOUND(J,I,K).LE.0) GO TO 300
HSING=HNEW(J,I,K)

C6A------CHECK LAYER TYPE TO SEE IF ONE STORAGE CAPACITY OR TWO
IF(LC.NE.3 .AND. LC.NE.2) GO TO 285

C6B------TWO STORAGE CAPACITIES
TP=TOP(J,I,KT)
SYA=SC2(J,I,KT)
SCFA=SCI(J,I,K)
SOLD=SYA
IF(HOLD(J,I,K).GT.TP) SOLD=SCFA
SNEW=SYA
IF(HSING.GT.TP) SNEW=SCFA
STRG=SOLD*(HOLD(J,I,K)-TP) + SNEW*TP - SNEW*HSING
GO TO 288

C6C------ONE STORAGE CAPACITY
285 SC=SCI(J,I,K)
STRG=SC*HOLD(J,I,K) - SC*HSING

C7-------STORE CELL-BY-CELL FLOW IN BUFFER AND ADD TO ACCUMULATORS
288 IF(IBD.EQ.1) BUFF(J,I,K)=STRG/DELT
   IF(STRG) 292,300,294
292 STOUT=STOUT-STRG
   GO TO 300
294 STOIN=STOIN+STRG

300 CONTINUE

C8-------IF IBD FLAG IS SET RECORD THE CONTENTS OF THE BUFFER
   IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,
   1     IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)

C9-------ADD TOTAL RATES AND VOLUMES TO VBVL & PUT TITLES IN VBNM
305 VBVL(1,MSUM)=VBVL(1,MSUM)+STOIN
   VBVL(2,MSUM)=VBVL(2,MSUM)+STOUT
   VBVL(3,MSUM)=STOIN/DELT
   VBVL(4,MSUM)=STOUT/DELT
   VBNM(1,MSUM)=TEXT(1)
   VBNM(2,MSUM)=TEXT(2)
   VBNM(3,MSUM)=TEXT(3)
   VBNM(4,MSUM)=TEXT(4)
   MSUM=MSUM+1

C10------CALCULATE FLOW FROM CONSTANT HEAD NODES
   CALL SBCF1F(VBNM,VBVL,MSUM,HNEW,CH,CC,CV,TP,DELT,
   1     NCOL,NROW,NLAY,KSTP,KPER,IBD,IBCFCB,ICBCFL,BUFF,IOUT)

C11------CALCULATE AND SAVE FLOW ACROSS CELL BOUNDARIES IF C-B-C
   IF(IBD.NE.0) CALL SBCF1B(HNEW,IBOUND,CR,CC,CTOP,DELT,
   1     NCOL,NROW,NLAY,KSTP,KPER,IBD,IBCFCB,BUFF,IOUT)

C12------RETURN
RETURN
END
### List of Variables for Module BCF1BD

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains the conductance between nodes (J,I,K) and (J,I+1,K).</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Module</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HOLD</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Head at the start of the current time step.</td>
</tr>
<tr>
<td>HSING</td>
<td>Module</td>
<td>Temporary label for element of HNEW.</td>
</tr>
<tr>
<td>I</td>
<td>Package</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBCFCB</td>
<td>Package</td>
<td>Flag and a unit number. &gt; 0, unit number on which the cell-by-cell flow terms will be recorded whenever ICBCFL is set.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell. &lt; 0, constant-head cell = 0, inactive cell &gt; 0, variable-head cell</td>
</tr>
<tr>
<td>ICBFL</td>
<td>Global</td>
<td>Flag. = 0, cell-by-cell flow terms will not be recorded or printed for the current time step.</td>
</tr>
<tr>
<td>IICFL</td>
<td>Package</td>
<td>= 0, cell-by-cell flow terms will be either printed or recorded (depending on IBCFCB) for the current time step.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISS</td>
<td>Package</td>
<td>Flag. = 0, simulation is transient.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
</tbody>
</table>

**Note:** ICBCFL and IBCFCB are additional variables used in the context of the module. ICBCFL is a flag indicating whether cell-by-cell flow terms for the current package will be printed or recorded. IBCFCB is a flag and a unit number used in cell-by-cell flow calculations. The behavior of these variables is dependent on specific conditions within the module's logic.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>KT</td>
<td>Module</td>
<td>Index for top of layers (also used for secondary storage terms).</td>
</tr>
<tr>
<td>LC</td>
<td>Module</td>
<td>Temporary name for LAYCON(K).</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VRNM.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>SC</td>
<td>Module</td>
<td>Temporary name for the storage capacity.</td>
</tr>
<tr>
<td>SCFA</td>
<td>Module</td>
<td>Temporary name for the primary storage capacity.</td>
</tr>
<tr>
<td>SC1</td>
<td>Package</td>
<td>DIMENSION (NCOL, NROW, NLAY), Primary storage capacity of each cell (S<em>DELC</em>DELR).</td>
</tr>
<tr>
<td>SC2</td>
<td>Package</td>
<td>DIMENSION (NCOL, NROW, NTOP), Secondary storage capacity of each cell in the grid. (NTOP is the number of layers for which LAYCON = 2 or 3.)</td>
</tr>
<tr>
<td>SNEW*</td>
<td>Module</td>
<td>Storage capacity at the end of the time step.</td>
</tr>
<tr>
<td>SOLD*</td>
<td>Module</td>
<td>Storage capacity at the start of the time step.</td>
</tr>
<tr>
<td>STOIN</td>
<td>Module</td>
<td>Sum of decreases in storage from individual cells.</td>
</tr>
<tr>
<td>STOUT</td>
<td>Module</td>
<td>Sum of increases in storage for individual cells.</td>
</tr>
<tr>
<td>STRG</td>
<td>Module</td>
<td>Volume of flow into or out of storage in a single cell.</td>
</tr>
<tr>
<td>SYA</td>
<td>Module</td>
<td>Temporary name for the secondary storage capacity.</td>
</tr>
<tr>
<td>TEXI</td>
<td>Module</td>
<td>Labels recorded along with the cell-by-cell flow terms.</td>
</tr>
<tr>
<td>TOP</td>
<td>Package</td>
<td>DIMENSION (NCOL, NROW, NTOP), Elevation of top of layers. (NTOP is the number of layers for which LAYCON = 2 or 3.)</td>
</tr>
<tr>
<td>TP</td>
<td>Module</td>
<td>Temporary label for TOP(J,I,K).</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION(4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION(4,20), Entries for the volumetric budget. For flow component N, the values in VBVL are: (1,N), Rate for the current time step into the flow field. (2,N), Rate for the current time step out of the flow field. (3,N), Volume into the flow field during simulation. (4,N), Volume out of the flow field during simulation.</td>
</tr>
</tbody>
</table>

*Note that the variables SOLD and SNEW have different meanings in this subroutine than in BCF1FM.*
Narrative for Module SBCFIN

This module insures that the transmissive properties of each cell agree with the codes specified in the boundary array (IBOUND) and calculates (1) horizontal-branch conductance in layers where transmissivity is constant, (2) vertical-branch conductance, and (3) storage capacity.

The array IBOUND indicates the status of every cell in the grid with the following codes.

<table>
<thead>
<tr>
<th>Code</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>zero</td>
<td>inactive</td>
</tr>
<tr>
<td>positive</td>
<td>variable head</td>
</tr>
<tr>
<td>negative</td>
<td>constant head</td>
</tr>
</tbody>
</table>

The values in the IBOUND array are read by the BAS1RP module; transmissive properties are read by module BCF1RP. This module (SBCFIN) insures that all transmissive parameters are equal to zero for cells designated inactive by the IBOUND array and that cells are designated "inactive" if all transmissive parameters are equal to zero.

Module SBCFIN is called by module BCF1RP and calls submodule SBCF1C. The SBCFIN module performs these functions in the following order:

1. Check the cell to see if it is designated inactive (IBOUND = 0). If it is inactive, set the vertical leakance (temporarily stored in CV), transmissivity (temporarily stored in CC), and hydraulic conductivity equal to zero.

2. Check the cell that is designated active to insure that there is at least one nonzero transmissive parameter. If there are no such nonzero transmissive parameters, designate the cell inactive and print an error message.

   (a) If the transmissivity is constant (LAYCON = 0 or 2), the transmissivity or vertical-hydraulic conductivity must be nonzero.

   (b) If the transmissivity is a function of head (LAYCON = 1 or 3), the hydraulic conductivity or vertical conductance must be nonzero.

3. Calculate the horizontal-branch conductances for layers where the transmissivity is constant (LAYCON = 0 or 2). Submodule SBCF1C is invoked to calculate the branch conductance from the transmissivity and cell dimensions.

4. Multiply the vertical leakance between cells (temporarily stored in CV) by the cell dimensions to get the vertical conductance.

5. If the simulation is transient, multiply the primary storage coefficient by DELR and DELC to get the primary storage capacity (SC1).

6. If the layer is confined/unconfined, multiply the secondary storage coefficient by DELR and DELC to get the secondary storage capacity (SCZ).

7. RETURN.
.layer TYPES are designated in the LAYCON table. Layer types are:

0 - confined
1 - unconfined
2 - constant/unconfined but transmissivity is constant
3 - confined/unconfined

Primary storage capacity is taken as specific yield times cell area for unconfined layers, and as confined storage coefficient times cell area for confined or confined/unconfined layers.

Secondary storage capacity is defined for confined/unconfined aquifers and is always taken as specific yield times cell area.
SUBROUTINE SBCF1N(HNEW, IBOUND, SC1, SC2, CR, CC, CV, HY, TRPY, DELR, DELC,
  1   ISS, NCOL, NROW, NLAY, IOUT)

C-----------------------------------------------------------------------------
C                   VERSION 1642 15MAY1987 SBCF1N
C-----------------------------------------------------------------------------
C                    INITIALIZE AND CHECK BCF DATA
C-----------------------------------------------------------------------------
C                    SPECIFICATIONS:
C-----------------------------------------------------------------------------
C
DOUBLE PRECISION HNEW, HCNV

DIMENSION HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW,NLAY)
1   SC1(NCOL, NROW, NLAY), CR(NCOL, NROW, NLAY)
2   CC(NCOL, NROW, NLAY), CV(NCOL, NROW, NLAY)
3   HY(NCOL, NROW, NLAY), TRPY(NLAY), DELR(NCOL), DELC(NROW)
4   SC2(NCOL, NROW, NLAY)

COMMON /FLWCOM/LAYCON(BO)

-----------------------------------------------------------------------------
C1----IF IBOUND=0, SET CV=0., CC=0., AND HY=0.
  KB=0
  DO 30 K=1,NLAY
    IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.1) KB=KB+1
  DO 30 I=1,NROW
  DO 30 J=1,NCOL
    IF(IBOUND(J,I,K).NE.0) GO TO 30
    IF(K.NE.NLAY) CV(J,I,K)=0.
    IF(K.NE.1) CV(J,I,K-1)=0.
    CC(J,I,K)=0.
    IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.1) HY(J,I,KB)=0.
  30 CONTINUE

C2----INSURE THAT EACH ACTIVE CELL HAS AT LEAST ONE NON-ZERO TRANSMISSIVE PARAMETER. IF NOT, CONVERT CELL TO NOFLOW.
  HCNV=888.88
  KB=0
  DO 60 K=1,NLAY
    IF(LAYCON(K).EQ.1 .OR. LAYCON(K).EQ.3) GO TO 55
    DO 54 I=1,NROW
      DO 54 J=1,NCOL
        IF(IBOUND(J,I,K).EQ.0) GO TO 54
        IF(CC(J,I,K).NE.0.) GO TO 54
        IF(K.EQ.NLAY) GO TO 51
        IF(CV(J,I,K).NE.0.) GO TO 54
      51 IF(K.EQ.1) GO TO 53
        IF(CV(J,I,K-1).NE.0.) GO TO 54
      53 IBOUND(J,I,K)=0
        HNEW(J,I,K)=HCNV
        WRITE(IOUT,52) K,I,J
      52 FORMAT(1X,'NODE (LAYER,ROW,COL) ',3I4,
        1   ' ELIMINATED BECAUSE ALL CONDUCTANCES TO NODE ARE 0')
      54 CONTINUE
  60 CONTINUE
55 KB=KB+1
  DO 59 I=1,NROW
  DO 59 J=1,NCOL
     IF(IBOUND(J,I,K).EQ.0) GO TO 59
     IF(HY(J,I,KB).NE.0.) GO TO 59
     IF(K.EQ.NLAY) GO TO 56
     IF(CV(J,I,K).NE.0.) GO TO 59
  56 IF(K.EQ.1) GO TO 57
  IF(CV(J,I,K-1).NE.0.) GO TO 59
  57 IBOUND(J,I,K)=0
    HNEW(J,I,K)=HCNV
    CC(J,I,K)=0.
    WRITE(IOUT,52) K,I,J
  59 CONTINUE
  60 CONTINUE

C3-------CALCULATE HOR. CONDUCTANCE(CR AND CC) FOR CONSTANT T LAYERS
  DO 65 K=1,NLAY
    KK=K
    IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.1) GO TO 65
    CALL SBCFLC(CR,CC,TRPY,DELR,DELC,KK,NCOL,NROW,NLAY)
  65 CONTINUE

C4--------MULTIPLY VERTICAL LEAKANCE BY AREA TO MAKE CONDUCTANCE
  IF(NLAY.EQ.1) GO TO 69
    K1=NLAY-1
    DO 68 K=1,K1
      DO 68 I=1,NROW
        DO 68 J=1,NCOL
           CV(J,I,K)=CV(J,I,K)*DELR(J)*DELC(I)
      68 CONTINUE
  69 IF(ISS.NE.0) GO TO 100
    KT=0
    DO 80 K=1,NLAY
      DO 70 I=1,NROW
        DO 70 J=1,NCOL
           SC1(J,I,K)=SC1(J,I,K)+DELR(J)*DELC(I)
      70 CONTINUE
  80 CONTINUE

C5--------IF TRANSIENT MULTIPLY PRIMARY STORAGE COEFFICIENT BY DELR &
C5-------DEL C TO GET PRIMARY STORAGE CAPACITY(SC1).
  IF(ISS.NE.0) GO TO 100
     KT=KT+1
     DO 75 I=1,NROW
       DO 75 J=1,NCOL
          SC1(J,I,KT)=SC1(J,I,KT)*DELR(J)*DELC(I)
    75 CONTINUE

C6-------IF LAYER IS CONF/UNCONF MULTIPLY SECONDARY STORAGE COEFFICIENT
C6-------BY DELR AND DELC TO GET SECONDARY STORAGE CAPACITY(SC2).
  IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.2) GO TO 80
    KT=KT+1
    DO 75 I=1,NROW
      DO 75 J=1,NCOL
         SC2(J,I,KT)=SC2(J,I,KT)*DELR(J)*DELC(I)
  75 CONTINUE

C
  80 CONTINUE
C
C7-------RETURN
  100 RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>Global</td>
<td>Dimension (NCOL, NROW, NLAY). Conductance in the column direction. CC(J, I, K) contains conductance between nodes (J, I, K) and (J, I+1, K). This array is used to temporarily hold transmissivity.</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>Dimension (NCOL, NROW, NLAY), Conductance in the row direction. CR(J, I, K) contains conductance between nodes (J, I, K) and (J+1, I, K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>Dimension (NCOL, NROW, NLAY-1), Conductance in the vertical direction. CV(J, I, K) contains conductance between nodes (J, I, K) and (J, I, K+1). This array is used to temporarily hold Vcont.</td>
</tr>
<tr>
<td>DELC</td>
<td>Global</td>
<td>Dimension (NROW), Cell dimension in the column direction. DELC(I) contains the width of row I.</td>
</tr>
<tr>
<td>DELR</td>
<td>Global</td>
<td>Dimension (NCOL), Cell dimension in the row direction. DELR(J) contains the width of column J.</td>
</tr>
<tr>
<td>HCNV</td>
<td>Module</td>
<td>Indicator in the HNEW array that the cell is inactive.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>Dimension (NCOL, NROW, NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HY</td>
<td>Package</td>
<td>Dimension (NCOL, NROW, NBOT). Hydraulic conductivity of the cell. (NBOT is the number of layers where LAYCON = 1 or 3.)</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
</tbody>
</table>
| IBOUND   | Global | Dimension (NCOL, NROW, NLAY), Status of each cell. < 0, constant-head cell  
|          |        | = 0, inactive cell  
|          |        | > 0, variable-head cell  
| IOUT     | Global | Primary unit number for all printed output. IOUT = 6.                                                                                                                                                   |
| ISS      | Package| Flag. = 0, simulation is transient.  
|          |        | ≠ 0, simulation is steady state.                                                                                                                                                                        |
| J        | Module | Index for columns.                                                                                                                                                                                      |
| K        | Module | Index for layers.                                                                                                                           |
| KB       | Module | Index for bottom of layers.                                                                                                               |
| KK       | Module | Temporary variable set equal to K. KK is used as an actual argument in subroutine calls to avoid using the DO loop variable K as an argument, which causes problems with some compilers. |
| KT       | Module | Index for top of layers.                                                                                                                   |
| K1       | Module | NLAY-1.                                                                                                                                   |
| LAYCON   | Package| Dimension(80), Layer type code:  
|          |        | 0 - Layer strictly confined.  
|          |        | 1 - Layer strictly unconfined.  
|          |        | 2 - Layer confined/unconfined (transmissivity is constant).  
|          |        | 3 - Layer confined/unconfined (transmissivity varies).  
| NCOL     | Global | Number of columns in the grid.                                                                                                            |
| NLAY     | Global | Number of layers in the grid.                                                                                                             |
| NROW     | Global | Number of rows in the grid.                                                                                                               |
| SC1      | Package| Dimension (NCOL, NROW, NLAY), Primary storage capacity of each cell (S*DELC*DELR).                                                       |
| SC2      | Package| Dimension (NCOL, NROW, NTOP), Secondary storage capacity of each cell in the grid. (NTOP is the number of layers for which LAYCON = 2 or 3.)       |
| TRPY     | Package| Dimension (NLAY), Ratio of transmissivity in the column direction to transmissivity in the row direction.                                                                                          |
Module SBCF1H calculates the horizontal-branch conductances (conductance between nodes) for a layer in which the transmissivity is a function of head (LAYCON = 1 or 3). It calculates the transmissivity internally and calls submodule SBCF1C to calculate the branch conductances. It is called by BCFlFM for each type 1 or type 3 layer at each iteration. Transmissivity is the product of hydraulic conductivity and saturated thickness. The saturated thickness of a completely saturated layer is computed as the elevation of the top (TOP) minus the elevation of the bottom (BOT), the thickness of the layer. For a partially saturated layer, saturated thickness is computed as the head in the cell minus the elevation of the bottom of the layer.

1. For each cell, calculate the transmissivity. DO STEPS 2-6.

2. If the cell is inactive, set the transmissivity equal to zero and move on to the next cell.

3. Calculate the thickness of the saturation. In a strictly unconfined layer, the thickness is the head (HNEW) minus the bottom (BOTTOM). In a confined/unconfined layer, the thickness is the head (HNEW) minus the bottom or the top (TOP) minus the bottom, whichever is greater.

4. Check to see if the saturated thickness is greater than zero.

5. If the thickness is greater than zero, the transmissivity of the cell is the thickness times the hydraulic conductivity.

6. If the saturated thickness is less than zero, the cell is dry. Print a message to that effect, set all branch conductances equal to zero, and set the boundary indicator (IBOUND) equal to zero.

7. Call submodule SBCF1C to calculate the horizontal-branch conductances for the layer.

8. RETURN.
Flow Chart for Module SBCF1H

1. ENTER SBCF1H

2. IS CELL ACTIVE?
   - NO: SET TRANSMISSIVITY EQUAL TO ZERO
   - YES: CALCULATE SATURATED THICKNESS

3. THICKNESS ≥ 0?
   - YES: MULTIPY THICKNESS BY HYDRAULIC CONDUCTIVITY
   - NO: PRINT A MESSAGE SAYING CELL WENT DRY

4. SET CONDUCTANCES EQUAL TO ZERO—SET IBOUND CODE TO "INACTIVE"

5. CALCULATE HORIZONTAL BRANCH CONDUCTANCE

6. RETURN
SUBROUTINE SBCFlH(HNEW, IBOUND, CR, CC, CV, HY, TRPY, DELR, DELC, K, KITER, KSTP, KPER, NCOL, NROW, NLAY, IOUT)

C -----VERSION 1442 31DEC1986 SBCFlH
C
C ***************************************************************
C COMPUTE CONDUCTANCE FROM SATURATED THICKNESS AND HYDRAULIC
C CONDUCTIVITY
C ***************************************************************
C
C SPECIFICATIONS:
C
DOUBLE PRECISION HNEW
C DIMENSION HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
1, CR(NCOL, NROW, NLAY), CC(NCOL, NROW, NLAY), CV(NCOL, NROW, NLAY),
2, HY(NCOL, NROW, NLAY), TRPY(NLAY), DELR(NCOL), DELC(NROW),
3, BOT(NCOL, NROW, NLAY), TOP(NCOL, NROW, NLAY)
C COMMON /FLWCOM/LAYCON(80)
C
C1-----CALCULATE TRANSMISSIVITY AT EACH ACTIVE CELL. TRANSMISSIVITY
C1-----WILL BE STORED TEMPORARILY IN THE CC ARRAY.
DO 200 I=1, NRCM
  DO 200 J=1, NCOL

C C2--- IF CELL IS INACTIVE THEN SET T=0 & MOVE ON TO NEXT CELL.
  IF(IBOUND(J, I, K).NE.0) GO TO 10
  CC(J, I, K)=0.
  GO TO 200
C
C C3-----CALCULATE SATURATED THICKNESS.
  10 HD=HNEW(J, I, K)
    IF(LAYCON(K).EQ.1) GO TO 50
    IF(HD.GT.TOP(J, I, KT)) HD=TOP(J, I, KT)
  50 THCK=HD-BOT(J, I, KB)
C
C C4-----CHECK TO SEE IF SATURATED THICKNESS IS GREATER THAN ZERO.
  IF(THCK.LE.0.) GO TO 100
C
C C5----- IF SATURATED THICKNESS>0 THEN T=K*THICKNESS.
  CC(J, I, K)=THCK*HY(J, I, KB)
C
C6-------WHEN SATURATED THICKNESS < 0, PRINT A MESSAGE AND SET
C6-------TRANSMISSIVITY, IBOUND, AND VERTICAL CONDUCTANCE =0
  100 WRITE(IOUT,150) K, I, J, KITER, KSTP, KPER
    150 FORMAT(1HO,10(***)**, 'NODE', 314, ' (LAYER,ROW,COL) WENT DRY',
160     ' AT ITERATION =', I3, ', TIME STEP =', I3,
170     ' STRESS PERIOD =', I3)

C
C C6------WHEN SATURATED THICKNESS < 0, PRINT A MESSAGE AND SET
C6------TRANSMISSIVITY, IBOUND, AND VERTICAL CONDUCTANCE =0
200 CONTINUE
C
C C7-----COMPUTE HORIZONTAL BRANCH CONDUCTANCES FROM TRANSMISSIVITY
C CALL SBCFlC(CR, CC, TRPY, DELR, DELC, K, NCOL, NROW, NLAY)
C
C C8-------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOT</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NBOT), Elevation of the bottom of each layer. (NBOT is the number of layers for which LAYCON = 1 or 3.)</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,MLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K). This array is used to temporarily hold transmissivity.</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,MLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,MLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).</td>
</tr>
<tr>
<td>DELC</td>
<td>Global</td>
<td>DIMENSION (NROW), Cell dimension in the column direction. DELC(I) contains the width of row I.</td>
</tr>
<tr>
<td>DELR</td>
<td>Global</td>
<td>DIMENSION (NCOL), Cell dimension in the row direction. DELR(J) contains the width of column J.</td>
</tr>
<tr>
<td>HD</td>
<td>Module</td>
<td>Temporary label for an element in HNEW.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,MLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HY</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NBOT), Hydraulic conductivity of the cell. (NBOT is the number of layers where LAYCON = 1 or 3.)</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,MLAY), Status of each cell. &lt; 0, constant-head cell = 0, inactive cell &gt; 0, variable-head cell</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KB</td>
<td>Module</td>
<td>Index for bottom of layers.</td>
</tr>
<tr>
<td>KITER</td>
<td>Global</td>
<td>Iteration counter. Reset at the start of each time step.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>KT</td>
<td>Module</td>
<td>Index for tops of layers.</td>
</tr>
<tr>
<td>LAYCON</td>
<td>Package</td>
<td>DIMENSION(80), Layer type code: 0 - Layer strictly confined. 1 - Layer strictly unconfined. 2 - Layer confined/unconfined (transmissivity is constant) 3 - Layer confined/unconfined (transmissivity varies).</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>THCK</td>
<td>Module</td>
<td>Saturated thickness.</td>
</tr>
<tr>
<td>TOP</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NTOP), Elevation of top of layers. (NTOP is number of layers for which LAYCON = 2 or 3.)</td>
</tr>
<tr>
<td>TRPY</td>
<td>Package</td>
<td>DIMENSION (MLAY), Ratio of transmissivity in the column direction to transmissivity in the row direction.</td>
</tr>
</tbody>
</table>
Narrative for Module SBCF1C

The module SBCF1C calculates horizontal-branch conductances for a layer from transmissivity and cell dimensions. It is called by submodules SBCF1N and SBCF1H. Recall that the branch conductances between two nodes can be expressed by

\[ C = \frac{C_1 C_2}{(C_1 + C_2)}. \]

However, \( C_1 \) and \( C_2 \) can be represented by

\[ C_1 = \frac{T_1 W}{(L_1/2)} \]
\[ C_2 = \frac{T_2 W}{(L_2/2)}. \]

Thus,

\[ C = \frac{2T_1 T_2 W}{(T_1 L_2 + T_2 L_1)}. \]

This equation is used to calculate conductances along rows and columns. When calculating conductance along rows, \( L_1 \) and \( L_2 \) are DELR(J) and DELR(J+1), respectively, and \( W \) is DELC(I). When calculating conductance along columns, \( L_1 \) and \( L_2 \) are DELC(I) and DELC(I+1), respectively, and \( W \) is DELR(J).

Conductance along columns is also multiplied by TRPY(K), the ratio of conductivity in the column direction to conductivity in the row direction in layer K.

1. Process cells one at a time calculating branch conductances from that cell to the one on the right and the one in front.

2. If the transmissivity is equal to zero, set the branch conductance equal to zero and skip to the next cell.

3. If the transmissivity of the cell is not zero and if there is a cell to the right, calculate the branch conductance (CR) along the row.

4. If the transmissivity of the cell is not zero and there is a cell in front, calculate the conductance along the column.

5. RETURN.

Note: Transmissivity, which was temporarily stored in CC, will be lost when conductances are calculated.

\[ \text{CR}(J,I,K) \text{ contains the conductance } \text{CR}_{i,j+1/2,k} \text{ between node } J,I,K \text{ and node } J+1,I,K. \]
Flow Chart for Module SBCF1C

1. FOR EACH CELL
2. T = 0?
   - NO: NEXT
   - YES: SET CONDUCTANCE EQUAL TO ZERO
3. LAST COLUMN?
   - NO: CALCULATE BRANCH CONDUCTANCE TO RIGHT
   - YES: NEXT
4. LAST ROW?
   - NO: CALCULATE BRANCH CONDUCTANCE DOWN COLUMN
   - YES: RETURN
SUBROUTINE SBCFIC(CR, CC, TRPY, DELR, DELC, K, NCOL, NROW, NLAY)

C
C-----VERSION 1334 22AUG1987 SBCFIC
C
C COMPUTE BRANCH CONDUCTANCE USING HARMONIC MEAN OF BLOCK
C CONDUCTANCES -- BLOCK TRANSMISSIVITY IS IN CC UPON ENTRY
C
C ******************************************************
C
C SPECIFICATIONS:
C
C
C DIMENSION CR(NCOL, NROW, NLAY), CC(NCOL, NROW, NLAY)
C 2 , TRPY(NLAY), DELR(NCOL), DELC(NROW)
C
C ******************************************************
C
C
C YX=TRPY(K)*2.
C
C1------FOR EACH CELL CALCULATE BRANCH CONDUCTANCES FROM THAT CELL
C1------TO THE ONE ON THE RIGHT AND THE ONE IN FRONT.
C1
C
DO 40 I=1, NROW

    DO 40 J=1, NCOL

    T1=CC(J, I, K)

C
C2------IF T=0 THEN SET CONDUCTANCE EQUAL TO 0. GO ON TO NEXT CELL.
C2

    IF(T1.NE.0.) GO TO 10

    CR(J, I, K)=0.

    GO TO 40

C
C3------IF THIS IS NOT THE LAST COLUMN(RIGHTMOST) THEN CALCULATE
C3------BRANCH CONDUCTANCE IN THE ROW DIRECTION (CR) TO THE RIGHT.
C3

10 IF(J.EQ.NCOL) GO TO 30

    T2=CC(J+1, I, K)

    CR(J, I, K)=2.*T2*T1*DELC(I)/(T1*DELR(J+1)+T2*DELR(J))

C
C4------IF THIS IS NOT THE LAST ROW(FRONTMOST) THEN CALCULATE
C4------BRANCH CONDUCTANCE IN THE COLUMN DIRECTION (CC) TO THE FRONT.
C4

30 IF(I.EQ.NROW) GO TO 40

    T2=CC(J, I+1, K)

    CC(J, I, K)=YX*T2*T1*DELR(J)/(T1*DELC(I+1)+T2*DELC(I))

40 CONTINUE

C
C5------RETURN

RETURN

END

5-79
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K). This array is used to temporarily hold transmissivity.</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>DELC</td>
<td>Global</td>
<td>DIMENSION (NROW), Cell dimension in the column direction. DELC(I) contains the width of row I.</td>
</tr>
<tr>
<td>DELR</td>
<td>Global</td>
<td>DIMENSION (NCOL), Cell dimension in the row direction. DELR(J) contains the width of column J.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>TRPY</td>
<td>Package</td>
<td>DIMENSION (NLAY), Ratio of transmissivity in the column direction to transmissivity in the row direction.</td>
</tr>
<tr>
<td>T1</td>
<td>Module</td>
<td>Temporary field for CC(J,I,K).</td>
</tr>
<tr>
<td>T2</td>
<td>Module</td>
<td>Temporary field for CC(J+1,I,K).</td>
</tr>
<tr>
<td>YX</td>
<td>Module</td>
<td>TRPY(K)*2.</td>
</tr>
</tbody>
</table>
Narrative for Module SBCF1B

This module calculates flow across cell faces. It is called by module BCFlBD when the user has requested cell-by-cell flow terms. It performs its tasks in the following order:

1. Clear the buffer (BUFF) in which cell-by-cell flow terms are gathered as they are calculated.

2. For each cell, calculate the flow in the row direction through the right face of the cell and store it in the buffer.

3. Call utility module UBUDSV to write the contents of the buffer.

4. Clear the buffer (BUFF) in which cell-by-cell flow terms are gathered as they are calculated.

5. For each cell, calculate the flow in the column direction through the front face of the cell and store it in the buffer.

6. Call utility module UBUDSV to write the contents of the buffer.

7. Clear the buffer (BUFF) in which cell-by-cell flow terms are gathered as they are calculated.

8. For each cell, calculate the flow in the vertical direction through the lower face of the cell and store it in the buffer.

9. Call utility module UBUDSV to write the contents of the buffer.

10. RETURN.
BUFFER: the buffer is an array with one element for each cell in the grid. It is used to store the results of cell-by-cell calculations until all cells have been processed. The contents of the buffer are then recorded as a unit.
SUBROUTINE SBCF1B(HNEW, IBOUND, CR, CC, CV, TOP, NCOL, NROW, NLAY,
1       KSTP, KPER, IBCFCB, BUFF, IOUT)
C-----VERSION 1548 12MAY1987 SBCF1B
C
C******************************************************************************
C COMPUTE FLOW ACROSS EACH CELL WALL
C******************************************************************************

C SPECIFICATIONS:
C---------------------------------------------------------------
CHARACTER*4 TEXT
DOUBLE PRECISION HNEW, HD

DIMENSION HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
1       CR(NCOL, NROW, NLAY), CC(NCOL, NROW, NLAY),
2       CV(NCOL, NROW, NLAY), TOP(NCOL, NROW, NLAY),
3       BUFF(NCOL, NROW, NLAY)

COMMON /FLWCOM/LAYCON(80)
DIMENSION TEXT(12)

DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4), TEXT(5), TEXT(6), TEXT(7),
1       TEXT(8), TEXT(9), TEXT(10), TEXT(11), TEXT(12)
2       '/FLOW', 'RIG', 'HT F', 'ACE',
2       '/FLOW', 'FR', 'F', 'ACE', 'FLOW', 'LOW', 'FR', 'ACE '/

NCML=NCOL-1
IF(NCML.LT.1) GO TO 405

C1-----CLEAR THE BUFFER
DO 310 K=1, NLAY
DO 310 I=1, NROW
DO 310 J=1, NCOL
BUFF(J, I, K)=0.
310 CONTINUE

C2-----FOR EACH CELL CALCULATE FLOW THRU RIGHT FACE & STORE IN BUFFER
DO 400 K=1, NLAY
DO 400 I=1, NROW
DO 400 J=1, NCM1
IF((IBOUND(J, I, K).LE.0) .AND. (IBOUND(J+1, I, K).LE.0)) GO TO 400
HDIFF=HNEW(J, I, K)-HNEW(J+1, I, K)
BUFF(J, I, K)=HDIFF*CR(J, I, K)
400 CONTINUE

C3-----RECORD CONTENTS OF BUFFER
CALL UBUDSV(KSTP, KPER, TEXT(1), IBCFCB, BUFF, NCOL, NROW, NLAY, IOUT)
C4------CLEAR THE BUFFER
405 NRM1=NROW-1
   IF(NRM1.LT.1) GO TO 505
   DO 410 K=1,NLAY
   DO 410 I=1,NROW
   DO 410 J=1,NCOL
   BUFF(J,I,K)=0.
   410 CONTINUE

C5------FOR EACH CELL CALCULATE FLOW THRU FRONT FACE & STORE IN BUFFER
DO 500 K=1,NLAY
   DO 500 I=1,NRM1
   DO 500 J=1,NCOL
   IF((IBOUND(J,I,K),LE.0) .AND. (IBOUND(J,I+1,K),LE.0)) GO TO 500
   HDIFF=HNEW(J,I,K)-HNEW(J,I+1,K)
   BUFF(J,I,K)=HDIFF*CC(J,I,K)
      500 CONTINUE

C6------RECORD CONTENTS OF BUFFER.
   CALL UBUDSV(KSTP,KPER,TEXT(5),IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
505 NLM1=NLAY-1
   IF(NLM1.LT.1) GO TO 1000

C7------CLEAR THE BUFFER
   DO 510 K=1,NLAY
   DO 510 I=1,NROW
   DO 510 J=1,NCOL
   BUFF(J,I,K)=0.
      510 CONTINUE

C8------FOR EACH CELL CALCULATE FLOW THRU LOWER FACE & STORE IN BUFFER
   KT=0
   DO 600 K=1,NLM1
   DO 600 I=1,NROW
   DO 600 J=1,NCOL
   IF((IBOUND(J,I,K),LE.0) .AND. (IBOUND(J,I,K+1),LE.0)) GO TO 600
   HD=HNEW(J,I,K+1)
   IF(LAYCON(K+1),NE.3 .AND. LAYCON(K+1),NE.2) GO TO 580
   TMP=HD
   IF(TMP,LT,TOP(J,I,KT+1)) HD=TOP(J,I,KT+1)
      580 CONTINUE
   HDIFF=HNEW(J,I,K)-HD
   BUFF(J,I,K)=HDIFF*CV(J,I,K)
   600 CONTINUE

C9------RECORD CONTENTS OF BUFFER.
   CALL UBUDSV(KSTP,KPER,TEXT(9),IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)

C10------RETURN
1000 RETURN
END
List of Variables for Module SBCFlB

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K).</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).</td>
</tr>
<tr>
<td>HD</td>
<td>Module</td>
<td>Temporary field for head.</td>
</tr>
<tr>
<td>HDIFF</td>
<td>Module</td>
<td>Head difference between two adjacent nodes.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell. &gt; 0, unit number on which the cell-by-cell flow terms will be recorded whenever ICBCFL is set. = 0, cell-by-cell flow terms will be not be printed or recorded &lt; 0, flow from each constant-head cell will be printed whenever ICBCFL is set.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>KT</td>
<td>Module</td>
<td>Index for tops of layers.</td>
</tr>
<tr>
<td>LAYCON</td>
<td>Package</td>
<td>DIMENSION(80), Layer type code: 0 - Layer strictly confined. 1 - Layer strictly unconfined. 2 - Layer confined/unconfined (transmissivity is constant). 3 - Layer confined/unconfined (transmissivity varies).</td>
</tr>
<tr>
<td>NCM1</td>
<td>Module</td>
<td>NCOL-1.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NLM1</td>
<td>Module</td>
<td>NLAY-1.</td>
</tr>
<tr>
<td>NRM1</td>
<td>Module</td>
<td>NRw-1.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with array data.</td>
</tr>
<tr>
<td>TMP</td>
<td>Module</td>
<td>Temporary field for head.</td>
</tr>
<tr>
<td>TOP</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NTOP), Elevation of top of layers. (NTOP is number of layers for which LAYCON = 2 or 3.)</td>
</tr>
</tbody>
</table>
Narrative for Module SBCFlF

This module calculates flow from constant-head cells. The flows are accumulated by sign to get flow into (CHIN) and out of (CHOUT), the flow field for inclusion in the overall volumetric budget. The flows are also accumulated by cell to get the total flow from each constant-head cell on a cell-by-cell basis. Module SBCFlF is called by module BCF1BD and calls utility module UBUDSV.

Module SBCFlF performs its functions in the following order:

1. Clear the fields CHIN and CHOUT in which flow into and out of the flow field, respectively, will be accumulated.

2. If cell-by-cell flow terms will be recorded, clear the buffer (BUFF) in which they will be stored as they are calculated.

3. For each cell, calculate the flow to and from constant-head cells. DO STEPS 4-12.

4. If the cell is not a constant-head cell, skip further processing and go on to the next cell.

5. Clear the six fields corresponding to the six faces through which the flows will be calculated.

6. For each face of the cell, calculate the flow out of the cell through that face (STEPS 7-11).

7. If there is not a variable-head cell which shares the face, go on to the next face.
8. Calculate the flow through the face into the adjacent cell.

9. Test the sign of the flow to see if it is positive (into the adjacent variable-head cell from the constant-head cell) or negative (out of the adjacent variable-head cell into the constant-head cell). GO TO EITHER STEP 10 OR 11.

10. If the sign is negative, add the flow rate to CHOUT (flow out of the flow domain).

11. If the sign is positive, add the flow rate to CHIN (flow out of the flow domain).

12. Add together the flow terms \( x_1, x_2, x_3, x_4, x_5, x_6 \) corresponding to the six faces and leave in the field RATE.

13. If the user specified a negative number for IBCFCB, and ICBCFL ≠ 0, print the flows (RATE) from the constant-head cell into the aquifer.

14. If the cell-by-cell terms are to be recorded, add the six flow rates out of the cell and store them in the buffer until all cells are finished.

15. If the cell-by-cell terms are to be recorded, call utility module UBUDSV to record them.

16. Put flow rates, into and out of the flow domain from constant-head cells, into the VBVL array for inclusion in the overall volumetric budget. Put labels for those budget terms into VBNM.

17. RETURN.
Flow Chart for Module SBCF1F

CHIN is a field in which flows, into the flow domain from constant-head cells, will be accumulated.

CHOUT is a field in which flows, out of the flow domain to constant-head cells, will be accumulated.

BUFF is a buffer in which cell-by-cell flow terms will be stored as they are calculated prior to recording them on disk.

INTERNAL CELLS are those in which head varies. They are in opposition to EXTERNAL CELLS (inactive or constant head) which are on or outside of a boundary.
SUBROUTINE SBCF1F(VBNM, VBVL, MSUM, HNEW, IBOUND, CR, CC, CV,
  1 TOP, DELT, NCOL, NROW, NLAY, KSTP, KPER, IBD, IBCFCB, IBCFL, 
  2 BUFF, IOUT)

C-----VERSION 1549 12MAY1987 SBCF1F
C
C
C
C
C
C
C
C
C
C
C*************************************************************
COMPUTE FLCK FROM CONSTANT HEAD NODES
*************************************************************

SPECIFICATIONS:
****************************************************************
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW, HD

DIMENSION HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
  1 CR(NCOL, NROW, NLAY), CC(NCOL, NROW, NLAY),
  2 CV(NCOL, NROW, NLAY), VBNM(4, 20), VBVL(4, 20),
  3 TOP(NCOL, NROW, NLAY), BUFF(NCOL, NROW, NLAY)

COMMON /FLWCOM/LAYCON(80)
DIMENSION TEXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /' C', 'ONST', 'ANT ', 'HEAD'/
****************************************************************

C1-------CLEAR BUDGET ACCUMULATORS
CHIN=0.
CHOUT=0.

C2-------CLEAR BUFFER IF CELL-BY-CELL FLOW TERM IBD IS SET
IF (IBD.EQ.0) GO TO 8
DO 5 K=1, NLAY
DO 5 I=1, NROW
DO 5 J=1, NCOL
BUFF(J, I, K)=0.
5 CONTINUE

C3-------FOR EACH CELL IF IT IS CONSTANT HEAD COMPUTE FLOW ACROSS 6
C3-------FACES.
8 KT=0
DO 200 K=1, NLAY
LC=LAYCON(K)
IF (LC.EQ.3 .OR. LC.EQ.2) KT=KT+1
DO 200 I=1, NROW
DO 200 J=1, NCOL

C4-------IF CELL IS NOT CONSTANT HEAD SKIP IT & GO ON TO NEXT CELL.
IF (IBOUND(J, I, K).GE.0) GO TO 200

C5-------CLEAR FIELDS FOR SIX FLOW RATES.
X1=0.
X2=0.
X3=0.
X4=0.
X5=0.
X6=0.

C6-------FOR EACH FACE OF THE CELL CALCULATE FLOW THROUGH THAT FACE
C6-------OUT OF THE CONSTANT HEAD CELL AND INTO THE FLOW DOMAIN.
C6-------COMMENTS 7-11 APPEAR ONLY IN THE SECTION HEADED BY COMMENT 6A
C6-------BUT THEY APPLY IN A SIMILAR MANNER TO THE SECTIONS HEADED
C6-------BY COMMENTS 6A-6F.
C

5-89
CALCULATE FLOW THROUGH THE LEFT FACE

IF (J .EQ. 1) GO TO 30
IF (IBOUND(J-I-I,K).LE.0) GO TO 30
HDIF=HNEW(J-I,K)-HNEW(J-I-I,K)

CALCULATE FLOW THROUGH THIS FACE INTO THE ADJACENT CELL.

X1=HDIF*CR(J-I-I,K)

IF(J.EQ.1) GO TO 30
IF(IBOUND(J-I-I,K).LE.0) GO TO 30
HDIF=HNEW(J-I-K)-HNEW(J-I-I,K)
X2=HDIF*CR(J-I,K)
IF(X2) 40,60,50
40 CHOUT=CHOUT-X2
GO TO 60
50 CHIN=CHIN+X2

CALCULATE FLOW THROUGH THE RIGHT FACE.

30 IF(J.EQ.NCOL) GO TO 60
IF (IBOUND(J+I-I,K).LE.0) GO TO 60
HDIF=HNEW(J+I,K)-HNEW(J+I-I,K)
X3=HDIF*CC(J,I-I,K)
IF(X3) 70,120,80
70 CHOUT=CHOUT-X3
GO TO 120
80 CHIN=CHIN+X3

CALCULATE FLOW THROUGH THE BACK FACE.

90 IF(I.EQ.NROW) GO TO 120
IF (IBOUND(J+I-I,K).LE.0) GO TO 120
HDIF=HNEW(J+I,K)-HNEW(J+I-I,K)
X4=HDIF*CC(J,I-K)
IF(X4) 100,120,110
100 CHOUT=CHOUT-X4
GO TO 120
110 CHIN=CHIN+X4

CALCULATE FLOW THROUGH THE UPPER FACE.

120 IF(K.EQ.1) GO TO 150
IF (IBOUND(J+I-I,K-1).LE.0) GO TO 150
HD=HNEW(J+I,K)
IF (LC.NE.3 .AND. LC.NE.2) GO TO 122
TMP=HD
IF(TMP.LT.TOP(J,I,K)) HD=TOP(J,I,K)
122 HDIFF=HD-HNEW(J+I,K-1)
X5=HDIFF*CV(J+I,K-1)
IF(X5) 130,150,140
130 CHOUT=CHOUT-X5
GO TO 150
140 CHIN=CHIN+X5
C6F----CALCULATE FLOW THROUGH THE LOWER FACE.
150 IF(K.EQ.NLAY) GO TO 180
   IF(IBOUND(J,I,K+1).LE.0) GO TO 180
      HD=HNEW(JrIrK+1)
      IF(LAYCON(K+1).NE.3 .AND. LAYCON(K+1).NE.2) GO TO 152
   TMP=HD
   IF(TMP.LT.TOP(JrIrKT+1)) HD=TOP(JrIrKT+1)
152 HDIFF=HNEW(JrIrK)-HD
   X6=HDIFF*CV(JrIrK)
   IF(X6) 160,180,170
160 CHDUT=CHOUT-X6
   GO TO 180
170 CHIN=CHIN+X6
C
C12-----SUM UP FLOWS THROUGH SIX SIDES OF CONSTANT HEAD CELL.
   RATE=X1+X2+X3+X4+X5+X6
C
C13-----PRINT THE INDIVIDUAL RATES IF REQUESTED(IBCFCB<0).
   IF(IBCFCB.LT.0.AND.ICBCF<.0) WRITE(IOUT,900) (TEXT(N),N=1,4),
      1  KPER,KSTP,K,I,J,RATE
900 FORMAT(I4,4A4,' PERIOD',I3,' STEP',I3,' LAYER',I3,
     1  ' ROW',I4,' COL',I4,' RATE ',F15.7)
C
C14-----IF CELL-BY-CELL FLAG SET STORE SUM OF FLOWS FOR CELL IN BUFFER
   IF(IBC.0) BUFF(JrIrK)=RATE
   200 CONTINUE
C
C15-----IF CELL-BY-CELL FLAG SET THEN RECORD CONTENTS OF BUFFER
   IF(IBC.0) CALL UBUDSY(KSTP,KPER,TEXT(1),
      1  IBCFCB,BUFF,NOOL,NROW,NLAY,IOUT)
C
C16-----SAVE TOTAL CONSTANT HEAD FLOWS AND VOLUMES IN VBVL TABLE
C16-----FOR INCLUSION IN BUDGET, PUT LABELS IN VBNM TABLE.
   VBVL(1,MSUM)=VBVL(1,MSUM)+CHIN*DELT
   VBVL(2,MSUM)=VBVL(2,MSUM)+CHOUT*DELT
   VBVL(3,MSUM)=CHIN
   VBVL(4,MSUM)=CHOUT
C
C17-----RETURN
   RETURN
END
### List of Variables for Module SBCFlF

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY). Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY). Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K).</td>
</tr>
<tr>
<td>CHIN</td>
<td>Module</td>
<td>Accumulator for flow into the model area from constant heads.</td>
</tr>
<tr>
<td>COUT</td>
<td>Module</td>
<td>Accumulator for flow out of the model area to constant heads.</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>HD</td>
<td>Module</td>
<td>Temporary field containing a value from HNEW.</td>
</tr>
<tr>
<td>HDIFF</td>
<td>Module</td>
<td>Head difference between one node and the adjacent node.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBCFCB</td>
<td>Package</td>
<td>Flag and a unit number.</td>
</tr>
<tr>
<td>IBD</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td>ICBCFL</td>
<td>Global</td>
<td>Flag.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>Variable</td>
<td>Range</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td>------------</td>
</tr>
<tr>
<td>KT</td>
<td>Module</td>
<td>Index for tops of layers.</td>
</tr>
<tr>
<td>LAYCON</td>
<td>Package</td>
<td>DIMENSION(80), Layer type code:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - Layer strictly confined.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - Layer strictly unconfined.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - Layer confined/unconfined (transmissivity is constant).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - Layer confined/unconfined (transmissivity varies).</td>
</tr>
<tr>
<td>LC</td>
<td>Module</td>
<td>Temporary label for an element of LAYCON.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>RATE</td>
<td>Module</td>
<td>Flow from the constant-head cell into the aquifer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Reverse the sign to get the flow from the aquifer into the constant-head cell.)</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with array data.</td>
</tr>
<tr>
<td>TOP</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW,NTOP), Elevation of top of layers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NTOP is the number of layers for which LAYCON = 2 or 3.)</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For flow component N, the values in VBVL are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1,N), Rate for the current time step into the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(2,N), Rate for the current time step out of the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,N), Volume into the flow field during simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4,N), Volume out of the flow field during simulation.</td>
</tr>
<tr>
<td>X1</td>
<td>Module</td>
<td>Flow through the left face.</td>
</tr>
<tr>
<td>X2</td>
<td>Module</td>
<td>Flow through the right face.</td>
</tr>
<tr>
<td>X3</td>
<td>Module</td>
<td>Flow through the back face.</td>
</tr>
<tr>
<td>X4</td>
<td>Module</td>
<td>Flow through the front face.</td>
</tr>
<tr>
<td>X5</td>
<td>Module</td>
<td>Flow through the upper face.</td>
</tr>
<tr>
<td>X6</td>
<td>Module</td>
<td>Flow through the lower face.</td>
</tr>
</tbody>
</table>
Chapter 6
RIVER PACKAGE
Conceptualization and Implementation

Rivers and streams contribute water to the ground-water system or drain water from it depending on the head gradient between the stream and the ground-water regime. The purpose of the River Package is to simulate the effects of flow between surface-water features and ground-water systems. To accomplish this, terms representing seepage to or from the surface features must be added to the ground-water flow equation (equation (26)) for each cell affected by the seepage.

Figure 32 shows a stream divided into reaches so that each reach is completely contained in a single cell. Stream aquifer seepage is simulated between each reach and the model cell that contains that reach.

The cross-section of figure 33-a shows a situation in which the open water of a stream is separated from the ground-water system by a layer of low permeability streambed material. Figure 33-b shows an idealization of this system in which the stream-aquifer interconnection is represented as a simple conductance through which one-dimensional flow occurs. The system of figure 33 is helpful in conceptualizing and describing the simulation of stream-aquifer interaction; however, it must be recognized that, in many instances, no discrete low-permeability streambed layer is present. The techniques of simulation developed through the conceptualization of figure 33 can still be applied to represent these situations, provided the proper interpretation is placed on the various terms and parameters that are used.

Figure 34 shows an isolated view of the idealized streambed conductance of figure 33-b, as it crosses an individual cell. The length of the
Figure 32.—Discretization of a stream into reaches. Some small reaches are ignored.
Figure 33.—(a) Cross section of an aquifer containing a stream and (b) Conceptual representation of stream-aquifer interconnection in simulation.
Thickness of Riverbed $M$

Width of River $W$

Streambed Conductance $= KLW/M$

Figure 34.—Idealization of streambed conductance in an individual cell.
conductance block is taken as the length of the stream, $L$, as it crosses the node; the width is taken as the stream width, $W$; the distance of flow is taken as the thickness, $M$, of the streambed layer; and the hydraulic conductivity of the streambed material is designated $K$. The assumption is made that measurable head losses between the stream and the aquifer are limited to those across the streambed layer itself—that is, that there is no significant head loss between the bottom of the streambed layer and the point represented by the underlying model node. It is further assumed that the underlying model cell remains fully saturated—that is, that its water level does not drop below the bottom of the streambed layer. Under these assumptions, flow between the stream and the ground-water system is given by

$$Q_{RIV} = \frac{KLM}{M} (HRIV - h_{i,j,k}) \quad (63-a)$$

or

$$Q_{RIV} = CRIV (HRIV - h_{i,j,k}) \quad (63-b)$$

where $Q_{RIV}$ is the flow between the stream and the aquifer, taken as positive if it is directed into the aquifer; $HRIV$ is the head in the stream; $CRIV$ is the hydraulic conductance of the stream-aquifer interconnection ($KLW/M$), and $h_{i,j,k}$ is the head at the node in the cell underlying the stream reach.

If the assumption is satisfied that all significant head loss occurs across a discrete streambed layer, the application of equations (63) is straightforward. More frequently, however, equations (63) must be applied to situations in which no discrete streambed layer can be identified, or in which head losses are not restricted to those across such a layer. In these cases, the task is to formulate a single conductance term, $CRIV$, which can be used in (63-b) to relate flow between the stream and the depth
represented by node $i,j,k$ to the corresponding head difference. This flow is in general a three-dimensional process, and its representation through a single conductance term can never be more than approximate. If reliable field measurements of stream seepage and associated head difference are available, they may be used to calculate an effective conductance. Otherwise, a conductance value must be chosen more or less arbitrarily and adjusted during model calibration. Certain rules can be formulated to guide the initial choice of conductance. For example, the assumed cross-sectional area of flow should normally be of the same order of magnitude as the product of channel width and stream reach length within the cell; the assumed distance of flow should not exceed the vertical interval between the streambed and node $i,j,k$; and, if distinct layers can be recognized within this interval, these should normally be treated as conductances in series in formulating an equivalent conductance. In general, however, it should be recognized that formulation of a single conductance term to account for a three-dimensional flow process is inherently an empirical exercise, and that adjustment during calibration is almost always required.

Equations (63) normally provide an acceptable approximation of stream-aquifer interaction over a certain range of aquifer head values. In most cases, however, if water levels in the aquifer fall below a certain point, seepage from the stream ceases to depend on head in the aquifer. This can be visualized by returning to the concept of a discrete streambed layer. Figure 35-a shows the situation described by equations (63); water level in the aquifer is above the bottom of the streambed layer, and flow through that layer is proportional to the head difference between the stream and the aquifer. In figure 35-b, water level in the aquifer has fallen below the bottom of the streambed layer, leaving an unsaturated interval beneath that
Figure 35.—Cross sections showing the relation between head at the bottom of the streambed layer and head in the cell. Head in the cell is equal to the water-table elevation.
layer; if it is assumed that the streambed layer itself remains saturated, 
the head at its base will simply be the elevation at that point. If this 
elevation is designated RBOT, the flow through the streambed layer is given 
by

$$QRIV = CRIV \cdot (HRIV - RBOT)$$  \hspace{1cm} (64)

where QRIV, CRIV, and HRIV are as defined for equation (63-b). Obviously, 
further declines in head below RBOT produce no increase in flow through the 
streambed layer; the flow simply retains the constant value given by equa-
tion (64), as long as head remains below RBOT. The model described herein 
utilizes these concepts in simulating stream-aquifer interaction—that is, 
flow between a stream and a node i,j,k is simulated according to the equa-
tion set

$$QRIV = CRIV \cdot (HRIV - h_{i,j,k}), \quad h_{i,j,k} > RBOT$$

$$QRIV = CRIV \cdot (HRIV - RBOT), \quad h_{i,j,k} \leq RBOT$$  \hspace{1cm} (65)

Figure 36 shows a graph of flow between the stream and cell i,j,k as 
a function of the head, h_{i,j,k}, as calculated using equations 65. Flow is 
zero when h_{i,j,k} is equal to the water level in the stream, HRIV. For 
higher values of h_{i,j,k}, flow is negative, that is, into the stream; for 
lower values of h_{i,j,k}, flow is positive, that is, into the aquifer. This 
positive flow increases linearly as h_{i,j,k} decreases, until h_{i,j,k} reaches 
RBOT; thereafter the flow remains constant.

A relationship similar to that of equations (65) and figure (36) 
generally prevails in stream-aquifer interaction whether or not a discrete 
streambed layer is present. For example, once a break in saturation occurs 
between the stream and the aquifer, seepage from the stream to the aquifer 
must become independent of head in the aquifer. In most cases, moreover, 
this independence is established even before a break in saturation occurs.
Figure 36.—Plot of flow, QRIV, from a stream into a cell as a function of head, h, in the cell where RBOT is the elevation of the bottom of the streambed and HRIV is the head in the stream.
Figure 37 shows a situation in which levels in the aquifer have fallen far enough below a stream so that only a narrow saturated connection exists between the streambed and the regional water table. Examination of figure 37 will show that the head gradient in the saturated connection must be approximately unity, and that further lowering of the regional water table will not increase this gradient. Thus, once a condition similar to that in figure 37 is established, seepage from the stream is independent of further head decline in the aquifer. The situation shown in figure 37 is itself an overimplification of field conditions, which may often involve complex patterns of saturated and unsaturated material beneath the stream. In all situations, however, seepage from the stream must at some point become independent of head in the aquifer, as that head continues to decline.

If hydrologic conditions indicate that seepage from a stream will increase as the local water table elevation declines, but will reach the limiting condition illustrated in figure 37 when the water table reaches an elevation $h_l$, RBOT should be taken equal to $h_l$. Because the vertical hydraulic gradient beneath the stream is approximately one under the conditions of figure 37, seepage from the stream into cell $i,j,k$ is given approximately by the product $KLW$, where $K$ is the vertical hydraulic conductivity of the material in the saturated column beneath the stream, and again $L$ is the length of the stream as it crosses cell $i,j,k$, and $W$ is stream width. A value of the stream-aquifer conductance term, $CRIV$, consistent with this estimated value of limiting seepage and with the selected value of RBOT can be obtained by substituting $KLW$ for $QRIV$ in equation (64) and solving for $CRIV$. This yields

$$ CRIV = \frac{KLW}{HRIV - RBOT} $$

(66)
Figure 37.—Limiting seepage from a stream at unit hydraulic gradient
In summary, if the limiting condition of stream seepage is expected to follow the model of figure 37, RBOT should be chosen as the water table elevation at which the transition to this limiting seepage can be expected, and CRIV may be calculated from equation (66). The model simulation technique based on equations (65) should then provide a reasonable approximation to the stream-aquifer interaction.

The simplified model of stream-aquifer interaction utilized here assumes that this interaction is independent of the location of the stream reach within the cell, and that the level of water in the stream is uniform over the reach, and constant over each stress period. The latter assumption implies that conditions of flow in the stream do not vary significantly during the stress period--for example, that the stream does not go dry or overflow its banks, or that such events are of such short duration as to have no effect on stream-aquifer interaction.

Data describing each river are specified by the user for each stress period. Input consists of six entries for each river reach, specifying the layer, row, and column of the cell containing the reach, and the three parameters needed to calculate seepage--stream level or stage (HRIV), the conductance of the stream-aquifer interconnection (CRIV), and the "bottom elevation," or level at which the limiting value of stream seepage is attained (RBOT).

At the start of each iteration, terms representing river seepage are added to the flow equation for each cell containing a river reach. The choice of which river seepage equation to use, equation 63 or equation 64, is made by comparing the most recent value of head at the cell to the value
of RBOT for the reach. Since this process is done at the start of each iteration, the most current value of head (HNEW) is the value from the previous iteration. Thus, the check for which river seepage equation to use lags behind the seepage calculations by one iteration. If equation 63 is selected, the term -CRIV is added to the term HCOF and the term -CRIV*HRIV is added to RHS. If equation 64 is selected, the term -CRIV (HRIV - RBOT) is added to the term RHS.
River Package Input

Input to the River (RIV) Package is read from the unit specified in IUNIT(4).

FOR EACH SIMULATION

RIV1AL

1. Data: MXRIVR IRIVCB
   Format: I10 I10

FOR EACH STRESS PERIOD

RIV1RP

2. Data: ITMP
   Format: I10

3. Data: Layer Row Column Stage Cond Rbot
   Format: I10 I10 I10 F10.0 F10.0 F10.0

  (Input item 3 normally consists of one record for each river reach. If ITMP is negative or zero, item 3 is not read.)

Explanation of Fields Used in Input Instructions

MXRIVR--is the maximum number of river reaches active at one time.

IRIVCB--is a flag and a unit number.

If IRIVCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL (see Output Control) is set.

If IRIVCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IRIVCB < 0, river leakage for each reach will be printed whenever ICBCFL is set.

ITMP--is a flag and a counter.

If ITMP < 0, river data from the last stress period will be reused.

If ITMP > 0, ITMP will be the number of reaches active during the current stress period.

Layer--is the layer number of the cell containing the river reach.

Row--is the row number of the cell containing the river reach.
Column—is the column number of the cell containing the river reach.

Stage—is the head in the river.

Cond—is the riverbed hydraulic conductance.

Rbot—is the elevation of the bottom of the riverbed.
<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(MXRIVR, IRIVCB)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>(ITMP) FOR FIRST STRESS PERIOD</td>
<td>3 55</td>
</tr>
<tr>
<td>3</td>
<td>(LAYER, ROW, COLUMN, STAGJ, COND, RBOT) FOR FIRST REACH</td>
<td>2 6 4 220. 0.7 212.</td>
</tr>
<tr>
<td>3</td>
<td>(LAYER, ROW, COLUMN, STAGJ, COND, RBOT) FOR SECOND REACH</td>
<td>2 7 4 225. 0.9 217.</td>
</tr>
<tr>
<td>3</td>
<td>(LAYER, ROW, COLUMN, STAGJ, COND, RBOT) FOR THIRD REACH</td>
<td>2 5 4 210. 0.8 200.</td>
</tr>
<tr>
<td>2</td>
<td>(ITMP) FOR SECOND STRESS PERIOD</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>(ITMP) FOR THIRD STRESS PERIOD</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>(ITMP) FOR FOURTH STRESS PERIOD</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>(LAYER, ROW, COLUMN, STAGJ, COND, RBOT) FOR FIRST REACH</td>
<td>2 5 4 210. 0.8 210.</td>
</tr>
<tr>
<td>3</td>
<td>(LAYER, ROW, COLUMN, STAGJ, COND, RBOT) FOR SECOND REACH</td>
<td>2 6 4 220. 0.7 212.</td>
</tr>
<tr>
<td>2</td>
<td>(ITMP) FOR FIFTH STRESS PERIOD</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>(ITMP) FOR SIXTH STRESS PERIOD</td>
<td>-1</td>
</tr>
</tbody>
</table>
Module Documentation for the River Package

The River Package (RIV1) consists of four modules, all of which are called by the MAIN program. The modules are:

**RIV1AL**
Allocates space for a list (RIVR) which will contain an entry for each river reach. Each entry will consist of the location of the cell containing the reach, riverhead, conductance of the riverbed, and the elevation of the bottom of the riverbed.

