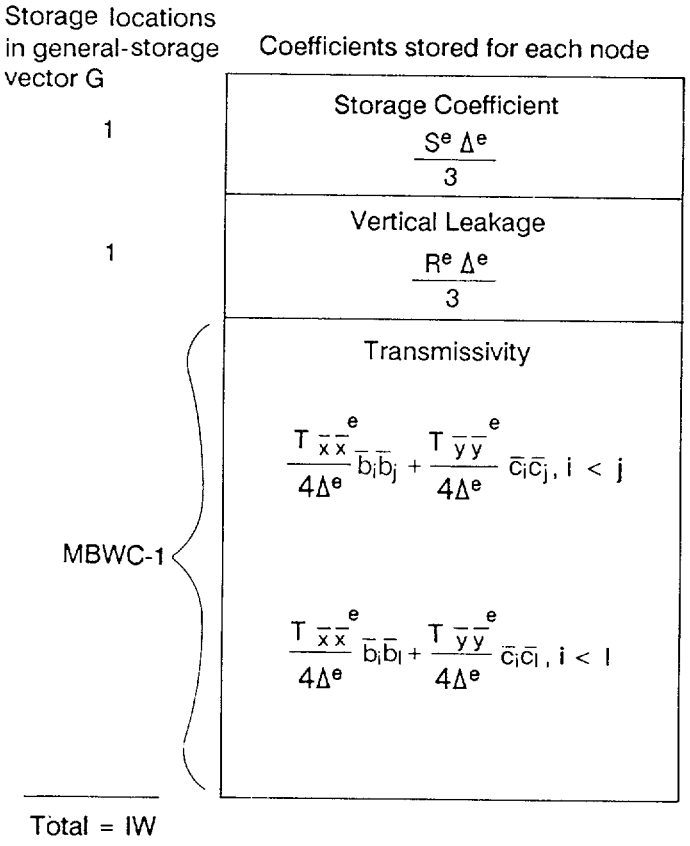
A MODULAR FINITE-ELEMENT MODEL (MODFE) FOR AREAL AND AXISYMMETRIC GROUND-WATER-FLOW PROBLEMS, PART 3: DESIGN PHILOSOPHY AND PROGRAMMING DETAILS



EXPLANATION

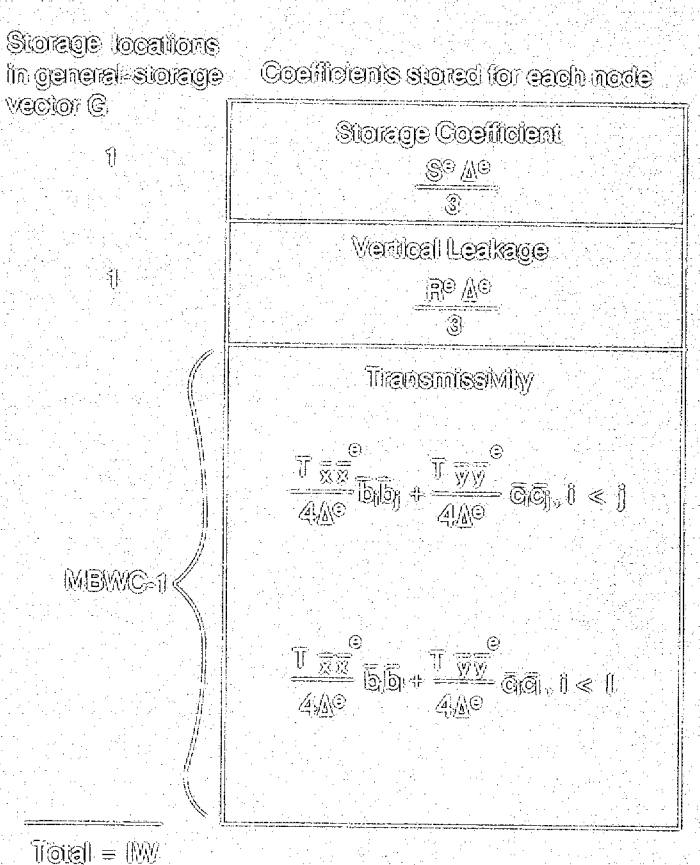
MBWC	Condensed-matrix bandwidth
S^e	Aquifer storage coefficient for element e
Δ^e	Area of element e
$T_{\bar{x}\bar{x}}^e$	Aquifer transmissivity, x direction, for element e
$T_{\bar{y}\bar{y}}^e$	Aquifer transmissivity, y direction, for element e
R^e	Vertical hydraulic conductance (vertical hydraulic conductivity divided by thickness) of confining bed for element e
$\bar{b}_i, \bar{b}_j, \bar{b}_l$ $\bar{c}_i, \bar{c}_j, \bar{c}_l$	Coordinate (basis) functions

U.S. Geological Survey

Open-File Report 91-471

Received with Ltr dated 2/1/93

A MODULAR FINITE-ELEMENT MODEL (MODFE) FOR AREAL AND AXISYMMETRIC GROUND-WATER-FLOW PROBLEMS, PART 3: DESIGN PHILOSOPHY AND PROGRAMMING DETAILS



EXPLANATION

MBWC Condensed-matrix bandwidth

S^e Aquifer storage coefficient for element e

Δ^e Area of element e

$T_{\bar{x}\bar{x}}^{e}$ Aquifer transmissivity, x direction, for element e

$T_{\bar{y}\bar{y}}^{e}$ Aquifer transmissivity, y direction, for element e

R^e Vertical hydraulic conductance (vertical hydraulic conductivity divided by thickness) of confining bed for element e

$\bar{b}_i, \bar{b}_j, \bar{b}_l$ Coordinate (basis) functions
$\bar{a}_i, \bar{a}_j, \bar{a}_l$

U.S. Geological Survey

Open-File Report 91-471

102.2

A MODULAR FINITE-ELEMENT MODEL (MODFE) FOR AREAL AND AXISYMMETRIC GROUND-WATER-FLOW PROBLEMS, PART 3: DESIGN PHYLOSOPHY AND PROGRAMMING DETAILS

By Lynn J. Torak

U.S. GEOLOGICAL SURVEY

Open-File Report 91-471

Doraville, Georgia

1992

U.S. DEPARTMENT OF THE INTERIOR

MANUEL LUJAN, JR., Secretary

U.S. GEOLOGICAL SURVEY

Dallas L. Peck, Director

For additional information
write to:

U.S. Geological Survey
Office of Ground Water
411 National Center
Reston, Virginia 22092
Telephone: (703) 648-5001

Copies of this report can be
purchased from:

U.S. Geological Survey
Books and Open-File Reports Section
Federal Center, Bldg. 810
Box 25425
Denver, Colorado 80225

CONTENTS

ILLUSTRATIONS

TABLES

A MODULAR FINITE-ELEMENT MODEL (MODFE) FOR AREAL AND AXISYMMETRIC GROUND-WATER-FLOW PROBLEMS

PART 3: DESIGN PHILOSOPHY AND PROGRAMMING DETAILS

By

Lynn J. Torak

ABSTRACT

A MODular, Finite-Element, digital-computer program (MODFE) was developed to simulate steady or unsteady-state, two-dimensional or axisymmetric ground-water-flow. The modular structure of MODFE places the computationally independent tasks that are performed routinely by digital-computer programs simulating ground-water flow into separate subroutines, which are executed from the main program by control statements. Each subroutine consists of complete sets of computations, or modules, which are identified by comment statements, and can be modified by the user without affecting unrelated computations elsewhere in the program. Simulation capabilities can be added or modified by either adding or modifying subroutines that perform specific computational tasks, and the modular-program stucture allows the user to create versions of MODFE that contain only the simulation capabilities that pertain to the ground-water problem of interest. MODFE is written in a Fortran programming language that makes it virtually device independent and compatible with desk-top personal computers and large mainframes.

MODFE uses computer storage and execution time efficiently by taking advantage of symmetry and sparseness within the coefficient matrices of the finite-element equations. Parts of the matrix coefficients are computed and stored as single-subscripted variables, which are assembled into a complete coefficient just prior to solution. Computer storage is reused during simulation to decrease storage requirements. Descriptions of subroutines that execute the computational steps of the modular-program structure are given in tables that cross reference the subroutines with particular versions of MODFE. Programming details of linear and nonlinear hydrologic terms are provided. Structure diagrams for the main programs show the order in which subroutines are executed for each version and illustrate some of the linear and nonlinear versions of MODFE that are possible. Computational aspects of changing stresses and boundary conditions with time and of mass-balance and error terms are given for each hydrologic feature. Program variables are listed and defined according to their occurrence in the main programs and in subroutines. Listings of the main programs and subroutines are given.

INTRODUCTION

This report describes the philosophy of the modular approach that was taken in designing a MODular, Finite-Element, digital-computer program (MODFE) for simulating ground-water flow in two dimensions, and the details concerning computational aspects of its simulation capabilities. It is intended to be a companion report to Part 1 (Torak, 1992), which is a user's manual that describes the simulation of ground-water flow by using MODFE, and to Part 3 (Cooley, 1992), which develops the finite-element equations that are approximated by MODFE. Simulation capabilities of MODFE are:

- transient or steady-state conditions,
- nonhomogeneous and anisotropic flow where directions of anisotropy change within the model region,
- vertical leakage from a semiconfining layer that contains laterally nonhomogeneous properties and elastic storage effects,
- point and areally distributed sources and sinks,
- specified-head (Dirichlet), specified-flux (Neumann), and head-dependent (Cauchy-type) boundary conditions,
- vertical cross sections,
- flow in axisymmetric, cylindrical coordinates (radial flow), confined and unconfined (water-table) conditions,
- partial drying and resaturation of a water-table aquifer,
- conversion between confined- and unconfined-aquifer conditions,
- nonlinear-leakage functions (for simulating line, point, or areally distributed sources and sinks),
- changing stresses and boundary conditions on a stress-period basis, time-step basis, or both, and
- zoned input of hydraulic properties and boundary conditions.

Background

Descriptions of the numerical representation of physical processes and hydrologic features contained in this and the companion reports have evolved within the past 10 years from material presented by the authors in the course "Finite-Element Modeling of Ground-Water Flow," held at the U.S. Geological Survey National Training Center in Denver, Colorado. This report formalizes the course material, which has been revised to incorporate comments and suggestions from attendees of the courses.

To demonstrate the modular approach used by MODFE, the computational steps that are required to simulate any ground-water flow problem are arranged into an ordered list or a generalized flow chart. Subroutines are listed in tables according to the computational steps that are performed, and brief descriptions of the subroutines are given. Versions of MODFE that require each subroutine also are indicated in the tables for reference.

Hydrologic terms are grouped into two categories: linear and nonlinear, and programming details for each group are given. These details, together with the variable lists and definitions, allow the interested user to modify the existing simulation capabilities of MODFE or to create additional capabilities.

Computational aspects of steady-state flow, changing stresses and boundary conditions with time, and of the water-balance summary are described to give the user necessary background information for either invoking these simulation capabilities or identifying the location within the program where computations occur. Details of computations within the equation-solving subroutines are provided to demonstrate how the theoretical development described in Cooley (1992) is put into practice with MODFE.

Methods of computer-storage allocation and usage that enhance the efficiency of MODFE are explained by using tables and diagrams. These descriptions are provided to promote an understanding of details of program design and to allow the user to add or modify simulation capabilities that are consistent with the design philosophy.

Purpose and Scope

This report provides technical details about the modular-program structure and about specific computations for simulation capabilities of MODFE. Discussions of modules, either sets of subroutines that add a particular simulation capability to MODFE, or sets of complete mathematical computations within subroutines, are kept brief, but are presented to familiarize the user with the overall philosophy of program design. Descriptions of how and where specific computational tasks are performed within MODFE, and definitions of corresponding program variables, are provided so that a user can follow the progression of computations that are required to transform the mathematical expressions described in Cooley (1992) into a working computer program. Together with lists of the main programs and subroutines, these descriptions and definitions provide the user with background information that is necessary for modifying existing simulation capabilities, adding new capabilities, or simply using MODFE in an appropriate manner to solve a particular ground-water-flow problem.

DESIGN PHILOSOPHY FOR A MODULAR APPROACH

The philosophy behind designing MODFE was to construct a digital-computer model of two-dimensional ground-water flow in which the independent computational steps that are necessary for simulation are contained in separate subroutines (fig. 1). The delineation of computational steps by subroutines is termed a modular approach. The order in which these steps are executed is controlled by a main program. Simulation capabilities and matrix-equation solvers are modified or added to the model by either modifying or adding subroutines and their controlling statements to the program. Consequently, a user can construct different versions of MODFE that contain different simulation capabilities and solvers, depending on the subroutines that are included (or excluded) in the program structure, and on the arrangement of controlling statements to subroutines in the main programs. Program structures for versions of MODFE that are possible with the modular approach are described in a later section.

Modules and Subroutines

The modular approach and design philosophy described above for the main programs were applied to the subroutines. Each subroutine consists of one or several sets of program statements (or modules) that perform a specific task. Each module requires specific input from the previous module, and all modules function to complete one of the computational steps shown in figure 1. For example, two modules within the

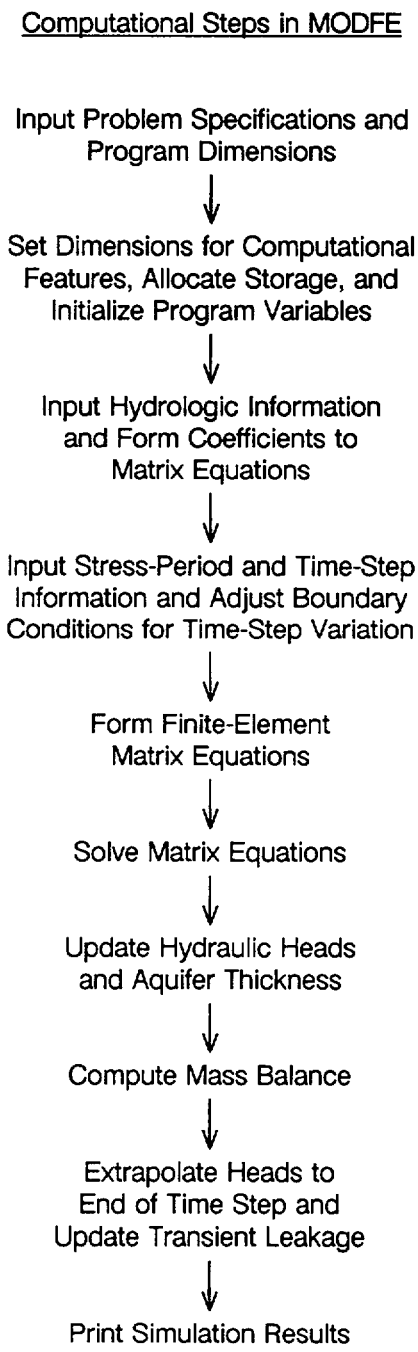


Figure 1.-- Diagram of generalized flow chart showing computational steps performed by subroutines of MODular Finite-Element model (MODFE).

subroutine that forms coefficients to matrix equations (subroutine FMCO) are: 1) computing the coefficients for coordinate functions and, 2) computing the element contribution to distributed recharge or discharge. The coordinate functions from the first module are used to compute element areas, which are used in the second module to compute the coefficient for distributed recharge or discharge. Modules are defined in the subroutines by comment statements and may be modified by the user to suit the specific needs of an aquifer problem. A listing of subroutines is given in the Appendices.

Efficient Use of Computer Storage and Processing Time

Another design consideration in the development of MODFE was the efficient use of computer storage and processing time. MODFE was designed to be compatible with computers that have small core storage, such as minicomputers or personal computers (PC's). All information is stored and processed either as single-subscripted variables (vector storage) or as unsubscripted variables; there are no multidimensional variables (array storage) used in MODFE. A general-storage vector, G, is used to store the vector lengths, and computer storage for each vector length is determined at the time of execution of MODFE. Storage locations within G are reused routinely during simulation to decrease storage requirements. Details about storage locations in G are given in the section "General-Storage Vector G."

Coefficients to matrix equations are formed and stored in a manner that uses computer storage and processing time efficiently. The information needed to assemble matrix equations for a node is stored as close together in the computer as physically possible. This storage scheme decreases the amount of transferring of information into and out of computer memory (commonly known as paging). Further decreases in computer storage and processing time are realized by eliminating equations at specified-head nodes from the matrix equation that is solved. Details of these processes are given in subsections of the section "Allocation of Computer Storage and Processing Time."

Device-Independent Programming

MODFE has been programmed in USA Standard Fortran, defined by the American National Standards Institute, Incorporated (ANSI) as standard USAS X3.9-1966, commonly termed Fortran 66. No extensions to this programming language that may be available from a particular computer manufacturer have been used in MODFE. However, each computer installation may require unique statements for program identification or for allocating large amounts of vector storage. For example, to use MODFE on the PRIME[1] computer, the general-storage vector G has been placed in a Fortran COMMON statement.

Adherence to the ANSI standard Fortran 66 allows MODFE to be used on many different types of computers. During development, MODFE had been used on at least eight different computers, including minicomputers and PC's. Each computer contained either different storage lengths for single-precision variables or either virtual or nonvirtual memory. Simulations on these computers produced nearly identical results, and only slight modifications were required at the beginning of the programs, as described above. Thus, MODFE is programmed to be as device independent as possible.

[1]Use of brand names in this report is for identification purposes only, and does not constitute endorsement by the U.S. Geological Survey. Use of brand names in this report is for identification purposes only, and does not constitute endorsement by the U.S. Geological Survey.

SIMULATION CAPABILITIES AND VERSIONS OF MODULAR FINITE-ELEMENT MODEL (MODFE)

The versions of MODFE that are possible from the modular program design are divided among three classes of ground-water-flow problems: (1) linear, steady or nonsteady state, (2) nonlinear, nonsteady state, and (3) nonlinear steady state. The linear versions of MODFE simulate hydrologic processes in which the equation formulation is unchanged during simulation. A list of linear versions and simulation capabilities is given in table 1. The nonlinear versions of MODFE simulate hydrologic processes with complex equation formulations that change during simulation. Separate nonlinear versions were constructed for steady- and nonsteady-state conditions because of the unique equation formulation required to simulate these types of ground-water-flow problems. Simulation capabilities of the nonlinear versions are listed in table 2 for nonsteady-state conditions and in table 3 for steady-state conditions.

Table 1.-- *Linear versions of MODular Finite-Element model (MODFE) and simulation capabilities*

Simulation capabilities of linear versions of MODFE	
Nonhomogeneous, anisotropic flow having changing directions of anisotropy within the model region Steady vertical leakage (no storage effects) Point and areally distributed sources and sinks Specified head (Dirichlet), specified flux (Neumann), and head-dependent (Cauchy-type) boundary conditions	Axi-symmetric radial flow Zoned input of hydraulic properties and boundary conditions Nonsteady-state or steady-state conditions Vertical cross sections Changing stresses and boundary conditions with time

Simulation options	Solver options	
	Direct, symmetric-Doolitle method	Iterative, MICCG method
Steady vertical leakage (no storage effects)	LMFE1	LMFE2
Vertical leakage having storage effects (transient leakage)	LMFE3	LMFE4

Table 2.-- *Nonlinear versions of MODular Finite-Element model (MODFE) and simulation capabilities*

Simulation capabilities of nonlinear versions of MODFE	
Nonhomogeneous, anisotropic flow having changing directions of anisotropy within the model region Steady vertical leakage (no storage effects) Point and areally distributed sources and sinks Specified head (Dirichlet), specified flux (Neumann), and head-dependent (Cauchy-type) boundary conditions Axi-symmetric radial flow	Zoned input of hydraulic properties and boundary conditions Nonsteady-state conditions Unconfined (water-table) conditions Partial drying and resaturation of a water-table aquifer Conversion between confined- and unconfined-aquifer conditions Change stresses and boundary conditions with time

Simulation options	Solver options	
	Direct, triangular-decomposition method	Iterative, MICCG method
Steady vertical leakage (no storage effects)	NLMFE1	NLMFE2
Vertical leakage having storage effects (transient leakage)	NLMFE3	NLMFE4
Nonlinear steady vertical leakage	NLMFE5	NLMFE6
Nonlinear head-dependent (Cauchy-type) boundaries	NLMFE7	NLMFE8

Table 3.-- *Nonlinear steady-state versions of MODular Finite-Element model (MODFE) and simulation capabilities*

Simulation capabilities of nonlinear versions of MODFE	
Nonhomogeneous, anisotropic flow with changing directions of anisotropy within the model region Steady vertical leakage (no storage effects) Point and areally distributed sources and sinks Specified head (Dirichlet), Specified flux (Neumann), and head-dependent (Cauchy-type) boundary conditions Axi-symmetric radial flow	Zoned input of hydraulic properties and boundary conditions Steady-state conditions Unconfined (water-table) conditions Partial drying and resaturation of a water-table aquifer Conversion between confined- and unconfined-aquifer conditions

Simulation options	Solver options	
	Iterative, MICCG method	Direct, triangular-decomposition method
Water-table conditions only	NSSFE1	NSSFE2
Nonlinear steady vertical leakage	NSSFE3	NSSFE4
Nonlinear head-dependent (Cauchy-type) boundaries	NSSFE5	NSSFE6

Each version of MODFE contains different options for simulation capabilities and for solution methods to the finite-element matrix equations. Hence, simulation or solution options create different structures of the main program and different versions of MODFE. Selection of options and program structures is user dependent; that is, the user determines which simulation or solution options are applicable to the ground-water-flow problem and constructs an appropriate version of MODFE. Structures of main programs to the versions listed in tables 1-3 are given in the following section. Details of using the simulation capabilities are given in Torak (1992).

The following naming convention is adopted to identify each version of MODFE. The four linear versions listed in table 1 are termed LMFEn for "Linear Modular Finite Element, version n," where n = 1-4. The eight nonlinear versions listed in table 2 are termed NLMFEn for "NonLinear, Modular Finite Element, version n," where n = 1-8. The six nonlinear steady-state versions listed in table 3 are termed NSSFEn for "Nonlinear, Steady-State, Finite-Element, version n," where n = 1-6. Note that the linear versions of MODFE simulate nonsteady-state and steady-state conditions, whereas separate versions are used for the nonlinear conditions.

Program Structures and Lists of Main Programs

The order in which subroutines are executed and computational steps are performed by the main programs is shown diagrammatically in figures 2-12 for the versions of MODFE listed in tables 1-3. Each subroutine name shown in the structure diagrams represents one or more computational steps in the generalized flow chart of figure 1. The computational steps corresponding to each subroutine are described in tables in the following section, and the versions of MODFE that use each subroutine are indicated in the tables for reference. Subroutine names in the structure diagrams are replaced by Fortran CALL statements to form the main programs. Computer listings of main programs and subroutines are given in the Appendices.

Subroutine names that are linked to the main programs by dashed lines in the structure diagrams contain the simulation options listed in tables 1-3. The Fortran CALL statements to these subroutines are inserted into the main program according to the order indicated by the structure diagrams, and the corresponding subroutines are added to MODFE for compilation. Thus, if a simulation capability defined by these subroutines is not required for the aquifer problem, then the Fortran CALL statements and the corresponding subroutines are removed from the program creating a version of MODFE having a new program structure.

Subroutine names appearing in parentheses in the structure diagrams replace those that are located beside them in the structure. For example, in figure 2, subroutines required for the iterative, MICCG, solution method (subroutines INITCG, SETCG, and MICCG) replace subroutines used for the direct method (subroutines INITB, SETB, and BAND). This replacement creates another version of MODFE, which is indicated by the program name in parentheses at the top of the structure diagram (LMFE2 in this example).

Two new program structures result from combining simulation capabilities of the nonlinear versions (figs. 8 and 12). The program structure shown in figure 8 represents a nonsteady-state version of MODFE that can simulate nonlinear steady vertical leakage, nonlinear head-dependent (Cauchy-type) boundaries, and nonlinear point sinks. A steady-state version of MODFE having the same simulation capabilities as the version shown in figure 8 is represented by the structure diagram in figure 12. Note that the order in which Fortran CALL statements are added to the main program determines the order of inputs and other computations for these features. Therefore, although subroutines for nonlinear steady vertical leakage (subroutine names that begin with "VN") can be called by the main program before subroutines for nonlinear head-dependent (Cauchy-type) boundaries and nonlinear point sinks (subroutine names that begin with "GN"), this new order of subroutines in the main program is not consistent with the order of inputs given in the input instructions in Torak (1992). To preserve the order of inputs as given in Torak (1992), the main programs for these versions of MODFE should be structured according to the diagrams in figures 8 and 12.

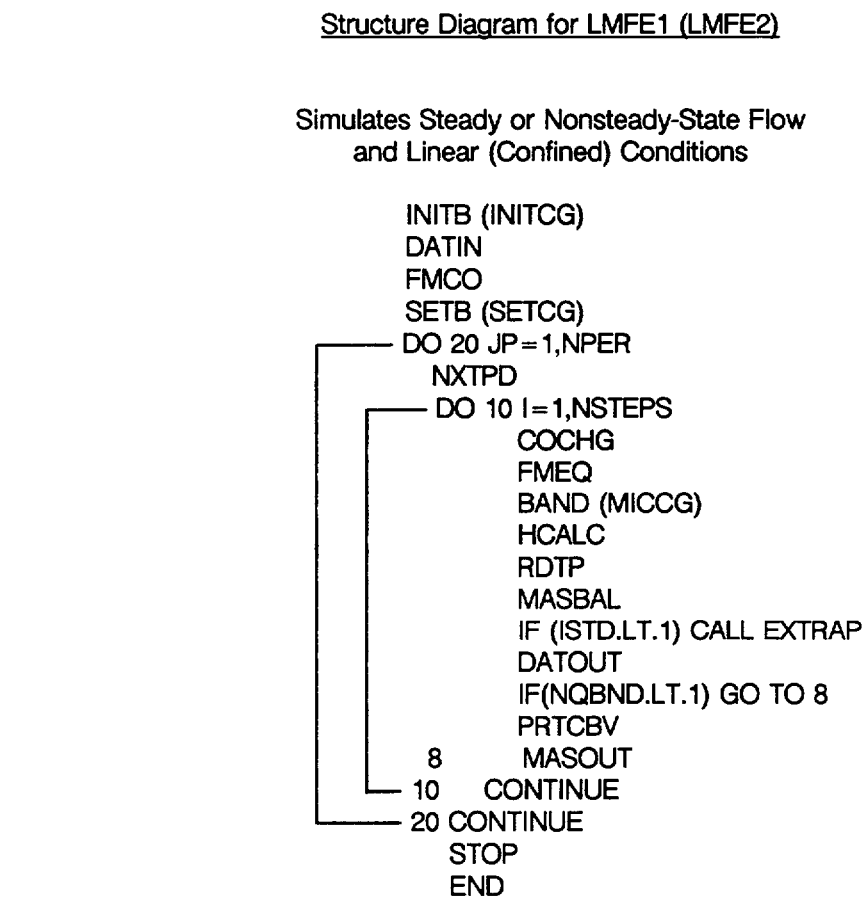


Figure 2.-- Main-program structure for MODular Finite-Element model (MODFE) version LMFE1 (LMFE2), simulates steady or nonsteady-state flow and linear (confined) conditions.