**RIV1RP**
Reads, for each river reach, the location of the cell containing the reach, riverhead, conductance of the riverbed, and elevation of the bottom of the riverbed.

**RIV1FM**
Adds, for each river reach, the appropriate terms to the accumulators HCOF and RHS.

**RIV1BD**
Calculates the rates and accumulated volume of river leakage into and out of the flow system.
Narrative for Module RIVIAL

This module allocates space in the X array to store the list of river reaches.

1. Print a message identifying the package and initialize NRIIVER (number of river reaches).

2. Read and print MXRIVR (the maximum number of river reaches) and IRIVCB (the unit number for saving cell-by-cell flow terms or a flag indicating whether cell-by-cell flow terms should be printed).

3. Set LCRIVR, which will point to the first element in the river list (RIVR), equal to ISUM, which is currently pointing to the first unallocated element in the X array.

4. Calculate the amount of space needed for the river list (six values for each reach--row, column, layer, riverhead, riverbed conductance, and riverbed bottom elevation) and add it to ISUM.

5. Print the number of elements in the X array used by the River Package.

6. RETURN.
NRIVER is the number of river reaches being simulated at any given time.

MXRIVR is the maximum number of river reaches simulated.

IRIVCB is a flag and a unit number.

If IRIVCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL (see RIV1BD module) is set.

If IRIVCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IRIVCB < 0, river leakage for each reach will be printed whenever ICBCFL is set.

LCRIVR is the location in the X array of the list of river data (RIVR).
SUBROUTINE RIVAL(ISUM, LENX, LCRIVR, MXRIVR, NRIVER, IN, IOUT, 1  IRIVCB)
C
C-------VERSION 1554 12MAY1987 RIVAL
C
C ALLOCATE ARRAY STORAGE FOR RIVERS
C
C
C SPECIFICATIONS:
C
C
C1-------IDENTIFY PACKAGE AND INITIALIZE NRIVER.
C
WRITE(IOUT,1)IN
1 FORMAT(1HO,'RIV1 -- RIVER PACKAGE, VERSION 1, 9/1/87', 1! INPUT READ FROM UNIT',I3)
NRIVER=0
C
C2-------READ & PRINT MXRIVR & IRIVCB(UNIT OR FLAG FOR C-B-C FLOWS)
READ(IN,2)MXRIVR,IRIVCB
2 FORMAT(21I0)
WRITE(IOUT,3)MXRIVR
3 FORMAT(1H , 'MAXIMUM OF',I5,' RIVER NODES')
   IF(IRIVCB.GT.0) WRITE(IOUT,9) IRIVCB
9 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE RECORDED ON UNIT',I3)
   IF(IRIVCB.LT.0) WRITE(IOUT,8)
8 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE PRINTED')
C
C3-------SET LCRIVR EQUAL TO ADDRESS OF FIRST UNUSED SPACE IN X.
LCRIVR=ISUM
C
C4-------CALCULATE AMOUNT OF SPACE USED BY RIVER LIST.
ISP=6*MXRIVR
ISUM=ISUM+ISP
C
C5-------PRINT AMOUNT OF SPACE USED BY RIVER PACKAGE.
WRITE (IOUT,4)ISP
4 FORMAT(1X,I8,' ELEMENTS IN X ARRAY ARE USED FOR RIVERS')
   ISUM1=ISUM+1
WRITE(IOUT,5)ISUM1,LENX
5 FORMAT(1X,I8,' ELEMENTS OF X ARRAY USED OUT OF',I8)
   IF(ISUM1.GT.LENX) WRITE(IOUT,6)
6 FORMAT(1X,' ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C7-------RETURN
RETURN
END
## List of Variables for Module RIV1AL

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IRIVCB</td>
<td>Package</td>
<td>Flag and a unit number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, unit number on which cell-by-cell flow terms will be recorded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>whenever ICBCFL (see RIV1BD module) is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be printed or recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, river leakage for each reach will be printed whenever ICBCFL is set.</td>
</tr>
<tr>
<td>ISP</td>
<td>Module</td>
<td>Number of words in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been</td>
</tr>
<tr>
<td></td>
<td></td>
<td>allocated. When space is allocated for an array, the size of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>array is added to ISUM.</td>
</tr>
<tr>
<td>ISUM1</td>
<td>Module</td>
<td>ISUM-1.</td>
</tr>
<tr>
<td>LCRIVR</td>
<td>Package</td>
<td>Location in the X array of the first element of array RIVR.</td>
</tr>
<tr>
<td>LENX</td>
<td>Global</td>
<td>Length of the X array in words. This should always be equal to the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dimension of X specified in the MAIN program.</td>
</tr>
<tr>
<td>MXRIVR</td>
<td>Package</td>
<td>Maximum number of river reaches active at any one time.</td>
</tr>
<tr>
<td>NRIVER</td>
<td>Package</td>
<td>Number of river reaches active during the current stress period.</td>
</tr>
</tbody>
</table>
Narrative for Module RIV1RP

This module reads data to build the river list.

1. Read ITMP. ITMP is the number of river reaches or a flag indicating that river reaches specified for the previous stress period should be reused.

2. Test ITMP. If ITMP is less than zero, the river data read for the last stress period will be reused. Print a message to that effect and RETURN.

3. If ITMP is greater than or equal to zero, it is the number of reaches for this stress period. Set the number of river reaches (NRIVER) in the current stress period equal to ITMP.

4. Compare the number of river reaches (NRIVER) in the current stress period to the number specified as the maximum for the simulation (MXRIVR). If NRIVER is greater than MXRIVR, STOP.

5. Print the number of river reaches in the current stress period (NRIVER).

6. See if there are any river reaches. If there are no river reaches in the current stress period (NRIVER = 0), bypass further river processing.

7. Read and print the layer, row, column, riverhead, riverbed conductance, and the elevation of the bottom of the riverbed for each reach.

8. RETURN.
ITMP is both a flag and a counter. If it is greater than or equal to zero, it is the number of reaches to be simulated during the current stress period. If it is less than zero, it indicates that the reaches simulated in the last stress period should be simulated in the current stress period.

MXRIVR is the maximum number of reaches to be simulated.
SUBROUTINE RIVLRP(RIVR,NRIVER,MXRIVR,IN,IOUT)

C     -----VERSION 1319 25AUG1982 RIVLRP
C     **********************************************
C     READ RIVER HEAD, CONDUCTANCE AND BOTTOM ELEVATION
C     **********************************************
C
C     SPECIFICATIONS:
C     --------------------------------------------------------
DIMENSION RIVR(6,MXRIVR)
C     --------------------------------------------------------
C
C1-------READ ITMP(NUMBER OF RIVER REACHES OR FLAG TO REUSE DATA)
READ(IN,8)ITMP
8 FORMAT(I10)
C
C2-------TEST ITMP.
   IF(ITMP.GE.0)GO TO 50
C
C2A-------IF ITMP < 0 THEN REUSE DATA FROM LAST STRESS PERIOD.
   WRITE(IOUT,7)
   7 FORMAT(1HO,'REUSING RIVER REACHES FROM LAST STRESS PERIOD')
   GO TO 260
C
C3-------IF ITMP=0 THEN IT IS THE NUMBER OF RIVER REACHES
50 NRIVER=ITMP
C
C4-------IF NRIVER>MXRIVR THEN STOP.
   IF(NRIVER.LE.MXRIVR)GO TO 100
   WRITE(IOUT,99)NRIVER,MXRIVR
   99 FORMAT(1HO,'NRIVER(',I4,') IS GREATER THAN MXRIVR(',I4,')')
C
C4A-------ABNORMAL STOP.
   STOP
C
C5-------PRINT NUMBER OF RIVER REACHES IN THIS STRESS PERIOD.
   100 WRITE(IOUT,1)NRIVER
   1 FORMAT(1HO,//1X,I5,'RIVER REACHES')
C
C6-------IF THERE ARE NO RIVER REACHES THEN RETURN.
   IF(NRIVER.EQ.0) GO TO 260
C
C7-------READ AND PRINT DATA FOR EACH RIVER REACH.
   WRITE(IOUT,3)
   3 FORMAT(1HO,15X,'LAYER',5X,'ROW',5X,'COL ',1,' STAGE CONDUCTANCE BOTTOM ELEVATION RIVER REACH')
   2/I1X,15X,80(1F10.0))
   DO 250 II=1,NRIVER
   READ(IN,4)K,I,J,RIVR(4,II),RIVR(5,II),RIVR(6,II)
   4 FORMAT(31I0,3F10.0)
   WRITE(IOUT,5)K,I,J,RIVR(4,II),RIVR(5,II),RIVR(6,II),II
   5 FORMAT(1X,15X,14,I9,18,G13.4,G14.4,G19.4,110)
   RIVR(1,II)=K
   RIVR(2,II)=I
   RIVR(3,II)=J
   250 CONTINUE
C
C8-------RETURN
   260 RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Module</td>
<td>Row number.</td>
</tr>
<tr>
<td>II</td>
<td>Module</td>
<td>Index for river reach.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ITMP</td>
<td>Module</td>
<td>Flag or number of rivers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, number of rivers active during the current stress period.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, same rivers active during the last stress period will be active during the current stress period.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Column number.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Layer number.</td>
</tr>
<tr>
<td>MXRIVR</td>
<td>Package</td>
<td>Maximum number of river reaches active at any one time.</td>
</tr>
<tr>
<td>NRIVER</td>
<td>Package</td>
<td>Number of river reaches active during the current stress period.</td>
</tr>
<tr>
<td>RIVR</td>
<td>Package</td>
<td>DIMENSION (6, MXRIVR), For each reach: layer, row, column, river head, riverbed conductance and elevation of bottom of riverbed.</td>
</tr>
</tbody>
</table>
Narrative for Module RIV1FM

This module adds terms representing river leakage to the accumulators HCOF and RHS.

1. If NRIVER is less than or equal to zero, in the current stress period, there are no river reaches. RETURN.

2. For each reach in the RIVR list, NO STEPS 3-8.

3. Determine the column (IC), row (IR), and layer (IL).

4. If the cell is external (IBOUND(IC, IR, IL) ≤ 0), bypass processing on this reach and go on to the next reach.

5. Since the cell is internal, get the river data (riverhead conductance of the riverbed and elevation of the bottom of the riverbed).

6. Compare the head in the aquifer (HNEW) to the elevation of the bottom of the riverbed (RBOT).

7. If the head in the aquifer (HNEW) is greater than the elevation of the bottom of the riverbed (RBOT), add the term -CRIV*HRIV to the accumulator RHS and the term -CRIV to the accumulator HCOF. (CRIV is the riverbed conductance; HRIV is the riverhead.)

8. If the head in the aquifer (HNEW) is less than or equal to RBOT, add the term -CRIV*(HRIV - RBOT) to the accumulator RHS.

9. RETURN.
RHS is an accumulator in which the right hand side of the equation is formulated.

HCOF is an accumulator in which the coefficient of head in the cell is formulated.
SUBROUTINE RIV1FM(NRIVER, MXRIVR, RIVR, HNEW, HCOF, RHS, IBOUND, 1 NCOL, NROW, NLAY)

C
C -----VERSION 0915 27AUG1982 RIV1FM
C
C ADD RIVER TERMS TO RHS AND HCOF
C
C **************************************************************************
C
C SPECIFICATIONS:
C -----------------------------------------------
C
DOUBLE PRECISION HNEW
DIMENSION RIVR(6,MXRIVR),HNEW(NCOL,NROW,NLAY),
1 HCOF(NCOL,NROW,NLAY), RHS(NCOL,NROW,NLAY),
2 IBOUND(NCOL,NROW,NLAY)

C
C
C1------IF NRIVER<=0 THERE ARE NO RIVERS. RETURN.
IF(NRIVER.LE.0)RETURN
C
C2------PROCESS EACH CELL IN THE RIVER LIST.
DO 100 L=1,NRIVER
C
C3------GET COLUMN, ROW, AND LAYER OF CELL CONTAINING REACH
IL=RIVR(1,L)
IR=RIVR(2,L)
IC=RIVR(3,L)
C
C4------IF THE CELL IS EXTERNAL SKIP IT.
IF(IBOUND(IC,IR,IL).LE.0)GO TO 100
C
C5------SINCE THE CELL IS INTERNAL GET THE RIVER DATA.
HRIV=RIVR(4,L)
CRIV=RIVR(5,L)
RBOT=RIVR(6,L)
HHNEW=HNEW(IC,IR,IL)
C
C6------COMPARE AQUIFER HEAD TO BOTTOM OF STREAM BED.
IF(HHNEW.LE.RBOT)GO TO 96
C
C7------SINCE HEAD>BOTTOM ADD TERMS TO RHS AND HCOF.
RHS(IC,IR,IL)=RHS(IC,IR,IL)-CRIV*HRIV
HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-CRIV
GO TO 100
C
C8------SINCE HEAD<BOTTOM ADD TERM ONLY TO RHS.
96 RHS(IC,IR,IL)=RHS(IC,IR,IL)-CRIV*(HRIV-RBOT)
100 CONTINUE
C
C9------RETURN
RETURN
END

6-28
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRIV</td>
<td>Module</td>
<td>Riverbed conductance.</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>in the finite-difference equation.</td>
</tr>
<tr>
<td>HHNEW</td>
<td>Module</td>
<td>HNEW (J,I,K), Single precision.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HRIV</td>
<td>Module</td>
<td>Head in the river.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Column number of the cell containing the river reach.</td>
</tr>
<tr>
<td>IL</td>
<td>Module</td>
<td>Layer number of the cell containing the river reach.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Row number of the cell containing the river reach.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for river reaches.</td>
</tr>
<tr>
<td>MXRIVR</td>
<td>Package</td>
<td>Maximum number of river reaches active at any one time.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NRIVER</td>
<td>Package</td>
<td>Number of river reaches active during the current stress period.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>RBOT</td>
<td>Module</td>
<td>Temporary field: elevation of the river bottom.</td>
</tr>
<tr>
<td>RHS</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>equation. RHS is an accumulation of terms from several different packages.</td>
</tr>
<tr>
<td>RIVR</td>
<td>Package</td>
<td>DIMENSION (6, MXRIVR), For each reach: layer, row, column, riverhead,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>riverbed conductance and elevation of bottom of riverbed.</td>
</tr>
</tbody>
</table>
Narrative for Module RIV1BD

This module calculates rates and volumes transferred between the aquifer and rivers.

1. Initialize the cell-by-cell flow-term flag (IBD) and the rate accumulators (RATIN and RATOUT).

2. If there are no reaches (NRIVER ≤ 0), skip down to step 17 and put zeros into the budget terms for rivers.

3. Test to see if the cell-by-cell flow terms are to be saved on the disk. They will not be saved if either of the following conditions hold: (1) This is not the proper time step (ICBCFL = 0) or (2) cell-by-cell flow terms are not to be saved for rivers during this simulation (IRIVCB ≤ 0). If cell-by-cell flow terms will be saved for this package, set the cell-by-cell flow-term flag (IBD) and clear the buffer in which they will be accumulated (BUFF).

4. For each reach, do steps 5-15 accumulating flows from or into the river.

5. Determine the row, column, and layer of the cell containing the reach.

6. If the cell is external (IBOUND(I,J,K) ≤ 0), bypass further processing of this reach.

7. Get the river parameters from the river list.

8. Check to see if the head in the cell is greater than the elevation of the bottom of the riverbed.
9. If the head in the cell is greater than the elevation of the bottom of the riverbed, set RATE equal to the conductance of the riverbed times the riverhead minus the head in the cell (\( \text{RATE} = \text{CRIV} \times (\text{HRIV} - \text{HNEW}) \)).

10. If the head in the cell is less than or equal to the elevation of the bottom of the riverbed, set RATE equal to the conductance of the riverbed times the riverhead minus the elevation of the bottom of the riverbed (\( \text{RATE} = \text{CRIV} \times (\text{HRIV} - \text{RBOT}) \)).

11. If the cell-by-cell flow terms are to be printed, print RATE.

12. If the cell-by-cell flow terms are to be saved, add the RATE to the buffer (BUFF).

13. Check to see whether the flow is into or out of the aquifer.

14. If RATE is negative, add it to RATOUT.

15. If RATE is positive, add it to RATIN.

16. See if the cell-by-cell flow terms are to be saved (IBD = 1). If they are, call module UBUDSV to record the buffer (BUFF) onto the disk.

17. Move RATIN and RATOUT into the VBVL array for printing by BAS1OT. Add RATIN and RATOUT multiplied by the time-step length to the volume accumulators in VBVL for printing by BAS1OT. Move the river budget term labels to VBNM for printing by BAS1OT.

18. Increment the budget-term counter (MSUM).

19. RETURN.
IBD is a flag which, if set, causes cell-by-cell flow terms for river leakage to be recorded.

EXTERNAL: a cell is said to be external if it is either no flow or constant head (i.e., an equation is not formulated for the cell).

RATE is the leakage rate into the aquifer from the river in a cell.

BUFFER is an array in which values are stored as they are being gathered for printing or recording.

RATOUT is an accumulator to which all flows out of the aquifer are added.

RATIN is an accumulator to which all flows into the aquifer are added.

IRIVCB is a flag and a unit number.

If IRIVCB > 0, it is the unit number on which cell-by-cell flow terms for rivers will be recorded whenever ICBCFL is set.

If IRIVCB = 0, cell-by-cell flow terms for rivers will not be printed or recorded.

If IRIVCB < 0, river leakage for each reach will be printed whenever ICBCFL is set.

ICBCFL is a flag.

If ICBCFL ≠ 0, cell-by-cell flow terms will be printed or recorded (depending on IRIVCB) for the current time step.
SUBROUTINE RIVlBD(NRIVER,MXRIVR,RIVR,IBOUND,HNEW,
   1 NCOL,NROW,NLAY,DFLT,VBVL,VBNM,MSUM,KSTEP,KPER,IRIVCB,
   2 ICBCFL,BUFF,IOUT)
C-----VERSION 1556 12MAY1987 RIVlBD
C ***~******************************************************************************
C CALCULATE VOLUMETRIC BUDGET FOR RIVERS ************************~******************************************************************************
C
C SPECIFICATIONS:
C---------------------------------------------------------------------------------------------------
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW
DIMENSION RIVR(6,MXRIVR),IBOUND(NCOL,NROW,NLAY),
1 HNEW(NCOL,NROW,NLAY),VBVL(4,20),VBNM(4,20),
2 BUFF(NCOL,NROW,NLAY)
DIMENSION TEXT(4)
DATA TEXT(1),TEXT(2),TEXT(3),TEXT(4) /' R','IVER',' LEA','KAGE'/
C
C1-------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1-------ACCUMULATORS (RATIN AND RATOUT).
   IBD=0
   RATIN=0.
   RATOUT=0.
C
C2-------IF NO REACHES KEEP ZEROES IN ACCUMULATORS.
   IF(NRIVER.EQ.0)GO TO 200
C
C3-------TEST TO SEE IF CELL-BY-CELL FLOW TERMS ARE NEEDED.
   IF(ICBCFL.EQ.0 .OR. IRIVCB.LE.0 ) GO TO 10
C
C3A-------CELL-BY-CELL FLOW TERMS ARE NEEDED SET IBD AND CLEAR BUFFER.
   IBD=1
   DO 5 IL=1,NLAY
   DO 5 IR=1,NROW
   DO 5 IC=1,NCOL
   BUFF(IC,IR,IL)=0.
   5 CONTINUE
C
C4-------FOR EACH RIVER REACH ACCUMULATE RIVER FLOW (STEPS 5-15)
   10 DO 100 L=1,NRIVER
C
C5------GET LAYER, ROW & COLUMN OF CELL CONTAINING REACH.
   IL=RIVR(1,L)
   IR=RIVR(2,L)
   IC=RIVR(3,L)
C
C6------IF CELL IS EXTERNAL MOVE ON TO NEXT REACH.
   IF(BOUND(IC,IR,IL).LE.0)GO TO 100
C
C7------GET RIVER PARAMETERS FROM RIVER LIST.
   HRIV=RIVR(4,L)
C
6-33
CRIV=RIVR(5,L)
RBOT=RIVR(6,L)
HHNEW=HNEW(IC,IR,IL)

C8------COMPARE HEAD IN AQUIFER TO BOTTOM OF RIVERBED.
C
C9------AQUIFER HEAD > BOTTOM THEN RATE=CRIV*(HRIV-HNEW).
     IF(HHNEW.GT.RBOT) RATE=CRIV*(HRIV-HNEW)
C
C10------AQUIFER HEAD < BOTTOM THEN RATE=CRIV*(HRIV-RBOT)
     IF(HHNEW.LE.RBOT) RATE=CRIV*(HRIV-RBOT)
C
C11------PRINT THE INDIVIDUAL RATES IF REQUESTED(IRIVCB<0).
     IF(IRIVCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT,900) (TEXT(N),N=1,4),
       1    KPER,KSTP,L,IR,IC,RATE
     900 FORMAT(1HO,4A4,' PERIOD',I3,' STEP',I3,' REACH',I4,
       1    ' LAYER',I3,' ROW',I4,' COL',I4,' RATE',G15.7)
C
C12------IF C-B-C FLOW TERMS ARE TO BE SAVED THEN ADD RATE TO BUFFER.
     IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+RATE
C
C13------SEE IF FLOW IS INTO AQUIFER OR INTO RIVER.
     IF(RATE)94,100,96
C
C14------AQUIFER IS DISCHARGING TO RIVER SUBTRACT RATE FROM RATOUT.
     94 RATOUT=RATOUT-RATE
     GO TO 100
C
C15------AQUIFER IS RECHARGED FROM RIVER ADD RATE TO RATIN.
     96 RATIN=RATIN+RATE
     100 CONTINUE
C
C16------IF C-B-C FLOW TERMS WILL BE SAVED CALL UBUDSV TO RECORD THEM.
     IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IRIVCB,BUFF,NCOL,NROW,
       1    NLAY,IOUT)
C
C17------MOVE RATES, VOLUMES & LABELS INTO ARRAYS FOR PRINTING.
   200 VBVL(3,MSUM)=RATIN
      VBVL(4,MSUM)=RATOUT
      VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
      VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
      VBNM(1,MSUM)=TEXT(1)
      VBNM(2,MSUM)=TEXT(2)
      VBNM(3,MSUM)=TEXT(3)
      VBNM(4,MSUM)=TEXT(4)
C
C18------INCREMENT BUDGET TERM COUNTER
       MSUM=MSUM+1
C
C19------RETURN
       RETURN
       END
### List of Variables for RIV1BD

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>CRIV</td>
<td>Module</td>
<td>Conductance of the bed of the river reach.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>HHNEW</td>
<td>Module</td>
<td>HNEW (J,I,K), Single precision.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY). Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HRIV</td>
<td>Module</td>
<td>Head in the river.</td>
</tr>
<tr>
<td>IBD</td>
<td>Module</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms for this package will not be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, cell-by-cell flow terms for this package will be recorded.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>ICBCFL</td>
<td>Global</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be recorded or printed for the current time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, cell-by-cell flow terms will be either printed or recorded (depending on IRIVCB) for the current time step.</td>
</tr>
<tr>
<td>IL</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IRIVCB</td>
<td>Package</td>
<td>Flag and a unit number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be printed or recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, river leakage for each reach will be printed whenever ICBCFL is set.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>Variable</td>
<td>Range</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td>------------</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for river reaches.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>MXRIVR</td>
<td>Package</td>
<td>Maximum number of river reaches active at any one time.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NRIVER</td>
<td>Package</td>
<td>Number of river reaches active during the current stress period.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>RATE</td>
<td>Module</td>
<td>Flow from the river into the cell. (Reverse the sign to get the flow into the river.)</td>
</tr>
<tr>
<td>RATIN</td>
<td>Module</td>
<td>Accumulator for the total flow into the flow field from rivers.</td>
</tr>
<tr>
<td>RATOUT</td>
<td>Module</td>
<td>Accumulator for the total flow out of flow field into rivers.</td>
</tr>
<tr>
<td>RBOT</td>
<td>Module</td>
<td>Elevation of the bottom of the riverbed.</td>
</tr>
<tr>
<td>RIVR</td>
<td>Package</td>
<td>DIMENSION (6, MXRIVR), For each reach: layer, row, column, riverhead, riverbed conductance, and elevation of the bottom of the riverbed.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget. For flow component N, the values in VBVL are: (1,N), Rate for the current time step into the flow field. (2,N), Rate for the current time step out of the flow field. (3,N), Volume into the flow field during simulation. (4,N), Volume out of the flow field during simulation.</td>
</tr>
</tbody>
</table>
CHAPTER 7

RECHARGE PACKAGE

Conceptualization and Implementation

The Recharge (RCH) Package is designed to simulate areally distributed recharge to the ground-water system. Most commonly, areal recharge occurs as a result of precipitation that percolates to the ground-water system. Recharge applied to the model is defined as

$$QR_{i,j} = I_{i,j} \cdot \text{DELR}_j \cdot \text{DELC}_i$$

(67)

where $QR_{i,j}$ is the recharge flow rate applied to the model at horizontal cell location $(i,j)$ expressed as a fluid volume per unit time; and $I_{i,j}$ is the recharge flux (in units of length per unit time) applicable to the map area, $\text{DELR}_j \cdot \text{DELC}_i$, of the cell. The recharge, $QR_{i,j}$, is applied to a single cell within the vertical column of cells located at $(i,j)$. There is no need to allow for recharge to occur simultaneously at multiple depths in the same vertical column because natural recharge enters the ground-water system at its top. In the simplest situation, the top of the ground-water system will occur in model layer 1; however, the vertical position of the top of the system may vary with horizontal location and with time as the water-table rises and falls. Three options for specifying the cell in each vertical column of cells that receives the recharge have been implemented as described below. The RCH Package can potentially be used to simulate recharge from sources other than precipitation -- for example, artificial recharge. If the ability to apply recharge to more than one cell in a vertical column of cells is required, then the Well Package, which allows recharge or discharge to be specified at any model cell, can be used.
In the package described herein, values of recharge flux, $I_{i,j}$, are read into a two dimensional array, $\text{RECH}_{i,j}$, at each stress period (unless an option is exercised to use recharge fluxes from the previous stress period). These values of recharge flux are immediately multiplied by horizontal cell areas, $\text{DELR}_j \text{DELC}_i$, to obtain values of $QR_{i,j}$, which are maintained in the RECH array. The cell within each vertical column to which the recharge is applied is specified through the recharge option code, NRCHOP, and optional array IRCH. The options include: (1) application of the recharge to model layer 1; (2) application of the recharge to any cell in the vertical column as specified by layer numbers contained in two dimensional array $\text{IRCH}_{i,j}$; and (3) application of the recharge to the uppermost active cell in the vertical column, provided there is no constant head cell above it in the column. Under options 1 and 2, if a cell designated to receive recharge is no-flow, then no recharge is added. Under the third option, if there is a constant head cell in a vertical column of cells and there is no active cell above, then no recharge is applied to this column because it is assumed that any recharge would be intercepted by the constant head source. Recharge flux values that are read into the model must be expressed in units that are consistent with the length and time units used to represent all other model parameters.

In the formation of the matrix equations, the recharge flow rate, $QR_{i,j}$, associated with a given horizontal cell location $(i,j)$ and vertical location, $k$, that is determined by the recharge option is subtracted from the value of $\text{RHS}_{i,j,k}$ (equation (26) or (29)). This is done at each iteration for all cells that receive recharge. Because recharge as defined is independent of aquifer head, nothing is added to the coefficient of head, $\text{HCOF}_{i,j,k}$. 

7-2
Figure 38.—Hypothetical problem showing which cells receive recharge under the three options available in the Recharge Package.
Careful consideration should be given to the problem under study and to the other options employed in its simulation before deciding which of the three recharge options listed above to utilize in a given situation. For example, figure 38 shows a situation in which a cross sectional model has been used to simulate a hypothetical problem involving recharge, seepage from a stream, and seepage into a stream (figure 38-a). Using the provision described in Chapter 5 for horizontal conductance formulation under water table conditions, the model mesh has been progressively truncated during simulation so that the uppermost active cells in each vertical column fall approximately at the water table. This process yields the final distribution of active, constant head and no-flow cells shown in figure 38-b.

Figure 38-C illustrates the recharge distribution to the model if option 1 above is utilized. Under this option recharge is permitted only to the top layer of the model. Thus once, the water table shape has been simulated by the use of no flow cells in the top layer, recharge to the vertical columns beneath those cells is shut off. This clearly fails to simulate the given system.

Figure 38-d illustrates the recharge distribution if option 2 is utilized, assuming that the user specifies recharge cells prior to the simulation on the basis of an estimated water table position, which differs slightly from that finally obtained in the simulation process. Four of the cells which the user had designated as recharge cells have converted to an inactive condition and thus receive no recharge.
Figure 38-e illustrates the Simulation under the third option, which turns out to be the one best suited for this particular situation. Under this option, recharge enters the uppermost active cell in each vertical column, except where constant head cells have been used to represent the streams. Thus, a continuous distribution of recharge to the water table is simulated.

For the typical situation of recharge from precipitation, option 3 is the easiest to use. The model user does not have to be concerned about determining which is the highest active cell in a vertical column because the program automatically determines this throughout the simulation. Option 1, however, can be useful in situations where recharge should not pass through the no-flow cells in layer 1. For example, some cells may be designated no-flow because they are impermeable. Any recharge specified for those cells should not pass into layer 2. Of course option 3 could still be used in this situation by specifying that the recharge rate is zero at the impermeable cells. The user should select the option that will result in the least effort for specification of input data. Similarly, option 2 may be useful when layers other than layer 1 have outcrop areas and when recharge to the specified layers should not penetrate through no-flow cells to a lower layer. Other factors to consider when choosing the best option are that option 2 uses more computer memory than options 1 and 3, and option 3 uses slightly more computer time than options 1 and 2.
Recharge Package Input

Input to the Recharge (RCH) Package is read from the unit specified in IUNIT(8).

FOR EACH SIMULATION

RCHIAL

1. Data: NRCHOP IRCHCB
   Format: I10  I10

FOR EACH STRESS PERIOD

RCH1RP

2. Data: INRECH INIRCH
   Format: I10  I10

3. Data: RECH(NCOL,NROW)
   Module: U2DREL

IF THE RECHARGE OPTION IS EQUAL TO 2

4. Data: IRCH(NCOL,NROW)
   Module: U2DINT

Explanation of Fields Used in Input Instructions

NRCHOP—is the recharge option code. Recharge fluxes are defined in a two-dimensional array, RECH, with one value for each vertical column. Accordingly, recharge is applied to one cell in each vertical column, and the option code determines which cell in the column is selected for recharge.

1 - Recharge is only to the top grid layer.

2 - Vertical distribution of recharge is specified in array IRCH.

3 - Recharge is applied to the highest active cell in each vertical column. A constant-head node intercepts recharge and prevents deeper infiltration.
IRCHCB--is a flag and a unit number.

If IRCHCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL (see Output Control) is set.

If IRCHCB ≤ 0, cell-by-cell flow terms will not be printed or recorded.

INRECH--is the RECH read flag.

If INRECH > 0, an array of recharge fluxes, RECH, is read.

If INRECH < 0, recharge rates from the preceding stress period are used.

INIRCH--is the IRCH read flag. When NRCHOP is two,

If INIRCH > 0, an array of layer numbers (IRCH) is read.

If INIRCH < 0, the array (IRCH) used in the preceding stress period is reused.

Note: When NRCHOP is one or three, INIRCH is ignored.

RECH--is the recharge flux (Lt⁻¹). Read only if INRECH is greater than or equal to zero.

IRCH--is the layer number array that defines the layer in each vertical column where recharge is applied. Read only if NRCHOP is two and if INIRCH is greater than or equal to zero.
### SAMPLE INPUT TO THE RECHARGE PACKAGE USING RECHARGE OPTION 1

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(NRCHOP, IRCHCB)</td>
<td>1 0</td>
</tr>
<tr>
<td>2</td>
<td>Stress period 1—(IRECH)</td>
<td>12 3.17E-8 (10F4.0)</td>
</tr>
<tr>
<td>3</td>
<td>Control record for recharge array</td>
<td>1.0 1.0 1.0 1.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0 1.0 1.0 1.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0 1.0 1.0 1.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0 1.0 1.0 1.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0 1.0 1.1 1.1</td>
</tr>
<tr>
<td></td>
<td>Recharge rates</td>
<td>1.1 1.1 1.1 1.1</td>
</tr>
<tr>
<td>2</td>
<td>Stress period 2—(IRECH)</td>
<td>-1</td>
</tr>
<tr>
<td>3</td>
<td>Stress period 4—(IRECH)</td>
<td>-1</td>
</tr>
<tr>
<td>3</td>
<td>Control record for recharge array</td>
<td>12 3.17E-8 (10F4.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2 1.2 1.2 1.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2 1.2 1.3 1.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0 1.0 1.3 1.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.2 1.3 1.3 1.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.3 1.3 1.4 1.4</td>
</tr>
</tbody>
</table>

### SAMPLE INPUT TO THE RECHARGE PACKAGE USING RECHARGE OPTION 2

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(NRCHOP, IRCHCB)</td>
<td>2 0</td>
</tr>
<tr>
<td>2</td>
<td>Stress period 1—(IRECH, INIRCH)</td>
<td>0 3.17E-8 (2012)</td>
</tr>
<tr>
<td>3</td>
<td>Control record for recharge array</td>
<td>12 2 2 3</td>
</tr>
<tr>
<td>4</td>
<td>Control record for layer indicator array</td>
<td>1 2 2 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 1 1 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 1 1 1</td>
</tr>
<tr>
<td></td>
<td>Layer numbers</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Stress period 2—(IRECH, INIRCH)</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>Stress period 3—(IRECH, INIRCH)</td>
<td>0 1.56E-8</td>
</tr>
</tbody>
</table>

**FIELDS IN ARRAY CONTROL RECORDS ARE:** LOCAT, CONST, FMTIN, IPRN
The Recharge Package (RCH1) consists of four modules, all of which are called by the MAIN program. The modules are:

- **RCH1AL**: Allocates space to contain recharge flow rate (RECH) and, if option 2 is specified, the layer-indicator array (IRCH).
- **RCH1RP**: Reads recharge flux (in flow per unit area) and indicator array (if option 2 is specified). Multiplies recharge flux by cell area.
- **RCH1FM**: Subtracts the recharge flow rate from the accumulator in which RHS is formulated.
- **RCH1BD**: Calculates the rate and accumulated volume of recharge into the flow system.
Narrative for Module RCHIAL

This module allocates space in the X array to store data relating to areally distributed recharge.

1. Print a message identifying the package.

2. Read and print the option indicator (NRCHOP) and the unit number for cell-by-cell flow terms (IRCHCR).

3. See if the recharge option (NRCHOP) is legal. If NRCHOP is illegal (not 1, 2, or 3), print a message saying the option is illegal. Do not allocate storage. STOP.

4. If NRCHOP is legal, print NRCHOP.

5. If cell-by-cell flow terms are to be recorded, print the unit number where they will be recorded.

6. Allocate space for the recharge array (RECH). Space is allocated by setting the first element of RECH (LCRECH) equal to the location (ISUM) of the first unused element in the X array and adding the size of the array to ISUM.

7. If the recharge option (NRCHOP) is equal to two, allocate space for a layer-indicator array (IRCH).

8. Calculate and print the number of elements in the X array used by the Recharge Package.

9. RETURN.
NRCHOP is the recharge option.

1 - Recharge is to the top layer.

2 - Recharge is to the layer specified by the user in the indicator array (IRCH).

3 - Recharge is to the uppermost active cell.

IRCHCB is the unit number on which cell-by-cell flow terms for recharge will be written.

RECH is an array which contains a recharge rate for each horizontal cell location.

IRCH is an array which contains the layer number to which recharge is applied for each horizontal location. It is used only if option 2 has been specified.
SUBROUTINE RCHIAL(ISUM, LENX, LCIIRCH, LCRECH, NRCHOP,
   1 NCOL, NROW, IN, IOUT, IRCHCB)
C
C-----VERSION 1559 12MAY1987 RCHIAL
C ******************************************************************************
C ALLOCATE ARRAY STORAGE FOR RECHARGE
******************************************************************************
C
C SPECIFICATIONS:
C -----------------------------------------------------------------------------
C
C1-------IDENTIFY PACKAGE.
   WRITE(IOUT,11)IN
   1 FORMAT(1H0,'RCH1 -- RECHARGE PACKAGE, VERSION 1, 9/1/87',
      1 ' INPUT READ FROM UNIT',I3)
C
C2-------READ NRCHOP AND IRCHCB.
   READ(IN,2)NRCHOP,IRCHCB
   2 FORMAT(2I10)
C
C3-------CHECK IF OPTIION IS LEGAL.
   IF(NRCHOP.GE.1.AND.NRCHOP.LE.3)GO TO 200
C
C3A-------IF ILEGAL PRINT A MESSAGE AND ABORT SIMULATION
   WRITE(IOUT,8)
   8 FORMAT(1X,'ILLEGAL OPTION CODE. SIMULATION ABORTING')
   STOP
C
C4-------IF OPTION IS LEGAL PRINT OPTION CODE.
   200 IRK=ISUM
   IF(NRCHOP.EQ.1) WRITE(IOUT,201)
   201 FORMAT(1X,'OPTION 1 -- RECHARGE TO TOP LAYER')
   IF(NRCHOP.EQ.2) WRITE(IOUT,202)
   202 FORMAT(1X,'OPTION 2 -- RECHARGE TO ONE SPECIFIED NODE IN EACH',
      1 ' VERTICAL COLUMN')
   IF(NRCHOP.EQ.3) WRITE(IOUT,203)
   203 FORMAT(1X,'OPTION 3 -- RECHARGE TO HIGHEST ACTIVE NODE IN EACH',
      1 ' VERTICAL COLUMN')
C
C5-------IF CELL-BY-CELL FLOW TERMS TO BE SAVED THEN PRINT UNIT #
   IF(IRCHCB.GT.0) WRITE(IOUT,204) IRCHCB
   204 FORMAT(1X,'CELL-BY-CELL FLOW TERMS WILL BE RECORDED ON UNIT',I3)
C
C6-------ALLOCATE SPACE FOR THE RECHARGE ARRAY(RECH).
   LCRECH=ISUM
   ISUM=ISUM+NCOL*NROW
C
C7-------IF OPTION 2 THEN ALLOCATE SPACE FOR INDICATOR ARRAY(IRCH)
   LCIIRCH=ISUM
   IF(NRCHOP.NE.2)GO TO 300
   ISUM=ISUM+NCOL*NROW
C
C8-------CALCULATE AND PRINT AMOUNT OF SPACE USED BY RECHARGE.
   300 IRK=ISUM-IRK
   WRITE(IOUT,4)IRK
   4 FORMAT(1X,16,' ELEMENTS OF X ARRAY USED FOR RECHARGE')
   ISUM=ISUM-1
   WRITE(IOUT,5)ISUM,LENX
   5 FORMAT(1X,16,' ELEMENTS OF X ARRAY USED OUT OF',I0)
   IF(ISUM.GT.LENX)WRITE(IOUT,6)
   6 FORMAT(1X,16' ***X ARRAY MUST BE MADE LARGER***')
C
C9-------RETURN
   RETURN
   END
### List of Variables for Module RCH1AL

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUM</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOU = 6.</td>
</tr>
<tr>
<td>IRCHCB</td>
<td>Package</td>
<td>Flag. IRCHCB &lt; 0, cell-by-cell flow terms will not be printed or recorded. IRCHCB &gt; 0 and ICBCFL ≠ 0, cell-by-cell flow terms for the RCH1 Package will be recorded on UNIT = IRCHCB.</td>
</tr>
<tr>
<td>IRK</td>
<td>Module</td>
<td>Before this module allocates space, IRK is set equal to ISUM. After allocation, IRK is subtracted from ISUM to get the amount of space in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.</td>
</tr>
<tr>
<td>ISUM1</td>
<td>Module</td>
<td>ISUM+1.</td>
</tr>
<tr>
<td>LCIRCH</td>
<td>Package</td>
<td>Location in the X array of the first element of array IRCH.</td>
</tr>
<tr>
<td>LCRECH</td>
<td>Package</td>
<td>Location in the X array of the first element of array RECH.</td>
</tr>
<tr>
<td>LENX</td>
<td>Global</td>
<td>Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NRCH1OP</td>
<td>Package</td>
<td>Recharge option:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 1, recharge is to the top grid layer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 2, recharge is to the grid layer specified in array IRCH.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 3, recharge is to the highest variable-head cell which is not below a constant-head cell.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
</tbody>
</table>
This module reads data used to calculate the terms which represent areally distributed recharge.

1. Read the values INRECH and INIRCH which indicate whether the data contained in arrays RECH and IRCH used during the last stress period are to be used for the current stress period.

2. Test INRECH to see where the recharge flux (RECH) is coming from. If INRECH is less than zero, the recharge rate used in the last stress period will be used again in this stress period. Print a message to that effect. GO TO STEP 5.

3. If INRECH is greater than or equal to zero, CALL U2DREL to read the recharge rate (RECH).

4. Multiply the specified recharge flux rates by the cell areas to get the volumetric-recharge rate.

5. If the recharge option (NRCHOP) is not equal to two, a layer-indicator array is not needed. GO TO STEP 8.

6. If INIRCH is less than zero, the data in IRCH left over from the last stress period will be used in this stress period. Print a message to that effect. GO TO STEP 8.

7. If INIRCH is greater than or equal to zero, CALL U2DINT to read the IRCH array.

8. RETURN.
INRECH is a flag which, when set, indicates that recharge rates (RECH) should be read for the current stress period. If it is clear (< 0), recharge rates from the last stress period should be reused.

INIRCH is a flag similar to INRECH used for the layer indicator array IRCH.

RECH is an array containing a recharge rate for every horizontal cell location.

IRCH is an array containing a recharge indicator for each horizontal cell location. For each horizontal cell location, it indicates the layer number of the cell at that location which gets recharge. It is used only if the recharge option (NRCHOP) is equal to two.
SUBROUTINE RCH1RP(NRCHOP, IRCH, RECH, DELR, DELC, NROW, NCOL, 
   1       IN, IOUT)
C
C-------VERSION 1634 24JUL1987 RCH1RP
C  ******************************************
C  READ RECHARGE RATES
C  ******************************************
C
C  SPECIFICATIONS:
C  -----------------------------------------------------------------------
CHARACTER*4 ANAME
DIMENSION IRCH(NCOL, NROW), RECH(NCOL, NROW), 
   1       ANAME(6, 2), DELR(NCOL), DELC(NROW)
C
DATA ANAME(1, 1), ANAME(2, 1), ANAME(3, 1), ANAME(4, 1), ANAME(5, 1), 
   1       ANAME(6, 1) /'AR','E','CH','L','A','Y'/
DATA ANAME(1, 2), ANAME(2, 2), ANAME(3, 2), ANAME(4, 2), ANAME(5, 2), 
   1       ANAME(6, 2) /'R','E','C','H','L','A','R','G','E'/
C
C1-------READ FLAGS SHOWING WHETHER DATA IS TO BE REUSED.
   READ(IN, 4) INRECH, INIRCH
   4 FORMAT(2I10)
C
C2-------TEST INRECH TO SEE WHERE RECH IS COMING FROM.
   IF(INRECH.GE.0) GO TO 32
C
C2A------IF INRECH<0 THEN REUSE RECHARGE ARRAY FROM LAST STRESS PERIOD
   WRITE(IOUT, 3)
   3 FORMAT(1HO, 'REUSING RECH FROM LAST STRESS PERIOD')
   GO TO 55
C
C3-------IF INIRCH<0 THEN CALL U2DREL TO READ RECHARGE RATE.
   32 CALL U2DREL(RECH, ANAME(1, 2), NROW, NCOL, 0, IN, IOUT)
C
C4-------MULTIPLY RECHARGE RATE BY CELL AREA TO GET VOLUMETRIC RATE.
   DO 50 IR = 1, NROW
   DO 50 IC = 1, NCOL
   RECH(IC, IR) = RECH(IC, IR) * DELR(IC) * DELC(IR)
   50 CONTINUE
C
C5-------IF NRCHOP=2 THEN A LAYER INDICATOR ARRAY IS NEEDED.
   55 IF (NRCHOP.NE.2) GO TO 60
C
C6-------IF INIRCH<0 THEN REUSE LAYER INDICATOR ARRAY.
   IF(INIRCH.GE.0) GO TO 58
   WRITE(IOUT, 2)
   2 FORMAT(1HO, 'REUSING IRCH FROM LAST STRESS PERIOD')
   GO TO 60
C
C7-------IF INIRCH>0 CALL U2DINT TO READ LAYER IND ARRAY(IRM)
   58 CALL U2DINT(IRCH, ANAME(1, 1), NROW, NCOL, 0, IN, IOUT)
C
C8-------RETURN
   60 RETURN
END
### List of Variables for Module RCHIRP

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANAME</td>
<td>Module</td>
<td>Label for printout of the input array.</td>
</tr>
<tr>
<td>DELC</td>
<td>Global</td>
<td>DIMENSION (NROW), Cell dimension in the column direction. DELC(I) contains the width of row I.</td>
</tr>
<tr>
<td>DELR</td>
<td>Global</td>
<td>DIMENSION (NCOL), Cell dimension in the row direction. DELR(J) contains the width of column J.</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
</tbody>
</table>
| INIRCH   | Module  | Flag.  
 |          |         | ≥ 0, IRCH array will be read.  
 |          |         | < 0, IRCH array already in memory from the last stress period will be used. |
| INRECH   | Module  | Flag.  
 |          |         | ≥ 0, RECH array will be read.  
 |          |         | < 0, RECH array already in memory from the last stress period will be used. |
| IOUT     | Global  | Primary unit number for all printed output. IOUT = 6. |
| IR       | Module  | Index for rows. |
| IRCH     | Package | DIMENSION (NCOL,NROW), Layer number for each horizontal cell location to which recharge will be applied if the recharge option (NRCHOP) is equal to 2. |
| NCOL     | Global  | Number of columns in the grid. |
| NRCHOP   | Package | Recharge option:  
 |          |         | = 1, recharge is to the top grid layer.  
 |          |         | = 2, recharge is to the grid layer specified in array IRCH.  
 |          |         | = 3, recharge is to the highest variable-head cell which is not below a constant-head cell. |
| NROW     | Global  | Number of rows in the grid. |
| RECH     | Package | DIMENSION (NCOL,NROW), Recharge flow rate. Recharge flux is read into RECH and then multiplied by cell area to obtain recharge flow rate. |
This module adds terms representing areally distributed recharge to the accumulators in which the terms HCOF and RHS are formulated.

1. If the recharge option (NRCHOP) is equal to one, recharge is to the top layer. For each horizontal location, DO STEPS (a) AND (b).

(a) If the cell is external (IBOUND(I,J,K) \leq 0), ignore it. SKIP STEP (b).

(b) Subtract the recharge flow rate from the RHS accumulator.

2. If the recharge option is two, recharge is only to the cells specified in the layer-indicator array (IRCH).

   (a) Get the layer index from the layer-indicator array (IRCH).

   (b) If the cell is external, ignore it. SKIP STEP (c).

   (c) Subtract the recharge flow rate from the RHS accumulator.

3. If the recharge option is three, recharge is in the uppermost internal cell. For each horizontal cell location:

   (a) If the cell is constant head, there will be no recharge below it. Move on to the next horizontal cell location.

   (b) If the cell is no flow, move down a cell and go back to (a).

   (c) Subtract the recharge flow rate from the RHS accumulator. Move on to the next horizontal cell location.

4. RETURN
RHS is the right hand side of the finite-difference equation. It includes all terms that are independent of head at the end of the time step.

IRCH is an array which contains the layer number to which recharge is applied for each horizontal location. It is used only if option 2 has been specified.

NRCHOP is the recharge option.

1 - Recharge is to the top layer.

2 - Recharge is to the layer specified by the user in the indicator array (IRCH).

3 - Recharge is to the uppermost active cell.
SUBROUTINE RCH1FM(NRCHOP, IRCH, RECH, RHS, IBOUND, NCOL, NROW, NLAY)

C---------------------------------------------------------------
C SUBTRACT RECHARGE FROM RHS
C---------------------------------------------------------------
C
C SPECIFICATIONS:
C---------------------------------------------------------------
DIMENSION IRCH(NCOL,NROW), RECH(NCOL, NROW, 1), RHS(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY)
C---------------------------------------------------------------
C
C1------IF NRCHOP IS 1 RECHARGE IS IN TOP LAYER. LAYER INDEX IS 1.
   IF(NRCHOP.NE.1) GO TO 15
   DO 10 IR=1,NROW
   DO 10 IC=1,NCOL
   C1A------IF CELL IS EXTERNAL THERE IS NO RECHARGE INTO IT.
   IF(IBOUND(IC,IR,1).LE.0) GO TO 10
   C1B------SUBTRACT RECHARGE RATE FROM RIGHT-HAND-SIDE.
   RHS(IC,IR,1)=RHS(IC,IR,1)-RECH(IC,IR)
  10 CONTINUE
   GO TO 100
C
C2------IF OPTION IS 2 THEN RECHARGE IS INTO LAYER IN INDICATOR ARRAY
   IF(NRCHOP.NE.2) GO TO 25
   DO 20 IR=1,NROW
   DO 20 IC=1,NCOL
   C2A------LAYER INDEX IS IN INDICATOR ARRAY.
   IL=IRCH(IC,IR)
   C2B------IF THE CELL IS EXTERNAL THERE IS NO RECHARGE INTO IT.
   IF(IBOUND(IC,IR,IL).LE.0) GO TO 20
   C2C------SUBTRACT RECHARGE FROM RIGHT-HAND-SIDE.
   RHS(IC,IR,IL)=RHS(IC,IR,IL)-RECH(IC,IR)
  20 CONTINUE
   GO TO 100
C
C3------IF OPTION IS 3 RECHARGE IS INTO HIGHEST INTERNAL CELL.
   IF(NRCHOP.NE.3) GO TO 100
   C3A------IF CELL IS CONSTANT HEAD MOVE ON TO NEXT HORIZONTAL LOCATION.
   IF(IBOUND(IC,IR,IL).LT.0) GO TO 30
   C3B------IF CELL IS INACTIVE MOVE DOWN A LAYER.
   IF(IBOUND(IC,IR,IL).EQ.0) GO TO 28
   C3C------SUBTRACT RECHARGE FROM RIGHT-HAND-SIDE.
   RHS(IC,IR,IL)=RHS(IC,IR,IL)-RECH(IC,IR)
  28 CONTINUE
  20 CONTINUE
  100 CONTINUE
C
C4------RETURN
RETURN
END
**List of Variables for Module RCHFM**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
</table>
| IBOUND  | Global | DIMENSION (NCOL,NROW,NLAY), Status of each cell.  
|          |        | < 0, constant-head cell  
|          |        | = 0, inactive cell  
|          |        | > 0, variable-head cell |
| IC      | Module | Index for columns.  |
| IL      | Module | Index for layers.  |
| IOUT    | Global | Primary unit number for all printed output. IOUT = 6.  |
| IR      | Module | Index for rows.  |
| IRCH    | Package | DIMENSION (NCOL,NROW), Layer number for each horizontal cell location to which recharge will be applied if the recharge option (NRCHOP) is equal to 2.  |
| NCOL    | Global | Number of columns in the grid.  |
| NLAY    | Global | Number of layers in the grid.  |
| NRCHOP  | Package | Recharge option:  
|          |        | = 1, recharge is to the top grid layer.  
|          |        | = 2, recharge is to the grid layer specified in array IRCH.  
|          |        | = 3, recharge is to the highest variable-head cell which is not below a constant-head cell.  |
| NROW    | Global | Number of rows in the grid.  |
| RECH    | Package | DIMENSION (NCOL,NROW), Recharge flow rate.  |
| RHS     | Global | DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.  |
This module calculates rates and volumes added to the aquifer by areally distributed recharge.

1. Clear the rate accumulators RATIN and RATOUT.

2. If cell-by-cell flow terms will be saved, clear the buffer (BUFF) in which they will be accumulated.

3. If the recharge option is one, the recharge goes into the top layer. Process the horizontal locations one at a time.
   (a) If the cell is external, do not calculate budget.
   (b) If cell-by-cell flow terms will be saved, add recharge to the buffer.
   (c) If the recharge is positive, add it to RATIN; otherwise, add it to RATOUT.

4. If the recharge option is two, recharge goes into the layer specified in indicator array (IRCH). Process the horizontal locations one at a time.
   (a) Get the cell layer from indicator array (IRCH).
   (b) If the cell is external, do not calculate budget.
   (c) If cell-by-cell flow terms will be saved, add the recharge to the buffer.
   (d) If the recharge is positive, add it to RATIN; otherwise, add it to RATOUT.
5. If the recharge option is three, the recharge goes into the top variable-head cell provided there is not a constant-head cell above it. Process the horizontal locations one at a time. Start with the top cell and work down.

(a) If the cell is inactive, there is no recharge into that cell; move down to the next one.

(b) If the cell is constant, there is no recharge at this horizontal location; move on to the next horizontal location.

(c) If cell-by-cell flow terms are to be saved, add the recharge to the buffer.

(d) If the recharge is positive, add it to RATIN; otherwise, add it to RATOUT.

6. If cell-by-cell flow terms will be saved, call module UBUDDSV to write the buffer (BUFF) onto disk.

7. Move RATIN and RATOUT into the VBVL array for printing by BAS1OT.

8. Add RATOUT multiplied by the time-step length to the volume accumulators in VBVL for printing by BAS1OT.

9. Move the recharge budget-term labels to VBNM for printing by BAS1OT.

10. Increment the budget-term counter (MSUM).

11. RETURN.
RATIN is an accumulator to which all flows into the aquifer are added.

RATOUT is an accumulator to which all flows out of the aquifer are added.

BUFFER is an array in which values are stored as they are being gathered for printing or recording.

NRCHOP is the recharge option.

1 - Recharge is to the top layer.

2 - Recharge is to the layer specified by the user in the indicator array (IRCH).

3 - Recharge is to the uppermost active cell.

IRCH is an array containing a recharge indicator for each horizontal cell. It is used only if the recharge option (NRCHOP) is equal to two.

VBVL is a table of budget entries calculated by component-of-flow packages for use in calculating the volumetric budget.

VBNM is a table of labels for budget terms.

EXTERNAL: a cell is external if it is either no flow (inactive) or constant head.
SUBROUTINE RCHIBD(NRCHOP,IRCH,RECH,IBOUND,NROW,NCOL,NLAY,  
DELT,VBVL,VBNM,MSUM,KPER,IRCHCB,ICBCFL,BUFF,IOUT)

C-----VERSION 1602 12MAY1987 RCHIBD
C*********************************************************************************
C CALCULATE VOLUMETRIC BUDGET FOR RECHARGE
C*********************************************************************************
C SPECIFICATIONS:
C*********************************************************************************
CHARACTER*4 VBNM,TEXT
DIMENSION IRCH(NCOL,NROW),RECH(NCOL,NROW),  
IBOUND(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY),  
VBVL(4,20),VBNM(4,20)
DIMENSION TEXT(4)
DATA TEXT(1),TEXT(2),TEXT(3),TEXT(4) /' ',' ','RECH','ARGE'/'
C*********************************************************************************
C
C------CLEAR THE RATE ACCUMULATORS.
RATIN=0.
RATOUT=0.
C
C2------IF CELL-BY-CELL FLOW TERMS WILL BE SAVED THEN CLEAR THE BUFFER.
IBD=0
IF(ICBCFL.EQ.0 .OR. IRCHCB.EQ.0) GO TO 5
IBD=1
DO 2 IL=1,NLAY
         DO 2 IR=1,NROW
         DO 2 IC=1,NCOL
         BUFF(IC,IR,IL)=0.
2 CONTINUE
C
C3------IF NRCHOP=1 RECH GOES INTO LAYER 1. PROCESS EACH HORIZONTAL
C3------CELL LOCATION.
5 IF(NRCHOP.NE.1) GO TO 15
C
C3A------RECHARGE IS APPLIED TO TOP LAYER
DO 10 IR=1,NROW
         DO 10 IC=1,NCOL
3AQ+----IF CELL-BY-CELL FLOW TERMS WILL BE SAVED THEN ADD RECH TO BUFF
IF(IBD.EQ.0) Buff(IC,IR,IL)=Q
C
C3B------IF RECH POSITIVE ADD IT TO RATIN ELSE ADD IT TO RATOUT.
IF(Q) 8,10,7
7 RATIN=RATIN+Q
         GO TO 10
8 RATOUT=RATOUT-Q
10 CONTINUE
GO TO 100
C
C4------IF NRCHOP=2 RECH IS IN LAYER SHOWN IN INDICATOR ARRAY(IRCH).
C4------PROCESS HORIZONTAL CELL LOCATIONS ONE AT A TIME.
15 IF(NRCHOP.NE.2) GO TO 25
         DO 20 IR=1,NROW
         DO 20 IC=1,NCOL
C
C4A------GET LAYER INDEX FROM INDICATOR ARRAY(IRCH).
IL=IRCH(IC,IR)
C
C4B------IF CELL IS EXTERNAL DO NOT CALCULATE BUDGET FOR IT.
         IF_IBOUND(IC,IR,IL).LE.0)GO TO 20
Q=RECH(IC,IR)

C4C-----IF C-D-C FLOW TERMS WILL BE SAVED THEN ADD RECHARGE TO BUFFER.
   IF(IBD.EQ.1) BUFF(IC,IR,IL)=Q

C
C4D-----IF RECHARGE IS POSITIVE ADD TO RATIN ELSE ADD IT TO RATOUT.
   IF(Q) 18,20,17
   17 RATIN=RATIN+Q
   GO TO 20
   18 RATOUT=RATOUT-Q
   20 CONTINUE
   GO TO 100

C
C5-----IF OPTION=3 RECHARGE IS INTO HIGHEST INTERNAL CELL. IT WILL NOT
C5------PASS THROUGH A CONSTANT HEAD CELL. PROCESS HORIZONTAL CELL
C5------LOCATIONS ONE AT A TIME.
   25 IF(NRCHOP.NE.3)GO TO 100
   DO 30 IR=1,NROW
   DO 30 IC=1,NCOL
   DO 28 IL=1,NLAY

C5A------IF CELL IS CONSTANT HEAD MOVE ON TO NEXT HORIZONTAL LOCATION.
   IF(IBOUND(IC,IR,IL).LT.0) GO TO 30

C5D------IF CELL IS INACTIVE MOVE DOWN TO NEXT CELL.
   IF(IBOUND(IC,IR,IL).EQ.0)GO TO 28
   Q=RECH(IC,IR)

C
C5E-----IF C-D-C FLOW TERMS TO BE SAVED THEN ADD RECHARGE TO BUFFER.
   IF(IBD.EQ.1) BUFF(IC,IR,IL)=Q

C
C5D-----IF RECH IS POSITIVE ADD IT TO RATIN ELSE ADD IT TO RATOUT.
   IF(Q) 27,30,26
   26 RATIN=RATIN+Q
   GO TO 30
   27 RATOUT=RATOUT-Q
   GO TO 30
   28 CONTINUE
   30 CONTINUE

C
100 CONTINUE

C
C6-----IF C-D-C FLOW TERMS TO BE SAVED CALL UBUDSV TO WRITE THEM.
   IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IRCHCB,BUFF,NCOL,NROW,
   1
   NLAY,IOUT)

C
C7-----MOVE TOTAL RECHARGE RATE INTO VBVL FOR PRINTING BY BASIOT.
   VBVL(4,MSUM)=RATOUT
   VBVL(3,MSUM)=RATIN

C
C8-----ADD RECHARGE FOR TIME STEP TO RECHARGE ACCUMULATOR IN VBVL.
   VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
   VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT

C
C9-----MOVE BUDGET TERM LABELS TO VBMM FOR PRINT BY MODULE BAS-OT.
   VBMM(1,MSUM)=TEXT(1)
   VBMM(2,MSUM)=TEXT(2)
   VBMM(3,MSUM)=TEXT(3)
   VBMM(4,MSUM)=TEXT(4)

C
C10------INCREMENT BUDGET TERM COUNTER.
   MSUM=MSUM+1

C
C11------RETURN
   RETURN
   END

7-26
## List of Variables for Module RCH1BD

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>IBD</td>
<td>Module</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0, cell-by-cell flow terms for this package will not be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, cell-by-cell flow terms for this package will be recorded.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>ICBCFL</td>
<td>Global</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0, cell-by-cell flow terms will not be recorded or printed for the current time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, cell-by-cell flow terms will be recorded for the current time step.</td>
</tr>
<tr>
<td>IL</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IRCH</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Layer number for each horizontal cell location to which recharge will be applied if the recharge option (NRCHOP) is equal to 2.</td>
</tr>
<tr>
<td>IRCHCB</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRCHCB ≤ 0, cell-by-cell flow terms will not be recorded or printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRCHCB &gt; 0 and ICBCFL ≠ 0, cell-by-cell flow terms for the RCH1 Package will be recorded on UNIT = IRCHCB.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>Variable</td>
<td>Range</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NRCHOP</td>
<td>Package</td>
<td>Recharge option:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 1, recharge is to the top grid layer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 2, recharge is to the grid layer specified in array IRCH.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 3, recharge is to the highest variable-head cell which is not below a constant-head cell.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>Q</td>
<td>Module</td>
<td>Flow from recharge into a cell. (Reverse the sign to get flow out of the cell.)</td>
</tr>
<tr>
<td>RATIN</td>
<td>Module</td>
<td>Accumulator for the total flow into the flow field from recharge.</td>
</tr>
<tr>
<td>RATOUT</td>
<td>Module</td>
<td>Accumulator for the total flow out of the flow field to recharge.</td>
</tr>
<tr>
<td>RECH</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Recharge flow rate.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For flow component N, the values in VBVL are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1,N), Rate for the current time step into the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(2,N), Rate for the current time step out of the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,N), Volume into the flow field during simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4,N), Volume out of the flow field during simulation.</td>
</tr>
</tbody>
</table>
The Well Package is designed to simulate features such as wells which withdraw water from the aquifer (or add water to it) at a specified rate during a given stress period, where the rate is independent of both the cell area and the head in the cell. The discussion in this section is developed on the assumption that the features to be simulated are actually wells, either discharging or recharging.