Structure Diagram for LMFE3 (LMFE4)

Simulates Steady or Nonsteady-State Flow, Linear (Confined)
Conditions and Vertical Leakage of Water Stored Elastically in a Confining Bed
(Transient Leakage)

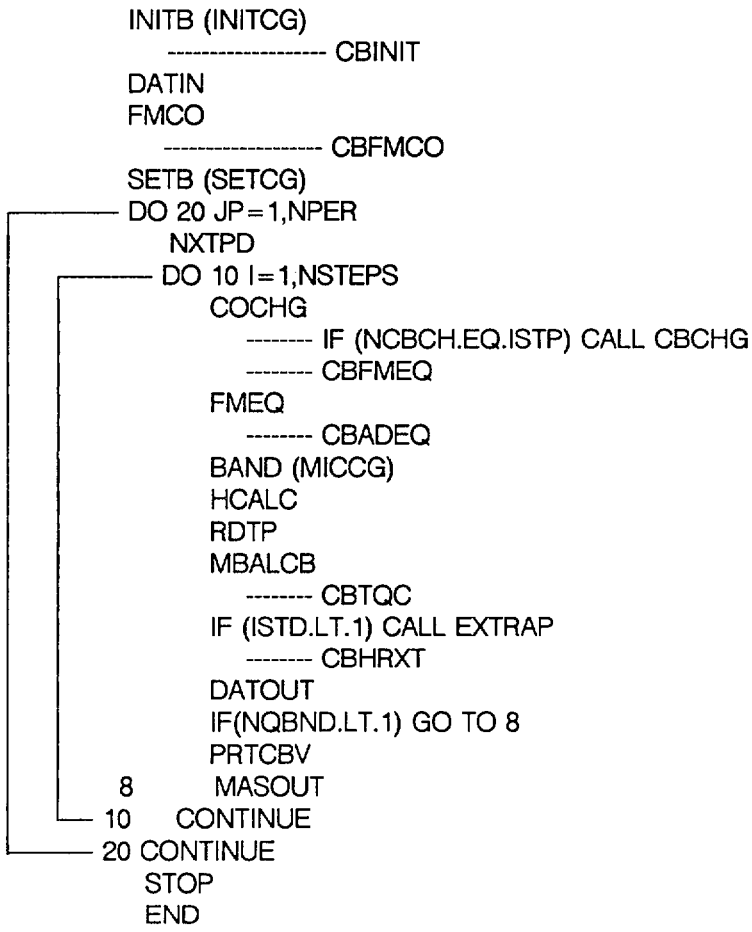


Figure 3.-- Main-program structure for MODular Finite-Element model(MODFE) version LMFE3 (LMFE4), simulates steady or nonsteady-state flow, linear (confined) conditions and vertical leakage of water stored elastically in a confining bed (transient leakage).

Structure Diagram for NLMFE1 (NLMFE2)

Simulates Nonsteady-State Flow and
Unconfined (Water-Table) Conditions

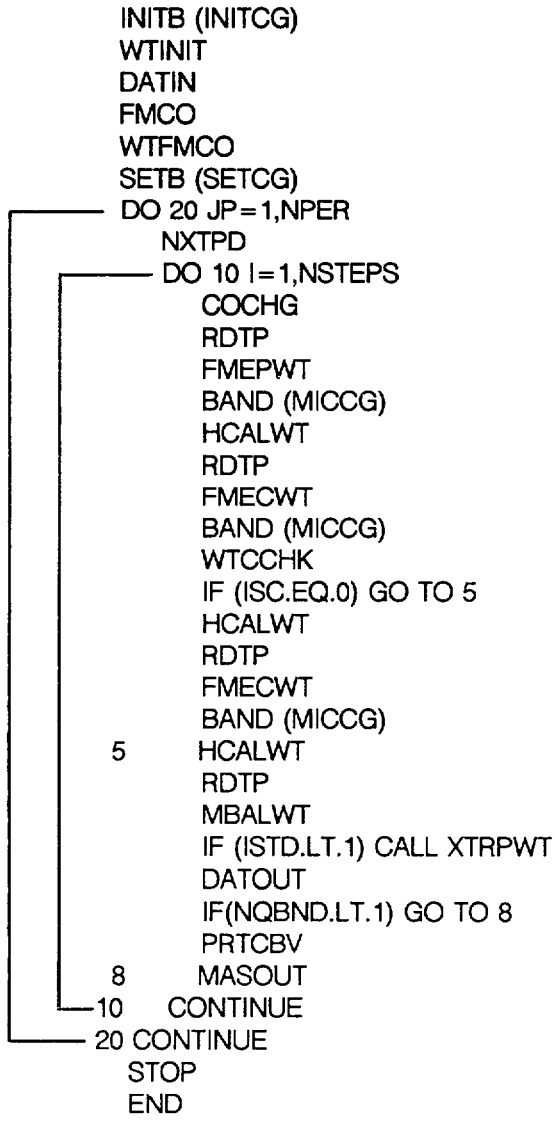


Figure 4.-- Main-program structure for MODular Finite-Element model (MODFE) version NLMFE1 (NLMFE2), simulates nonsteady-state flow and unconfined (water-table) conditions.

Structure Diagram for NLMFE3 (NLMFE4)

Simulates Nonsteady-State Flow, Unconfined (Water-Table) Conditions, and Vertical Leakage of Water Stored Elastically in a Confining Bed (Transient Leakage)

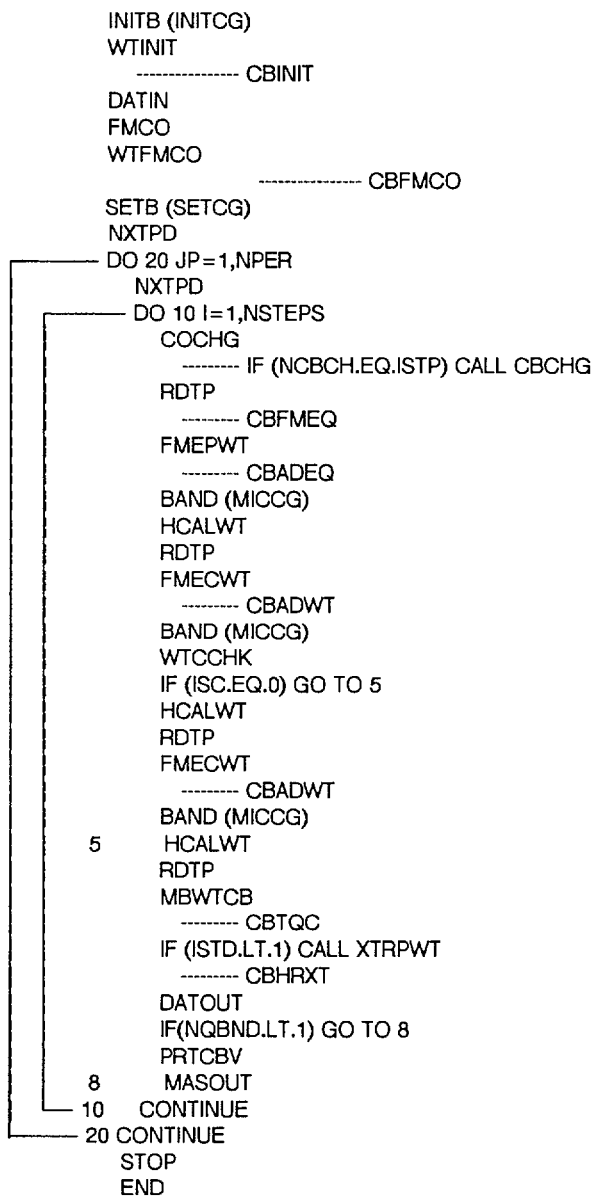


Figure 5.-- Main-program structure for MODular Finite-Element model (MODFE) version NLMFE3 (NLMFE4), simulates nonsteady-state flow, unconfined (water-table) conditions, and vertical leakage of water stored elastically in a confining bed (transient leakage).

Structure Diagram for NLMFE5 (NLMFE6)

Simulates Nonsteady-State Flow, Unconfined (Water-Table) Conditions, and Nonlinear Steady Vertical Leakage

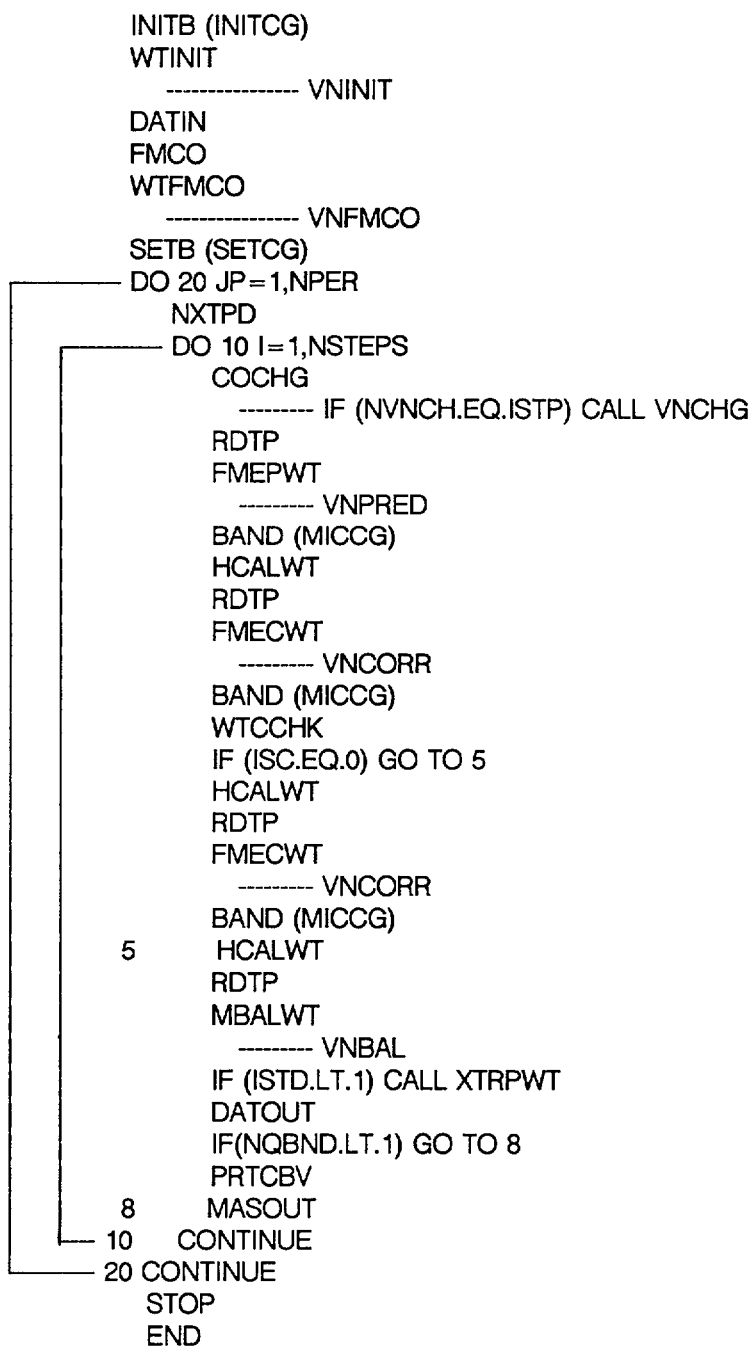


Figure 6.-- Main-program structure for MODular Finite-Element model (MODFE) version NLMFE5 (NLMFE6), simulates nonsteady-state flow, unconfined (water-table) conditions, and nonlinear steady vertical leakage.

Structure Diagram for NLMFE7 (NLMFE8)

Simulates Nonsteady-State Flow, Unconfined (Water-Table)
Conditions, Nonlinear Head-Dependent (Cauchy-type)
Boundaries, and(or) Nonlinear Point Sinks

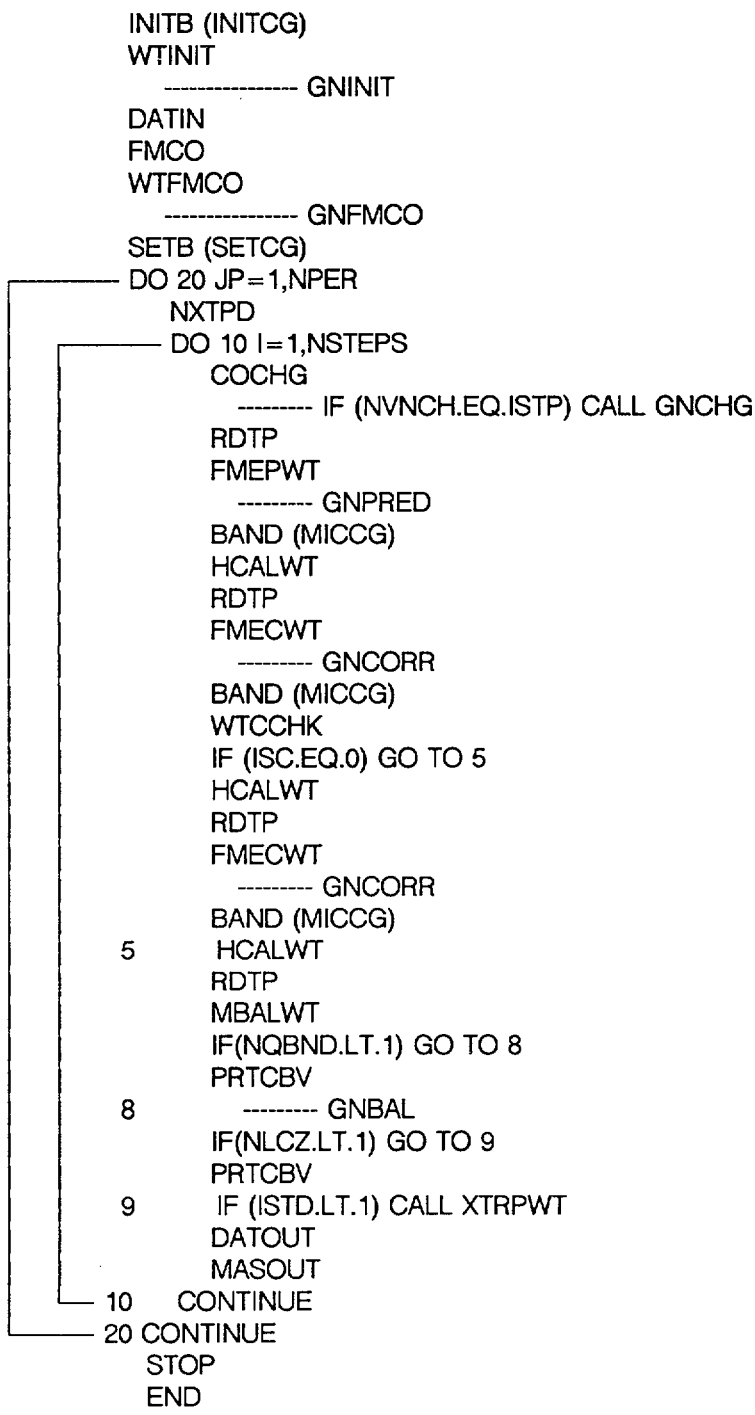


Figure 7.-- Main-program structure for MODular Finite-Element model (MODFE) version NLMFE7 (NLMFE8), simulates nonsteady-state flow, unconfined (water-table) conditions, nonlinear head-dependent (Cauchy-type) boundaries, and (or) nonlinear point sinks.

Structure Diagram for NLMFE5 (NLMFE6) Combined with NLMFE 7 (NLMFE8)

Simulates Nonsteady-State Flow, Unconfined (Water-Table) Conditions, Nonlinear Head-Dependent (Cauchy-type) Boundaries and (or) Nonlinear Point Sinks, and Nonlinear Steady Vertical Leakage

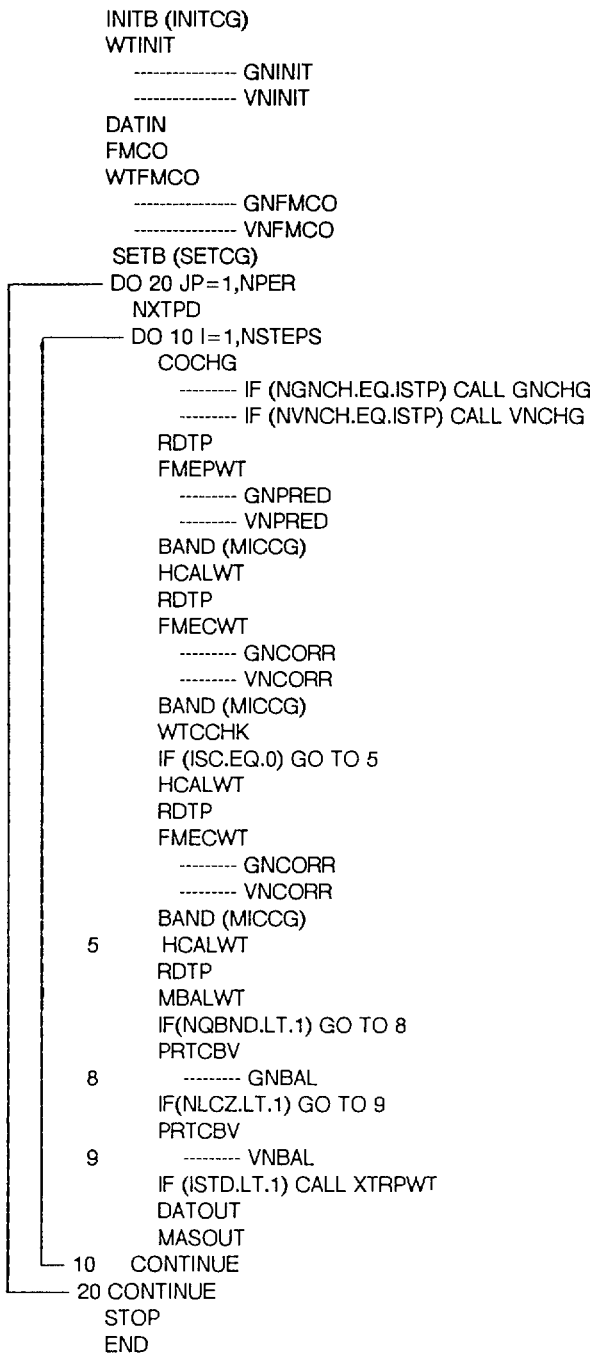


Figure 8.-- Main-program structure for MODular Finite-Element model (MODFE) version NLMFE5 (NLMFE6) combined with NLMFE7 (NLMFE8), simulates nonsteady-state flow, unconfined (water-table) conditions, nonlinear head-dependent (Cauchy-type) boundaries and (or) nonlinear point sinks, and nonlinear steady vertical leakage.

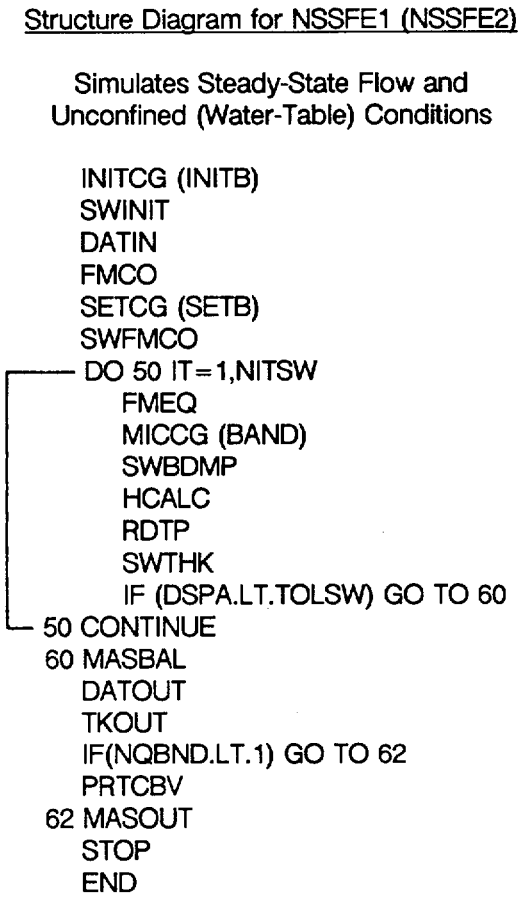


Figure 9.-- Main-program structure for MODular Finite-Element model (MODFE) version NSSFE1 (NSSFE2), simulates steady-state flow and unconfined (water-table) conditions.

Structure Diagram for NSSFE3 (NSSFE4)

Simulates Steady-State Flow, Unconfined (Water-Table)
Conditions, and Nonlinear Steady Vertical Leakage

```
      INITCG (INITB)
      SWINIT
         ---------- VNINIT
      DATIN
      FMCO
      SETCG (SETB)
      SWFMCO
         ---------- VNFMCO
   ---- DO 50 IT=1,NITSW
   |     FMEQ
   |     ---------- VNPRED
   |     MICCG (BAND)
   |     SWBDMP
   |     HCALC
   |     RDTP
   |     SWTHK
   |     IF (DSPA.LT.TOLSW) GO TO 60
   -- 50 CONTINUE
   60 MASBAL
         ---------- VNBLSS
      DATOUT
      TKOUT
      IF(NQBND.LT.1) GO TO 62
      PRTCBV
   62 MASOUT
      STOP
      END
```

Figure 10.-- Main-program structure for MODular Finite-Element model (MODFE) version NSSFE3 (NSSFE4), simulates steady-state flow, unconfined (water-table) conditions, and nonlinear steady vertical leakage.

Structure Diagram for NSSFE5 (NSSFE6)

Simulates Steady-State Flow, Unconfined (Water-Table)
Conditions, Nonlinear Head-Dependent (Cauchy-type)
Boundaries, and (or) Nonlinear Point Sinks

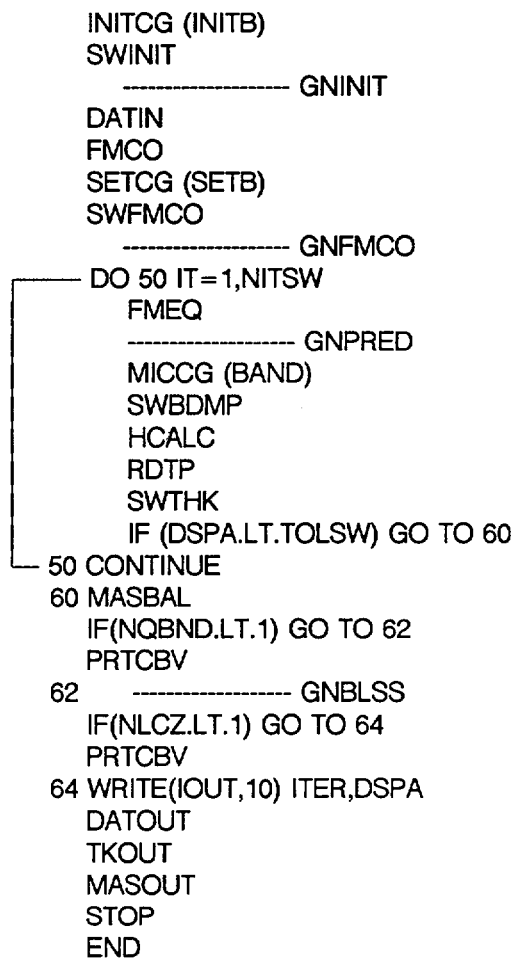


Figure 11.-- Main-program structure for MODular Finite-Element model (MODFE) version NSSFE5 (NSSFE6), simulates steady-state flow, unconfined (water-table) conditions, nonlinear head-dependent (Cauchy-type) boundaries, and (or) nonlinear point sinks.

Structure Diagram for NSSFE3 (NSSFE4) Combined with NSSFE5 (NSSFE6)

Simulates Steady-State Flow, Unconfined (Water-Table) Conditions, Nonlinear Head-Dependent (Cauchy-type) Boundaries and (or) Nonlinear Point Sinks, and Nonlinear Steady Vertical Leakage

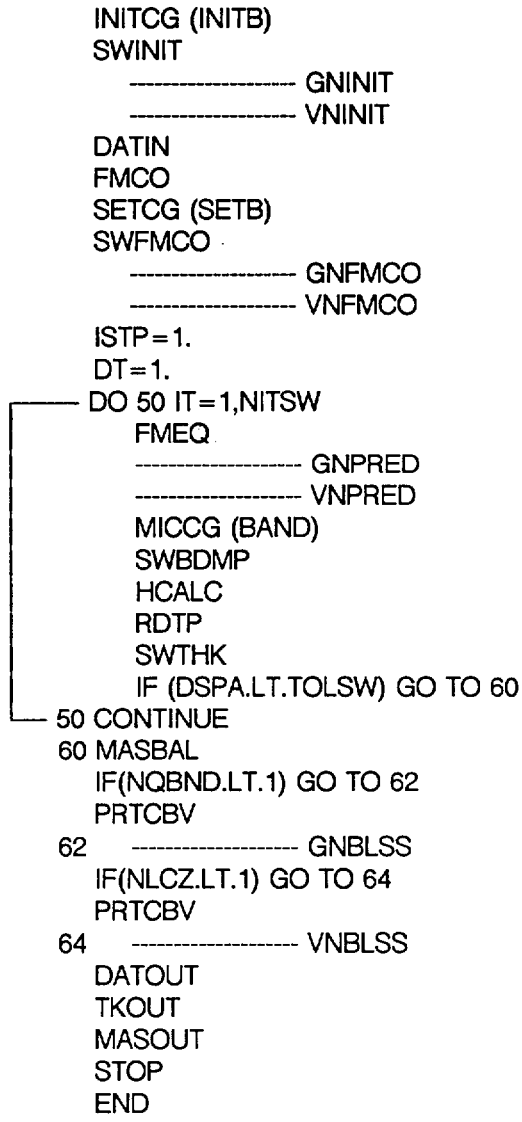


Figure 12.-- Main-program structure for MODular Finite-Element model(MODFE) version NSSFE3 (NSSFE4) combined with NSSFE5 (NSSFE6), simulates steady-state flow, unconfined (water-table) conditions, nonlinear head-dependent (Cauchy-type) boundaries and (or) nonlinear point sinks, and nonlinear steady vertical leakage.

Table 4.-- *Naming convention for subroutines*

Identifier	Computational step
B	Solves matrix equations by using the direct symmetric-Doolittle method.
CG	Solves matrix equations by using the conjugate-gradient method.
CHG	Adjust or change stresses and boundary conditions (table 8).
EQ, FME PRED, CORR	Form finite-element matrix equations (table 9).
FMCO	Form coefficients to matrix equations (table 7).
HCAL	Update hydraulic heads (table 11).
INIT	Input problem specifications and initialize program variables (tables 5 and 6).
MAS, BAL MB, BL	Compute mass balance (table 12).
OUT, PRT	Print hydrologic information and simulation results.
XT	Extrapolate heads to end of time step (table 13).
	Simulation capability
CB	Vertical leakage with storage effects (transient leakage).
GN	Nonlinear head-dependent (Cauchy-type) boundaries and nonlinear point sinks.
SS, SW	Nonlinear, steady-state conditions.
VN	Nonlinear steady vertical leakage.
WT	Water-table conditions.

Descriptions of the computational steps performed by each subroutine in MODFE are given in tables 5-14. Subroutines are listed according to the computational steps in the generalized flow chart (fig. 1). Versions of MODFE that require a subroutine are indicated by x's in the tables. Subroutines that are required by the versions listed in parentheses are indicated by the symbol (x). Subroutines that perform utility-type functions, such as printing or reading information and computing the reduced-matrix bandwidth (see section "Reduced Matrix A"), are given in table 15.

Table 5.-- *Subroutines that input problem specifications and program dimensions*

Subroutine	Description of computations (numbers in parentheses corrspond to equations in Cooley, 1992)	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
CBHRXT	Update source-bed head for transient leakage at end of time step by extrapolation according to equation (57) applied to source-bed head.		x (x)		x (x)					
CBTQC	Compute transient leakage flux for time step according to equation (174).		x (x)		x (x)					
EXTRAP	Update hydraulic head at end of time step by extrapolation according to equation (57).	x (x)	x (x)							
XTRPWT	Update hydraulic head and aquifer thickness at end of time step by extrapolation according to equations (57) and (85).			x (x)	x (x)	x (x)	x (x)			

PROGRAMMING DETAILS OF LINEAR HYDROLOGIC TERMS

The following sections give programming details about mathematical processes and computations that pertain to input and to the formation of coefficients and equations for the linear versions of MODFE (table 1). Additional details concerning specific computations or inputs are given with the program-variable descriptions in the appendices and in sections of Torak (1992) that describe the corresponding simulation capability. A development of the equations referenced in the following sections is contained in Cooley (1992).

Descriptions of the programming details for linear hydrologic terms in MODFE refer to the finite-element matrix equations for steady- and nonsteady-state conditions frequently; thus, these equations are restated here for reference. For nonsteady-state conditions, the finite-element matrix equation is given by equation (58) in Cooley (1992):

$$\left(\frac{\underline{C}}{\left(\frac{2}{3}\right)\Delta t_{n+1}}\right)\underline{\delta} = \underline{B} - \underline{A}\hat{h}_n \qquad (1)$$

For steady-state conditions, the finite-element matrix equation is given by equation (232) in Cooley (1992):

$$\underline{A}\,\underline{\delta} = \underline{B} - \underline{A}\underline{h}_0 \qquad (2)$$

Terms contained in equations (1) and (2) are defined by the following equations in Cooley (1992): the C and

Table 6.-- *Subroutines that set dimensions, allocate storage, and initialize variables*

Subroutine	Description of computations	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
CBINIT	Allocate storage for transient leakage terms and initialize storage locations to zero.		x (x)		x (x)					
GNINIT	Allocate storage for α coefficients, node numbers, and controlling heads and altitudes for nonlinear head-dependent (Cauchy-type) boundaries and nonlinear point sinks. Initialize storage locations to zero.						x (x)			x (x)
INITB	Set dimensions for storage vectors, allocate storage for condensed matrix, reduced matrix, element areas and incidences, nodal coordinates, known fluxes, hydraulic head, boundary or external head, α coefficient, specified flux, boundary-node numbers, source-bed head, boundary-condition zone numbers, time steps, and node-indicator vector. Initialize time and flow-balance terms and storage locations for above terms to zero.	x	x	x	x	x	x	x	x	x
INITCG	Same as INITB, and allocate storage for vectors X, P, and R for MICCG method.	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)
SWINIT	Allocate storage for aquifer thickness and altitude of top of aquifer, and initialize storage locations to zero.							x (x)	x (x)	x (x)
VNINIT	Allocate storage for hydraulic conductance term and altitudes for nonlinear steady vertical leakage. Initialize storage locations to zero.					x (x)			x (x)	
WTINIT	Allocate storage for head-change vector, aquifer thickness, altitude of top of aquifer, and specific yield for water-table simulations. Initialize storage locations to zero.			x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)

Table 7.-- *Subroutines that input hydrologic information and form coefficients to matrix equations*

Subroutine	Description of input or coefficients (numbers in parentheses correspond to equations in Cooley, 1992)	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
DATIN	Input indicators for axi-symmetric radial flow, scale factor and title, indicators to suppress or print selected input, nodal coordinates, hydraulic head, source bed head, and linear boundary conditions. Add volumetric flow rate from point sources or sinks to vector of known terms, (50). Evaluate vectors that store node indicators, mean head, and mean head change.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
CBFMCO	Input zone values for vertical hydraulic conductivity and specific storage for transient leakage. Form transient leakage coefficients according to (167) and assign values to regression parameters.		x (x)		x (x)					
FMCO	Input hydraulic properties by zone for aquifer and confining bed and form coefficients for transmissivity, (39), (40), or (43), or hydraulic conductivity, (72); storage coefficient, (36); vertical hydraulic conductance and linear boundary terms, (37); and known fluxes, (50). Input element incidences.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
GNFMCO	Input indicators to suppress or print values for nonlinear head-dependent (Cauchy-type) boundaries and nonlinear point sinks, input values for nonlinear head-dependent (Cauchy-type) boundaries, and nonlinear point sinks. Form coefficients according to (155).						x (x)			x (x)
SWFMCO	Input indicators to suppress or print aquifer thickness or altitude of aquifer top, input aquifer thickness and altitude of aquifer top. Form transmissivity coefficients according to (244).							x (x)	x (x)	x (x)
VNFMCO	Input indicators to suppress or print values for nonlinear steady vertical leakage. Form coefficients according to (119) and (132).					x (x)			x (x)	
WTFMCO	Input aquifer thickness, altitude of top of aquifer, and specific yield for water-table simulations. Form coefficients according to (36) and (47).			x (x)	x (x)	x (x)	x (x)			

Table 8.-- *Subroutines that input stress-period and time-step information and adjust boundary conditions for time-step variation*

Subroutine	Description of input or computation	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
CBCHG	Changes to source-bed head for transient leakage and to indicators for additional changes.		x (x)		x (x)					
COCHG	Changes to point source and sink, areally distributed source and sink, source-bed head for steady-vertical leakage, specified flux or head on Cauchy-type boundary, and specified head, and to indicators for additional changes.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
GNCHG	Changes to boundary and external head on nonlinear, head-dependent (Cauchy-type) boundary and to indicators for additional changes.						x (x)			
NXTPD	Initial values of indicators for time varying stresses and boundary conditions and time-step sizes.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
VNCHG	Changes to controlling head or altitude for nonlinear steady vertical leakage and to indicators for additional changes.					x (x)				

Table 9.-- *Subroutines that form finite-element matrix equations*

Subroutine	Description of computations or components formed (numbers in parentheses correspond to equations in Cooley,1992)	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
CBADEQ	Adds transient leakage components into matrix equations (58) and (76).		x (x)		x (x)					
CBADWT	Adds transient leakage components into corrector-step equation (80).				x (x)					
CBFMEQ	Transient leakage components formed according to equations (199)-(205).		x (x)		x (x)					
FMECWT	Water-table components of corrector-step equation (80).			x (x)	x (x)	x (x)	x (x)			
FMEPWT	Water-table components of predictor-step equation (76).			x (x)	x (x)	x (x)	x (x)			
FMEQ	Assembles transmissivity, storage coefficient, and linear boundary-condition terms to matrix equations (58), (232), and (234).	x (x)	x (x)					x (x)	x (x)	x (x)
GNCORR	Forms components of nonlinear head-dependent (Cauchy-type) flux and nonlinear point sinks according to equations (108)-(110) and (124)-(127) and adds components into corrector-step equation (80).						x (x)			
GNPRED	Forms components of nonlinear head-dependent (Cauchy-type) flux and nonlinear point sinks according to equations (108), (124), (127), (246), (247), and (253). Adds components into matrix equation (76), for predictor step, and (234) for nonlinear steady-state conditions.						x (x)			x (x)
VNCORR	Forms components of nonlinear, steady vertical leakage according to equations (124)-(127) and (144)-(151) and adds components into corrector-step equation (80).					x (x)				
VNPRED	Forms components of nonlinear, steady vertical leakage according to equations (124), (127), (144), (147), and (248)-(251). Adds components into matrix equation (76), for the predictor step, and and (234) for nonlinear steady-state conditions.					x (x)				x (x)
WTCCHK	Checks for unpredicted conversion or nonconversion between confined and unconfined conditions at a node and sets value of program variable ISC to indicate need need for second corrector step.			x (x)	x (x)	x (x)	x (x)			

Table 10.-- *Subroutines that solve matrix equations*

Subroutine	Description of computations	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
BAND	Computes average head change for time step or total head change for iteration level by using the direct, symmetric-Doolittle method.	x	x	x	x	x	x	x	x	x
MICCG	Computes average head change for time step or total head change for iteration level by using an iterative, modified incomplete-Cholesky, conjugate-gradient method.	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)

Table 11.-- *Subroutines that update hydraulic head and aquifer thickness*

Subroutine	Description of computations	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
HCALC	Updates hydraulic head to average value for time step or to new value for iteration level by using head changes from matrix solver.	x (x)	x (x)					x (x)	x (x)	x (x)
HCALWT	Updates hydraulic head to average value for predictor or corrector steps by using head changes obtained from matrix solver.			x (x)	x (x)	x (x)	x (x)			
SWBDMP	Damps computed head change obtained during water-table iteration.							x (x)	x (x)	x (x)
SWTHK	Updates aquifer thickness and transmissivity after water-table iteration.							x (x)	x (x)	x (x)

Table 12.-- *Subroutines that compute mass balance*

Subroutine	Terms computed	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
GNBAL	Volumetric rates for time step and total volumes during simulation for nonlinear head-dependent (Cauchy-type) boundaries and nonlinear point sinks. Augments flow rate at specified-head boundaries and flow imbalance of rates.						x (x)			
GNBLSS	Volumetric rates for time step and total volumes during simulation for nonlinear head-dependent (Cauchy-type) boundaries and nonlinear point sinks. Augments flow rate at specified-head boundaries and flow imbalance of rates.									x (x)
MASBAL	Volumetric rates for time step or for steady-state simulation for: accumulation of water in aquifer storage, steady vertical leakage, flow across specified flux and head-dependent (Cauchy-type) boundaries and at specified-head boundaries, and flow imbalance.	x (x)						x (x)	x (x)	x (x)
MBALCB	Same as MASBAL, and volumetric flow rate from transient leakage.		x (x)							
MBALWT	Same as MASBAL, except storage effects from conversion between confined and unconfined conditions are computed for accumulation of water in aquifer storage.			x (x)		x (x)	x (x)			
MBWTCB	Combined terms from MBALCB and MBALWT.				x (x)					
VNBAL	Volumetric rates for time step and total volumes during simulation for nonlinear steady vertical leakage. Augments flow rate at specified-head boundaries and flow imbalance of rates.					x (x)				
VNBLSS	Volumetric rates for time step and total volumes during simulation for nonlinear steady vertical leakage. Augments flow rate at specified-head boundaries and flow imbalance of rates.								x (x)	

Table 13.-- *Subroutines that extrapolate heads to end of time step and update transient leakage*

Subroutine	Description of computations (numbers in parentheses corrspond to equations in Cooley, 1992)	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
CBHRXT	Update source-bed head for transient leakage at end of time step by extrapolation according to equation (57) applied to source-bed head.		x (x)		x (x)					
CBTQC	Compute transient leakage flux for time step according to equation (174).		x (x)		x (x)					
EXTRAP	Update hydraulic head at end of time step by extrapolation according to equation (57).	x (x)	x (x)							
XTRPWT	Update hydraulic head and aquifer thickness at end of time step by extrapolation according to equations (57) and (85).			x (x)	x (x)	x (x)	x (x)			

Table 14.-- *Subroutines that print simulation results*

Subroutine	Description of terms printed	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
DATOUT	Hydraulic head at end of time step or steady-state simulation.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
MASOUT	Volumetric rates for time step or steady-state simulation and volumes for total simulation time for: specified-head boundaries, accumulation of water in aquifer storage, point sources or sinks, areally distributed sources or sinks, steady or transient vertical leakage, and linear and nonlinear head-dependent (Cauchy-type) boundaries, and flow imbalance of volumes and volumetric rates.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
PRTCBV	Nodal values of volumetric rates for time step or steady-state simulation and volume and rates by zone for linear and nonlinear head-dependent (Cauchy-type) boundaries.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
TKOUT	Computed aquifer thickness by node.							x (x)	x (x)	x (x)

Table 15.-- *Subroutines that perform utility-type functions*

Subroutine	Description of utility functions	Versions of MODFE that require subroutine								
		LMFE1 (LMFE2)	LMFE3 (LMFE4)	NLMFE1 (NLMFE2)	NLMFE3 (NLMFE4)	NLMFE5 (NLMFE6)	NLMFE7 (NLMFE8)	NSSFE1 (NSSFE2)	NSSFE3 (NSSFE4)	NSSFE5 (NSSFE6)
PRTOA	Prints values in three columns for: initial and computed hydraulic head, initial source-bed head and aquifer thickness, altitude of top of aquifer, controlling head or altitude for nonlinear steady vertical leakage, and computed aquifer thickness.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
PRTOB	Prints values of nodal coordinates (two vectors) in two double columns.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
RDTP	Reads terms for condensed matrix previously written to storage by using Fortran unit 55.	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)	x (x)
SETB	Computes reduced-matrix bandwidth and writes condensed-matrix components, element areas, and incidences to temporary storage by using Fortran units 55 and 56.	x	x	x	x	x	x	x	x	x
SETCG	Writes condensed-matrix components, element areas, and incidences to temporary storage by using Fortran units 55 and 56.	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)	(x)

A matrices and the B vector are defined by equations (46)-(50), for Cartesian coordinates, and by equations (224)-(229) for axisymmetric cylindrical coordinates, δ is defined by equation (57), and δ_0 and h_0 are defined in equation (231) and subsequent discussion. Programming details of the linear components of the C and A matrices and of the B vector are described in the following sections.

Transmissivity

Transmissivity terms are computed in subroutine FMCO according to equation (43) in Cooley (1992). These terms represent element contributions to the off-diagonal transmissivity coefficients in the G matrix, which, together with the V matrix, comprise the matrix A in equations (1) and (2) (see development in Cooley (1992) leading to equation (46)). Values for the local x and y transmissivity tensor, $T_{\bar{x}\bar{x}}$ and $T_{\bar{y}\bar{y}}$, are input by hydraulic-property zone as program variables XTR and YTR, respectively. The transmissivity values are first multiplied (scaled) by 0.5, then divided by twice the element area, represented as program variable AREA, before being multiplied by the coordinate functions $\bar{b}_i$, $\bar{b}_j$, $\bar{c}_i$, and $\bar{c}_j$ to create the transmissivity terms. The coordinate functions and the transmissivity terms are computed by using coordinates that have been rotated from the global x-y system to the local $\bar{x}$-$\bar{y}$ system. Transmissivity terms are computed for each node in the element, and are represented by the program variables TFL(i), where i = 1-3.

For axisymmetric (radial) flow, the computations described above are performed by using values of hydraulic-conductivity in the radial (XTR) and vertical (YTR) directions. The hydraulic conductivity terms are multiplied by the centroidal radius of the element in subroutine FMCO to obtain the element contribution to the transmissivity coefficient for each node. The centroidal radius is computed as the arithmetic mean of the r coordinates, which are input and stored in MODFE as the vector XG. The element contributions (TFL terms) are summed according to equation (49) in Cooley (1992) to yield the coefficients, which are stored in the appropriate locations for each node in the program vector A. Because of symmetry in the G matrix and properties of the coordinate functions, the summations involve only transmissivity terms that link a node in the matrix equation to higher-numbered nodes. Details about the coordinate functions and the symmetry of the G matrix are given in Cooley (1992). Because the transmissivity coefficients to the G matrix are stored in the vector A, the element contributions (TFL terms) for each node are not stored within MODFE.

The transmissivity coefficients are used to form parts of the G matrix in finite-element equations (1) and (2) in subroutine FMEQ. The off-diagonal terms that were stored in the program vector A by node in subroutine FMCO are written to other storage locations within the A vector that represent off-diagonal entries of the G matrix. The off-diagonal entries are identified by the storage locations A(NME+J). The transmissivity coefficient for the main diagonal is assembled by summing the negative values of the off diagonal terms according to equation (43) in Cooley (1992) and storing the sum in the program vector AD for each node. After the matrix components are assembled for all equations, the main-diagonal terms are written to the storage locations A(NME) in the A vector.

Note that program vector A is used in subroutine FMEQ to assemble all components of the C and G matrices in the finite-element matrix equations and in subroutines BAND or MICCG for solution of the matrix equations. The manner in which storage locations in the A vector are allocated and reused during completion of other computational steps is described in the section "Allocation of Computer Storage and Processing Time."

Storage Coefficient

Inputs and computations for the storage-coefficient terms are performed in subroutine FMCO. The artesian storage coefficient is input by hydraulic-property zone as the program variable STR. The storage-coefficient terms are computed according to equation (36) in Cooley (1992). Values of STR are multiplied (scaled) by the square of the scaling factor for nodal coordinates (program variable SCALE) and divided by 6 to account for use of scaled values for nodal coordinates and to permit a subsequent multiplication of STR by twice the element area (program variable AREA). Multiplication of STR by AREA yields the storage-coefficient term, c^e_{ii} of equation (36) in Cooley (1992), for node i in element e. These terms represent contributions to the C matrix of equations (1) and (2) for each node in the element and are represented by program variables TESJ, TESK, and TESL. Values of TESJ, TESK, and TESL are identical in simulations that use Cartesian coordinates. For axisymmetric cylindrical coordinates, the element contributions to the C matrix are multiplied by the centroidal radius associated with one-third of the element area that corresponds to each node.

Element contributions to the C matrix coefficients (TESJ, TESK, and TESL terms) are summed by node in subroutine FMCO according to equation (47) in Cooley (1992). The sums represent the main-diagonal entries to the C matrix of equation (1) and are stored for each node in the A vector. The main-diagonal entries for each node in an element are identified in program vector A by the storage locations A(ICA), A(ICB), and A(ICC).

The storage-coefficient terms are assembled into finite-element matrix equation (1) in subroutine FMEQ. The main-diagonal terms are identified during the assembly by the storage locations A(NC) in program vector A. Values in the A(NC) locations for each equation are divided by $(2/3)\Delta t$ and summed into other storage locations in program vector A, A(NME), which represent the main diagonals of the C, A matrices of the matrix equations.

Steady Vertical Leakage

Terms for steady vertical leakage (no storage effects from a confining bed) are computed in subroutine FMCO in the manner expressed by the R^e term of equation (37) in Cooley (1992). The vertical hydraulic conductance of the confining bed (vertical hydraulic conductivity divided by confining-bed thickness), R^e, is input to MODFE in subroutine FMCO by hydraulic property zone as the program variable VLC. This value is multiplied (scaled) by the square of the scaling factor, SCALE, for length terms, and divided by 6 to allow subsequent multiplication by AREA, which has been described previously in the section "Storage Coefficient." The multiplication by AREA creates the steady-vertical leakage term, $(1/3)R^e\Delta^e$, which is used to form part of the diagonal entry of matrix V, v^e_{kk} of equation (37) in Cooley (1992), for each node k in element e. This term is represented by program variable TEL, and is the element contribution from each node to the V matrix of equations (46) and (48) in Cooley (1992).

The element contributions, TEL, are summed by node in subroutine FMCO to create the steady leakage coefficient, V_{ij}, of equation (48) in Cooley (1992). The sum is stored in one storage location within the A vector for each node for assembly into a finite-element matrix equation by subroutine FMEQ. Details about storage of matrix components and assembly of the matrix equation are given in the section "Allocation of Computer Storage and Processing Time." With the V matrix coefficients computed and stored in the A vector, the vertical hydraulic conductances (TEL terms) from each element are not required for further processing; hence, they are not stored in MODFE.

Vertical Leakage of Water Stored Elastically in a Confining Bed

Terms used to simulate vertical leakage of water stored elastically in a confining bed, or transient leakage, are computed in subroutine CBFMCO. The steady vertical leakage component of transient leakage is not computed in this subroutine. Instead, the terms A, α, B, and β, which are used to approximate transient leakage functions $M1_i(\Delta t_d)$ and $M2_i(\Delta t_d)$ of equations (188) and (189), respectively, in Cooley (1992), are assigned values in subroutine CBFMCO. These terms are represented, resepectively, by program vectors AC, ALF, BC, and BTA. Values for the effective vertical hydraulic conductivity and the effective specific storage of the confining bed, expressed by the summations in the numerator and denominator, respectively, of equation (167) in Cooley (1992) are computed as an intermediate step to computing nodal values of γ_i of this equation. The effective vertical hydraulic conductivity at a node in element e, $(1/3)K'^e_{zz}\Delta^e$, is computed as the program variable TEVC. These values are summed by element for each node and are stored in the program vector WVCN. Likewise, the effective specific storage, $(1/3)Ss'^e\Delta^e$, for a node in element e is computed as the program variable TESA, summed by element for each node, and stored temporarily as the program vector GMA. These values are combined with the steady vertical-leakage coefficient, $(1/3)R^e\Delta^e$, discussed in the previous section, to create the γ_i term. This computation is performed only if the value of the effective specific storage (stored in GMA) is greater than "zero," as transient leakage can occur only if the specific storage is nonzero. The zero condition for specific storage is evaluated by MODFE at 10^{-30}, where γ_i is computed if the effective specific storage is larger than this value. Values of γ_i are computed and stored in the vector GMA to be used later in forming the finite-element matrix equations.

Computations for the exponential series M1 and M2 are performed in subroutine CBFMEQ. The series M1 is represented as the program variable SM1. Values of SM1 are used later in this subroutine to form the term Ch of equation (203) in Cooley (1992), which is part of the A matrix of equation (1). The series M2 is not represented by program variables; instead, its computation is contained within computations for other transient-leakage terms, discussed below.

Parts of the transient leakage flux given by the P and Q terms of equations (199)-(202) and the term CH given by equation (204) in Cooley (1992) are computed for each node and are represented by the program vector CBQ in subroutine CBFMEQ. Terms in P that correspond to the series M1 are computed and stored in the program vector QOM1(NT) for each term NT of the series. Likewise, terms in P that correspond to the series M2 are represented by the program vector QOM2(NT). These values represent the transient leakage flux from the previous time step, which is expressed by equations (179) and (181) in Cooley (1992), written for time-step n. Actually, the n+1 time level in these equations becomes the n level as the time step is incremented during simulation. Terms for the series contained in equation (201) in Cooley (1992) are computed and stored by node as the program vector CBTQ. The CBTQ vector also is used to store terms for the expression $QH_{i,n} + CH_{i,n+1}[H_{i,n+1}/\Delta t_{n+1}]$ contained in equation (206) in Cooley (1992) for each node i. Two storage locations in CBTQ for each node are required for these terms and three storage locations are required for the series contained in equation (201) in Cooley (1992); thus, five storage locations are allocated for each node to store terms in the vector CBTQ. All terms just described are combined according to equation (206) in Cooley (1992) and are stored by node in the program vector CBQ. These terms represent components of the B vector of equation (1).

The steady vertical-leakage term associated with the transient leakage flux, CRi, given by equation (204) in Cooley (1992), is not formed by the "CB" subroutines which are added to the program structure of MODFE to simulate transient leakage. Terms that represent steady vertical leakage are computed as described in the previous section and are assembled into finite-element matrix equation (254) in Cooley (1992) for transient leakage.

Assembly of the transient-leakage terms into the finite-element matrix equation occurs in subroutine CBADEQ. The A vector, described in previous sections, is reused for the equation formation. Details about the reuse of computer storage and the equation formation are described in the section "Allocation of Computer Storage and Processing Time." The A-matrix term for transient leakage, CH(L), for node L, is added to the location in the A vector that corresponds to the main diagonal of the A matrix in equation (254) in Cooley (1992). This location is identified in subroutine CBADEQ as A(NME) for node L. The CBQ terms are added to the program vector B at location B(K), which stores the right side of the matrix equation for node L.

After computed heads are obtained for the time step, the part of the leakage flux associated with head changes is computed in subroutine CBTQC. From equation (198) in Cooley (1992), the new component of transient leakage, $\{[h_{i,n+1} - h_{i,n}]/\Delta t_{n+1}\gamma_i\}M1[\gamma_i\Delta t_{n+1}]$ for node i and advanced time step n+1, is added to the old fluxes. The new flux is an approximation of $I_{mi,n+1}$, given by equation (179) in Cooley (1992), and is stored in the first three of five locations allocated to each node in program vector CBTQ. As the time-step index is incremented, these terms actually define $I_{mi,n}$, given in equations (178) in Cooley (1992), which are stored in program vector QOM1 in subroutine CBFMEQ. These terms are used for computing the P terms for the following time step, as described previously.

Areally Distributed Sources and Sinks

The term for areally distributed sources and sinks expressed as $(1/3)W^e\Delta^e$ in equation (50) in Cooley (1992) for element e is computed in subroutine FMCO. This term represents the element contribution to the areally distributed flux at a node, which is part of the B vector of equations (1) and (2). Values for the volumetric flow rate per unit area [length/time], or the unit rate, are input by hydraulic-property zone as the program variable QD. The element contribution to areally distributed recharge or discharge expressed in equation (50) in Cooley (1992) is formed by computations similar to those performed for the storage-coefficient term: the unit rate is multiplied by the square of the scaling factor SCALE, divided by 6, and multiplied by twice the element area (program variable AREA see section "Storage Coefficient"). The resulting term is represented by program variable TEQ, and is added to the B vector of equations (1) and (2) for each node in the element. The B vector is represented by program vector Q. These computations are performed for all elements in the hydraulic-property zone and for all zones, thus satisfying the summation of areally distributed terms over all elements, as indicated by equation (50) in Cooley (1992).

The TEQ terms are summed in subroutine FMCO according to sign-- positive for recharge-- to obtain the terms used in the water-balance summary for areally distributed sources and sinks. It is convenient to perform the summation of TEQ terms at this location in MODFE as these terms are no longer needed after they are added into storage locations in the Q vector. The terms used in the water-balance summary for areally distributed sources and sinks are represented by DQI for recharge and DQO for discharge.

Point Sources and Sinks

Computations for incorporating point sources and sinks into matrix equations (1) and (2) are performed in subroutine DATIN. The volumetric rate [length^3/time] for point stresses are input by node as the program variable QWEL, and are added to the appropriate locations in the program vector Q that stores the B vector of the matrix equations.

The nodal values for point sources and sinks are not stored in MODFE after their incorporation into program vector Q; hence, computations for the water-balance summary pertaining to point sources and sinks are performed in subroutine DATIN. Values of QWEL are summed according to sign, positive for a point source, and stored as program variable WQI for point sources and WQO for point sinks.

Specified-Head Boundaries

Inputs and computations involving specified-head boundaries occur in subroutines DATIN, EXTRAP, and XTRPWT. Values for specified-head boundaries are input by node as the program variable HB in subroutine DATIN. The head difference, δ_i, and average head, h_i, at each specified-head node i are computed in subroutine DATIN according to equations (57) and (63) in Cooley (1992). The average head replaces the initial head at the node, stored in program vector H, and the average head difference is stored in program vector DHB. These values are used to form components of finite-element matrix equation (254) in Cooley (1992) for nodes that connect to the specified-head boundary. Details of the formation of finite-element matrix equations are given in Cooley (1992) and in the section "Reordering Finite-Element Equations for Solution."

The average head, h_i, is replaced by the value of the specified head, and the head change, δ_i, is set to zero following the first time step that uses the specified-head values. These computations occur in subroutine EXTRAP for linear versions of MODFE, and in subroutine WTXTRP for nonlinear versions, although the boundary condition is linear.

Nodes used to simulate specified-head boundaries are identified by an indexing scheme that distinguishes them from other nodes in the finite-element mesh. Because finite-element equations are neither formed nor solved at specified-head nodes (see Cooley (1992), section "Definition of Matrix Equation"), the indexing provides a mechanism for decreasing the order of the finite-element matrix A in equation (254) in Cooley (1992), and for numbering specified-head boundaries and equations to be solved. A description of the indexing of specified-head nodes and nodes where matrix equations are formed is given in the section "Reordering of Finite-Element Equations for Solution."

Specified-Flux Boundaries

The volumetric flow rate per unit length, or unit discharge, q_B, [$length_2$/time] across the specified-flux boundary is input by boundary-condition zone in subroutine DATIN. Two options are given in the input instructions for MODFE (Torak, 1992) that determine the program variables used to represent q_B for input. Option 1: If each element side within the boundary zone has the same unit discharge, then q_B represents a zone value and is input as program variable QBNZ. After inputting the boundary-side number, J, and the node numbers KQB(J) and LQB(J) that define the element side on the boundary, QBNZ is stored in the program vector QBND(J) for side J. Option 2: If each element side within the boundary zone has a unique unit discharge, then unique q_B values for each boundary side are input as the program variable QBND(J), for each side J, along with values for J, KQB(J), and LQB(J). An indicator variable, IZIN, is input for each zone to identify the input option that is used. Details about preparing inputs to boundary-condition zones are given in Torak (1992).

The specified-flux term given by equation (50) in Cooley (1992) for Cartesian coordinates, and equation (229) in Cooley (1992) for axisymmetric cylindrical coordinates, is computed in subroutine FMCO. Values of QBND(J) for side J are assigned to the program variable QB so that QBND(J) also can store the α term associated with the head-dependent (Cauchy-type) boundary (discussed in the following section). The unit discharge, Q_B, is multiplied by one-half the length of the element side on the boundary (program variable DIST) to form the specified-flux term in Cartesian coordinates for both nodes on the boundary. The value of DIST is stored for each node as the program variables TMPA and TMPB. For axisymmetric cylindrical coordinates, TMPA and TMPB are multiplied (weighted) by the average radial distance corresponding to DIST for each node (see equation (229) in Cooley, 1992). The lengths TMPA and TMPB are stored in the program vectors CFDK(J) and CFDL(J) for nodes KQB(J) and LQB(J), respectively, on boundary side J. Values of CFDK(J) and CFDL(J) are multiplied by the α term if a head-dependent (Cauchy-type) boundary also is simulated along side J.