Well discharge is handled in the Well Package by specifying the rate, \( Q \), at which each individual well adds water to the aquifer or removes water from it, during each stress period of the simulation. Negative values of \( Q \) are used to indicate well discharge, while positive values of \( Q \) indicate a recharging well.

At the beginning of each stress period, the WEL1RP module reads four values for each well—the row, column and layer number of the cell in which the well is located, and the discharge or recharge rate, \( Q \), of the well during that stress period. At each iteration, as the matrix equations are formulated, the value of \( Q \) for each well is subtracted from the RHS value (equation (26) or (29)) for the cell containing that well. Where more than one well falls within a single cell, the calculation is repeated for each well as the RHS term for that cell is assembled. Thus the user specifies the discharge associated with each individual well, and these are in effect summed within the program to obtain the total discharge from the cell.
The Well Package, as it is presently formulated, does not accommodate wells which are open to more than one layer of the model. However, a well of this type can be represented as a group of single-layer wells, each open to one of the layers tapped by the multi-layer well, and each having an individual Q term specified for each stress period. If this approach is used, the discharge of the multi-layer well must be divided or apportioned in some way among the individual layers, externally to the model program. A common method of doing this is to divide the well discharge in proportion to the layer transmissivities i.e.

\[
\frac{Q_l}{Q_w} = \frac{T_l}{\Sigma T}
\]  

(68)

where \( Q_l \) is the discharge from layer 1 to a particular well in a given stress period, \( Q_w \) is the well discharge in that stress period, \( T_l \) is the transmissivity of layer 1 and \( \Sigma T \) represents the sum of the transmissivities of all layers penetrated by the well. Again, it's important to note that equation (68), or some other method of apportioning the discharge, must be implemented by the user externally to the program for each multi-layer well, and for each stress period.

This approach, in which a multi-layer well is represented as a group of single layer wells, fails to take into account the interconnection between various layers provided by the well itself, and is thus an incomplete representation of the problem. A package which will provide an improved approximation of multi-layer well effects is under development.
Well Package Input

Input for the Well (WEL) Package is read from the unit specified in IUNIT(2).

FOR EACH SIMULATION

WEL1AL

1. Data: MXWELL IWELCB
   Format: I10 I10

FOR EACH STRESS PERIOD

WEL1RP

2. Data: ITMP
   Format: I10

3. Data: Layer Row Column Q
   Format: I10 I10 I10 F10.0

(Input item 3 normally consists of one record for each well. If ITMP is negative or zero, item 3 is not read.)

Explanation of Fields Used in Input Instructions

MXWELL—is the maximum number of wells used at any time.

IWELCB—is a flag and a unit number.

If IWELCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL (see Output Control) is set.

If IWELCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IWELCB < 0, well recharge will be printed whenever ICBCFL is set.

ITMP—is a flag and a counter.

If ITMP < 0, well data from the last stress period will be reused.

If ITMP > 0, ITMP will be the number of wells active during the current stress period.

Layer—is the layer number of the model cell that contains the well.

Row—is the row number of the model cell that contains the well.

Column—is the column number of the model cell that contains the well.

Q—is the volumetric recharge rate. A positive value indicates recharge and a negative value indicates discharge.
<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[MXWEL, WELCO]</td>
<td>6 24</td>
</tr>
<tr>
<td>2</td>
<td>STRESS PERIOD 1 (ITMP)</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>FIRST WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>SECONED WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>THIRD WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>FOURTH WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>STRESS PERIOD 2 (ITMP)</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>STRESS PERIOD 3 (ITMP)</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>STRESS PERIOD 4 (ITMP)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>FIRST WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>SECOND WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>THIRD WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>FOURTH WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>FIFTH WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>SIXTH WELL (Layer, Row, Column, Q)</td>
<td>2</td>
</tr>
</tbody>
</table>
The Well Package (WEL) consists of four modules, all of which are called by the MAIN program. The modules are:

- **WELIAL**: Allocates space for the list of wells (WELL).

- **WEL1RP**: Reads location and Q value (discharge or recharge rate) for all wells.
  
  Note: Q is entered as a negative number for well discharge and as a positive number for well recharge.

- **WEL1FM**: Subtracts Q values from the term RHS for each cell containing pumping wells.

- **WEL1BD**: Calculates the rates and accumulated volume of recharge to or discharge from the flow system by pumping wells.
This module allocates space in the X array to store the list of wells. The X array is a pool of memory space from which space is allocated for tables, lists, and arrays.

1. Print a message identifying the package and initialize NWELLS (a counter containing the number of wells).

2. Read and print MXWELL (the maximum number of wells) and IWELBD (the unit number for cell-by-cell flow terms or a flag indicating that cell-by-cell flow terms should be printed).

3. Set LCWELL, which will point to the first element in the well list (WELL), equal to ISUM, which is currently pointing to the first unallocated element in the X array.

4. Calculate the amount of space needed for the well list (four values for each cell—row, column, layer, and rate) and add it to ISUM.

5. Print the number of elements in the X array used by the Well Package.

6. If the pointer to the lowest unallocated element in the X array (ISUM) is greater than the length of the X array (LENX), print a message warning that the X array will have to be enlarged.

7. RETURN.
MXWELL is the maximum number of wells that will be active at any one time during the simulation.

IWELCB is a flag and a unit number.

If IWELCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.

If IWELCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IWELCB < 0, well recharge will be printed whenever ICBCFL is set.

LCWELL is a location pointer to the first storage location occupied by the well list.

ISUM is the location of the lowest unallocated storage location in the X array.

X array is the pool of memory space allocated for storing specific tables, arrays, and lists.

LENX is the size of the X array.
SUBROUTINE WEL1AL(ISUM, LENX, LCWELL, MXWELL, NWELLS, IN, IOUT, IWELCB)
C
C-----VERSION 1538 12MAY1987 WEL1AL
C
C ALLOCATE ARRAY STORAGE FOR WELL PACKAGE
C **********~**************************************************~ **********
C SPECIFICATIONS:
C -----------------------------------------------
C -----------------------------------------------
C
C1------IDENTIFY PACKAGE AND INITIALIZE NWELLS
C
WRITE(IOUT,1)IN
1 FORMAT(1HO,'WELL -- WELL PACKAGE, VERSION 1, 9/1/87',
     1' INPUT READ FROM',I3)
NWELLS=0

C
C2------READ MAX NUMBER OF WELLS AND
C2-------UNIT OR FLAG FOR CELL-BY-CELL FLOW TERMS.
C
READ(IN,2) MXWELL, IWELCB
2 FORMAT(2I10)
 WRITE(IOUT,3) MXWELL
3 FORMAT(1H , 'MAXIMUM OF',I5,' WELLS')
   IF(IWELCB.GT.0) WRITE(IOUT,9) IWELCB
9 FORMAT(1X, 'CELL-BY-CELL FLOWS WILL BE RECORDED ON UNIT',I3)
     IF(IWELCB.LT.0) WRITE(IOUT,8)
8 FORMAT(1X, 'CELL-BY-CELL FLOWS WILL BE PRINTED WHEN ICBCFL NOT 0')
C
C3------SET LCWELL EQUAL TO LOCATION OF WELL LIST IN X ARRAY.
LCWELL=ISUM
C
C4------ADD AMOUNT OF SPACE USED BY WELL LIST TO ISUM.
ISP=4*MXWELL
 ISUM=ISUM+ISP
C
C5------PRINT NUMBER OF SPACES IN X ARRAY USED BY WELL PACKAGE.
 WRITE(IOUT,4) ISP
4 FORMAT(1X,'I8,' ELEMENTS IN X ARRAY ARE USED FOR WELLS')
 ISUM1=ISUM-1
 WRITE(IOUT,5) ISUM1, LENX
5 FORMAT(1X, 'I8,' ELEMENTS OF X ARRAY USED OUT OF',I8)
C
C6------IF THERE ISN'T ENOUGH SPACE IN THE X ARRAY THEN PRINT
C6-------A WARNING MESSAGE.
   IF(ISUM1.GT.LENX) WRITE(IOUT,6)
6 FORMAT(1X,' ***X ARRAY MUST BE DIMENSIONED LARGER***')
C7------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISP</td>
<td>Module</td>
<td>Number of words in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.</td>
</tr>
<tr>
<td>ISUM1</td>
<td>Module</td>
<td>ISUM-1.</td>
</tr>
<tr>
<td>IWELCB</td>
<td>Package</td>
<td>Flag and a unit number. &gt; 0, unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set. = 0, cell-by-cell flow terms will not be printed or recorded. &lt; 0, well recharge will be printed whenever ICBCFL is set.</td>
</tr>
<tr>
<td>LCWELL</td>
<td>Package</td>
<td>Location in the X array of the first element of array WELL</td>
</tr>
<tr>
<td>LENX</td>
<td>Global</td>
<td>Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.</td>
</tr>
<tr>
<td>MXWELL</td>
<td>Package</td>
<td>Maximum number of wells active at any one time.</td>
</tr>
<tr>
<td>NWELLS</td>
<td>Package</td>
<td>Number of wells active during the current stress period.</td>
</tr>
</tbody>
</table>
Narrative for Module WELLRP

This module reads data to build the WELL list.

1. Read ITMP.
   
   (a) If ITMP is less than zero, the well data read for the last stress period will be reused. Print a message to that effect and RETURN.
   
   (b) If ITMP is greater than or equal to zero, it is equal to the number of wells (NWELLS) in the current stress period.

2. If the number of wells (NWELLS) in the current stress period is greater than the number specified as the maximum for the simulation (MXWELL), STOP.

3. Print the number of wells in the current stress period (NWELLS).

4. If there are no wells in the current stress period (NWELLS), bypass further well processing.

5. For each well, read and print the layer, row, column, and well recharge rate.

6. RETURN.
ITMP is a flag and/or the number of wells. If it is less than zero, it is a flag which indicates that the well data from the last stress period will be reused. If it is greater than or equal to zero, it is the number of wells active during the current stress period.

NWELLS is the number of wells active during the current stress period.

MXWELL is the maximum number of wells which will be active at any one time during the simulation.
SUBROUTINE WEL1RP(WELL,NWELLS,MXWELL,IN,IOUT)

C
C******VERSION 1544 22DEC1982 WEL1RP
C
C READ NEW WELL LOCATIONS AND STRESS RATES

C
C ---------------------------SPECIFICATIONS:
C
DIMENSION WELL(4,MXWELL)

C
C C1------READ ITMP(NUMBER OF WELLS OR FLAG SAYING REUSE WELL DATA)
C READ (IN,1) ITMP
1 FORMAT(I10)
 IF(ITMP.GE.0) GO TO 50
C
C1A------IF ITMP LESS THAN ZERO REUSE DATA. PRINT MESSAGE AND RETURN.
WRITE(IOUT,6)
 6 FORMAT(1HO,'REUSING WELLS FROM LAST STRESS PERIOD')
 RETURN
C
C1B------ITMP>=0. SET NWELLS EQUAL TO ITMP.
 50 NWELLS=ITMP
 IF(NWELLS.LE.MXWELL) GO TO 100
C
C2------NWELLS>MXWELL. PRINT MESSAGE. STOP.
WRITE(IOUT,99) NWELLS,MXWELL
 99 FORMAT(1HO,'NWELLS(',I4,') IS GREATER THAN MXWELL(',I4,')')
STOP
C
C3------PRINT NUMBER OF WELLS IN CURRENT STRESS PERIOD.
100 WRITE (IOUT,2) NWELLS
2 FORMAT(1HO,10X,I4,'WELLS')
C
C4------IF THERE ARE NO ACTIVE WELLS IN THIS STRESS PERIOD THEN RETURN
 IF(NWELLS.EQ.0) GO TO 260
C
C5------READ AND PRINT LAYER,ROW,COLUMN AND RECHARGE RATE.
 WRITE(IOUT,3)
3 FORMAT(1HO,47X,'LAYER ROW COL STRESS RATE WELL NO.'/ 1,48X,45(''-'))
 DO 250 II=1,NWELLS
 READ (IN,4) K,I,J,Q
4 FORMAT(3110,F10.0)
 WRITE (IOUT,5) K,I,J,Q,II
5 FORMAT(48X,I3,18,17,G16.5,18)
 WELL(1,II)=K
 WELL(2,II)=I
 WELL(3,II)=J
 WELL(4,II)=Q
250 CONTINUE
C
C6------RETURN
260 RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Module</td>
<td>Row number of cell containing well.</td>
</tr>
<tr>
<td>II</td>
<td>Module</td>
<td>Index for wells.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
</tbody>
</table>
| ITMP     | Module | Flag or number of wells.  
|          |       | > 0, number of wells active during the current stress period.  
|          |       | < 0, same wells active during the last stress period will be active during the current stress period. |
| J        | Module | Column number of cell containing well. |
| K        | Module | Layer number of cell containing well. |
| MXWELL   | Package | Maximum number of wells active at any one time. |
| NWELLS   | Package | Number of wells active during the current stress period. |
| Q        | Module | Rate at which the well adds water to the aquifer (negative for discharging well). |
| WELL     | Package | DIMENSION (4,MXWELL), For each well: layer, row, column, and recharge rate of the well. |
This module adds terms representing well recharge to the accumulator in which the term RHS is formulated.

1. If NWELLS is less than or equal to zero in the current stress period, there are no wells. RETURN.

2. For each well in the WELL list:

   (a) If the cell containing the well is external (IBOUND (IC,IR,IL) ≤ 0), bypass processing on this well and go on to the next well.

   (b) If the cell containing the well is active, subtract the value of Q from the accumulator RHS for that cell.

3. RETURN.
SUBROUTINE WEL1FM(NWELLS, MXWELL, RHS, WELL, IBOUND,
  NCOL, NROW, NLAY)

C-----VERSION 1233 12MAY1987 WEL1FM
C
C SUBTRACT Q FROM RHS
C
C SPECIFICATIONS:
C---------------------------------------------------------------
DIMENSION RHS(NCOL, NROW, NLAY), WELL(4, MXWELL),
  IBOUND(NCOL, NROW, NLAY)
C---------------------------------------------------------------
C1------IF NUMBER OF WELLS <= 0 THEN RETURN.
  IF(NWELLS.LE.0) RETURN
C
C2------PROCESS EACH WELL IN THE WELL LIST.
  DO 100 L=1, NWELLS
    IR=WELL(2,L)
    IC=WELL(3,L)
    IL=WELL(1,L)
    Q=WELL(4,L)
    C2A------IF THE CELL IS INACTIVE THEN BYPASS PROCESSING.
      IF(IBOUND(IC, IR, IL).LE.0) GO TO 100
    C2B------IF THE CELL IS VARIABLE HEAD THEN SUBTRACT Q FROM
      THE RHS ACCUMULATOR.
      RHS(IC, IR, IL)=RHS(IC, IR, IL)-Q
  100 CONTINUE
C
C3------RETURN
  RETURN
END
### List of Variables for Module WEL1FM

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
</table>
| IBOUND   | Global  | DIMENSION (NCOL,NROW,NLAY), Status of each cell.  
|          |         | < 0, constant-head cell  
|          |         | = 0, inactive cell  
|          |         | > 0, variable-head cell |
| IC       | Module  | Index for columns. |
| IL       | Module  | Index for layers. |
| IOUT     | Global  | Primary unit number for all printed output. IOUT = 6. |
| IR       | Module  | Index for rows. |
| L        | Module  | Index for wells. |
| MXWELL   | Package | Maximum number of wells active at any one time. |
| NCOL     | Global  | Number of columns in the grid. |
| NLAY     | Global  | Number of layers in the grid. |
| NROW     | Global  | Number of rows in the grid. |
| NWELLS   | Package | Number of wells active during the current stress period. |
| Q        | Module  | Rate at which the well adds water to the aquifer (negative for discharging wells). |
| RHS      | Global  | DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages. |
| WELL     | Package | DIMENSION (4,MXWELL), For each well: layer, row, column, and recharge rate of the well. |
This module calculates rates and volumes transferred between the aquifer and wells.

1. Clear the rate accumulators RATIN and RATOUT and the flag (IBD) which indicates that cell-by-cell flow terms should be recorded on a disk.

2. If there are no wells, skip down to step 7.

3. Determine if the cell-by-cell flow terms for wells will be written on a disk. They will be if (1) this is the proper time step (ICBCFL is not equal to zero), (2) if the channel for well-budget terms (IWELCB) is greater than zero, and (3) if the number of wells (NWELLS) is greater than zero.

4. If budget terms are to be written on a disk, set IBD = 1 and clear the buffer (BUFF) in which they will be accumulated.

5. If the number of wells in the current stress period (NWELLS) is not equal to zero, then for each cell in the well list:

   (a) If the cell containing the well is external (IBOUND(I,J,K) < 0), bypass further processing of the cell.

   (b) If the user has requested that cell-by-cell rates be printed (IWELCB < 0 and ICBCFL ≠ 0), print the rate (Q).

   (c) If the budget terms are to be saved on a disk, add the flow rate (Q) to the buffer (BUFF).

   (d) If Q is positive, add it to RATIN.

   (e) If Q is negative, add it to RATOUT.

6. If the cell-by-cell flow terms are to be recorded, call module UBDCSV to write the contents of buffer (BUFF) onto the disk.

7. Move RATIN and RATOUT into the VBVL array for printing by BAS1OT.

8. Add RATIN and RATOUT multiplied by the time-step length to the volume accumulators in the VRVL array for printing by BAS1OT.

9. Move the well budget term labels to VBNM for printing by BAS1OT.

10. Increment the budget-term counter (MSUM).

11. RETURN.
Flow Chart for Module WEL1BD

RATIN is an accumulator to which all flows into the aquifer are added.

RATOUT is an accumulator to which all flows out of the aquifer are added.

IBD is a flag which, if set, causes cell-by-cell flow terms for well flow to be recorded.

BUFFER is an array in which values are stored as they are being gathered for printing or recording.

EXTERNAL: a cell is said to be external if it is either no flow or constant head (i.e., an equation is not formulated for the cell).

Q is the rate at which the well recharges the aquifer. A discharging well is represented by a negative rate.

IWELCB is a flag and a unit number.

If IWELCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.

If IWELCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IWELCB < 0, well recharge rate will be printed whenever ICBCFL is set.

ICBCFL is a flag.

If ICBCFL ≠ 0, cell-by-cell flow terms will be either printed or recorded (depending on IWELCB) for the current time step.
SUBROUTINE WEL1BD(NWELLS, MXWELL, VBNM, VBVL, MSUM, WELL, IBOUND, DELT,
1                  NCOL, NROW, NLAY, KSTP, KPER, IWELCB, ICBCFL, BUFF, IOUT)

C----- VERSION 1509 12MAY1987 WEL1BD
C******************************************~*************~*************~
C CALCULATE VOLUMETRIC BUDGET FOR WELLS
C******************************************~*************~*************~
C SPECIFICATIONS:
C-------------------------------------------------------------------
CHARACTER*4 VBNM, TEXT
DIMENSION VBNM(4,MSUM), VBVL(4,MSUM), WELL(4, MXWELL),
1       IBOUND(NCOL, NROW, NLAY), BUFF(NCOL, NROW, NLAY)
DIMENSION TEXT(4)
C
DATA TEXT(1),TEXT(2),TEXT(3),TEXT(4) /' ',' ',' W','ellS'/
C-------------------------------------------------------------------
C1-------CLEAR RATIN AND RATOUT ACCUMULATORS.
  RATIN=0.
  RATOUT=0.
  IBD=0
C
C2-------IF THERE ARE NO WELLS DO NOT ACCUMULATE FLOW
  IF(NWELLS.EQ.0) GO TO 200
C
C3-------TEST TO SEE IF CELL-BY-CELL FLOW TERMS WILL BE RECORDED.
  IF(ICBCFL.EQ.0 .OR. IWELCB.LE.0) GO TO 60
C
C4-------IF CELL-BY-CELL FLOWS WILL BE SAVED THEN CLEAR THE BUFFER.
  IBD=1
  DO 50 IL=1, NLAY
  DO 50 IR=1, NROW
  DO 50 IC=1, NCOL
  BUFF(IC,IR,IL)=0.
  50 CONTINUE
C
C5-------PROCESS WELLS ONE AT A TIME.
  60 DO 100 L=1, NWELLS
    IR=WELL(2,L)
    IC=WELL(3,L)
    IL=WELL(1,L)
    Q=WELL(4,L)
C
C5A------IF THE CELL IS EXTERNAL IGNORE IT.
IF(BOUND(IC,IR,IL).LE.0) GO TO 100

C5B---PRINT THE INDIVIDUAL RATES IF REQUESTED(IWELCB<0).
    IF(IWELCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT,900) (TEXT(N),N=1:4),
    1 KPER,KSTP,L,IL,IR,IC,Q
900 FORMAT(110,4/A4,' PERIOD',I3,', STEP',I3,' WELL',I4,
    1 ' LAYER',I3,' ROW ',I4,' COL',I4,' RATE',G15.7)
C
C5C-----IF CELL-BY-CELL FLOWS ARE TO BE SAVED THEN ADD THEM TO BUFFER.
    IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+Q
    IF(Q) 90,100,80
C
C5D-----PUMPING RATE IS POSITIVE(RECHARGE). ADD IT TO RATIN.
    80 RATIN=RATIN+Q
    GO TO 100
C
C5E-----PUMPING RATE IS NEGATIVE(DISCHARGE). ADD IT TO RATOUT.
    90 RATOUT=RATOUT-Q
    100 CONTINUE
C
C6-------IF CELL-BY-CELL FLOWS WILL BE SAVED CALL UBUDSV TO RECORD THEM
    IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IWELCB,BUFF,NCOL,NROW,
    1 NLAY,IOUT)
C
C7-------MOVE RATES INTO VBVL FOR PRINTING BY MODULE BAS10T.
    200 VBVL(3,MSUM)=RATIN
    VBVL(4,MSUM)=RATOUT
C
C8-------MOVE RATES TIMES TIME STEP LENGTH INTO VBVL ACCUMULATORS.
    VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
    VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
C
C9-------MOVE BUDGET TERM LABELS INTO VBNM FOR PRINTING.
    VBNM(1,MSUM)=TEXT(1)
    VBNM(2,MSUM)=TEXT(2)
    VBNM(3,MSUM)=TEXT(3)
    VBNM(4,MSUM)=TEXT(4)
C
C10------INCREMENT BUDGET TERM COUNTER(MSUM).
    MSUM=MSUM+1
C
C11------RETURN
    RETURN
END

8-20
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>IBD</td>
<td>Module</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms for this package will not be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td># 0, cell-by-cell flow terms for this package will be recorded.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>ICBCFL</td>
<td>Global</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be recorded or printed for the current time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td># 0, cell-by-cell flow terms will be either printed or recorded (depending on IWELCB) for the current time step.</td>
</tr>
<tr>
<td>IL</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IWELCB</td>
<td>Package</td>
<td>Flag and a unit number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be printed or recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, well recharge rate will be printed whenever ICBCFL is set.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for wells.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>MXWELL</td>
<td>Package</td>
<td>Maximum number of wells active at any one time.</td>
</tr>
</tbody>
</table>

8-21
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>NWELLS</td>
<td>Package</td>
<td>Number of wells active during the current stress period.</td>
</tr>
<tr>
<td>Q</td>
<td>Module</td>
<td>Rate at which the well adds water to the aquifer (negative for discharging wells).</td>
</tr>
<tr>
<td>RATIN</td>
<td>Module</td>
<td>Accumulator for the total flow into the flow field from wells.</td>
</tr>
<tr>
<td>RATOUT</td>
<td>Module</td>
<td>Accumulator for the total flow out of the flow field into wells.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget. For flow component N, the values in VBVL are: (1,N) Rate for the current time step into the flow field. (2,N) Rate for the current time step out of the flow field. (3,N) Volume into the flow field during simulation. (4,N) Volume out of the flow field during simulation.</td>
</tr>
<tr>
<td>WELL</td>
<td>Package</td>
<td>DIMENSION (4,MXWELL), For each well: layer, row, column, and recharge rate of the well.</td>
</tr>
</tbody>
</table>
CHAPTER 9

DRAIN PACKAGE

Conceptualization and Implementation

The Drain Package is designed to simulate the effects of features such as agricultural drains, which remove water from the aquifer at a rate proportional to the difference between the head in the aquifer and some fixed head or elevation, so long as the head in the aquifer is above that elevation, but which have no effect if head falls below that level. The discussion in this section is phrased on the assumption that the features to be simulated are actually agricultural drains.

Figure 39 shows a cross section through a cell, illustrating concepts underlying the simulation of drains in the model. The drain is assumed to run only partially full, so that head within the drain is approximately equal to the median drain elevation, \(d_{i,j,k}\). The head computed by the model for cell \(i,j,k\) (\(h_{i,j,k}\)) is actually an average value for the cell, and is normally assumed to prevail at some distance from the drain itself. The drain head, \(d_{i,j,k}\) prevails only locally, within the drain--it does not characterize the cell as a whole. Between the drain and the area in which head \(h_{i,j,k}\) prevails there exists a radial or semiradial flow pattern in the vertical plane, normally characterized by progressively steeper head gradients as the drain is approached. The head loss within this converging flow pattern forms one part of the head difference \(h_{i,j,k} - d_{i,j,k}\). An additional component of head loss may occur in the immediate vicinity of the drain if the hydraulic conductivity in that region differs from the average value used for cell \(i,j,k\)--because of the presence of foreign material around
Figure 39.—Cross section through cell i,j,k illustrating head loss in convergent flow into drain.
the drain pipe, or channel-bed material in the case of an open drain (figure 40). Finally, head losses occur through the wall of a drain pipe, depending upon the number and size of the openings in the pipe, and the degree to which those openings may be blocked by chemical precipitates, plant roots, etc.

The three processes discussed above—convergent flow toward the drain, flow through material of different conductivity immediately around the drain, and flow through the wall of the drain—each generate head losses which may be assumed proportional to the discharge, $Q_d$, through the system—that is, the discharge from cell i,j,k into the drain. Because these head losses occur in series, the total head loss $h_{i,j,k} - d_{i,j,k}$ may also be taken as proportional to $Q_d$. This has been done in the method of simulation embodied in the Drain Package. That is, it has been assumed that the drain function is described by the equation pair

$$Q_{D_{i,j,k}} = CD_{i,j,k}(h_{i,j,k} - d_{i,j,k}) \quad \text{for} \quad h_{i,j,k} > d_{i,j,k} \quad (69)$$

$$Q_{D_{i,j,k}} = 0 \quad \text{for} \quad h_{i,j,k} \leq d_{i,j,k} \quad (70)$$

The coefficient $CD_{i,j,k}$ of equation (69) is a lumped (or equivalent) conductance describing all of the head loss between the drain and the region of cell i,j,k in which the head $h_{i,j,k}$ can be the assumed to prevail. It depends on the characteristics of the convergent flow pattern toward the drain, as well as on the characteristics of the drain itself and its immediate environment.

One could attempt to calculate values for $CD$ by developing approximate equations for conductance for the three flow processes, and then calculate the equivalent series conductance. The conductance for each process would be
Figure 40.—Factors affecting head loss immediately around a drain: (a) buried drain pipe in backfilled ditch and (b) open drain.
based on the formulation of a one-dimensional flow equation. The formulations vary significantly depending on the specific drain system being simulated, so no general formulation for calculating CD is presented here. Also, in most situations a specific formulation would require detailed information that is not usually available, such as detailed head distribution around the drain, aquifer hydraulic conductivity near the drain, distribution of the fill material, hydraulic conductivity of fill material, number and size of the drain pipe openings, the amount of clogging materials, and the hydraulic conductivity of the clogging materials. In practice, it is more common to calculate CD from measured values of QD and h-d using equation (69). If h-d is not accurately known, CD is usually adjusted during model calibration in order to match measured values of QD to model calculated values.

Figure 41 shows a graph of QD vs. $h_{i,j,k}$ as defined by equations (69) and (70); the function is similar to that for flow between a surface stream and the aquifer (figure 36) except that flow into the aquifer is excluded, and positive values of QD have been taken as corresponding to flow into the drain. With proper selection of coefficients, the River Package could in fact be utilized to perform the functions of the Drain Package.

Because $QD_{i,j,k}$ in equation (69) has been taken as a flow out of cell $i,j,k$, it must be subtracted from the left side of equation (24) for each cell affected by a drain, provided the head $h_{i,j,k}$ is above the drain elevation. This is accomplished in the Drain Package by testing to determine whether head exceeds drain elevation, and if so, by adding the term $-QD_{i,j,k}$ to $HCOF_{i,j,k}$ (equation (26)) and adding the term $-CD_{i,j,k}d_{i,j,k}$ to RHS$_{i,j,k}$, as the matrix equations are assembled.
Figure 41.—Plot of flow, QD, into a drain as a function of head, h, in a cell where the elevation of the drain is d and the conductance is CD.
Drain Package Input

Input to the Drain (DRN) Package is read from the unit specified in IUNIT(3).

FOR EACH SIMULATION

DRNIAL

1. Data: MXDRN IDRNCB
   Format: I10 I10

FOR EACH STRESS PERIOD

DRNIRP

2. Data: ITMP
   Format: I10

3. Data: Layer Row Col Elevation Cond
   Format: I10 I10 I10 F10.0 F10.0

   (Input item 3 normally consists of one record for each drain. If ITMP is negative or zero, item 3 will not be read.)

Explanation of Fields Used in Input Instructions

MXDRN—is the maximum number of drain cells active at one time.

IDRNCB—is a flag and a unit number.

   If IDRNCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL (see Output Control) is set.

   If IDRNCB = 0, cell-by-cell flow terms will not be printed or recorded.

   If IDRNCB < 0, drain leakage for each cell will be printed whenever ICBCFL is set.

ITMP—is a flag and a counter.

   If ITMP < 0, drain data from the last stress period will be reused.

   If ITMP > 0, ITMP will be the number of drains active during the current stress period.
Layer—is the layer number of the cell containing the drain.

Row—is the row number of the cell containing the drain.

Column—is the column number of the cell containing the drain.

Elevation—is elevation of the drain.

Condo—is the hydraulic conductance of the interface between the aquifer and the drain.
### SAMPLE INPUT TO THE DRAIN PACKAGE

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[MXDRIR, IDRNCB]</td>
</tr>
<tr>
<td>2</td>
<td>[ITMP: FOR FIRST STRESS PERIOD]</td>
</tr>
<tr>
<td>3</td>
<td>[LAYER, ROW, COLUMN, ELEVATION, COND]</td>
</tr>
<tr>
<td>4</td>
<td>[LAYER, ROW, COLUMN, ELEVATION, COND]</td>
</tr>
<tr>
<td>5</td>
<td>[LAYER, ROW, COLUMN, ELEVATION, COND]</td>
</tr>
<tr>
<td>6</td>
<td>[ITMP: FOR SECOND STRESS PERIOD]</td>
</tr>
<tr>
<td>7</td>
<td>[ITMP: FOR THIRD STRESS PERIOD]</td>
</tr>
<tr>
<td>8</td>
<td>[ITMP: FOR FOURTH STRESS PERIOD]</td>
</tr>
<tr>
<td>9</td>
<td>[LAYER, ROW, COLUMN, ELEVATION, COND]</td>
</tr>
<tr>
<td>10</td>
<td>[LAYER, ROW, COLUMN, ELEVATION, COND]</td>
</tr>
<tr>
<td>11</td>
<td>[ITMP: FOR FIFTH STRESS PERIOD]</td>
</tr>
</tbody>
</table>

### INPUT RECORDS

<table>
<thead>
<tr>
<th>RECORD</th>
<th>LAYER</th>
<th>ROW</th>
<th>COLUMN</th>
<th>ELEVATION</th>
<th>COND</th>
<th>ITMP</th>
<th>FOR FIRST DRAIN</th>
<th>FOR SECOND DRAIN</th>
<th>FOR THIRD DRAIN</th>
<th>FOR SECOND DRAIN</th>
<th>FOR THIRD DRAIN</th>
<th>FOR SECOND DRAIN</th>
<th>FOR THIRD DRAIN</th>
<th>FOR SECOND DRAIN</th>
<th>FOR THIRD DRAIN</th>
<th>FOR SECOND DRAIN</th>
<th>FOR THIRD DRAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>210</td>
<td>.7</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>210</td>
<td>.8</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>220</td>
<td>.7</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>
Module Documentation for the Drain Package

The Drain Package (DRN1) consists of four modules, all of which are called by the MAIN program. The modules are:

- **DRN1AL**: Allocates space for an array that contains the drain list (DRAI).
- **DRN1RP**: Reads location, drain elevation, and drain conductance of each cell containing a drain.
- **DRN1FM**: Adds the terms \(-CD_{i,j,k}\) and \(-CD_{i,j,k}d_{i,j,k}\) to the accumulators \(HCOFi,j,k\) and \(RHSi,j,k\), respectively.
- **DRN1BD**: Calculates the rates and accumulated volume of drainage from the flow system.
Narrative for Module DRNIAL

This module allocates space in the X array to store the list of drains.

1. Print a message identifying the package and initialize NDRAIN (number of drains).

2. Read and print MXDRAN (the maximum number of drains) and IDRNCB (the file number for saving cell-by-cell flow terms or a flag indicating that cell-by-cell flow terms should be printed).

3. Set LCDRAI (which will point to the first element in the drain list) equal to ISUM (which points to the first unallocated element in the X array).

4. Calculate the amount of space needed for the drain list (five values for each drain—row, column, layer, drain elevation, and drain conductance).

5. Print the number of elements in the X array used by the Drain Package.

6. RETURN.
Flow Chart for Module DRNIAL

NDRAIN is the number of drains being simulated at any given time.

MXDRN is the maximum number of drains simulated.

IDRNCB is a flag and a unit number.

If IDRNCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.

If IDRNCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IDRNCB < 0, drain leakage for each drain will be printed whenever ICBCFL is set.

LCDRAI is the location, in the X array, of the list of drain data (DRAI).
SUBROUTINE DRN1AL(ISUM, LENX, LCDRAI, NDRAIN, MXDRN, IN, IOUT, IDRNCB)

C ***VERSION 1604 12MAY1987 DRN1AL*************
C ALLOCATE ARRAY STORAGE FOR DRAIN PACKAGE
C SPECIFICATIONS:
C
C IDENTIFY PACKAGE AND INITIALIZE NDRAIN.
C WRITE(IOUT,1)IN
C 1 FORMAT(1HO,'DRN1 -- DRAIN PACKAGE, VERSION 1, 9/1/87', I3)
C NDRAIN=0
C
C READ & PRINT MXDRN & IDRNCB(UNIT & FLAG FOR CELL-BY-CELL FLOW)
C READ(IN,2) MXDRN,IDRNCB
C WRITE(IOUT,3) MXDRN
C 3 FORMAT(1H, 'MAXIMUM OF', I5, ' DRAINS')
C IF(IDRNCB.GT.0) WRITE(IOUT,9) IDRNCB
C 9 FORMAT(1X, 'CELL-BY-CELL FLOWS WILL BE RECORDED ON UNIT', I3)
C IF(IDRNCB.LT.0) WRITE(IOUT,8)
C 8 FORMAT(1X, 'CELL-BY-CELL FLOWS WILL BE PRINTED WHEN ICBCFL NOT 0')
C
C SET LCDRAI EQUAL TO ADDRESS OF FIRST UNUSED SPACE IN X.
C LCDRAI=ISUM
C
C CALCULATE AMOUNT OF SPACE USED BY THE DRAIN PACKAGE.
C ISP=5*MXDRN
C ISUM=ISUM+ISP
C
C PRINT AMOUNT OF SPACE USED BY DRAIN PACKAGE.
C WRITE(IOUT,4) ISP
C 4 FORMAT(1X, I8, ' ELEMENTS IN X ARRAY ARE USED FOR DRAINS')
C ISUM1=ISUM-1
C WRITE(IOUT,5) ISUM1, LENX
C 5 FORMAT(1X, I8, ' ELEMENTS OF X ARRAY USED OUT OF', I8)
C IF(ISUM1.GT.LENX) WRITE(IOUT,6)
C 6 FORMAT(1X, ' ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C RETURN
C RETURN
C END
**List of Variables for Module DRNIAL**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
</table>
| IDRNCB   | Package | Flag and a unit number.  
  > 0, unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.  
  = 0, cell-by-cell flow terms will be neither printed nor recorded.  
  < 0, leakage for each drain will be printed. |
| IN       | Package | Primary unit number from which input for this package will be read. |
| IOUT     | Global  | Primary unit number for all printed output. IOUT = 6. |
| ISP      | Module  | Number of words in the X array allocated by this module. |
| ISUM     | Global  | Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM. |
| ISUM1    | Module  | ISUM - 1. |
| LCDRAI   | Package | Location in the X array of the first element of array DRAI. |
| LENX     | Global  | Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN Program. |
| MXDRN    | Package | Maximum number of drains active at any one time. |
| NDRAIN   | Package | Number of drains active during the current stress period. |
This module reads data to build the drain list.

1. Read ITMP. ITMP is the number of drains or a flag indicating that drain data from the previous stress period should be reused.

2. Test ITMP. If ITMP is less than zero, the drain data read for the last stress period will be reused. Print a message to that effect and RETURN.

3. If ITMP is greater than or equal to zero, it is the number of drains for this stress period. Set the number of drains (NDRAIN) in the current stress period equal to ITMP.

4. Compare the number of drains (NDRAIN) in the current stress period to the number specified as the maximum for the simulation (MXDRN). If NDRAIN is greater than MXDRN, STOP.

5. Print the number of drains in the current stress period (NDRAIN).

6. See if there are any drains. If there are no drains in the current stress period (NDRAIN = 0), bypass further drain processing.

7. Read and print the layer, row, column, elevation, and conductance for each drain.

8. RETURN.
ITMP is both a flag and a counter. If it is greater than or equal to zero, it is the number of drains to be simulated during the current stress period. If it is less than zero, it indicates that the drains simulated in the last stress period should be simulated in the current stress period.

MXDRN is the maximum number of drains to be simulated.
SUBROUTINE DRN1RP(DRAI,NDRAIN,MXDRN,IN,IOUT)

C
C------VERSION 1603 25APR1983 DRN1RP
C ******************************************************************
C READ DRAIN LOCATIONS, ELEVATIONS, AND CONDUCTANCES
C ******************************************************************

C SPECIFICATIONS:
C DIMENSION DRAI(5,MXDRN)
C
C1------READ ITMP(NUMBER OF DRAIN CELLS OR FLAG TO REUSE DATA)
READ(IN,8) ITMP
8 FORMAT(1I0)
C2------TEST ITMP
IF(ITMP.GE.0) GO TO 50
C
C2A------IF ITMP<0 THEN REUSE DATA FROM LAST STRESS PERIOD.
WRITE(IOUT,7) 7 FORMAT(1H0,'REUSING DRAINS FROM LAST STRESS PERIOD')
RETURN
C3------IF ITMP>=0 THEN IT IS THE NUMBER OF DRAINS.
50 NDRAIN=ITMP
IF(NDRAIN.LE.MXDRN) GO TO 100
C
C4------IF NDRAIN>MXDRN THEN STOP
WRITE(IOUT,99) NDRAIN,MXDRN
99 FORMAT(1H0,'NDRAIN(',I4,') IS GREATER THAN MXDRN(',I4,')')
STOP
C
C5------PRINT NUMBER OF DRAINS IN THIS STRESS PERIOD.
100 WRITE(IOUT,1) NDRAIN
1 FORMAT(1H0,'//I5,' DRAINS')
C
C6------IF THERE ARE NO DRAINS THEN RETURN.
IF(NDRAIN.EQ.0) GO TO 260
C
C7------READ AND PRINT DATA FOR EACH DRAIN.
WRITE(IOUT,3)
3 FORMAT(1H0,15X,'LAYER',5X,'ROW',5X,1,'COL ELEVATION CONDUCTANCE DRAIN NO.'/1X,15X,60('-'))
DO 250 II=1,NDRAIN
READ (IN,4) K,I,J,DRAI(4,II),DRAI(5,II)
4 FORMAT(3110,2F10.0)
WRITE (IOUT,5) K,I,J,DRAI(4,II),DRAI(5,II),II
5 FORMAT(1X,15X,I4,I9,I8,Gl3.4,Gl4.4,I8)
DRAI(1,II)=K
DRAI(2,II)=I
DRAI(3,II)=J
250 CONTINUE
C
C8------RETURN
260 RETURN
C
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRAI</td>
<td>Package</td>
<td>DIMENSION (5,MXDRN), For each drain: layer, row, column, head in drain, and conductance into drain.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>II</td>
<td>Module</td>
<td>Index for drains.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ITMP</td>
<td>Module</td>
<td>Flag or number of drains. &gt; 0, number of drains active during the current stress period. &lt; 0, same drains active during the last stress period will be active during the current stress period.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>MXDRN</td>
<td>Package</td>
<td>Maximum number of drains active at any one time.</td>
</tr>
<tr>
<td>NDRAIN</td>
<td>Package</td>
<td>Number of drains active during the current stress period.</td>
</tr>
</tbody>
</table>
Narrative for Module DRNIFM

This module adds terms representing drain leakage to the accumulators HCOF and RHS.

1. If NDRAIN is less than or equal to zero in the current stress period, there are no drains. RETURN.

2. For each drain in the drain list, DO STEPS 3-7.

3. Determine the column (IC), row (IR), and layer (IL).

4. If the cell is external (IBOUND(IC, IR, IL) ≤ 0), bypass processing on this drain and go on to the next drain.

5. If the cell is internal, get the drain data (elevation and conductance).

6. If the head in the aquifer (HHNEW) is greater than the elevation of the drain, there is no drain leakage. RETURN.

7. If the head in the aquifer (HHNEW) is greater than the elevation of the drain (EL), add the term \(-C*EL\) (C is the drain conductance) to the accumulator RHS and the term \(-C\) to the accumulator HCOF.

8. RETURN.
RHS is an accumulator in which the right hand side of the equation is formulated.

HCOF is an accumulator in which the coefficient of head in the cell is formulated.

Flow Chart for Module DRN1FM

1. ENTER DRN1FM

2. FOR EACH DRAIN

3. DETERMINE ROW, COLUMN, AND LAYER

4. IS THIS CELL NO FLOW OR CONSTANT HEAD?

5. YES

6. AQUIFER HEAD < DRAIN ELEVATION?

7. NO

8. ADD TERMS TO RHS AND HCOF FOR THIS CELL

9. RETURN
SUBROUTINE DRN1FM(NDRAIN, MXDRN, DRAI, HNEW, HCOF, RHS, IBOUND,
   1   NCOL, NROW, NLAY)

C -----VERSION 1030 10APR1985 DRN1FM
C
C ************************************************************
C ADD DRAIN FLOW TO SOURCE TERM
C ************************************************************
C
C SPECIFICATIONS:
C
C ------------------------------------------------------------------
C DOUBLE PRECISION HNEW
C
C DIMENSION DRAI(5, MXDRN), HNEW(NCOL, NROW, NLAY),
C           RHS(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
C           HCOF(NCOL, NROW, NLAY)
C ------------------------------------------------------------------
C
C1------IF NDRAIN<=0 THERE ARE NO DRAINS. RETURN
   IF(NDRAIN.LE.0) RETURN
C
C2------PROCESS EACH CELL IN THE DRAIN LIST
   DO 100 L=1,NDRAIN
C
C3------GET COLUMN, ROW AND LAYER OF CELL CONTAINING DRAIN.
   IL=DRAI(1,L)
   IR=DRAI(2,L)
   IC=DRAI(3,L)
C
C4------IF THE CELL IS EXTERNAL SKIP IT.
   IF(BOUND(IC,IR,IL),LE.0) GO TO 100
C
C5------IF THE CELL IS INTERNAL GET THE DRAIN DATA.
   EL=DRAI(4,L)
   HNEW=HNEW(IC,IR,IL)
C
C6------IF HEAD IS LOWER THAN DRAIN THEN SKIP THIS CELL.
   IF(HNEW.LE.EL) GO TO 100
C
C7------HEAD IS HIGHER THAN DRAIN. ADD TERMS TO RHS AND HCOF.
   C=DRAI(5,L)
   HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-C
   RHS(IC,IR,IL)=RHS(IC,IR,IL)-C*EL
   100 CONTINUE
C
C8------RETURN
   RETURN
C
END
List of Variables for Module DRN1FM

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Module</td>
<td>Conductance into the drain.</td>
</tr>
<tr>
<td>DRAI</td>
<td>Package</td>
<td>DIMENSION (5,MXDRN), For each drain: layer, row, column, head in the drain and conductance into the drain.</td>
</tr>
<tr>
<td>EL</td>
<td>Module</td>
<td>Elevation of the drain (head in the drain).</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K) in the finite-difference equation.</td>
</tr>
<tr>
<td>HHNEW</td>
<td>Module</td>
<td>Head in the cell containing the drain.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell. (&lt; 0), constant-head cell (= 0), inactive cell (&gt; 0), variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>IL</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>MXDRN</td>
<td>Package</td>
<td>Maximum number of drains active at any one time.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NDRAIN</td>
<td>Package</td>
<td>Number of drains active during the current stress period.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>RHS</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.</td>
</tr>
</tbody>
</table>
Narrative for Module DRN1BD

This module calculates rates and volumes transferred between the aquifer and drains.

1. Initialize the cell-by-cell flow-term flag (IBD) and the rate accumulator (RATOUT).

2. If there are no drains (NDRAIN $\leq$ 0), skip down to step 12 and put zeros into the budget terms for drains.

3. Test to see if cell-by-cell flow terms are to be saved on disk. They will not be saved if either of the following conditions hold: (1) this is not the proper time step (ICBCFL = 0) or (2) cell-by-cell flow terms are not needed for drains during this simulation (IDRNCB $\leq$ 0). If cell-by-cell flow terms will be saved for drains, set the cell-by-cell flow-term flag (IBD) and clear the buffer in which they will be accumulated (BUFF).

4. For each drain, do steps 3-11 accumulating flows into drains.

5. Determine the row, column, and layer of the cell containing the drain.

6. If the cell is external (IBOUND(I,J,K), $\leq$ 0), bypass further processing of this drain.

7. Get the drain parameters from the drain list.

8. If the head in the cell is less than the elevation of the drain, bypass further processing of this drain.
9. If the head in the cell is greater than the elevation of the drain, set "Q" equal to the conductance of the drain (C) times the drain elevation (EL) minus the head in the cell (HHNEW) (Q = C*(EL - HHNEW)). Add Q to the accumulator RATOUT to get the total flow from the aquifer into drains.

10. If the cell-by-cell flow terms are to be printed (IDRNCB < 0 and ICBCFL ≠ 0), print Q.

11. If the cell-by-cell flow terms for drains are to be saved, add Q to the buffer (BUFF).

12. See if the cell-by-cell flow terms are to be saved (IBD = 1). If they are, call module UBUDSV to record the buffer (BUFF) onto disk.

13. Move RATOUT into the VBVL array for printing by BAS10T. Add RATOUT multiplied by the time-step length to the volume accumulator in VBVL for printing by BAS10T. Move the drain budget-term labels to VBNM for print by BAS10T.

14. Increment the budget-term counter (MSUM). See the section in the Basic Package for a detailed explanation of VBVL, VBNM, and MSUM.

15. RETURN.
IBD is a flag which, if set, causes cell-by-cell flow terms for drains to be recorded.

EXTERNAL: a cell is said to be external if it is either no flow or constant head (i.e., an equation is not formulated for the cell).

BUFFER is an array in which values are stored as they are being gathered for printing or recording.

RATOUT is an accumulator to which all flows out of the aquifer are added.

Q is the negative of discharge to a drain.

EL is the elevation of the drain.

IDRNCB is a flag and a unit number.

If IDRNCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.

If IDRNCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IDRNCB < 0, drain leakage for each drain will be printed whenever ICBCFL is set.

ICBCFL is a flag.

If ICBCFL ≠ 0, cell-by-cell flow terms will be either printed or recorded for the current time step.
SUBROUTINE DRN1BD(NDRAIN, MXDRAI, VBNM, VBVL, MSUM, DRAI, DELT, HNEW,
   1   NCOL, NROW, NLAY, IBOUND, KSTEP, KPER, IDRNCB, ICBCFL, BUFF, IOUT)

C*****************************************************************
C CALCULATE VOLUMETRIC BUDGET FOR DRAINS
C*****************************************************************

C SPECIFICATIONS:
C---------------------------------------------------------------
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW

DIMENSION VBNM(4, MSUM), VBVL(4, MSUM), DRAI(5, MXDRAI),
   1   HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
   2   BUFF(NCOL, NROW, NLAY)
DIMENSION TEXT(4)

DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) / ' ', ' ', 'DRAIN' /
---------------------------------------------------------------

C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATORS (RATIN AND RATOUT).
RATOUT=0.
IBD=0

C2------IF THERE ARE NO DRAINS THEN DO NOT ACCUMULATE DRAIN FLOW
IF(NDRAIN.LE.0) GO TO 200

C3------TEST TO SEE IF CELL-BY-CELL FLOW TERMS ARE NEEDED.
IF(ICBCFL.EQ.0 .OR. IDRNCB.LE.0) GO TO 60

C3B------CELL-BY-CELL FLOW TERMS ARE NEEDED SET IBD AND CLEAR BUFFER.
   IBD=1
   DO 50 IL=1, NLAY
   DO 50 IR=1, NROW
   DO 50 IC=1, NCOL
   BUFF(IC, IR, IL)=0.
   50 CONTINUE

C4------FOR EACH DRAIN ACCUMULATE DRAIN FLOW
   60 DO 100 L=1, NDRAIN

C5------GET LAYER, ROW & COLUMN OF CELL CONTAINING REACH.
   IL=DRAI(1, L)
   IR=DRAI(2, L)

9-26
IC=DRAI(3,L)
C
C6------IF CELL IS EXTERNAL IGNORE IT.
    IF(IBOUND(IC,IR,IL).LE.0) GO TO 100
C
C7------GET DRAIN PARAMETERS FROM DRAIN LIST.
    EL=DRAI(4,L)
    C=DRAI(5,L)
    HHNEW=HNEW(IC,IR,IL)
C
C8------IF HEAD LOWER THAN DRAIN THEN FORGET THIS CELL.
    IF(HHNEW.LE.EL) GO TO 100
C
C9------HEAD HIGHER THAN DRAIN. CALCULATE Q=\text{C}*(EL-HHNEW).
C9------SUBTRACT Q FROM RATOUT.
    Q=C*(EL-HHNEW)
    RATOUT=RATOUT-Q
C
C10------PRINT THE INDIVIDUAL RATES IF REQUESTED(IDRNCB<0).
    IF(IDRNCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT,900) (TEXT(N),N=1,4),
      1  KPER,KSTP,L,IR,IC,Q
900 FORMAT(1H0,4A4,' PERIOD',I3,' STEP',I3,' DRAIN',I4,
      1 ' LAYER',I3,' ROW',I4,' COL',I4,' RATE',G15.7)
C
C11------IF C-B-C FLOW TERMS ARE TO BE SAVED THEN ADD Q TO BUFFER.
    IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+Q
    100 CONTINUE
C
C12------IF C-B-C FLOW TERMS WILL BE SAVED CALL UBUDSV TO RECORD THEM.
    IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IDRNCB,BUFF,NCOL,NROW,
      1  NLAY,IOUT)
C
C13------MOVE RATES, VOLUMES & LABELS INTO ARRAYS FOR PRINTING.
    200 VBVL(3,MSUM)=0.
    VBVL(4,MSUM)=RATOUT
    VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
    VBNM(1,MSUM)=TEXT(1)
    VBNM(2,MSUM)=TEXT(2)
    VBNM(3,MSUM)=TEXT(3)
    VBNM(4,MSUM)=TEXT(4)
C
C14------INCREMENT BUDGET TERM COUNTER
    MSUM=MSUM+1
C
C15------RETURN
RETURN
END
## List of Variables for Module DRN1BD

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>C</td>
<td>Module</td>
<td>Conductance into drains.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>DRAI</td>
<td>Package</td>
<td>DIMENSION (5,MXDRN), For each drain: layer, row, column, head in the drain and conductance into the drain.</td>
</tr>
<tr>
<td>EL</td>
<td>Module</td>
<td>Elevation of the drain (head in the drain).</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
</tbody>
</table>
| IBD      | Package| Flag.  
|          |       | = 0, cell-by-cell flow terms for this package will not be recorded.  
|          |       | ≠ 0, cell-by-cell flow terms for this package will be recorded. |
| IBOUND   | Global| DIMENSION (NCOL,NROW,NLAY), Status of each cell.  
|          |       | < 0, constant-head cell  
|          |       | = 0, inactive cell  
|          |       | > 0, variable-head cell |
| IC       | Module| Index for columns. |
| ICBCFL   | Global| Flag.  
|          |       | = 0, cell-by-cell flow terms will not be recorded or printed for the current time step.  
|          |       | ≠ 0, cell-by-cell flow terms will be recorded for the current time step. |
| IDRNCB   | Package| Flag.  
<p>|          |       | &gt; 0 and if ICBCFL ≠ 0, cell-by-cell flow terms for the DRN1 Package will be recorded on UNIT = IDRNCB. |
| IL       | Module| Index for layers. |
| IOUT     | Global| Primary unit number for all printed output. IOUT = 6. |
| IR       | Module| Index for rows. |
| KPER     | Global| Stress period counter. |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for drains.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>MXDRN</td>
<td>Package</td>
<td>Maximum number of drains active at any one time.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NDRAIN</td>
<td>Package</td>
<td>Number of drains active during the current stress period.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>Q</td>
<td>Module</td>
<td>Flow from a drain into a cell. (Reverse the sign to get the flow into the drain.)</td>
</tr>
<tr>
<td>RATOUT</td>
<td>Module</td>
<td>Accumulator for the total flow out of the flow field into the drains.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For flow component N, the values in VBVL are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1,N) Rate for the current time step into the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(2,N) Rate for the current time step out of the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,N) Volume into the flow field during simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4,N) Volume out of the flow field during simulation.</td>
</tr>
</tbody>
</table>
The Evapotranspiration (ET) Package simulates the effects of plant transpiration and direct evaporation in removing water from the saturated ground water regime. The approach is based on the following assumptions: (1) when the water table is at or above a specified elevation, termed the "ET surface" in this report, evapotranspiration loss from the water table occurs at a maximum rate specified by the user; (2) when the depth of the water table below the ET surface elevation exceeds a specified interval, termed the "extinction depth" or "cutoff depth" in this report, evapotranspiration from the water table ceases; and (3) between these limits, evapotranspiration from the water table varies linearly with water table elevation.

This can be expressed in equation form as

\[ \begin{align*}
  RET_{i,j} &= RET_{Mi,j} \quad \text{if } h_{i,j,k} > h_{si,j} \\
  RET_{i,j} &= 0 \quad \text{if } h_{i,j,k} < h_{si,j} - d_{i,j} \\
  RET_{i,j} &= RET_{Mi,j} \left( \frac{h_{i,j,k} - (h_{si,j} - d_{i,j})}{d_{i,j}} \right) \quad \text{if } (h_{si,j} - d_{i,j}) \leq h_{i,j,k} \leq h_{si,j}
\end{align*} \]

where \( RET_{i,j} \) is the rate of loss per unit surface area of water table due to evapotranspiration, in volume of water per unit area per unit time, within the map area \( \Delta L_{j,DELC_i} \); \( h_{i,j,k} \) is the head, or water table elevation in the cell from which the evapotranspiration occurs; \( RET_{Mi,j} \) is the maximum possible value of \( RET_{i,j} \); \( h_{si,j} \) is the ET surface elevation, or the water table elevation at which this maximum value of evapotranspiration loss occurs; and \( d_{i,j} \) is the cutoff or extinction depth, such that when the water table elevation exceeds \( h_{si,j} - d_{i,j} \), evapotranspiration ceases.
distance between $h_{s_{i,j}}$ and $h_{i,j,k}$ exceeds $d_{i,j}$ evapotranspiration ceases.

In implementing the finite difference approach the volumetric rate of evapotranspiration loss from a given cell is required. This is given as the product of the loss rate per unit area, and the horizontal surface area, $DELR_jDEL_C_i$, of the cell from which the loss occurs, i.e.

$$Q_{ET_{i,j}} = RET_{i,j} * DELR_j * DELC_i$$

(74)

where $Q_{ET_{i,j}}$ is the evapotranspiration, in volume of water per unit time, through the area $DELR_jDEL_C_i$. If the maximum value of $Q_{ET_{i,j}}$ (corresponding to $RET_{i,j}$) is designated $Q_{ETM_{i,j}}$, equations (71)-(73) can be expressed in terms of volumetric discharge as

$$Q_{ET_{i,j}} = Q_{ETM_{i,j}}$$

$$Q_{ET_{i,j}} = 0$$

$$Q_{ET_{i,j}} = \frac{h_{i,j,k} - (h_{s_{i,j}} - d_{i,j})}{d_{i,j}}$$

(75)

(76)

(77)

Figure 42 shows a graph of evapotranspiration loss, $Q_{ET_{i,j}}$, vs head in cell $i,j,k$ based on equations (75)-(77). Comparison of the ET function with the river or drain functions shows that the three are mathematically similar, except that the linear portion of the ET function is bounded at both ends by constant values, rather than only at the lower end.

Evapotranspiration is drawn from only one cell in the vertical column beneath the map area $DELR_j*DELC_i$; the user designates the cell (i.e. the layer, $k$) using one of two options. Under the first option, evapotranspiration is always drawn from the uppermost layer of the model; under the second, the user specifies the cell, within the vertical column at $i,j$, from which the evapotranspiration is to be taken. In either case the computed evapotrans-
Figure 42.—Plot of volumetric evapotranspiration, \( Q_{ET} \), as a function of head, \( h \), in a cell where \( d \) is the cutoff depth and \( h_s \) is the ET surface elevation.
piration has no influence on the simulation if the designated cell is either
a no-flow cell or a constant head cell.

For each cell location, \((i, j)\), in the horizontal plane, and for each
stress period (unless an option is exercised to use prior values) the ET
package reads values of \(R_{ETM}\) (maximum evapotranspiration loss per unit area
per unit time) into an array labelled EVTR. These rates are immediately
multiplied by cell areas, \(DELR_j \times DELC_i\), to obtain the maximum volumetric
rate of evapotranspiration from each cell, \(Q_{ETM}\); these maximum volumetric
rates then replace the values of \(R_{ETM,i,j}\) in the EVTR array. Thus, the input
to the EVTR array consists of maximum evapotranspiration rates per unit
area, and as such must have dimensions \(Lt^{-1}\). In the calculation carried
out within the model, however, the entries in the EVTR array appear as
maximum volumetric rates, having dimensions \(L^3t^{-1}\).

Values of \(h_{si,j}\), the ET surface elevation (or water table elevation at
which evapotranspiration is maximum), are read into the two dimensional
array SURF by the ET package; values of the cutoff depth or extinction
depth are read into the two-dimensional array EXDP. Because the term \(Q_{ET,i,j}\)
of equations (75)-(77) has been defined as an outflow from the aquifer it
must be subtracted from the left side of equation (24). In terms of the
expressions HCOF and RHS of equation (26), this is accomplished in the ET
package as follows:

1) if \(h_{i,j,k} < (h_{si,i,j,k} - d_{i,j})\) no changes are made in the terms HCOF
or RHS for cell \(i,j,k\);
2) if \(h_{i,j,k} > h_{si,i,j}\), \(Q_{ET,i,j}\) is added to RHS\(_{i,j}\); and
3) if \((h_{si,i,j,k} - d_{i,j}) < h_{i,j,k} \leq h_{si,i,j,k}\) \(-Q_{ET,i,j}/d_{i,j}\) is added to

\[
HCOF_{i,j,k} \text{ and } RHS_{i,j,k} + \frac{(h_{si,i,j,k} - d_{i,j}) - Q_{ET,i,j}}{d_{i,j}}
\]

10-4
The value of $h_{s,i,j}$, the water table elevation at which evapotranspiration is maximum, should normally be taken as the average land surface elevation in the map area $\text{DELR}_j \text{DELC}_i$; the cutoff or extinction depth, $d_{i,j}$, is then frequently assumed to be on the order of six to eight feet below land surface (although considerable variation can be introduced by climatic factors, the presence of deep-rooted phreatophytes, or so on). Where the distance from land surface to the water table varies extensively within the area of a cell, care must be exercised in implementing the ET package and in choosing the various parameters of equation (70), or misleading results may be obtained.

The options for selection of the layer from which ET is to be drawn provide some flexibility in adapting the package to special situations, but also require some care in implementation. Figure 43 shows a situation similar to that discussed for the recharge package, in which a cross sectional model has been progressively truncated to follow the water table, using the provision for horizontal conductance formulation under water table conditions (Chapter 5). Figure 43-a shows the hydrologic situation under study, and figure 43-b the final distribution of variable head and inactive (no flow) cells obtained in the simulation.

Under option 1 (figure 43-c), evapotranspiration is drawn only from the uppermost layer of the model; in the problem shown, the presence of no flow cells in this layer deletes evapotranspiration from the right half of the model, so that the simulation fails to represent field conditions.

Figure 43-d shows the situation which could be achieved through the use of option 2, assuming that the simulation was carried out in stages and
Figure 43.—Hypothetical problem showing cells from which ET will be abstracted under the two options available in the ET Package.
that the user interacted with the simulation process, designating the cells from which evapotranspiration was to be drawn as the truncation of the mesh developed.
Evapotranspiration Package Input

Input to the Evapotranspiration (EVT) Package is read from the unit specified in IUNIT (5).

FOR EACH SIMULATION

EVTIAL

1. Data: NEVTOP IEVTCB
   Format: I10   I10

FOR EACH STRESS PERIOD

EVTIRP

2. Data: INSURF INEVTR INEXDP INIEVT
   Format: I10  I10  I10  I10

3. Data: SURF
   Module: U2DREL

4. Data: EVTR
   Module: U2DREL

5. Data: EXDP
   Module: U2DREL

IF THE ET OPTION IS EQUAL TO TWO

6. Data: IEVT
   Module: U2DINT

Explanation of Fields Used in Input Instructions

NEVTOP— is the evapotranspiration (ET) option code. ET parameters (ET surface, maximum ET rate, and extinction depth) are specified in two-dimensional arrays, SURF, EVTR, and EXDP, with one value for each vertical column. Accordingly, ET is calculated for one cell in each vertical column. The option codes determine for which cell in the column ET will be calculated.