The specified-flux term appearing in equations (50) and (229) in Cooley (1992) is added to the B vector of equations (1) and (2) in subroutine FMCO. These terms are computed as the product of Q_B and either TMPA or TMPB, and are added to the appropriate storage location in program vector Q.

Values of known fluxes, stored in the vector Q, are assembled into finite-element matrix equations (1) and (2) in subroutine FMEQ. Nodal fluxes in the Q vector are added into the program vector B, which stores the right side of the matrix equations. Because the specified flux is a known condition, the equation formulation only involves the right side of these equations.

Head-Dependent (Cauchy-Type) Flux Boundaries

The computational steps of input, coefficient formation, and matrix-equation assembly for head-dependent (Cauchy-type) boundaries occur in the same subroutines as, and nearly concurrently with, the specified-flux terms described in the previous section. Inputs of the α term and boundary head, H_B, defining the head-dependent (Cauchy-type) boundary (Cooley, 1992; Torak, 1992) are made in subroutine DATIN along with inputs for specified-flux conditions. The boundary, or external head, H_B, is represented by the program variables HK(J) and HL(J), respectively, for nodes k and l defining boundary-side J. Node numbers for the boundary side are represented by the program variables KQB(J) and LQB(J) for nodes k and l, respectively, on the boundary-side J. Two options of inputting α are available. Zone values for α are represented by the program variable ALPHZ, and values for each boundary side in the zone are represented by the program variable ALPH(J).

Terms that represent head-dependent (Cauchy-type) boundaries in the V matrix and B vector of equations (1) and (2) are formed in subroutine FMCO. These terms are expressed as the α terms of equations (37), (50), and (229) in Cooley (1992). The length of the boundary side that multiplies the α term in these equations is represented by program variables TMPA and TMPB. These variables contain the same value (equal to one-half the length of the boundary side) for simulations that use Cartesian coordinates, and are weighted for axisymmetric cylindrical coordinates, as described in the previous section. The head-dependent (Cauchy-type) terms given by the above equations are computed and stored by the program vectors CFDK(J) and CFDL(J) for nodes KQB(J) and LQB(J), respectively, on boundary-side J. However, if the α term is zero, indicating that only the specified-flux condition is present at the boundary, then CFDK(J) and CFDL(J), respectively, store the lengths that were computed as TMPA and TMPB. The node number stored in KQB(J) is converted to a negative value as an indicator to exclude this element side from other computational steps that involve head-dependent (Cauchy-type) boundaries.

The terms stored in CFDK(J) and CFDL(J) for head-dependent (Cauchy-type) boundaries are used in subroutine FMEQ to assemble finite-element matrix equation (254) in Cooley (1992). The terms CFDK(J) and CFDL(J) are added to the appropriate locations in program vector A for storing the main diagonal of the A matrix for nodes on the boundary. The right side of the matrix equation for nodes on the boundary contains the product of the head difference, HK(I) – H(K) or HL(I) – H(L), and the head-dependent (Cauchy-type) boundary terms, CFDK(J) or CFDL(J). These products are added to storage locations of program vector B, which stores the right side of the matrix equation during the matrix-equation assembly.

PROGRAMMING DETAILS OF NONLINEAR HYDROLOGIC TERMS

Programming details about the mathematical processes and computations related to input, formation, and assembly of matrix-equation components for terms contained in nonlinear versions of MODFE (tables 2 and 3) are given in the following sections. The nonlinear versions differ from linear versions (table 1) in that each nonlinear version simulates hydrologic processes that require reformulation of matrix-equation components

during simulation. Reformulation is required for conditions where the hydraulic head in the aquifer changes position with reference to a controlling head or altitude associated with the hydrologic process such that different equations are needed to define the process mathematically. An example of a nonlinear condition is leakage from a riverbed when the water level in the aquifer fluctuates above and below the altitude of the bottom of the riverbed sediments during simulation. The changes in aquifer head with regard to the reference altitude (the altitude of the bottom of the riverbed sediments) require different equations for defining the volumetric flow rate between the aquifer and the river. The possible changes to the equation formation of nonlinear hydrologic processes are taken into account by the nonlinear versions of MODFE.

During the following descriptions of programming details for nonlinear hydrologic terms in MODFE, frequent reference is made to the finite-element matrix equations for steady- and nonsteady-state conditions; thus, these equations are restated here. For nonsteady-state conditions, two matrix equations are used that correspond to the predictor and corrector steps. For the predictor step, equation (76) in Cooley (1992) is given as

$$\left(\frac{\underline{\underline{C}}}{(\frac{2}{3})\Delta t_{n+1}} + \underline{\underline{G}}_n + \underline{\underline{V}}\right)\underline{\delta}^* = \underline{\overline{B}} - \left(\underline{\underline{G}}_n + \underline{\underline{V}}\right)\underline{\hat{h}}_n \qquad (3)$$

For the corrector step, equation (80) in Cooley (1992) is given as

$$\left(\frac{\underline{\underline{C}}}{(\frac{2}{3})\Delta t_{n+1}} + \underline{\underline{\tilde{G}}}^* + \underline{\underline{V}}\right)\underline{\delta} = \underline{\overline{B}} - \left(\underline{\underline{\overline{G}}}^* + \underline{\underline{V}}\right)\underline{\hat{h}}_n \qquad (4)$$

For nonlinear, steady-state conditions, finite-element matrix equation (234) in Cooley (1992) is given as

$$\underline{\underline{A}}_\ell \underline{\delta}_\ell = \underline{r}_\ell \qquad (5)$$

where similar terms in equations (1) and (2) have been described in the section "Programming Details of Linear Hydrologic Terms." The new terms in the above equations are derived from the following equations in Cooley (1992): $\underline{\delta}^*$, equation (79); $\underline{\underline{\tilde{G}}}^*$, equation (81); $\underline{\underline{\overline{G}}}^*$ equation (82); $\underline{\delta}_s$, equation (237); and $\underline{r}_s$, equation (238). The head change for the corrector step, $\underline{\delta}$, is described in Cooley (1992) in the development following equation (80).

Water-Table (Unconfined) Conditions

Inputs for water-table conditions are made to MODFE in subroutines FMCO, SWFMCO, and WTFMCO. Hydraulic conductivity in the x and y directions are input in subroutine FMCO by hydraulic-property zone as program variables XTR and YTR, respectively. Nodal values of aquifer thickness and altitude of the top of the aquifer (either land surface or the altitude of the bottom of an overlying confining bed) are input to subroutine WTFMCO for nonsteady-state conditions, and are input to subroutine SWFMCO for steady-state conditions. Aquifer thickness is represented by the program vector THK; the altitude of the top of the aquifer is represented by program vector TOP.

The specific yield of the unconfined aquifer is input to subroutine WTFMCO by hydraulic-property zone, and is represented by the program variable SY. Input of the artesian storage coefficient may be required in addition to specific yield when simulating unconfined conditions that can convert to confined conditions during

simulation. Details about inputs that are required for simulating conversion between confined and unconfined conditions are given in the appropriate sections of Torak (1992). Programming details about forming storage-coefficient (capacitance) and transmissivity terms, and about updating aquifer thickness for unconfined conditions are given in the following sections.

Storage Coefficient (Capacitance)

The storage-coefficient, or capacitance, terms for the C matrix in equations (3) and (4) are computed in subroutine WTFMCO according to equations (36) and (47) in Cooley (1992). Although these equations describe the formulation for confined conditions, the mathematical processes indicated by these equations are applied to unconfined conditions by replacing the artesian storage coefficient, S, with the specific yield, SY. The capacitance term c^e_{ii} of equation (36) in Cooley (1992) represents the contribution from element e to the C matrix for node i. These element contributions are computed as the program variable TESY by multiplying the specific yield (SY) by one-third the element area, which is stored in the program vector AR. Values of TESY are identical for each node in the element, and are summed for all elements connected to node i according to equation (47) in Cooley (1992) by adding their value to the appropriate storage location in program vector ASY. This summation yields the main-diagonal term in the C matrix for capacitance.

The capacitance term is assembled into finite-element matrix equations (3) and (4) in subroutines FMEPWT and FMECWT, respectively. The main-diagonal terms that are stored in program vector ASY are assigned to the storage locations A(NC) of program vector A. Values in the A(NC) locations for each equation are multiplied by $3/[2\Delta t_{n+1}]$, represented by program variable DTM, to effect the division by $(2/3)\Delta t_{n+1}$ indicated in the matrix equations. Program variable TMPA is used to store the product of A(NC) and DTM. Values of TMPA are added to the storage locations in program vector A, A(NME), that represent main diagonals of the C, V, and G matrices. The formulation of the capacitance term for vector-B of the corrector step is performed by multiplying TMPA by the average predicted head change for the time step, δ^*, represented by program variable DH.

Transmissivity

Computations for the transmissivity terms contained in the G matrices of equations (3)-(5) are performed in subroutines FMCO, FMECWT, FMEPWT, SWFMCO, and SWTHK. Element contributions to the hydraulic-conductivity terms given by equation (72) in Cooley (1992) are computed in subroutine FMCO as the program variables TFL(1), TFL(2), and TFL(3). These computations are identical to the computations described previously in the section "Transmissivity," as the same subroutine is used to form transmissivity terms for linear and nonlinear conditions. The element contributions (TFL terms) to each node are summed in subroutine FMCO to form the hydraulic-conductivity terms, Dij, given by equation (75) in Cooley (1992). The Dij terms are represented by storage locations in the program vector A.

The transmissivity terms of the G matrices in equations (3)-(5) are formed in subroutines FMECWT, FMEPWT, SWFMCO, and SWTHK. For nonsteady-state simulations, subroutine FMEPWT is used for the predictor step and subroutine FMECWT is used for the corrector step. For steady-state simulations, the transmissivity terms are computed initially in subroutine SWFMCO and are updated on each iteration in subroutine SWTHK. Details of these computations are given in the following paragraphs.

The formation of transmissivity terms for the predictor step, equation (3), is made in subroutine FMEPWT. Off-diagonal transmissivity terms for the Gn matrix of this equation are computed according to equation (77) in Cooley (1992), and are represented by the program variable TMPA. Aquifer thickness in these computations is represented by program variables THKI and THKL for the two nodes that are linked by the

transmissivity term. The hydraulic-conductivity term, D_{ij}, of equation (77) in Cooley (1992) is represented by the location A(ND+J) of program vector A.

The transmissivity coefficients computed as TMPA are assembled into the G_n matrix of finite-element equation (3) in subroutine FMEPWT. The off-diagonal terms (values of TMPA) are assigned to the storage locations A(NME+J) of program vector A. The main diagonal of the G_n matrix is formed according to equation (77) in Cooley (1992) by adding the negative values of TMPA to the appropriate storage location in program vector AD. After matrix components to all equations are formed, the main diagonal is stored in the A vector, which is identified in this subroutine by the location A(NME).

Formation of transmissivity terms for corrector-step equation (5) is made in subroutine FMECWT. Off-diagonal terms for $\tilde{G}^*$ and $\bar{G}^*$ in this equation are computed, respectively, according to equations (81) and (82) in Cooley (1992). In these computations, aquifer thickness at the beginning of the time step and average thickness changes from the predictor step are used to multiply the hydraulic-conductivity terms, D_{ij}, to obtain the off-diagonal transmissivity terms. Aquifer thickness is represented by program variables THKI and THKL, and thickness changes from the predictor step, δ^* of equation (81) in Cooley (1992), are represented by program variables DTK(I) and DTK(L) for the two nodes that are linked by the transmissivity term. The hydraulic-conductivity term, Dij in equation (75) in Cooley (1992), is represented by the location A(ND+J) in program vector A. The off-diagonal term, $\tilde{G}^*_{ij}$, is computed as program variable TMPA.

Computations for the off-diagonal term, $\bar{G}^*_{ij}$, are combined with computations contained in-B that involve $\tilde{G}^*_{ij}$, to yield the term for the right side of the corrector equation (see development leading to equation (80) in Cooley, 1992). Terms for the right side of the corrector equation are stored in program vector B. Computations of the right-side terms are identified by program statements that contain the multiplier, (1/16), which is represented by program variable C3.

The main-diagonal terms for $\tilde{G}^*_{ij}$ are computed according to equation (43) in Cooley (1992) by summing the negative values of the off-diagonal terms into storage location in the program vector AD. These computations are identified by statements that contain the program vector AD and either TMPA or an explicit formulation of the off-diagonal term involving aquifer thickness (THKI and THKL) and predicted thickness changes (DTK(I) and DTK(L)). After these computations have been performed for all matrix equations, the main-diagonal terms are stored in the locations A(NME) of program vector A.

For steady-state conditions, the transmissivity terms are computed in subroutines SWFMCO and SWTHK. Initial values for the off-diagonal transmissivity terms, given by equation (74) in Cooley (1992), are computed in subroutine SWFMCO after inputs of aquifer thickness (THK) and altitude of the aquifer top (TOP) are made. The hydraulic-conductivity terms, D_{ij}, stored in program vector A are multiplied by the average aquifer thickness along an element side to yield the off-diagonal transmissivity term. Aquifer thickness for both nodes along the element side are represented by storage locations THK(I) and THK(K), and the hydraulic-conductivity and transmissivity terms are represented by the storage location A(ND+J).

Transmissivity terms are computed during iteration in subroutine SWTHK. Updated values of aquifer thickness are used to multiply the hydraulic-conductivity term, A(ND+J), to obtain the off-diagonal transmissivity term given in equation (244) in Cooley (1992). Aquifer thickness for iteration level λ is represented by program variables THKI and THKK for the two nodes along an element side corresponding to the transmissivity term. The off-diagonal transmissivity term is stored in location A(ND+J) of program vector A.

Transmissivity terms are assembled into the finite-element matrix equation (5) in subroutine FMEQ. Programming details for adding transmissivity terms to the A vector to form the equations for nonlinear steady-state conditions are identical to those presented for the linear case. (See section "Transmissivity" under "Programming Details of Linear Hydrologic Terms.")

Updates to Aquifer Thickness

Aquifer thickness is updated at the end of the current time step for nonsteady-state conditions in subroutine XTRPWT, and at end of the current iteration level for steady-state conditions in subroutine SWTHK. Note that for nonsteady-state conditions, aquifer thicknesses stored in program vector THK are not updated to provide estimates of transmissivity terms for the corrector step (discussed in the previous section). Instead, predicted thickness changes, stored in program vector DTK, are used to augment aquifer thickness, stored in THK, when computing transmissivity terms for the corrector step.

Updates to aquifer-thickness values stored in program vector THK are made in subroutine XTRPWT according to equation (85) in Cooley (1992). These updates are made at each node following extrapolation of hydraulic head from average values for the time step to values at the end of the time step. Aquifer heads at the beginning and end of the time step are represented by program variables HO and the vector H, respectively. These values are compared with the altitude of the top of the aquifer, stored in program variable TOP, to determine if the total head change for the time step can be used to update aquifer thickness. The total head change is computed from the average head change, stored in program variable DHC, as 1.5 $\times$ DHC. If conversion between confined and unconfined conditions occurred at a node, then the thickness change is limited to the head change that occurred within the unsaturated part of the aquifer (fig. 13).

Aquifer thickness is updated during each water-table iteration in subroutine SWTHK by computations that are similar to those described previously for subroutine XTRPWT (fig. 13). The initial aquifer head, HO, and the head after the water-table iteration, H(I), are compared with TOP(I) to determine if the total head change during iteration can be used to update aquifer thickness. The head change is represented by program vector B.

Conversion Between Confined- and Unconfined-Aquifer Conditions

Computations for conversion between confined and unconfined aquifer conditions are performed automatically in all nonlinear versions of MODFE. In nonsteady-state versions (table 2), subroutines FMECWT and FMEQWT perform the appropriate comparisons of aquifer head at the beginning and end of the time step with the altitude of the top of the aquifer (or base of an overlying confining bed) to determine if conversion has occurred at a node. In steady-state versions (table 3), these comparisons occur in subroutine SWTHK by using the computed head for the current iteration.

Aquifer head, H(I), is compared to the altitude of the top of the aquifer, TOP(I), for each node I in predictor equation (3) in subroutine FMEPWT. Based on these comparisons, either the aquifer storage coefficient or specific yield is used to form the capacitance term for the C matrix in equation (3). The storage coefficient is represented by the locations A(NC) in the A vector, and the specific yield is represented by the vector ASY. After comparing H(I) with TOP(I), the appropriate storage term is assigned to the A(NC) locations, which are used to form the matrix equations.

For corrector equation (4), aquifer heads at the beginning and end of the time step are compared with TOP in subroutine FMECWT to determine if conversion between confined and unconfined aquifer conditions has occurred. The predicted aquifer head at the end of the time step is represented by program variable HP, and head at the beginning of the time step is represented by program variable HO. After comparing HP and HO with TOP(I), a predicted change in aquifer thickness, DTK(I), is computed for node I. Values for DTK(I) are assigned as either the total predicted head change for the time step, computed as program variable DHP, or the part of the predicted head change that occurred below TOP(I) during the conversion (fig. 14). Predicted thickness changes are multiplied by 2/3 to obtain average thickness changes for the time step. These values and

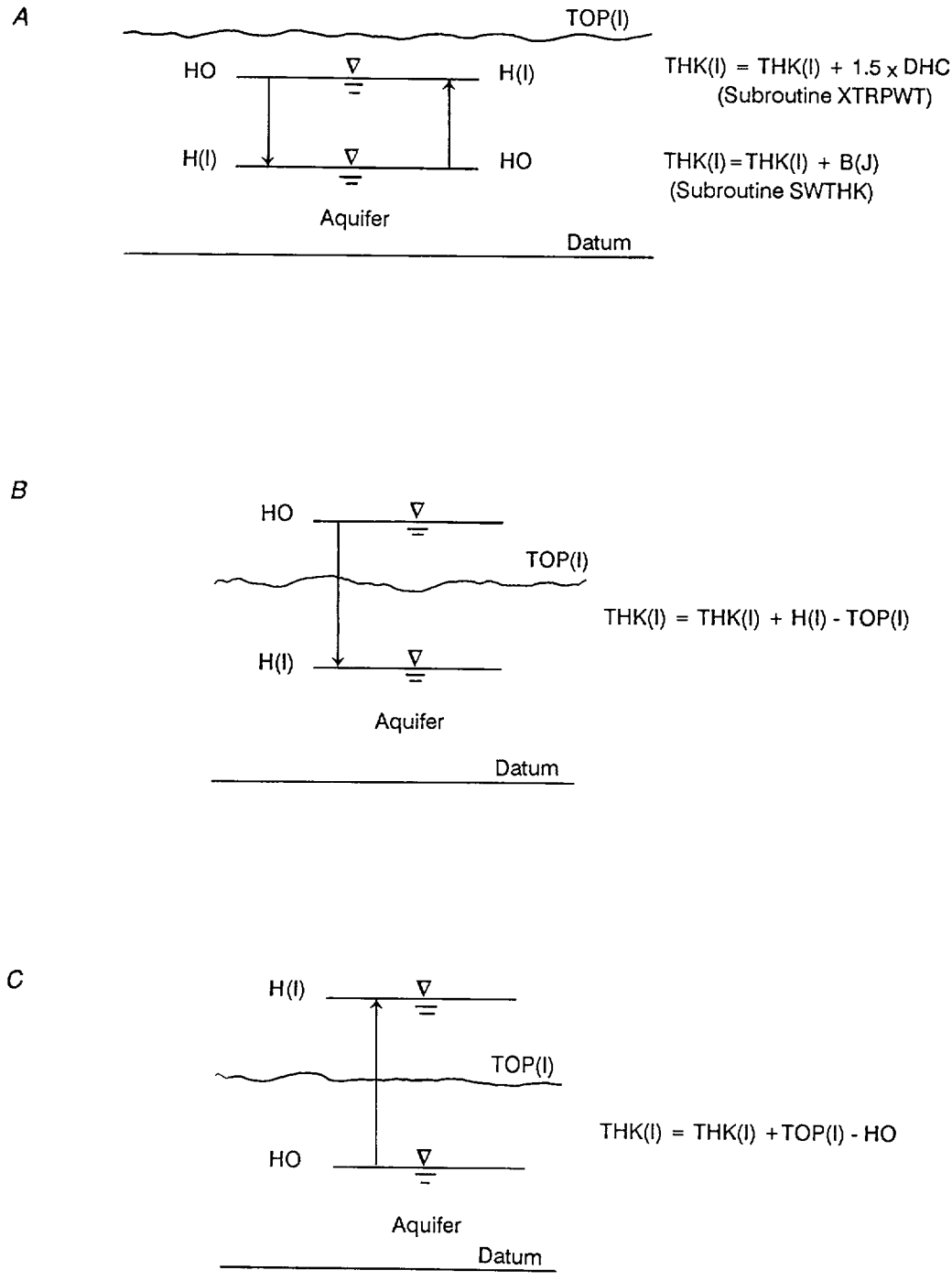


Figure 13.-- Positions of aquifer head, HO, at beginning of time step, and H(I), at end of time step, and computations in MODular Finite-Element model (MODFE) for updating aquifer thicknesses, THK(I), at node I for (*A*) water-table (unconfined) conditions without conversion, (*B*) conversion from confined to unconfined conditions, and (*C*) conversion from unconfined to confined conditions.

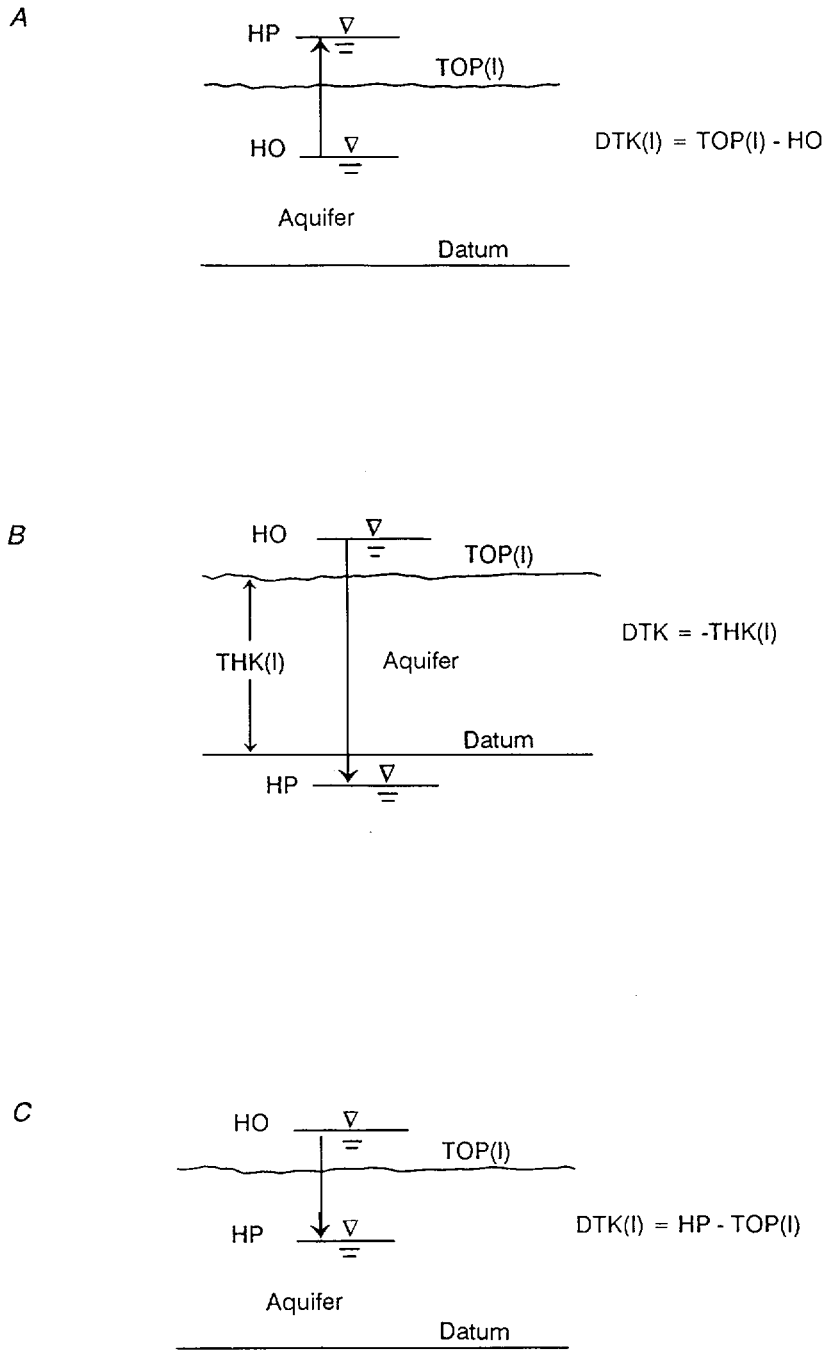


Figure 14.-- Positions of aquifer heads, HO and HP, and computation of predicted thickness change, DTK(I), at node I for conversion from (*A*) unconfined to confined conditions; and (*B*) and (*C*) confined to unconfined conditions.

the thickness values from the beginning of the time step, program vector THK, are used to compute transmissivity coefficients for the corrector step.

For confined to unconfined conversions, the predicted head, HP, is revised in subroutine FMECWT according to equation (100) in Cooley (1992) to account for inaccuracies that are associated with computing HP on the predictor step by using only the artesian storage coefficient (see development leading to equation (100) in Cooley 1992). The revised predicted head, HP, is used to recompute the predicted thickness change, DTK(I), predicted head change, DHP, and the average head change, which is represented by program vector DH.

Changes to the corrector equation (4) caused by conversion between confined and unconfined aquifer conditions are made in subroutine FMECWT. The capacitance term in the C matrix of equation (4), stored in location A(NC) in program vector A, is assigned the value of specific yield, stored in program vector ASY, for the node that converts to unconfined conditions. The right side of equation (4) is modified to contain the capacitance term shown in equation (98) in Cooley (1992). These computations involve program vector B, which stores the right side of equation (4), and occur as program statements B(J) = B(J) + (ASY(I) − A(NC)) × (TOP(I) − HO)/DT for confined to unconfined conversions, and B(J) = B(J) + (A(NC) − ASY(I)) × DTK(I)/DT for unconfined to confined conversions, where DT is the time-step size.

Hydrologic conditions that cause unpredicted conversions or nonconversions are evaluated in subroutine WTCCHK. Predicted heads at the end of the time step are computed as program variable HP by using average head changes from the predictor step, stored as program vector DH. After comparing HP and heads at the beginning of the time step with the altitude TOP, HP is recomputed by using the head changes from the corrector step, which are stored in program vector B. If new values of HP indicate conversion, then either an unpredicted conversion or nonconversion had occurred during the corrector step, and the indicator variable ISC is set to 1. The indicator is evaluated by the main programs of the nonlinear (nonsteady-state) versions of MODFE (table 2); for ISC=1, the corrector step is repeated using thickness and transmissivity updates from the previous corrector as predicted values.

Computations for conversion during steady-state simulations are performed in subroutine SWTHK and involve computing transmissivity terms with updated thickness values from the current water-table iteration. At nodes that experience conversion, the thickness change that is added to program vector THK is that part of the head change that occurs below the altitude of the top of the aquifer or base of the overlying confining bed (fig. 13A,B). Programming details of updating aquifer thickness in this subroutine are given in the previous section.