1 - ET is calculated only for cells in the top grid layer.

2 - The cell for each vertical column is specified by the user in array IEVT.
IEVT--is the maximum ET rate (ETR) read flag.
If INEVTR > 0, an array containing the maximum ET rate will be read.
If INEVTR < 0, the maximum ET rate from the preceding stress period will be reused.

INEXDP--is the extinction depth (EXDP) read flag.
If INEXDP > 0, an array containing the extinction depth (EXDP) will be read.
If INEXDP < 0, the extinction depth from the preceding stress period will be reused.

INIEVT--is the layer indicator (IEVT) read flag. It is used only if the ET option (NEVTOP) is equal to two.
If INIEVT > 0, an array containing the layer indicators (IEVT) will be read.
If INIEVT < 0, layer indicators used during the preceding stress period will be reused.

SURF--is the elevation of the ET surface.

EVTR--is the maximum ET rate (volume of water per unit area (L\text{-1})).

EXDP--is the ET extinction depth.

IEVT--is the layer indicator array. For each horizontal location, it indicates the layer from which ET is removed. It is needed only if the ET option is equal to two.
SAMPLE INPUT TO THE EVAPOTRANSPIRATION PACKAGE USING ET OPTION 1

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(NEVTOP, IEVTCE)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Stress period 1---(INSURF, INEVTR, INEXD?, IN.EVT)</td>
<td>1 0 1</td>
</tr>
<tr>
<td>3</td>
<td>Control record for ET surface array</td>
<td>710 715 720 725 730 715 720 725 730 735 720 725 730 735 740 725 730 735 740 745 750 735 740 745 750 755</td>
</tr>
<tr>
<td>4</td>
<td>Control record for maximum ET rate</td>
<td>0 9.65E-7</td>
</tr>
<tr>
<td>5</td>
<td>Control record for extinction depth array</td>
<td>0 10</td>
</tr>
<tr>
<td>6</td>
<td>Stress period 2---(INSURF, INEVTR, INEXD?, IN.EVT)</td>
<td>-1 1 -1</td>
</tr>
<tr>
<td>7</td>
<td>Control record for maximum ET rate</td>
<td>0 8.23E-7</td>
</tr>
<tr>
<td>8</td>
<td>Control record for maximum ET rate</td>
<td>27 9.65E-7 (10F4.0)</td>
</tr>
</tbody>
</table>

ET surface

max ET rate

SAMPLE INPUT TO THE EVAPOTRANSPIRATION PACKAGE USING ET OPTION 2

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(NEVTOP, IEVTCE)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Stress period 1---(INSURF, INEVTR, INEXD?, IN.EVT)</td>
<td>2 45 1 1</td>
</tr>
<tr>
<td>3</td>
<td>Control record for ET surface array</td>
<td>710 715 720 725 730 715 720 725 730 735 720 725 730 735 740 725 730 735 740 745 730 735 740 745 750 735 740 745 750 755</td>
</tr>
<tr>
<td>4</td>
<td>Control record for maximum ET rate</td>
<td>0 9.65E-7</td>
</tr>
<tr>
<td>5</td>
<td>Control record for extinction depth array</td>
<td>0 10</td>
</tr>
<tr>
<td>6</td>
<td>Control record for layer indicator array</td>
<td>1 2 2 3 1 2 2 2 1 1 2 2 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>7</td>
<td>Stress period 2---(INSURF, INEVTR, INEXD?, IN.EVT)</td>
<td>-1 1 -1</td>
</tr>
<tr>
<td>8</td>
<td>Control record for maximum ET rate</td>
<td>0 8.23E-7</td>
</tr>
</tbody>
</table>

Layer numbers

FIELDS IN ARRAY CONTROL RECORDS ARE---{ LOCAT, CONST, FMTIN, IFRN }
The Evapotranspiration Package (EVT1) consists of four modules, all of which are called by the MAIN program. The modules are:

**EVTIAL** Allocates space for arrays to contain maximum ET rate (EVTR), surface elevation (SURF), extinction depth (EXDP), and, if option 2 is specified, the layer indicator (IEVT).

**EVT1RP** Reads arrays containing the maximum ET rate (in terms of a volume per unit area), surface elevation, extinction depth, and, if option 2 is specified, the layer indicator. Maximum ET rates are multiplied by cell area to get the maximum ET for each node as a volumetric rate.

**EVT1FM** Determines, for each horizontal location, which cell is at the surface. Determines if there is ET from that cell. If there is ET, add the appropriate terms to HCOF and RHS.

**EVT1BD** Calculates the rates and accumulated volume of ET out of the flow system.
This module allocates space in the X array to store data relating to evapotranspiration.

1. Print a message identifying the package.

2. Read and print the option indicator (NEVTOP) and the unit number for cell-by-cell flow terms (IEVTCB).

3. See if the ET option (NEVTOP) is legal. If NEVTOP is illegal (not 1 or 2), print a message saying the option is illegal. Do not allocate storage. STOP.

4. If NEVTOP is legal, print NEVTOP.

5. If the cell-by-cell flow terms are to be recorded, print the unit number (IEVTCB) where they will be recorded.

6. Allocate space for the maximum ET-rate array (EVTR), the extinction-depth array (EXDP), and the ET-surface array (SURF).

7. If the ET option (NEVTOP) is equal to two, allocate space for a layer-indicator array (IEVT).

8. Calculate and print the number of elements in the X array used by the ET package.

9. RETURN.
NEVTOP is the ET option.

If NEVTOP = 1, ET is from the top layer.

If NEVTOP = 2, ET is from the layer specified by the user in the indicator array (IEVT).

IEVTCB is the unit number on which cell-by-cell flow terms for ET will be written.

EVTR is an array which contains the maximum ET rate for each horizontal cell location.

SURF is an array which contains the elevation of the ET surface.

EXDP is an array which contains the extinction depth for ET.

IEVT is an array which contains the layer number from which ET is taken for each horizontal location. It is used only if option 2 has been specified.
SUBROUTINE EVTIAL(ISUM,LENX,LCEVT,LCEVTR,LCEXDP,LCSURF,  
   NCOL,NROW,NEVTOP,IN,IOUT,IEVTCB)
C
C-----VERSION 1607 12MAY1987 EVT1AL
C
C ****************************************************
C ALLOCATE ARRAY STORAGE FOR EVAPOTRANSPIRATION
C ****************************************************
C
C SPECIFICATIONS:
C
C********************************************************************************
C
C1------IDENTIFY PACKAGE.
C WRITE(IOUT,1)IN
  1   1 FORMAT(IHD,'EVT1 -- EVAPOTRANSPIRATION PACKAGE, VERSION 1,',  
          '9/1/87',' INPUT READ FROM UNIT!',I3)
C
C2------READ NEVTOP AND IEVTCB.
C      READ(IN,3)NEVTOP,IEVTCB
C      3 FORMAT(2110)
C
C3------CHECK TO SEE THAT ET OPTION IS LEGAL.
C      WHILE(IOUT,B)
C      8 FORMAT(IHX,'ILLEGAL ET OPTION CODE. SIMULATION ABORTING')
C      STOP
C
C4------IF THE OPTION IS LEGAL THEN PRINT THE OPTION CODE.
C      200 IF(NEVTOP.GE.1.AND.NEVTOP.LE.P)GO TO 200
C      201 FORMAT(IHX,'OPTION 1 -- EVAPOTRANSPIRATION FROM TOP LAYER')
C      202 FORMAT(IHX,'OPTION 2 -- EVAPOTRANSPIRATION FROM ONE SPECIFIED',  
                'NODE IN EACH VERTICAL COLUMN')
C      IRK=ISUM
C
C5------IF CELL-BY-CELL TERMS TO BE SAVED THEN PRINT UNIT NUMBER.
C      IF(IEVTCB.GT.0) WRITE(IOUT,203) IEVTCB
C      203 FORMAT(IHX,'CELL-BY-CELL FLOW TERMS WILL BE SAVED ON UNIT!',I3)
C
C6------ALLOCATE SPACE FOR THE ARRAYS EVTR, EXDP AND SURF.
C      LCEVTR=ISUM
C      ISUM=ISUM+NCOL*NROW
C      LCEXDP=ISUM
C      ISUM=ISUM+NCOL*NROW
C      LCSURF=ISUM
C      ISUM=ISUM+NCOL*NROW
C
C7------IF OPTION 2 THEN ALLOCATE SPACE FOR THE INDICATOR ARRAY(IEVT)
C      LCIERYT=ISUM
C      IF(NEVTOP.NE.2)GO TO 300
C      ISUM=ISUM+NCOL*NROW
C
C8------CALCULATE & PRINT AMOUNT OF SPACE USED BY ET PACKAGE.
C      300 IRK=ISUM-IRK
C      WRITE(IOUT,4)IRK
C      4 FORMAT(IHX,' ELEMENTS OF X ARRAY USED FOR EVAPOTRANSPIRATION')
C      ISUM1=ISUM-1
C      WRITE(IOUT,5)ISUM1,LENX
C      5 FORMAT(IHX,' ELEMENTS OF X ARRAY USED OUT OF',181  
                ' ***X ARRAY MUST BE MADE LARGER***')
C
C9------RETURN.
C      RETURN
C      END

10-14
List of Variables for Module EVTIAL

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEVTCB</td>
<td>Package</td>
<td>Flag. If IEVTCB &gt; 0 and ICBCFL \neq 0, cell-by-cell flow terms for the EVT1 Package will be recorded on UNIT = IEVTCB.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Module</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IRK</td>
<td>Module</td>
<td>Before this module allocates space, IRK is set equal to ISUM. After allocation, IRK is subtracted from ISUM to get the amount of space in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.</td>
</tr>
<tr>
<td>ISUM1</td>
<td>Module</td>
<td>Index number of the last element of the X array allocated by this module.</td>
</tr>
<tr>
<td>LCEVTR</td>
<td>Package</td>
<td>Location in the X array of the first element of array EVTR.</td>
</tr>
<tr>
<td>LCEXDP</td>
<td>Package</td>
<td>Location in the X array of the first element of array EXDP.</td>
</tr>
<tr>
<td>LCIEVT</td>
<td>Package</td>
<td>Location in the X array of the first element of array IEVT.</td>
</tr>
<tr>
<td>LCSURF</td>
<td>Package</td>
<td>Location in the X array of the first element of array SURF.</td>
</tr>
<tr>
<td>LENV</td>
<td>Global</td>
<td>Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
</tbody>
</table>
| NEVTOP   | Package | ET option: 
|          |       | = 1, ET is from the top layer.  
|          |       | = 2, ET at each horizontal-cell location is from the layer specified by the user in the layer-indicator array (IEVT). |
| NROW     | Global  | Number of rows in the grid. |
This module reads data used to calculate the terms which represent evapotranspiration.

1. Read the values INSURF, INEXDP, INEVTR, and INIEVT which indicate whether the data contained in arrays SURF, EXDP, EVTR, and IEVT, respectively, used during the last stress period, are to be used for the current stress period.

2. Test INSURF to see where the ET-surface array (SURF) is coming from. If INSURF is less than zero, the ET-surface elevation used in the last stress period will be used again in this stress period. Print a message to that effect and GO TO 4.

3. INSURF is greater than or equal to zero. CALL U2DREL to read SURF.

4. Test INEVTR to see where the maximum ET rate (EVTR) is coming from. If INEVTR is less than zero, the maximum ET rate used in the last stress period will be used again in this stress period. Print a message to that effect and GO TO 7.

5. INEVTR is greater than or equal to zero. CALL U2DREL to read the maximum ET rate (EVTR).

6. Multiply the maximum ET rate by the area to get a volumetric rate.

7. Test INEXDP to see where the extinction rate is coming from. If INEXDP is less than zero, the extinction depth used in the last stress period will be used again in this stress period. Print a message to that effect and GO TO 9.

8. If INEXDP is greater than or equal to zero, CALL U2DREL to read the extinction depth.

9. If the ET option (NEVTOP) is equal to two, a layer-indicator array is needed.

10. Test INIEVT to see where the layer indicator is coming from. If INIEVT is less than zero, the indicator array used in the last stress period will be used again in this stress period. Print a message to that effect and GO TO 12.

11. If INIEVT is greater than or equal to zero, CALL U2DINT to read the IEVT array.

12. RETURN.
INEVTR is a flag which, when set, indicates that the maximum ET rate EVTR should be read for the current stress period. If it is clear (less than zero), maximum ET rates from the last stress period should be reused.

INIEVT, INSURF, and INEXDP are flags similar to INEVTR used for the layer indicator array (IEVT), the ET surface array (SURF), and the extinction depth array (EXDP), respectively.

EVTR is an array containing the maximum ET rate for every horizontal cell location.

SURF is an array containing the ET surface elevation for each horizontal cell location.

EXDP is an array containing the extinction depth for each horizontal cell location.

IEVT is an array containing a layer indicator for each horizontal cell location. For each horizontal cell location, it indicates the layer number of the cell at that location from which ET is taken. It is used only if the ET option (NEVTOP) is equal to two.

NEVTOP is the ET option.

If NEVTOP = 1, ET is from the top layer.

If NEVTOP = 2, ET is from the layer specified by the user in the indicator array (IEVT).
SUBROUTINE EVTlRP(NEVTCP,IEVT,EVTR,EXDP,NSURF,DELR,DELC,
   1 NCOL,NROW,IOUT)

--VERSION 1635 24JUL1987 EVTlM=
READ EVAPOTRANSPIRATION DATA

SPECIFICATIONS:

CHARACTER*4 ANAME
DIMENSION IEVT(NCOL,NROW),EVTR(NCOL,NROW),EXDP(NCOL,NROW),
   1 SURF(NCOL,NROW),NAME(6,4),DELR(NCOL),DELC(NROW)

DATA NAME(1,1),NAME(1,2),NAME(1,3),NAME(1,4),NAME(1,5),
1 NAME(1,6) /'ET','LAI','E','LAI','E','LAI'/'
DATA NAME(2,1),NAME(2,2),NAME(2,3),NAME(2,4),NAME(2,5),
1 NAME(2,6) /'ET','LAI','E','LAI','E','LAI'/'
DATA NAME(3,1),NAME(3,2),NAME(3,3),NAME(3,4),NAME(3,5),
1 NAME(3,6) /'ET','LAI','E','LAI','E','LAI'/'
DATA NAME(4,1),NAME(4,2),NAME(4,3),NAME(4,4),NAME(4,5),
1 NAME(4,6) /'ET','LAI','E','LAI','E','LAI'/'

READ(16)INSURF,INEVTR,INEXDP,INIEVT
6 FORMAT(410)

C2-------READ FLAGS SHOWING WHETHER DATA IS TO BE REUSED.
   32 IF(INSURF.GE.0)GO TO 32
C
C4-------READ EVAPOTRANSPIRATION DATA
   35 IF(INEXDP.GE.0)GO TO 37
C
C6-------READ INDICATOR ARRAY
   49 IF(NEVTCP.NE.1)GO TO 50
C
C12------RETURN
   50 RETURN
END
### List of Variables for Module EVT1RP

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANAME</td>
<td>Module</td>
<td>Label for printout of the input array.</td>
</tr>
<tr>
<td>DELC</td>
<td>Global</td>
<td>DIMENSION (NROW), Cell dimension in the column direction. DELC(I) contains the width of row I.</td>
</tr>
<tr>
<td>DELR</td>
<td>Global</td>
<td>DIMENSION (NCOL), Cell dimension in the row direction. DELR(J) contains the width of column J.</td>
</tr>
<tr>
<td>EVTR</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Maximum ET rate.</td>
</tr>
<tr>
<td>EXDP</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Extinction depth.</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>IEVT</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Layer number for each horizontal cell location from which ET will be taken if the ET option (NEVTOP) is equal to two.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>INEVTR</td>
<td>Module</td>
<td>Flag. &lt; 0, EVTR array already in memory from the last stress period will be used.</td>
</tr>
<tr>
<td>INEXDP</td>
<td>Module</td>
<td>Flag. &lt; 0, EXDP array already in memory from the last stress period will be used.</td>
</tr>
<tr>
<td>INIEVT</td>
<td>Module</td>
<td>Flag. &lt; 0, IEVT array already in memory from the last stress period will be used.</td>
</tr>
<tr>
<td>INSURF</td>
<td>Module</td>
<td>Flag. &lt; 0, SURF array already in memory from the last stress period will be used.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
</tbody>
</table>
| NEVTOP     | Package   | ET option. 
  = 1, ET is from the top layer. 
  = 2, ET at each horizontal-cell location is from the layer specified in the layer-indicator array (IEVT). |
| NROW       | Global    | Number of rows in the grid.                                               |
| SURF       | Package   | DIMENSION (NCOL,NROW), Elevation of the ET surface.                       |
Narrative for Module EVT1FM

This module adds terms representing ET to the finite-difference equations.

1. For each horizontal-cell location, determine which layer ET comes from and add the appropriate terms to the equation for the cell. DO STEPS 1-7.

2. Set the layer index equal to one.

3. If option 2 was invoked, get the layer index from the indicator array (IEVT).

4. If the cell is external, move on to the next horizontal-cell location. SKIP STEPS 5-7.

5. If the head in the aquifer is greater than or equal to the ET-surface elevation, add EVTR to RHS and move on to the next horizontal-cell location. SKIP STEPS 6 AND 7.

6. If the head in the aquifer is less than the extinction elevation (ET surface minus extinction depth), no terms need to be added to the finite-difference equation. Move on to the next horizontal-cell location. SKIP STEP 7.

7. Add the term -EVTR/EXDP to HCOF and subtract the term -EVTR( EXDP - SURF)/EXDP from RHS.

8. RETURN.
IEVT is an array containing a layer indicator for each horizontal cell location. For each horizontal cell location, it indicates the layer number of the cell at that location from which ET is taken. It is used only if the ET option (NEVTOP) is equal to two.

SURF is an array containing the maximum ET rate for every horizontal cell location.

EVTR is an array containing the maximum ET rate for every horizontal cell location.

RHS is an accumulator in which the right hand side of the finite-difference equation is formulated.

HCOF is an accumulator in which a coefficient of head in the finite-difference equation is formulated.

NEVTOP is the ET option.

If NEVTOP = 1, ET is from the top layer.

If NEVTOP = 2, ET is from the layer specified by the user in the indicator array (IEVT).
SUBROUTINE EVTlFM(NEVTOP, IEVT, EVTR, EXDP, SURF, RHS, HCOF, 
1     IBOUND, HNEW, NCOL, NROW, NLAY)
C
C-----VERSION 1031 10APR1985 EVTlFM
C
C**********************~**************************************~****
C
C ADD EVAPOTRANSPIRATION TO RHS AND HCOF
C
C *************************************************************~****
C
C SPECIFICATIONS:
C
DOUBLE PRECISION HNEW
DIMENSION IEVT(NCOL, NROW), EVTR(NCOL, NROW), EXDP(NCOL, NROW),
1       SURF(NCOL, NROW), RHS(NCOL, NROW, NLAY),
2       HCOF(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
3       HNEW(NCOL, NROW, NLAY)

C1-------PROCESS EACH HORIZONTAL CELL LOCATION
DO 10 IR=1,NROW
DO 10 IC=1,NCOL
C
C2-------SET THE LAYER INDEX EQUAL TO 1
IL=1
C
C3-------IF OPTION 2 IS SPECIFIED THEN GET LAYER INDEX FROM IEVT ARRAY
IF(NEVTOP.EQ.2)IL=IEVT(IC,IR)
C
C4-------IF THE CELL IS EXTERNAL IGNORE IT.
IF(IBOUND(IC,IR,IL).LE.0)GO TO 10
C=EVTR(IC,IR)
S=SURF(IC,IR)
H=HNEW(IC,IR,IL)
C
C5-------IF AQUIFER HEAD IS GREATER THAN OR EQUAL TO SURF, ET IS CONSTANT
IF(H.LT.S) GO TO 5
C
C5A------SUBTRACT -EVTR FROM RHS
RHS(IC,IR,IL)=RHS(IC,IR,IL)+C
GO TO 10
C
C6-------IF DEPTH TO WATER>=EXTINCTION DEPTH THEN ET IS 0
5 D=S-H
X=EXDP(IC,IR)
IF(D.GE.X)GO TO 10
C
C7-------LINEAR RANGE. ADD ET TERMS TO BOTH RHS AND HCOF.
RHS(IC,IR,IL)=RHS(IC,IR,IL)+C-C*S/X
HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-C/X
10 CONTINUE
C
C8-------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Module</td>
<td>Maximum ET rate.</td>
</tr>
<tr>
<td>D</td>
<td>Module</td>
<td>Depth to water.</td>
</tr>
<tr>
<td>EVTR</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Maximum ET rate.</td>
</tr>
<tr>
<td>EXDP</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Extinction depth.</td>
</tr>
<tr>
<td>H</td>
<td>Module</td>
<td>Head in the cell.</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K) in the finite-difference equation.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell. &lt; 0, constant-head cell; = 0, inactive cell; &gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>IEVT</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Layer number, for each horizontal-cell location, from which ET will be taken if the ET option (NEVTOP) is equal to two.</td>
</tr>
<tr>
<td>IL</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NEVTOP</td>
<td>Package</td>
<td>ET option. = 1, ET is from the top layer; = 2, ET at each horizontal cell location is from the layer specified in the layer-indicator array (IEVT).</td>
</tr>
<tr>
<td>NHLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>RHS</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Right hand side of finite-difference equation. RHS is an accumulation of terms from several different packages.</td>
</tr>
<tr>
<td>S</td>
<td>Module</td>
<td>ET surface elevation for a cell.</td>
</tr>
<tr>
<td>SURF</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Elevation of the ET surface.</td>
</tr>
<tr>
<td>X</td>
<td>Module</td>
<td>Extinction depth for a cell.</td>
</tr>
</tbody>
</table>
This module calculates rates and volumes removed from the aquifer by evapotranspiration.

1. Clear the rate accumulator RATOUT.

2. If budget terms will be saved, clear the buffer (BUFF) in which they will be accumulated.

3. Process each horizontal-cell location one at a time calculating flow to evapotranspiration (STEPS 4-11).

4. Set the layer index (IL) equal to one.

5. If option 2 is in effect, get the layer index from the layer-indicator array (IEVT).

6. If the cell is external (IBOUND < 0), bypass processing of the cell. SKIP STEP 9.

7. If the head in the aquifer is greater than the elevation of the ET surface, set the ET rate for the cell equal to the maximum ET rate. SKIP STEPS 8 AND 9.

8. If the depth to the water is greater than the extinction depth, bypass further processing of this cell. SKIP STEP 9.

9. Calculate the ET flow into the model using the linear approximation.

10. Subtract the ET flow from the accumulator (RATOUT).

11. If the cell-by-cell flow terms are to be saved, add the ET rate to the buffer (BUFF).

12. If the cell-by-cell flow terms are to be saved, call module UBUDSV to write the buffer (BUFF) onto a disk.

13. Move RATOUT into the VBVL array for printing by BASIOT.

14. Add RATOUT multiplied by the time-step length to the volume accumulators in VBVL for printing by BASIOT.

15. Move the ET budget-term labels to VBNM for printing by BASIOT.

16. Increment the budget-term counter (MSUM).

17. RETURN.
RATOUT is an accumulator to which all flows out of the aquifer are added.

BUFFER is an array in which values are stored as they are being gathered for printing or recording.

IEVT is an array containing a layer indicator for each horizontal cell location. For each horizontal cell location, it indicates the layer number of the cell at that location from which ET is taken. It is used only if NEVTOP is equal to two.

SURF is an array containing the ET surface elevation for each horizontal cell location.

Q is the flow to ET from an individual cell.

VBVL is a table of budget entries calculated by component-of-flow packages for use in calculating the volumetric budget.

VBNM is a table of labels for budget terms.
SUBROUTINE EVTlBD(NEVTOP, IEVT, EVTR, EXDP, SURF, IBOUND, HNEW,
1 NCOL, NROW, NLAY, DELT, VBVL, VBNM, MSUM, KSTP, KPER,
2 IEVTCB, IBBCFL, BUFF, IOUT)

C-----VERSION 1608 12MAY1987 EVTlBD
C
C CALCULATE VOLUMETRIC BUDGET FOR EVAPOTRANSPIRATION
C
C SPECIFICATIONS:
C
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW
DIMENSION IEVT(NCOL, NROW), EVTR(NCOL, NROW), EXDP(NCOL, NROW),
1 SURF(NCOL, NROW), IBOUND(NCOL, NROW, NLAY),
2 VBVL(4,20), VBNM(4,20), HNEW(NCOL, NROW, NLAY),
3 BUFF(NCOL, NROW, NLAY)
DIMENSION TEXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /' ',' ',' ',' ET'/

C1------CLEAR THE RATE ACCUMULATOR.
RATOUT=0
C
C2------IF CELL-BY-CELL FLOW TERMS WILL BE SAVED THEN CLEAR THE BUFFER.
IBD=0
IF(IEVTCB.LE.0 .OR. IBBCFL.EQ.0) GO TO 5
IBD=1
DO 4 IL=1, NLAY
DO 4 IR=1, NROW
DO 4 IC=1, NCOL
BUFF(IC, IR, IL)=0.
4 CONTINUE
C
C3------PROCESS EACH HORIZONTAL CELL LOCATION
5 DO 10 IR=1, NROW
   DO 10 IC=1, NCOL
C
C4------SET THE LAYER INDEX EQUAL TO 1
IL=1
C
C5------IF OPTION 2 IS SPECIFIED THEN GET LAYER INDEX FROM IEVT ARRAY
   IF(NEVTOP.EQ.2) IL=IEVT(IC, IR)
C
C6------IF CELL IS EXTERNAL THEN IGNORE IT.
   IF(IBOUND(IC, IR, IL).LE.0) GO TO 10
   C=EVTR(IC, IR)
   S=SURF(IC, IR)

10-26
H=HNEW(IC,IR,IL)

C7------IF AQUIFER HEAD => SURF, SET Q=MAX ET RATE
   IF(H.LT.S) GO TO 7
   Q=Q
   GO TO 9

C8------IF DEPTH=>EXTINCTION DEPTH, ET IS 0
   7 X=EXDP(IC,IR)
      D=H
      IF(D.GE.X)GO TO 10

C9------LINEAR RANGE . Q=-EVTR(H-EXEL)/EXDP
      Q=C*D/X-C

C10------ACCUMULATE TOTAL FLOW RATE
    9 RATOUT=RATOUT-Q

C11------IF CELL-BY-CELL FLOW TERMS TO BE SAVED THE ADD Q TO BUFFER.
      IF(IBD.EQ.1) BUFF(IC,IR,IL)=Q
      10 CONTINUE

C12------IF C-B-C TO BE SAVED CALL MODULE UBUDSY TO RECORD THEM.
      IF(IBD.EQ.1) CALL UBUDSY(KSTP,KPER,TEXT,IEVTCB,BUFF,NCOL,NROW,
      1
      NLAY,IOUT)

C13------MOVE TOTAL ET RATE INTO VBVL FOR PRINTING BY BAS1OT.
      VBVL(3,MSUM)=0.
      VBVL(4,MSUM)=RATOUT

C14------ADD ET(ET_RATE TIMES STEP LENGTH) TO VBVL
      VBVL(1,MSUM)=0.
      VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT

C15------MOVE BUDGET TERM LABELS TO VBNM FOR PRINT BY MODULE BAS1OT
      VBNM(1,MSUM)=TEXT(1)
      VBNM(2,MSUM)=TEXT(2)
      VBNM(3,MSUM)=TEXT(3)
      VBNM(4,MSUM)=TEXT(4)

C16------INCREMENT BUDGET TERM COUNTER
      MSUM=MSUM+1

C17------RETURN
      RETURN
      END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>C</td>
<td>Module</td>
<td>Maximum ET rate at a cell.</td>
</tr>
<tr>
<td>D</td>
<td>Module</td>
<td>Depth to water below the ET surface.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>EVTR</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Maximum ET rate.</td>
</tr>
<tr>
<td>EXDP</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Extinction depth.</td>
</tr>
<tr>
<td>H</td>
<td>Module</td>
<td>Head in the cell.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
</tbody>
</table>
| IBD     | Module| Flag.  
|         |       | = 0, cell-by-cell flow terms for this package will not be recorded.  
|         |       | ≠ 0, cell-by-cell flow terms for this package will be recorded. |
| IBOUND  | Global| DIMENSION (NCOL,NROW,NLAY), Status of each cell.  
|         |       | < 0, constant-head cell  
|         |       | = 0, inactive cell  
|         |       | > 0, variable-head cell |
| IC      | Module| Index for columns. |
| ICBCFL  | Global| Flag.  
|         |       | = 0, cell-by-cell flow terms will not be recorded or printed for the current time step.  
|         |       | ≠ 0, cell-by-cell flow terms will be recorded for the current time step. |
| IEVT    | Package| DIMENSION (NCOL,NROW), Layer number for each horizontal-cell location from which ET will be taken if the ET option (NEVTOP) is equal to two. |
| IEVTCB  | Package| Flag.  
|         |       | If IEVTCB > 0 and ICBCFL ≠ 0, cell-by-cell flow terms for the EVT1 Package will be recorded on UNIT = IEVTCB. |
| IL      | Module| Index for layers. |
| IOUT    | Global| Primary unit number for all printed output. IOUT = 6. |
| IR      | Module| Index for rows. |
| KPER    | Global| Stress period counter. |
| KSTP    | Global| Time step counter. Reset at the start of each stress period. |
| MSUM    | Global| Counter for budget entries and labels in VBVL and VBNM. |
| NCOL    | Global| Number of columns in the grid. |
| NEVTOP  | Package| ET option.  
|         |       | = 1, ET is from the top layer.  
|         |       | = 2, ET at each horizontal-cell location is from the layer specified in the layer-indicator array (IEVT). |
| NROW    | Global| Number of rows in the grid. |
| Q       | Module| Flow from ET into the cell. (Reverse the sign to get the flow to ET.) |
### List of Variables for Module EVT1BD (Continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RATOUT</td>
<td>Module</td>
<td>Accumulator for the total flow out of the flow field to ET.</td>
</tr>
<tr>
<td>S</td>
<td>Module</td>
<td>Elevation of the ET surface for a cell.</td>
</tr>
<tr>
<td>SURF</td>
<td>Package</td>
<td>DIMENSION (NCOL,NROW), Elevation of the ET surface.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget. For flow component N, the values in VBVL are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1,N), Rate for the current time step into the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(2,N), Rate for the current time step out of the flow field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,N), Volume into the flow field during simulation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4,N), Volume out of the flow field during simulation.</td>
</tr>
<tr>
<td>X</td>
<td>Module</td>
<td>Extinction depth for a cell.</td>
</tr>
</tbody>
</table>

10-29
CHAPTER 11

GENERAL-HEAD BOUNDARY PACKAGE

Conceptualization and Implementation

The function of the General-Head Boundary (GHB) Package is mathematically similar to that of the River, Drain and ET Packages, in that flow into or out of a cell \(i,j,k\), from an external source is provided in proportion to the difference between the head in the cell, \(h_{i,j,k}\), and the head assigned to the external source, \(h_{bi,j,k}\). Thus a linear relationship between flow into the cell and head in the cell is established, i.e.

\[
Q_{bi,j,k} = C_{bi,j,k} (h_{bi,j,k} - h_{i,j,k})
\]

where \(Q_{bi,j,k}\) is the flow into cell \(i,j,k\) from the source; \(C_{bi,j,k}\) is the conductance between the external source and cell \(i,j,k\); \(h_{bi,j,k}\) is the head assigned to the external source; and \(h_{i,j,k}\) is the head in cell \(i,j,k\).

The relationship between cell \(i,j,k\) and the external source is shown schematically in figure 44. The constant-head source is represented by the apparatus on the right in figure 44, which holds the source head at the level \(h_b\) regardless of other factors; the link between the source and cell \(i,j,k\) is represented by the block of porous material \(C_{bi,j,k}\). Note that figure 44 shows no mechanism to limit flow in either direction as \(h_{i,j,k}\) rises or falls.

A graph of \(Q_{bi,j,k}\) versus \(h_{i,j,k}\) as given by equation (78) is shown in figure 45. In contrast to the River, Drain and ET Packages, the GHB Package provides no limiting value of flow to bound the linear function in either direction; and as the head difference between cell \(i,j,k\) and the source increases, flow into or out of the cell continues to increase without
Figure 44.—Schematic diagram illustrating principle of general-head boundary package.
Figure 45.—Plot of flow, \( Q_b \), from a general-head boundary source into a cell as a function of head, \( h \), in the cell where \( h_b \) is the source head.
Because $Q_{bi,j,k}$ of equation (78) is defined as an inflow to the aquifer it must be added to the left side of equation (24). In terms of the expressions $HCOF$ and $RHS$ of equation (26), this is accomplished in the model by subtracting the term $C_{bi,j,k}$ from $HCOF_{i,j,k}$ and subtracting the term $C_{bi,j,k}h_{bi,j,k}$ from $RHS_{i,j,k}$ as the matrix equations are assembled.
General-Head Boundary Package Input

Input for the General-Head Boundary (GHB) Package is read from the unit specified in IUNIT(7).

FOR EACH SIMULATION

GHBIAL

1. Data: MXBND  IGHBCB
   Format: I10  I10

FOR EACH STRESS PERIOD

GHB1RP

2. Data: ITMP
   Format: I10

3. Data: Layer  Row  Column  Boundary
   Format: I10  I10  I10  F10.0  F10.0

   (Input item 3 normally consists of one record for each GHB. If ITMP is negative or zero, item 3 is not read.)

Explanation of Fields Used in Input Instructions

**MXBND**--is the maximum number of general-head boundary cells at one time.

**IGHBCB**--is a flag and a unit number.

   If IGHBCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL (see Output Control) is set.

   If IGHBCB = 0, cell-by-cell flow terms will not be printed or recorded.

   If IGHBCB < 0, boundary leakage for each cell will be printed whenever ICBCFL is set.

**ITMP**--is a flag and a counter.

   If ITMP < 0, GHB data from the preceding stress period will be reused.

   If ITMP > 0, ITMP is the number of general-head boundaries during the current stress period.
Layer--is the layer number of the cell affected by the head-dependent boundary.

Row--is the row number of the cell affected by the head-dependent boundary.

Column--is the column number of the cell affected by the head-dependent boundary.

Boundary head--is the head on the boundary.

Cond--is the hydraulic conductance of the interface between the aquifer cell and the boundary.
### Sample Input to the General Head Boundary Package

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{MXRN, ICUBCB}</td>
<td>6 24</td>
</tr>
<tr>
<td>2</td>
<td>STRESS PERIOD 1 {ITMP}</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>FIRST BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 5 6 235.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>SECOND BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 4 6 230.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>THIRD BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 5 8 250.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>FOURTH BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 7 6 235.0 .012</td>
</tr>
<tr>
<td>2</td>
<td>STRESS PERIOD 2 {ITMP}</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>STRESS PERIOD 3 {ITMP}</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>STRESS PERIOD 4 {ITMP}</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>FIRST BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 5 6 235.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>SECOND BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 4 6 230.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>THIRD BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 5 8 250.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>FOURTH BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 7 6 235.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>FIFTH BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 9 6 235.0 .012</td>
</tr>
<tr>
<td>3</td>
<td>SIXTH BOUNDARY [Layer, Row, Column, Head, Conductance]</td>
<td>2 10 6 250.0 .012</td>
</tr>
</tbody>
</table>
The General-Head Boundary Package (GHBl) consists of four modules, all of which are called by the MAIN program. The modules are:

**GHBlAL**
- Allocates space for an array that contains the general-head boundary list (BNDS).

**GHBlRP**
- Reads location, boundary head, and boundary conductance ($C_m$) of each cell containing general-head boundary $m$.

**GHBlFM**
- Adds the terms $-C_m$ and $-C_mHB_m$ to the accumulators $HCOF_{i,j,k}$ and $RHS_{i,j,k}$, respectively.

**GHBlBD**
- Calculates the rates and accumulated volume of flow to and from general-head boundaries.
This module allocates space in the X array to store the list of general-head boundaries (GHB).

1. Print a message identifying the package and initialize NBOUND (number of general-head boundaries).

2. Read and print MXBND (the maximum number of general-head boundaries) and IGHBCB (the unit number for saving cell-by-cell flow terms or a flag indicating that cell-by-cell flow terms should be printed).

3. Set LCBNDS, which will point to the first element in the boundary list (BNDS), equal to ISUM which is currently pointing to the first unallocated element in the X array.

4. Calculate the amount of space needed for the boundary list (five values for each boundary--row, column, layer, head, and conductance) and add it to ISUM so that it continues to point to the first unallocated element in X.

5. Print the number of elements in the X array used by the GHB Package.

6. RETURN.
Flow Chart for Module GHIAL

NBOUND is the number of general-head boundaries being simulated at any given time.

MXBND is the maximum number of general-head boundaries simulated.

IGHBCB is a flag and a unit number.

   If IGHBCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.

   If IGHBCB = 0, cell-by-cell flow terms will not be printed or recorded.

   If IGHBCB < 0, the boundary leakage for each cell will be printed whenever ICBCFL is set.

LCBNDS is the location in the X array of the list of general-head boundaries data (BNDS).

BNDS is a table containing data for general-head boundaries.
SUBROUTINE GHBlAL(ISUM, LENX, LCBNDS, NBOUND, MXBND, IN, IOUT, 
1 IGHBCB)
C
C-----VERSION 1610 12MAY1987 GHBlAL
C ****************************~*************************************
C ****************************~*************************************
C
C SPECIFICATIONS:
C
C ALLOCATE ARRAY STORAGE FOR HEAD-DEPENDENT BOUNDARIES
C ****************************~*************************************
C
C IDENTIFY PACKAGE AND INITIALIZE # OF GENERAL HEAD BOUNDS
WRITE(IOUT,1)IN
1 FORMAT(1HO,'GHBl -- GH Bl PACKAGE, VERSION 1, 9/1/87', 
1' INPUT READ FROM UNIT',I3)
NBOUND=0
C
C READ AND PRINT MXBND AND IGHBCB (MAX # OF BOUNDS AND UNIT
C FOR CELL-BY-CELL FLOW TERMS FOR GHBl)
READ(IN,2) MXBND, IGHBCB
2 FORMAT(21I0)
WRITE(IOUT,3) MXBND
3 FORMAT(1H , 'MAXIMUM OF',I5,' HEAD-DEPENDENT BOUNDARY NODES')
IF(IGHBCB.GT.0) WRITE(IOUT,9) IGHBCB
9 FORMAT(1X,'CELL-BY-CELL FLOW WILL BE RECORDED ON UNIT',I3)
IF(IGHBCB.LT.0) WRITE(IOUT,8)
8 FORMAT(1X,'CELL-BY-CELL FLOW WILL BE PRINTED WHEN ICBCFL NOT 0')
C
C SET LCBNDS EQUAL TO ADDRESS OF FIRST UNUSED SPACE IN X.
LCBNDS=ISUM
C
C CALCULATE AMOUNT OF SPACE USED BY THE GENERAL HEAD LIST.
ISP=5*MXBND
ISUM=ISUM+ISP
C
C PRINT AMOUNT OF SPACE USED BY THE GH Bl PACKAGE
WRITE(IOUT,4) ISP
4 FORMAT(1X,'I8,' ELEMENTS IN X ARRAY ARE USED FOR HEAD', 
1 'DEPENDENT BOUNDARIES')
ISUM1=ISUM-1
WRITE(IOUT,5) ISUM1, LENX
5 FORMAT(1X,'I8,' ELEMENTS OF X ARRAY USED OUT OF',I8)
IF(ISUM1.GT.LENX) WRITE(IOUT,6)
6 FORMAT(1X,' ***X ARRAY MUST BE DIMENSIONED LARGER***)
C
RETURN
RETURN
END
## List of Variables for Module GHBIAL

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
</table>
| IGHRCR   | Package | Flag and a unit number.  
|          |         | > 0, unit number on which the cell-by-cell flow terms will be recorded whenever ICBCFL is set.  
|          |         | = 0, cell-by-cell flow terms will not be printed or recorded.  
|          |         | < 0, boundary leakage for each cell will be printed whenever IGHBFL is set. |
| IOUT     | Global  | Primary unit number from which input for this package will be read.  
| ISP      | Module  | Primary unit number for all printed output. IOUT = 6.  
| ISUM     | Global  | Number of words in the X array allocated by this module.  
|          |         | Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.  
| ISUM1    | Module  | ISUM-1.  
| LCBNDS   | Package | Location in the X array of the first element of array BNDS.  
| LENX     | Global  | Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.  
| MXBND    | Package | Maximum number of head boundaries active at any one time.  
| NBOUND   | Package | Number of head boundaries active during the current stress period.  

11-12
Narrative for Module GHBlRP

This module reads data to build the general-head boundary list.

1. Read ITMP. ITMP is the number of general-head boundaries or a flag indicating that data from the previous stress period should be reused.

2. Test ITMP. If ITMP is less than zero, the general-head boundary data read for the last stress period will be reused. Print a message to that effect and RETURN.

3. If ITMP is greater than or equal to zero, it is the number of general-head boundaries for this stress period. Set the number of general-head boundaries (NBOUND) in the current stress period equal to ITMP.

4. Compare the number of general-head boundaries (NBOUND) in the current stress period to the number specified as the maximum for the simulation (MXBND). If NBOUND is greater than MXBND, STOP.

5. Print the number of general-head boundaries in the current stress period (NBOUND).

6. See if there are any general-head boundaries. If there are none in the current stress period (NBOUND = 0), bypass further boundary processing (SKIP STEP 7).

7. Read and print the layer, row, column, head, and conductance for each general-head boundary.

8. RETURN.
ITMP is both a flag and a counter. If it is greater than or equal to zero, it is the number of general-head boundaries to be simulated during the stress period. If it is less than zero, it indicates that the boundaries simulated in the last stress period should be simulated in the current stress period.

MXBND is the maximum number of general-head boundaries to be simulated.
SUBROUTINE GHB1RP(BNDS,NBOUND,MXBND,IN,IOUT)

C-----VERSION 1651 02FEB1983 GHB1RP
C
READ DATA FOR GHB
C
C SPECIFICATIONS:
C ------------------------------------------------------------------
DIMENSION BNDS(5,MBEN)
C ------------------------------------------------------------------
C
C1------READ ITMP(# OF GENERAL HEAD BOUNDS OR FLAG TO REUSE DATA.)
READ(IN,8) ITMP
8 FORMAT(I10)
C
C2------TFST ITMP
IF(ITMP.GE.0) GO TO 50
C
C2A------IF ITMP<0 THEN REUSE DATA FROM LAST STRESS PERIOD
WRITE(IOUT,7)
7 FORMAT(IHO,'REUSING HEAD-DEPENDENT BOUNDS FROM LAST STRESS',
     'PERIOD')
GO TO 260
C
C3------IF ITMF=>0 THEN IT IS THE # OF GENERAL HEAD BOUNDS.
50 NBOUND=ITMP
C
C4------IF MAX NUMBER OF BOUNDS IS EXCEEDED THEN STOP
IF(NBOUND.LE.MXBND) GO TO 100
WRITE(IOUT,99) NBOUND,MXBND
99 FORMAT(IH0,'NBOUND(',I4,') IS GREATER THAN MXBND(',I4,')')
C
C4A------ABNORMAL STOP
STOP
C
C5------PRINT # OF GENERAL HEAD BOUNDS THIS STRESS PERIOD
100 WRITE(IOUT,1) NBOUND
     1 FORMAT(IHO,'HEAD-DEPENDENT BOUNDARY NODES')
C
C6------IF THERE ARE NO GENERAL HEAD BOUNDS THEN RETURN.
IF(NBOUND.EQ.0) GO TO 260
C
C7------READ & PRINT DATA FOR EACH GENERAL HEAD BOUNDARY.
WRITE(IOUT,3)
3 FORMAT(IHO,I5X,'LAYER',5X,'ROW',5X,
     'COL ELEVATION CONDUCTANCE BOUND NO.',/I,X,5X,60('-.'))
DO 250 II=1,NBOUND
    READ (IN,4) K,I,J,BNDS(4,II),BNDS(5,II)
4 FORMAT(3I10,2F10.0)
    WRITE (IOUT,5) K,I,J,BNDS(4,II),BNDS(5,II),II
5 FORMAT(1X,15X,14,19,18,G13.4,G14.4,18)
    BNDS(1,II)=K
    BNDS(2,II)=I
    BNDS(3,II)=J
    250 CONTINUE
C
C8------RETURN
260 RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BNDS</td>
<td>Package</td>
<td>DIMENSION (5,MXBND), Layer, row, column, head and conductance from boundary for each general-head boundary.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Row number.</td>
</tr>
<tr>
<td>II</td>
<td>Module</td>
<td>Index for general-head boundaries.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ITMP</td>
<td>Module</td>
<td>Flag or number of boundaries. &lt; 0, same bounds active during the last stress period will be active during the current stress period.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Column number.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Layer number.</td>
</tr>
<tr>
<td>MXBND</td>
<td>Package</td>
<td>Maximum number of head boundaries active at any one time.</td>
</tr>
<tr>
<td>NBOUND</td>
<td>Package</td>
<td>Number of head boundaries active during the current stress period.</td>
</tr>
</tbody>
</table>
Narrative for Module GHBlFM

This module adds terms representing riverhead boundaries to the accumulators HCOF and RHS.

1. If NBOUND is less than or equal to zero in the current stress period, there are no general-head boundaries. RETURN.

2. For each boundary in the BNDS list, DO STEPS 3-6.

3. Determine the column (IC), row (IR), and layer (IL).

4. If the cell is external (IBOUND(IC, IR, IL) < 0), bypass processing on this boundary and go on to the next one.

5. If the cell is internal, get the boundary data (head and conductance).

6. Add the \( -C \times HB \) term (C is the conductance and HB is the boundary head) to the accumulator RHS and the term \(-C\) to the accumulator HCOF.

7. RETURN.
RHS is an accumulator in which the right hand side of the equation is formulated.

HCOF is an accumulator in which the coefficient of head in the cell is formulated.
SUBROUTINE GHBlFM(NBOUND, MXBND, BNDS, HCOF, RHS, IBOUND,
  1 NCOL, NROW, NLAY)

C-----VERSION 1037 10APR1985 GHBlFM
C  *******************************************************
C  ADD GHB TERMS TO RHS AND HCOF
C  *******************************************************
C
C SPECIFICATIONS:
C  **************************************************************************
DIMENSION BNDS(5,MBXBD),HCOF(NCOL,NROW,NLAY),
  1 RHS(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY)
C  **************************************************************************
C
C1------IF NBOUND<=0 THEN THERE ARE NO GENERAL HEAD BOUNDS. RETURN.
  IF(NBOUND.LE.0) RETURN
C
C2------PROCESS EACH ENTRY IN THE GENERAL HEAD BOUND LIST (BNDS)
  DO 100 L=1,NBOUND
C
C3------GET COLUMN, ROW AND LAYER OF CELL CONTAINING BOUNDARY
  IL=BNDS(1,L)
  IR=BNDS(2,L)
  IC=BNDS(3,L)
C
C4------IF THE CELL IS EXTERNAL THEN SKIP IT.
  IF(BOUND(IC,IR,IL).LE.0) GO TO 100
C
C5------SINCE THE CELL IS INTERNAL GET THE BOUNDARY DATA.
  HB=BNDS(4,L)
  C=BNDS(5,L)
C
C6------ADD TERMS TO RHS AND HCOF
  HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-C
  RHS(IC,IR,IL)=RHS(IC,IR,IL)-C*HB
  100 CONTINUE
C
C7------RETURN
  RETURN
  END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BNDS</td>
<td>Package</td>
<td>DIMENSION (5,MXBND), Layer, row, column, head and conductance from boundary for each general-head boundary.</td>
</tr>
<tr>
<td>C</td>
<td>Module</td>
<td>Conductance from the external boundary.</td>
</tr>
<tr>
<td>HB</td>
<td>Module</td>
<td>Head on boundary.</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K) in the finite-difference equation.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell. &lt; 0, constant-head cell = 0, inactive cell &gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>IL</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for boundaries.</td>
</tr>
<tr>
<td>MXBND</td>
<td>Package</td>
<td>Maximum number of head boundaries active at any one time.</td>
</tr>
<tr>
<td>NBOUND</td>
<td>Package</td>
<td>Number of head boundaries active during the current stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>RHS</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.</td>
</tr>
</tbody>
</table>
This module calculates rates and volumes transferred between the aquifer and general-head boundaries.

1. Initialize the cell-by-cell flow-term flag (IBD) and the rate accumulator (RATOUT).

2. If there are no general-head boundaries (NBOUND = 0), skip down to step 13 and put zeros into the budget terms for general-head boundaries.

3. Test to see if cell-by-cell flow terms are to be saved on disk. They will not be saved if either of the following conditions hold: (1) this is not the proper time step (ICBCFL = 0) or (2) cell-by-cell flow terms are not needed for general-head boundaries during this simulation (IGHBCB < 0). If cell-by-cell flow terms will be saved for this package, clear the buffer in which they will be accumulated (BUFF) and set the cell-by-cell flow-term flag (IBD).

4. For each general-head boundary, DO STEPS 5-13 accumulating flows from or into the general-head boundary.

5. Determine the row, column, and layer of the cell containing the general-head boundary.

6. If the cell is external (IBOUND(I,J,K) = 0), bypass further processing of this boundary.

7. Get the boundary parameters from the boundary list (BNDS).

8. Set RATE equal to the boundary conductance times the boundary head minus the head in the cell (RATE = C*(HB - HHNEW)).
9. If cell-by-cell flow terms are to be printed (IGHBCB < 0 and ICBCFL ≠ 0), print RATE.

10. If budget terms for individual cells are to be saved, add the RATE to the buffer (BUFF).

11. Check to see whether flow is into or out of the aquifer.

12. If RATE is negative, add it to RATOUT.

13. If RATE is positive, add it to RATIN.

14. See if cell-by-cell flow terms for individual cells are to be saved (IBD = 1). If they are, call module UBUDSV to record the buffer (BUFF) onto disk.

15. Move RATIN and RATOUT into the VBVL array for printing by BAS1OT. Add RATIN and RATOUT multiplied by the time-step length to the volume accumulators in VBVL for printing by BAS1OT. Move the general-head boundary budget-term labels to VBNM for printing by BAS1OT.

16. Increment the budget-term counter (MSUM). See the section in the Basic Package for a detailed explanation of VBVL, VBNM, and MSUM.

17. RETURN.
IBD is a flag which, if set, causes cell-by-cell flow terms for general-head boundary to be recorded.

EXTERNAL: a cell is said to be external if it is either no flow or constant head (i.e., an equation is not formulated for the cell).

RATE is the leakage rate into the aquifer from the boundary in a cell.

BUFFER is an array in which values are stored as they are being gathered for printing or recording.

RATOUT is an accumulator to which all flows out of the aquifer are added.

RATIN is an accumulator to which all flows into the aquifer are added.

C is the conductance between the boundary and the cell.

HB is the boundary head.

HHNEW is the head in the cell.

IGHBCB is a flag and a unit number.

If IGHBCB > 0, it is the unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.

If IGHBCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IGHBCB < 0, boundary leakage for each cell will be printed whenever ICBCFL is set.

ICBCFL is a flag.

If ICBCFL ≠ 0, cell-by-cell flow terms will be either recorded or printed depending on IGHBCB for the current time step.
SUBROUTINE GHBlBD(NBOUND, MXBND, VBNM, VBVL, MSUM, BNDS, DELT, HNEW,
     1 NCOL, NROW, NLAY, IBOUND, KSTP, KPER, IGHBCB, ICBCFL, BUFF, IOUT)

C------VERSION 1612 12MAY1987 GHBlBD
C ************************************************************
C CALCULATE VOLUMETRIC BUDGET FOR GHB
C ************************************************************
C
C SPECIFICATIONS:
C--------------------------------------------------------------------------------
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW
DIMENSION VBNM(4,MSUM), VBVL(4,MSUM), BNDS(5,MXBND),
1 HNEW(NCOL,NROW,NLAY), IBOUND(NCOL,NROW,NLAY),
2 BUFF(NCOL,NROW,NLAY)
DIMENSION TEXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) / ' HEA', 'D DE', 'P BO', 'UNDS' /
C--------------------------------------------------------------------------------
C
C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATORS (RATIN AND RATOUT)
   IBD=0
   RATOUT=0.
   RATIN=0.
C
C2------IF NO BOUNDARIES THEN KEEP ZEROES IN ACCUMULATORS.
   IF(NBOUND.EQ.0) GO TO 200
C
C3------TEST TO SEE IF CELL-BY-CELL FLOW TERMS ARE NEEDED.
   IF(ICBCFL.EQ.0 .OR. IGHBCB.LE.0) GO TO 10
C
C3A------SINCE CELL-BY-CELL FLOW TERMS ARE NEEDED CLEAR BUFFER & SET
C3A------THE FLAG IBD.
   IBD=1
   DO 5 IL=1,NLAY
     DO 5 IR=1,NROW
       DO 5 IC=1,NCOL
         BUFF(IC,IR,IL)=0.
     5 CONTINUE
C
C4------FOR EACH GENERAL HEAD BOUND ACCUMULATE FLOW INTO AQUIFER
   10 DO 100 L=1,NBOUND
C
C5------GET LAYER, ROW AND COLUMN OF EACH GENERAL HEAD BOUNDARY.
   IL=BNDS(1,L)
   IR=BNDS(2,L)
   IC=BNDS(3,L)
C
C6------IF CELL IS EXTERNAL THEN IGNORE IT.
   IF(IBOUND(IC,IR,IL).LE.0) GO TO 100

C7-------GET PARAMETERS FROM BOUNDARY LIST.
    HHNEW=HNEW(IC,IR,IL)
    HB=BNDS(4,L)
    C=BNDS(5,L)
C
C8-------CALCULATE THE FLOW RATE INTO THE CELL
    RATE=C*(HB-HHNEW)
C
C9-------PRINT THE INDIVIDUAL RATES IF REQUESTED(IGHBCB<0).
    IF(IGHBCB.LT.0.AND.IGBCFL.NE.0) WRITE(IOUT,900) (TEXT(N),N=1,4),
    1       KPER,KSTP,L,IR,IC,RATE
    900 FORMAT(1HD,4A4,' PERIOD',I3,' STEP',I3,' BOUNDARY',I4,
    1       ' LAYER',I3,' ROW',I4,' COL',I4,' RATE',G15.7)
C
C10------IF CELL-BY-CELL TERMS ARE TO BE SAVED THEN PUT RATE IN BUFFER
    IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+RATE
C
C11------SEE IF FLOW IS INTO AQUIFER OR OUT OF AQUIFER.
    IF(RATE).LT.0.AND.ICBCFL.NE.0) WRITE(IOUT,94,100,96
C
C12------FLOW IS OUT OF AQUIFER SUBTRACT RATE FROM RATOUT
    94 RATOUT=RATOUT-RATE
    GO TO 100
C
C13------FLOW IS INTO AQUIFER ADD RATE TO RATIN
    96 RATIN=RATIN+RATE
    100 CONTINUE
C
C14------UTILITY MODULE UBUDSV
    IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IGHBCB,BUFF,NCOL,NROW,
    1       NLAY,IOUT)
C
C15------MOVE RATES, VOLUMES AND LABELS INTO ARRAYS FOR PRINTING
    200 VBVL(3,MSUM)=RATIN
    VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
    VBVL(4,MSUM)=RATOUT
    VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
    VBNM(1,MSUM)=TEXT(1)
    VBNM(2,MSUM)=TEXT(2)
    VBNM(3,MSUM)=TEXT(3)
    VBNM(4,MSUM)=TEXT(4)
C
C16------INCREMENT THE BUDGET TERM COUNTER
    MSUM=MSUM+1
C
C17------RETURN
    RETURN
    END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BNDS</td>
<td>Package</td>
<td>DIMENSION (5,MXBND), layer, row, column, head and conductance from the boundary for each general-head boundary.</td>
</tr>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>C</td>
<td>Module</td>
<td>Conductance from the external boundary.</td>
</tr>
<tr>
<td>DELT</td>
<td>Global</td>
<td>Length of the current time step.</td>
</tr>
<tr>
<td>HB</td>
<td>Module</td>
<td>Head on boundary.</td>
</tr>
<tr>
<td>HHNEW</td>
<td>Module</td>
<td>HNEW (J,I,K), Single precision.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>IBD</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms for this package will not be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, cell-by-cell flow terms for this package will be recorded.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>IGHBCB</td>
<td>Package</td>
<td>Flag and a unit number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, unit number on which cell-by-cell flow terms will be recorded whenever ICBCFL is set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be printed or recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, boundary leakage for each cell will be printed whenever ICBCFL is set.</td>
</tr>
<tr>
<td>ICBCFL</td>
<td>Global</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, cell-by-cell flow terms will not be recorded or printed for the current time step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, cell-by-cell flow terms will be either printed or recorded (depending on IGHBCB) for the current time step.</td>
</tr>
<tr>
<td>I1</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for general-head boundaries.</td>
</tr>
<tr>
<td>MSUM</td>
<td>Global</td>
<td>Counter for budget entries and labels in VBVL and VBNM.</td>
</tr>
<tr>
<td>MXBND</td>
<td>Package</td>
<td>Maximum number of head boundaries active at any one time.</td>
</tr>
<tr>
<td>NBOUND</td>
<td>Package</td>
<td>Number of head boundaries active during the current stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>Variable</td>
<td>Range</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RATE</td>
<td>Module</td>
<td>Flow from a bound into a cell. (Reverse the sign to get flow into the bound.)</td>
</tr>
<tr>
<td>RATIN</td>
<td>Module</td>
<td>Accumulator for the total flow into the flow field out of the bounds.</td>
</tr>
<tr>
<td>RATOUT</td>
<td>Module</td>
<td>Accumulator for the total flow out of the flow field into the bounds.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
<tr>
<td>VBNM</td>
<td>Global</td>
<td>DIMENSION (4,20), Labels for entries in the volumetric budget.</td>
</tr>
<tr>
<td>VBVL</td>
<td>Global</td>
<td>DIMENSION (4,20), Entries for the volumetric budget.</td>
</tr>
</tbody>
</table>

For flow component N, the values in VBVL are:

(1,N), Rate for the current time step into the flow field.
(2,N), Rate for the current time step out of the flow field.
(3,N), Volume into the flow field during simulation.
(4,N), Volume out of the flow field during simulation.
CHAPTER 12

STRONGLY IMPLICIT PROCEDURE PACKAGE

Conceptualization and Implementation

General Theory

The discussion of the Strongly Implicit Procedure (SIP) presented here utilizes certain general concepts of matrix algebra and numerical analysis which may be reviewed in any standard reference, including those noted earlier by Peaceman (1977), Crichlow (1977) or Remson, Hornberger and Molz (1971). In addition to general background material, these three references provide descriptions of the Strongly Implicit Procedure itself which may be consulted to supplement the discussion presented here.

SIP is a method for solving a large system of simultaneous linear equations by iteration. The finite difference equation for a single cell, \( i,j,k \), was shown in Chapter 2 to be of the form

\[
CV_{i,j,k-1/2}h_{i,j,k} + CC_{i-1/2,j,k}h_{i-1,j,k} + CR_{i,j-1/2,k}h_{i,j-1,k} + (-CV_{i,j,k-1/2} - CC_{i-1/2,j,k} - CR_{i,j-1/2,k} - CR_{i,j+1/2,k} - CC_{i+1/2,j,k} - CV_{i,j,k+1/2} + HC_{i,j,k})h_{i,j,k} + CR_{i,j+1/2,k}h_{i,j+1,k} + CC_{i+1/2,j,k}h_{i+1,j,k} + CV_{i,j,k+1/2}h_{i,j,k+1} = \text{RHS}_{i,j,k}.
\]

One equation of this form is written for each cell in the finite-difference grid, expressing the relationship among the heads at node \( i,j,k \), and at each of the six adjacent nodes at the end of a time step. Because each equation may involve up to seven unknown values of head, and because the set of unknown head values changes from one equation to the next through the grid, the equations for the entire grid must be solved simultaneously at each time step. The solution consists of one value of head for each node, for the end of the step.
The discussion of the SIP procedure presented here is based on the notation of Weinstein, Stone and Kwan (1969), the developers of SIP. Using their notation, equation (79) may be written:

\[ Z_{i,j,k-1} + B_{i,j,k-1} + D_{i,j,k} + E_{i,j,k} + F_{i,j,k} + S_{i,j,k} = Q_{i,j,k} \]  

(30)

The coefficients in equation (80) all are labelled with the index \( i,j,k \) to show that they are associated with the equation for node \( i,j,k \). Thus \( Z_{i,j,k} \) of equation (80) is equivalent to \( C_{i,j,k-1/2} \) of equation (79); \( E_{i,j,k} \) of equation (80) is equivalent to the expression \((-C_{i,j,k-1/2} - C_{i-1/2,j,k} - C_{i+1/2,j,k} - C_{i+1/2,j,k} + H_{C_{i,j,k+1/2}})\) of equation (79); and so on.

As pointed out in Chapter 2, the entire set of equations of the form of (80) can be summarized in matrix form as

\[ [A] \{h\} = \{q\} \]  

(31)

where \([A]\) is the matrix of coefficients of head, \(\{h\}\) is a vector of head values, and \(\{q\}\) is a vector of the right-hand terms of equation (80). Figure 46 shows the elements of the coefficient matrix and of the two vectors for a mesh of three rows, four columns and two layers. Notice that the matrix \([A]\) is sparse--i.e., that there are very few nonzero elements--and that these are all located on just seven diagonals, as indicated in figure 47.

Examination of equations (79) and (80) will show that the term \( C_{i,j,k-1/2} \) of equation (79) appears both as the coefficient \( Z \) in equation (80) for node \( i,j,k \), and as the coefficient \( S \) in the corresponding equation for node \( i,j,k-1 \), that is

\[ Z_{i,j,k} = S_{i,j,k-1} \]  

(32)
### Figure 46

Correspondence between the finite-difference equations and the matrix equation for a grid of three rows, four columns, and two layers.

- $E_1$, $F_1$, $H_1$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0$, $0
Brackets indicate horizontal spacing, in matrix columns, between nonzero diagonals (e.g., diagonals E and F are adjacent).

Figure 47.—Structure of coefficient matrix showing nonzero diagonals.
Similarly,
\[ B_{i,j,k} = H_{i-1,j,k} \]  
and  
\[ D_{i,j,k} = F_{i,j-1,k} \]  

Replacing each \( Z, B, \) and \( D \) coefficient in the matrix of Figure 46 with the equivalent \( S, H, \) or \( F \) element, as defined by equations (82) - (84), yields the matrix of Figure 48, which is readily seen to be symmetric. Thus the coefficient matrix \([A]\) of equation (81) is symmetric as well as sparse.

A system of equations of the form of (81) can be solved by direct methods if \([A]\) can be factored into two matrices \([L^*]\) and \([U^*]\), such that \([L^*]\) is in lower triangular form (all nonzero elements are on or below the main diagonal), while \([U^*]\) is in upper triangular form (all nonzero elements are on or above the main diagonal), and all elements on the main diagonal of \([U^*]\) are equal to one. Figure (49) illustrates the characteristics of \([L^*]\) and \([U^*]\) relative to \([A]\) for a 3 x 3 matrix \([A]\). Once this factoring has been accomplished, a technique known as "backward and forward substitution" can be used to complete the solution. However, a difficulty arises in that, even though \([A]\) is a sparse matrix, \([L^*]\) and \([U^*]\) are generally not sparse, and a great deal of computer memory and time may be needed to calculate all of their nonzero elements. In addition, round-off errors may become unacceptably large.

The Strongly Implicit Procedure seeks to find a matrix \([B]\) such that the sum matrix \([A + B]\) can be factored easily into two matrices \([L]\) and \([U]\), where \([A + B]\), \([L]\), and \([U]\) meet the following conditions:

1. \([A + B]\) is "close" to \([A]\);
2. \([L]\) is in lower triangular form while \([U]\) is in upper triangular form, and all entries along the main diagonal of \([U]\) are equal to unity;
|   | $E_{1,1,1}$ | $F_{1,1,1}$ | $H_{1,1,1}$ | $S_{1,1,1}$ | $E_{1,2,1}$ | $F_{1,2,1}$ | $H_{1,2,1}$ | $S_{1,2,1}$ | $E_{1,3,1}$ | $F_{1,3,1}$ | $H_{1,3,1}$ | $S_{1,3,1}$ | $E_{2,1,1}$ | $F_{2,1,1}$ | $H_{2,1,1}$ | $S_{2,1,1}$ | $E_{2,2,1}$ | $F_{2,2,1}$ | $H_{2,2,1}$ | $S_{2,2,1}$ | $E_{2,3,1}$ | $F_{2,3,1}$ | $H_{2,3,1}$ | $S_{2,3,1}$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Figure 48.—Symmetric coefficient matrix for a grid containing two rows, three columns, and two layers.
Figure 49.—Decomposition of a coefficient matrix into lower and upper triangular matrices.

\[
\begin{bmatrix}
1 & 2 & 1 \\
-1 & 1 & 2 \\
3 & 2 & -2
\end{bmatrix}
\begin{bmatrix}
h_1 \\
h_2 \\
h_3
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
2 \\
-3
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 0 & 0 \\
-1 & 3 & 0 \\
3 & -4 & -1
\end{bmatrix}
\begin{bmatrix}
1 & 2 & 1 \\
0 & 1 & 1 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
h_1 \\
h_2 \\
h_3
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
2 \\
-3
\end{bmatrix}
\]
(3) $[L]$ and $[U]$ are both sparse; and

(4) both $[L]$ and $[U]$ have just four nonzero diagonals.

Suppose a matrix $[B]$ is constructed in an attempt to satisfy these conditions; the term $[B] \{h\}$ can be added to each side of equation (81) to give

$$[A + B] \{h\} = \{q\} + [B] \{h\} \quad (85)$$

A solution vector $\{h\}$ for equation (85) must also be a solution vector for equation (81). The presence of the vector $\{h\}$ on both sides of equation (85) presents an immediate difficulty; however, if an iterative approach to the solution is utilized (chapter 2), values of $h$ from the preceding iteration may be used in the head vector on the right. That is, equation (85) may be expressed in the form

$$[A + B] \{h^k\} = \{q\} + [B] \{h^{k-1}\} \quad (86)$$

where $\{h^k\}$ is a vector of head values from iteration $k$, and $\{h^{k-1}\}$ a vector of head values from iteration $k-1$. In equation (86), $\{h^{k-1}\}$ is actually used as an approximation to $\{h^k\}$. If the matrix $[B]$ were known, solution of (86) would be straightforward; for according to the properties postulated above, $[A + B]$ could be factored easily into the sparse matrices $[L]$ and $[U]$, allowing the use of forward and backward substitution. Thus the problem of solving equation (86) is equivalent to that of finding an appropriate matrix $[B]$. In practice, however, the solution is pursued in terms of the matrices $[A]$, $[A + B]$, $[L]$ and $[U]$. The term $[A + B] \{h^{k-1}\}$ is subtracted from each side of (86) to yield

$$[A + B] \{h^k\} - [A + B] \{h^{k-1}\} = \{q\} - [A] \{h^{k-1}\} \quad (87)$$

or

$$[A + B] \{h^k - h^{k-1}\} = \{q\} - [A] \{h^{k-1}\} \quad (88)$$
In order that the conditions specified above for \([L]\), \([U]\), and \([A + B]\) may be satisfied, \([A + B]\) must contain six nonzero diagonals which were not present in \([A]\), as shown in figure 50; the effect of these additional nonzero diagonals is to introduce new terms into the equation for node \(i,j,k\), involving heads at nodes not adjacent to \(i,j,k\). The relationship between the elements of \([A + B]\) and the elements of \([L]\) and \([U]\) is as given in the following equations, where as indicated in figures 50 and 51, a, b, c, and d, refer to elements of \([L]\), e, f, and g, refer to elements of \([U]\) above the main diagonal, and capital letters refer to elements of \([A + B]\).

\[
\begin{align*}
Z'_{i,j,k} &= a_{i,j,k} \quad (89-a) \\
A'_{i,j,k} &= a_{i,j,k}e_{i,j,k-1} \quad (89-b) \\
T'_{i,j,k} &= a_{i,j,k}f_{i,j,k-1} \quad (89-c) \\
B'_{i,j,k} &= b_{i,j,k} \quad (89-d) \\
C'_{i,j,k} &= e_{i-1,j,k}b_{i,j,k} \quad (89-e) \\
D'_{i,j,k} &= c_{i,j,k} \quad (89-f) \\
E'_{i,j,k} &= a_{i,j,k}g_{i,j,k-1} + b_{i,j,k}f_{i-1,j,k} \\
&\quad + e_{i-1,j,k}c_{i,j,k} + d_{i,j,k} \quad (89-g) \\
F'_{i,j,k} &= d_{i,j,k}e_{i,j,k} \quad (89-h) \\
G'_{i,j,k} &= f_{i,j-1,k}c_{i,j,k} \quad (89-i) \\
H'_{i,j,k} &= f_{i,j,k}d_{i,j,k} \quad (89-j) \\
U'_{i,j,k} &= b_{i,j,k}g_{i-1,j,k} \quad (89-k) \\
R'_{i,j,k} &= g_{i,j-1,k}c_{i,j,k} \quad (89-l) \\
S'_{i,j,k} &= g_{i,j,k}d_{i,j,k} \quad (89-m)
\end{align*}
\]

If the subscript of an element in equations (89-a...m) places the element outside of the grid boundary, the element is assumed to be equal to zero. The 13 equations contain 20 unknown values, the elements of \([L]\), \([U]\), and
Brackets indicate horizontal spacing, in matrix columns, between nonzero diagonals (e.g., diagonals E and D are adjacent); asterisks mark nonzero diagonals which are not present in matrix $A$. 

Figure 50.—Structure of matrix $[A+B]$ showing nonzero diagonals.
Brackets indicate vertical spacing, in matrix rows, between nonzero diagonals (e.g., diagonals d and e are adjacent).

Figure 51.—Structure, showing nonzero diagonals, of the lower triangular factor \([L]\) and the upper triangular factor \([U]\) of matrix \([A+B]\).
This indicates that there are many matrices \([B]\) which can be added to \([A]\) so that the sum can be factored into upper and lower triangular matrices of the form of \([L]\) and \([U]\). However, the requirement that \([A + B]\) must be "close" to \([A]\), or equivalently that

\[
[A + B] \{h\} = [A] \{h\}
\] (90)

has not yet been used. In terms of the elements of \([A + B] \{h\}\) and \([A] \{h\}\) associated with an individual node, \(i,j,k\), equation (90) implies that

\[
Z'_{i,j,k}hi,j,k-1 + A'_{i,j,k}hi,j,k+1,k-1 + T'_{i,j,k}hi+1,j,k-1
+ B'_{i,j,k}hi-1,j,k + C'_{i,j,k}hi-1,j,k+1,k + D'_{i,j,k}hi,j,k-1,k
+ E'_{i,j,k}hi,j,k + F'_{i,j,k}hi,j,k+1,k + G'_{i,j,k}hi+1,j,k-1,k
+ H'_{i,j,k}hi+1,j,k + U'_{i,j,k}hi,j,k-1,k + R'_{i,j,k}hi,j,k-1,k+1
+ S'_{i,j,k}hi,j,k+1,k+1 = Z_{i,j,k}hi,j,k-1 + B_{i,j,k}hi-1,j,k
+ D_{i,j,k}hi,j,k-1,k + E_{i,j,k}hi,j,k+1,k + F_{i,j,k}hi,j,k+1,k + H_{i,j,k}hi+1,j,k
+ S_{i,j,k}hi,j,k+1,k+1
\] (91)

Equation (91) can be rearranged so that the terms from the six nonzero diagonals not present in \([A]\) are all on the right side, while the left side is made up of differences between elements of matrix \([A]\) and corresponding elements of matrix \([A + B]\), i.e.

\[
(Z_{i,j,k}-Z'_{i,j,k})hi,j,k-1 + (B_{i,j,k}-B'_{i,j,k})hi,j,k
+ (D_{i,j,k}-D'_{i,j,k})hi,j,k-1 + (E_{i,j,k}-E'_{i,j,k})hi,j,k
+ (F_{i,j,k}-F'_{i,j,k})hi,j,k+1 + (H_{i,j,k}-H'_{i,j,k})hi+1,j,k
+ (S_{i,j,k}-S'_{i,j,k})hi,j,k+1 = A'_{i,j,k}hi,j,k+1,k-1
+ T'_{i,j,k}hi+1,j,k-1 + C'_{i,j,k}hi-1,j,k+1,k + G'_{i,j,k}hi+1,j,k-1,k
+ U'_{i,j,k}hi,j,k+1,k+1 + R'_{i,j,k}hi,j,k-1,k+1
\] (92)

The terms on the right side of (92), corresponding to the six nonzero diagonals of \([A + B]\) not appearing in \([A]\), are all derived from the matrix \(B\), and all involve the heads at nodes not adjacent to node, \(i,j,k\); by contrast,
the terms on the left side of (92) are derived from both $[A]$ and $[B]$, and involve the heads at $i,j,k$ and the six adjacent nodes.

To reduce the effect of the terms corresponding to nodes not adjacent to $i,j,k$, three parameters, here termed $\alpha$, $\beta$, and $\gamma$, and all chosen between zero and one, are introduced as multipliers of the terms on the right side of equation (92). Ultimately, as the solution of the matrix equations ((85) or (86)) is implemented, these multipliers take on the role of iteration parameters. They are brought into equation (92) as follows:

\[
\begin{align*}
& (Z_{i,j,k} - Z'_{i,j,k}) h_{i,j,k-1} + (B_{i,j,k} - B'_{i,j,k}) h_{i-1,j,k} \\
& + (D_{i,j,k} - D'_{i,j,k}) h_{i,j-1,k} + (E_{i,j,k} - E'_{i,j,k}) h_{i,j,k} \\
& + (F_{i,j,k} - F'_{i,j,k}) h_{i,j+1,k} + (H_{i,j,k} - H'_{i,j,k}) h_{i+1,j,k} \\
& + (S_{i,j,k} - S'_{i,j,k}) h_{i,j,k+1} = \alpha A_{i,j,k} h_{i,j,k+1} \\
& + \beta T_{i,j,k} h_{i,j+1,k-1} + \gamma C_{i,j,k} h_{i-1,j,k+1} \\
& + \gamma G_{i,j,k} h_{i+1,j-1,k} + \beta U_{i,j,k} h_{i,j-1,k+1} \\
& + \alpha R_{i,j,k} h_{i,j,k+1}
\end{align*}
\]

(93)

Next the heads on right side of (93), corresponding to nodes not adjacent to $i,j,k$, are expressed in terms of heads at nodes which are adjacent to $i,j,k$. This is done by noting that, for example, node $i,j+1,k-1$ lies at the corner of a rectangle, the other three corners of which are: $i,j,k-1; i,j+1,k;$ and $i,j,k$. Thus using the rules in interpolation illustrated in figure 52, $h_{i,j+1,k-1}$ is given approximately by

\[
h_{i,j+1,k-1} = h_{i,j+1,k} + h_{i,j,k-1} - h_{i,j,k},
\]

(94-a)

Similarly,

\[
h_{i+1,j,k-1} = h_{i+1,j,k} + h_{i,j,k-1} - h_{i,j,k},
\]

(94-b)

\[
h_{i-1,j,k+1} = h_{i-1,j,k} + h_{i,j+1,k} - h_{i,j,k},
\]

(94-c)
Suppose the Function $f$ is Known at 2, 3 and 4.

By interpolation the Function at the Center Can be Approximated by

$$f_1(c) \approx \frac{f(2) + f(3)}{2}$$

and

$$f_2(c) \approx \frac{f(1) + f(4)}{2}$$

Suppose

$$f_1(c) = f_2(c)$$

Then

$$\frac{f(2) + f(3)}{2} \approx \frac{f(1) + f(4)}{2}$$

Therefore

$$f(1) \approx f(2) + f(3) - f(4)$$

Figure 52.—Estimation of a function at one corner of a rectangle in terms of the values of the function at the other three corners.
\[ h_{i+1,j,k-1} = h_{i+1,j,k} + h_{i,j-1,k} - h_{i,j,k} \tag{94-d} \]
\[ h_{i-1,j,k+1} = h_{i-1,j,k} + h_{i,j-1,k} - h_{i,j,k} \tag{94-e} \]
\[ h_{i,j-1,k+1} = h_{i,j,k+1} + h_{i,j-1,k} - h_{i,j,k} \tag{94-f} \]

Substituting equations (94-a...f) into equation (93) and reorganizing gives

\[
\begin{align*}
& (Z_i,j,k - Z_i,j,k + \alpha A_i,j,k + \beta T_i,j,k) \ h_{i-1,j,k} \\
& + (B_i,j,k - B_i,j,k + \gamma C_i,j,k + \beta U_i,j,k) \ h_{i,j-1,k} \\
& + (D_i,j,k - D_i,j,k + \gamma G_i,j,k + \alpha R_i,j,k) \ h_{i,j+1,k} \\
& + (E_i,j,k - E_i,j,k - \alpha A_i,j,k - \beta T_i,j,k - \gamma C_i,j,k \\
& - \gamma G_i,j,k - \beta U_i,j,k - \alpha R_i,j,k) \ h_{i,j,k} \\
& + (F_i,j,k - F_i,j,k + \alpha A_i,j,k + \gamma C_i,j,k) \ h_{i+1,j,k} \\
& + (H_i,j,k - H_i,j,k + \beta T_i,j,k + \gamma G_i,j,k) \ h_{i,j,k} \\
& + (S_i,j,k - S_i,j,k + \beta U_i,j,k + \alpha R_i,j,k) \ h_{i,j,k+1} = 0 
\end{align*}
\tag{95}
\]

The relation expressed in equation (95) can be satisfied if each coefficient is approximately equal to zero. Setting these coefficients equal to zero yields the equations

\[
\begin{align*}
& Z_i,j,k - Z_i,j,k + \alpha A_i,j,k + \beta T_i,j,k = 0 \tag{96-a} \\
& B_i,j,k - B_i,j,k + \gamma C_i,j,k + \beta U_i,j,k = 0 \tag{96-b} \\
& D_i,j,k - D_i,j,k + \gamma G_i,j,k + \alpha R_i,j,k = 0 \tag{96-c} \\
& E_i,j,k - E_i,j,k - \alpha A_i,j,k - \beta T_i,j,k - \gamma C_i,j,k \\
& - \gamma G_i,j,k - \beta U_i,j,k - \alpha R_i,j,k = 0 \tag{96-d} \\
& F_i,j,k - F_i,j,k + \alpha A_i,j,k + \gamma C_i,j,k = 0 \tag{96-e} \\
& H_i,j,k - H_i,j,k + \beta T_i,j,k + \gamma G_i,j,k = 0 \tag{96-f} \\
& S_i,j,k - S_i,j,k + \beta U_i,j,k + \alpha R_i,j,k = 0 \tag{96-g} 
\end{align*}
\]

Equations (96-a...g) and (89-a...m) form a system of 20 equations in 20 unknowns which when solved, will yield the entries of \([A + B], [L]\) and \([U]\) such that \([A + B]\) is "close" to \([A]\), and can be readily factored into

12-15
[L] and [U], where [L] and [U] are both sparse and have the required
lower triangular and upper triangular forms. For example, substituting
equations (89-a, -b, and -c) into equation (96-a) and rearranging yields
\[ a_{i,j,k} = z_{i,j,k} / (1 + \alpha e_{i,j,k-1} + \beta f_{i,j,k-1}) \]
Similarly,
\[ b_{i,j,k} = b_{i,j,k} / (1 + \gamma e_{i-1,j,k} + \delta g_{i-1,j,k}) \]
\[ c_{i,j,k} = c_{i,j,k} / (1 + \gamma f_{i,j-1,k} + \alpha g_{i,j-1,k}) \]
\[ A'_{i,j,k} = a_{i,j,k} - e_{i,j,k} \]
\[ C'_{i,j,k} = c_{i,j,k} / (1 + \gamma f_{i,j-1,k} + \alpha g_{i,j-1,k}) \]
\[ G'_{i,j,k} = f_{i,j-1,k} - c_{i,j,k} \]
\[ R'_{i,j,k} = r_{i,j-1,k} - b_{i,j,k} \]
\[ T'_{i,j,k} = a_{i,j,k} - e_{i-1,j,k} \]
\[ U'_{i,j,k} = b_{i,j,k} - f_{i,j-1,k} \]
\[ d_{i,j,k} = d_{i,j,k} + \alpha A'_{i,j,k} + \beta T'_{i,j,k} \]
\[ + e_{i,j,k} / (1 + \gamma C'_{i,j,k} + \gamma G'_{i,j,k}) \]
\[ + \alpha R'_{i,j,k} - a_{i,j,k} \]
\[ + \beta U'_{i,j,k} - a_{i,j,k} \]
\[ + e_{i,j,k} / (1 + \gamma C'_{i,j,k} + \gamma G'_{i,j,k}) \]
Using these relations to provide the elements of [L] and [U], [A + B] may
be replaced with the product [L][U] in (88) to yield
\[ [L][U] \{ h^{R} - h^{R-1} \} = [A] \{ h^{R-1} \} \]
where again the superscript \( R \) refers to the current iteration level, and
\( R-1 \) to the preceding iteration level. We next define the vector \{RES\} by
The vector \( \{v\} \) in the equation
\[
[D-1] \{v\} = \{R_{Eg}\}
\]
where \( \{v\} = [U] \{R_{Eg-h_{a}-1}\} \). The vector \( \{v\} \) determined in this way is then utilized in a process of back substitution to solve for the vector
\( \{h_{R-h_{R-1}}\} \) in the equation
\[
[U] \{h_{R-h_{R-1}}\} = \{v\}
\] (102)

In earlier discussions, the coefficients of the equations and hence the elements of the matrices were identified by the indices of the cells, as shown in figure 53-a. To illustrate the process of forward substitution, used to calculate the elements of the vector \( \{v\} \), it is convenient to renumber the equations sequentially using a single index, as shown in figure 53-b. Because all elements in \( [L] \) above the main diagonal are zero, the first linear equation represented by matrix equation (101) is
\[
d_1 v_1 = R_{E_1^{a}}
\] (103)

In equation (103), the term \( d_1 \) has been determined through equation (97-j), and \( R_{E_1^{a}} \) has been calculated through equation (99) as an element of the vector \( \{R_{Eg}\} \); thus (103) can be solved immediately for the value of \( v_1 \). The second equation represented by matrix equation (101) is
\[
c_2 v_1 + d_2 v_2 = R_{E_2^{a}}
\] (104)
Figure 53.—Cell numbering schemes for a grid using three indices and using one index.
Again, $c_2$ and $d_2$ are known from equations (97), and $\text{RESZ}_2^2$ is known from equation (99); using the value of $v_1$ from the solution of equation (103), (104) can be solved for $v_2$.

The general equation for an element of $\{v\}$ has the form

$$v_n = \frac{(\text{RES}_n^2 - a_n v_{n-NRC} - b_n v_n - c_n v_{n+1})}{d_n} \quad (105)$$

where NRC is the number of cells in the layer, NCOL is the number of columns in the model, the coefficients $a_n, b_n, c_n$ and $d_n$ are all determined through equations (97) and $\text{RES}_n^2$ is determined through equation (99).

The terms $a_n$ and $b_n$ are zero for the first and second equations ((103) and (104)); and each equation involves elements of $\{v\}$ determined earlier in the sequence. This procedure of forward substitution, in which the elements of $\{v\}$ are determined in sequence, is possible because of the lower triangular form of the matrix $[L]$--i.e., because $[L]$ has only zeros to the right of the main diagonal.

Back substitution is next used to calculate the elements of the vector $\{h^k - h^{k-1}\}$ from the elements of $\{v\}$, thus solving equation (102). The process of back substitution is similar to that of forward substitution except that, because the matrix $[U]$ is upper triangular, the order of calculation is reversed. When the vector $\{h^k - h^{k-1}\}$ has been calculated, it is added to the vector $\{h^{k-1}\}$ to obtain $\{h^k\}$, the vector of head values corresponding to iteration $k$.

In summary, the problem of solving the equation

$$[A]\{h\} = \{q\} \quad (106)$$

has thus been converted into an iterative process in which: (1) the matrices $[L]$ and $[U]$ are determined using equations (97); (2) the vector $\{\text{RES}_k\}$ is calculated using the vector $\{q\}$, the matrix $[A]$ and heads from the preceding iteration; (3) equation (100) is then solved using forward
and backward substitution to obtain the vector \( \{h^L - h^{L-1}\} \); and (4) the vector \( \{h^L - 1\} \) is added to the vector \( \{h^L - h^{L-1}\} \) to obtain the vector \( \{h^L\} \). However, while these are the essential steps of the SIP procedure, several aspects of the method remain to be discussed.

Transfer of Arrays

As noted previously, the coefficient matrix \([A]\) is sparse, with only seven nonzero diagonals. Rather than passing the entire matrix to the SIP Package, only the nonzero diagonals are needed; and because of symmetry of the matrix, only the main diagonal and the three lower diagonals are needed. The three lower diagonals correspond to the conductance arrays \(CC\), \(CR\), and \(CV\). The main diagonal is formed from the three conductance arrays and the array HCOF described in Chapter 2. The right hand side of the matrix equation, \(\{q\}\), corresponds to the array RHS described in Chapter 2. The latest estimate of the head distribution \(\{h^{L-1}\}\), corresponds to the array HNEW. As new estimates of head are calculated by SIP, they are stored in HNEW replacing the previous estimates. Thus input to SIP consists of the following arrays: \(CC\), \(CR\), \(CV\), \(RHS\), \(HCOF\), and \(HNEW\). Output from SIP consists of a new HNEW array. As explained in Chapter 3, the Formulate Procedure is inside the iteration loop; therefore, the input arrays may be modified at each iteration.
Experience has shown that if the finite-difference equations are solved in two different orders on alternate iterations, the number of iterations needed to converge to a solution is reduced. The order assumed in the discussion, to this point, has been to begin at the first column, the first row, and the first layer, and to proceed in ascending column order, ascending row order, and ascending layer order. An alternative is to start at the first column, the last row, and the last layer, and to proceed in ascending column order, descending row order, and descending layer order. Using the same ordering of diagonal names used in figure 51, equations similar to equations (97-a...m) can be developed. They are

\[
\begin{align*}
    a_{i,j,k} &= z_{i,j,k}/(1+\alpha e_{i+1,j,k}+\beta f_{i,j+k+1}) \\
    b_{i,j,k} &= B_{i,j,k}/(1+\gamma e_{i+1,j,k}+\beta g_{i,j+1,k}) \\
    c_{i,j,k} &= D_{i,j,k}/(1+\gamma f_{i,j-1,k}+\alpha g_{i,j-1,k}) \\
    A'_{i,j,k} &= a_{i,j,k}e_{i,j,k+1} \\
    C'_{i,j,k} &= e_{i+1,j,k}b_{i,j,k} \\
    G'_{i,j,k} &= f_{i,j-1,k}c_{i,j,k} \\
    R'_{i,j,k} &= g_{i,j-1,k}c_{i,j,k} \\
    T'_{i,j,k} &= a_{i,j,k}f_{i,j,k+1} \\
    U'_{i,j,k} &= b_{i,j,k}g_{i,j+1,k} \\
    d_{i,j,k} &= E_{i,j,k} + \alpha A'_{i,j,k} + \beta T'_{i,j,k} + \gamma C'_{i,j,k} + \gamma G'_{i,j,k} + \beta U'_{i,j,k} \\
    &+ \alpha R'_{i,j,k} - a_{i,j,k}f_{i,j,k+1} - b_{i,j,k}g_{i,j+1,k} - e_{i,j-1,k}c_{i,j,k}
\end{align*}
\]
\[ e_{i,j,k} = (F_{i,j,k} - \alpha A'_{i,j,k} - \gamma C'_{i,j,k} - d_{i,j,k})/d_{i,j,k} \]  
(107-k)

\[ f_{i,j,k} = (H_{i,j,k} - \beta T'_{i,j,k} - \gamma G'_{i,j,k} - d_{i,j,k})/d_{i,j,k} \]  
(107-l)

\[ g_{i,j,k} = (S_{i,j,k} - \alpha R'_{i,j,k} - \beta U'_{i,j,k} - d_{i,j,k})/d_{i,j,k} \]  
(107-m)

In the model described herein, equations (107-a...m) and equations (97-a...m) are in effect invoked alternately in successive iterations.

The model program actually uses one general set of equations in which the variables are identified by single indices. The ordering of (97) or of (107) is then achieved through the sequence of values assigned to the indices. In the following list of these general equations, the index nll refers to the cell in the previous layer calculated, but in the same row and column as cell n; the indices nrl and ncl are defined analogously.

Also, in these equations, the iteration parameters \( \alpha \), \( \beta \) and \( \gamma \) have each been replaced by a single parameter \( \omega \) as explained in the following section. Note that one additional equation has been added to the list—the equation for \( v_n \), the element of the vector \( \{v\} \) corresponding to cell \( n \). This equation can be added inasmuch as \( v_n \) can be calculated as soon as the \( n^{th} \) rows of the matrices \( [L] \) and \( [U] \) have been calculated. The equations are

\[ a_n = z_n/(1 + \omega(e_{n11} + \tau_{n11})) \]  
(108-a)

\[ b_n = z_n/(1 + \omega(e_{nrl} + g_{nrl})) \]  
(108-b)

\[ c_n = d_n/(1 + \omega(f_{ncl} + g_{ncl})) \]  
(108-c)

\[ A'_{n} = a_{n}e_{n11} \]  
(108-d)

\[ C'_{n} = b_{n}e_{nrl} \]  
(108-e)

\[ G'_{n} = c_{n}f_{ncl} \]  
(108-f)

\[ R'_{n} = c_{n}g_{ncl} \]  
(108-g)

\[ T'_{n} = a_{n}f_{n11} \]  
(108-h)

\[ U'_{n} = b_{n}g_{nrl} \]  
(108-i)
\[ d_n = E_n + \omega (A_n' + T_n' + C_n' + G_n' + U_n' + R_n') \]
\[ e_n = (F_n - \omega (A_n' + C_n'))/d_n \]  
\[ f_n = (H_n - \omega (T_n' + G_n'))/d_n \]  
\[ g_n = (S_n - \omega (R_n' + U_n'))/d_n \]  
\[ v_n = (R_n - \omega (R_n + U_n))/d_n \]

Since the backward substitution requires all values of \( e_n, f_n, g_n, \) and \( v_n \), space is allocated in the SIP Package for four arrays to store those values. Each of these arrays has as many elements as there are cells in the grid.

**Iteration Parameters**

While Weinstein, Stone and Kwan (1969) define three iteration parameters in their theoretical development, they utilize a single value in practice. Thus the terms \( a, b \) and \( c \) of equation (93) are replaced by a single parameter, \( \omega \), which multiplies each term on the right side of the equation; however, \( \omega \) must be cycled through a series of values in successive iterations to achieve satisfactory rates of convergence. In the model described herein, values of \( \omega \) are calculated from the expression

\[ \omega (\lambda) = 1-(WSEED)(\lambda-1)/(NPARM-1) \]

where \( NPARM \) is the total number of \( \omega \) values to be used; \( \lambda \) is an index taking on integral values from 1 to \( NPARM \); \( \omega (\lambda) \) is the corresponding iteration parameter value; and WSEED is the iteration parameter "seed", calculated according to rules outlined below, and used as a basis for determining the sequence of \( \omega \) values.

The value of WSEED is in turn developed as follows. The terms \( \rho_1, \rho_2, \) and \( \rho_3 \) are calculated for each cell in the mesh using the conductances between that cell and its neighbors, as follows
\[ \rho_1 = \frac{CC_{\text{max}} + CV_{\text{max}}}{CR_{\text{min}}} \quad (110) \]

\[ \rho_2 = \frac{CR_{\text{max}} + CV_{\text{max}}}{CC_{\text{min}}} \quad (111) \]

\[ \rho_3 = \frac{CR_{\text{max}} + CC_{\text{max}}}{CV_{\text{min}}} \quad (112) \]

where \( CC_{\text{max}} \) for a given cell, \( i,j,k \), is the larger of \( CC_{i-1/2,j,k} \) and \( CC_{i+1/2,j,k} \), while \( CC_{\text{min}} \) is the smaller of these values; and similarly \( CR_{\text{max}} \) is the larger of \( CR_{i,j-1/2,k} \) and \( CR_{i,j+1/2,k} \), while \( CR_{\text{min}} \) is the smaller, and \( CV_{\text{max}} \) is the larger of \( CV_{i,j,k-1/2} \) and \( CV_{i,j,k+1/2} \), while \( CV_{\text{min}} \) is the smaller. Using these values, the terms

\[ \frac{\pi^2}{2(NCOL)^2(1+\rho_1)} \quad , \quad \frac{\pi^2}{2(NROW)^2(1+\rho_2)} \quad \text{and} \quad \frac{\pi^2}{2(NLAY)^2(1+\rho_3)} \]

are computed for each cell in the grid, where again \( NCOL \) is the number of columns in the model, \( NROW \) is the number of rows and \( NLAY \) is the number of layers; and the minimum value for each of these three terms is taken to be the cell seed. The seed term, \( WSEED \), is then taken as average of all the cell seeds. The iteration parameters, \( \omega(\lambda) \), generated from the \( WSEED \) value are then used sequentially in successive iterations, recycling each time the entire set has been used (i.e., each \( NPARM \) iterations).

The process described above for calculating the sequence of \( \omega \) values differs slightly from that used by Weinstein, Stone and Kwan (1969), but produces values that are in the same range and that appear to function well in many problems. However, several points should be made regarding iteration parameter selection. First, the process is essentially empirical,
The model described herein provides for an additional iteration parameter, referred to here as the acceleration parameter to distinguish it from $\omega$. This parameter, which is designated ACCL in the program, functions as a multiplier of $\{\text{RES}^k\}$; thus where a parameter of this type is not to be used, ACCL is simply assigned a value of one. Parameters similar to ACCL have been used in various versions of SIP (Peaceman, 1977, page 130) although Weinstein, Stone and Kwan (1969) do not employ a parameter of this type. ACCL is not cycled, but rather is assigned a single value by the user. As a general rule, it should initially be given a value of one, and improvement in the rate of convergence should be pursued through adjustment of the seed term, as explained below. If problems with convergence persist, values of ACCL other than one can be tried.

Experience has shown that setting the acceleration/relaxation parameter (ACCL) to 1 and using the seed value calculated by the program does not always produce optimum convergence—that is, the number of iterations required to achieve convergence is not minimum. Convergence rates will deviate from the optimum if the absolute value of head change in each iteration is consistently either too small or too large. When the head change is too large, the computed head overshoots the correct value, and oscillations occur as the head change repeatedly reverses to compensate
for the overshoot. Severe overshoot causes divergence, while moderate overshoot simply slows down convergence. When head change is too small, the opposite problem occurs; head tends to approach the correct value monotonically, but very slowly. In severe situations, the head changes at each iteration may be so small that the criterion for convergence is satisfied at all points, even though the computed heads are still far from the correct values. In such situations, a significant volumetric budget imbalance will occur.

Weinstein, Stone, and Kwan (1969) suggest that a trial and error method can be used to improve the choice of seed. This can be done by making an initial run using the seed calculated by the program or chosen from experience, and using ACCL=1. The trend of head change per iteration, with increasing iteration number, is observed for the iterations of a single time step. There is normally some variation in head change from one iteration to the next due to the cycling of iteration parameters, but this variation is often superimposed on an overall trend in which head change tends either to increase or decrease as iterations continue; it is this overall trend (which is often most evident in the later iterations of the test) that is of interest here. Some oscillations (reversals in sign) of the computed head change are normal during convergence; however, repeated oscillation is a sign of overshoot, indicating that computed head changes are too great for optimal convergence. Head changes which are too small, on the other hand, are indicated by a very flat overall trend. For proper evaluation of the trend, the trial should generally be run for a number of iterations equal to 4 or 5 times the number of iteration parameters, unless convergence occurs before this.
Following the initial trial, the seed is multiplied by a number between two and ten if head changes in the initial trial appear to be too great, and divided by a number between two and ten if those head changes appear to be too small. If the trend in the initial trial is unclear either multiplication or division of the seed may be tried. In any case, a second run is made using the new seed value, and the trend of head change vs. iteration level is again examined. The results are compared with those of the initial trial to see if the rate of convergence has improved. If both runs have converged, the comparison is based on the number of iterations required for convergence; if they have failed to converge, the comparison is based on the head changes observed in the final iterations.

The trial runs can be continued to further refine the choice of seed; in general the seed value will be multiplied or divided by progressively smaller numbers at each step of the procedure. However, it is usually not worthwhile to carry the process too far; multiplication or division of the seed by factors less than 2 is seldom warranted.

In most cases, a satisfactory seed value developed by this procedure will remain satisfactory even though changes in the model are introduced—for example, additional stresses, modifications in boundary conditions or changes in the model mesh. However, if convergence problems arise after such changes, the trial and error procedure can be repeated. It should be noted that the more strongly diagonal the coefficient matrix, the less important the choice of seed will be. Thus, source terms such as evapotranspiration or stream seepage, which affect only coefficients on the main diagonal, normally tend to make the choice of seed less critical; and the addition of such terms to a model seldom necessitates modification of the seed.
For each iteration, the program stores the value of $|\Delta h_x|_{\text{max}}$, where $|\Delta h_x|$ refers to the absolute value of the head change computed during iteration $i$ at a given node, and $|\Delta h_x|_{\text{max}}$ is the maximum such absolute value for the entire mesh in that iteration. For the last time step of each stress period, a table of $|\Delta h_x|_{\text{max}}$ values for each iteration of the time step is printed in the program output; at the user's option, a table of these values may be printed for every time step, or for time steps which fall at specified intervals. (See Narrative for Module $\text{SIPIAP}$ and Sample Problem Output in Appendix D.) In addition to the $|\Delta h_x|_{\text{max}}$ value, each entry in the table shows the indices of the node at which the maximum change was recorded, and a sign indicating whether the change was positive or negative. This table can be used in the head change trend evaluations described above, under the assumption that the behavior of the $|\Delta h_x|_{\text{max}}$ values is representative of the behavior of head change throughout the mesh.

Improvements in the rate of convergence can also be obtained by adjusting the acceleration parameter, ACCL. Increases in ACCL will cause increases in the head change at each iteration, while decreases in ACCL will cause decreases in head change. The trial procedure described above can be used for this case as well; however changes in the seed and in ACCL should never be attempted in the same set of trial runs.

It is sometimes necessary to slow the process of convergence in order to prevent cells from converting to the no-flow condition as a result of head overshoot during an iteration. In these situations, optimal convergence cannot be considered convergence in the minimum number of iterations, but rather convergence in the smallest number of iterations that does not involve head overshoot. The procedure of examining head
change per iteration and adjusting iteration parameters can again be used to determine when this condition is being met, and to develop the required seed or ACCL terms.
Strongly Implicit Procedure Package Input

Input to the Strongly Implicit Procedure (SIP) Package is read from the unit specified in IUNIT(9).

FOR EACH SIMULATION

SIPIAL

1. Data: MXITER NPARM
   Format: I10 I10

SIPIRP

2. Data: ACCL HCLOSE IPCALC WSEED IPRSIP
   Format: F10.0 F10.0 I10 F10.0 I10

Explanation of Fields Used in Input Instructions

MXITER—is the maximum number of times through the iteration loop in one time step in an attempt to solve the system of finite-difference equations. Fifty iterations are generally sufficient.

NPARM—is the number of iteration parameters to be used. Five parameters are generally sufficient.

ACCL—is the acceleration parameter. It must be greater than zero and is generally equal to one. If a zero is entered, it is changed to one.

HCLOSE—is the head change criterion for convergence. When the maximum absolute value of head change from all nodes during an iteration is less than or equal to HCLOSE, iteration stops.

IPCALC—is a flag indicating where the iteration parameter seed will come from.
   0 - the seed will be entered by the user.
   1 - the seed will be calculated at the start of the simulation from problem parameters.

WSEED—is the seed for calculating iteration parameters. It is only specified if IPCALC is equal to zero.

IPRSIP—is the printout interval for SIP. If IPRSIP is equal to zero, it is changed to 999. The maximum head change (positive or negative) is printed for each iteration of a time step whenever the time step is an even multiple of IPRSIP. This printout also occurs at the end of each stress period regardless of the value of IPRSIP.

12-30
Module Documentation for the Strongly Implicit Procedure Package

The Strongly Implicit Procedure Package (SIPl) consists of three primary modules and two submodules. They are:

Primary Modules

SIPlAL  Allocates space for SIP work arrays.

SIPlRP  Reads control information needed by the SIP Package and calculates iteration parameters if the seed is specified by the user.

SIPlAP  Performs one iteration of the strongly implicit procedure.

Submodules

SSIPlP  Prints the largest head change for each iteration.

SSIPlI  Calculates iteration parameters when the seed is calculated by the program.
Module SIPlAL allocates space in the X array for SIP arrays. The four
arrays, EL, FL, GL, and V hold intermediate results during the solution
process. Each of these contains one element for each model cell.
Additionally, three arrays, HDCG, LRCH, and W are required. HDCG holds the
maximum head change each iteration and LRCH holds the cell location at
which the maximum occurred. The number of elements in HDCG is MXITER, and
that in LRCH is three times MXITER, where MXITER is the maximum number of
iterations allowed in a time step and is specified by the user. Array W
holds iteration parameters. One element in W is used for each of the NPARM
iteration parameters. (NPARM is specified by the user.)

Module SIPlAL performs its functions in the following order:

1. Print a message identifying the SIP Package.

2. Read and print MXITER and NPARM.

3. Allocate the required space in the X array. The X-array location
   pointer (ISUM) is saved in variable ISOLD prior to allocation so that the
   space required for SIP can be calculated in step 4.

4. Calculate and print the space used in the X array. The space
   used by SIP is ISUM - ISOLD. The total space allocated by all packages so
   far is ISUM - 1.

5. RETURN.
X array is the pool of memory space from which space is allocated for arrays used by various packages.

Flow Chart for Module SIP1AL

1. ENTER SIP1AL

2. PRINT A MESSAGE IDENTIFYING SIP PACKAGE

3. READ AND PRINT MAXIMUM # OF ITERATIONS AND # OF ITERATION PARAMETERS

4. ALLOCATE SPACE FOR SIP ARRAYS IN THE X ARRAY

5. CALCULATE AND PRINT SPACE USED IN THE X ARRAY

RETURN
SUBROUTINE SIPlAL(ISUM, LENX, LCEL, LCFL, LCGL, LCV, LCHDCG, LCLRCH, 
  LCH, MXITER, NPARM, NCOL, NROW, NLAY, IN, IOUT)

C-----VERSION 1110 31DEC1986 SIPlAL
C
C ******************************************************************
C
C1-------PRINT A MESSAGE IDENTIFYING SIP PACKAGE
WRITE(IOUT,1) IN
  1 FORMAT(1HO,'SIPl -- STRONGLY IMPLICIT PROCEDURE SOLUTION PACKAGE'
    , ', VERSION 1, 9/1/87', ' INPUT READ FROM UNIT', I3)
C
C2-------READ AND PRINT MXITER AND NPARM
READ(IN,2) MXITER, NPARM
  2 FORMAT(21101
WRITE(IOUT, MXITER, NPARM
  3 FORMAT(1X,'MAXIMUM OF', I4, ' ITERATIONS ALLOWED FOR CLOSURE'/
    1X, I2, ' ITERATION PARAMETERS')
C
C3------ALLOCATE SPACE FOR THE SIP ARRAYS
ISOLD=ISUM
NRC=NROW*NCOL
ISIZ=NRC*NLAY
LCEL=ISUM
ISUM=ISUM+ISIZ
LCFL=ISUM
ISUM=ISUM+ISIZ
LCGL=ISUM
ISUM=ISUM+ISIZ
LCV=ISUM
ISUM=ISUM+ISIZ
LCHDCG=ISUM
ISUM=ISUM+MXITER
LCLRCH=ISUM
ISUM=ISUM+3*MXITER
LCW=ISUM
ISUM=ISUM+NPARM
C
C4-------CALCULATE AND PRINT THE SPACE USED IN THE X ARRAY
ISP=ISUM-ISOLD
WRITE(IOUT,4) ISP
  4 FORMAT(1X,I8,' ELEMENTS IN X ARRAY ARE USED BY SIP')
ISUM1=ISUM-1
WRITE(IOUT,5) ISUM1, LENX
  5 FORMAT(1X,I8,' ELEMENTS OF X ARRAY USED OUT OF', I8)
  IF(ISUM1.GT.LENX) WRITE(IOUT,6)
  6 FORMAT(1X,' ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C5-------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISIZ</td>
<td>Module</td>
<td>Number of cells in the grid.</td>
</tr>
<tr>
<td>ISOLD</td>
<td>Package</td>
<td>Before this module allocates space, ISOLD is set equal to ISUM. After allocation, ISOLD is subtracted from ISUM to get ISP, the amount of space in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISP</td>
<td>Module</td>
<td>Number of words in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.</td>
</tr>
<tr>
<td>ISUMI</td>
<td>Module</td>
<td>ISUM - 1.</td>
</tr>
<tr>
<td>LCEL</td>
<td>Package</td>
<td>Location in the X array of the first element of array EL.</td>
</tr>
<tr>
<td>LCFL</td>
<td>Package</td>
<td>Location in the X array of the first element of array FL.</td>
</tr>
<tr>
<td>LCGL</td>
<td>Package</td>
<td>Location in the X array of the first element of array GL.</td>
</tr>
<tr>
<td>LCHDCG</td>
<td>Package</td>
<td>Location in the X array of the first element of array HDCG.</td>
</tr>
<tr>
<td>LCLRCH</td>
<td>Package</td>
<td>Location in the X array of the first element of array LRCH.</td>
</tr>
<tr>
<td>LCV</td>
<td>Package</td>
<td>Location in the X array of the first element of array V.</td>
</tr>
<tr>
<td>LCW</td>
<td>Package</td>
<td>Location in the X array of the first element of array W.</td>
</tr>
<tr>
<td>LENX</td>
<td>Global</td>
<td>Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.</td>
</tr>
<tr>
<td>MXITER</td>
<td>Package</td>
<td>Maximum number of iterations.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NPARM</td>
<td>Package</td>
<td>Number of iteration parameters.</td>
</tr>
<tr>
<td>NRC</td>
<td>Module</td>
<td>Number of cells in a layer.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
</tbody>
</table>
Narrative for Module SIPlRP

Module SIPlRP reads data for the SIP package: the acceleration parameter (ACCL), the closure criterion (HCLOSE), the iteration-parameter seed (WSEED), a flag indicating whether WSEED is to be calculated or specified by the user (IPCALC), and the interval for printing head change (IPRSlP). If IPCALC is zero, iteration parameters are calculated using WSEED as the seed. Module SIPlRP performs its functions in the following order:

1. Read the data. If ACCL is zero, substitute a default of 1.0. If IPRSlP is less than or equal to zero, substitute an interval of 999 time steps. The defaults are provided as a convenience to the user.

2. Print the data read in step 1.

3. Check IPCALC which is a flag that indicates the source of the iteration-parameter seed (WSEED).

   (a) If IPCALC is not zero, submodule SSIPlI will calculate a seed and the resulting iteration parameters at the start of the first iteration. Print a message telling of this option.

   (b) If IPCALC is zero, use WSEED to calculate iteration parameters. The i-th iteration parameter \( (I_i) \) is given by the expression

\[
I_i = 1 - (WSEED)^{NPARM-1}.
\]

Print the parameters.

4. RETURN.
ACCL is a multiplier of calculated head change which is used to control the convergence rate.

HCLOSE is the head change closure criterion. When head change in all model cells is less than or equal to HCLOSE, iteration stops.

WSEED is the seed, specified by the user, on which the calculation of iteration parameters is based if IPCALC is zero.

IPRSIP is the time step interval for printing the maximum head change for each iteration of a time step. Head change is printed every IPRSIIP time step. Head change is printed at the end of a stress period regardless of the interval.

IPCALC is a flag. If it is set equal to one, iteration parameters will be calculated from a seed calculated within the program. If it is clear (equal to zero), iteration parameters will be calculated from a seed provided by the user.

READ AND PRINT ACCL, HCLOSE, WSEED, IPCALC, AND IPRSIIP

IS IPCALC 0?

PRINT MESSAGE EXPLAINING THAT ITERATION PARAMETERS WILL BE CALCULATED FROM MODEL CALCULATED SEED

CALCULATE AND PRINT ITERATION PARAMETERS USING WSEED

RETURN
SUBROUTINE SIP1RP(NPARM,MXITER,ACCL,HCLOSE,W,IN,IPCALC,IPRSIP,
                  IOUT)
C-----VERSION 0925 16DEC1982 SIP1RP
C
C *******************************************************
C READ DATA FOR SIP
C *******************************************************
C
C SPECIFICATIONS:
C --------------------------------------------------------
DIMENSION W(NPARM)
C --------------------------------------------------------
C
C1-------READ ACCL,HCLOSE,WSEED,IPCALC,IPRSIP
READ(IN,1) ACCL,HCLOSE,IPCALC,WSEED,IPRSIP
1 FORMAT(2F10.0,I10,F10.0,I10)
   IF(ACCL.EQ.0.) ACCL=1.
C
C2-------PRINT DATA VALUES JUST READ
WRITE(IOUT,100)
100 FORMAT(1HO,///57X,'SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE'
   1/57X,43(' '),1)
   WRITE(IOUT,115) ACCL
   WRITE(IOUT,120) HCLOSE
   WRITE(IOUT,125) ACCL
   WRITE(IOUT,130) IPRSIP
110 FORMAT(1HO,63X,'ACCELERATION PARAMETER =',E15.5)
115 FORMAT(1H0,47X,'MAXIMUM ITERATIONS ALLOWED FOR CLOSURE =',I9)
120 FORMAT(1H0,125) HCLOSE
125 FORMAT(1H0,52X,'HEAD CHANGE CRITERION FOR CLOSURE =',E15.5)
129 FORMAT(1H0,52X,'SIP HEAD CHANGE PRINTOUT INTERVAL =',I9)
   IF(IPRSIP.LE.0) IPRSIP=999
130 FORMAT(1HO,52X,'SIP HEAD CHANGE PRINTOUT INTERVAL =',I9)
   WRITE(IOUT,130) IPRSIP
C
C3-------CHECK IF SPECIFIED VALUE OF WSEED SHOULc BE USED OR IF
C3------SEED SHOULD BE CALCULATED
   IF(IPCALC.EQ.0) GO TO 150
C
C3A------CALCULATE SEED & ITERATION PARAMETERS PRIOR TO 1ST ITERATION
   WRITE(IOUT,140)
   140 FORMAT(1HO,52X,'CALCULATE ITERATION PARAMETERS FROM MODEL,'
   1/57X,43(' '),1)
   WRITE(IOUT,160)
   150 P1=1.
   P2=NPARM-1
   DO 160 I=1,NPARM
      P1=P1+1.
   160 W(I)=1.-WSEED**(P1/P2)
   WRITE(IOUT,161) NPARM,WSEED,(W(J),J=1,NPARM)
   161 FORMAT(1HO,63X,'ITERATION PARAMETERS CALCULATED FROM'
   1/57X,43(' '),1)
   WRITE(IOUT,161) NPARM,WSEED,(W(J),J=1,NPARM)
   'SPECIFIED WSEED =',F11.8,' :'//(10X,6E15.7))
C
C4-------RETURN
1000 RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCL</td>
<td>Package</td>
<td>Acceleration parameter.</td>
</tr>
<tr>
<td>HCLOSE</td>
<td>Package</td>
<td>Closure criterion for the iterative procedure.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for iteration parameters.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPCALC</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>≠ 0, seed for iteration will be calculated in the program.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, seed will be specified by the user.</td>
</tr>
<tr>
<td>IPRSIP</td>
<td>Package</td>
<td>Frequency (in time steps) with which the maximum head changes for each iteration will be printed.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for iteration parameters.</td>
</tr>
<tr>
<td>MXITER</td>
<td>Package</td>
<td>Maximum number of iterations.</td>
</tr>
<tr>
<td>NPARM</td>
<td>Package</td>
<td>Number of iteration parameters.</td>
</tr>
<tr>
<td>P1</td>
<td>Module</td>
<td>I - 1.</td>
</tr>
<tr>
<td>P2</td>
<td>Module</td>
<td>NPARM - 1.</td>
</tr>
<tr>
<td>W</td>
<td>Package</td>
<td>DIMENSION (NPARM), Iteration parameters.</td>
</tr>
<tr>
<td>WSEED</td>
<td>Module</td>
<td>Seed for calculating iteration parameters.</td>
</tr>
</tbody>
</table>
Module SIPlAP performs one iteration of the Strongly Implicit Procedure (SIP) algorithm for solving the flow equation. To save computational time, all arrays are declared one dimensional. The one-dimensional indexes are calculated from the layer, row, and column indexes normally used to access the arrays in three dimensions. Improvement in computational time is achieved because knowledge of the geometry is used to increase computational efficiency, and because calculations are not repeated for identical indices as would be done by internal FORTRAN addressing routines if three-dimensional subscripts were used.

This module is somewhat complex, partly because the SIP solution process requires that the same calculations be performed with two methods of ordering the equations. This is implemented by a generalized algorithm that uses the same computer statements to handle both ordering schemes. Checks are made to detect which ordering scheme is used, and array indexes are calculated accordingly.

Double precision is used for most calculations in this module in order to allow accurate answers to be calculated for a wide range of problems. Mixed precision arithmetic has been avoided by setting double-precision variables equal to single-precision values and then using the double-precision variables to generate fully double-precision expressions, and where necessary by doing the reverse to generate fully single-precision expressions.
In the explanations below, no attempt has been made to discuss each use of an assignment statement to change precision, because of the large amount of text this would require. However, when changing this module, care should be used to maintain expressions that have unmixed precision, as mixed precision expressions can cause erroneous results with some compilers.

Module SIPIAP performs its functions in the following order:

1. If the user has specified \((\text{IPCALC} \neq 0)\) that iteration parameters should be calculated by the program, CALL submodule SSIP1I to calculate both the seed and the parameters.

2. Assign values to fields that are constant during an iteration.

3. Initialize the variables that track maximum head change curing an iteration.

4. Clear SIP work arrays.

5. Determine the ordering of equations and set the ordering flag \((\text{IDIR})\) accordingly. This flag alternates between 1 and -1 each iteration. Calculate indexes \(\text{IDNRC}\) and \(\text{IDNCOL}\) which are used when calculating locations of neighboring cells.

6. Calculate the matrix \([U]\) and intermediate vector \([V]\) using forward substitution. The elements in matrix \([L]\) are used as they are calculated; therefore, they are not saved. In the explanation of SIP concepts, the diagonals in the matrix \([U]\) were designated "e," "f," and "g." The corresponding field names in the program are EL (e lower case), FL (f lower case),...
and GL (g lower case). Similarly, the diagonals in the [L] array which are "a", "b", "c", and "d" in the explanation are "AL", "BL", "CL", and "DL" in the program. The codes for the diagonals in matrix [A] in the explanation are the same in the program. The codes for diagonals in [A+B] in the explanation are followed by a "P" in the program. Hence, Z' in the explanation is ZP (Z prime) in the program. The intermediate vector \{V\} in the explanation is the array "V" in the program.

(a) Set current cell indexes, II, JJ, KK. For normal ordering, the equation order is the same as the order of the loop indexes I,J,K. For reverse ordering, loop indexes I and K are inverted to produce the proper sequence of cells.

(b) Calculate the one-dimensional subscript of the current cell. If this cell is constant head or no flow, skip calculations for this cell and go on to the next.

(c) Calculate the one-dimensional subscripts for the six neighboring cells.

(d) Calculate the one-dimensional subscripts for conductance to each of the six neighboring cells. Since conductances between cells are assigned to array elements at specific cells (for example, CR(I,J,K) stores conductance between cells I,J,K and I,J+1,K), the four or five conductance subscripts are not simply the cell locations of the six neighboring cells as calculated in step 6(c). Also, the subscripts depend on equation ordering.
(e) Calculate or assign variables that are required for forward substitution and involve neighboring cells. Whenever a neighboring cell is outside of the grid, the variables are set to zero.

(1) Neighboring cell is one row back.
(2) Neighboring cell is one row ahead.
(3) Neighboring cell is one column back.
(4) Neighboring cell is one column ahead.
(5) Neighboring cell is one layer back.
(6) Neighboring cell is one layer ahead.

(f) Calculate the components of the upper and lower triangular matrices \([U]\) and \([L]\), which are the factors of matrix \([A+B]\).

(g) Calculate the residual \([\text{RES}]\). The calculation of HNW times HCOF is done in single precision so that the calculation will have precision comparable to similar calculations made in the formulation modules, all of which use single precision.

(h) Calculate the intermediate vector \([V]\), which is stored in array \(V\). This step completes the forward-substitution process for one cell.

7. Step through the cells solving for head change using back substitution.

(a) Set current cell indexes \(II, JJ, KK\). The ordering is the reverse of that used for forward substitution (step 6(a)).

(b) Calculate the one-dimensional subscript of the current cell. If this cell is constant head or no flow, skip calculations for this cell and go to the next.
(c) Calculate the one-dimensional subscripts for the three neighboring cells behind (relative to the direction of the back-substitution ordering) the current cell.

(d) Back substitute, solving for head change. Store head change in array \( V \) in place of the intermediate values of vector \( \{V\} \). This doubling up of storage is used to save the cost of additional computer storage.

(e) Save the value of head change whose absolute value is largest during this iteration. Also, save the cell location where this head change occurred and the absolute value of the head change.

(f) Add the head change this iteration to head from the previous iteration to get a new estimate of head.

8. Store the head change whose absolute value is greatest this iteration and its cell location in arrays HDCG and LRCH. These may be printed in step 10 at the end of the time step. Set the convergence flag to one if the convergence criterion is met.

9. If the iteration is complete, print the number of iterations for the step; otherwise, RETURN.

10. Print the maximum head change and cell location each iteration if the SIP printout interval (IPRSIP) is reached. Printout occurs at the end of a stress period regardless of the interval.

11. RETURN.
IPCALC is a flag. If it is set equal to one, the program calculates a seed from which iteration parameters are calculated. It may be set by the user at the beginning of the simulation. It is cleared during the first iterations. SSIPII will never be called more than once. If IPCALC is not set equal to zero, the user specifies the seed for the iteration parameters.

IDIR indicates whether the ordering of equations is normal (1) or reverse (-1).

SSIPII is a submodule which calculates iteration parameters.
Flow Chart for Module SIPIAP (Continued)

Single Cell Index: In this module, a single index is used to identify each cell. This is in opposition to the three indices (I,J,K) used in most other modules.
SUBROUTINE SIP1AP(HNEW,IBOUND,CR,CC,CV,ROOF,RS,CL,FL,QL,V,
  1  W,HDCG,LRCH,NPAR,M,KITER,HOLD,ACCI,INWG,KSTP,KPER,
  2  IPCALC,IPRMIP,NSTP,M,NROW,NLAY,NODES,INOUT)
C-----VERSION 1056 24JUL1987 SIP1AP
C
C **********I**************~********************~*****~*******u***~***

C
C SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE -- 1 ITERATION
C
C SPECIFICATIONS:
C
DOUBLE PRECISION HNEW,DITPAR,AC,HHNCF,RH N,X1,DZERO,DONE,RES
DOUBLE PRECISION Z,B,E,F,H,S,AP,TP,CP,UP,RP
DOUBLE PRECISION ZHNEW,BHNEW,HHNEW,FINW,HNEW,SHNEW
DOUBLE PRECISION AEL,CL,CL,ELNCL,FL,ELNCL,ELNCL
DOUBLE PRECISION ELNLR,FLNLR,FLNL,FLNN,FLNLL,FLNLL
DOUBLE PRECISION VNRL,VLNRL,FLXI,FLXI,FLXI,FLXI

DIMENSION HNEW(NODES), IBOUND(NODES), CR(NODES), CC(NODES),
  1 CV(NODES), ROOF(NODES), RS(NODES), EL(NODES), FL(NODES),
  2 GL(NODES), VNRL(NODES), VNRL(NODES), FLXI, VNRL, VNRL

C
C1-------CALCULATE ITERATION PARAMETERS IF FLAG IS SET. THEN
C1-------CLEAR THE FLAG SO THAT CALCULATION IS DONE ONLY ONCE.
C1 IF(IPCALC.NE.0)
C1 CALL SSIPlI(CR,CC,CV,IBOUND,NPAR,M,NROW,NLAY,NODES,NOUT)
C2 IPCALC=0

C2-------ASSIGN VALUES TO FIELDS THAT ARE CONSTANT DURING AN ITERATION
DZERO=0.
DONE=1.
AC=ACU
NRC=NROW*NCOL
NTH=MOD(KITER-1,NPARM)+1
DITPAR=W(NTH)

C3-------INITIALIZE VARIABLE THAT TRACKS MAXIMUM HEAD CHANGE DURING
C3-------THE ITERATION
BIGG=0.

C4-------CLEAR SIP WORK ARRAYS.
DO 100 I=1,NODES
  100 EL(I)=0.
  FL(I)=0.
  GL(I)=0.
  V(I)=0.