Aquifer Drying and Resaturation

Computations for aquifer drying and resaturation are incorporated into subroutines that form coefficients and that form and solve finite-element matrix equations. A node is considered dry when the thickness, stored in program vector THK, is computed to be zero or negative. Resaturation occurs when the thickness at a dry node is computed to be greater than zero. Checks for zero or negative thickness values are placed in subroutines FMECWT and FMEPWT for nonsteady-state conditions, equations (3) and (4), and in subroutine SWTHK for steady-state conditions, equation (5). Aquifer thickness is updated after each time step and iteration level in subroutines XTRPWT and SWTHK, respectively (see discussion in section "Updates to Aquifer Thickness"). Hydrologic implications of a negative aquifer thickness at a node are explained in the section "Drying and Resaturation of Aquifer Material" in Torak (1992).

Computations for transmissivity coefficients use temporary-valued variables for aquifer thickness and for changes in thickness to account for aquifer drying at a node. Values for aquifer thickness, stored in program vector THK, are assigned to program variables THKI and THKL for the two nodes where a transmissivity coefficient is computed. If either or both nodes is dry (that is, if the corresponding value in THK is less than

or equal to zero), then the corresponding variables, THKI, THKL, or both, are set to zero. For corrector-step computations in subroutine FMECWT, different values for the predicted thickness change, stored in program vector DTK, are assigned to nodes predicted to go dry depending on whether the node was saturated or dry at the beginning of the time step. Dry nodes that are predicted to stay dry have zero values assigned to the appropriate location in vector DTK, while nondry nodes that are predicted to go dry have values of -THK assigned to vector DTK. The computations for transmissivity coefficients proceeds as described in the section "Transmissivity" for nonlinear hydrologic terms by using values assigned to program variables THKI and THKL, and to program vector DTK.

Negative fluxes (indicating discharge) that are used to form the right side of equation (4) for the corrector step and (5) for steady-state conditions are decreased by half of their current value at nodes that are predicted to go dry. For the corrector step, a zero or negative value for the predicted aquifer thickness at a node (program variable THKP) is used to determine the drying condition for changing the negative fluxes. Values of THKP are computed as the sum of aquifer thickness, THK(I), and predicted aquifer thickness, DTK(I), for each node I. For steady-state conditions, zero or negative values for the updated aquifer thickness, stored in program vector THK, are used to determine dry nodes and subsequent decreases to the known fluxes. Fluxes for the right side of equations (4) and (5) are stored in location Q(I) in program vector Q for node I.

Decreases in negative fluxes at dry nodes are accounted for in the water-balance summary by decreasing the term WQO used to represent point sinks. The decrease to WQO is completely arbitrary, as values for individual known fluxes are not stored in MODFE, and the net negative flux at a dry node may not be attributed to point sinks. Therefore, a message is printed out from subroutines FMECWT and SWTHK giving the value of the decrease in net negative flux at the dry node so the user can adjust the appropriate terms in the water-balance summary.

Head-Dependent (Cauchy-Type) Flux

Inputs for nonlinear head-dependent (Cauchy-type) boundaries are made in subroutines GNINIT and GNFMCO. Values for the number of element sides and the number of boundary-condition zones that represent nonlinear head-dependent (Cauchy-type) boundaries are input in subroutine GNINIT as program variables NBNC and NLCZ, respectively. These values are used to allocate computer storage for the boundary condition. Inputs for defining boundary-condition zones and for selecting an option for entering values for the α_r term of equation (153) in Cooley (1992) are made in subroutine GNFMCO. These inputs consist of the zone number, the number of sides in the zone, and an indicator variable for the input of α_r, which are represented, respectively, by program variables KZ, NOS, and IZIN. Formats and descriptions of these inputs are given in the section "Input Instructions" of Torak (1992). Other inputs for nonlinear head-dependent (Cauchy-type) boundaries that are made in subroutine GNFMCO are the number of the element side on the boundary (program variable J), node numbers defining the element side (program variables KR(J) and LR(J)), boundary or external heads (program variables HRK(J) and HRL(J)), controlling heads or altitudes (program variables ZRK(J) and ZRL(J)), and the α_r term that controls the flux. The α_r term is input either as program variable GCZ for the boundary-condition zone, or as GC(J) for each side J.

Formulation of the coefficient Cr of equation (155) in Cooley (1992), for nonlinear, head-dependent (Cauchy-type) boundaries occurs in subroutine GNFMCO. The α_r term, stored in program vector GC, is multiplied by half the length of the element side on the boundary to complete the formulation. Program variable DIST is used to compute and store the length term used in these computations.

Components of matrix equations (3)-(5) for nonlinear, head-dependent (Cauchy-type) boundaries are assembled in subroutines GNPRED and GNCORR according to the formulations (with the appropriate substitutions) given by equations (124)-(127) in Cooley (1992) for areal, head-dependent leakage (see

development following equation (155) in Cooley (1992)). Each formulation of the nonlinear boundary condition is identified in these subroutines by a COMMENT statement containing a case number, 1-4, which corresponds to equations (124)-(127), respectively. Comparisons are made between aquifer head and controlling head or altitude to identify the formulation, or case, that applies to a particular boundary side. For the predictor step and for steady-state conditions, aquifer head is represented by program vector H and controlling head or altitude is represented by program vector ZR. The boundary or external head is represented by program vector HR. Cases 1 and 4 are evaluated in subroutine GNPRED for forming coefficients for the predictor step and for steady-state conditions; all four cases are evaluated in subroutine GNCORR for the corrector step.

Computations for nonlinear, head-dependent (Cauchy-type) boundaries are simplified in subroutine GNCORR for forming coefficients for the corrector step by using the average aquifer head and average boundary or external head, and by representing heads, altitudes, and head differences by nonsubscripted variables. The average aquifer head, $\bar{h}$, and average boundary or external head, $\bar{h}_r$, are obtained by applying equation (63) in Cooley (1992) to h and hr, and are represented by program vectors H and HR, respectively. Aquifer head at the beginning and end of the time step is represented by program variables HO and HP, respectively. For each node I on the boundary, the controlling head or altitude is represented by program variable ZRI, and the boundary or external head is represented by program variable HRI. Head or altitude differences HRI – ZRI and HO – ZRI are represented by program variables DRZ and DHZ, respectively. The values for DRZ and DHZ, and the predicted head change for the time step, program variable DHP, are used to compute the ρ terms, given by equations (111) and (115) in Cooley (1992), which are used to proportion equations for the nonlinear boundary condition over the time step for cases in which two expressions for the boundary flux apply during the time step. (See development in Cooley (1992) leading to equations for the ϕ terms.)

Terms for nonlinear head-dependent (Cauchy-type) boundaries are added into matrix equations (3)-(5) in subroutine GNPRED, for the predictor step and for steady-state conditions, and in subroutine GNCORR, for the corrector step. For each case, terms for the G matrix and the right side of matrix equations (3)-(5) are added to the appropriate locations in program vectors A and B. The main diagonal for the V matrix is represented by location A(N) in the A vector and the right side is represented by the location B(LE) in the B vector, where indexes N and LE correspond to the equation number that is being formed. Depending on the case, the Cr coefficient, represented by program variable GC(J), is added to the main diagonal of the V matrix, and the product of Cr and a head or altitude difference is added to the right side of the matrix equations. Also, the main diagonal and right-side components of the matrix equations is multiplied by the ϕ terms for cases where two expressions are needed to represent the boundary flux are during the time step (cases 2 and 3). The head or altitude difference is computed for the appropriate cases and stored as program variable TMPA, and the ϕ terms are represented by program variables PHI and PHC. The product of TMPA and GC(J) is added to the storage location B(LE) to complete the formulation of terms for the right side of the matrix equations.

Point Sinks

The computational steps of providing input, forming coefficients, and forming equations for nonlinear point sinks are performed in subroutines GNFMCO, GNPRED, and GNCORR in a manner similar to that described in the previous section for nonlinear, head-dependent (Cauchy-type) boundaries. The number of nonlinear point-sink boundaries is input to subroutine GNINIT as program variable NPNB, and is used to allocate storage locations for program vectors associated with this boundary condition. Inputs that define nodes as nonlinear point-sink boundaries are made in subroutine GNFMCO and consist of the number of the boundary, I, the node number on the boundary, KP(I), coefficient Cp_i of equation (106) in Cooley (1992), and controlling head or altitude, ZP(I). The coefficient Cp_i is input as program variable GCP and is stored in program vector GC by using storage locations IP that follow the locations assigned to similar terms for nonlinear, head-dependent (Cauchy-type) boundaries.

Because of the manner in which point-sink boundaries are defined, coefficients for matrix equations (3)-(5) are formulated completely by the values of Cp that were input in subroutine GNFMCO and are stored in program vector GC. Hence, additional computations to form coefficients for nonlinear point sinks are unnecessary.

Components of matrix equations (3)-(5) for nonlinear point sinks are assembled in subroutine GNPRED for the predictor step and for steady-state conditions, and in subroutine GNCORR for the corrector step. Aquifer head is compared with controlling head or altitude to determine the appropriate formulation of the boundary condition for the matrix equations. Four different cases are possible for representing nonlinear point-sink boundaries in the matrix equations. Three of the four cases are expressed by equations (108)-(110) in Cooley (1992); the fourth case requires no formulation (see development in Cooley (1992) leading to these equations). Depending on the case that is formulated, terms for nonlinear point-sink boundaries are added to the main diagonal of the V matrix and to the right side of the matrix equations. The main diagonal for the V matrix is represented by location A(N) in the A vector and the right side is represented by the location B(KE) in the B vector, where indexes N and KE correspond to the equation that is being formed.

Matrix-equation components of nonlinear point sinks for case 1 are formulated in subroutine GNPRED for the predictor step and for steady-state conditions. (Cases 2 and 3 are not applicable to the predictor step or to steady-state conditions and case 4 requires no formulation.) The Cp coefficient, represented by program vector GC, is added to the main diagonal of the G matrix, and the product of Cp and the head or altitude difference, ZP(I) – H(K), is added to the right side of the matrix equations, where the indexes define node K on boundary side I.

Matrix-equation components to nonlinear point sinks for cases 1-3 are formulated in subroutine GNCORR for the corrector step. Computations are simplified from those given by equations (108)-(110) in Cooley (1992) by using the average aquifer head for case 1 and by using unsubscripted variables to represent aquifer head and head or altitude differences for cases 2 and 3. The average aquifer head is represented by program vector H, and heads at the beginning and end of the time step are represented, respectively, by program variables HO and HP. The controlling head or altitude, ZP, is represented by program variable ZPI, and the difference, ZPI - HO, is represented by program variable DHZ. These values are used to compute the ϕ terms according to equations (111) and (115), which are used in formulating cases 2 and 3 to account for changes in the boundary-flux equations during the time step. The ϕ terms are represented as program variables PHI and PHC. They multiply the main-diagonal term, GP(IP), for case 3 and the right-side terms for cases 2 and 3. Head or altitude differences are computed for the appropriate cases and stored as the program variable TMPA. The product of TMPA and GC(IP) is added to the storage location B(KE) to complete the formulation of terms for the right side of the matrix equations for the three cases.

Steady Vertical Leakage

Input of hydrologic information, coefficient formation, and matrix-equation assembly for nonlinear steady vertical leakage occur in subroutines VNINIT, VNFMCO, VNCORR, and VNPRED. The number of hydraulic-property zones for nonlinear steady vertical leakage is input to subroutine VNINIT as program variable NVNZ. This value is used in the other "VN" subroutines as a counter to limit computations and inputs that pertain to this boundary condition. Values for the steady vertical leakage terms R_a and R_e of equations (117) and (129), respectively, in Cooley (1992) are input by hydraulic-property zone in subroutine VNFMCO. Program variable VNCF is used to represent the "R" terms. Other inputs defining the zone consist of the zone number, the beginning element number for the zone, and the number of elements in the zone. These values are represented, respectively, by program variables L, NBE, and NO. The controlling head or altitude, given by H_a and z_e in the equations referenced above is input by node as program vector HS. Details about assigning values to these variables and establishing hydraulic-property zones are given in Torak (1992).

Computations of coefficients for steady vertical leakage that add into the matrix equations are performed in subroutine VNFMCO. The coefficients C_a and C_e, given by equations (119) and (132), respectively, in Cooley (1992), are computed for each element in the hydraulic-property zone. These terms represent the contribution to the V matrix of equations (3) and (4) for each node where an equation is formed. The formulation of these terms is similar to the formulation described previously for linear steady vertical leakage. Values for each element that contribute to these coefficients are computed as the product of the R term (VNCF) and one-third of the element area (stored in program vector AR). The element contributions are computed as program variable EC and are summed by program vector E over all elements that are connected to a node. The summation is performed efficiently by computing the indexes NA, NB, and NC for the three nodes in an element and by adding EC to storage locations E(NA), E(NB), and E(NC). These computations are repeated for all elements in the hydraulic-property zone and for all zones to complete the formulation of coefficients.

Matrix-equation components for nonlinear steady vertical leakage are added to the appropriate locations in equations (3)-(5) in subroutines VNPRED for the predictor step and for steady-state conditions, and in subroutine VNCORR for the corrector step. The main diagonal of the G matrix in these equations is represented by the location A(NME) of program vector A, and the right side terms are represented by the location B(K) of program vector B. The indexes NME and K correspond to the equation that is being formed for a node. The main-diagonal and right-side components of nonlinear steady vertical leakage are added to these locations based on evaluation of the leakage case, or formulation, that applies at the node.

The type of leakage that is simulated and subsequent equation formulation is determined by evaluating the sign of coefficient values stored in program vector E; negative values for discharge only, and positive values for recharge and discharge (see appropriate sections of Torak (1992) for details). Computations are bypassed at a node for zero values stored in the corresponding location of program vector E. For discharge-only leakage, nine different formulations, or cases, are possible by comparing aquifer head with land-surface and controlling altitudes. These cases are given by eight equations, (135)-(140) and (142)-(143) in Cooley (1992); case 7 requires no formulation. For the recharge and discharge type of leakage, four cases similar to those developed for the point head-dependent sink functions arise by comparing aquifer head with the altitude of the top of the aquifer or of the base of the overlying confining layer. However, only cases 1 and 4 of either type of leakage are formulated in subroutine VNPRED; the remaining cases do not apply to equations for the predictor step or for steady-state conditions, and case 7 of the discharge-only type requires no formulation.

Matrix-equation components for nonlinear steady vertical leakage are assembled in subroutine VNPRED by assigning the coefficient stored in program vector E to program variable EI. Values of EI are made positive for cases involving the discharge-only function. Based on comparisons of aquifer head at the beginning of the time step with the controlling head or altitude, HS, and with the altitude of the top of the aquifer (or base of the overlying confining bed), TOP, a head or altitude difference is computed, which multiplies EI to become the right-side term of the matrix equations. The head or altitude difference is represented by program variable TMPA. The leakage coefficient, EI, is added to the location A(NME) to form the main-diagonal component of the V matrix in equations (3) and (4).

Computations that assemble matrix-equation components in subroutine VNCORR are simplified by representing aquifer head, controlling altitudes, and head and altitude differences as unsubscripted variables. Formulations for some of the leakage cases are further simplified by computing average values over the time step for aquifer and controlling heads according to equation (63) in Cooley (1992), and by combining computations that involve the ρ terms and the right side of the matrix equations. Aquifer heads at the beginning and end of the time step are represented, respectively, by program variables HO and HP. Values for HP are computed by using the predicted head change, computed as program variable DHP. The controlling altitudes or heads that are stored in program vector HS are represented as program variables ZE for the discharge-only leakage, and as HA for recharge and discharge. The altitude of the top of the aquifer is represented as program variable ZT. Head or elevation differences are represented by the following program variables: DEH, for ZE−HO; DHT,

for HO – ZT; DET, for ZE – ZT; and DAT, for HA – ZT. Frequent computations involving the ϕ terms are represented by the following program variables: PHC, PHCF, PHE, PHI, and PHT. The head or altitude differences that multiply the leakage coefficient EI are represented by program variable TMPA. These values are modified by the ϕ terms for some of the leakage cases. The product of TMPA and EI comprises the component for the right side of matrix equation (4) for nonlinear steady vertical leakage. Values of EI are similarly modified by the ϕ terms and are added to the location A(NME) to form the main-diagonal component of the G matrix in the corrector-step equation.

COMPUTATIONAL ASPECTS OF SIMULATION FEATURES

Computational aspects of features to MODFE that enhance its ability to simulate ground-water flow are provided in this section. Details are given about specific computations, program structures, and versions that pertain to simulating linear and nonlinear steady-state conditions, changing stresses and boundary conditions with time, and computing a water-balance summary to assist the user in applying MODFE to a particular aquifer problem.

Steady-State Flow

Steady-state conditions in an aquifer are simulated by either linear or nonlinear versions of MODFE. Linear versions solve equation (2) for the head change, δ, that occurs as a result of stresses and boundary conditions to the simulated region. The head changes are computed without providing updates to hydrologic terms in the A matrix or B vector. Nonlinear versions solve equation (5) for the head change or displacement, δ , that occurs during iteration. The displacements occur not only as a result of stresses and boundary conditions that are acting upon the simulated region, but because of changes (nonlinearities) in hydrologic terms contained in the A matrix and r vector, which require updating after each iteration. Linear conditions of steady-state flow are simulated by using the same versions of MODFE that are used to simulate linear, nonsteady-state conditions. However, nonlinear steady-state flow is simulated by versions of MODFE that are structured specifically for simulating nonlinear, steady-state conditions (see section "Simulation Capabilities and Versions of MODFE"). Computational aspects of versions of MODFE that simulate linear and nonlinear steady-state conditions are described in the following sections.

Linear Conditions

Linear steady-state conditions are simulated by versions of MODFE shown in the structure diagrams as LMFE1-LMFE4 (figs. 1 and 2). Formulation of matrix equation (2) is performed by inputting a value of 1 for the indicator variable ISTD. The indicator variable is evaluated in subroutine DATIN prior to computing known head changes at specified-head boundaries. For steady-state conditions, known head changes represent the total change in head from the initial conditions. For nonsteady-state conditions, the head change represents the average change in head for the time step, where the total head change is multiplied by 2/3. Known head changes are represented by program vector DHB.

Although the steady-state solution is not dependent on initial conditions, these conditions are incorporated into the matrix equations so that the equations solve for head changes, δ, rather than head. This decreases the magnitude of the unknowns in the matrix equations, thus decreasing roundoff error in the solution (see section "Evaluation of Time Integral" in Cooley, 1992).

The value of the indicator variable, ISTD, is checked in the main programs of MODFE to bypass computations that extrapolate head changes for nonsteady-state conditions. Extrapolation is necessary for

nonsteady-state conditions as the solution vector, δ, in equation (1) represents average head changes during the time step. For steady-state conditions, the solution vector, δ, represents total head change from the initial conditions; therefore, extrapolation is not required.

A time-step size of unity (=1) is required as input to MODFE for simulating linear steady-state conditions. Although the steady-state solution is not time dependent, the time-step size is used in computations that assemble matrix equations and compute a water-balance summary. A value of 1 for the time-step size eliminates division by zero and permits values that appear in the water-balance summary to represent rates and volumes for one time unit. Instructions for inputting the time-step size are given in the section "Input Instructions" in Torak (1992).

Nonlinear Conditions

Nonlinear steady-state conditions are simulated by the versions of MODFE shown in figures 9-12. These versions provide solutions to matrix equation (5) for the displacement, δ_ℓ, or head change for iteration level ℓ. Values for the displacement vector, δ, are computed alternately with updating nonlinear terms in the matrix equation by an iteration process. Because all nonlinearities represented in MODFE involve either water-table (unconfined) conditions or the potential for water-table conditions to occur during simulation, the iteration process is termed water-table iteration in the following discussions. Computations for water-table iterations necessitated that steady-state versions be constructed separately from nonsteady-state versions of MODFE.

Inputs for controlling water-table iterations are made in subroutine SWINIT, and consist of the maximum number of water-table iterations, the maximum allowable displacement, and the closure tolerance for achieving steady state. The maximum number of water-table iterations is represented as program variable NITSW, and the maximum allowable displacement is represented as program variable DSMX. The closure tolerance for achieving steady state is represented as program variable TOLSW. Values for DSMX and TOLSW are selected so that an adequate steady-state solution is obtained within a minimum number of iterations. A discussion of values that are normally input for these variables is given in the sections "Nonlinear Case" in Cooley (1992) and "Nonlinear Conditions" in Torak (1992).

The indicator variable, ISTD, described in the previous section, is used to indicate that the total known head change at specified head nodes is to be computed for nonlinear steady-state conditions. The indicator variable is input in subroutine DATIN, and computations for the total head change also are performed in this subroutine. Head changes at specified-head boundaries are represented as program vector DHB.

The displacements, δ_ℓ, in equation (5) are decreased, or damped, automatically during each water-table iteration in subroutine SWBDMP. The value of the maximum allowable displacement, DSMX, determines the amount of damping that is applied to the displacements. It is used in computing the damping parameter, ρ_ℓ, which multiplies the displacements. The damping parameter is represented as program variable RP and is computed according to the three-step process given by equations (241)-(243) in Cooley (1992). The displacements are represented as program vector B. Other terms that are involved in computing RP are the maximum displacement and its absolute value for the current iteration level, ℓ, and the ratio of damped-to-undamped displacements. These values are represented, respectively, as program variables DSP, DSPA, and SPR.

Hydraulic head for the advanced iteration level, $\ell+1$, is computed in subroutine HCALC according to equation (239) in Cooley (1992) by using the damped displacements. The updated heads are stored in program vector H and are used in subsequent computations for updating aquifer thickness or for determining the formulation of nonlinear boundary conditions (see previous sections for details about these computations).

Changing Stresses and Boundary Conditions With Time

Inputs and computations for changing stresses and boundary conditions at the beginning of a time step or stress period are made in subroutines CBCHG, COCHG, GNCHG, and VNCHG. Changes are made by evaluating indicator variables that correspond to each stress or boundary condition that is allowed to vary in MODFE. If the value of an indicator variable equals the current time-step number, then changes are made to the corresponding stress or boundary condition. Subroutines and indicator variables that correspond to changing stresses and boundary conditions with time are listed in table 16.

Table 16.-- *Program variables and subroutines used to change stresses and boundary conditions with time*

Stress or boundary condition	Number of time steps to implement change	Indicator variable	Old value	New value	Subroutine
Source-bed head for transient leakage	1	NCBCH	--	HRJ	CBCHG
Point sources/sinks	2	NWCH	QOLD	QNEW	COCHG
Areally distributed recharge/discharge	2	NQCH	QOLD	QNEW	COCHG
Source-bed head for steady leakage	2	NHRCH	--	HR(J)	COCHG
Specified-flux boundary	2	NBQCH	--	QNEW	COCHG
Head-dependent (Cauchy-type) boundary	2	NBQCH	-- --	HK(J) HL(J)	COCHG
Specified-head boundary	1	NHCH	--	HB	COCHG
Nonlinear head-dependent (Cauchy-type) boundary	2	NGNCH	-- --	HRK(J) HRL(J)	GNCHG
Controlling head for nonlinear steady leakage	2	NVNCH	--	HS(J)	VNCHG

Changes to most stresses and boundary conditions that are listed in table 16 take two time steps to implement in MODFE because of the manner in which known terms in the B vector of matrix equations (1), (4), and (5) are represented. To allow for a time-varying B vector, components at the current and advanced time steps are needed to compute average values for the time step in which a change occurs (see development of a time-varying B vector, $\overline{B}$, in equations (60)-(62) in Cooley (1992)). However, to minimize computer storage, B-vector components at only one time step are represented in MODFE. Therefore, computations for components of the average B vector, $\overline{B}$, are made by the user, external to MODFE, according to equation (62) in Cooley (1992). Details of these computations are given in the section "Changing Stresses and Boundary Conditions With Time" in Torak (1992).

Changes to source-bed heads for transient leakage and to specified-head boundaries require one time step to implement. Computations for these features use the new values of head at the advanced time step to obtain the appropriate head changes for formulating matrix-equation components. For transient leakage, the head change is computed as program vector DHR and represents the difference in source-bed head for the two

time steps; no average heads or head changes are used. For specified-head boundaries, the new values of head are used to compute average heads and average head changes for the first time step that uses the new boundary head. The head change is represented by program vector DHB and the average head is stored in program vector H.

For changing values of point sources and sinks and of areally distributed recharge and discharge, the old (or current) values of stress are input along with the new values. The old values are required because known fluxes at a node are not stored individually in MODFE. Thus, the old values, input as program variable QOLD, are subtracted from the vector Q, which contains the known fluxes at a node. New values for changing stresses and boundary conditions with time are represented by the program variables listed in table 16.

Water-Balance Summary and Flow Imbalance

A water-balance (mass-balance) summary and a flow imbalance are computed in MODFE by evaluating water-balance equations at each node that are based on equations (1) through (5). Volumetric flows [length^3/time] which represent hydrologic processes in these equations are summed by node to obtain a quantitative summary of inflow and outflow rates for each time step (or for steady-state conditions) and of total inflow and outflow volumes from the beginning of the simulation. A flow imbalance is computed by summing rates and volumes that correspond to hydrologic processes according to the system of nodal mass-balance equations given by equation (64) in Cooley (1992). The flow imbalance provides a quantitative measure of the numerical accuracy of the solution.

The matrix equations that are evaluated in MODFE for the water-balance summary are given in Cooley (1992) as equations (58), (83), (230), and (233). Components of these equations are defined in the development of equations (62)-(64) and in the discussion in the section "Extensions to the Basic Equations" in Cooley (1992). The matrix equations are restated here for reference. For linear, nonsteady-state conditions, the water-balance summary is based on equation (58) in Cooley (1992), which is restated as

$$\left(\frac{\underline{\underline{C}}}{(\frac{2}{3})\Delta t_{n+1}} + \underline{\underline{A}}\right)\underline{\delta} = \underline{B} + \underline{\underline{A}}\hat{\underline{h}}_n \qquad (6)$$

For nonlinear, nonsteady-state conditions, the water-balance summary is given by equation (83) in Cooley (1992):

$$\frac{\underline{\underline{C}}}{(\frac{2}{3})\Delta t_{n+1}}\underline{\delta} + (\overline{\underline{\underline{G}}} + \underline{V}) + (\tilde{\underline{\underline{G}}} - \overline{\underline{\underline{G}}})\underline{\delta} - \overline{\underline{B}} \approx 0 \qquad (7)$$

For linear, steady-state conditions, the matrix equation for the water-balance summary is given by equation (230) in Cooley (1992):

$$\underline{\underline{A}}\hat{\underline{h}} - \underline{B} \approx 0 \qquad (8)$$

For nonlinear, steady-state conditions, equation (233) in Cooley (1992) is evaluated for the water-balance summary:

$$\underline{A}(\hat{\underline{h}})\hat{\underline{h}} - \underline{B}(\hat{\underline{h}}) \approx 0 \qquad (9)$$

Each water-balance term contains program variables that represent inflow and outflow rates for the time step and total inflow and outflow volumes from the beginning of the simulation (table 17). Program variables for inflow have names that end with the letter "I", and program variables for outflow have names that end with the letter "O". An exception to this naming convention appies to water-balance terms for the accumulation of water in aquifer storage, where one program variable (SA) represents the volumetric flow rate, and another (TSA) represents the total volume from the beginning of the simulation. Positive values of SA and TSA indicate an increase in water that is stored in the aquifer; negative values indicate the release of water from aquifer storage.

Program variables that represent total volumes of inflow or outflow from the beginning of the simulation for each water-balance term have names that begin with the letter "T" (table 17). These terms are computed by multiplying the volumetric rates, given by the corresponding program variables, by the time step size, and by summing the results for each time step.

Computations of water-balance terms that are contained in the C, G, and V matrices and B vector of equations (6) through (9) are performed by different subroutines according to the version of MODFE that is used. Program structures of each version and subroutine names are given in the section "Program Structures and Lists of Main Programs." Matrix and vector components that are stored in program vectors A and Q are assembled by hydrologic term at each node for the water-balance computations. For linear conditions, formulation of components to the C, G, and V matrices and B vector follows identically from computations that were described in previous sections for forming coefficients and equations for hydrologic terms. For nonlinear conditions, formulation of these components differ from that used for linear conditions in that updated values of hydraulic head from either the previous iteration level (steady-state) or the corrector step (nonsteady-state) are used to evaluate and compute water-balance terms.

Hydraulic head is represented by program vector H for computations of the water-balance summary. For nonsteady-state conditions, program vector H represents the average hydraulic head, $\bar{h}$, for the time step. For steady-state conditions, program vector H represents the steady-state solution, $\hat{h}$. Both h and $\hat{h}$ are computed in subroutine HCALC prior to computing the water-balance summary.

The average head change, δ, of equation (57) in Cooley (1992) and appearing in equations (6) and (7), is represented by different program variables for computation of the water-balance summary depending on the version of MODFE that is used. For linear and nonlinear nonsteady-state versions, program vector DHB represents average head changes at specified head nodes. For linear nonsteady-state conditions, program vector

Table 17.-- *Program variables and subroutines used to compute water-balance summary and flow imbalance*

Water-balance term	Subroutine(s)	Program variable(s)	Component; matrix equation
Accumulation of water in aquifer storage	MASBAL, MBALCB MBALWT, MBWTCB MASOUT	SA TSA	C matrix; (6) and (7)
Steady vertical leakage	MASBAL, MBALCB MBALWT, MBWTCB MASOUT	VLQI, VLQO TLQI, TLQO	V matrix and B vector; (6)-(9)
Transient leakage	MBALCB, MBWTCB MASOUT	VLQI, VLQO TLQI, TLQO	V matrix and B vector; (6) and and (7)
Areally distributed sources and sinks	FMCO MASOUT	DQI, DQO TDQI, TDQO	B vector; (6)-(9)
Point sources and sinks	DATIN MASOUT	WQI, WQO TWQI, TWQO	B vector; (6)-(9)
Specified-head boundaries	MASOUT	HBQI, HBQO THBQI, THBQO	All components; (6)-(9)
Specified-flux boundaries	MASBAL, MBALCB MBALWT, MBWTCB MASOUT	BQI, BQO TBQI, TBQO	B vector; (6)-(9)
Head-dependent (Cauchy-type) flux boundaries	MASBAL, MBALCB MBALWT, MBWTCB MASOUT	BQI, BQO TBQI, TBQO	V matrix and B vector; (6)-(9)
Nonlinear, head-dependent (Cauchy-type) flux	GNBAL, GNBLSS	BNQI, BNQO TBNQI, TBNQO	V matrix and B vector; (7)-(9)
Nonlinear point sinks	GNBAL, GNBLSS	PNQO, TPNQO	V matrix and B vector; (7)-(9)
Nonlinear, steady vertical leakage	VNBAL, VNBLSS	VNLQI, VNLQO TNLQI, TNLQO	V matrix and B vector; (7)-(9)
Flow imbalance	MASBAL, MBALCB MBALWT, MBWTCB GNBAL, GNBLSS VNBAL, VNBLSS MASOUT	ER TER	G and V matrices and B vector, (7)-(9)

Flow across specified-head boundaries is computed for the water-balance summary by evaluating equations (6) through (9) for the patch of elements that is centered on each specified-head node. The computed head at nodes within the patch and the specified head are used to compute a flow balance at each specified-head node. The computations are performed by the subroutines indicated in table 17, with the net flow for each specified-head boundary represented by program vector R. Values in the R vector are summed according to sign (positive for inflow or recharge) to obtain the terms, HBQI and HBQO, in the water-balance summary for volumetric rates of recharge and discharge, respectively. The flow rates are multiplied by the time-step size and summed over all time steps to obtain the total volume of water that recharges (THBQI) or discharges (THBQO) across specified-head boundaries from the beginning of the simulation. For steady-state simulations, the time-step size is set to unity (=1), and only one time step is used; thus, the flow rates, HBQI and HBQO, and the volumes, THBQI and THBQO represent inflows and outflows, respectively, for one time unit.

The water-balance summary can be used to determine the accuracy at which matrix equations (1) through (5) had been solved during the simulation. A printout of the water-balance summary is made after each time step or at the end of the water-table iteration. For linear conditions, the flow imbalance within the water-balance summary indicates the accuracy of the solution method used by MODFE to solve the matrix equations. Acceptable values for the flow imbalance, program variable ER, are about 5 or 6 orders of magnitude smaller than the largest value among the other water-balance terms. Values for ER within this range indicate that the matrix equations were solved to within the limit of accuracy permitted by the computer.

The flow imbalance within the water-balance summary gives a quantitative measure of the accuracy in approximating nonlinear hydrologic processes using MODFE. For nonlinear conditions, differences can exist between values in the C, G, and V matrices and B vector that are computed for the last water-table iteration or corrector step and values that are computed for the water-balance summary. These differences are manifested in the flow imbalance, where values for ER or TER that are larger than the criterion given above can exist if either the predictor-corrector technique or the water-table iteration cannot approximate accurately the nonlinear processes within the aquifer problem. Usually, large flow imbalances can be decreased to acceptable values by performing the same simulation with smaller time-step sizes, which are contained in program vector DT (nonsteady state), or a decreased error tolerance, program variable TOLSW (steady state). Details of inputting these values are given in the section "Input Instructions" in Torak (1992).

COMPUTATIONAL ASPECTS OF SOLUTION METHODS

Direct-- Symmetric-Doolittle Decomposition

Direct solution of finite-element matrix equations (1)-(5) by symmetric- Doolittle decomposition (see section "Symmetric-Doolittle Method," in Cooley, 1992) is implemented in MODFE by subroutines INITB, SETB, and BAND. Inputs are made to subroutine INITB that are used to determine the number of equations that are solved and to allocate computer storage for solution. These inputs define the number of nodes, number of specified-head boundaries, maximum condensed-matrix bandwidth, and the maximum reduced-matrix bandwidth, and are represented by program variables NNDS, NHDS, MBWC, and MBW, respectively. Descriptions of the maximum condensed-matrix bandwidth and the maximum reduced-matrix bandwidth are given in the section "Node Numbering and Determining Bandwidth," in Torak (1992). A discussion of the effects of the bandwidth determinations on computer storage is given in the section "Allocation of Computer Storage and Processing Time."

Computations for determining the maximum reduced-matrix bandwidth and the writing of model information to computer files are performed in subroutine SETB. Although a value for the maximum reduced-matrix bandwidth is input as program variable MBW, this value is used only as an estimate for allocating computer storage. The actual value for the maximum reduced-matrix bandwidth is determined in subroutine

SETB and is represented by program variable IBND. The value for IBND is used to compute the amount of storage locations that is overwritten during solution by the direct method. Components of the C, V, and G matrices and B vector of equations (1)-(5), element areas, and node incidences are written to computer files by this subroutine to enable overwriting of storage locations during solution. The process of overwriting storage locations by the solution methods is described in the section "Solution Methods." Other functions of subroutine SETB are outlined in table 15 and are indicated by comment statements within the subroutine (see section "List of Subroutines").

Matrix equations (1)-(5) are solved in subroutine BAND. The solution procedure is given by equations (263)-(265) in Cooley (1992), and consists of three steps: factorization, equation (263); forward substitution, equation (264); and back substitution, equation (265). These steps are outlined by comment statements within the subroutine (see section "List of Subroutines").

Iterative-- Modified Incomplete-Cholesky Conjugate Gradient

The iterative, modified incomplete-Cholesky conjugate gradient (MICCG) method of solution is implemented in MODFE by subroutines INITCG, SETCG, and MICCG. Inputs associated with the MICCG method are made in subroutine INITCG. These inputs consist of values for the maximum number of iterations and the closure tolerance, and are represented by program variables NIT and TOL, respectively. Variables that define the beginning of storage locations within the general storage vector G also are evaluated in subroutine INITCG. Details about the allocation of computer storage in MODFE are given in the following section.

Components of the C, V, and G matrices and B vector of equations (1)-(5), element areas, and node incidences are written to computer files by subroutine SETCG. This process enables storage locations that are occupied by these terms to be overwritten during solution by the MICCG method. Overwriting storage locations by the solution methods is described in the section "Solution Methods."

Computations for solving equations (1)-(5) by the MICCG method are contained in subroutine MICCG. The solution process begins by approximately factoring the reduced matrix into an upper triangular matrix, $\tilde{U}$, which is represented by program vector AF. The factorization is performed according to equation (274) in Cooley (1992), and is modified by the row-sums agreement given by equations (276)-(278) in Cooley (1992). An intermediate vector, $\tilde{y}$, is computed by forward substitution, and the displacement vector, s, is computed by backward substitution according to equations (283) and (284) in Cooley (1992). These computations are identified in the subroutine by comment statements.

Computations for the generalized conjugate-gradient algorithm of equation (271) in Cooley (1992) are performed in subroutine MICCG by formulating iteration parameters β_k and the A-orthogonal vectors, $\underline{p}_k$. The iteration parameter used for each vector is represented by program variable S, and the vectors are represented by program variable P. The scaled displacements, $\alpha_k\underline{p}_k$, are represented by program variable TMPA, and are summed for each equation and stored in program variable B to give the total displacement from the initial condition. Similarly, scaled residuals, $-\alpha_k\underline{\underline{A}}\underline{p}_k$, are computed for each equation, and the sum is stored in program variable R. The largest absolute value of the displacements is determined according to equation (285) in Cooley (1992) by comparing absolute values of TMPA, and the result is represented as program variable PMAX. Similarly, the largest absolute scaled residual, given by equation (289) in Cooley (1992) is determined, and the result is represented by program variable RMAX. The iterative process is concluded (has converged) when values of RMAX and PMAX are less than the closure tolerance, which is represented by program variable TOL.

Messages are printed out from subroutine MICCG that describe the progress toward attaining a solution. If convergence is achieved, then a message is printed to that effect along with the number of iterations that was required. If convergence was not achieved, then a message is printed indicating that the solution failed to

converge, followed by values for the maximum number of iterations (NIT), the largest absolute scaled residual (RMAX), and the computed head changes (program vector B).

ALLOCATION OF COMPUTER STORAGE AND PROCESSING TIME

Several techniques for storing and representing program variables have been used during the design of MODFE to minimize computer storage and processing time. A single program vector, G, which is dimensioned at the time of execution of MODFE, is used to store most of the information that is needed to conduct a simulation. Components of matrix equations (1)-(5) are stored within vector G in a manner that decreases the range of indexes that are searched by the computer during assembly of the matrix equations. The order of the coefficient matrix is decreased by eliminating specified-head nodes from the solution process, thereby allowing a reduced-matrix equation to be solved. Use of the reduced matrix for solution results in fewer equations and storage locations than if specified-head nodes were retained in the formulation. Components of the reduced matrix are stored in a condensed form, thus eliminating as many zero entries from storage as possible. In addition, storage locations within program vector G are overwritten, or reused, routinely during solution of the matrix equations. Details of these techniques are given in the following sections.

General-Storage Vector G

Nearly all program information that is needed to form and solve matrix equations (1)-(5) and to compute a water-balance summary is contained in the general-storage vector G. The vector G is subdivided into smaller vector lengths of storage, and all values are stored in single-subscripted form. Computer storage is allocated at the time MODFE is executed by determining the beginning locations (starting addresses) of the vector lengths that correspond to program vectors stored in general-storage vector G. Values for program variables that represent starting addresses are computed automatically by MODFE from inputs that define program dimensions and problem specifications (see section "Input Instructions" in Torak (1992)). Descriptions of the terms for which computer storage is allocated within the general-storage vector G and names of subroutines that perform the storage allocation are given in table 6. Program variables that represent starting addresses, vector lengths, and vectors stored within the general-storage vector G are given in the section "Variable Lists and Definitions."

The amount of computer storage that is allocated to the general-storage vector G is assigned by the main programs of MODFE in a Fortran statement given as COMMON/PRIME/G(nnnn), where "nnnn" defines the number of single-precision storage locations assigned to vector G. During execution of MODFE, the number of storage locations that actually is used to dimension program vectors is printed out by the subroutines listed in table 6. This value is represented by program variable ISUM. For MODFE to execute successfully, the value of nnnn must be equal to or greater than the value of ISUM. Hence, the user must determine the size of the simulation (value of ISUM) and make necessary adjustments to the dimension of the general-storage vector G in the COMMON statement so that the G vector can accommodate all storage requirements.

The first five vector lengths within the general-storage vector G are overwritten and reused routinely when forming and solving matrix equations (1)-(5). These locations contain terms for the C, V, and G matrices and B vector (stored in condensed form in program vector A) of these equations, element areas, element incidences, and x and y coordinates of nodes. Values contained in the first three vector lengths of general-storage vector G (nodal coordinates are excluded) are written to computer files by subroutines SETB and SETCG prior to being overwritten in subroutines FMEQ and WTFMEQ, where the matrix equations are assembled in condensed form. These locations in general-storage vector G are overwritten again by the equation-solving subroutines, BAND and MICCG, as the condensed-matrix equations are expanded to the reduced-matrix form. Values that were written to the computer files are read and placed in the original storage

locations in general-storage vector G by subroutine RDTP for forming and solving equations on the corrector step (if appropriate) and for computing the water-balance summary. Details about forming the condensed matrix, reduced matrix, and the expansion of the condensed matrix to the reduced form during solution are given in the following sections.

To ensure that the first five storage locations in the general-storage vector G provide at least as much computer storage as required by subroutines BAND and MICCG, the value of ISUM is compared with the storage requirements of both solution methods after computer storage for the fifth location is calculated. The amount of computer storage required by either solution method is represented in the corresponding subroutines, INITB and INICG, as program variable NC. If ISUM is less than NC, then ISUM is assigned the value of NC + 1.

Other storage locations in the general-storage vector G than those described above are reused during simulation. These are identified in the section "Variable Lists and Definitions" as multiple descriptions for storage locations in the general-storage vector G.

Reduced Matrix A

Matrix equations (1)-(5) are placed in what is termed a reduced form in MODFE by eliminating equations at specified-head boundaries (nodes) from the solution process. This decreases the order of the coefficient matrix that is formed, and creates a smaller matrix termed the reduced matrix A. Use of the reduced matrix A results in a savings of computer storage and processing time from what would be needed to form and solve a larger matrix containing entries for specified-head nodes.

Formation of the reduced Matrix A is explained by the following example. Given a finite-element mesh consisting of 3 elements and 5 nodes (fig. 15*A*), the coefficient matrix contains nonzero entries at the locations indicated in figure 15*B*. Values of the coefficients are determined according to the matrix equation that is solved, but generally consist of elements of the C, V, and G matrices in equations (1)-(5). Each entry to the coefficient matrix multiplies an entry in the solution vector, which is either a head change, δ, or a displacement, s_x, depending on the the equation that is solved. Because the head change or displacement is known at specified-head nodes, these equations are be removed from the matrix equation, and terms in the remaining equations that multiply the known head change or displacement at the specified-head nodes are placed on the right side of the matrix equation that is formed. The result is a reduced matrix A of a lower order and fewer equations to be solved than the coefficient matrix (fig. 15*C*); thus, decreasing computer storage and processing time.

Computer storage for forming and solving the finite-element matrix equations is allocated on the basis of the dimensions of the reduced matrix A. Values for the number of nodes and specified-head boundaries, program variables NNDS and NHDS, respectively, are used to determine the order of the reduced matrix (fig. 15*C*) and the number of equations that are formed. The order of the reduced-matrix, given by the difference NNDS – NHDS, is used in MODFE to eliminate matrix assembly and solution at nodes that represent specified-head boundaries. The reduced-matrix bandwidth, program variable MBW (fig. 15*C*), is used to allocate storage for elements of reduced matrix A that are needed to solve matrix equations (1)-(5) by the direct-solution method (subroutine BAND). Because of symmetry, only entries on and to the right of the main diagonal in reduced matrix A (fig. 15*C*) are needed for solution by subroutine BAND.

A value for the reduced-matrix bandwidth, MBW, is required as input to subroutine INITB. Details for estimating the maximum value of the reduced-matrix bandwidth for input as MBW are given in Torak (1992).

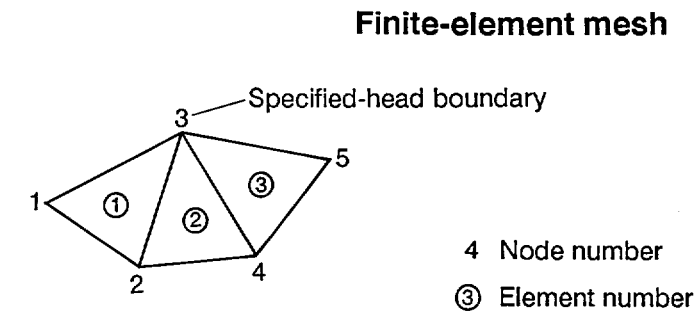


Figure 15.-- Diagrams of (*A*) three-element, five-node, finite-element mesh containing a specified-head boundary at node 3, (*B*) coefficient matrix, and (*C*) reduced matrix A.

Table 18.-- *Values for indicator vector, IN, corresponding to nodes in finite-element mesh in Figure 15*

Node I	IN(I)
1	1
2	2
3*	-1
4	3
5	4

* Specified-head node

Condensed Matrix

Components of the C, V, and G matrices of equations (1)-(5) are stored in a compact, or condensed form that saves computer storage and processing time thereby increasing the computational efficiency of MODFE. The condensed matrix contains only the nonzero entries of reduced matrix A that are located on and to the right of the main diagonal (fig. 16*A*,*B*). Because of symmetry in the reduced matrix, entries to the left of the main diagonal are identical to entries that are located in the corresponding transpose positions to the right of the main diagonal. The main diagonal of reduced matrix A is stored as the first column of the condensed matrix, and nonzero terms to the right of the main diagonal in the reduced matrix are stored in consecutive locations to the right of the first column in the condensed matrix.

Although the reduced matrix A and the condensed matrix are shown in matrix form (figs. 15 and 16), they are computed and stored in vector form as program vector A. A fixed number of storage locations in program vector A is used to represent each row of the condensed matrix. The number of storage locations assigned to each row of the condensed matrix is termed the condensed-matrix bandwidth, and is determined by the maximum number of nonzero entries in each row of reduced matrix A. The condensed-matrix bandwidth is represented in MODFE as program variable MBWC (fig. 16*B*). The value of MBWC is input to MODFE in subroutines INITB and INITCG.

The condensed matrix is assembled in the equation-forming subroutines (table 9) by accumulating components of the C, V, and G matrices of equations (1)-(5) and storing the results in the appropriate locations within program vector A (fig. 16*C*). This assembly overwrites locations in program vector A that contain components of the C, V, and G matrices for all nodes. During assembly by subroutines FMECWT, FMEPWT, and FMEQ, main-diagonal entries are first stored in program vector AD and then transferred to program vector A.

Components of the C, V, and G matrices that are used to form the condensed matrix are stored in a compact manner within program vector A, which allows matrix assembly to be computationally efficient. For each node, components of the C, V, and G matrices are stored in consecutive locations within program vector A (fig. 17). The amount of computer storage that is assigned to program vector A for matrix components of all nodes is computed in subroutines INITB and ININTCG as program variable NA (fig. 17). The amount of storage in program vector A that is assigned to each node is a function of the maximum condensed-matrix bandwidth, MBWC, and is defined as program variable IW, where IW = MBWC + 1. Components for the condensed matrix are assembled by incrementing indexes to program vector A by multiples of IW, where the number of multiples of IW corresponds to the difference between node numbers in an element. Because the node numbers within an element usually are close in value, all components that are needed to assemble the condensed matrix at a node are stored within a small number of multiples of IW storage locations in program vector A. In addition, node numbering of the finite-element mesh that minimizes the reduced matrix bandwidth also minimizes the number of multiples of IW storage locations that need to be indexed by the computer in order to locate components of the condensed matrix for any node. By placing all matrix components for a node in consecutive storage locations, matrix assembly is more computationally efficient than if each component, such as storage coefficient, vertical leakage, and transmissivity, were represented by individual matrices or vectors.

Solution Methods

Computer storage for the solution methods is allocated in the general-storage vector G by subroutine INITB for the direct method and by subroutine INITCG for the conjugate-gradient method. The amount of single-precision storage that is required by each solution method is represented in these subroutines as program variable NC. For the direct method, NC is computed as

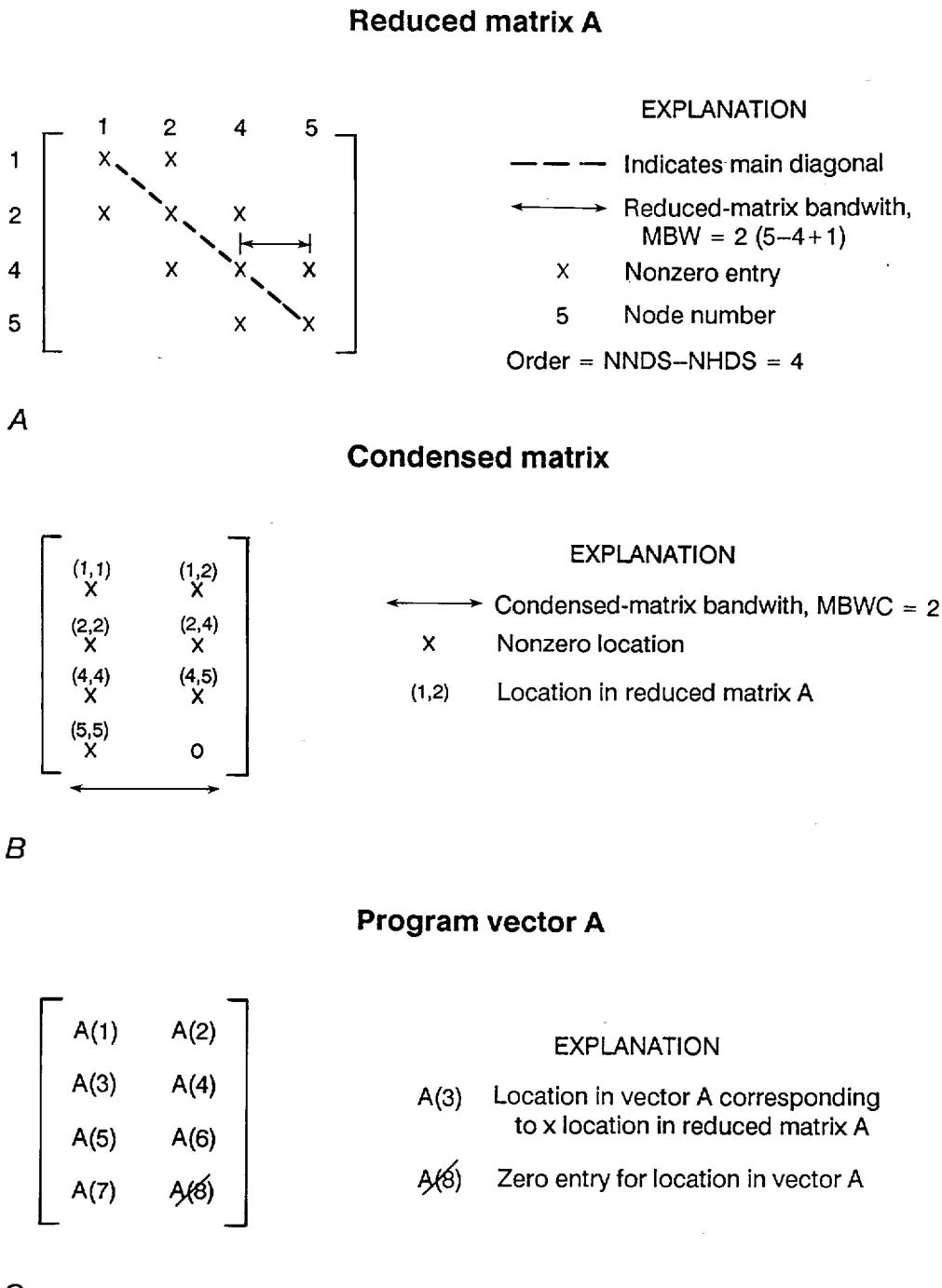


Figure 16.-- Diagrams of (*A*) reduced matrix A, (*B*) condensed matrix, and (*C*) condensed matrix represented by program vector A.

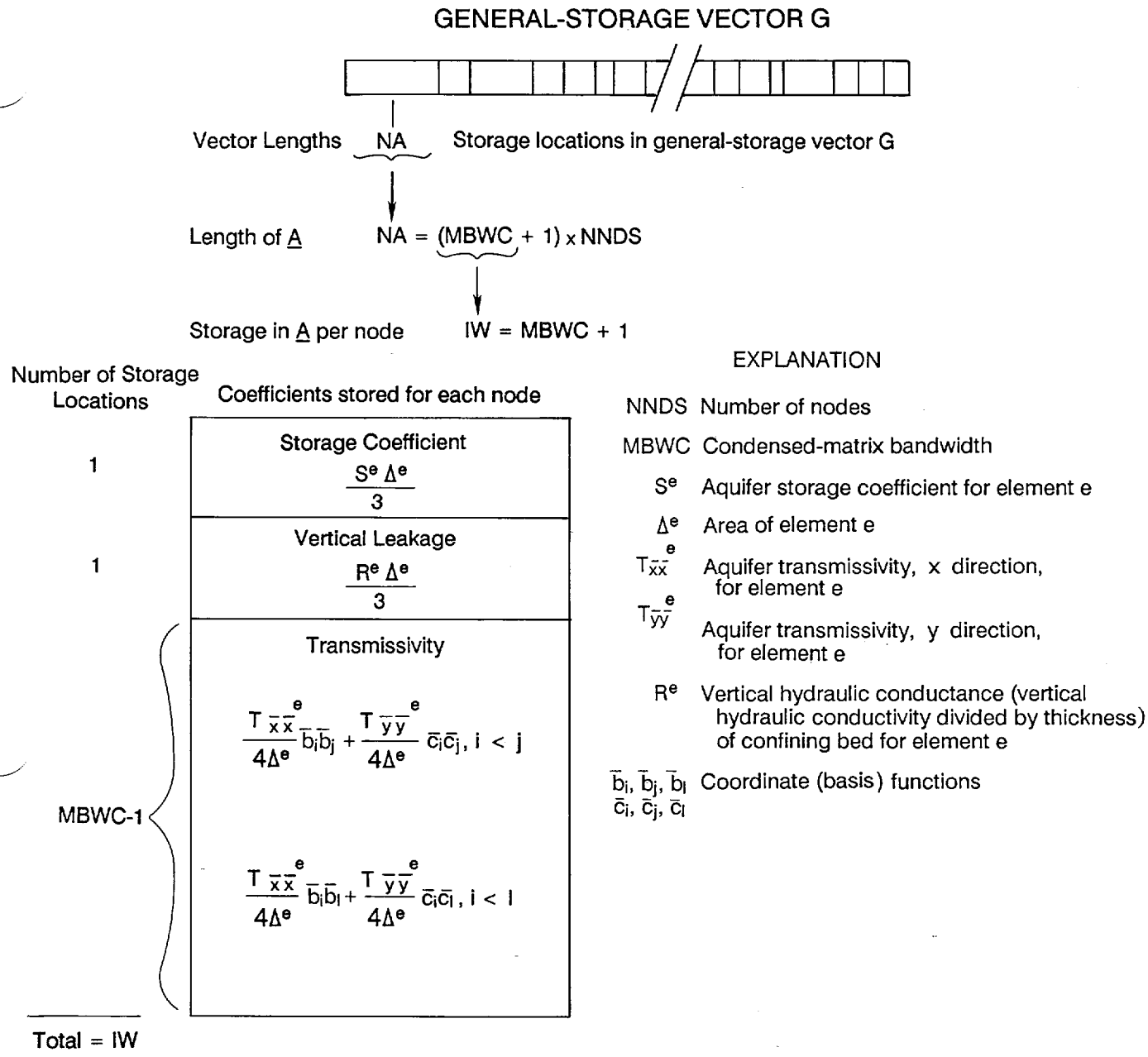


Figure 17.-- Storage locations in general-storage vector G and matrix components stored in program vector A.

$$NC = MBW \times (NNDS - NHDS - MBW) + (MBW \times (MBW + 1))/2 , \qquad (10)$$

where MBW is the estimate of the reduced-matrix bandwidth, NNDS is the number of nodes, and NHDS is the number of specified-head boundaries. Values for these terms are input to subroutine INITB prior to computing NC.