C5-------SET NORMAL/REVERSE EQUATION ORDERING FLAG (1 OR -1) AND
C5-------CALCULATE INDEXES DEPENDENT ON ORDERING
IDIR=1
IF(MOD(KITER,2).EQ.0)IDIR=-1
INWG=IDIR*NRC
INDCOL=IDIR*NCOL

C6-------STEP THROUGH CELLS CALCULATING INTERMEDIATE VECTOR V
C6-------USING FORWARD SUBSTITUTION
DO 150 K=1,NLAY
  DO 150 I=1,NROW
  DO 150 J=1,NCOL

C6A-------SET UP CURRENT CELL LOCATION INDEXES. THESE ARE DEPENDENT
C6A-------ON THE DIRECTION OF EQUATION ORDERING.
IF(IDIR.LE.0)GO TO 120
II=I
J=J
K=K
GO TO 122
120 II=NROW-I+1
J=J
K=NLAY-K+1

12-48
CALCULATE 1 DIMENSIONAL SUBSCRIPT OF CURRENT CELL AND
SKIP CALCULATIONS IF CELL IS NOFLOW OR CONSTANT HEAD

\[ \text{IF}(\text{BOUND}(N).	ext{LE}.0) \text{GO TO 150} \]

CALCULATE 1 DIMENSIONAL SUBSCRIPTS FOR LOCATING THE 6 SURROUNDING CELLS

\[ \begin{align*}
\text{NRN} & = N + \text{IDNCOL} \\
\text{NRH} & = N + 1 \\
\text{NRB} & = N \\
\text{NLN} & = N + \text{IDNRC} \\
\text{NLL} & = N + 1 \\
\text{NLZ} & = N \\
\end{align*} \]

GO TO 126

CALCULATE 1 DIMENSIONAL SUBSCRIPTS FOR CONDUCTANCE TO THE 6 SURROUNDING CELLS. THESE DEPEND ON ORDERING OF EQUATIONS.

\[ \text{IF}(\text{IDIR} . \text{LE}.0) \text{GO TO 124} \]

ASSIGN VARIABLES IN MATRICES A & U INVOLVING ADJACENT CELLS

\[ \text{IF}(1.	ext{EQ}.1) \text{ GO TO 128} \]

\[ \begin{align*}
\text{B} & = \text{CC}(	ext{NRB}) \\
\text{ELNR} & = \text{EL}(\text{NR}) \\
\text{FLNR} & = \text{FL}(\text{NR}) \\
\text{GLNR} & = \text{GL}(\text{NR}) \\
\text{BHNEW} & = B + \text{HNEW} (\text{NR}) \\
\text{VNRL} & = \text{V} (\text{NR}) \\
\end{align*} \]

\[ \text{IF}(1.	ext{EQ}.1) \text{ GO TO 132} \]

\[ \begin{align*}
\text{D} & = \text{CC} (\text{NCD}) \\
\text{ELNC} & = \text{EL} (\text{NU}) \\
\text{FLNC} & = \text{FL} (\text{NU}) \\
\text{GHNC} & = \text{GL} (\text{NU}) \\
\text{DHNEW} & = D + \text{HNEW} (\text{NCL}) \\
\text{VNC} & = \text{V} (\text{NCL}) \\
\end{align*} \]

\[ \text{IF}(1.	ext{EQ}.1) \text{ GO TO 132} \]

\[ \begin{align*}
\text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]

\[ \text{IF}(\text{NR} . \text{EQ} . \text{NRW}) \text{ GO TO 130} \\
\text{H} & = \text{CC} (\text{NRW}) \\
\text{HHNEW} & = H * \text{HNEW} (\text{NRN}) \\
\end{align*} \]
C
C6E4----NEIGHBOR IS 1 COLUMN AHEAD
132 F=DZERO
   FHNEW=DZERO
   IF(J.EQ.NCOL) GO TO 134
   F=CR(NCF)
   FHNEW=F*HNEW(NCN)
C
C6E5----NEIGHBOR IS 1 LAYER BEHIND
134 Z=DZERO
   FLNL=DZERO
   GLNL=DZERO
   ZHNEW=DZERO
   VINL=DZERO
   IF(K.EQ.1) GO TO 136
   Z=CV(NLZ)
   FLNL=FL(NLL)
   GLNL=GL(NLL)
   ZHNEW=Z*HNEW(NLL)
   VINL=V(NLL)
C
C6E6----NEIGHBOR IS 1 LAYER AHEAD
136 S=DZERO
   SHNEW=DZERO
   IF(K.EQ.NLAY) GO TO 138
   S=CV(NLS)
   SHNEW=S*HNEW(NLN)
C
C6E7----CALCULATE THE NEGATIVE SUM OF ALL CONDUCTANCES TO NEIGHBORING CELLS
136 E=-Z-B-D-F-H-S
C
C6F-----CALCULATE COMPONENTS OF THE UPPER AND LOWER MATRICES, WHICH ARE THE FACTORS OF MATRIX (A+B)
   AL=Z/(DONE+DITPAR*(ELNLL+FLNLL))
   BL=B/(DONE+DITPAR*(ELNR+GLNRL))
   CL=D/(DONE+DITPAR*(FLNCL+GLNCL))
   AP=AL*ELNLL
   CP=BL*ELNR
   GP=CL*GLNRL
   TP=AL*FLNLL
   UP=BL*FLNRL
   HHCOF=HHCOF(N)
   DL=E+HHCOF+DITPAR*(AP+TP+CP+GP+RP)-AL*GLNLL-BL*FLNRL-CL*FLNCL
   EL(N)=(F-DITPAR*(AP+CP))/DL
   FL(N)=(H-DITPAR*(TP+GP))/DL
   GL(N)=(S-DITPAR*(BL+RP))/DL
C
C6G-----CALCULATE THE RESIDUAL
   RRHS=RHS(N)
   HNW=HNEW(N)
   HCQFHM=HCQF(N)
   RES=RRHS-ZHNEW-BHNEW-DHNEW-EHNEW(N)-HCQFHM-FHNEW-HNEW-SHNEW
C
C6H-----CALCULATE THE INTERMEDIATE VECTOR V
   V(N)=(AC*RES-AL*VNLL-BL*VNR-CL*VNU)/DL
C
C150 CONTINUE
C
C7------STEP THROUGH EACH CELL AND SOLVE FOR HEAD CHANGE BY BACK
C7------SUBSTITUTION
   DO 160 K=1,NLAY
   DO 160 I=1,NROW
   DO 160 J=1,NCOL
160 CONTINUE
C7A-----SET UP CURRENT CELL LOCATION INDEXES. THESE ARE DEPENDENT
C7A-----ON THE DIRECTION OF EQUATION ORDERING.
IF(IDIR.LT.0) GO TO 152
KK=NLAY+K-I
II=NRW+I-I
JJ=NCOL+J-J+1
GO TO 154
152 KK=K
II=I
JJ=NCOL-J+1
C
C7B------CALCULATE 1 DIMENSIONAL SUBSCRIPT OF CURRENT CELL AND
C7B------SKIP CALCULATIONS IF CELL IS NOFLOW OR CONSTANT HEAD
154 N=J+(I-I)*NCOL+(K-K-1)*MNRC
IF(IBOUND(N).LE.0) GO TO 160
C
C7C------CALCULATE 1 DIMENSIONAL SUBSCRIPTS FOR THE 3 NEIGHBORING CELLS
C7C------BEHIND (RELATIVE TO THE DIRECTION OF THE BACK SUBSTITUTION ORDERING) THE CURRENT CELL.
NC=N+1
NR=N+IDNRC
NL=N+IDNRC
C
C7D------BACK SUBSTITUTE, STORING HEAD CHANGE IN ARRAY V IN PLACE OF
C7D------INTERMEDIATE FORWARD SUBSTITUTION VALUES.
ELXI=DZERO
FLXI=DZERO
GLXI=DZERO
IF(JJ .NE. NCOL) ELXI=EL(N)*V(NC)
IF(K .NE. 1) FLXI=FL(N)*V(N)
IF(JJ .NE. 1) GLXI=GL(N)*V(NL)
W=V(N)
Y=N-W-ELXI-FLXI-GLXI
C
C7E------GET THE ABSOLUTE HEAD CHANGE, IF IT IS MAX OVER GRID SO FAR.
C7E------THEN SAVE IT ALONG WITH CELL INDICES AND HEAD CHANGE.
TCHK=ABS(V(N))
IF (TCHK.LE.BIGG) GO TO 155
BIGG=TCHK
BIG=V(N)
IB=II
JB=JJ
KGB=KK
C
C7F------ADD HEAD CHANGE THIS ITERATION TO HEAD FROM THE PREVIOUS ITERATION TO GET A NEW ESTIMATE OF HEAD.
155 XI=V(N)
HNEW(N)=HNEW(N)+XI
C
C160 CONTINUE
C
C2------STORE THE LARGEST ABSOLUTE HEAD CHANGE (THIS ITERATION) AND
C2------AND ITS LOCATION.
HDG(KITER)=BIG
LRCH(1,KITER)=KB
LRCH(2,KITER)=IB
LRCH(3,KITER)=JB
ICNVG=0
IF(BIGG.LE.HCLOSE) ICNVG=1
C
C3------END OF TIME STEP, PRINT # OF ITERATIONS THIS STEP
IF(ICNVG.EQ.0 .AND. KITER.NE.MXITER) GO TO 600
IF(KSTP.EQ.1) WRITE(IOUT,500)
500 FORMAT(1HD)
WRITE(IOUT,501) KITER,KSTP,KPER
501 FORMAT(1X.IS,' ITERATIONS FOR TIME STEP',I4,' IN STRESS PERIOD',I4)
C
C10------PRINT HEAD CHANGE EACH ITERATION IF PRINTOUT INTERVAL IS REACHED
IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP .OR. MOD(KSTP,IPRSIP).EQ.0)
1 CALL SSIP1P(HDGG,LRCH,KITER,1OUT)
C
C11------RETURN
600 RETURN
C
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC</td>
<td>Module</td>
<td>Double-precision acceleration parameter (ACCL).</td>
</tr>
<tr>
<td>ACCL</td>
<td>Package</td>
<td>Acceleration parameter.</td>
</tr>
<tr>
<td>AL</td>
<td>Module</td>
<td>Diagonal from the lower factor. (AL stands for A-lower case.)</td>
</tr>
<tr>
<td>AP</td>
<td>Module</td>
<td>Diagonal element in the modified coefficient matrix. (AP stands for A-prime.)</td>
</tr>
<tr>
<td>B</td>
<td>Module</td>
<td>Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the last row. (BL stands for B-lower case.)</td>
</tr>
<tr>
<td>BHNEW</td>
<td>Module</td>
<td>Head in the adjacent cell which is in the last row.</td>
</tr>
<tr>
<td>BIG</td>
<td>Module</td>
<td>Largest head change for an iteration.</td>
</tr>
<tr>
<td>BIGG</td>
<td>Module</td>
<td>Largest absolute value of head change for an iteration.</td>
</tr>
<tr>
<td>BL</td>
<td>Module</td>
<td>Diagonal from the lower factor. (BL stands for B-lower case.)</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CL</td>
<td>Module</td>
<td>Diagonal from the lower factor. (CL stands for C-lower case.)</td>
</tr>
<tr>
<td>CP</td>
<td>Module</td>
<td>Diagonal element in the modified coefficient matrix. (CP stands for C-prime.)</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>D</td>
<td>Module</td>
<td>Diagonal label in the coefficient matrix. Conductance from the adjacent node which is in the last column.</td>
</tr>
<tr>
<td>DONE</td>
<td>Module</td>
<td>Double-precision field containing a one.</td>
</tr>
<tr>
<td>DZERO</td>
<td>Module</td>
<td>Double-precision field containing a zero.</td>
</tr>
<tr>
<td>E</td>
<td>Module</td>
<td>Main diagonal in the coefficient matrix.</td>
</tr>
<tr>
<td>EL</td>
<td>Module</td>
<td>DIMENSION (NODES), Diagonal from the upper factor. (EL stands for E-lower case.)</td>
</tr>
<tr>
<td>ELNCL</td>
<td>Module</td>
<td>EL (E-lower case) from the cell in the last column.</td>
</tr>
<tr>
<td>ELNLL</td>
<td>Module</td>
<td>EL (E-lower case) from the cell in the last layer.</td>
</tr>
<tr>
<td>ELNRL</td>
<td>Module</td>
<td>EL (E-lower case) from the cell in the last row.</td>
</tr>
<tr>
<td>ELXI</td>
<td>Module</td>
<td>Intermediate result.</td>
</tr>
<tr>
<td>F</td>
<td>Module</td>
<td>Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the next column.</td>
</tr>
<tr>
<td>FHNEW</td>
<td>Module</td>
<td>Head in the adjacent cell which is in the next column.</td>
</tr>
<tr>
<td>FL</td>
<td>Module</td>
<td>DIMENSION (NODES), Diagonal from the upper factor. (FL stands for F-lower case.)</td>
</tr>
<tr>
<td>FLNCL</td>
<td>Module</td>
<td>FL (F-lower case) from the cell in the last column.</td>
</tr>
<tr>
<td>FLNLL</td>
<td>Module</td>
<td>FL (F-lower case) from the cell in the last layer.</td>
</tr>
<tr>
<td>FLNRL</td>
<td>Module</td>
<td>FL (F-lower case) from the cell in the last row.</td>
</tr>
<tr>
<td>Variable</td>
<td>Range</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>------------</td>
</tr>
<tr>
<td>FLXI</td>
<td>Module</td>
<td>Intermediate result.</td>
</tr>
<tr>
<td>GL</td>
<td>Module</td>
<td>DIMENSION (NODES), Diagonal from the upper factor. (GL stands for G-lower case.)</td>
</tr>
<tr>
<td>GLNCL</td>
<td>Module</td>
<td>GL (G-lower case) from the cell in the last column.</td>
</tr>
<tr>
<td>GLNLL</td>
<td>Module</td>
<td>GL (G-lower case) from the cell in the last layer.</td>
</tr>
<tr>
<td>GLNRL</td>
<td>Module</td>
<td>GL (G-lower case) from the cell in the last row.</td>
</tr>
<tr>
<td>GLXI</td>
<td>Module</td>
<td>Intermediate result.</td>
</tr>
<tr>
<td>GP</td>
<td>Module</td>
<td>Diagonal element in the modified coefficient matrix. (GP stands for G-prime.)</td>
</tr>
<tr>
<td>H</td>
<td>Module</td>
<td>Diagonal label in the coefficient matrix. Conductance from the adjacent node which is in the next row.</td>
</tr>
<tr>
<td>HCFHNW</td>
<td>Module</td>
<td>Product of head and HCOF for a cell.</td>
</tr>
<tr>
<td>HCLOSE</td>
<td>Package</td>
<td>Closure criterion for the iterative procedure.</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K) in the finite-difference equation. (GP stands for G-prime.)</td>
</tr>
<tr>
<td>HDCG</td>
<td>Package</td>
<td>DIMENSION (MXITER), Maximum head change for each iteration.</td>
</tr>
<tr>
<td>HHCOF</td>
<td>Module</td>
<td>Double-precision HCOF.</td>
</tr>
<tr>
<td>HHNEW</td>
<td>Module</td>
<td>Head in the adjacent cell which is in the next row.</td>
</tr>
<tr>
<td>HNEW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>HNW</td>
<td>Module</td>
<td>Temporary field for HNEW(N).</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for nodes and rows.</td>
</tr>
<tr>
<td>IB</td>
<td>Module</td>
<td>Row number of the cell having the largest head change.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
<tr>
<td>ICNVG</td>
<td>Global</td>
<td>Flag is set equal to one when the iteration procedure has converged.</td>
</tr>
<tr>
<td>IDIR</td>
<td>Module</td>
<td>Indicator for direction of solution algorithm.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+1 - forward</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1 - reverse</td>
</tr>
<tr>
<td>IDNCOL</td>
<td>Module</td>
<td>Intermediate result used to calculate indices.</td>
</tr>
<tr>
<td>IDNRC</td>
<td>Module</td>
<td>Intermediate result used to calculate indices.</td>
</tr>
<tr>
<td>II</td>
<td>Module</td>
<td>Row number.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPCALC</td>
<td>Package</td>
<td>Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, iteration parameter seed (WSEED) is entered by the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 1, seed is calculated in the program.</td>
</tr>
<tr>
<td>IPRSIP</td>
<td>Package</td>
<td>Frequency (in time steps) with which the maximum head changes for each iteration will be printed.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>JB</td>
<td>Module</td>
<td>Column number of the cell having the largest head change.</td>
</tr>
<tr>
<td>JJ</td>
<td>Module</td>
<td>Column index.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KB</td>
<td>Module</td>
<td>Layer of the cell having the largest head change.</td>
</tr>
<tr>
<td>KITER</td>
<td>Global</td>
<td>Iteration counter. Reset at the start of each time step.</td>
</tr>
</tbody>
</table>
### List of Variables for Module SIPlAP (Continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>KK</td>
<td>Module</td>
<td>Layer index.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>LRCH</td>
<td>Package</td>
<td>DIMENSION (MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.</td>
</tr>
<tr>
<td>MXITER</td>
<td>Package</td>
<td>Maximum number of iterations.</td>
</tr>
<tr>
<td>N</td>
<td>Module</td>
<td>Cell index.</td>
</tr>
<tr>
<td>NC</td>
<td>Module</td>
<td>Index for the adjacent cell in the last column.</td>
</tr>
<tr>
<td>NCU</td>
<td>Module</td>
<td>One-dimensional subscript of conductance to the adjacent cell which is in the last column.</td>
</tr>
<tr>
<td>NCF</td>
<td>Module</td>
<td>One-dimensional subscript of conductance to the adjacent cell which is in the next column.</td>
</tr>
<tr>
<td>NCL</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the last column.</td>
</tr>
<tr>
<td>NCN</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the next column.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NL</td>
<td>Module</td>
<td>Index for the adjacent cell in the last layer.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NLL</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the last layer.</td>
</tr>
<tr>
<td>NLN</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the next layer.</td>
</tr>
<tr>
<td>NLS</td>
<td>Module</td>
<td>One-dimensional subscript of conductance to the adjacent cell which is in the next layer.</td>
</tr>
<tr>
<td>NLZ</td>
<td>Module</td>
<td>One-dimensional subscript of conductance to the adjacent cell which is in the last layer.</td>
</tr>
<tr>
<td>NODES</td>
<td>Global</td>
<td>Number of cells (nodes) in the finite-difference grid.</td>
</tr>
<tr>
<td>NPARM</td>
<td>Package</td>
<td>Number of iteration parameters.</td>
</tr>
<tr>
<td>NR</td>
<td>Module</td>
<td>Index for the adjacent cell in the last row.</td>
</tr>
<tr>
<td>NRB</td>
<td>Module</td>
<td>One-dimensional subscript of conductance to the adjacent cell which is in the last row.</td>
</tr>
<tr>
<td>NRL</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the last row.</td>
</tr>
<tr>
<td>NRN</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the next row.</td>
</tr>
<tr>
<td>NRL</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the next row.</td>
</tr>
<tr>
<td>NRL</td>
<td>Module</td>
<td>One-dimensional subscript of the cell index of the adjacent cell which is in the next row.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>NSTP</td>
<td>Global</td>
<td>Number of time steps in the current stress period.</td>
</tr>
<tr>
<td>NTH</td>
<td>Module</td>
<td>Index for iteration parameters.</td>
</tr>
<tr>
<td>RES</td>
<td>Module</td>
<td>Residual.</td>
</tr>
<tr>
<td>RHS</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.</td>
</tr>
</tbody>
</table>
### List of Variables for Module SIP1AP (Continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RP</td>
<td>Module</td>
<td>Diagonal element in the modified coefficient matrix. (RP stands for R-prime.)</td>
</tr>
<tr>
<td>RRHS</td>
<td>Module</td>
<td>Double-precision right hand side of the equation.</td>
</tr>
<tr>
<td>S</td>
<td>Module</td>
<td>Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the next layer.</td>
</tr>
<tr>
<td>TCHK</td>
<td>Module</td>
<td>Absolute value of head change for a single cell.</td>
</tr>
<tr>
<td>TP</td>
<td>Module</td>
<td>Diagonal element in the modified coefficient matrix. (TP stands for T-prime.)</td>
</tr>
<tr>
<td>UP</td>
<td>Module</td>
<td>Diagonal element in the modified coefficient matrix. (UP stands for U-prime.)</td>
</tr>
<tr>
<td>V</td>
<td>Package</td>
<td>DIMENSION (NODES), Intermediate result.</td>
</tr>
<tr>
<td>VN</td>
<td>Module</td>
<td>Temporary double-precision V(N).</td>
</tr>
<tr>
<td>VNCL</td>
<td>Module</td>
<td>Element in the intermediate vector for the cell in the last column.</td>
</tr>
<tr>
<td>VNLL</td>
<td>Module</td>
<td>Element in the intermediate vector for the cell in the last row.</td>
</tr>
<tr>
<td>VNRL</td>
<td>Module</td>
<td>Element in the intermediate vector for the cell in the last row.</td>
</tr>
<tr>
<td>W</td>
<td>Package</td>
<td>DIMENSION (NPARM), Iteration parameters.</td>
</tr>
<tr>
<td>XI</td>
<td>Module</td>
<td>Double-precision V(N).</td>
</tr>
<tr>
<td>Z</td>
<td>Module</td>
<td>Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the last layer.</td>
</tr>
<tr>
<td>ZHNEW</td>
<td>Module</td>
<td>Head in the adjacent cell which is in the last layer.</td>
</tr>
</tbody>
</table>
Submodule SSIP1P prints the largest value of head change (HDCG) out of all cells for each iteration of a time step. Also printed is the cell location (LRCH) where the change occurs. The submodule is so short that no numbered comments are used and no flow chart is provided.
SUBROUTINE SSIPlP(HDCG, LRCH, KITER, MXITER, IOUT)

C -----VERSION 1636 24JUL1987 SSIPlP
C
C PRINT MAXIMUM HEAD CHANGE FOR EACH ITERATION DURING A TIME STEP
C
SPECIFICATIONS:
-----------------------------------------------
C
DIMENSION HDCG(MXITER), LRCH(3, MXITER)
-----------------------------------------------
C
WRITE(IOUT,5)
5 FORMAT(1HO,'MAXIMUM HEAD CHANGE FOR EACH ITERATION:/'
  1HO,5('  HEAD CHANGE LAYER,ROW,COL')/lX,132('-'))
WRITE (IOUT,10) (HDCG(J),(LRCH(I,J),I=lr3),J=lrKITER)
10 FORMAT((lx,5(i12.4,' ~',13,',',13,',',13,'~')))
WRITE(IOUT,11)
11 FORMAT(lHO)
C
RETURN
C
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDCG</td>
<td>Package</td>
<td>DIMENSION (MXITER), Maximum head change for each iteration.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for cell location.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for iterations.</td>
</tr>
<tr>
<td>KITER</td>
<td>Global</td>
<td>Iteration counter. Reset at the start of each time step.</td>
</tr>
<tr>
<td>LRCH</td>
<td>Package</td>
<td>DIMENSION (MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.</td>
</tr>
<tr>
<td>MXITER</td>
<td>Package</td>
<td>Maximum number of iterations.</td>
</tr>
</tbody>
</table>
Submodule SSIP1I calculates an iteration-parameter seed using model-conductance values and grid dimensions. Although a single seed is required, the method of calculation requires that three-directional seeds be calculated for each active cell. Then a cell seed, the minimum of the three, is selected. Finally, all the cell seeds are averaged to give the grid seed. This grid seed is then used to calculate the iteration parameters. The minimum cell seed is also printed.

Submodule SSIP1I performs its functions in the following order:

1. Calculate constants and initialize variables. In order to calculate the average cell seed, accumulators AVGSUM (sum of the cell seeds) and NODES (sum of the active cells for which a seed is calculated) are required. These are initialized to zero. WMINMN is used to store the smallest cell seed. Since this value must always be less than one, it is initialized to 1.0. The three coefficients, CCOL, CROW, and CLAY are set equal to $\pi^2/2(NCOL)^2$, $\pi^2/2(NROW)^2$, and $\pi^2/2(NLAY)^2$, respectively.

2. Loop through all cells, calculating a cell seed for each active cell.

   (a) Find the conductances from the cell to each of the six adjacent cells. Conductance across the grid boundary is set equal to zero.

   (b) Find the maximum and minimum of the two conductances in the row direction (DFMX, DFMN), in the column direction (BHMX, BHMN), and in the vertical direction (ZSMX, ZSMN). If the minimum is zero (which indicates that a neighbor is no flow), set the minimum equal to the maximum.
(c) Calculate three-directional seeds (WCOL, WROW, WLAY) using the relations

\[
WCOL = \frac{CCOL}{1 + \frac{(BHMX + ZSMX)}{DFMN}}; \\
WROW = \frac{CROW}{1 + \frac{(DFMX + ZSMX)}{BHMN}}; \text{ and} \\
WLAY = \frac{CLAY}{1 + \frac{(DFMX + BHMX)}{ZSMN}}.
\]

If the minimum conductance is zero (that is, both the minimum and the maximum are zero), set the seed equal to 1.0. This value will be ignored when the cell seed (the minimum-directional seed) is selected in step 2(d) because any valid seed will be less than 1.0.

(d) Select the minimum of the three-directional seeds as the cell seed. If it is the smallest cell seed used so far, store it in WMINMN. Accumulate the sum of the cell seeds and the total number of active cells so that the average of all cell seeds can be calculated in step 3.

3. Calculate the grid seed (the average cell seed) and print it along with the minimum seed.

4. Calculate and print iteration parameters using the grid seed with the relation

\[
I_i = 1 - \frac{1}{(SEED)^{NPARM - 1}}
\]

where

\( I_i \) is the \( i \)-th iteration parameter, and

\( NPARM \) is the number of iteration parameters.

5. RETURN.
Seed: the "grid seed" is the single parameter used to calculate the iteration parameters. To calculate the grid seed, several intermediate variables, called "cell seeds," are used. For each cell, three "directional seeds" are calculated. The minimum directional seed for a cell is the "cell seed." The "grid seed" is the average of the cell seeds.

AVGSUM is an accumulator to which each cell seed is added. It is then divided by the number of cells to obtain the average cell seed which is used as the grid seed.
SUBROUTINE SSIP1I(CR, CC, CV, IBOUND, NPARM, W, NCOL, NROW, NLAY, 
1 IOUT)
C
C------VERSION 1417 12MAY1987 SSIP1I
C ********************************************************************************
C CALCULATE AN ITERATION PARAMETER SEED AND USE IT TO CALCULATE SIP
C ITERATION PARAMETERS
C ********************************************************************************
C
SPECIFICATIONS:
--------------------------------------------------------------------------------
DIMENSION CR(NCOL,NROW,NLAY),CC(NCOL,NROW,NLAY),CV(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),W(NPARM)

DOUBLE PRECISION DWMIN, AVGSUM
--------------------------------------------------------------------------------
C1------CALCULATE CONSTANTS AND INITIALIZE VARIABLES
      
PIEPIE=9.869604
R=NROW
C=NCOL
ZL=NLAY
CCOL=PIEPIE/(Z*C*C)
CROW=PIEPIE/(Z*R*R)
CLAY=PIEPIE/(Z*ZL*ZL)
WMINMN=1.
AVGSUM=0.
NODES=0

C
C2------LOOP THROUGH ALL CELLS, CALCULATING A SEED FOR EACH CELL
C2------THAT IS ACTIVE
DO 100 K=1,NLAY
DO 100 I=1,NROW
DO 100 J=1,NCOL
IF(IBOUND(J,I,K).LE.0) GO TO 100

C2A------CONDUCTANCE FROM THIS CELL
C2A------TO EACH OF THE 6 ADJACENT CELLS
   D=0.
   IF(J.NE.1) D=CR(J-1,I,K)
   F=0.
   IF(J.NE.NCOL) F=CR(J,I,K)
   B=0.
   IF(I.NE.1) B=CC(J,I-1,K)
   H=0.
   IF(I.NE.NROW) H=CC(J,I,K)
   Z=0.
   IF(K.NE.1) Z=CV(J,I,K-1)
   S=0.
   IF(K.NE.NLAY) S=CV(J,I,K)

C2B------FIND THE MAXIMUM AND MINIMUM OF THE 2 CONDUCTANCE COEFFICIENTS
C2B------IN EACH PRINCIPAL COORDINATE DIRECTION
DFMX = AMAX1(D, F)
BHMX = AMAX1(B, H)
ZSMX = AMAX1(Z, S)
DFMN = AMIN1(D, F)
BHMN = AMIN1(B, H)
ZSMN = AMIN1(Z, S)

IF(DFMN.EQ.0.) DFMN = DFMX
IF(BHMN.EQ.0.) BHMN = BHMX
IF(ZSMN.EQ.0.) ZSMN = ZSMX

C2C------CALCULATE A SEED IN EACH PRINCIPAL COORDINATE DIRECTION
WCOL = 1.
IF(DFMN.NE.0.) WCOL = CCOL/(1. + (BHMX + ZSMX)/DFMN)
WROW = 1.
IF(BHMN.NE.0.) WROW = CROW/(1. + (DFMX + ZSMX)/BHMN)
WLAY = 1.
IF(ZSMN.NE.0.) WLAY = CLAY/(1. + (DFMX + BHMX)/ZSMN)

C4------CALCULATE AND PRINT ITERATION PARAMETERS FROM THE AVERAGE SEED

C2D------SELECT THE CELL SEED, WHICH IS THE MINIMUM SEED OF THE 3.
WMIN = AMIN1(WCOL, WROW, WLAY)
WMINMN = AMIN1(WMINMN, WMIN)

C2E------ADD THE CELL SEED TO THE ACCUMULATOR AVGSUM FOR USE
AVGSUM = AVGSUM + DWMN
NODES = NODES + 1

C2D------SELECT THE MINIMUM SEED OVER THE WHOLE GRID.

C2E------IN GETTING THE AVERAGE SEED.

DO 100 CONTINUE

C3-------CALCULATE THE AVERAGE SEED OF THE CELL SEEDS, AND PRINT
100 FORMAT(1HO,'AVERAGE SEED = ', F11.8/1X,'MINIMUM SEED = ', F11.8)

C3-------THE AVERAGE AND MINIMUM SEEDS.

C4------RETURN

RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVGMIN</td>
<td>Module</td>
<td>Mean WMIN.</td>
</tr>
<tr>
<td>AVGSUM</td>
<td>Module</td>
<td>Sum of all of WMIN's.</td>
</tr>
<tr>
<td>B</td>
<td>Module</td>
<td>Conductance between this node and the one to the rear.</td>
</tr>
<tr>
<td>BHMIN</td>
<td>Module</td>
<td>Minimum of B and H (if the minimum is 0, it is the maximum).</td>
</tr>
<tr>
<td>BHMX</td>
<td>Module</td>
<td>Maximum of B and H.</td>
</tr>
<tr>
<td>C</td>
<td>Module</td>
<td>Number of columns.</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K).</td>
</tr>
<tr>
<td>CROW</td>
<td>Module</td>
<td>Intermediate factor.</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).</td>
</tr>
<tr>
<td>D</td>
<td>Module</td>
<td>Conductance between this node and the one to the left.</td>
</tr>
<tr>
<td>DFMN</td>
<td>Module</td>
<td>Minimum of D and F (if the minimum is 0, it is the maximum).</td>
</tr>
<tr>
<td>DFMX</td>
<td>Module</td>
<td>Maximum of D and F.</td>
</tr>
<tr>
<td>DWMIN</td>
<td>Module</td>
<td>Double precision WMIN.</td>
</tr>
<tr>
<td>F</td>
<td>Module</td>
<td>Conductance between this node and the one to the right.</td>
</tr>
<tr>
<td>H</td>
<td>Module</td>
<td>Conductance between this node and the one to the front.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, constant-head cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, inactive cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, variable-head cell</td>
</tr>
</tbody>
</table>
List of Variables for Module SSIPII (Continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NODES</td>
<td>Module</td>
<td>Number of variable-head (IBOUND &gt; 0) cells in the grid.</td>
</tr>
<tr>
<td>N Parm</td>
<td>Package</td>
<td>Number of iteration parameters.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>PIEPIE</td>
<td>Module</td>
<td>PI squared.</td>
</tr>
<tr>
<td>P1</td>
<td>Module</td>
<td>Index for the number of parameters.</td>
</tr>
<tr>
<td>P2</td>
<td>Module</td>
<td>DIMENSION (NPARM), Iteration parameters.</td>
</tr>
<tr>
<td>R</td>
<td>Module</td>
<td>Number of rows.</td>
</tr>
<tr>
<td>S</td>
<td>Module</td>
<td>Conductance between this node and the one below.</td>
</tr>
<tr>
<td>TMP</td>
<td>Module</td>
<td>Temporary field for nodes.</td>
</tr>
<tr>
<td>W</td>
<td>Package</td>
<td>Seed in the column direction for a cell.</td>
</tr>
<tr>
<td>WCOL</td>
<td>Module</td>
<td>Seed in the layer direction for a cell.</td>
</tr>
<tr>
<td>WLAY</td>
<td>Module</td>
<td>Minimum of (WCOL, WLAY, WROW).</td>
</tr>
<tr>
<td>WMIN</td>
<td>Module</td>
<td>Minimum WMIN.</td>
</tr>
<tr>
<td>WMIN MN</td>
<td>Module</td>
<td>Seed in the row direction for a cell.</td>
</tr>
<tr>
<td>Z</td>
<td>Module</td>
<td>Conductance between this node and the one above.</td>
</tr>
<tr>
<td>ZL</td>
<td>Module</td>
<td>Number of layers.</td>
</tr>
<tr>
<td>ZSMN</td>
<td>Module</td>
<td>Minimum of Z and S (if minimum is 0, it is the maximum).</td>
</tr>
<tr>
<td>ZSMX</td>
<td>Module</td>
<td>Maximum of Z and S.</td>
</tr>
</tbody>
</table>
CHAPTER 13
SLICE-SUCCESSIVE OVERRELAXATION PACKAGE

Conceptualization and Implementation

Successive overrelaxation is another method for solving large systems of linear equations by means of iteration. It is implemented in the model discussed herein through the Slice Successive Overrelaxation (SSOR) Package. Background material on the successive overrelaxation approach can be found in many standard references, including those already noted by Peaceman (1977), Crichlow (1977) and Remson, Hornberger and Molz (1971).

The successive overrelaxation technique is implemented in the SSOR Package by dividing the finite difference grid into vertical "slices," as shown in figure 54, and grouping the node equations into discrete sets, each set corresponding to a slice. In every iteration, these sets of equations are processed in turn, resulting in a new set of estimated head values for each slice. As the equations for each slice are processed, they are first expressed in terms of the change in computed head between successive iterations. The set of equations corresponding to the slice is then solved directly by Gaussian elimination, treating the terms for adjacent slices as known quantities (that is, inserting the most recently computed values of head for the adjacent slices as "known" values in the equations for the slice being processed). The values of head change computed for the slice in this Gaussian elimination process are then each multiplied by an acceleration parameter, \( \omega \), generally taken between 1 and 2; the results are taken as the final values of head change in that iteration.
Figure 54.—Division of the three-dimensional model array into vertical slices for processing in the SSOR package.
for the slice. They are added to the respective head values from the preceding iteration to obtain the final estimates of head for the iteration, for that slice. This procedure is repeated for each slice in sequence until all of the slices in the three-dimensional array have been processed, thus completing a single iteration. The entire sequence is then repeated, in successive passes through the series of slices, until the differences between the head values computed in successive iterations is less than the closure criterion at all nodes in the mesh.

It should be noted that even though a direct method of solution (Gaussian elimination) is used within each iteration to process the equations for each individual slice, the overall solution procedure is not direct but iterative. Each direct solution produces only interim values or estimates of head change based on the most recently computed heads in adjacent slices; as successive slices are processed, the computed values continue to change until closure is achieved.

The process of solution described above can be illustrated in more detail through consideration of the node equations. The equation of flow for an individual cell, as developed in chapter 2, is reproduced below with the addition of a second superscript to indicate iteration level

\[
\begin{align*}
CV_{i,j,k-1/2} &+ \frac{m_{i,j,k-1,\ell}}{2}h_{i,j,k-1} - CR_{i-1/2,j,k}h_{i-1,j,k-1} &+ CR_{i,j-1/2,k}h_{i,j-1,k} &+ CR_{i,j+1/2,k}h_{i,j+1,k} \\
&+ (- CV_{i,j,k-1/2} - CR_{i-1/2,j,k} - CR_{i,j-1/2,k} - CR_{i,j+1/2,k})h_{i,j,k-1/2} \\
&+ CC_{i+1/2,j,k}h_{i+1,j,k} &+ CV_{i,j,k+1/2}h_{i+1,j,k} &+ RHS_{i,j,k}
\end{align*}
\]

(113)
In equation (113), the superscript $m$ refers to the time step, while the superscript $\ell$ refers to the iteration level. If an equation of the form of (113) is written for the following iteration level, $\ell+1$, and the left side of equation (113) is then subtracted from each side of the new equation, the result can be written as

$$
CV_{i,j,k-1/2}^{m,\ell+1} (h_{i,j,k-1} - h_{i,j,k-1}) + CC_{i-1/2,j,k}^{m,\ell+1} (h_{i-1,j,k} - h_{i,j,k}) + CR_{i,j-1/2,k}^{m,\ell+1} (h_{i,j,k-1} - h_{i,j,k}) + (-CV_{i,j,k-1/2}^{m,\ell+1} - CC_{i-1/2,j,k}^{m,\ell+1} - CR_{i,j-1/2,k}^{m,\ell+1} - CR_{i,j+1/2,k}^{m,\ell+1} - CC_{i+1/2,j,k}^{m,\ell+1} + HCOF_{i,j,k}^{m,\ell+1}) (h_{i,j,k} - h_{i,j,k}) + CR_{i,j+1/2,k}^{m,\ell} (h_{i,j+1,k} - h_{i,j,k}) + CC_{i+1/2,j,k}^{m,\ell} (h_{i,j+1,k} - h_{i,j,k}) + CV_{i,j,k+1/2}^{m,\ell} (h_{i,j,k+1} - h_{i,j,k}) = w_{m,11}^{RHS_{i,j,k}} - CV_{i,j,k-1/2}^{m,\ell} (h_{i,j,k-1} - h_{i,j,k}) - CC_{i-1/2,j,k}^{m,\ell} (h_{i,j,k-1} - h_{i,j,k}) - CR_{i,j-1/2,k}^{m,\ell} (h_{i,j,k-1} - h_{i,j,k}) - (-CV_{i,j,k-1/2}^{m,\ell+1} - CC_{i-1/2,j,k}^{m,\ell+1} - CR_{i,j-1/2,k}^{m,\ell+1} - CR_{i,j+1/2,k}^{m,\ell+1} - CC_{i+1/2,j,k}^{m,\ell+1} + HCOF_{i,j,k}^{m,\ell+1}) (h_{i,j,k} - h_{i,j,k}) - CR_{i,j+1/2,k}^{m,\ell} (h_{i,j+1,k} - h_{i,j,k}) - CC_{i+1/2,j,k}^{m,\ell} (h_{i,j+1,k} - h_{i,j,k}) - CV_{i,j,k+1/2}^{m,\ell} (h_{i,j,k+1} - h_{i,j,k}) (114)
$$

In equation (114) the unknown terms are taken as the changes in computed head between iteration $\ell$ and iteration $\ell+1$--for example, $(h_{i,j,k} - h_{i,j,k})$. Note that when the $\ell$th iteration has been completed, the right hand side of (114) consists entirely of known terms--it includes the RHS and conductance terms assembled in the formulation process, and estimates of head already obtained during iteration $\ell$. Thus, the unknown terms can be replaced by suitable empirical or other data.
Now suppose that we divide the mesh into vertical slices taken along rows, as shown in figure 54, and isolate the equations associated with the nodes of an individual slice--for example, slice 4 of figure 54, which is taken along row 4 of the three dimensional array. In terms of equation (114), if we are processing slice \(i\), corresponding to row \(i\), we retain the head changes at nodes within this slice as unknown terms, but consider the head changes at nodes in the two adjacent slices to be known values. Thus the terms \(CC_{i-1/2,j,k}^{m,\ell+1} (h_{i-1,j,k} - h_{i-1,j,k})\) and \(CC_{i+1/2,j,k}^{m,\ell} (h_{i+1,j,k} - h_{i-1,j,k})\), on the left side of equation (114), are treated as known quantities. If we move these two expressions to the right side of the equation and rearrange, we find that the terms in \(h_{i-1,j,k}\) and \(h_{i+1,j,k}\) drop out, leaving

\[
CV_{i,j,k-1/2}^{m,\ell+1} (h_{i,j,k-1} - h_{i,j,k-1}) + CR_{i,j-1/2,k}^{m,\ell} (h_{i,j-1,k} - h_{i,j-1,k}) =\]

\[
- CC_{i+1/2,j,k}^{m,\ell+1} (h_{i+1,j,k} - h_{i,j,k}) + CV_{i,j,k+1/2}^{m,\ell} (h_{i,j,k+1} - h_{i,j,k+1}) - CR_{i,j+1/2,k}^{m,\ell} (h_{i,j+1,k} - h_{i,j+1,k}) + HCO_{i,j,k}^{m,\ell} (h_{i,j,k} - h_{i,j,k})
\]

\[
RHS_{i,j,k} - CR_{i,j-1/2,k}^{m,\ell} (h_{i,j-1,k} - h_{i,j-1,k}) - CR_{i,j+1/2,k}^{m,\ell} (h_{i,j+1,k} - h_{i,j+1,k}) + CV_{i,j,k-1/2}^{m,\ell} (h_{i,j,k-1} - h_{i,j,k-1}) + CR_{i,j+1/2,k}^{m,\ell} (h_{i,j+1,k} - h_{i,j+1,k})
\]

\[
(115)
\]
Now suppose the slices are processed in the order of increasing row number, \( i \); then calculations for slice \( i-1 \) will be completed in each iteration before calculations for slice \( i \) are initiated. It follows that a value of \( h_{i-1,j,k} \) will be available when the processing of slice \( i \) is initiated in iteration \( i+1 \), whereas a value of \( h_{i+1,j,k} \) will not be available. Thus the term \( C C_{i-1/2,j,k} h_{i-1,j,k} \) can be incorporated directly as a known term in the processing of slice \( i \), but the term \( C C_{i+1/2,j,k} h_{i+1,j,k} \) cannot. To circumvent this difficulty, the value of \( h_{i+1,j,k} \) from the preceding iteration, \( h_{i+1,j,k} \), is substituted for \( h_{i+1,j,k} \) on the right side of (115). (Thus in effect we are using the most recently calculated value of head for each adjacent slice.) The resulting equation is

\[
\begin{align*}
& CV_{i,j,k-1/2} (h_{i,j,k-1}-h_{i,j,k-1}) + CR_{i,j-1/2,k}(h_{i,j,k-1}-h_{i,j,k-1}) \\
& + (\text{\text{-}CV_{i,j,k-1/2} - CC_{i-1/2,j,k} - CR_{i,j-1/2,k} - CR_{i,j+1/2,k}}) \\
& - CR_{i,j+1/2,k}(h_{i,j+1/2,k}-h_{i,j+1/2,k}) + CV_{i,j,k+1/2}(h_{i,j,k+1/2,k}-h_{i,j,k+1/2,k}) \\
& + \text{RHS}_{i,j,k} - CV_{i,j,k-1/2} h_{i,j,k-1} - CC_{i-1/2,j,k} h_{i-1,j,k} \\
& - CR_{i,j+1/2,k}(h_{i,j+1/2,k}-h_{i,j+1/2,k}) - CC_{i+1/2,j,k} h_{i+1,j,k} - CV_{i,j,k+1/2} h_{i,j,k+1/2,k} \\
& - \text{(116)}
\end{align*}
\]

In equation (116), the notation \( h \) has been introduced for the head terms in slice \( i \) at iteration \( i+1 \). The purpose of this notation will become...
clear as the solution process is described. The number of nodes in the slice is \( NC \times NL \), where \( NC \) is the number of columns in the model and \( NL \) the number of layers; and an equation of the form of (116) is formulated at each node. Thus a system of \( NC \times NL \) equations in \( NC \times NL \) unknowns is established. Because the number of layers is usually small, the total number of equations is generally small enough so that direct solution by Gaussian elimination is an efficient approach (note that such a procedure would generally not be feasible for the larger set of equations associated with the entire three-dimensional model array.)

The set of equations associated with an individual slice, \( i \), can be written in matrix form as

\[
[A]_i \{\Delta h\}_i = \{R\}_i
\]

where \([A]_i\) is the coefficient matrix for slice \( i \); \( \{\Delta h\}_i \) is a vector of estimates, \( \hat{h}_{i,j,k} - h_{i,j,k} \), for the change in computed head at each node in the slice between iteration \( \ell \) and iteration \( \ell + 1 \); and \( \{R\}_i \) is the vector of "constant" terms, representing the right side of equation (116), for slice \( i \).

The Gaussian elimination procedure applied to the matrix equations (117) yields one value of the term \( \hat{h}_{i,j,k} - h_{i,j,k} \) for each node in the slice. These terms are taken as first estimates for the change in computed head from iteration \( \ell \) to iteration \( \ell + 1 \). Each is multiplied by the acceleration parameter, \( \omega \), and each result is added to the corresponding head from the preceding iteration to obtain the final estimate of head for iteration \( \ell + 1 \); that is,

\[
\hat{h}_{i,j,k} = h_{i,j,k} + \omega (\hat{h}_{i,j,k} - h_{i,j,k})
\]
When values of $h_{i,j,k}^{m+1}$ have been computed for each node $(j,k)$ in slice $i$, the procedure of calculation is initiated for the succeeding slice, $i+1$. When all slices have been processed the iteration is complete, and calculations are initiated for the next iteration unless closure has been achieved.

As illustrated in figure 55-a, the matrix of coefficients $[A]_i$ of equation (117) is symmetric and banded, with a maximum half-bandwidth equal to the number of layers. Because of the symmetry of the matrix, only the lower triangular portion has to be stored; this storage is provided in the program in a two-dimensional array, as illustrated in figure 55-b, with dimensions $NL*NC$ and $NL+1$. In this example, $NL=NC=3$.

Adjustment of the acceleration parameter is frequently necessary in SSOR to achieve optimal rates of convergence. For this purpose, methods similar to the trial and error procedure described in Chapter 12, for adjustment of the SIP "seed" value can be applied.
(a) Coefficient matrix for an individual slice

(b) Two dimensional array for storage of matrix elements

Figure 55.—Coefficient matrix for slice equations and corresponding computer storage array.
Slice-Successive Overrelaxation Package Input

Input to the Slice-Successive Overrelaxation (SOR) Package is read from the unit specified in IUNIT(11).

FOR EACH SIMULATION

SOR1AL

1. Data: MXITER
   Format: I10

SOR1RP

2. Data: ACCL HCLOSE IPRSOR
   Format: F10.0 F10.0 I10

Explanation of Fields Used in Input Instructions

MXITER--is the maximum number of iterations allowed in a time step.

ACCL--is the acceleration parameter, usually between 1.0 and 2.0.

HCLOSE--is the head change criterion for convergence. When the maximum absolute value of head change from all nodes during an iteration is less than or equal to HCLOSE, iteration stops.

IPRSOR--is the printout interval for SOR. IF IPRSOR is equal to zero, it is changed to 999. The maximum head change (positive or negative) is printed for each iteration of a time step whenever the time step is an even multiple of IPRSOR. This printout also occurs at the end of each stress period regardless of the value of IPRSOR.
The Slice-Successive Overrelaxation Package (SORl) consists of three primary modules and one submodule. They are:

**Primary Modules**

- **SORIAL**: Allocates space for arrays.
- **SORlRP**: Reads control information needed by the SORl Package.
- **SORlAP**: Performs one iteration of slice-successive overrelaxation.

**Submodule**

- **SSORlB**: Solves a system of linear equations.
Module SORIAL allocates space in the X array for SOR arrays. The SOR arrays are A, RES, IEQPNT, HDCG, and LRCH. "A" holds the main diagonal and the lower diagonals of the symmetric coefficient matrix for a single slice. RES holds the residual vector (the right hand sides) for a single slice. IEQPNT holds a sequential identification number for each cell in a slice. HDCG holds the maximum head change for each iteration. LRCH holds the location of the cell (row, column, and layer) which had the maximum head change for each iteration.

Module SORIAL performs its functions in the following order:

1. Print a message identifying the SOR Package.

2. Read and print the maximum number of iterations.

3. Allocate the required space in the X array. The X-array location pointer (ISUM) is saved in the variable ISOLD prior to allocation so that the space required for SOR can be calculated in step 4. To allocate space for an array, the array-location variable is set equal to ISUM. Then ISUM is incremented by the required number of elements.

4. Calculate and print the space used in the X array. The space used by SOR is ISUM - ISOLD.

5. RETURN
X array is the pool of memory space from which space is allocated for arrays used by various packages.
SUBROUTINE SORIAL(ISUM, LENX, LCA, LCRES, LCHDCG, LCLRCH, LCIEQP,
  1  MXITER, NCOL, NLAY, NSLICE, MBW, IN, IOUT)

C------VERSION 1638 24 JUL 1987 SORIAL
C ********************************~*~*****************~*************
C ALLOCATE STORAGE FOR SOR ARRAYS
C ************************~***************************~*******~~*****
C SPECIFICATIONS:
C --------------------____________________------------.--------------
C ----------------------------------------------------~-------------
C       --- PRINT A MESSAGE IDENTIFYING SOR PACKAGE
C
C1------READ AND PRINT MXITER (MAXIMUM # OF ITERATIONS)
       WRITE(IOUT,1)IN
       1 FORMAT(1HO,'SOR1 -- SLICE-SUCCESSIVE OVERRELAXATION PACKAGE',
            1', VERSION 1, 9/1/87 INPUT READ FROM UNIT',I3)

C C2------ALLOCATE SPACE FOR THE SOR ARRAYS
       ISOLD=ISUM
       NSLICE=NCOL*NLAY
       MBW=NLAY+1
       LCA=ISUM
       ISUM=ISUM+NSLICE*MBW
       LCRES=ISUM
       ISUM=ISUM+NSLICE
       LCIEQP=ISUM
       ISUM=ISUM+NSLICE
       LCHDCG=ISUM
       ISUM=ISUM+MXITER
       LCLRCH=ISUM
       ISUM=ISUM+3*MXITER
       ISF=ISUM-ISOLD

C C3------CALCULATE AND PRINT THE SPACE USED IN THE X ARRAY
       WRITE(IOUT,4) ISP
       4 FORMAT(1X,I8,' ELEMENTS IN X ARRAY ARE USED BY SOR')
       ISP=ISUM-1
       WRITE(IOUT,5) ISP,LENX
       5 FORMAT(1X,I8,' ELEMENTS OF X ARRAY USED OUT OF',I8)
       IF(ISUM1.GT.LENX) WRITE(IOUT,6)
       6 FORMAT(1X,' ***X ARRAY MUST BE DIMENSIONED LARGER***')

C C5------RETURN
       RETURN
       END
### List of Variables for Module SORIAL

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>ISOLD</td>
<td>Package</td>
<td>Before this module allocates space, ISOLD is set equal to ISUM. After allocation, ISOLD is subtracted from ISUM to get ISP, the amount of space in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISP</td>
<td>Module</td>
<td>Number of words in the X array allocated by this module.</td>
</tr>
<tr>
<td>ISUM</td>
<td>Global</td>
<td>Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.</td>
</tr>
<tr>
<td>ISUM1</td>
<td>Module</td>
<td>Index number of the last element of the X array allocated by this module.</td>
</tr>
<tr>
<td>LCA</td>
<td>Package</td>
<td>Location in the X array of the first element of array A.</td>
</tr>
<tr>
<td>LCHDCG</td>
<td>Package</td>
<td>Location in the X array of the first element of array HDCG.</td>
</tr>
<tr>
<td>LCIEQP</td>
<td>Package</td>
<td>Location in the X array of the first element of array IEQPNT.</td>
</tr>
<tr>
<td>LCLRCH</td>
<td>Package</td>
<td>Location in the X array of the first element of array LRCH.</td>
</tr>
<tr>
<td>LCRES</td>
<td>Package</td>
<td>Location in the X array of the first element of array RES.</td>
</tr>
<tr>
<td>LENX</td>
<td>Global</td>
<td>Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.</td>
</tr>
<tr>
<td>MBW</td>
<td>Package</td>
<td>Maximum bandwidth of the coefficient matrix +1.</td>
</tr>
<tr>
<td>MXITER</td>
<td>Package</td>
<td>Maximum number of iterations.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NSLICE</td>
<td>Package</td>
<td>Number of cells in a slice.</td>
</tr>
</tbody>
</table>
Module SOR1RP reads data for the SOR package: the acceleration parameter (ACCL), also called the relaxation factor; the closure criterion (HCLOSE); and the time-step interval (IPRSOR) for printing head change. This module does not have a flow chart. Module SOR1RP performs its functions in the following order:

1. Read the acceleration parameter (ACCL), the closure criterion (HCLOSE), and the interval for printing head change (IPRSOR). If ACCL is zero, substitute a default value of 1.0. If IPRSOR is less than one, set it equal to 999.

2. Print the maximum number of iterations (MXITER), the acceleration parameter (ACCL), the closure criterion (HCLOSE), and the head-change interval (IPRSOR).

3. RETURN.
SUBROUTINE SOR1RP(MXITER, ACCL, HCLOSE, IN, IPRSOR, IOUT)

C-----VERSION 1005 16MAR1983 SOR1RP
C ***************************************************************
C READ PARAMETERS FOR SOR
C ***************************************************************

SPECIFICATIONS:
------------------------------------------------------------------
READ(IN,1) ACCL, HCLOSE, IPRSOR
1 FORMAT(2F10.0,1I0)
  IF(ACCL.EQ.0.) ACCL=1.
  IF(IPRSOR.LT.1) IPRSOR=999

C2------PRINT ACCL, HCLOSE, IPRSOR
WRITE(IOUT,100)
100 FORMAT(1HO, //57X,'SOLUTION BY SLICE-SUCCESSIVE OVERRELAXATION'
  1/57X,43('-'))
WRITE(IOUT,115) MXITER
115 FORMAT(1HO,47X,'MAXIMUM ITERATIONS ALLOWED FOR CLOSURE =',I9)
WRITE(IOUT,120) ACCL
120 FORMAT(1H,63X,'ACCELERATION PARAMETER =',G15.5)
WRITE(IOUT,125) HCLOSE
125 FORMAT(1H,52X,'HEAD CHANGE CRITERION FOR CLOSURE =',E15.5)
WRITE(IOUT,130) IPRSOR
130 FORMAT(1H,52X,'SOR HEAD CHANGE PRINTOUT INTERVAL =',I9)

C3------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCL</td>
<td>Package</td>
<td>Acceleration parameter.</td>
</tr>
<tr>
<td>HCLOSE</td>
<td>Package</td>
<td>Closure criterion for the iterative procedure.</td>
</tr>
<tr>
<td>IN</td>
<td>Package</td>
<td>Primary unit number from which input for this package will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPRSOR</td>
<td>Package</td>
<td>Frequency (in time steps) with which the maximum head changes for each iteration will be printed.</td>
</tr>
<tr>
<td>MXITER</td>
<td>Package</td>
<td>Maximum number of iterations.</td>
</tr>
</tbody>
</table>
Module SOR1AP performs one iteration of the Slice-Successive Over-relaxation (SSOR) algorithm for solving the system of finite-difference equations. The conductances CC, CR, and CV and the composite terms HCOF and RHS (see equation (27)) which are calculated by the formulation procedure are combined, row by row (slice by slice), to form the coefficient matrix \([A]\) and the vector \([\text{RES}]\) on the right hand side of the matrix equation for a single slice. Since the coefficient matrix is symmetric and banded, only main diagonals and NLAY subdiagonals are saved. As heads are calculated, they are stored in the array HNEW. The matrix \([A]\) and the vector \([\text{RES}]\) are passed to a submodule SSOR1B which solves the matrix equation for a vector of approximate head changes which is then multiplied by the relaxation factor to get the final head changes for the iteration. The final head changes are added to the heads from the preceding iteration to get the heads for the current iteration. The final head changes for the iteration are compared to the closure criterion to see if the iterative procedure has closed.

Module SOR1AP performs its functions in the following order:

1. Calculate the number of elements in the compressed coefficient matrix \([A]\).

2. Process the slices (rows) one at a time (DO STEPS 3-7).

3. Clear the A array.

4. Assign integers sequentially to the active cells in the slice (remember that finite-difference equations are formulated only for active cells).
5. Calculate the elements in the compressed coefficient matrix \([A]\) and the residual vector \([\text{RES}]\). Process the cells in the slice one cell at a time.

If the cell is inactive, move on to the next cell. The elements in the main diagonal of the coefficient matrix (the multipliers of \(h_{i,j,k}\)) will consist of HCOF plus conductances to the six adjacent cells. They will be formed in an accumulator called EE. The contents of EE multiplied by the head from the previous iteration (HNEW) are subtracted from an accumulator (R) to form the residual.

(a) Determine the equation number (NEQ) of the cell. If NEQ is zero, the cell is inactive. Move on to the next cell.

(b) Set the accumulators EE and R equal to HCOF and RHS, respectively. Note: HNEW contains head from the last iteration.

(c) If there is a node to the left, subtract the conductance from EE and subtract the conductance times HNEW from R.

(d) If there is a node to the right, subtract the conductance from EE, and subtract the conductance times HNEW from R; and, if the cell to the right is active, move the conductance into the compressed coefficient matrix \([A]\). Remember that the coefficient matrix is symmetric so the conductance to the left in step 5(c) did not have to be stored.

(e) If there is a node to the rear, subtract the conductance from EE and subtract the conductance times HNEW from R.

(f) If there is a node to the front, subtract the conductance from EE and subtract the conductance times HNEW from R. Remember that the
form of the SSOR equations does not have terms containing head in adjacent rows on the left hand side.

(g) If there is a node above, subtract the conductance from EE and subtract the conductance times HNEW from R.

(h) If there is a node below, subtract the conductance from EE and subtract the conductance times HNEW from R; and, if the cell below is active, move the conductance into A.

(i) Move EE into the first row of A. The first row in A corresponds to the main diagonal in the "full" coefficient matrix. Subtract EE times HNEW from R and store it in the residual vector.

6. If there are no equations for this slice, go on to the next slice. If there is only one equation, solve it directly and leave the result in the residual vector \{RES\}. If there are two or more equations, call submodule SSORIB to solve the system of equations for the slice leaving the results (first estimate of head change for this iteration) in the vector \{RES\}.

7. For each cell in the slice, calculate the head for the current iteration.

(a) Multiply the first estimate of head change for this iteration by the relaxation factor to get the final estimate of head change for this iteration.

(b) Add the final head change for this iteration to the head from the last iteration to get the head for this iteration.
(c) If the head change for this cell is greater than that for any other cell, store the head change and the location of the cell.

8. Save the largest head change from this iteration so that it can be printed at the end of the time step.

9. Compare the biggest head change (BIGG) to the closure criterion (HCLOSE). If HCLOSE is greater than BIGG, set the convergence flag (ICNVG) equal to one.

10. If you have not converged and you have not exceeded the maximum number of iterations, RETURN.

11. Print the number of iterations.

12. If convergence failed, or this is the last time step, or this is the time step interval specified by the user, print the maximum head change for each iteration in this time step.

13. RETURN.
A is a compressed coefficient matrix for a slice. It contains the main diagonal of the full matrix and the NLAY diagonals below it. (NLAY is the number of layers.)

Sequence Number is a number used to identify the internal (variable-head) cells in a slice and also the equations for each internal cell.

RES is a vector containing the residuals for a slice. It consists of RHS (from the basic finite-difference equation) plus all of those terms which are moved to the right hand side to get the equations ready for solution in residual form.

First estimates of head change: these are the head changes calculated by simultaneously solving the equations for a slice. They will be multiplied by the relaxation factor to get final estimates of head change.

Final estimates of head change: these are the head changes calculated by multiplying first estimates by the relaxation factor. They are added to the heads from the previous iteration to get head for the current iteration.

ICNVG is the convergence flag. It is set in the approximator and returned to the MAIN Program so that the iteration loop can be terminated.
SUBROUTINE SORLAP(HNEW, IBOUND, CR, CC, CV, HCF, RHS, A, RES, IEQPNT, 
                   HDCG, LRCH, KITER, HCLOSE, ACCL, ICNVG, KSTP, KPER, 
                   IPRSOR, MXITER, NSTP, NCOL, NROW, NLAY, NSLICE, MBW, IOUT)
C-----VERSION 0936 09MAY1983 SORLAP
C
C SOLUTION BY SLICE-SUCCESSIVE OVERRELAXATION -- 1 ITERATION
C
C SPECIFICATIONS:
C
DOUBLE PRECISION HNEW, DIFF, DP, EE, R, HCFHNM, HCOF
C
DIMENSION HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY), 
1     CR(NCOL, NROW, NLAY), CC(NCOL, NROW, NLAY), 
1     CV(NCOL, NROW, NLAY), HCF(NCOL, NROW, NLAY), 
2     HDCG(MXITER), LRCH(MXITER), A(MBW, NSLICE), RES(NSLICE), 
3     IEQPNT(NLAY, NCOL)
C
C1-------CALCULATE # OF ELEMENTS IN COMPRESSED MATRIX A AND
C1-------INITIALIZE FIELDS TO SAVE LARGEST HEAD CHANGE.
   NA=MBW*NSLICE
   BIG=0.
   ABSBIG=0.
   IB=0
   JB=0
   KB=0
C
C2-------PROCESS EACH SLICE
   DO 500 I=1, NROW
C
C3-------CLEAR A
   DO 110 J=1, NSLICE
   DO 110 K=1, MBW
    110 A(K,J)=0.
   E4
C
C4-------ASSIGN A SEQUENCE # TO EACH VARIABLE HEAD CELL.
   NEQT=0
   DO 200 J=1, NCOL
   DO 200 K=1, NLAY
    IEQPNT(K,J)=0
    IF(BOUND(J,I,K).LE.0) GO TO 200
    NEQT=NEQT+1
    IEQPNT(K,J)=NEQT
200 CONTINUE
C
C5-------FOR EACH CELL LOAD MATRIX A AND VECTOR RES
   DO 300 J=1, NCOL
   DO 300 K=1, NLAY
C
C5A-------IF SEQUENCE # IS 0 (CELL IS EXTERNAL) GO ON TO NEXT CELL
   NEQ=IEQPNT(K,J)
   IF(NEQ.EQ.0) GO TO 300
C
C5B-------INITIALIZE ACCUMULATORS EE AND R
   EE=0.
   R=RHS(J,I,K)
C
C5C-----IF NODE TO LEFT SUBTRACT TERMS FROM EE AND R
   IF(J,EQ.1) GO TO 120
   DP=CR(J-1,I,K)
   R=R-DP*HNEW(J-1,I,K)
   EE=EE-DP
C
C5D-----IF NODE TO RIGHT SUBTRACT TERMS FROM EE & R, MOVE COND TO A
   120 IF(J,EQ.NCOL) GO TO 125
       SP=CR(J,I,K)
       DP=SP
       R=R-SP*HNEW(J+1,I,K)
       EE=EE-SP
       NXT=IEQPNT(K,J+1)
       IF(NXT.GT.0) A(1+NXT-NEQ,NEQ)=SP
C
C5E-----IF NODE TO REAR SUBTRACT TERMS FROM EE AND R
   125 IF(I,EQ.1) GO TO 130
   DP=CC(J,I-1,K)
   R=R-SP*HNEW(J,I-1,K)
   EE=EE-SP
C
C5F-----IF NODE TO FRONT SUBTRACT TERMS FROM EE AND R
   130 IF(I,EQ.NROW) GO TO 132
   DP=CV(J,I,K-1)
   R=R-SP*HNEW(J,I-1,K)
   EE=EE-SP
   NXT=IEQPNT(K+1,J).GT.0) A(2,NEQ)=SP
C
C5G-----IF NODE ABOVE SUBTRACT TERMS FROM EE & R AND MOVE COND TO A
   132 IF(K,EQ.1) GO TO 134
   DP=CV(J,I,K)
   R=R-SP*HNEW(J,I+1,K)
   EE=EE-SP
   IF(IEQPNT(K+1,J).GT.0) A(2,NEQ)=SP
C
C5H-----MOVE EE INTO A, SUBTRACT EE TIMES LAST HEAD FROM R TO GET RES
   134 IF(K,EQ.NLAY) GO TO 136
       HHCQF=HCOF(J,I,K)
       A(1,NEQ)=EE+HHCQF
       HNW=HNEW(J,I,K)
       HCFHNW=HNW*HCOF(J,I,K)
       RES(NEQ)=R-EE*HNEW(J,I,K)-HCFHNW
   300 CONTINUE
C
C6------IF NO EQUATIONS GO TO NEXT SLICE, IF ONE EQUATION SOLVE
C6------DIRECTLY, IF 2 EQUATIONS CALL SSORIB TO SOLVE FOR FIRST
C6------ESTIMATE OF HEAD CHANGE FOR THIS ITERATION.
   IF(NEQT.LT.1) GO TO 500
   IF(NEQT.EQ.1) RES(1)=RES(1)/A(1,1)
   IF(NEQT.GE.2) CALL SSORIB(A,RES,NEQT,NA,MBW)
C
C7------FOR EACH CELL IN SLICE CALCULATE FINAL HEAD CHANGE THEN HEAD.
   DO 400 J=1,NCOL
       DO 400 K=1,NLAY
           NEQ=IEQPNT(K,J)
           IF(NEQ.EQ.0) GO TO 400

400 CONTINUE

300
C7A-----MULTIPLY FIRST ESTIMATE OF HEAD CHANGE BY RELAX FACTOR TO
    GET FINAL ESTIMATE OF HEAD CHANGE FOR THIS ITERATION.
    DH = RES(NEQ) * ACCL
    DIFF = DH
C
C7B-----ADD FINAL ESTIMATE TO HEAD FROM LAST ITERATION TO GET HEAD
    FOR THIS ITERATION.
    HNEW(J, I, K) = HNEW(J, I, K) + DIFF
C
C7C-----SAVE FINAL ESTIMATE IF IT IS THE LARGEST
    ABSDH = ABS(DH)
    IF (ABSDH, LE, ABSBIG) GO TO 400
    ABSBIG = ABSDH
    BIG = DH
    IB = J
    JB = I
    KB = K
    400 CONTINUE
C
    500 CONTINUE
C
C8-------SAVE LARGEST HEAD CHANGE FOR THIS ITERATION
    HDCG(KITER) = BIG
    LRCH(1, KITER) = KB
    LRCH(2, KITER) = IB
    LRCH(3, KITER) = JB
C
C9-------IF LARGEST HEAD CHANGE IS SMALLER THAN CLOSURE THEN SET
    CONVERGE FLAG (ICNVG) EQUAL TO 1.
    IF (ABSBIG, LE, HCLOSE) ICNVG = 1
C
C10------IF NOT CONVERGED AND NOT EXCEEDED ITERATIONS THEN RETURN
    IF (ICNVG, EQ, 0 .AND. KITER, NE, MXITER) RETURN
    IF (KSTP, EQ, 1) WRITE(IOUT, 600)
    600 FORMAT(1HO)
C
C11------PRINT NUMBER OF ITERATIONS
    WRITE(IOUT, 601) KITER, KSTP, KPER
    601 FORMAT(1X, I5,' ITERATIONS FOR TIME STEP', I4,' IN STRESS PERIOD',
    1 I3)
C
C12------IF FAILED TO CONVERGE OR LAST TIME STEP OR PRINTOUT
    INTERVAL SPECIFIED BY USER IS HERE THEN PRINT MAXIMUM
    HEAD CHANGES FOR EACH ITERATION.
    IF (ICNVG, NE, 0 .AND. KSTP, NE, NSTP .AND. MOD(KSTP, IPRSOR), NE, 0)
    1 GO TO 700
    WRITE(IOUT, 5)
    5 FORMAT(1HO,'MAXIMUM HEAD CHANGE FOR EACH ITERATION:/
    1 1HO,' HEAD CHANGE LAYER, ROW, COL')/1X, 120('-')
    WRITE (IOUT, 10) (HDCG(J), (LRCH(I, J), I=1, 3), J=1, KITER)
    10 FORMAT(1X, 4(4X, GL2.4, ' (' , I3, ',', I3, ',', I3, ',', I3, ')'))
    WRITE(IOUT, 11)
    11 FORMAT(1HO)
C
C13------RETURN
    700 RETURN
C
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Package</td>
<td>DIMENSION (MBW,NSLICE), Compressed coefficient matrix for a slice.</td>
</tr>
<tr>
<td>ABSDIG</td>
<td>Module</td>
<td>Largest ABSDH for this iteration.</td>
</tr>
<tr>
<td>ABSDH</td>
<td>Module</td>
<td>Absolute value of head change in a cell for the current iteration.</td>
</tr>
<tr>
<td>ACCL</td>
<td>Global</td>
<td>Acceleration parameter.</td>
</tr>
<tr>
<td>CC</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K).</td>
</tr>
<tr>
<td>CR</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).</td>
</tr>
<tr>
<td>CV</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).</td>
</tr>
<tr>
<td>DH</td>
<td>Module</td>
<td>Change in head in a cell during one iteration.</td>
</tr>
<tr>
<td>DIFF</td>
<td>Module</td>
<td>Double-precision change in head (DH).</td>
</tr>
<tr>
<td>DP</td>
<td>Module</td>
<td>Double-precision temporary field.</td>
</tr>
<tr>
<td>EE</td>
<td>Module</td>
<td>Main diagonal term in the finite-difference equation.</td>
</tr>
<tr>
<td>HCLOSE</td>
<td>Package</td>
<td>Closure criterion for the iterative procedure.</td>
</tr>
<tr>
<td>HCOF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K) in the finite-difference equation.</td>
</tr>
<tr>
<td>HDCG</td>
<td>Package</td>
<td>DIMENSION (MXITER), Maximum head change for each iteration.</td>
</tr>
<tr>
<td>HNFW</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IBOUND</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Status of each cell. &lt; 0, constant-head cell = 0, inactive cell &gt; 0, variable-head cell</td>
</tr>
<tr>
<td>ICNVG</td>
<td>Global</td>
<td>Flag is set equal to one when the iteration procedure has converged.</td>
</tr>
<tr>
<td>IEQPNT</td>
<td>Global</td>
<td>DIMENSION (NLAY,NCOL), Sequence numbers for variable-head cells in a slice.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPRSOR</td>
<td>Package</td>
<td>Frequency (in time steps) with which the maximum head changes for each iteration will be printed.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Column number of the cell containing the largest head change.</td>
</tr>
<tr>
<td>JB</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for layers.</td>
</tr>
<tr>
<td>KB</td>
<td>Module</td>
<td>Layer number of the cell containing the largest head change.</td>
</tr>
<tr>
<td>KITER</td>
<td>Global</td>
<td>Iteration counter. Reset at the start of each time step.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>Variable</td>
<td>Range</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>LRCH</td>
<td>Package</td>
<td>DIMENSION (MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.</td>
</tr>
<tr>
<td>MBW</td>
<td>Package</td>
<td>Maximum bandwidth of the coefficient matrix +1.</td>
</tr>
<tr>
<td>MXITER</td>
<td>Package</td>
<td>Maximum number of iterations.</td>
</tr>
<tr>
<td>NA</td>
<td>Package</td>
<td>Number of elements in the compressed coefficient matrix (A).</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NEQ</td>
<td>Module</td>
<td>Index for equations (variable-head cells) in a slice.</td>
</tr>
<tr>
<td>NEQT</td>
<td>Package</td>
<td>Number of equations (variable-head cells) in a slice.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>NSLICE</td>
<td>Package</td>
<td>Number of cells in a slice.</td>
</tr>
<tr>
<td>NSTP</td>
<td>Global</td>
<td>Number of time steps in the current stress period.</td>
</tr>
<tr>
<td>NXT</td>
<td>Module</td>
<td>Sequence number of the cell to the right.</td>
</tr>
<tr>
<td>R</td>
<td>Module</td>
<td>Right hand side of the finite-difference equation as modified (terms for the adjacent rows moved to the right) for solution by the slice-successive overrelaxation.</td>
</tr>
<tr>
<td>RES</td>
<td>Package</td>
<td>DIMENSION (NSLICE), Residual.</td>
</tr>
<tr>
<td>RHS</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.</td>
</tr>
<tr>
<td>SP</td>
<td>Module</td>
<td>Single-precision temporary field.</td>
</tr>
</tbody>
</table>
Module SSOR1B uses Gaussian elimination to solve a matrix equation of the form \([A]\{X\} = \{B\}\). The matrix \([A]\) is symmetric and banded with rank "N." It is stored in the compressed format shown in figure 55 and uses a single subscript. The vector \(\{X\}\), as it is calculated, is stored in the space reserved for vector \(\{R\}\).

The indices used in the module flow chart are those for standard matrix organization. The indices actually used in the program are based on the compressed format and a single index. Module SSOR1B performs its functions in the following order:

1. Work through the first \(N-1\) rows using each one, in sequence, as the pivot row (row \(I\)).

2. Calculate the inverse of the main diagonal element---\(a_{I,I}\). (The index (ID) points to \(a_{I,I}\) which is the first element of column \(I\) in the compressed matrix \([A]\).)

3. Modify each of the rows after row \(I\) so that the terms corresponding to the pivot term are eliminated. Since the coefficient matrix is banded, there are only \(MBW-1\) equations (where \(MBW\) is the maximum half-bandwidth plus one) where the term to be eliminated is not already equal to zero. The rows are indexed by "L." (The equation corresponding to row "L" is referred to as equation "L").

4. Calculate the coefficient \(C\) which when multiplied by the pivot equation and subtracted from equation \(L\) will eliminate a term in equation \(L\).
5. Calculate the new coefficients in equation L for each of the terms to the right of the coefficient that is being eliminated. Because the matrix is banded, there are only MBW-1 nonzero terms to the right of the pivot. Therefore, at most, MBW-1 coefficients have to be calculated.

6. Subtract C times a coefficient in the pivot equation from the corresponding coefficient in equation L.

7. Subtract C times the right side of the pivot equation from the right side of equation L. (The index LB points to the coefficient in equation L which must be calculated.)

8. Solve equation N for X(N) putting the result in B(N).

9. Work backward from equation N-1 solving each equation (equation L) for X(L).

10. Set the accumulator "SUM" equal to zero.

11. Multiply the coefficient to the right of the main diagonal (in equation L) by the corresponding value of X and add it to the sum.

12. Calculate the value of X(L) and store it in B(L).

13. RETURN.
MBW is the maximum half bandwidth of the coefficient matrix plus one.

Flow Chart for Module SSOR1B

1. ENTER SSOR1B
2. FOR EACH ROW I (I=1,..., N-1)
   3. CALCULATE C1, THE INVERSE OF THE PIVOT C1=1/a11
   4. FOR EACH ROW J (J=1,..., MBW-1)
      5. CALCULATE C = a1,J+K * C1
      6. SUBTRACT C * B(I) FROM B(I+J)
   7. FOR EACH COLUMN I+K
      K=J,..., MBW-1
      8. SOLVE THE LAST EQUATION FOR X(N) AND STORE IT IN B(N)
   9. FOR EACH EQUATION L (L=N-1, N-2,..., 1)
      10. CLEAR THE ACCUMULATOR SUM
   11. FOR EACH TERM L+J
      J=1,..., MBW-1
      12. ADD THE TERM aL,L+J * X[L+J] TO SUM

RETURN

13-31
SUBROUTINE SSOR1B(A, B, N, NA, MBW)

C----VERSION 1359 31MAR1903 SSOR1B
C*************************************************************************
C SOLVE A SYMMETRIC SET OF EQUATIONS
C A IS COEFFICIENT MATRIX IN COMPRESSED FORM
C B IS RIGHT HAND SIDE AND IS REPLACED BY SOLUTION
C N IS NUMBER OF EQUATIONS TO BE SOLVED
C MBW IS BANDWIDTH OF A
C NA IS ONE-DIMENSION SIZE OF A
*************************************************************************
C
C SPECIFICATIONS:
C---------------------------------------------
DIMENSION A(NA), B(N)
---------------------------------------------
C
NMI = N-1
MBW1 = MBW - 1
ID = 1 - MBW
C
C1------SEQUENTIALLY USE EACH OF THE FIRST N-1 ROWS AS
C1------THE PIVOT ROW.
   DO 20 I=1, NMI

C
C2------CALCULATE THE INVERSE OF THE PIVOT.
   ID = ID + MBW
   CI = 1./A(ID)
   LD = ID
   L = 1

C
C3------FOR EACH ROW AFTER THE PIVOT ROW (THE TARGET ROW)
C3------ELIMINATE THE COLUMN CORRESPONDING TO THE PIVOT.
   DO 15 J=1, MBW1
   L = L + 1
   IF (L.GT.N) GO TO 20
   IB = ID + J

C
C4------CALCULATE THE FACTOR NEEDED TO ELIMINATE A TERM IN THE
C4------TARGET ROW.
   C = A(IB) * CI
   LD = LD + MBW

15 CONTINUE
20 CONTINUE
LB=LD-1

C
C5------MODIFY THE REST OF THE TERMS IN THE TARGET ROW.
   DO 10 K=J,MBW1
   LB=LB+1
   A(LB)=A(LB)-C*A(ID+K)
   10 CONTINUE
C
C6------SUBTRACT THE FACTOR TIMES A TERM IN THE PIVOT ROW
C6------FROM THE CORRESPONDING COLUMN IN THE TARGET ROW.
   LB=LB+1
   A(LB)=A(LB)-C*B(I)
   15 CONTINUE
   20 CONTINUE
   ID=ID-MBW
C
C7------MODIFY THE RIGHT SIDE OF THE EQUATION CORRESPONDING
C7------TO THE TARGET ROW.
   B(I+J)=B(I+J)-C*B(I)
   15 CONTINUE
   20 CONTINUE
   ID=ID-MBW
C
C8------SOLVE THE LAST EQUATION.
   B(N)=B(N)/A(ID)
C
C9------WORKING BACKWARDS SOLVE THE REST OF THE EQUATIONS.
   DO 70 I=1,NM1
   ID=ID-MBW
   70 CONTINUE
C
C10------CLEAR THE ACCUMULATOR SUM.
   SUM=0.0
   L=N-I
   MBW1M=MINO(MBW1,I)
C
C11------ADD THE KNOWN TERMS IN EQUATION L TO SUM.
   DO 60 J=1,MBW1M
   SUM=SUM+A(ID+J)*B(L+J)
   60 CONTINUE
C
C12------SOLVE FOR THE ONE UNKNOWN IN EQUATION L.
   B(L)=(B(L)-SUM)/A(ID)
   70 CONTINUE
C
C13------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Package</td>
<td>DIMENSION (MBW, NSLICE), Compressed coefficient matrix for a slice.</td>
</tr>
<tr>
<td>B</td>
<td>Package</td>
<td>DIMENSION (N), Right-hand-side vector.</td>
</tr>
<tr>
<td>C</td>
<td>Module</td>
<td>Factor needed to eliminate a term in the target row.</td>
</tr>
<tr>
<td>Cl</td>
<td>Module</td>
<td>Inverse of pivot.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows in the SSOR matrix (not the grid).</td>
</tr>
<tr>
<td>IB</td>
<td>Module</td>
<td>Index for elements to the right of the pivot.</td>
</tr>
<tr>
<td>ID</td>
<td>Module</td>
<td>Index of pivots.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>L</td>
<td>Module</td>
<td>Index for equations.</td>
</tr>
<tr>
<td>LB</td>
<td>Module</td>
<td>Index for elements in the target row to the right of the main diagonal.</td>
</tr>
<tr>
<td>LD</td>
<td>Module</td>
<td>Index of the main diagonal elements in the target rows.</td>
</tr>
<tr>
<td>MBW</td>
<td>Package</td>
<td>Maximum bandwidth of the coefficient matrix +1.</td>
</tr>
<tr>
<td>MBW1</td>
<td>Module</td>
<td>Maximum bandwidth of the coefficient matrix.</td>
</tr>
<tr>
<td>MBW1M</td>
<td>Module</td>
<td>Maximum possible number of the nonzero elements to the right of the main diagonal.</td>
</tr>
<tr>
<td>N</td>
<td>Package</td>
<td>Number of equations to be solved.</td>
</tr>
<tr>
<td>NA</td>
<td>Package</td>
<td>One-dimension size of compressed matrix &quot;A.&quot;</td>
</tr>
<tr>
<td>NM1</td>
<td>Module</td>
<td>N-1.</td>
</tr>
<tr>
<td>SUM</td>
<td>Module</td>
<td>In back substitution--in equation L, sum of terms to the right of the main diagonal term (L,L).</td>
</tr>
</tbody>
</table>
Utility modules are those submodules which perform general tasks common to several different packages. The name of a utility module always consists of a "U" followed by a five-character mnemonic. There are eight utility modules:

- **UBUDSV**: Writes an unformatted record consisting of an array with one real number for each cell in the grid.
- **ULASAV**: Writes an unformatted record consisting of an array with one real number for each cell in a layer.
- **ULAPRS** and **ULAPRW**: Prints one two-dimensional array which contains one real number for each cell in a layer. ULAPRS prints, in strip form, the first $N$ columns (where $N$ is the number of values that can fit on one print line) of each row and then the next $N$ columns, etc., until all columns of each row are printed (fig. 56). ULAPRW prints, in wrap form, all of row 1, all of row 2, and all of row 3, etc. The format for printing arrays is shown in table 2.
- **UCOLNO**: Prints column numbers at the top of each page of data printed by ULAPRS and ULAPRW.
- **U2DREL**: Reads a two-dimensional array of real numbers.
- **U2DINT**: Reads a two-dimensional array of integers.
- **U1DREL**: Reads a one-dimensional array of real numbers.
Figure 56.—Illustration of wrap and strip forms of printed output for a layer containing 7 rows and 17 columns.
<table>
<thead>
<tr>
<th>IPRN</th>
<th>FORMAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11G10.3</td>
</tr>
<tr>
<td>2</td>
<td>9G13.6</td>
</tr>
<tr>
<td>3</td>
<td>15F7.1</td>
</tr>
<tr>
<td>4</td>
<td>15F7.2</td>
</tr>
<tr>
<td>5</td>
<td>15F7.3</td>
</tr>
<tr>
<td>6</td>
<td>15F7.4</td>
</tr>
<tr>
<td>7</td>
<td>20F5.0</td>
</tr>
<tr>
<td>8</td>
<td>20F5.1</td>
</tr>
<tr>
<td>9</td>
<td>20F5.2</td>
</tr>
<tr>
<td>10</td>
<td>20F5.3</td>
</tr>
<tr>
<td>11</td>
<td>20F5.4</td>
</tr>
<tr>
<td>12</td>
<td>10G11.4</td>
</tr>
</tbody>
</table>
Input Instructions For Array Reading Utility Modules

The real two-dimensional array reader (U2DREL), the integer two-dimensional array reader (U2DINT), and the real one-dimensional array reader (UI1DREL) read one array-control record and, optionally, a data array in a format specified on the array-control record. The control record is read from the input unit number specified for the major option that is requesting the array. For example, the Recharge Package uses U2DREL to read the RECH array. The input unit for the recharge option is contained in IUNIT (8), and accordingly, the RECH array control record is read on this input unit.

FOR REAL ARRAY READER (U2DREL or UI1DREL)

Data: LOCAT  CNSTNT  FMTIN  IPRN
Format: I10  F10.0  5A4  I10

FOR INTEGER ARRAY READER (U2DINT)

Data: LOCAT  ICONST  FMTIN  IPRN
Format: I10  I10  5A4  I10

Explanation of Fields Used in Input Instructions

LOCAT—indicates the location of the data which will be put in the array.

If LOCAT < 0, the sign is reversed to give the unit number from which an unformatted record will be read.

If LOCAT = 0, every element in the array will be set equal to the value CNSTNT/ICONST.

If LOCAT > 0, it is the unit number from which data values will be read in the format specified in the third field of the array-control record (FMTIN).

CNSTNT/ICONST—is a constant. Its use depends on the value of LOCAT.

If LOCAT = 0, every element in the array is set equal to CNSTNT/ICONST.

If LOCAT ≠ 0, and if CNSTNT/ICONST ≠ 0, every element in the array is multiplied by CNSTNT/ICONST.

FMTIN—is the format of records containing the array values. It is used only if the first field in the array-control record (LOCAT) contains a positive number. The format must be enclosed in parentheses; for example, (15F5.0) for real data and (15I5) for integer data.
IPRN—isa flag indicating that the array being read should be printed and a code for indicating the format that should be used. It is used only if LOCAT is not equal to zero. The format codes are different for each of the three modules. IPRN is set to zero when the specified value exceeds those defined in the chart below. If IPRN is less than zero, the array will not be printed.

<table>
<thead>
<tr>
<th>IPRN</th>
<th>U2DREL</th>
<th>U2DINT</th>
<th>U1DREL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10G11.4</td>
<td>10I11</td>
<td>10G12.5</td>
</tr>
<tr>
<td>1</td>
<td>11G10.3</td>
<td>60I1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>9G13.6</td>
<td>40I2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>15F7.1</td>
<td>30I3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15F7.2</td>
<td>25I4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>15F7.3</td>
<td>20I5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>15F7.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>20F5.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>20F5.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>20F5.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>20F5.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>20F5.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>10G11.4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Utility module UBUDSV writes an unformatted record consisting of an array dimensioned (NCOL, NROW, NLAY). The record containing the array is preceded by an unformatted record containing identifying information. The identifying information consists of:

- **KSTP**: current time step, integer, 1 word
- **KPER**: current stress period, integer, 1 word
- **TEXT**: label, character string, 4 words
- **NCOL**: number of columns, integer, 1 word
- **NROW**: number of rows, integer, 1 word
- **NLAY**: number of layers, integer, 1 word

Documentation of this module consists only of comments in the program and a list of variables.
SUBROUTINE UBUDSV(KSTP, KPER, TEXT, IBDCHN, BUFF, NCOL, NROW, NLAY, IOUT)

C-----VERSION 1637 12MAY1987 UBUDSV
C *********************************************************
C RECORD CELL-BY-CELL FLOW TERMS FOR ONE COMPONENT OF FLOW.
C *********************************************************

C SPECIFICATIONS:
C

C CHARACTER*4 TEXT
C DIMENSION TEXT(4), BUFF(NCOL, NROW, NLAY)

C C1------WRITE AN UNFORMATTED RECORD CONTAINING IDENTIFYING
C INFORMATION.
C WRITE(IOUT,1) TEXT, IBDCHN, KSTP, KPER
1 FORMAT(1X, '"', 4A4, '" BUDGET VALUES WILL BE SAVED ON UNIT', I3, 
     1 ' AT END OF TIME STEP', I3, ', STRESS PERIOD', I3)

C WRITE(IBDCHN) KSTP, KPER, TEXT, NCOL, NROW, NLAY

C C2------WRITE AN UNFORMATTED RECORD CONTAINING VALUES FOR
C EACH CELL IN THE GRID. THE ARRAY IS DIMENSIONED
C2----- (NCOL, NROW, NLAY)
C WRITE(IBDCHN) BUFF

C C3------RETURN
C RETURN
END
<table>
<thead>
<tr>
<th>Variation</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFF</td>
<td>Global</td>
<td>DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.</td>
</tr>
<tr>
<td>IBDCHN</td>
<td>Module</td>
<td>Unit number on which the array will be recorded.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NLAY</td>
<td>Global</td>
<td>Number of layers in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
</tbody>
</table>
Utility module ULASAV writes an unformatted record consisting of an array dimensioned (NCOL, NROW). The record containing the array is preceded by an unformatted record containing identifying information. The identifying information consists of:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Data Type</th>
<th>Word Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSTP</td>
<td>current time step</td>
<td>integer</td>
<td>1 word</td>
</tr>
<tr>
<td>KPER</td>
<td>current stress period</td>
<td>integer</td>
<td>1 word</td>
</tr>
<tr>
<td>PERTIM</td>
<td>elapsed time in the current stress period</td>
<td>real</td>
<td>1 word</td>
</tr>
<tr>
<td>TOTIM</td>
<td>elapsed time in the simulation</td>
<td>real</td>
<td>1 word</td>
</tr>
<tr>
<td>TEXT</td>
<td>label</td>
<td>character string</td>
<td>4 words</td>
</tr>
<tr>
<td>NCOL</td>
<td>number of columns</td>
<td>integer</td>
<td>1 word</td>
</tr>
<tr>
<td>NROW</td>
<td>number of rows</td>
<td>integer</td>
<td>1 word</td>
</tr>
<tr>
<td>ILAY</td>
<td>layer number</td>
<td>integer</td>
<td>1 word</td>
</tr>
</tbody>
</table>
SUBROUTINE ULASAV (BUF, TEXT, KSTP, KPER, PERTIM, TOTIM, NCOL,
NROW, ILAY, ICHN)

C-----VERSION 1642 12MAY1987 ULASAV
C *************************************************
C SAVE 1 LAYER ARRAY ON DISK
C *************************************************
C
C SPECIFICATIONS:
C ------------------------------------------------------------------
CHARACTER*4 TEXT
DIMENSION BUF(NCOL,NROW), TEXT(4)
C ------------------------------------------------------------------

C1------WRITE AN UNFORMATTED RECORD CONTAINING IDENTIFYING
C1------INFORMATION.
WRITE(ICHN) KSTP, KPER, PERTIM, TOTIM, TEXT, NCOL, NROW, ILAY
C
C2------WRITE AN UNFORMATTED RECORD CONTAINING ARRAY VALUES
C2------THE ARRAY IS DIMENSIONED (NCOL, NROW)
WRITE(ICHN) ((BUF(IC,IR), IC=1, NCOL), IR=1, NROW)
C
C3------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUF</td>
<td>Module</td>
<td>Buffer containing data to be printed or recorded.</td>
</tr>
<tr>
<td>IC</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>ICHN</td>
<td>Module</td>
<td>Unit number on which the array is to be recorded.</td>
</tr>
<tr>
<td>ILAY</td>
<td>Module</td>
<td>Layer number.</td>
</tr>
<tr>
<td>IR</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>PERTIM</td>
<td>Package</td>
<td>Elapsed time during the current stress period.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
<tr>
<td>TOTIM</td>
<td>Package</td>
<td>Elapsed time in the simulation.</td>
</tr>
</tbody>
</table>
Narrative for Module ULAPRS

Module ULAPRS prints a two-dimensional array in strip form (fig. 56) using one of twelve FORTRAN formats. Module ULAPRS performs its tasks in the following order:

1. Get the format code (IP). If it is less than 1 or greater than 12, set it equal to 12 (the default).

2. Use the format code (IP) to determine the number of values (NCAP) to be printed on one line.

3. Calculate the number of spaces used for each value (NCPF) and the number of strips (NSTRIP). Initialize the fields to store the first column (J1) and the last column (J2) for each strip.

4. Loop through the strips (DO STEPS 5-8).

5. Calculate the first (J1) and last (J2) column for this strip.

6. Print a title on each strip.

7. Call module UCOLNO to print the column numbers above each strip.

8. Loop through the rows printing columns J1 through J2 using the appropriate format (IP).

9. RETURN.
IPRN is a code indicating the format to be used in printing array values. If it is not between 1 and 12, it is set equal to 12.
SUBROUTINE ULAPRS(BUF, TEXT, KSTP, KPER, NCOL, NROW, ILAY, IPRN, IOUT)
C
C VERSION 1640 12MAY1987 ULAPRS
C
C -------------------------------ULAPRS----------------------------------
C
C PRINT A 1 LAYER ARRAY IN STRIPS
C
C -------------------------------ULAPRS----------------------------------
C
C SPECIFICATIONS:
C
C
C CHARACTER*4 TEXT
C DIMENSION BUF(NCOL,NROW), TEXT(4)
C
C
C1------MAKE SURE THE FORMAT CODE (IP OR IPRN) IS BETWEEN 1
C1------AND 12.
C1 IF(IP.LT.1 .OR. IP.GT.12) IP=12
C
C2------DETERMINE THE NUMBER OF VALUES (NCAP) PRINTED ON ONE LINE.
IF(IP.EQ.1) NCAP=11
IF(IP.EQ.2) NCAP=9
IF(IP.GT.2 .AND. IP.LT.7) NCAP=15
IF(IP.GT.6 .AND. IP.LT.12) NCAP=20
IF(IP.EQ.12) NCAP=10
C
C3------CALCULATE THE NUMBER OF STRIPS (NSTRIP).
NCPF=129/NCAP
ISP=0
IF(NCAP.GT.12) ISP=3
NSTRIP=(NCOL-1)/NCAP + 1
J1=1-NCAP
J2=0
C
C4------LOOP THROUGH THE STRIPS.
DO 2000 N=1,NSTRIP
C
C5------CALCULATE THE FIRST(J1) & THE LAST(J2) COLUMNS FOR THIS STRIP
J1=J1+NCAP
J2=J2+NCAP
IF(J2.GT.NCOL) J2=NCOL
C
C6------PRINT TITLE ON EACH STRIP
WRITE(IOUT,1) TEXT, ILAY, KSTP, KPER
1 FORMAT('IN LAYER',I3,' AT END OF TIME STEP',I3,
1 ' IN STRESS PERIOD',I3/11X,H71('-'))
C
C7------PRINT COLUMN NUMBERS ABOVE THE STRIP
CALL UCOLNO(J1,J2,ISP,NCAP,NCPF,IOUT)
C
C8------LOOP THROUGH THE ROWS PRINTING COLS J1 THRU J2 WITH FORMAT IP
DO 1000 I=1,NROW
GO TO(10,20,30,40,50,60,70,80,90,100,110,120, IP
C
C FORMAT 10610.3
10 WRITE(IOUT,11) I,(BUF(J,I),J=J1,J2)
11 FORMAT(1H0,I3,2X,PG10.3,10(1X,G10.3))
GO TO 1000
C
C Format 8G13.6
20 WRITE(IOUT,21) I,(BUF(J,I),J=J1,J2)
21 FORMAT(1HO,I3,2X,1PG13.6,8(1X,G13.6))
    GO TO 1000
C
C--------------FORMAT 15F7.1
30 WRITE(IOUT,31) I,(BUF(J,I),J=J1,J2)
31 FORMAT(1HO,I3,1X,15(1X,F7.1))
    GO TO 1000
C
C--------------FORMAT 15F7.2
40 WRITE(IOUT,41) I,(BUF(J,I),J=J1,J2)
41 FORMAT(1HO,I3,1X,15(1X,F7.2))
    GO TO 1000
C
C--------------FORMAT 15F7.3
50 WRITE(IOUT,51) I,(BUF(J,I),J=J1,J2)
51 FORMAT(1HO,I3,1X,15(1X,F7.3))
    GO TO 1000
C
C--------------FORMAT 20F5.0
60 WRITE(IOUT,61) I,(BUF(J,I),J=J1,J2)
61 FORMAT(1HO,I3,1X,20(1X,F5.0))
    GO TO 1000
C
C--------------FORMAT 20F5.1
70 WRITE(IOUT,71) I,(BUF(J,I),J=J1,J2)
71 FORMAT(1HO,I3,1X,20(1X,F5.1))
    GO TO 1000
C
C--------------FORMAT 20F5.2
80 WRITE(IOUT,81) I,(BUF(J,I),J=J1,J2)
81 FORMAT(1HO,I3,1X,20(1X,F5.2))
    GO TO 1000
C
C--------------FORMAT 20F5.3
90 WRITE(IOUT,91) I,(BUF(J,I),J=J1,J2)
91 FORMAT(1HO,I3,1X,20(1X,F5.3))
    GO TO 1000
C
C--------------FORMAT 20F5.4
100 WRITE(IOUT,101) I,(BUF(J,I),J=J1,J2)
101 FORMAT(1HO,I3,1X,20(1X,F5.4))
    GO TO 1000
C
C--------------FORMAT 9G11.4
110 WRITE(IOUT,111) I,(BUF(J,I),J=J1,J2)
111 FORMAT(1HO,I3,2X,1PG11.4,9(1X,G11.4))
C
1000 CONTINUE
2000 CONTINUE
C
C9------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUF</td>
<td>Module</td>
<td>Buffer containing data to be printed or recorded.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>ILAY</td>
<td>Module</td>
<td>Layer number.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Module</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IP</td>
<td>Module</td>
<td>Format code.</td>
</tr>
<tr>
<td>IPRN</td>
<td>Module</td>
<td>Code for the format to be used when printing arrays.</td>
</tr>
<tr>
<td>ISP</td>
<td>Module</td>
<td>Number of spaces.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>J1</td>
<td>Module</td>
<td>First column in a strip.</td>
</tr>
<tr>
<td>J2</td>
<td>Module</td>
<td>Last column in a strip.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>N</td>
<td>Module</td>
<td>Index for strips.</td>
</tr>
<tr>
<td>NCAP</td>
<td>Module</td>
<td>Number of columns on a line.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NCPF</td>
<td>Module</td>
<td>Number of columns per field.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>NSTRIP</td>
<td>Module</td>
<td>Number of strips.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
</tbody>
</table>
Module ULAPRW prints a two-dimensional array in wrap form (fig. 56) using one of twelve FORTRAN formats. Module ULAPRW performs its tasks in the following order:

1. Print a header.

2. Set the format code (IP). If it is less than 1 or greater than 12, set it equal to 12 (the default).

3. Call the module UCOLNO to print column numbers.

4. Loop through the rows printing each one in its entirety using the appropriate format code.

5. RETURN.
IPRN is a code indicating the format to be used in printing array values. If it is not between 1 and 12, it is set equal to 12.
SUBROUTINE ULAPRW(BUF, TEXT, KSTP, KPER, NCOL, NROW, ILAY, IPRN, IOUT)

C

C-----VERSION 1642 12MAY1987 ULAPRW
C ********************************************
C PRINT 1 LAYER ARRAY
C ********************************************
C

C SPECIFICATIONS:
-----------------------------------------------
CHARACTER*4 TEXT
DIMENSION BUF(NCOL, NROW), TEXT(4)
-----------------------------------------------
C
C1------PRINT A HEADER
 IF(ILAY.LE.0) GO TO 5
 WRITE(IOUT,1) TEXT, ILAY, KSTP, KPER
 1 FORMAT(1H1,10X,4A4,' IN LAYER', I3, ' AT END OF TIME STEP', I3,
 1 ' IN STRESS PERIOD', I3/11X,71('-'1)

C2------MAKE SURE THE FORMAT CODE (IP OR IPRN) IS BETWEEN 1 AND 12.
  5 IP=IPRN
     IF(IP.LT.1 .OR. IP.GT.12) IP=12

C3------CALL THE UTILITY MODULE UCOLNO TO PRINT COLUMN NUMBERS.
     IF(IP.EQ.1) CALL UCOLNO(1, NCOL, 0, 11, 11, IOUT)
     IF(IP.EQ.2) CALL UCOLNO(1, NCOL, 0, 9, 14, IOUT)
     IF(IP.GT.2 .AND. IP.LT.7) CALL UCOLNO(1, NCOL, 3, 15, 8, IOUT)
     IF(IP.GT.6 .AND. IP.LT.12) CALL UCOLNO(1, NCOL, 3, 20, 6, IOUT)
     IF(IP.EQ.12) CALL UCOLNO(1, NCOL, 0, 10, 12, IOUT)

C4------LOOP THROUGH THE ROWS PRINTING EACH ONE IN ITS ENTIRETY.
    DO 1000 I=1, NROW
     GO TO(10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, IP

C ---------------- FORMAT 11G10.3
  10 WRITE(IOUT,11) I, (BUF(J,I), J=1, NCOL)
     11 FORMAT(1H0,13,1X,11(1X,G10.3)/(5X,11(1X,G10.3)))
     GO TO 1000

C ---------------- FORMAT 9G13.6
  20 WRITE(IOUT,21) I, (BUF(J,I), J=1, NCOL)
     21 FORMAT(1H0,13,1X,19(1X,G13.6)/(5X,9(1X,G13.6)))
     GO TO 1000

C ---------------- FORMAT 15F7.1
  30 WRITE(IOUT,31) I, (BUF(J,I), J=1, NCOL)
     31 FORMAT(1H0,13,1X,15(1X,F7.1)/(5X,15(1X,F7.1)))
     GO TO 1000

14-19
C ------------ FORMAT 15F7.2
  40 WRITE(IOUT,41) I, (BUF(J,I), J=1, NCOL)
  41 FORMAT(1HO, I3, 1X, 15(1X, F7.2)/(5X, 15(1X, F7.2)))
     GO TO 1000
C ------------ FORMAT 15F7.3
  50 WRITE(IOUT,51) I, (BUF(J,I), J=1, NCOL)
  51 FORMAT(1HO, I3, 1X, 15(1X, F7.3)/(5X, 15(1X, F7.3)))
     GO TO 1000
C ------------ FORMAT 15F7.4
  60 WRITE(IOUT,61) I, (BUF(J,I), J=1, NCOL)
  61 FORMAT(1HO, I3, 1X, 15(1X, F7.4)/(5X, 15(1X, F7.4)))
     GO TO 1000
C ------------ FORMAT 20F5.0
  70 WRITE(IOUT,71) I, (BUF(J,I), J=1, NCOL)
  71 FORMAT(1HO, I3, 1X, 20(1X, F5.0)/(5X, 20(1X, F5.0)))
     GO TO 1000
C ------------ FORMAT 20F5.1
  80 WRITE(IOUT,81) I, (BUF(J,I), J=1, NCOL)
  81 FORMAT(1HO, I3, 1X, 20(1X, F5.1)/(5X, 20(1X, F5.1)))
     GO TO 1000
C ------------ FORMAT 20F5.2
  90 WRITE(IOUT,91) I, (BUF(J,I), J=1, NCOL)
  91 FORMAT(1HO, I3, 1X, 20(1X, F5.2)/(5X, 20(1X, F5.2)))
     GO TO 1000
C ------------ FORMAT 20F5.3
 100 WRITE(IOUT,101) I, (BUF(J,I), J=1, NCOL)
101 FORMAT(1HO, I3, 1X, 20(1X, F5.3)/(5X, 20(1X, F5.3)))
     GO TO 1000
C ------------ FORMAT 20F5.4
 110 WRITE(IOUT,111) I, (BUF(J,I), J=1, NCOL)
111 FORMAT(1HO, I3, 1X, 20(1X, F5.4)/(5X, 20(1X, F5.4)))
     GO TO 1000
C ------------ FORMAT 10G11.4
 120 WRITE(IOUT,121) I, (BUF(J,I), J=1, NCOL)
121 FORMAT(1HO, I3, 2X, 1PG11.4, 9(1X, G11.4)/(5X, 10(1X, G11.4)))
C 1000 CONTINUE
C 55------RETURN
   RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUF</td>
<td>Module</td>
<td>Buffer containing data to be printed or recorded.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>ILAY</td>
<td>Module</td>
<td>Layer number.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IP</td>
<td>Module</td>
<td>Format code.</td>
</tr>
<tr>
<td>IPRN</td>
<td>Module</td>
<td>Code for the format to be used when printing arrays.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>KPER</td>
<td>Global</td>
<td>Stress period counter.</td>
</tr>
<tr>
<td>KSTP</td>
<td>Global</td>
<td>Time step counter. Reset at the start of each stress period.</td>
</tr>
<tr>
<td>NCOL</td>
<td>Global</td>
<td>Number of columns in the grid.</td>
</tr>
<tr>
<td>NROW</td>
<td>Global</td>
<td>Number of rows in the grid.</td>
</tr>
<tr>
<td>TEXT</td>
<td>Module</td>
<td>Label to be printed or recorded with the array data.</td>
</tr>
</tbody>
</table>
Module UCOLNO prints column numbers at the top of a page when arrays of numbers are printed. It performs its functions in the following order:

1. Calculate the number of columns to be printed (NLBL), the width of a line (NTOT), and the number of lines needed to print all of the column numbers (NWRAP). Initialize the fields J1 and J2 which contain the first and last column number on each print line.

2. Build and print each line (DO STEPS 3-6).

3. Clear the line buffer (BF) in which the line is built.

4. Determine the first (J1) and last (J2) column number for the current line.

5. Put the column numbers in the line buffer. They are selected from the array DG. The indices I1, I2, and I3 point to the units digit, tens digit, and hundreds digit, respectively.

6. Print the line.

7. Print a line of dots.

8. RETURN.
Flow Chart for Module UCOLNO

1. ENTER UCOLNO

2. CALCULATE THE NUMBER OF LINES NEEDED TO PRINT ALL OF THE COLUMN NUMBERS

3. CLEAR THE BUFFER IN WHICH THE LINE IS ASSEMBLED

4. DETERMINE THE RANGE OF COLUMN NUMBERS WHICH GO IN THE LINE

5. MOVE THE COLUMN NUMBERS INTO THE BUFFER

6. PRINT THE CONTENTS OF THE BUFFER

7. PRINT A LINE OF DOTS

8. RETURN
SUBROUTINE UCOLNO(NLBL1, NLBL2, NSPACE, NCPL, NDIG, IOUT)

C
C-----VERSION 1638 12MAY1987 UCOLNO
C
C ***********~**C****************************~~******~**************
C OUTPUT COLUMN NUMBERS ABOVE A MATRIX PRINTOUT
C NLBL1 IS THE START COLUMN LABEL (NUMBER)
C NLBL2 IS THE STOP COLUMN LABEL (NUMBER)
C NSPACE IS NUMBER OF BLANK SPACES TO LEAVE AT START OF LINE
C NCPL IS NUMBER OF COLUMN NUMBERS PER LINE
C NDIG IS NUMBER OF CHARACTERS IN EACH COLUMN FIELD
C IOUT IS OUTPUT CHANNEL
C
C ************************************************************
C
C SPECIFICATIONS:
C
C DATA DOT,SPACE,DG,BF
C DIMENSION BF(130),DG(10)
C
C DATA DG(1),DG(2),DG(3),DG(4),DG(5),DG(6),DG(7),DG(8),DG(9),DG(10)/
1   '0','1','2','3','4','5','6','7','8','9' /
2   '/
C------------------------------------------------------------------------------
C
15 CL-----CALCULATE # OF COLUMNS TO BE PRINTED (NLBL), WIDTH
C15 OF A LINE (NTOT), NUMBER OF LINES (NWREP).
WRITE(IOUT,1)  
1 FORMAT(1X)  
  NLBL=NLBL2-NLBL1+1
  N=NLBL
  IF(NLBL.GT.NCPL) N=NCPL
  NTOT=NSPACE+N*NDIG
  IF(NTOT.GT.130) GO TO 50
  NWREP=(NLBL-1)/NCPL + 1
  J1=NLBL1-NCPL
  J2=NLBL1-1

C2------BUILD AND PRINT EACH LINE
DO 40 I=1,NWREP
C
C3------CLEAR THE BUFFER (BF).
   DO 20 I=1,130
   BF(I)=SPACE
   20 CONTINUE
   NBF=NSPACE

C4------DETERMINE FIRST (J1) AND LAST (J2) COLUMN # FOR THIS LINE.
   J1=J1+NCPL
   J2=J2+NCPL
   IF(J2.GT.NLBL2) J2=NLBL2

C5------LOAD THE COLUMN #S INTO THE BUFFER,
   DO 30 J=J1,J2
   NBF=NBF+NDIG
   I2=J/10
   I1=J-I2*10+1
   BF(NBF)=DG(I1)
   IF(I2.EQ.0) GO TO 30
   I3=I2/10
   I2=I2-I3*10+1
   BF(NBF+1)=DG(I2)
   IF(I3.EQ.0) GO TO 30
   BF(NBF+2)=DG(I3+1)
   30 CONTINUE

   WRITE(IOUT,31) (BF(I),I=1,NBF)
   31 FORMAT(1X,130Al)
   40 CONTINUE
C
C7------PRINT A LINE OF DOTS (FOR ESTHETIC PURPOSES ONLY).
   50 NTOT=NTOT+5
   IF(NTOT.GT.130) NTOT=130
   WRITE(IOUT,51) (DOT,I=1,NTOT)
   51 FORMAT(1X,130Al)

C8------RETURN
RETURN
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF</td>
<td>Module</td>
<td>DIMENSION (130), Buffer in which a line is assembled.</td>
</tr>
<tr>
<td>DG</td>
<td>Module</td>
<td>DIMENSION (10), Digits 0 through 9.</td>
</tr>
<tr>
<td>DOT</td>
<td>Module</td>
<td>Field containing a period.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for BF.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>I1</td>
<td>Module</td>
<td>Index for DG (units digit).</td>
</tr>
<tr>
<td>I2</td>
<td>Module</td>
<td>Index for DG (tens digit).</td>
</tr>
<tr>
<td>I3</td>
<td>Module</td>
<td>Index for DG (hundreds digit).</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for column numbers.</td>
</tr>
<tr>
<td>J1</td>
<td>Module</td>
<td>First column number on the current line.</td>
</tr>
<tr>
<td>J2</td>
<td>Module</td>
<td>Last column number on the current line.</td>
</tr>
<tr>
<td>N</td>
<td>Module</td>
<td>Number of column numbers.</td>
</tr>
<tr>
<td>NBF</td>
<td>Module</td>
<td>Index for BF.</td>
</tr>
<tr>
<td>NCPL</td>
<td>Module</td>
<td>Number of column numbers per line.</td>
</tr>
<tr>
<td>NDIG</td>
<td>Module</td>
<td>Number of characters in each column number field.</td>
</tr>
<tr>
<td>NLBL</td>
<td>Module</td>
<td>Number of column numbers to be printed.</td>
</tr>
<tr>
<td>NLBL1</td>
<td>Module</td>
<td>Start column number.</td>
</tr>
<tr>
<td>NLBL2</td>
<td>Module</td>
<td>Stop column number.</td>
</tr>
<tr>
<td>NSPACE</td>
<td>Module</td>
<td>Number of blank spaces at start of line.</td>
</tr>
<tr>
<td>NTOT</td>
<td>Module</td>
<td>Total number of characters on a line.</td>
</tr>
<tr>
<td>NWRAP</td>
<td>Module</td>
<td>Number of lines needed in wrap format.</td>
</tr>
<tr>
<td>SPACE</td>
<td>Module</td>
<td>Field containing blanks.</td>
</tr>
</tbody>
</table>
Module U2DREL reads values for a two-dimensional real array. First it reads an "array-control record." Then, based on the contents of the array-control record, it may read array values. The array-control record contains four fields: location (LOCAT), constant (CNSTNT), format (FMTIN), and printout indicator (IPRN). The LOCAT field determines where array values will come from. If LOCAT is positive, it is the unit number from which array values will be read in the format specified in FMTIN. If LOCAT is negative, the sign is reversed to give the unit number from which an unformatted record containing the array values will be read. (Before the array record is read, a record will be read and ignored. Thus output from the module ULASAV can be read.) If LOCAT is zero, all of the array values will be set equal to CNSTNT. When LOCAT is not zero and CNSTNT is not zero, the array values will be multiplied by the value of CNSTNT. The field IPRN contains a code number for a FORTRAN format to be used when printing the array.

Module U2DREL performs its tasks in the following order:

1. Read the array-control record (LOCAT, CNSTNT, FMTIN, and IPRN).

2. Use LOCAT to determine where the array values are coming from. GO TO STEPS 3, 4, OR 5.

3. If LOCAT equals zero, set all array values equal to CNSTNT, print a message to that effect, and RETURN.

4. If LOCAT is greater than zero, read array values according to the format in FMTIN. GO TO STEP 6.

5. If LOCAT is less than zero, read an unformatted dummy record and then read an unformatted record containing the array values. GO TO STEP 6.

6. If CNSTNT is not equal to zero, multiply array values by CNSTNT.

7. If IPRN is greater than or equal to zero, call utility module ULAPRW to print the array values using IPRN as the format code.

8. RETURN.
Array Control Record controls the input of array values. It contains four fields: LOCAT, CNSTNT, FMTIN, and IPRN.

LOCAT is a code showing where array values will come from.

If LOCAT < 0, array values will be read from an unformatted record from a unit number equal to -LOCAT.

If LOCAT = 0, array values will be set equal to CNSTNT.

If LOCAT > 0, array values will be read from the unit number equal to LOCAT in the format specified in FMTIN.

CNSTNT is the constant to which all array values are set if LOCAT is equal to zero, and it is the constant by which all array values are multiplied if LOCAT is not equal to zero.

FMTIN is the format in which array values are read if LOCAT is greater than zero.

IPRN is a code showing the format to be used if array values are to be printed.
SUBROUTINE U2DREL(A, ANAME, II, JJ, K, IN, IOUT)
C
C-----VERSION 1648 13MAY1987 U2DREL
C
C ROUTINE TO INPUT 2-D REAL DATA MATRICES
A IS ARRAY TO INPUT
ANAME IS 24 CHARACTER DESCRIPTION OF A
II IS NO. OF ROWS
JJ IS NO. OF COLS
K IS LAYER NO. (USED WITH NAME TO TITLE PRINTOUT UNLESS K IS 0)
IN IS INPUT UNIT
IOUT IS OUTPUT UNIT
C
C SPECIFICATIONS:
C
C
CHARACTER*4 ANAME
CHARACTER*20 FMTIN
DIMENSION A(JJ,II),ANAME(6)
C
C1-----READ ARRAY CONTROL RECORD.
READ (IN,1) LOCAT,CNSTNT,FMTIN,IPRN
1 FORMAT(I10,F10.0,A20,I10)
C
C2-----USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
IF (LOCAT) 200,50,90
C
C3-----IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN
50 DO 80 I=1,II
80 DO 80 J=1,JJ
80 A(J,I)=CNSTNT
IF(K.GT.0) WRITE(IOUT,2) ANAME,CNSTNT,K
2 FORMAT(I30,52X,6A4,' =',G15.7,' FOR LAYER',I3)
IF(K.LE.0) WRITE(IOUT,3) ANAME,CNSTNT
3 FORMAT(I30,52X,6A4,' =',G15.7)
RETURN
C
C4-----IF LOCAT>0 THEN READ FORMATTED RECORDS USING FORMAT FMTIN.
90 IF(K.GT.0) WRITE(IOUT,4) ANAME,K,LOCAT,FMTIN
4 FORMAT(I30,///30X,6A4,' WILL BE READ ON UNIT*',
1 13,' USING FORMAT: ',A20,96('-')
IF(K.LE.0) WRITE(IOUT,5) ANAME,LOCAT,FMTIN
5 FORMAT(I30,///30X,6A4,' WILL BE READ ON UNIT*',
1 13,' USING FORMAT: ',A20,83('-')
DO 100 I=1,II
READ (LOCAT,FMTIN) (A(J,I),J=1,JJ)
100 CONTINUE
GO TO 300
C
C5-----LOCAT<0 THEN READ UNFORMATTED RECORD CONTAINING ARRAY VALUES
200 LOCAT=-LOCAT
201 IF(K.GT.0) WRITE(IOUT,201) ANAME,K,LOCAT
201 FORMAT(I30,///30X,6A4,' WILL BE READ UNFORMATTED ON UNIT*',
1 13,' WILL BE READ UNFORMATTED ON UNIT*',
1 13,' USING FORMAT: ',A20,83('-')
202 FORMAT(I30,///30X,6A4,' WILL BE READ UNFORMATTED ON UNIT*',
1 13,' WILL BE READ UNFORMATTED ON UNIT*',
1 13,' USING FORMAT: ',A20,60('-')
DO 100 I=1,II
READ(LOCAT) A
100 CONTINUE
C
C6-----IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.
300 IF(CNSTNT.EQ.0.) GO TO 320
DO 310 I=1,II
310 DO 310 J=1,JJ
A(J,I)=A(J,I)*CNSTNT
310 CONTINUE
C
C7-----IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
320 IF(IPRN.LT.0) RETURN
CALL ULAPRW(A,ANAME,O,O,JJ,II,O,IPRN,IOUT)
RETURN
C
C8-----RETURN
END
## List of Variables for Module U2DREL

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Module</td>
<td>DIMENSION (JJ,II), Input array.</td>
</tr>
<tr>
<td>ANAME</td>
<td>Module</td>
<td>Label for printout of the input array.</td>
</tr>
<tr>
<td>CNSTNT</td>
<td>Module</td>
<td>Constant to which all array values are set if LOCAT is equal to zero or by which all array values are multiplied if LOCAT is not equal to zero.</td>
</tr>
<tr>
<td>FMTIN</td>
<td>Module</td>
<td>Format under which array values will be read.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>II</td>
<td>Module</td>
<td>Number of rows.</td>
</tr>
<tr>
<td>IN</td>
<td>Module</td>
<td>Unit number from which the array control record will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPRN</td>
<td>Module</td>
<td>Code for format to be used when printing the arrays.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>JJ</td>
<td>Module</td>
<td>Number of columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Layer number.</td>
</tr>
<tr>
<td>LOCAT</td>
<td>Module</td>
<td>Location of values to fill in the array.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; 0, read an unformatted record containing the array values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 0, set all the array values equal to constant (CNSTNT).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&gt; 0, read the formatted records containing the array values.</td>
</tr>
</tbody>
</table>
Narrative for Module U2DINT

Module U2DINT reads values for a two-dimensional integer array. First it reads an "array-control record." Then, based on the contents of the array-control record, it may read array values. The array-control record contains four fields: location (LOCAT), constant (ICONST), format (FMTIN), and printout indicator (IPRN). The LOCAT field determines where array values will come from. If LOCAT is positive, it is the unit number from which array values will be read in the format specified in FMTIN. If LOCAT is negative, the sign is reversed to give the unit number from which an unformatted record containing the array values will be read. (Before the array record is read, a record will be read and ignored. Thus output from the module ULASAV can be read.) If LOCAT is zero, all of the array values will be set equal to ICONST. When LOCAT is not zero and ICONST is not zero, the array values will be multiplied by the value of ICONST. The field IPRN (table 1) contains a code number for a FORTRAN format to be used when printing the array.

Module U2DINT performs its tasks in the following order:

1. Read the array-control record (LOCAT, ICONST, FMTIN, and IPRN).

2. Use LOCAT to determine where the array is coming from. GO TO STEPS 3, 4, OR 5.

3. If LOCAT equals zero, set all array values equal to CNSTNT, print a message to that effect, and RETURN.

4. If LOCAT is greater than zero, read array values according to the format in FMTIN. GO TO STEP 6.

5. If LOCAT is less than zero, read an unformatted dummy record and then read an unformatted record containing the array values. GO TO STEP 6.

6. If ICONST is not equal to zero, multiply array values by ICONST.

7. If IPRN is greater than or equal to zero, print the array values using IPRN as the format code.

8. Call utility module UCOLNO to print column numbers at the top of the page.

9. Print each row in the array.

10. Select the format for printing.

11. RETURN.
Array Control Record controls the input of array values. It contains four fields: LOCAT, ICONST, FMTIN, and IPRN.

LOCAT is a code showing where array values will come from.

If LOCAT < 0, array values will be read from an unformatted record from a unit number equal to -LOCAT.

If LOCAT = 0, array values will be set equal to ICONST.

If LOCAT > 0, array values will be read from the unit number equal to LOCAT in the format specified in FMTIN.

ICONST is the constant to which all array values are set if LOCAT is equal to zero, and it is the constant by which all array values are multiplied if LOCAT is not equal to zero.

FMTIN is the format in which array values are read if LOCAT is greater than zero.

IPRN is a code showing the format to be used if array values are to be printed.
SUBROUTINE U2DINT(IA, ANAME, II, JJ, K, IN, IOUT)

C ---VERSION 1645 12MAY1987 UZDINT
C ******************************************************************
C ROUTINE TO INPUT 2-D INTEGER DATA MATRICES
C IA IS ARRAY TO INPUT
C ANAME IS 24 CHARACTER DESCRIPTION OF IA
C II IS NO. OF ROWS
C JJ IS NO. OF COLS
C K IS LAYER NO. (USED WITH NAME TO TITLE PRINTOUT UNLESS K IS 0)
C IN IS INPUT UNIT
C IOUT IS OUTPUT UNIT
C ******************************************************************
C SPECIFICATIONS:
C CHARACTER*4 ANAME
C CHARACTER*20 FMTIN
C DIMENSION IA(JJ,II),ANAME(6)
C---------------------------------------------------------------
C 
C C1------READ ARRAY CONTROL RECORD.
C READ (IN,1) LOCAT,ICONST,FMTIN,IPLY
1 FORMAT(I10,I10,A20,I10)
C 
C C2------USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
C IF (LOCAT) 200,50,90
C 
C C3------IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO ICONST. RETURN
50 DO 80 I=1,II
51 DO 80 J=1,JJ
80 IA(J,I)=ICONST
IF(K.GT.0) WRITE(IOUT,2) ANAME,ICONST,K
2 FORMAT(1H0,52X,6A4,' = ',I15,' FOR LAYER',I3)
IF(K.LE.0) WRITE(IOUT,3) ANAME,ICONST
3 FORMAT(1H0,52X,6A4,' = ',I15)
RETURN
C 
C C4------IF LOCAT>0 THEN READ FORMATTED RECORDS USING FORMAT FMTIN.
90 IF(K.GT.0) WRITE(IOUT,4) ANAME,K,LOCAT,FMTIN
4 FORMAT(1H0,30X,6A4,' FOR LAYER',I3,' WILL BE READ ON UNIT',I3,' USING FORMAT: ',A20/30X,96('-'))
IF(K.LE.0) WRITE(IOUT,5) ANAME,LOCAT,FMTIN
5 FORMAT(1H0,30X,6A4,' WILL BE READ ON UNIT',I3,' USING FORMAT: ',A20/30X,83('-'))
DO 100 I=1,II
READ (LOCAT,FMTIN) (IA(J,I),J=1,JJ)
100 CONTINUE
GO TO 300
C 
C C5------LOCAT<0 THEN READ UNFORMATTED RECORD CONTAINING ARRAY VALUES
200 LOCAT=-LOCAT
IF(K.GT.0) WRITE(IOUT,201) ANAME,K,LOCAT
201 FORMAT(1H0,30X,6A4,' WILL BE READ UNFORMATTED ON UNIT',I3/30X,73('-'))
IF(K.LE.0) WRITE(IOUT,202) ANAME,LOCAT
202 FORMAT(1H0,30X,6A4,
1 ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,60('-'))
READ AN UNFORMATTED DUMMY RECORD FIRST.
READ(LOCAT) IA

IF ICONST NOT ZERO THEN MULTIPLY ARRAY VALUES BY ICONST.
300 IF(ICONST.EQ.0) GO TO 320
   DO 310 I=1,II
      DO 310 J=1,JJ
         IA(J,I)=IA(J,I)*ICONST
   310 CONTINUE

IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
320 IF(IPRN.LT.0) RETURN
   IF(IPRN.GT.5) IPRN=0
   IPRN=IPRN+1

PRINT COLUMN NUMBERS AT TOP OF PAGE.
   IF(IPRN.EQ.1) CALL UCOLNO(1, JJ, 0, 10, 12, IOUT)
   IF(IPRN.EQ.1) CALL UCOLNO(1, JJ, 4, NL, IPRN, IOUT)

PRINT EACH ROW IN THE ARRAY.
   DO 110 I=1,II
      WRITE(IOUT,lOO1) I,(IA(J,I),J=1,JJ)
   110 CONTINUE

SELECT THE FORMAT
GO TO(101,102,103,104,105,106), IPRN

--- FORMAT 10111
   101 WRITE(IOUT,1001) I,(IA(J,I),J=1,JJ)
   1001 FORMAT(1HO,I3,2X,I11,9(1X,I11)/(5X,10(1X,I11)))
   GO TO 110

--- FORMAT 4012
   102 WRITE(IOUT,1002) I,(IA(J,I),J=1,JJ)
   1002 FORMAT(1HO,I3,1X,60(I1X,I11)/(5X,60(I1X,I11)))
   GO TO 110

--- FORMAT 3013
   103 WRITE(IOUT,1003) I,(IA(J,I),J=1,JJ)
   1003 FORMAT(1HO,I3,1X,30(I1X,I3)/(5X,30(I1X,I3)))
   GO TO 110

--- FORMAT 2514
   104 WRITE(IOUT,1004) I,(IA(J,I),J=1,JJ)
   1004 FORMAT(1HO,I3,1X,25(I1X,I5)/(5X,25(I1X,I5)))
   GO TO 110

--- FORMAT 2015
   105 WRITE(IOUT,1005) I,(IA(J,I),J=1,JJ)
   1005 FORMAT(1HO,I3,1X,20(I1X,I5)/(5X,20(I1X,I5)))
   110 CONTINUE
RETURN

END
### List of Variables for Module U2DINT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANAME</td>
<td>Module</td>
<td>Label for the printout of input array.</td>
</tr>
<tr>
<td>FMTIN</td>
<td>Module</td>
<td>Format under which the array values will be read.</td>
</tr>
<tr>
<td>I</td>
<td>Module</td>
<td>Index for rows.</td>
</tr>
<tr>
<td>IA</td>
<td>Module</td>
<td>DIMENSION (JJ,II), Input array.</td>
</tr>
<tr>
<td>ICONST</td>
<td>Module</td>
<td>Constant to which all array values are set if LOCAT is equal to zero or by which all array values are multiplied if LOCAT is not equal to zero.</td>
</tr>
<tr>
<td>II</td>
<td>Module</td>
<td>Number of rows.</td>
</tr>
<tr>
<td>IN</td>
<td>Module</td>
<td>Unit number from which the array-control record will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPRN</td>
<td>Module</td>
<td>Code for format to be used when printing arrays.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Index for columns.</td>
</tr>
<tr>
<td>JJ</td>
<td>Module</td>
<td>Number of columns.</td>
</tr>
<tr>
<td>K</td>
<td>Module</td>
<td>Layer number.</td>
</tr>
</tbody>
</table>
| LOCAT    | Module | Location of values to fill in the array.  
  < 0, read an unformatted record containing the array values.  
  = 0, set all the array values equal to constant (CNSTNT).  
  > 0, read formatted records containing the array values. |
| NL       | Module | Number of columns per line. |
Narrative for Module U1DREL

Module U1DREL reads values for a one-dimensional real array. First it reads an "array-control record." Then, based on the contents of the array-control record, it may read array values. The array-control record contains four fields: location (LOCAT), constant (CNSTNT), format (FMTIN), and printout indicator (IPRN). The LOCAT field determines where array values will come from. If LOCAT is positive, it is the unit number from which array values will be read in the format specified in FMTIN. If LOCAT is zero, all of the array values will be set equal to CNSTNT. If LOCAT is not zero and CNSTNT is not zero, the array values will be multiplied by the value of CNSTNT. The field IPRN (table 2) contains a code number for a FORTRAN format to be used when printing the array.

Module U1DREL performs its tasks in the following order:

1. Read the array-control record (LOCAT, CNSTNT, FMTIN, and IPRN).

2. Use LOCAT to determine where the array is coming from (DO STEPS 3 OR 4).

3. If LOCAT equals zero, set all array values equal to CNSTNT and print a message to that effect. RETURN.

4. If LOCAT is greater than zero, read array values according to the format in FMTIN.

5. If CNSTNT is not equal to zero, multiply the array values by CNSTNT.

6. If IPRN is greater than or equal to zero, print the array values.

7. RETURN.
Array Control Record controls the input of array values. It contains four fields: LOCAT, CNSTNT, FMTIN, and IPRN.

LOCAT is a code showing where array values will come from.