For the conjugate-gradient method, the amount of computer storage that is required is computed as

$$NC = NA + (MBWC + 3) \times (NNDS - NHDS) , \qquad (11)$$

where MBWC is the condensed-matrix bandwidth and NA = (MBWC + 1) × NNDS. A discussion of bandwidth determination is given in the section "Node Numbering and Determining Bandwidth" in Torak (1992). Descriptions of the use of MBWC and MBW in computations and in allocating computer storage have been given in preceding sections.

A comparison of terms in equations (10) and (11) that are used to define storage requirements for both solution methods indicates that computer storage for the direct method is dependent on the reduced-matrix bandwidth, MBW, while computer storage for the conjugate-gradient method is dependent on the condensed-matrix bandwidth, MBWC. As discussed in the section "Node Numbering and Determining Bandwidth" in Torak (1992), the reduced-matrix bandwidth is a function of node numbering in the finite-element mesh, and the condensed-matrix bandwidth is a function of the mesh design. Numbering nodes so that the maximum difference between node numbers is minimized in elements that do not contain specified-head boundaries will decrease the reduced-matrix bandwidth, MBW, and decrease computer storage requirements for the direct-solution method. The condensed-matrix bandwidth, MBWC, is determined by the maximum number of connections from a node in the center of a patch of elements to higher-numbered nodes in the patch. Thus, by eliminating excessive connections of element sides to a node, the condensed-matrix bandwidth can be minimized, thereby decreasing computer storage for the conjugate-gradient method.

Although computer storage for the conjugate-gradient method is not dependent on node numbering of the finite-element mesh, and computer storage for the direct method is not dependent on mesh design, both node numbering and mesh design affect computer storage and processing time for both solution methods and for computations of the water-balance summary. Node numbering to minimize the reduced-matrix bandwidth, MBW, decreases the computer-processing time needed to form the condensed matrix and to compute a water-balance summary by minimizing the number of sets of IW storage locations that separate the IW locations corresponding to nodes in an element (see discussion of storage locations for each node in the preceding section). Similarly, a mesh design that minimizes the condensed-matrix bandwidth, MBWC, also decreases computer storage and processing time by eliminating unused storage locations that are assigned to the set of IW locations for each node. Usually, excessively large values of MBWC exist at only a few nodes in a mesh that is designed by hand. However, computer storage and processing time for all nodes are affected by the excessively large value of MBWC, which satisfies storage requirements at just a few nodes. Instructions about designing a finite-element mesh to minimize MBW and MBWC are given in Torak (1992).

REFERENCES

Cooley, R.L., 1992, A MODular Finite-Element model (MODFE) for areal and axisymmetric ground-water-flow problems, part 2: derivation of finite-element equations and comparisons with analytical solutions: U.S. Geological Survey Techniques of Water Resources Investigations, Book 6, Chapter A4 (in press).

Torak, L.J., 1992, A MODular Finite-Element model (MODFE) for areal and axisymmetric ground-water-flow problems, part 1: model description and user's manual: U.S. Geological Survey Open-File Report 90-194, 153 p.

APPENDICES

Lists of program variables and definitions and of subroutines are given as appendices. Program variables are listed and defined according to the criteria described in the following section. Subroutines are listed in alphabetical order following the variable lists.

Variable Lists and Definitions

Program variables are listed and defined according to their usage in MODFE. Where appropriate, tables list program variables that are used in Fortran CALL statements and their corresponding starting locations in the general-storage vector G. The following criteria were used to group program variables and definitions for this appendix.

- Fortran COMMON statements.
  COMMON statements are listed in alphabetical order. Program variables are listed and defined in the sequence in which they appear in each COMMON statement.

- General-storage vector G.
  Starting-location variables, lengths of vector storage in the general-storage vector G, and program variables that are represented by the storage locations are listed and defined in the sequence in which vector G is filled. General-storage vector G is filled according to the following sequence:

  - starting-location variables in Fortran COMMON/ADR/,
  - conjugate-gradient method of solution,
  - water-table (unconfined) conditions,
  - nonlinear, head-dependent (Cauchy-type) flux, point sinks, and steady vertical leakage, and
  - vertical leakage of water stored elastically in a confining bed (transient leakage).

- Text for general-storage vectore G.
  Computer storage for the above simulation capabilities is not allocated and starting locations are eliminated if MODFE does not contain the corresponding program structure. Reuse of storage within the general storage vector G is defined by multiple definitions or program variables for the same storage locations.

- Main program.
  Program variables that are not listed in Fortran COMMON statements, but are used in the main program and in subroutines to control computational steps.

- Linear versions.
  Program variables that are not listed in Fortran COMMON statements are listed and defined in alphabetical order.

- Transient leakage.
  Subroutines are listed in alphabetical order and program variables within each subroutine are listed and defined in alphabetical order.

- Changing time-step sizes, stresses, and boundary conditions.
  Subroutines and program variables within each subroutine are listed in alphabetical order.

- Nonlinear versions.
  Program variables that are not listed in Fortran COMMON statements are listed and defined in alphabetical order for each nonlinear hydrologic term or simulation feature according to the following sequence:

  - water-table (unconfined) conditions,
  - head-dependent (Cauchy-type) flux and point sinks,
  - steady vertical leakage, and
  - steady-state conditions.

- Solution methods.
  Subroutine names and program variables are listed and defined in alphabetical order. Program variables for the direct-solution method are presented first, followed by program variables for the conjugate-gradient method.

Fortran COMMON Statements

	Variable	Definition
COMMON/BAL/	SA	Volumetric rate of accumulation of water in aquifer storage [length^3/time].
	WQI	Volumetric recharge rate to aquifer by point sources [length^3/time].
	WQO	Volumetric discharge rate from aquifer by point sinks [length^3/time].
	DQI	Volumetric recharge rate to aquifer by areally distributed sources [length^3/time].
	DQO	Volumetric recharge rate from aquifer by areally distributed sinks [length^3/time].
	VLQI	Volumetric recharge rate to aquifer by vertical leakage from a confining bed (steady and transient leakage [length^3/time].
	VLQO	Volumetric discharge rate from aquifer by vertical leakage from a cofining bed (steady and transient leakage) [length^3/time].
	BQI	Volumetric recharge rate to aquifer by Cauchy-type boundaries [length^3/time].
	BQO	Volumetric discharge rate from aquifer by Cauchy-type boundaries [length^3/time].
	ER	Flow imbalance of volumetric rates for time step [length^3/time].
COMMON/CHG		
	NWCH	Number of the time step when point sources or sinks are changed.
	NQCH	Number of the time step when areally distributed sources or sinks are changed.

COMMON/CHG/--continued

Variable	Definition
NHRCH	Number of the time step when values of source-layer heads HR are changed in zones simulating steady-leakage (no transient effects from confining-bed storage).
NBQCH	Number of the time step when specified flux or head-dependent (Cauchy-type) flux is changed.
NHCH	Number of the time step when values of specified-head boundaries change.
NCBCH	Number of the time step when values of source-layer heads NR are changed in zones simulating transient leakage.
NVNCH	Number of the time step when controlling heads HS to nonlinear vertical-leakage functions are changed.
NGNCH	Number of the time step when controlling heads HRK and HRL to nonlinear, head-dependent (Cauchy-type) fluxes are changed.

COMMON/GDIM/

ISUM	Number of single-precision words of storage in G required to execute MODFE.

COMMON/GNBL/

BNQI	Volumetric recharge rate to aquifer by nonlinear head-dependent (Cauchy-type) fluxes [length^3/time].
BNQO	Volumetric discharge rate from aquifer by nonlinear head-dependent (Cauchy-type) fluxes [length^3/time].
TBNQI	Total volume of water [length^3] recharged to aquifer by nonlinear, head-dependent (Cauchy-type) boundaries.
TBNQO	Total volume of water [length^3] discharged from aquifer by nonlinear, head-dependent (Cauchy-type) boundaries.

COMMOM/GNBL/--continued

Variable	Definition
PNQO	Volumetric discharge rate from aquifer by nonlinear point sinks [length^3/time].
TPNQO	Total volume of water [length^3] discharged from aquifer by nonlinear point sinks.

COMMOM/IND/

IRAD	Indicator for axisymmetric radial flow.
IUNIT	Indicator for converting lengths from map-scale units to field units.
ISTD	Indicator for steady-state simulations.

COMMON/IPRN/

IPND	Indicator variable to suppress printout of node numbers for each element.

COMMON/ITP/

IIN	Fortran-unit number for reading input data (=50).
IOUT	Fortran-unit number for program output (=60).
ITA	Fortran-unit number for writing the A vector to peripheral storage (=55).
ITB	Fortran-unit number for writing vectors AR and ND to peripheral storage (=56).

COMMON/NO/

NELS	Number of elements.
NNDS	Number of nodes.

COMMON/NO/--continued

NSTEPS	Number of time steps.
NPER	Number of stress periods.
NZNS	Number of aquifer-property zones.
NWELS	Number of point sources or sinks.
NQBND	Number of Cauchy-type-boundary element sides.
NHDS	Number of specified-head-boundary nodes.
NEQ	Number of equations from reduced matrix.
MBWC	Maximum condensed-matrix bandwidth.
MBW	Maximum reduced-matrix bandwidth (if direct-solution method is used to solve equations).
NIT	Maximum number of iterations (if iterative method, MICCG, is used to solve equations).

COMMON/PRIME/

G(5000)	The general-storage vector G and its initial dimension.

COMMON/SCLE/

SCALE	Scaling factor to convert map units of length to field units.

COMMON/TBAL/

TSA	Total volume of water [length^3] accumulated in aquifer storage during the simulation.
TWQI	Total volume of water [length^3] recharged to aquifer during simulation by point sources.
TWQO	Total volume of water [length^3] discharged from aquifer during simulation by point sinks.
TDQI	Total volume of water [length^3] recharged to aquifer during simulation by areally distributed sources.

COMMON/TBAL (continued)

Variable	Definition
TDQO	Total volume of water [length^3] discharged from aquifer during simulation by areally distributed sinks.
TLQI	Total volume of water [length^3] recharged to aquifer during simulation by vertical leakage (steady and transient leakage).
TLQO	Total volume of water [length^3] discharged from aquifer during simulation by vertical leakage (steady and transient leakage).
TBQI	Total volume of water [length^3] recharged to aquifer during simulation by Cauchy-type boundaries.
TBQO	Total volume of water [length^3] discharged to aquifer during simulation by Cauchy-type boundaries.
THBQI	Total volume of water [length^3] recharged to aquifer during simulation by specified-head boundaries.
THBQO	Total volume of water [length^3] discharged to aquifer during simulation by specified-head boundaries.
TER	Flow imbalance of total volumes for simulation [length^3].

COMMON/VNLBL/

VNLQI	Volumetric inflow rate (recharge) to aquifer by nonlinear steady-leakage functions [length^3/time].
VNLQO	Volumetric outflow rate (discharge) from aquifer by nonlinear steady-leakage functions [length^3/time].
TNLQI	Total volume of water (length^3) recharged to aquifer during simulation by nonlinear steady-leakage functions.
TNLQO	Total volume of water (length^3) discharged from aquifer during simulation by nonlinear steady-leakage functions.

General Storage Vector G

Starting-location variables in Fortran COMMON/ADR/

Starting location variable	Vector length	Variable in G	Definition
IAA	(MBWC+1) × NNDS	A	Contains partially formulated coefficients for matrix equation of each node.
IARA	2 × NELS	AR	One-third the area of element e, $(1/3)\Delta^e$.
		AD	Temporary storage of main diagonal of upper-triangular matrix A.
		DTK	Predicted thickness change [length].
		VQK	Volumetric flow rate [length^3/time] for node K on head-dependent (Cauchy-type) boundary.
INDA	4 × NELS	ND	Node numbers for each element.
IXGA	NNDS	XG	Global x coordinates of nodes.
		VQL	Volumetric flow rate [length^3/time] for node L on head-dependent (Cauchy-type) boundary.
IYGA	NNDS	YG	Global y coordinates of nodes.
		DTK	Adjusted thickness changes over time step.
		R	Volumetric flow rates at specified head boundaries [length^3/time].
		WVCN	Effective vertical hydraulic conductivity $\sum_{e_i} K'^{e}_{zz} \Delta^{e}$ for node i [length^3/time].
IATA	MBW	AT	Temporary variable used to store one row of upper triangularized A matrix in subroutine BAND.
IQA	NNDS	Q	Sum of known stresses at each node [length^3/time].
IBA	NNDS-NHDS	IZN	Zone numbers for each element.

General Storage Vector G

Starting-location variables in Fortran COMMON/ADR/ (continued)

Starting location variable	Vector length	Variable in G	Definition
IHA	NNDS	H	Hydraulic head [length].
IHRA	NNDS	HR	Source-layer head [length].
IHBA	NNDS	DHB	Head difference over a time step at specified-head boundary [length].
IALA	NQBND	ALPH	The α term in equation (4) in Cooley (1992) [length/time] for head-dependent (Cauchy-type) boundaries.
IQBA	NQBND	QBND	The qB term in equation (4) in Cooley (1992) [length^2/time] for specified-flux boundaries, if $\alpha = 0$, or, qB/ α if qB and α are nonzero.
ICKA	NQBND	CFDK	The coefficient $(\alpha L)_{ik}/2$ if $\alpha \neq 0$, or $L_{ik}/2$ if $\alpha = 0$ and $qB \neq 0$, for node k on a Cauchy-type boundary.
ICLA	NQBND	CFDL	The coefficient $(\alpha L)_{ik}/2$ if $\alpha \neq 0$, or $L_{ik}/2$ if $\alpha = 0$ and $qB \neq 0$, for node l on a Cauchy-type boundary.
IHKA	NQBND	HK	Head external to aquifer region for Cauchy-type boundary at node k [length].
IHLA	NQBND	HL	Head external to aquifer region for Cauchy-type boundary at node l [length].
IDTA	MXSTPS	DELT	Time-step sizes [time].
IJPA	(MBWC-1) × NNDS	JPT	Pointer vector, stores node numbers of off-diagonal terms in the upper-triangular form of coefficient matrix A.

Starting-location variables in Fortran COMMON/ADR/ (continued)

Starting location variable	Vector length	Variable in G	Definition
INA	NNDS+1	IN	Indicator vector for ordering equations and specified-head-boundary nodes.
IKA	NQBND	KQB	Node k on a head-dependent (Cauchy-type) boundary.
ILA	NQBND	LQB	Node l on a head-dependent (Cauchy-type) boundary.
IDZA	NBCZ	IDZ	Zone number for head-dependent (Cauchy-type) boundary.
IDSA	NBCZ	IDS	Number of head-dependent (Cauchy-type) boundary sides in each zone.
Conjugate-gradient method			
IAFA	(MBWC+1) × (NNDS+1)	AF	Element $\tilde{\alpha}$ ii and $\tilde{u}$ ij of upper triangular matrix U.
IXA	MBWC × (NNDS-NHDS)	X	Elements of $\underline{y}$ and $\underline{\tilde{D}}$ in equation (281) and $\underline{x}$ in equation (271) in Cooley (1992).
IPA	NNDS-NHDS	P	Elements $\underline{p}^k$ that are A-orthogonal to $\underline{p}_{k-1}$ in equation (271) in Cooley (1992).
IRA	NNDS-NHDS	R	Elements of $\underline{r}_i$ of flow-balance residual interations.
Water-table (unconfined) conditions			
IDHA	NNDS-NHDS	DH	Average head change [length].
ITKA	NNDS	THK	Aquifer thickness [length].
ITPA	NNDS	TOP	Altitude of top of aquifer [length].

Nonlinear, head-dependent (Cauchy-type) flux, point sinks, and steady-vertical leakage

Starting location variable	Vector length	Variable in G	Definition
IGCA	NBNC+NPNB	GC	α term for nonlinear head-dependent (Cauchy-type) boundary and nonlinear point sink.
IHRK	NBNC	HRK	Boundary or external head hr [length] at node on nonlinear head-dependent (Cauchy-type) boundary .
IHRL	NBNC	HRL	Boundary or external head hr [length] at node I on nonlinear head-dependent (Cauchy-type) boundary.
INSA	NLZA	INLS	Number of nonlinear, head-dependent (Cauchy-type) boundary sides.
INZA	NLZA	INLZ	Zone number for nonlinear head-dependent (Cauchy-type) boundary.
IZRK	NBNC	ZRK	Controlling head or altitude [length] for nonlinear head-dependent (Cauchy-type) flux at node k.
IZRL	NBNC	ZRL	Controlling head or altitude [length] for nonlinear head-dependent (Cauchy-type) flux at node I.
IKRA	NBNC	KQB	Node k on a nonlinear head-dependent (Cauchy-type) boundary.
ILRA	NBNC	LQB	Node I on a nonlinear, head-dependent (Cauchy-type) boundary.
IZPA	NPNB	HZP	Reference altitude, zp, for nonlinear point sink [length].
IKPA	NPNB	KP	Node number of nonlinear point sink.

Nonlinear, head-dependent (Cauchy-type) flux, point sinks, and steady-vertical leakage (continued)

Starting location variable	Vector length	Variable in G	Definition
IECA	NNDS	E	Nodal coefficient $\sum_{e_i}$ (1/3) $R_e^e \Delta^e$ or $\sum_{e_i}$ (1/3) $R_a^e \Delta,^e$ for nonlinear steady vertical leakage [length^2/time].
IHSA	NNDS	HS	Controlling head Ha, or altitude,ze or zt, at node [length].
Transient leakage			
ICHA	MCBN	CH	Main-diagonal term $C_{hi,\,n+1}/\Delta t_{n+1}$ of equation (206) in Cooley (1992).
ICQA	MCBN	CBQ	Term from equation (206) in Cooley (1992) for right side of matrix equations (1) and (3).
IDHRA	NNDS	DHR	Change in source-layer head $H_{i,n+1}$-$H_{i,n}$.
ICTQA	5×MCBN	CBTQ	Transient-leakage terms $\sum_{m=1}^{N_1} e^{-\alpha_m \gamma_i \Delta t_{n+1}} \hat{I}_{mi,\,n}$ (three terms), and $\sum_{m=1}^{N_1} e^{-\beta_m \gamma_i \Delta t_{n+1}} \hat{I}_{mi,\,n}$ (two terms) in Cooley (1992).
IGMA	NNDS	GMA	(1) Effective specific storage, $\sum_{e_i} S_s'^e \Delta^e$, for node i, of equation (167) in Cooley (1992).

Transient leakage (continued)

Starting location variable	Vector length	Variable in G	Definition
IGMA	NNDS	GMA (continued)	(2) Nodal values of γ_i, given by equation (167) in Cooley and Torak (1992).
IALFA	3	ALF	Coefficients α_m for approximating $S_1(\Delta t_D)$ by $M_1(\Delta t_D)$ in equation (188) in Cooley (1992).
IACA	3	AC	Coefficient A_m for approximating $S_1(\Delta t_D)$ by $M_1(\Delta t_D)$ in equation (188) in Cooley (1992).
IBTA	2	BTA	Coefficient β_m for approximating $S_2(\Delta t_D)$ by $M_2(\Delta t_D)$ in equation (189) in Cooley (1992).
IBCA	2	BC	Coefficient β_m for approximating $S_2(\Delta t_D)$ by $M_2(\Delta t_D)$ in equation (189) in Cooley (1992).

Main Program

Variable	Definition
I	Index for time-step loop.
IT	Counter for iteration loop.
ITER	Number of current iteration.
JP	Index for stress-period loop.
MCBN	Maximum number of nodes where transient leakage is simulated.
NBCZ	Number of zones for head-dependent (Cauchy-type) boundaries.
NCBZ	Number of nodes where transient leakage is simulated.
NZLA	Number of nonlinear head-dependent (Cauchy-type) boundary zones.

Subroutine DATIN

Variable	Definition
ALPH(J)	α term for head-dependent (Cauchy-type) boundary [length/time].
HB	Head on specified-head boundary [length].
IPBC	Indicator variable to suppress printout of initial Cauchy-type boundary data.
IPH	Indicator variable to suppress printout of initial aquifer heads.
IPHB	Indicator variable to suppress printout of initial specified-head boundaries.
IPHR	Indicator variable to suppress printout of initial source-bed heads.
IPND	Indicator variable to suppress printout of node numbers for each element.
IPQW	Indicator variable to suppress printout of initial point sources and sinks.
IPXY	Indicator variable to suppress printout of x-y coordinates.
NDC	Index for vector containing node numbers.
NLHS	Number of specified-head nodes that are located at the end of the head vector.
QBND(J)	qB term for specified-flux boundary [length^2/time].
QWEL	Strength of point source or sink [length^3/time].
WF	Weighting factor for specified-head-boundary nodes, equals 2/3 for transient simulations and 1 for steady state.

Subroutines DATOUT and EXTRAP

Variables that are contained in these subroutines have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Subroutine FMCO

Variable	Definition
ALF	α term for head-dependent (Cauchy-type) boundary [length/time].
ANG	Rotation angle θ of equation (20) in Cooley (1992), to transform global Cartesian coordinates to local coordinates.
AREA	Twice the area of element e, $2\Delta^e$,[length^2].
BJ,BK,BL	Differences in nodal-coordinates used for coordinate functions N_i^e, y direction.
CA	Scaling factor for coefficients (= .5).
CB	Scaling factor for coefficients (= SCALE × SCALE/ 6).
CJ,CK,C1	Differences in nodal coordinate used for coordinate functions N_i^e , x direction.
CS	Cosine of θ.
DIST	Length of element side on head-dependent (Cauchy-type) or specified-flux boundary.
ICA,ICB,ICC	Index to the storage location in A of capacitance term for nodes of NA, NB, and NC, respectively.
IEL	Element number.
IW	Length of storage in A allocated to each node.
IZ	Zone counter.
IZN	Vector containing zone numbers for each element.
KNT	Index for vector IZN, counter for elements.

Subroutine FMCO (continued)

Variable	Definition
KZ	Zone number.
M	Index to the storage location in A of capacitance term for node NA.
MI	Index for JPT.
MBM1	Number of storage locations in JPT that is allocated for each node.
NA,NB,NC	Node numbers identifying an element.
NE	Index to node numbers for combined-element input.
NDC	Index for node numbers.
NDID(I)	Node numbers for each element, I=1,4.
ND	Number of elements in zone KZ.
NT	Index for locating nodes in an element.
NTE	Element index.
QB	qB term or specified-flux boundary [length^2/time].
QD	Unit rate of areally distributed flow [length/time].
RJ,RK,RL	Factors used in formulating coefficients for cartesian or radial coordinates.
R(1), R(2), R(3)	Factors used to convert from cartesian to radial coordinate system.
SN	Sine of θ.
STR	Storage coefficient [dimensionless], specific yield [dimensionless], or specific storage [length^{-1}].

Subroutine FMCO (continued)

Variable	Definition
TESJ,TESK,TESL	Nodal coefficient $(1/3)S^e\Delta^e$ for aquifer storage.
TEQ	Nodal coefficient $(1/3)W^e\Delta^e$ for areally distributed flow.
TFL(1),TFL(2),TFL(3)	$(T_{\bar{x}\bar{x}}/4\Delta^e)\,\bar{b}_i^e\bar{b}_j^e+(T_{\bar{y}\bar{y}}/4\Delta^e)\bar{c}_i^e\bar{c}_j^e$ for transmissivity "link" between nodes i and j, $i \neq j = j, k, l$.
TMPA	Difference in x coordinates for nodes on head-dependent (Cauchy-type) or specified-flux boundary, the length $L_{kl}/2$ for node K on boundary, unrotated x coordinate, and centroidal radius r.
TMPB	Difference in y coordinates for nodes on head-dependent (Cauchy-type) or specified-flux boundary, the length $L_{kl}/2$ for node L on boundary, unrotated y coordinate.
VLC	Hydraulic conductance, R, for steady vertical leakage [length $^{-1}$].
XL	Local $\bar{x}$ coordinate [length].
XNA,XNB,XNC	Local $\bar{x}$ coordinates for nodes NA, NB, NC, respectively [length].
XTR	Transmissivity in the local $\bar{x}$ direction [length^2/time].
YL	Local $\bar{y}$ coordinate [length].
YNA,YNB,YNC	Local y coordinates for NA, NB, NC, respectively [length].
YTR	Transmissivity in the local $\bar{y}$ direction [length^2/time].

Subroutine FMEQ

B	Right side of finite-element matrix equations.

Subroutine FMEQ (continued)

Variable	Definition
DT	Time-step size [time].
IW	Length of storage in A allocated to each node.
MBM1	Index to pointer vector JPT; defines maximum amount of storage that is allocated to JPT for each node.
NC	Index to the storage location in A where the capacitance term is located for each node.
ND	Index to the storage location in A for the transmissivity terms of each node.
NME	Index to the location of the main diagonal of the upper-triangular matrix A stored in condensed-matrix form, for each node.
NP	Index to the pointer vector JPT.
NVL	Index to storage location in program vector A for steady-vertical-leakage term for each node.
TMPA,TMPB	Steady-vertical-leakage terms and coefficients for head-dependent (Cauchy-type) and specified-flux boundaries.

Subroutine HCALC

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Subroutine MASBAL

DTM	The factor $1/(2/3)\Delta t$ for computing average volumetric rates from aquifer storage.
IW	Length of storage in A allocated to each node.

Subroutine MASBAL (continued)

Variable	Definition
MBM1	Maximum number of locations in JPT that is allocated for each node.
NC	Index to the storage location in A for capacitance (aquifer-storage) terms.
ND	Index to the storage location in A preceding the location where transmissivity terms are located, for each node.
NP	Index to the pointer vector JPT.
NVL	Index to storage location in A for the vertical-leakage coefficient.
TMPA	Temporary-storage variable that is used to compute nodal volumetric rates for aquifer storage, convective flow (involving transmissivity) and Cauchy-type boundaries.
TMPB	Temporary-storage variable that is used to compute nodal volumetic rates for steady vertical leakage and Cauchy-type boundaries.

Subroutine MASOUT

ISTP	Number of the current time step.

Subroutine PRTOA

L	Index for printing out values (VAL).
NO	The number of values that is printed out, such as NNDS.
NR	Number of rows of values.
VAL	The variable that is printed out in three columns, such as hydraulic head, H.

Subroutine PRTOB

Variable	Definition
L	Index for printing out values (VALA and VALB).
NO	The number of pairs of VALA and VALB that is printed out, such as NNDS.
NR	Number of rows of values.
VALA	The first of two values printed out by node in two columns per line, such as XG.
VALB	The second of two values printed out bu node in two columns per line, such as YG.

Subroutine PRTCBV

INS	Number of sides on a boundary-condition zone (linear or nonlinear).
INZ	Zone number for bondary condition.
J	Index to nodal flow rates for element sides.
JB	Index for beginning number of boundary nodes and flow rates in zone.
NBZ	Total number of boundary-condition zones.
NVLA	Node K on head-dependent (Cauchy-type) boundary (linear or nonlinear).
NVLB	Node L on head-dependent (Cauchy-type) boundary (linear or nonlinear).
NZ	Index for zone loop.
QNET	Net volumetric flow rate, positive for inflow, from head-dependent (Cauchy-type) boundary (linear or nonlinear) [length^3/time].

Subroutine PRTCBV (continued)

Variable	Definition
SMQI	Sum of volumetric inflow rates from head-dependent (Cauchy-type) boundary (linear or nonlinear) [length^3/time].
SMQO	Sum of volumetric outflow rates from head-dependent (Cauchy-type) boundary (linear or nonlinear) [length^3/time].
SMVI	Sum of inflow volumes from head-dependent (Cauchy-type) boundary (linear or nonlinear) [length^3].
SMVO	Sum of outflow volumes from head-dependent (Cauchy-type) boundary (linear or nonlinear) [length^3].
VALA	Nodal volumetric flow rate from head-dependent (Cauchy-type) boundary at node NVLA [length^3/time].
VALB	Nodal volumetric flow rate from head-dependent (Cauchy-type) boundary at node NVLB [length^3/time].
VNET	Net volume of water, positive for accumulation, derived from head-dependent (Cauchy-type) boundaries [length^3].