If LOCAT = 0, array values will be set equal to CNSTNT.

If LOCAT > 0, array values will be read from the unit number equal to LOCAT in the format specified in FMTIN.

CNSTNT is the constant to which all array values are set if LOCAT is equal to zero, and it is the constant by which all array values are multiplied if LOCAT is not equal to zero.

FMTIN is the format in which array values are read if LOCAT is greater than zero.

IPRN is a code showing the format to be used if array values are to be printed.
SUBROUTINE UIDREL(A,ANAME,JJ,IN,IOUT)

C-----VERSION 1643 12MAY1987 UIDREL
C
C ROUTINE TO INPUT 1-D REAL DATA MATRICES
C
A IS ARRAY TO INPUT
C
ANAME IS 24 CHARACTER DESCRIPTION OF A
C
JJ IS NO. OF ELEMENTS
C
IN IS INPUT UNIT
C
IOUT IS OUTPUT UNIT
C
C******************************************************************************
C
SPECIFICATIONS:

CHARACTER*4 ANAME
CHARACTER*20 FMTIN
DIMENSION A(JJ),ANAME(6)

C***********************************************************************~**
C
C1------READ ARRAY CONTROL RECORD.
   READ (IN,1) LOCAT,CNSTNT,FMTIN,IPRN
   1 FORMAT(I10,F10.O,A20,I10)
C
C2------USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
   IF(LOCAT.GT.0) GO TO 90
C
C3------IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN
   DO 80 J=1,JJ
       80 A(J)=CNSTNT
       WRITE(IOUT,3) ANAME,CNSTNT
       3 FORMAT(1H0,52X,6A4,' = ',G15.7)
       RETURN
C
C4------IF LOCAT>0 THEN READ FORMATTED RECORDS USING FORMAT FMTIN.
   90 WRITE(IOUT,5) ANAME,LOCAT,FMTIN
   5 FORMAT(1H0,///30X,6A4,' WILL BE READ ON UNIT','I3,
        ' USING FORMAT: ','A20/30X,79(('-'))/
          READ (LOCAT,FMTIN) (A(J),J=1,JJ)
C
C5------IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.
   IF(CNSTNT.EQ.0.) GO TO 120
   DO 100 J=1,JJ
       100 A(J)=A(J)*CNSTNT
C
C6------IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
   120 IF(IPRN.LT.0) RETURN
       WRITE(IOUT,1001) (A(J),J=1,JJ)
       1001 FORMAT((1X,1PG12.5,9(1X,G12.5)))
       RETURN
C
C7------CONTINUE
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Module</td>
<td>DIMENSION (JJ), Input array.</td>
</tr>
<tr>
<td>ANAME</td>
<td>Module</td>
<td>Label for printout of the input array.</td>
</tr>
<tr>
<td>CNSTNT</td>
<td>Module</td>
<td>Constant to which all array values are set if LOCAT is equal to zero or by which all array values are multiplied if LOCAT is not equal to zero.</td>
</tr>
<tr>
<td>FMTIN</td>
<td>Module</td>
<td>Format under which the array values will be read.</td>
</tr>
<tr>
<td>IN</td>
<td>Module</td>
<td>Unit number from which the array control record will be read.</td>
</tr>
<tr>
<td>IOUT</td>
<td>Global</td>
<td>Primary unit number for all printed output. IOUT = 6.</td>
</tr>
<tr>
<td>IPRN</td>
<td>Module</td>
<td>Code for the format to be used when printing the arrays.</td>
</tr>
<tr>
<td>J</td>
<td>Module</td>
<td>Array index.</td>
</tr>
<tr>
<td>JJ</td>
<td>Module</td>
<td>Number of elements in the array.</td>
</tr>
</tbody>
</table>
| LOCAT    | Module| Location of values to fill in the array.  
  \(< 0\), read an unformatted record containing the array values.  
  \(= 0\), set all the array values equal to constant (CNSTNT).  
  \(> 0\), read the formatted records containing the array values. |
REFERENCES


Rushton, K. R., Redshaw S. C., 1979, Seepage and groundwater flow-numerical analysis by analog and digital methods; New York, John Wiley and Sons


APPENDIX A

PROGRAM PORTABILITY

Introduction

One of the major design requirements for the model program was that it should be portable. A portable program is one that can be run with a minimum of modification on most computers that are physically capable of running a program of its type. The goal of portability for the model program has been attained as evidenced by the fact that it has run successfully in either its present form or the earlier form (McDonald and Harbaugh, 1984) on computers manufactured by many companies including mainframe computers, minicomputers, and microcomputers. The following discussion explains in more detail the concept of portability, what was done to maximize portability of the model program, and circumstances that might require that the program be changed in order to run successfully on a particular computer.

The Impact of the Programming Language on Portability

The programming language is the most important factor that determines program portability. There are a variety of programming languages available, and for each language, there are numerous versions which have resulted from the desire of vendors to improve the power of the language and to take advantage of the hardware features of their particular computers. The most commonly available language suitable to use for the model program is FORTRAN. There are two versions defined by the American National Standards Institute (ANSI) on which most commercial versions are based, ANSI X3.9-1966 and ANSI X3.9-1978.¹ These versions are commonly referred to as FORTRAN 66 and FORTRAN 77, respectively. FORTRAN 66 was selected for the original version of the model because, at the time, it was far more widely supported than was FORTRAN 77. Now that the FORTRAN 77 standard is more widely supported, the program has been converted to this standard.

The program in this report is nearly identical to the original program except for the changes required to make it comply with the FORTRAN 77 standard. These changes are minor and are described by McDonald and Harbaugh (1984, p. 505). The conversion was done solely for the purpose of making the program comply with the FORTRAN 77 standard; many of the features that make FORTRAN 77 more powerful than FORTRAN 66 were not used. An effort to make more extensive FORTRAN 77 revisions to the program is judged to be uneconomical. Such an effort would not result in significant improvements in program clarity or efficiency.


Most commercial versions of FORTRAN 77 compilers include some extended features not defined as part of the FORTRAN 77 standard. Such features vary widely among computer vendors and were not used in the model program. The program contains only one exception to complete compliance to the FORTRAN 77 standard that the authors are aware of. This exception, which is explained in a following section (see The Impact of Allocating Array Storage in a Single Array on Portability), is commonly allowed on computers and should not be a major restriction to portability. Because the program closely follows the FORTRAN 77 standard, the program should work on any computer supporting this language provided that the computer has adequate computational power.

It is recognized that some users may want the program converted back into the FORTRAN 66 language because they have access only to older compilers. Only a few changes are required to convert the program back to FORTRAN 66. Information about how to convert is provided in a following section (see Conversion to FORTRAN 66).

The Impact of Computational Precision on Portability

Variation of precision among computers causes some problems with program portability. Computational precision refers to the accuracy at which numbers are calculated and stored in the computer. To prevent the imposition of constraints on the computers on which FORTRAN is implemented, the computational precision was not defined as part of the standards. The accuracy of model results are dependent on the computational precision, so precision must be considered when moving the model program among computers.

The model program was developed on computers using 32 binary bits to represent single precision real numbers. This gives from 6 to 7 decimal digits of precision and includes values that range in magnitude from approximately $10^{-39}$ to $10^{38}$. Double precision real numbers are represented by 64 binary bits and range in magnitude from approximately $10^{-10000}$ to $10^{10000}$ with 14 to 15 decimal digits of precision. The head array, HNEW, and some variables in the solvers are stored as double precision, and accordingly many calculations in the solvers are double precision. This was necessary for accuracy under some conditions. The model program should perform adequately for most problems on computers that use 32 bits for single precision and 64 bits for double precision. However, the required precision depends on the problem being simulated. Thus, the user must ultimately determine if adequate precision is being used for solving a particular problem.

There are some situations for which there is a need to modify the program to make all real number calculations in double precision. If using a computer that represents single precision real numbers with less than 32 bits, then double precision is probably necessary for all real numbers and calculations. Even on computers that represent single precision numbers with 32 bits, certain problems are difficult to solve without making all real numbers and calculations double precision. Unfortunately, it is difficult to predict if a specific problem requires all double precision. Simulations with very large numbers of cells, for example more than 50000, are more likely to have precision problems than are smaller simulations. Simulations in which there are areas having significant ground-water flow and yet the heads in the adjacent model cells in these areas are equal within .01 percent or less are
also more likely to have precision problems. In addition, precision problems depend on the preciseness of the attempted solution. In general, the symptoms of inadequate precision are either a poor volumetric flow balance or lack of convergence by the solver. However, these same symptoms are more commonly caused by bad input data or improper adjustment of parameters that control iteration. Because precision problems are in general fairly unlikely and use of all double precision results in increased memory and computer time usage, conversion to double precision should probably be done as a last resort in order to solve convergence problems. If the program is converted to all double precision, almost twice the computer memory is required for model data. To convert all real numbers and calculations to double precision, do the following:

1. Declare all single precision real arrays and variables as DOUBLE PRECISION in the main program and in every subroutine. This can be done by adding the statement

   \[
   \text{IMPLICIT DOUBLE PRECISION (A-H,O-Z)}
   \]

   to the beginning of the specifications section of every subroutine and the main program. Alternately, some compilers have a special command that will accomplish this without the need to modify the program itself.

2. In the Basic Package Allocate module (subroutine BASIAL) change the statement

   \[
   \text{ISUM=} \text{ISUM}+2*\text{NRCL}
   \]

   to

   \[
   \text{ISUM=} \text{ISUM}+\text{NRCL}
   \]

3. Change the real intrinsic functions in subroutine SSIPII to their double precision equivalents. Specifically, change all occurrences of AMAX1 to DMAX1, and AMIN1 to DMIN1.

   If using a computer on which significantly more than 32 bits are used to represent single precision real numbers, then the partial use of double precision data and calculations as included in the present model program may be unnecessary. The program should still work on such computers without modification, but computational time and memory would be saved if double precision were eliminated. The program was designed so that changing to all single precision would be easy. For this reason, the use of double precision intrinsic functions and constants is avoided, and all conversions between real and double precision are done by implied type changes in assignment statements. If it is determined that the use of all single precision is acceptable, make the following changes to the program:

   1. Delete all DOUBLE PRECISION specification statements.

   2. In the Basic Package Allocate module (subroutine BASIAL) change the statement

   \[
   \text{ISUM=} \text{ISUM}+2*\text{NRCL}
   \]

   to

   \[
   \text{ISUM=} \text{ISUM}+\text{NRCL}
   \]
The model program was developed on computers that use 32 bits for integer numbers and calculations. This is more than adequate to represent the range of integer numbers that are used in the program under any conditions. The largest integer that the computer must be able to represent is the dimensioned size of the X array in the main program. Many FORTRAN compilers allow one to specify that integers be represented by 16 bits, which does not provide enough precision for larger model simulations. That is, a 16 bit integer in a typical computer can represent numbers in the range -32768 to 32767, and an X array dimension of 32767 is adequate only for simulations that have 2200 to 3000 cells, depending on what options are used.

The Impact of Allocating Array Storage in a Single Array on Portability

Because almost all model data are stored in the X array, which is dimensioned in the main program, this array can be quite large. It is generally 10 to 15 times the size of the number of model cells. Some computers require that special compiler options be used when an array exceeds a specified size. Users are cautioned to be aware of this and use the appropriate options as needed. The result of using the incorrect option can sometimes be that the program will execute without producing error messages, but answers will be incorrect.

All computers limit the amount of array space that a program can use. If a user's simulation exceeds this limit, the only options may be to use a different computer or make the simulation smaller. However, some computers make a distinction between total amount of memory used for all arrays and the total used by a single array. That is, some computers might allow 4 arrays each having 32000 elements, but not allow a single array consisting of 128000 elements. On such a computer, it is quite possible that one could exceed the size limit for a single array without exceeding the total array size limit because nearly all model data are stored in the single X array. If this situation occurs, it is possible to break the X array into smaller pieces. Although this can be done by modifying only the main program, the modification is fairly complex. Such modification requires a knowledgeable programmer who has a clear understanding of how model data are stored within the X array. Before making such a modification, it would be prudent to assess how long the desired simulation might take to execute. Generally, computer execution speed is more of a constraint on maximum problem size than is array size.

The only known exception to the use of the FORTRAN 77 standard in the model program is that the data type of actual subroutine arguments does not always match the type of the corresponding dummy arguments. The standard requires the data type of actual and dummy arguments to match. This only happens in the main program where it calls subroutines using actual arguments that are elements from the X array. The X array itself is data type real, and several model arrays stored with X are either integer (arrays IOFLG, IBOUND, IRCH, IEVT, and LRCH) or double precision (array HNEW). For example, actual argument X(LCIRCH) in the main program is passed to dummy argument IRCH in subroutine RCHIRP. X(LCIRCH) is of type real, and IRCH is of type integer. This practice has not been a problem in the past. Should this become a problem on future computers, required changes will be fairly minor, affecting only the main program and some of the Allocate modules.
The Impact on Portability of Preconnected File Units

FORTRAN 77 provides 2 ways for a file unit to become connected to a file -- preconnection and the OPEN statement. The model program uses preconnection. This means that the computer's operating system provides the connection between files and file units prior to program execution. Often the user must issue commands to the system, which connect the necessary files and file units, prior to running the program. The specific method for doing this varies among computers. Generally, preconnection is adequate, but it may be inconvenient on some computers. Modification of the main program to use OPEN statements is a simple task for a programmer.

Conversion to FORTRAN 66

Because the model program uses only one feature of FORTRAN 77 that is not part of FORTRAN 66, few changes are needed in order to make the program comply with the FORTRAN 66 standard. All of the required changes are a result of differences in the way character (alphanumeric) data are handled by the two versions of FORTRAN. Any variables or arrays holding character data must be declared to be the character data type in a FORTRAN 77 program; in a FORTRAN 66 program, character data are stored in numeric variables or arrays. The specific changes that are required to make the model program comply with the FORTRAN 66 standard are shown below. It is assumed that at least four characters can be stored in a single precision real variable or array element.

1. Delete all CHARACTER*4 statements throughout all subroutines and the main program.

2. In each of the array reading utility modules (UIDREL, U2DREL, and U2DINT), change the statement

   CHARACTER*20 FMTIN

   to

   DIMENSION FMTIN(5)

3. In each of the array reading utility modules (UIDREL, U2DREL, and U2DINT), change all occurrences of "A20" to "5A4". For example, change

   1   ' USING FORMAT: ',A20/30X,79('-')/1

   in subroutine UIDREL to

   1   ' USING FORMAT: ',5A4/30X,79('-')/1

4. To make the program strictly comply with the FORTRAN 66 standard, it is necessary to change character constants in DATA statements to Hollerith constants. However, most FORTRAN 66 compilers accept character constants specified using FORTRAN 77 notation (using apostrophes). Thus, it is unlikely that this change will be required.
APPENDIX B
SPACE REQUIREMENTS IN THE X ARRAY

The outline below gives the X-array space requirements for each package. The formulas can be used to calculate the exact size of the X array for a given problem. However, it is generally easier to simply run the model program and let it calculate the size. The total space required is printed as part of the model printout even if the X array is dimensioned too small. As a rough estimate, the X array is approximately 10 to 15 times the number of nodes in the model depending on the options selected. In the outline below, NODES is defined as NCOL*NROW*NLAY, the number of nodes in the model.

I. BAS Package
   A. $8 \times \text{NODES} + (\text{NLAY} - 1) \times \text{NCOL} \times \text{NROW} + \text{NROW} + \text{NCOL} + 4 \times \text{NLAY}$
   B. Additionally, add NODES if start head is saved (ISTRT is not 0), and add NODES if BUFF is separate from RHS (IAPART is not 0)

II. BCF Package
   A. NLAY
   B. If a transient simulation (ISS is 0), add NODES
   C. For each layer where LAYCON is one or three, add $2 \times \text{NCOL} \times \text{NROW}$
   D. For each layer where LAYCON is two or three, add NCOL*NROW
   E. If a transient simulation (ISS is 0), for each layer where LAYCON is two or three, add NCOL*NROW

III. WEL Package -- 4*MXWELL
IV. DRN Package -- 5*MXDRAN

V. RIV Package -- 6*MXRIVR

VI. EVT Package

A. Option 1 -- 3*NCOL*NROW

B. Option 2 -- 4*NCOL*NROW

VII. GHB Package -- 5*MXBND

VIII. RCH Package

A. Options 1 and 3 -- NCOL*NROW

B. Option 2 -- 2*NCOL*NROW

IX. SIP Package -- 4*NODES + 4*MXITER + NPARM

X. SOR Package -- (NLAY + 4)*NCOL*NLAY + 4*MXITER

Generally, it is advisable to have the X-array dimension relatively close to the amount of space needed for a specific problem. On the other hand, it is inconvenient to redimension the X array every time a new option is selected. Several load modules with the size of the X array differing by a factor of two should be adequate.
APPENDIX C

CONTINUATION OF A PREVIOUS RUN

There is often value in breaking long simulations into several short model runs. This allows one to decide, between runs, whether or not to continue the simulation. Although the model program in this report does not have a special option for making continuation runs, it is quite simple to continue a simulation by using the output of one run as input of the next. Simply save the heads from the run that is to be continued on a disk file, and specify that file as starting heads for the next run. The subroutine that reads starting heads (UZDREL) is capable of reading model-generated disk files of saved heads without the need for reformatting.

Because volumetric budget terms are always set to zero at the start of a model run, the printed budget on a model run represents only that one run, not the total of all runs in a series of continuation runs. If a total budget for a series of continuation runs is desired, the totals from each run can be added externally. Similarly, the model program keeps track of simulation time only for single model runs, but total simulation time for a series of continuation runs can be calculated externally by adding the simulation times of each run.
APPENDIX D

SAMPLE PROBLEM

This sample problem is intended to illustrate input and output from the program. There are three simulated layers, as shown in the accompanying illustration, which are separated from each other by confining layers. Each layer is a square 75,000 feet on a side and is divided by a grid into 15 rows and 15 columns which form squares 5,000 feet on a side. Flow within the confining layers is not simulated, but the effects of the confining layers on flow between the active layers are incorporated in the vertical leakance (Vcont) terms. Flow into the system is infiltration from precipitation; flow out of the system is to buried drain tubes, discharging wells, and a lake which is represented by a constant-head boundary.

Recharge to Layer 1 = 3X10^-4 ft/s

Between layers 1 and 2 vertical hydraulic conductivity divided by thickness = 2X10^-7 ft/s

Between layers 2 and 3 vertical hydraulic conductivity divided by thickness = 1X10^-8 ft/s
Setting starting heads equal to 0.0, the program was run to get a steady-state solution. The Strongly Implicit Procedure was used to solve the system of difference equations: the error criterion was set at 0.001 feet, the acceleration parameter was set to 1.0, and the maximum number of iterations was set equal to 50. A seed of 0.001 was specified for use in calculating the iteration parameters; 31 iterations were needed to close.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Row</th>
<th>Column</th>
<th>Layer</th>
<th>Row</th>
<th>Column</th>
<th>Elevation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>8</td>
<td>2</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>12</td>
<td>1</td>
<td>8</td>
<td>4</td>
<td>10.0</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td>8</td>
<td>5</td>
<td>20.0</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>10</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>30.0</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>12</td>
<td>1</td>
<td>8</td>
<td>7</td>
<td>50.0</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>14</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>70.0</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>8</td>
<td>1</td>
<td>8</td>
<td>9</td>
<td>90.0</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>10</td>
<td>1</td>
<td>8</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>12</td>
<td></td>
<td>13</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>10</td>
<td></td>
<td>13</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Input Data for Sample Problem

Note that input item names are included in many of the data records shown below. These are included for identification purposes. These names are ignored by the model program when it reads the data because they are placed in columns beyond those used to hold model data.

Input for FORTRAN unit 1 -- Basic Package:
SAMPLE----3 LAYERS, 15 ROWS, 15 COLUMNS; STEADY STATE; CONSTANT HEADS COLUMN 1, LAYERS 1 AND 2; RECHARGE, WELLS AND DRAINS

<table>
<thead>
<tr>
<th>Layer</th>
<th>Rows</th>
<th>Columns</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Data records follow with numerical values, line numbers, and notes for PERLEN, NSTP, TSMULT.]

D-3
Input for FORTRAN unit 11 -- Block-Centered Flow Package:

1 0 ISS, IBCFBD

0 1. TRPY
0 5000. DELR
0 5000. DELC
0 .001 HY-1
0 -150. BOT-1
0 2.E-8 WHY/THICK-1
0 .01 T-2
0 1.E-8 WHY/THICK-2
0 .02 T-3

Input for FORTRAN unit 18 -- Recharge Package:

1 0 NRCHOP, IRCHBD
1 INRECH
0 3.E-8 RECH-1

Input for FORTRAN unit 19 -- Strongly Implicit Procedure Package:

50 5 MXITER, NPARM
1. .001 0 .001 1 ACCL, ERR, IPCALC, WSEED

Input for FORTRAN unit 13 -- Drain Package:

9 0 MXDRAI, IDRNBD
9 NDRAIN
1 8 2 0. 1.E00
1 8 3 0. 1.E00
1 8 4 10. 1.E00
1 8 5 20. 1.E00
1 8 6 30. 1.E00
1 8 7 50. 1.E00
1 0 8 70. 1.E00
1 8 9 90. 1.E00
1 8 10 100. 1.E00

Input for FORTRAN unit 12 -- Well Package:

15 0 MXWELL, IWELBD
15 ITMP (NWELLS)
3 5 11 -5.
2 4 6 -5.
2 6 12 -5.
1 9 8 -5.
1 9 10 -5.
1 9 12 -5.
1 9 14 -5.
1 11 8 -5.
1 11 10 -5.
1 11 12 -5.
1 11 14 -5.
1 13 8 -5.
1 13 10 -5.
1 13 12 -5.
1 13 14 -5.
Sample Problem Output

Note that it should not be expected that the outputs from running the same problem on different computers will exactly match. Small variations in output can be caused by differences in the way real numbers are stored and calculated. Storage and calculation of real numbers depend on the specific computer hardware, the FORTRAN compiler, and the math library that is loaded with the compiled program. Output variations among computers also depend on the size of the problem, the number of iterations required for solution, and the precision used when printing results. In this sample problem, most values should not vary from one computer to another. Those few values that vary should generally vary only in the least significant digit; however, the budget term "IN-OUT", because it is the difference of two nearly identical numbers, may vary by as much as a factor of 3 from one computer to another.
SAMPLE—3 LAYERS, 15 ROWS, 15 COLUMNS; STEADY STATE; CONSTANT HEADS COLUMN 1, LAYERS 1 AND 2; RECHARGE, WELLS AND DRAINS

BOUNDARY ARRAY FOR LAYER 1 WILL BE READ ON UNIT 1 USING FORMAT: (1513)

```
1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
1 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
2 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
3 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
4 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
5 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
6 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
7 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
8 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
9 -1  1  1  1  1  1  1  1  1  1  1  1  1  1
10  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
11  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
12  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
13  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
14  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
15  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

BOUNDARY ARRAY FOR LAYER 2 WILL BE READ ON UNIT 1 USING FORMAT: (1513)

```
1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
1  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
2  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
3  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
4  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
5  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
6  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
7  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
8  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
9  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
10  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
11  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
12  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
13  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
14  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
15  -1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

D-6
BOUNDARY ARRAY = 1 FOR LAYER 3

AQUIFER HEAD WILL BE SET TO 999.99 AT ALL NO-FLOW NODES (BOUND=0).

INITIAL HEAD = 0.00000000E+00 FOR LAYER 1
INITIAL HEAD = 0.00000000E+00 FOR LAYER 2
INITIAL HEAD = 0.00000000E+00 FOR LAYER 3

DEFAULT OUTPUT CONTROL -- THE FOLLOWING OUTPUT COMES AT THE END OF EACH STRESS PERIOD:
TOTAL VOLUMETRIC BUDGET

HEAD

COLUMN TO ROW ANISOTROPY = 1.000000
DEL R = 5000.000
DEL C = 5000.000

HYD. COND. ALONG ROWS = 0.99999999E-03 FOR LAYER 1

BOTTOM = -150.0000 FOR LAYER 1

VERT HYD COND /THICKNESS = 0.20000000E-07 FOR LAYER 1
TRANS. ALONG ROWS = 0.10000000E-01 FOR LAYER 2

VERT HYD COND /THICKNESS = 0.10000000E-07 FOR LAYER 2
TRANS. ALONG ROWS = 0.20000000E-01 FOR LAYER 3

SCNTINUTION BY THE STRONGLY IMPLICIT PROCEDURE

SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE

MAXIMUM ITERATIONS ALLOWED FOR CLOSURE = 50
ACCELERATION PARAMETER = 1.0000
HEAD CHANGE CRITERION FOR CLOSURE = 0.10000000E-02
SIP HEAD CHANGE PRINTOUT INTERVAL = 1

5 ITERATION PARAMETERS CALCULATED FROM SPECIFIED WSEED = 0.00100000 :
0.00000000E+00 0.82217200E+00 0.96837720E+00 0.99437660E+00 0.99999999E+00
**STRESS PERIOD NO. 1, LENGTH = 86400.00**

**NUMBER OF TIME STEPS = 1**
**MULTIPLIER FOR DELT = 1.000**
**INITIAL TIME STEP SIZE = 86400.00**

### 15 WELLS

<table>
<thead>
<tr>
<th>Layer</th>
<th>Row</th>
<th>Col</th>
<th>Stressed Rate</th>
<th>Well No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
<td>11</td>
<td>-5.0000</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>6</td>
<td>-5.0000</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>12</td>
<td>-5.0000</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>8</td>
<td>-5.0000</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>10</td>
<td>-5.0000</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>12</td>
<td>-5.0000</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>14</td>
<td>-5.0000</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>8</td>
<td>-5.0000</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>10</td>
<td>-5.0000</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>12</td>
<td>-5.0000</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>14</td>
<td>-5.0000</td>
<td>11</td>
</tr>
<tr>
<td>1</td>
<td>13</td>
<td>8</td>
<td>-5.0000</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>13</td>
<td>10</td>
<td>-5.0000</td>
<td>13</td>
</tr>
<tr>
<td>1</td>
<td>13</td>
<td>12</td>
<td>-5.0000</td>
<td>14</td>
</tr>
<tr>
<td>1</td>
<td>13</td>
<td>14</td>
<td>-5.0000</td>
<td>15</td>
</tr>
</tbody>
</table>

### 13 DRAINS

<table>
<thead>
<tr>
<th>Layer</th>
<th>Row</th>
<th>Col</th>
<th>Elevation</th>
<th>Conductance</th>
<th>Drain No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>2</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>3</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>6</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>5</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>6</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>8</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>9</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>10</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>100</td>
<td>0.0000E+00</td>
<td>1.000</td>
<td>9</td>
</tr>
</tbody>
</table>

**MAXIMUM HEAD CHANGE FOR EACH ITERATION:**

<table>
<thead>
<tr>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
<th>Layer</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-22.41</td>
<td>3</td>
<td>15.39</td>
<td>15</td>
<td>48.21</td>
<td>3</td>
<td>15.39</td>
<td>15</td>
<td>48.21</td>
<td>3</td>
<td>15.39</td>
<td>15</td>
<td>48.21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2.492</td>
<td>3</td>
<td>3.190</td>
<td>3</td>
<td>6.214</td>
<td>3</td>
<td>7.411</td>
<td>3</td>
<td>2.019</td>
<td>1</td>
<td>2.302</td>
<td>1</td>
<td>2.019</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.5503</td>
<td>3</td>
<td>6.4821</td>
<td>2</td>
<td>0.4711</td>
<td>3</td>
<td>5.102</td>
<td>2</td>
<td>2.019</td>
<td>1</td>
<td>2.302</td>
<td>1</td>
<td>2.019</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1108</td>
<td>3</td>
<td>0.7059E-01</td>
<td>3</td>
<td>0.2019</td>
<td>1</td>
<td>1.141</td>
<td>3</td>
<td>0.3141</td>
<td>1</td>
<td>1.141</td>
<td>3</td>
<td>0.3141</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.7059E-02</td>
<td>3</td>
<td>0.1586E-01</td>
<td>3</td>
<td>0.1777E-01</td>
<td>3</td>
<td>0.1777E-01</td>
<td>3</td>
<td>0.1777E-01</td>
<td>3</td>
<td>0.1777E-01</td>
<td>3</td>
<td>0.1777E-01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.4168E-02</td>
<td>3</td>
<td>0.2555E-02</td>
<td>3</td>
<td>0.9771E-02</td>
<td>1</td>
<td>0.1082E-01</td>
<td>3</td>
<td>0.1082E-01</td>
<td>1</td>
<td>0.1082E-01</td>
<td>3</td>
<td>0.1082E-01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2426E-03</td>
<td>1</td>
<td>0.1108</td>
<td>2</td>
<td>0.4711</td>
<td>3</td>
<td>5.102</td>
<td>1</td>
<td>2.019</td>
<td>1</td>
<td>2.302</td>
<td>1</td>
<td>2.019</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MAXIMUM LOAD CHANGE FOR EACH DRAIN:**

-22.41  (3, 5, 11) 12.48  (1, 1, 15) 15.39  (3, 1, 14) 48.21  (1, 1, 15) 35.90  (3, 1, 13)
2.492  (3, 9, 14) 1.490  (3, 10, 13) 6.214  (1, 12, 14) 7.411  (3, 11, 14) 12.66  (1, 15, 10)
0.5503  (3, 6, 7) 0.4821  (2, 6, 9) 0.4711  (3, 5, 10) 2.019  (1, 11, 14) 2.302  (3, 5, 13)
0.1108  (3, 8, 12) 0.7059E-01  (3, 12, 11) 0.2019  (1, 14, 14) 0.3141  (3, 13, 14) 0.3321  (1, 15, 15)
0.7059E-02  (1, 13, 12) 0.1586E-01  (2, 11, 11) 0.1777E-01  (3, 11, 10) 0.7911E-01  (1, 14, 14) 0.8500E-01  (3, 7, 14)
0.4168E-02  (1, 13, 14) 0.2555E-02  (3, 14, 15) 0.9771E-02  (1, 14, 14) 0.1082E-01  (3, 13, 14) 0.1030E-01  (1, 15, 15)
0.2426E-03  (1, 13, 12)

**DRAIN NO.**

D-8
### Head in Layer 1 at End of Time Step 1 in Stress Period 1

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24.94</td>
<td>44.01</td>
<td>59.26</td>
<td>71.82</td>
<td>82.52</td>
<td>91.91</td>
<td>100</td>
<td>106.9</td>
<td>112.6</td>
<td></td>
</tr>
<tr>
<td>11.74</td>
<td>121.3</td>
<td>124.3</td>
<td>126.4</td>
<td>127.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>24.45</td>
<td>43.10</td>
<td>57.98</td>
<td>70.17</td>
<td>80.57</td>
<td>90.12</td>
<td>98.40</td>
<td>105.3</td>
<td>111.0</td>
<td></td>
</tr>
<tr>
<td>115.7</td>
<td>119.6</td>
<td>122.7</td>
<td>124.9</td>
<td>126.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>25.45</td>
<td>41.30</td>
<td>55.43</td>
<td>66.79</td>
<td>76.21</td>
<td>86.51</td>
<td>95.20</td>
<td>102.2</td>
<td>107.6</td>
<td></td>
</tr>
<tr>
<td>112.0</td>
<td>116.1</td>
<td>119.6</td>
<td>122.1</td>
<td>125.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>21.92</td>
<td>38.61</td>
<td>51.75</td>
<td>61.79</td>
<td>69.03</td>
<td>81.34</td>
<td>90.75</td>
<td>97.64</td>
<td>102.5</td>
<td></td>
</tr>
<tr>
<td>106.1</td>
<td>110.7</td>
<td>114.9</td>
<td>117.9</td>
<td>119.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>19.73</td>
<td>34.92</td>
<td>47.32</td>
<td>57.69</td>
<td>66.74</td>
<td>77.09</td>
<td>85.76</td>
<td>92.22</td>
<td>96.15</td>
<td></td>
</tr>
<tr>
<td>97.29</td>
<td>103.1</td>
<td>108.8</td>
<td>112.5</td>
<td>114.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>16.51</td>
<td>29.50</td>
<td>40.90</td>
<td>51.30</td>
<td>61.21</td>
<td>71.19</td>
<td>79.85</td>
<td>86.47</td>
<td>90.82</td>
<td></td>
</tr>
<tr>
<td>95.03</td>
<td>94.23</td>
<td>102.1</td>
<td>106.4</td>
<td>100.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>11.55</td>
<td>21.10</td>
<td>31.21</td>
<td>41.40</td>
<td>51.84</td>
<td>63.08</td>
<td>72.68</td>
<td>79.95</td>
<td>84.92</td>
<td></td>
</tr>
<tr>
<td>88.40</td>
<td>61.66</td>
<td>96.43</td>
<td>100.7</td>
<td>101.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3.483</td>
<td>6.832</td>
<td>16.25</td>
<td>26.30</td>
<td>36.97</td>
<td>52.59</td>
<td>64.31</td>
<td>72.52</td>
<td>77.25</td>
<td></td>
</tr>
<tr>
<td>81.99</td>
<td>85.00</td>
<td>89.27</td>
<td>91.72</td>
<td>94.33</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10.54</td>
<td>19.11</td>
<td>28.12</td>
<td>36.92</td>
<td>45.27</td>
<td>52.95</td>
<td>55.38</td>
<td>65.15</td>
<td>66.07</td>
<td></td>
</tr>
<tr>
<td>73.93</td>
<td>73.79</td>
<td>80.64</td>
<td>80.17</td>
<td>80.49</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>14.62</td>
<td>25.66</td>
<td>35.38</td>
<td>43.49</td>
<td>50.11</td>
<td>54.93</td>
<td>57.55</td>
<td>62.95</td>
<td>65.55</td>
<td></td>
</tr>
<tr>
<td>70.39</td>
<td>72.44</td>
<td>76.72</td>
<td>78.26</td>
<td>81.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>17.11</td>
<td>29.96</td>
<td>40.01</td>
<td>47.78</td>
<td>53.24</td>
<td>55.81</td>
<td>53.33</td>
<td>60.27</td>
<td>59.29</td>
<td></td>
</tr>
<tr>
<td>66.43</td>
<td>65.45</td>
<td>72.22</td>
<td>71.04</td>
<td>76.62</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>10.60</td>
<td>35.56</td>
<td>43.07</td>
<td>50.01</td>
<td>55.92</td>
<td>50.33</td>
<td>50.47</td>
<td>61.93</td>
<td>63.10</td>
<td></td>
</tr>
<tr>
<td>67.12</td>
<td>68.50</td>
<td>72.29</td>
<td>72.04</td>
<td>76.85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>19.67</td>
<td>34.74</td>
<td>46.14</td>
<td>53.01</td>
<td>68.04</td>
<td>59.10</td>
<td>56.74</td>
<td>60.70</td>
<td>60.01</td>
<td></td>
</tr>
<tr>
<td>67.22</td>
<td>65.75</td>
<td>71.90</td>
<td>70.35</td>
<td>76.48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>20.27</td>
<td>35.27</td>
<td>46.48</td>
<td>54.61</td>
<td>60.08</td>
<td>63.17</td>
<td>64.52</td>
<td>67.25</td>
<td>68.79</td>
<td></td>
</tr>
<tr>
<td>/1.04</td>
<td>/3.19</td>
<td>/5.04</td>
<td>/2.04</td>
<td>/1.09</td>
<td>/1.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>20.56</td>
<td>35.78</td>
<td>47.16</td>
<td>55.48</td>
<td>61.26</td>
<td>65.02</td>
<td>67.52</td>
<td>69.94</td>
<td>72.01</td>
<td></td>
</tr>
<tr>
<td>74.29</td>
<td>76.22</td>
<td>78.22</td>
<td>79.66</td>
<td>80.82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>----</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>1</td>
<td>24.66</td>
<td>43.73</td>
<td>59.02</td>
<td>71.61</td>
<td>82.32</td>
<td>91.72</td>
<td>99.86</td>
<td>106.7</td>
<td>112.5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>24.17</td>
<td>42.83</td>
<td>57.74</td>
<td>69.95</td>
<td>80.36</td>
<td>89.93</td>
<td>98.22</td>
<td>105.1</td>
<td>110.8</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>23.17</td>
<td>41.03</td>
<td>55.19</td>
<td>66.52</td>
<td>75.77</td>
<td>86.29</td>
<td>95.02</td>
<td>102.0</td>
<td>107.4</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>21.65</td>
<td>38.34</td>
<td>51.50</td>
<td>61.35</td>
<td>60.17</td>
<td>80.90</td>
<td>90.55</td>
<td>97.45</td>
<td>102.3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>19.48</td>
<td>34.65</td>
<td>47.07</td>
<td>57.44</td>
<td>66.30</td>
<td>76.85</td>
<td>85.57</td>
<td>92.00</td>
<td>95.41</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>17.27</td>
<td>29.24</td>
<td>40.65</td>
<td>51.07</td>
<td>60.98</td>
<td>70.98</td>
<td>79.65</td>
<td>86.28</td>
<td>90.54</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>11.38</td>
<td>20.95</td>
<td>31.05</td>
<td>41.25</td>
<td>51.70</td>
<td>62.90</td>
<td>72.48</td>
<td>79.76</td>
<td>84.73</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4.209</td>
<td>8.330</td>
<td>17.58</td>
<td>27.58</td>
<td>38.25</td>
<td>52.94</td>
<td>64.19</td>
<td>72.34</td>
<td>77.12</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10.38</td>
<td>25.61</td>
<td>35.15</td>
<td>43.27</td>
<td>49.91</td>
<td>54.76</td>
<td>57.48</td>
<td>62.79</td>
<td>65.49</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>14.40</td>
<td>70.24</td>
<td>72.37</td>
<td>76.57</td>
<td>80.84</td>
<td>86.30</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>16.07</td>
<td>39.70</td>
<td>72.16</td>
<td>71.75</td>
<td>77.61</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>18.43</td>
<td>66.10</td>
<td>72.16</td>
<td>71.75</td>
<td>77.61</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>19.42</td>
<td>52.80</td>
<td>59.78</td>
<td>57.50</td>
<td>62.53</td>
<td>61.65</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>20.02</td>
<td>61.76</td>
<td>66.18</td>
<td>72.55</td>
<td>76.69</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>20.80</td>
<td>74.11</td>
<td>76.04</td>
<td>78.04</td>
<td>79.49</td>
<td>80.65</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>1</td>
<td>1.800</td>
<td>24.34</td>
<td>43.36</td>
<td>58.70</td>
<td>71.33</td>
<td>82.06</td>
<td>91.48</td>
<td>99.63</td>
<td>106.5</td>
<td>112.3</td>
</tr>
<tr>
<td></td>
<td>117.0</td>
<td>120.9</td>
<td>125.9</td>
<td>126.0</td>
<td>127.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1.764</td>
<td>23.85</td>
<td>42.66</td>
<td>57.42</td>
<td>69.66</td>
<td>80.07</td>
<td>89.68</td>
<td>97.99</td>
<td>104.9</td>
<td>110.6</td>
</tr>
<tr>
<td></td>
<td>115.3</td>
<td>119.2</td>
<td>122.4</td>
<td>124.6</td>
<td>125.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1.691</td>
<td>22.86</td>
<td>40.67</td>
<td>54.87</td>
<td>66.20</td>
<td>75.28</td>
<td>85.98</td>
<td>94.77</td>
<td>101.7</td>
<td>107.2</td>
</tr>
<tr>
<td></td>
<td>111.5</td>
<td>115.7</td>
<td>119.3</td>
<td>121.7</td>
<td>125.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1.578</td>
<td>21.35</td>
<td>37.98</td>
<td>51.17</td>
<td>60.85</td>
<td>62.69</td>
<td>80.41</td>
<td>90.28</td>
<td>97.19</td>
<td>101.9</td>
</tr>
<tr>
<td></td>
<td>104.1</td>
<td>110.0</td>
<td>114.5</td>
<td>117.5</td>
<td>119.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1.415</td>
<td>19.16</td>
<td>34.30</td>
<td>46.75</td>
<td>57.10</td>
<td>65.00</td>
<td>76.54</td>
<td>85.30</td>
<td>91.67</td>
<td>94.17</td>
</tr>
<tr>
<td></td>
<td>77.46</td>
<td>100.7</td>
<td>108.2</td>
<td>112.1</td>
<td>114.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1.176</td>
<td>15.99</td>
<td>28.91</td>
<td>40.33</td>
<td>50.75</td>
<td>60.67</td>
<td>70.70</td>
<td>79.38</td>
<td>86.01</td>
<td>90.12</td>
</tr>
<tr>
<td></td>
<td>90.60</td>
<td>80.55</td>
<td>101.2</td>
<td>106.0</td>
<td>108.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.8273</td>
<td>11.21</td>
<td>20.79</td>
<td>30.88</td>
<td>41.09</td>
<td>51.55</td>
<td>62.67</td>
<td>72.22</td>
<td>79.50</td>
<td>84.46</td>
</tr>
<tr>
<td></td>
<td>61.90</td>
<td>90.77</td>
<td>95.94</td>
<td>99.41</td>
<td>101.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.4331</td>
<td>5.131</td>
<td>10.19</td>
<td>19.27</td>
<td>29.19</td>
<td>39.84</td>
<td>53.40</td>
<td>64.07</td>
<td>72.11</td>
<td>76.95</td>
</tr>
<tr>
<td></td>
<td>81.58</td>
<td>86.48</td>
<td>88.18</td>
<td>91.44</td>
<td>93.95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.7543</td>
<td>10.22</td>
<td>18.62</td>
<td>27.84</td>
<td>36.65</td>
<td>45.06</td>
<td>52.78</td>
<td>57.03</td>
<td>65.02</td>
<td>67.64</td>
</tr>
<tr>
<td></td>
<td>73.01</td>
<td>75.31</td>
<td>80.72</td>
<td>81.64</td>
<td>86.24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.039</td>
<td>14.13</td>
<td>25.29</td>
<td>34.05</td>
<td>42.99</td>
<td>49.65</td>
<td>54.54</td>
<td>57.44</td>
<td>62.61</td>
<td>65.44</td>
</tr>
<tr>
<td></td>
<td>70.05</td>
<td>72.33</td>
<td>76.39</td>
<td>78.15</td>
<td>81.43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1.224</td>
<td>16.59</td>
<td>29.37</td>
<td>39.47</td>
<td>47.28</td>
<td>52.79</td>
<td>55.53</td>
<td>55.01</td>
<td>60.16</td>
<td>60.94</td>
</tr>
<tr>
<td></td>
<td>66.33</td>
<td>67.06</td>
<td>72.13</td>
<td>72.60</td>
<td>77.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>1.341</td>
<td>18.15</td>
<td>31.97</td>
<td>42.54</td>
<td>50.32</td>
<td>55.47</td>
<td>57.94</td>
<td>58.37</td>
<td>61.00</td>
<td>65.08</td>
</tr>
<tr>
<td></td>
<td>66.80</td>
<td>68.41</td>
<td>71.97</td>
<td>73.36</td>
<td>76.49</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>1.415</td>
<td>19.14</td>
<td>32.65</td>
<td>44.61</td>
<td>52.53</td>
<td>57.60</td>
<td>59.63</td>
<td>58.39</td>
<td>62.48</td>
<td>62.84</td>
</tr>
<tr>
<td></td>
<td>67.12</td>
<td>67.35</td>
<td>71.80</td>
<td>71.00</td>
<td>76.24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1.460</td>
<td>19.73</td>
<td>34.68</td>
<td>45.96</td>
<td>54.13</td>
<td>59.63</td>
<td>62.76</td>
<td>64.24</td>
<td>66.87</td>
<td>68.52</td>
</tr>
<tr>
<td></td>
<td>71.27</td>
<td>72.91</td>
<td>75.47</td>
<td>76.77</td>
<td>76.71</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1.481</td>
<td>20.01</td>
<td>35.18</td>
<td>46.03</td>
<td>55.00</td>
<td>60.81</td>
<td>64.59</td>
<td>67.11</td>
<td>69.52</td>
<td>71.61</td>
</tr>
<tr>
<td></td>
<td>72.07</td>
<td>75.62</td>
<td>77.01</td>
<td>79.27</td>
<td>80.42</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Volumetric Budget for Entire Model at End of Time Step 1 in Stress Period 1

**Cumulative Volumes (L**3)**

<table>
<thead>
<tr>
<th>Component</th>
<th>Value</th>
<th>Component</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>STORAGE</td>
<td>0.000000E+00</td>
<td>STORAGE</td>
<td>0.000000E+00</td>
</tr>
<tr>
<td>CONSTANT HEAD</td>
<td>0.000000E+00</td>
<td>CONSTANT HEAD</td>
<td>0.000000E+00</td>
</tr>
<tr>
<td>WELLS</td>
<td>0.000000E+00</td>
<td>WELLS</td>
<td>0.000000E+00</td>
</tr>
<tr>
<td>DRAINS</td>
<td>0.000000E+00</td>
<td>DRAINS</td>
<td>0.000000E+00</td>
</tr>
<tr>
<td>RECHARGE</td>
<td>0.13608E+08</td>
<td>RECHARGE</td>
<td>157.50</td>
</tr>
<tr>
<td>TOTAL IN</td>
<td>0.13608E+08</td>
<td>TOTAL IN</td>
<td>157.50</td>
</tr>
</tbody>
</table>

**Rates for This Time Step (L**3/T)**

<table>
<thead>
<tr>
<th>Component</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>STORAGE</td>
<td>0.000000E+00</td>
</tr>
<tr>
<td>CONSTANT HEAD</td>
<td>0.000000E+00</td>
</tr>
<tr>
<td>WELLS</td>
<td>0.648000E+07</td>
</tr>
<tr>
<td>DRAINS</td>
<td>0.28011E+07</td>
</tr>
<tr>
<td>RECHARGE</td>
<td>0.000000E+00</td>
</tr>
<tr>
<td>TOTAL OUT</td>
<td>0.13608E+08</td>
</tr>
<tr>
<td>IN - OUT</td>
<td>157.49</td>
</tr>
</tbody>
</table>

### Time Summary at End of Time Step 1 in Stress Period 1

<table>
<thead>
<tr>
<th>Time Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME STEP LENGTH</td>
<td>86400.0</td>
</tr>
<tr>
<td>STRESS PERIOD TIME</td>
<td>86400.0</td>
</tr>
<tr>
<td>TOTAL SIMULATION TIME</td>
<td>86400.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECONDS</td>
<td>1440.00</td>
</tr>
<tr>
<td>MINUTES</td>
<td>24.0000</td>
</tr>
<tr>
<td>HOURS</td>
<td>1.0000</td>
</tr>
<tr>
<td>DAYS</td>
<td>0.273785E-02</td>
</tr>
<tr>
<td>YEARS</td>
<td>0.273785E-02</td>
</tr>
</tbody>
</table>

**Percent Discrepancy**

- In: 0.00
- Out: 0.00

---

D-12
APPENDIX E

ABBREVIATED INPUT INSTRUCTIONS

These input instructions are intended as a quick reference for the experienced user. Most explanations that are contained in the complete input instructions given in package documentation have been omitted. The format of input fields is given only for those records that contain fields that are not 10 characters wide. Each input item, for which format is not given, is identified as either a record or an array. For records, the fields contained in the record are given. For arrays, only the array name is given. Input fields which contain codes or flags are described. All other field and array descriptions have been dropped.

Array Input

The real two-dimensional array reader (U2DREL), the integer two-dimensional array reader (U2DINT), and the real one-dimensional array reader (UI2REL) read one array-control record and, optionally, a data array in a format specified on the array-control record.

FOR REAL ARRAY READER (U2DREL or UI2REL)

Data: LOCAT  CNSTNT  FMTIN  IPRN

Format:  110  F10.0  5A4  I10

FOR INTEGER ARRAY READER (U2DINT)

Data: LOCAT  ICONST  FMTIN  IPRN

Format:  110  110  5A4  I10

IPRN--is a flag indicating that the array being read should be printed and a code for indicating the format that should be used. It is used only if LOCAT is not equal to zero. The format codes are different for each of the three modules. IPRN is set to zero when the specified value exceeds those defined in the chart below. If IPRN is less than zero, the array will not be printed.

<table>
<thead>
<tr>
<th>IPRN</th>
<th>U2DREL</th>
<th>U2DINT</th>
<th>UI2REL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10G11.4</td>
<td>10I11</td>
<td>10G12.5</td>
</tr>
<tr>
<td>1</td>
<td>11G10.3</td>
<td>60I1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>9G13.6</td>
<td>40I2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>15F7.1</td>
<td>30I3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15F7.2</td>
<td>25I4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>15F7.3</td>
<td>20I5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>15F7.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>20F5.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>20F5.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>20F5.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>20F5.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>20F5.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>10G11.4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

LOCAT--indicates the location of the data which will be put in the array.

If LOCAT < 0, unit number for unformatted records.
If LOCAT = 0, all elements are set equal to CNSTNT or ICONST.
If LOCAT > 0, unit number for formatted records.
Basic Package Input

Input for the Basic (BAS) Package except for output control is read from unit 1 as specified in the main program. If necessary, the unit number for BAS input can be changed to meet the requirements of a particular computer. Input for the output control option is read from the unit number specified in IUNIT(12).

FOR EACH SIMULATION
1. Record: HEADNG(32)
2. Record: HEADNG (continued)
3. Record: NLAY NROW NCOL NPER ITMUNI
4. Data: IUNIT(24)
   Format: 2413
   (BCF WEL DRN RIV EVT XXX GHB RCH SIP XXX SOR OC)
   1 2 3 4 5 6 7 8 9 10 11 12
5. Record: IAPART ISTRT
6. Array: IBOUND(NCOL,NROW)
   (One array for each layer in the grid)
7. Record: HNOFLO
8. Array: Shead(NCOL,NROW)
   (One array for each layer in the grid)

FOR EACH STRESS PERIOD
9. Data: PERLEN NSTP TSMULT

ITMUNI— is the time unit of model data.
   0 - undefined
   1 - seconds
   2 - minutes
   3 - hours
   4 - days
   5 - years
Consistent length and time units must be used for all model data. The user may choose one length unit and one time unit to be used to specify all input data.

IUNIT is a 24-element table of input units for use by all major options.
IAPART— indicates whether array BUFF is separate from array RHS.
   If IAPART = 0, the arrays BUFF and RHS occupy the same space. This option conserves space. This option should be used unless some other package explicitly says otherwise.
   If IAPART ≠ 0, the arrays BUFF and RHS occupy different space.

ISTRT— indicates whether starting heads are to be saved.
   If ISTRT = 0, starting heads are not saved.
   If ISTRT ≠ 0, starting heads are saved.

IBOUND— is the boundary array.
   If IBOUND(I,J,K) < 0, cell I,J,K has a constant head.
   If IBOUND(I,J,K) = 0, cell I,J,K is inactive.
   If IBOUND(I,J,K) > 0, cell I,J,K is active.

HNOFLO— is the value of head to be assigned to all inactive cells.
Shead— is head at the start of the simulation.
PERLEN— is the length of a stress period.
NSTP— is the number of time steps in a stress period.
TSMULT— is the multiplier for the length of successive time steps.
Output Control Input

Input to Output Control is read from the unit specified in IUNIT(12). All printer output goes to unit 6 as specified in the main program. If necessary, the unit number for printer output can be changed to meet the requirements of a particular computer.

FOR EACH SIMULATION
1. Record: IHEDFM  IDDNFM  IHEDUN  IDDNUM

FOR EACH TIME STEP
2. Record: INCODE  IHDDFL  IBUDFL  ICBCFL
3. Record: Hdp r  Ddp r  Hds v  Dds v
(Record 3 is read 0, 1, or NLAY times, depending on the value of INCODE.)

IHEDFM--is a code for the format in which heads will be printed.
IDDNFM--is a code for the format in which drawdowns will be printed.

INCODE--is the head/drawdown output code.
   If INCODE < 0, layer-by-layer specifications from the last time steps are used. Input item 3 is not read.
   If INCODE = 0, all layers are treated the same way. Input item 3 will consist of one record. IOFLG array will be read.
   If INCODE > 0, input item 3 will consist of one record for each layer.

IHDDFL--is a head and drawdown output flag.
   If IHDDFL = 0, neither heads nor drawdowns will be printed or saved.
   If IHDDFL ≠ 0, heads and drawdowns will be printed or saved.

IBUDFL--is a budget print flag.
   If IBUDFL = 0, overall volumetric budget will not be printed.
   If IBUDFL ≠ 0, overall volumetric budget will be printed.

ICBCFL--is a cell-by-cell flow-term flag.
   If ICBCFL = 0, cell-by-cell flow terms are not saved or printed.
   If ICBCFL ≠ 0, cell-by-cell flow terms are printed or recorded on disk depending on flags set in the component of flow packages, i.e., IWELCB, IRCHCB, etc.

Hdpr--is the output flag for head printout.
   If Hdpr = 0, head is not printed for the corresponding layer.
   If Hdpr ≠ 0, head is printed for the corresponding layer.

Ddpr--is the output flag for drawdown printout.
   If Ddpr = 0, drawdown is not printed for the corresponding layer.
   If Ddpr ≠ 0, drawdown is printed for the corresponding layer.

Hds v--is the output flag for head save.
   If Hds v = 0, head is not saved for the corresponding layer.
   If Hds v ≠ 0, head is saved for the corresponding layer.

Dds v--is the output flag for drawdown save.
   If Dds v = 0, drawdown is not saved for the corresponding layer.
   If Dds v ≠ 0, drawdown is saved for the corresponding layer.
Block-Centered Flow Package Input

Input for the BCF Package is read from the unit specified in IUNIT(1).

FOR EACH SIMULATION
1. Record: ISS       IBCFCB
2. Data:  LAYCON(NLAY) (maximum of 80 layers)
   Format: 40I2
   (If there are 40 or fewer layers, use one record.)
3. Array:  TRPY(NLAY)
4. Array:  DELR(NCOL)
5. Array:  DELC(NROW)

All of the arrays (items 6-12) for layer 1 are read first; then all of the
arrays for layer 2, etc.

IF THE SIMULATION IS TRANSIENT
6. Array:  sf1(NCOL,NROW)

IF THE LAYER TYPE CODE (LAYCON) IS ZERO OR TWO
7. Array:  Tran(NCOL,NROW)

IF THE LAYER TYPE CODE (LAYCON) IS ONE OR THREE
8. Array:  HY(NCOL,NROW)
9. Array:  BOT(NCOL,NROW)

IF THIS IS NOT THE BOTTOM LAYER
10. Array:  Vcont(NCOL,NROW)

IF THE SIMULATION IS TRANSIENT AND THE LAYER TYPE CODE (LAYCON) is TWO OR THREE
11. Array:  sf2(NCOL,NROW)

IF THE LAYER TYPE CODE IS TWO OR THREE
12. Array:  TOP(NCOL,NROW)

ISS--is the steady-state flag.
   If ISS ≠ 0, the simulation is steady state.
   If ISS = 0, the simulation is transient.
IBCFCB--is a flag and a unit number.
   If IBCFCB > 0, cell-by-cell flow terms will be recorded if ICBCFL
   (see Output Control) is set.
   If IBCFCB = 0, cell-by-cell flow terms will not be printed or recorded.
   If IBCFCB < 0, print flow for constant-head cells if ICBCFL is set.
LAYCON--is the layer type table: 0 - confined, 1 - unconfined,
   2 - confined/unconfined (T constant), and 3 - confined/unconfined.
TRPY--is an anisotropy factor for each layer: T or K along a column to T or
   K along a row.
DELR--is the cell width along rows.
DELC--is the cell width along columns.
sf1--is the primary storage factor.
Tran--is the transmissivity along rows.
HY--is the hydraulic conductivity along rows.
BOT--is the elevation of the aquifer bottom.
Vcont--is the vertical hydraulic conductivity divided by the thickness from
   a layer to the layer beneath it.
sf2--is the secondary storage factor.
TOP--is the elevation of the aquifer top.
**River Package Input**

Input to the River (RIV) Package is read from the unit specified in IUNIT(4).

**FOR EACH SIMULATION**

1. Record: MXRIVR  IRIVCB

2. Record: ITMP

3. Record: Layer  Row  Column  Stage  Cond  Rbot
   (Input item 3 normally consists of one record for each river reach. If ITMP is negative or zero, item 3 is not read.)

**IRIVCB**—is a flag and a unit number.
   If IRIVCB > 0, cell-by-cell flow terms will be recorded.
   If IRIVCB = 0, cell-by-cell flow terms will not be printed or recorded.
   If IRIVCB < 0, river leakage will be printed if ICBCFL is set.

**ITMP**—is a flag and a counter.
   If ITMP < 0, river data from the last stress period will be reused.
   If ITMP ≥ 0, ITMP will be the number of reaches active during the current stress period.

**Recharge Package Input**

Input to the Recharge (RCH) Package is read from the unit specified in IUNIT(8).

**FOR EACH SIMULATION**

1. Record: NRCHOP  IRCHCB

2. Record: INRECH  INIRCH

3. Array: RECH(NCOL,NROW)

4. Array: IRCH(NCOL,NROW)

**NRCHOP**—is the recharge option code.
   1 - Recharge is only to the top grid layer.
   2 - Vertical distribution of recharge is specified in array IRCH.
   3 - Recharge is applied to the highest active cell in each vertical column.

**IRCHCB**—is a flag and a unit number.
   If IRCHCB > 0, unit number for cell-by-cell flow terms.
   If IRCHCB ≤ 0, cell-by-cell flow terms will not be printed or recorded.

**INRECH**—is the RECH read flag.
   If INRECH < 0, recharge fluxes from the preceding stress period are used.
   If INRECH > 0, an array of recharge fluxes, RECH (Lt-1), is read.

**INIRCH**—is similar to INRECH.
Well Package Input

Input for the Well (WEL) Package is read from the unit specified in IUNIT(2).

FOR EACH SIMULATION
1. Record: MXWELL IWELCB

FOR EACH STRESS PERIOD
2. Record: ITMP
3. Record: Layer Row Column 0
   (Input item 3 normally consists of one record for each
    well. If ITMP is negative or zero, item 3 is not read.)

MXWELL--is the maximum number of wells used at any time.
IWELCB--is a flag and a unit number.
   If IWELCB > 0, unit number for cell-by-cell flow terms.
   If IWELCB = 0, cell-by-cell flow terms will not be printed or
   recorded.
   If IWELCB < 0, well recharge will be printed whenever ICBCFL is set.

ITMP--is a flag and a counter.
   If ITMP < 0, well data from the last stress period will be reused.
   If ITMP > 0, ITMP will be the number of wells active during the
   current stress period.

Drain Package Input

Input to the Drain (DRN) Package is read from the unit specified in
IUNIT(3).

FOR EACH SIMULATION
1. Record: MXDRN IDRNCB

FOR EACH STRESS PERIOD
2. Record: ITMP
3. Record: Layer Row Col Elevation Cond
   (Input item 3 normally consists of one record for each drain.
    If ITMP is negative or zero, item 3 will not be read.)

MXDRN--is the maximum number of drain cells active at one time.
IDRNCB--is a flag and a unit number.
   If IDRNCB > 0, unit number for cell-by-cell flow terms.
   If IDRNCB = 0, cell-by-cell flow terms will not be printed or
   recorded.
   If IDRNCB < 0, drain leakage for each cell will be printed whenever
   ICBCFL is set.

ITMP--is a flag and a counter.
   If ITMP < 0, drain data from the last stress period will be reused.
   If ITMP > 0, ITMP will be the number of drains active during the current
   stress period.
Evapotranspiration Package Input

Input to the Evapotranspiration (EVT) Package is read from the unit specified in IUNIT (5).

FOR EACH SIMULATION
1. Record: NEVTOP IEVTCB

FOR EACH STRESS PERIOD
2. Record: INSURF INEVTR INEXDP INIEVT
3. Array: SURF
4. Array: EVTR
5. Array: EXDP

IF THE ET OPTION IS EQUAL TO TWO
6. Array: IEVT

NEVTOP--is the evapotranspiration (ET) option code.
1 - ET is calculated only for cells in the top grid layer.
2 - The cell for each vertical column is specified by the user in array IEVT.

IEVTCB--is a flag and a unit number.
If IEVTCB > 0, unit number for cell-by-cell flow terms.
If IEVTCB < 0, cell-by-cell flow terms will not be printed or recorded.

INSURF--is the ET surface (SURF) read flag.
If INSURF > 0, an array containing the ET surface elevation will be read.
If INSURF < 0, the ET surface from the preceding stress period will be reused.

INEVTR--is similar to INSURF.
INEXDP--is similar to INSURF.
INIEVT--is similar to INSURF.
General-Head Boundary Package Input

Input for the General-Head Boundary (GHB) Package is read from the unit specified in IUNIT(7).

FOR EACH SIMULATION
1. Record: MXBND IGHBCB

FOR EACH STRESS PERIOD
2. Record: ITMP Boundary
3. Record: Layer Row Column Head Cond
   (Input item 3 normally consists of one record for each GHB.
   If ITMP is negative or zero, item 3 is not read.)

MXBND--is the maximum number of general-head boundary cells at one time.

IGHBCB--is a flag and a unit number.
   If IGHBCB > 0, unit number for cell-by-cell flow terms.
   If IGHBCB = 0, cell-by-cell flow terms will not be printed or recorded.
   If IGHBCB < 0, boundary leakage for each cell will be printed whenever ICBCFL is set.

ITMP--is a flag and a counter.
   If ITMP < 0, GHB data from the preceding stress period will be reused.
   If ITMP > 0, ITMP is the number of general-head boundaries during the current stress period.

Strongly Implicit Procedure Package Input

Input to the Strongly Implicit Procedure (SIP) Package is read from the unit specified in IUNIT(9).

FOR EACH SIMULATION
1. Record: MXITER NPARM
2. Record: ACCL HCLOSE IPCALC WSEED IPRSIP

IPCALC--is a flag indicating where the iteration parameter seed will come from.
   0 - the seed will be entered by the user.
   1 - the seed will be calculated at the start of the simulation from problem parameters.

IPRSIP--is the printout interval for SIP.

Slice-Successive Overrelaxation Package Input

Input to the Slice-Successive Overrelaxation (SOR) Package is read from the unit specified in IUNIT(11).

FOR EACH SIMULATION
1. Record: MXITER
2. Record: ACCL HCLOSE IPRSOR

IPRSOR--is the printout interval for SOR.