Transient Leakage

Subroutine CBADEQ

L	Index to main diagonal and right side of matrix equation.
NME	Index to main diagonal of reduced matrix, A, stored in condensed-matrix form.

Subroutine CBADWT

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Subroutine CBCHG

Variable	Definition
HRJ	Value of new source-layer head [length].
HR(J)	Average source-layer head [length].
J	Node number where source-layer head is changed.
WF	Galerkin-weighting factor (=2/3) used to compute average source-layer head.

Subroutine CBFMCO

L	Zone number where transient leakage is simulated.
NA,NB,NC	Node numbers in an element.
NBE	Beginning (lowest) element number in zone L.
NDC	Index to ND for locating nodes in element.
NE	Counter used in computing transient leakage coefficients from combined-element input.
NEND	Ending (highest) element number in zone L.
NO	Number of elements contained in zone L.
NTE	Index to AR for element area.
NVL	Index to storage location in program vector A for hydraulic conductance term, C_{Ri} of equation (205) in Cooley (I992).
SPST	Specific storage [length $^{-1}$] of confining bed in zone L.
TESA	Effective specific storage $S_s^{'e}\Delta^e$ for element e.
TEVC	Effective vertical hydraulic conductivity $(1/3)K_{zz}^{'e}\Delta^e$ for element e.
VCON	Vertical hydraulic conductivity [length/time] of confining bed in zone L.

Subroutine CBFMEQ

Variable	Definition
DTD	Dimensionless time step $\Delta t_{n+1} \gamma_i$.
L	Index for storing main-diagonal and right-side coefficients for transient leakage, by node.
NQ	Index to transient-leakage, five terms per node.
NT	Counter for number of terms used to approximate either M_1 (NT = 3) or M_2 (NT = 2).
QOM1	The transient-leakage flow from the previous time step, given by $P_{hi,n}$ of equation (199) in Cooley (1992).
QOM2	The transient-lekage flow from the previous time step, given by $P_{Hi,n}$ in equation (200) in Cooley (1992).
SM1	$M_1(\Delta t_D)$, given by equation (188) in Cooley (1992).
TMPA	(1) Exponent $-\alpha_m \Delta t_D$ used to compute M_1.
	(2) Exponent $-\beta_m \Delta t_D$ used to compute M_2.
XP	(1) The term exp $(-\alpha_m \Delta t_D)$ used for approximating M_1.
	(2) The term exp $(-\beta_m \Delta t_D)$ used for approximating M_2.

Subroutine CBHRXT

DHR	Change in source-layer head, $H_{i,n+1} - H_{i,n}$ [length], for next time step
HR	Source-layer head, $H_{i,n+1}$ [length], at end of time step.
TMPA	Factor (= 1/3) for computing value of source-layer head at end of time step.

Subroutine CBINIT

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Subroutine CBTQC

Variable	Definition
DTM	1/(2/3) Δt_{n+1} used for coefficient formulation.
NQ	Index to transient-leakage terms CBTQ, five terms per node.
DTD	Dimensionless time step, Δt_D.
XP	The term exp $(-\alpha_m \Delta t_D)$.
TMPA	(1) The exponent $-\alpha_m \Delta t_D$. (2) $(\hat{h}_{i,n+1} - \hat{h}_{i,n})/\Delta t_D$, multiplies M_1 for computing transient-leakage term $I_{mi,\, n+1}$.
CBTQ	Part of transient-leakage flux from head changes during current time step [length/time].
TMPA	(1) Volumetric rate of accumulation of water in aquifer storage [length^3/time]. (2) Volumetric flow rate along a transmissivity "link" between nodes i and j on an element side [length^3/time] $i \neq j = k,l,m$.
TMPB	Volumetric flow rate from steady and transient leakage from a confining bed [length^3/time].
VLQI	Total volumetric rate of recharge from confining bed to aquifer caused by steady and transient leakage [length^3/time].
VLQO	Total volumetric rate of discharge from confining bed to aquifer caused by steady and transient leakage [length^3/time].

Subroutine MBWTCB

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines (the "WT" and "CB" subroutines and in subroutines MBALCB and MBALWT).

Table 19.-- *Variable names by subroutine for transient leakage*

Main program variable	Subroutine									
	CBADEQ	CBADWT	CBCHG	CBFMCO	CBFMEQ	CBHRXT	CBINIT	CBTQC	MBALCB	MBWTCB
DT					DT			DT	DT	DT
G							G			
G(IAA)	A	A		A	A				A	A
G(IACA)				AC	AC					
G(IALFA)				ALF	ALF			ALF		
G(IARA)				AR					DTK	DTK
G(IBA)	B	B						B	B	
G(IBCA)				BC	BC					
G(IBTA)				BTA	BTA					
G(ICHA)	CH	CH			CH				CH	CH
G(ICKA)									CFDK	CFDK
G(ICLA)									CFDL	CFDL
G(ICTQA)					CBTQ			CBTQ		
G(ICQA)	CBQ	CBQ			CBQ				CBQ	CBQ
G(IDHA)		DH								DH
G(IDHRA)			DHR		DHR	DHR				
G(IGMA)	GMA	GMA		GMA	GMA			GMA	GMA	GMA
G(IHA)									H	H
G(IHBA)								DHB	DHB	DHB
G(IHKA)									HK	HK
G(IHLA)									HL	HL
G(IHRA)			HR			HR			HR	HR
G(IJPA)									JPT	JPT
G(IKA)									KQB	KQB
G(ILA)									LQB	LQB
G(INA)	IN	IN			IN			IN	IN	IN
G(INDA)				ND						
G(IQA)									Q	Q
G(ISYA)										ASY
G(ITKA)										THK
G(ITPA)										TOP
G(IYGA)				WVCN					R	R
IACA							IACA			
IALFA							IALFA			
IBCA							IBCA			
IBTA							IBTA			
ICHA							ICHA			
ICQA							ICQA			
ICTQA							ICTQA			
IDHRA							IDHRA			
IGMA							IGMA			
ISTP			ISTP							
NCBZ				NCBZ			NCBZ			
TIME			TIME							

Changing Time-Step Size, Stresses, and Boundary Conditions

Subroutine COCHG

Variable	Definition
AREA	One-third the element area, $(1/3)\Delta^e$ [length^2].
DQ	Difference in volumetric flow rate [length^3/time] between old and new areally distributed flows.
HB	New value for specified head.
L	Zone number where changes to areally distributed flows are made.
M1	Length of storage needed for element areas in AR.
M2	Length of storage needed for node numbers in ND.
N	The number of stresses or controlling heads for boundary conditions that are changed; represents either nodes, element sides, or zones.
QNEW	(1) New value of point source or sink [length^3/time] at boundary J. (2) New value of unit areally distributed flow rate [length^3/time].
QOLD	(1) Old value of point source or sink at node J [length^3/time]. (2) Old value of unit areally distributed flow rate [length/time]. (3) Old value of specified-head part of Cauchy-type boundary, q_B [length^2/time] or q_b/α [length] if $\alpha \neq 0$.

Subroutine COCHG (continued)

TMPA	Total change to areally distributed inflows [length^3/time].
TMPB	Total change to areally distributed outflows [length^3/time].
WF	Weighting factor used to compute average head change at specified-head boundaries.

Subroutine NXTPD

Variable	Definition
NSTEPS	Counter equal to the number of time steps in the new stress period if either time-step sizes or the number of time steps change from the previous stress period.
NTMP	Time-step indicator; the number of time steps in the initial stress period and, either the number of time steps in the new stress period, or zero, if previous values are used.

Table 20.-- *Variable names by subroutine for changing time-step size, stresses, and boundary conditions*

Main program variable	Subroutine COCHG	Subroutine NXTPD
G(IALA)	ALFH	
G(IARA)	AR	
G(ICKA)	CFDK	
G(ICLA)	CFDL	
G(IDTA)		DELT
G(IHA)	H	
G(IHBA)	DHB	
G(IHKA)	HK	
G(IHLA)	HL	
G(IHRA)	HR	
G(IKA)	KQB	
G(ILA)	LQB	
G(INA)	IN	
G(INDA)	ND	
G(IQA)	Q	
G(IQBA)	QBND	
ISTP	ISTP	
JPER		JPER
TIME	TIME	

Nonlinear Versions

Water-table (unconfined) conditions

Subroutine FMECWT

Variable	Definition
C1,C2,C3	Galerkin weighting factors.
DHP	Predicted value of the total head change over the time step.
HO	Aquifer head at the beginning of the time step.
HP	Predicted aquifer head at the end of the time step.
THKI	Temporary-storage term for aquifer thickness at node I.
THKL	Temporary storage term for aquifer thickness at node L.
THKP	Negative aquifer thickness predicted to occur at a dry node.
TMPA	(1) Temporary-storage term for capcitance coefficient $(1/(2/3)\Delta t_{n+1})\ \underline{\underline{C}}$ of matrix equation (4). (2) Transmissivity term in $\tilde{G}_{ij}^{*}$ of matrix equation (4).
TMPB	Transmissivity terms in $(\underline{\underline{\tilde{G}}}^{*} - \underline{\underline{\bar{G}}}^{*})\ \underline{\bar{\delta}}^{*} - \underline{\underline{\bar{G}}}^{*}\ \underline{\bar{h}}^{*}$ for the right side of matrix equation (4).

Subroutine FMEPWP

THKI	Aquifer thickness for node I.
THKL	Aquifer thickness for node L.
TMPA	Transmissivity term given by equation (77) in Cooley (1992).
TMPB	Transmissivity term, predictor equation (3) right side.

Subroutine HCALWT

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Subroutine MBALWT

Variable	Definition
A(NC)	Effective storage coefficient resulting from integrations of equations (93) and (94) in Cooley (1992) for conversion between confined and unconfined conditions.
C	Galerkin-weighting factor for transmissivity formulation, equal to 1/16.
DHC	Total head change over the time step [length].
HO	Aquifer head, $\hat{h}_{i,n}$ [length], at beginning of time step.
HC	Aquifer head, $\hat{h}_{i,n+1}$ [length], at end of time step.
THET	Estimate of conversion point, θ_i, of equation (96) in Cooley (l992).
TMPA	(1) Volumetric rate of accumulation of water from aquifer storage [length^3/time]. (2) Transmissivity terms $\bar{\bar{G}}\,\bar{h} + (\tilde{\bar{G}} - \bar{\bar{G}})\,\bar{\delta}$ [length^3/time] equation (83) in Cooley (1992). (3) Volumetric-flow rates from Cauchy-type boundaries; first represents $(1/2)(\alpha L)_{kl}(h_{Bk}-\bar{h}_k)$, then $(1/2)(\bar{q}_B L)ij + (1/2)(\alpha L)_{kl}(h_{Bk}-\bar{h}_k)$,

Subroutine MBALWT (continued)

Variable	Definition
	for $\alpha \neq 0$ or, $(1/2)(\bar{q}_B L)ij$ for $\alpha = 0$ and $q_B \neq 0$,
	of equation (64) in Cooley (1992) [length^3/time].
TMPB	(1) Volumetric flow rate for confining-bed leakage
	$(1/3)R^e\Delta^e(H_i - \hat{h}_i)$.
TMPB	(2) Volumetric-flow rates from Cauchy-type
	boundaries; first represents $(1/2)(\alpha L)_{kl}(h_{Bl}-\bar{h}_l)$,
	then $(1/2)(\bar{q}_B L)ij + (1/2)(\alpha L)_{kl}(h_{Bl}-\bar{h}_l)$,
	for $\alpha \neq 0$; or, $(1/2)(\bar{q}_B L)ij$ for $\alpha = 0$ and $q_B \neq 0$,
	of equation (64) in Cooley (l992).

Subroutine WTCCHK

ISC	Indicator for unpredicted conversion or nonconversion of aquifer storage.

Subroutine WTFMCO

SY	Specific yield for aquifer-property zone [dimensionless].
TESY	Capacitance coefficient, C^e_{ii}, of equation (36) in Cooley (1992).

Subroutine WTINIT

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Table 21.-- *Variable Names by Subroutine for Water-table (Unconfined) Conditions*

Main program variable	Subroutine									
	FMECWT	FMEPWT	HCALWT	MBALWT	WTCCHK	WTFMCO	WTINIT	CBTQC	MBALCB	MBWTCB
DT	DT	DT		DT						
G							G			
G(IAA)	A	A		A						
G(IARA)	AD	AD		DTK		AR				
G(IBA)	B	B	B		B					
G(ICKA)	CFDK	CFDK		CFDK						
G(ICLA)	CFDL	CFDL		CFDL						
G(IDHA)	DH		DH	DH	DH					
G(IHA)	H	H	H	H	H					
G(IHBA)		DHB		DHB						
G(IHKA)	HK	HK		HK						
G(IHLA)	HL	HL		HL						
G(IHRA)	HR	HR		HR						
G(IJPA)	JPT	JPT							JPT	JPT
G(IKA)	KQB	KQB		KQB						
G(ILA)	LQB	LQB		LQB						
G(INA)	IN	IN	IN	IN	IN					
G(INDA)						ND				
G(IQA)	Q	Q		Q					Q	Q
G(IQBA)				QBND						
G(ISYA)	ASY	ASY		ASY		ASY				ASY
G(ITKA)	THK	THK		THK		THK				THK
G(ITPA)	TOP	TOP			TOP	TOP				TOP
G(IYGA)	DTK			R						
IDHA							IDHA			
ISC					ISC					
ISYA							ISYA			
ITKA							ITKA			
ITPA							ITPA			

Head-dependent (Cauchy-type) flux and point sinks

Subroutine GNBAL

Variable	Definition
CA,CB	Galerkin-weighting factors, equal to 1/3 and 2/3, respectively.
DHC	Total head change for time step; $\hat{h}_{i,n+1} - \hat{h}_{i,n}$ [length].
DHZ	Altitude difference, $\hat{h}_{i,n} - z_{ri}$, for nonlinear Cauchy-type boundaries; $z_{pi} - \hat{h}_{i,n}$ for nonlinear point sinks [length].
DRZ	Altitude difference, $\hat{h}_{ri} - z_{ri}$ [length].
HC	Hydraulic head, $\hat{h}_{i,n+1}$ [length], at end of time step.
HO	Hydraulic head, $\hat{h}_{i,n}$ [length], at beginning of time step.
HR	Controlling heads, HRK and HRL [length].
IP	Index to GC for nonlinear-point sinks.
NL	Boundary nodes, KR and LR.
PHC	Term $(1 - \phi_i)$ used for leakage expression in case 3.
TMPA	Head or altitude difference that multiplies coefficient C_{ri} for head-dependent (Cauchy-type) boundaries and C_{pi} for point sinks [length].
TMPB	Volumetric flow rate for head-dependent (Cauchy-type point sinks [length^3/time].
ZPI	Controlling altitude, ZP [length], for point sinks
ZR	Controlling altitude, ZR [length], for head-dependent (Cauchy-type) boundaries.

Subroutine GNBLSS

Variable	Definition
HL	Aquifer head on final iteration, $\hat{h}_i^{\ell+1}$ [length].
HR	Controlling head, h_{ri} [length].
QR	Nodal volumetric flow rate [length^3/time].
TMPA	Head or altitude difference that multiplies coefficient C_{ri} for head-dependent (Cauchy-type) boundaries and C_{pi} for point sink [length].
TMPB	Volumetric flow rate from head-dependent (Cauchy-type) boundaries and point sinks [length^3/time].

Subroutine GNCHG

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in subroutines GNBAL and GNBLSS.

Subroutine GNCORR

DHP	Predicted-aquifer head, $\hat{h}_{i,n+1}$ [length].
N	Index to $\underline{A}$ for the main diagonal of nodes k and l on the nonlinear boundary.
TMPA	Head or altitude difference that multiplies C_{ri} for head-dependent (Cauchy-type) boundaries and C_{pi} for point sinks [length].

Subroutine GNFMCO

CA	Factor for scaling lengths of boundary sides.
DIST	Length of boundary side.
GC	α term for nonlinear head-dependent (Cauchy-type boundary [length^2/time].

Subroutine GNFMCO (continued)

Variable	Definition
GCP	α term for nonlinear point sinks [length^2/time].
IPNC	Indicator variable to suppress printout of input for nonlinear head-dependent (Cauchy-type) boundaries.
IPNP	Indicator variable to suppress printout of input for nonlinear point sinks.
TMPA	Difference in x coordinates between nodes on boundary side.
TMPB	Difference in y coordinates between nodes on boundary side.

Subroutine GNINIT

NBNC	Number of nonlinear, head-dependent (Cauchy-type) boundaries.
NLCZ	Number of zones for nonlinear head-dependent (Cauchy-type) boundaries.
NPNB	Number of nonlinear, head-dependent point sinks.

Subroutine GNINIT

Variables that are used in this subroutine have been defined previously as variables in the main program and in storage vector G.

Subroutine GNPRED

IP	Index for α term for nonlinear point sinks.
N	Index to main-diagonal location of reduced matrix stored in condensed-matrix form.
TMPA	Head or altitude difference that multiplies coefficient C_{ri} for head-dependent (Cauchy-type) boundaries and C_{pi} for point sinks [length].

Table 22.-- *Variable names by subroutine for nonlinear head-dependent (Cauchy-type) boundaries*

Main program variable	Subroutine									
	GNBAL	GNBLSS	GNCHG	GNCORR	GNFMCO	GNINIT	GNPRED	CBTQC	MBALCB	MBWTCB
DT	DT	DT								
G						G				
G(IAA)				A			A			
G(IARA)	VQK	VQK								
G(IBA)				B			B			
G(IDHA)	DH			DH						
G(IGCA)	GC	GC		GC	GC		GC			
G(IHA)	H	H		H			H			
G(IHBA)	DHB									
G(IHRK)	HRK	HRK	HRK	HRK	HRK		HRK			
G(IHRL)	HRL	HRL	HRL	HRL	HRL		HRL			
G(IKPA)	KP	KP		KP	KP		KP			
G(IKRA)	KR	KR	KR	KR	KR		KR			
G(ILRA)	LR	LR	LR	LR	LR		LR			
G(INA)	IN	IN		IN			IN			
G(INSA)					INLS					
G(INZA)					INLZ					
G(IXGA)	VQL	VQL			XG					
G(IYGA)	R	R			YG					
G(IZPA)	ZP	ZP		ZP	ZP		ZP			
G(IZRK)	ZRK	ZRK		ZRK	ZRK		ZRK			
G(IZRL)	ZRL	ZRL		ZRL	ZRL		ZRL			
IGCA						IGCA				
IHRK						IHRK				
IHRL						IHRL				
IKPA						IKPA				
IKRA						IKRA				
ILRA						ILRA				
INSA						INSA				
INZA						INZA				
ISTP			ISTP							
IZPA						IZPA				
IZRK						IZRK				
IZRL						IZRL				
NBNC	NBNC	NBNC		NBNC	NBNC	NBNC	NBNC			
NLCZ						NLCZ				
NPNB	NPNB	NPNB		NPNB	NPNB	NPNB	NPNB			
TIME			TIME							

Steady vertical leakage

Subroutine VNBAL

Variable	Definition
CA,CB	Galerkin-weighting factors, equal to 1/3 and 2/3 respectively.
DAT	Altitude difference, $\overline{H}_{ai} - z_{ti}$ [length].
DEH	Altitude difference, $z_{ei} - \hat{h}_{i,n}$ [length].
DET	Altitude difference, $z_{ei} - z_{ti}$ [length].
DHC	Total-head change for time step; $\hat{h}_{i,n+1} - \hat{h}_{i,n}$ [length].
DHT	Altitude difference, $\hat{h}_{i,n} - z_{ti}$ [length].
DMHT	$(z_{ei} - \hat{h}_{i,n}) - (\hat{h}_{i,n} - z_{ti})$ used to compute the combination of changeover points, ϕ_{ti} and ϕ_{ei}, for leakage cases 8 and 9.
EI	Leakage coefficient, C_{ai} or C_{ei} of equations (119) and (132) respectively in Cooley (1992).
HA	Average value of controlling head, $\overline{H}_{ai}$, [length] for steady-vertical-leakage function.
HC	Aquifer head at end of time step; $\hat{h}_{i,n+1}$ [length].
HO	Aquifer head at beginning of time step; $\hat{h}_{i,n}$ [length].
PHC	Partial formulation of simplified ϕ terms, multiplies leakage coefficient for cases 2 and 6, discharge-only functions, and case 3, steady vertical-leakage function.
PHCF	Partial formulation of simplified ϕ terms, multiplies leakage coefficient for cases 8 and 9, discharge-only function.
PHE	Partial formulation of simplified ϕ terms multiplies leakage coefficient for case 5, discharge-only function.

Subroutine VNBAL (continued)

Variable	Definition
PHI	The changeover point, $-\phi_i$, of equation (115) in Cooley (1992) applied to steady vertical leakage.
PHT	Partial formulation of simplified ϕ terms, multiplies leakage coefficient for case 3, discharge-only function.
TMPA	Formulation of simplified ϕ terms and head or altitude differences, multiplies leakage coefficient to yield volumetric flow rate.
TMPB	Volumetric flow rate for nonlinear steady-leakage functions [length^3/time].
ZE	Extinction depth, z_{ei}, for discharge only function [length].
ZT	Altitude of aquifer top, z_{ti}, for nonlinear steady-leakage [length].

Subroutine VNBLSS

HI	Aquifer head from final iteration, $\hat{h}_i^{\ell+1}$ [length].
TMPA	(1) Altitude difference, $z_{ei} - z_{ti}$ for case1, or $z_{ei} - \hat{h}_i^{\ell+1}$ for case 4, that multiplies leakage coefficient for discharge-only function [length]. (2) Altitude difference, $H_{ai} - \hat{h}_i^{\ell+1}$ for case 1, or $H_{ai} - z_{ti}$, for case 4, that multiplies leakage coefficient for steady vertical-leakage function [length].
TMPB	Volumetric flow rate from nonlinear leakage functions [length^3/time].

Subroutine VNCHG

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in subroutines VNBAL and VNBLSS.

CA,CB	Galerkin-weighting factors equal to 1/3 and 2/3 respectively.
DHP	Predicted head change over the time step.
HA	Average controlling head $\overline{H}_{ai}$ [length] for steady vertical-leakage functions.
HO	Aquifer head, $\hat{h}_{i,n}$ [length] at beginning of time step.
HP	Predicted-aquifer head at the end of time step, $\hat{h}_{i,n+1}$ [length].
NME	Index to A for the main-diagonal location of nodes that simulate nonlinear-steady leakage.
PHE	Terms involving simplified ϕ terms that multiply the leakage coefficient to form the main-diagonal term, equal to $1 - \phi_{t\,i}$ for case 2 and $1 - \phi_{e\,i}$ for case 6, discharge-only functions, and equal to $1 - \phi_i$ for case 3, steady vertical-leakage functions.
TMPA	Terms involving head and altitude differences and/or the simplified ϕ terms that multiply the leakage coefficient EI to form the right side term for nonlinear steady leakage, by case.

Subroutine VNFMCO

EC	Leakage coefficient (1/3) $R_a^e\Delta^e$ or(1/3)$R_e^e\Delta^e$ of equations (119) and (132) respectively in Cooley (1992).
L	Zone number where nonlinear-steady leakage is simulated.
NA,NB,NC	Node numbers in an element.
NBE	Beginning (lowest) element number in zone L.

Subroutine VNFMCO (continued)

Variable	Definition
NDC	Index to ND for locating nodes in element.
NE	Counter used in computing leakage coefficient E from combined-element input.
NEND	Ending (highest) element number in zone L.
NO	Number of elements in zone L.
NTE	Index variable for element areas.
VNCF	Conductance terms, R_a^e or R_e^e for steady vertical leakage [time $^{-1}$].

Subroutine VNINIT

NVNZ	Number of zones for nonlinear, steady vertical leakage.

Subroutine VNPRED

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Table 23.-- *Variable names by subroutine for nonlinear steady vertical leakage*

Main program variable	Subroutine									
	VNBAL	VNBLSS	VNCHG	VNCORR	VNFMCO	VNINIT	VNPRED	CBTQC	MBALCB	MBWTCB
DT	DT	DT								
G						G				
G(IAA)				A			A			
G(IARA)					AR					
G(IBA)				B			B			
G(IDHA)	DH			DH						
G(IECA)	E	E		E	E		E			
G(IHA)	H	H		H	H		H			
G(IHBA)	DHB									
G(IHSA)	HS	HS	HS	HS	HS		HS			
G(INA)	IN	IN		IN			IN			
G(INDA)					ND					
G(ITPA)	TOP	TOP		TOP			TOP			
G(IYGA)	R	R								
IECA						IECA				
IHSA						IHSA				
ISTP			ISTP							
NVNZ					NBNC	NBNC				
TIME			TIME							

Steady-state conditions

Subroutine SWBDMP

Variable	Definition
B	(1) Computed displacement, $\underline{s}_{\ell}$, from subroutines BAND and MICCG.
	(2) Damped displacements, $\rho_{\ell}\,\underline{s}_{\ell}$, of equation (239) in Cooley (1992).
DSPA	Absolute value of maximum displacement, $\lvert e_{\ell} \rvert$, of equation (243) in Cooley (1992).
DSP	Maximum displacement, e_{ℓ}, from the current iteration.
DSPO	Initial value of maximum displacement, e_{ℓ}, or the maximum value from the previous iteration.
RP	Estimate of the damping parameter ρ^*.
SPR	Ratio of damped to undamped displacements, given by equation (241) in Cooley (1992).

Subroutine SWFMCO

IW	Number of storage locations assigned to each node for storing components of reduced matrix A.
IPTK	Indicator variable to supress printout of aquifer thickness at each node.
IPTP	Indicator variable to suppress printout of altitude of aquifer top.
MBM1	Number of storage locations allocated to each node for storing elements of JPT.
ND	Index to A for locating transmissivity terms.
NP	Index to JPT.

Subroutine SWINIT

Variable	Definition
DSMX	Maximum allowable displacement [length].
DSP	Maximum computed displacement [length].
NITSW	Maximum number of water-table iterations.
TOLSW	Closure tolerance for steady state, ε_s, of equation (240) in Cooley (1992).

Subroutine SWTHK

HO	Aquifer head from previous iteration, $\hat{h}^{\ell}$ [length].
THKI	Temporary-storage variable for aquifer thickness at node I.
THKK	Temporary-storage variable for aquifer thickness at node K.

Subroutine TKOUT

Variables that are contained in this subroutine have been defined in previous sections that give definitions of program variables in the general-storage vector G, in Fortran COMMON statements, and in other subroutines.

Table 24.-- *Variable names by subroutine for nonlinear steady-state conditions*

Main program variable	Subroutine				
	SWBDMP	SWFMCO	SWINIT	SWTHK	TKOUT
G(IAA)		A		A	
G(IBA)	B			B	
G(IHA)				H	
G(IJPA)		JPT		JPT	
G(INA)		IN		IN	
G(ITKA		THK		THK	THK
G(ITPA)		TOP		TOP	
G(IQA)				Q	
DSMX	DSMX				
DSP	DSP		DSP		
DSPA	DSPA			DSPA	
DSPO	DSPO				
G			G		
RP	RP		RP		
TOLSW			TOLSW	TOLSW	
ITER	ITER			ITER	
ITKA			ITKA		
ITPA			ITPA		
NITSW			NITSW		

Direct-Solution Method

Subroutine BAND

Variable	Definition
A	Element of upper triangular matrix A.
B	The intermediate vector y when it appears in the DO 150 and DO 160 loops; the computed values of head difference when used in the DO 170 and DO 180 loops.
C	A factor u_{ij}/α_{ii} used in upper triangularization of matrix A when it appears in the DO 140 loop. A coefficient used to compute y when it appears in the DO 160 loop.
C1	The main diagonal in D, $1/\alpha_{ii}$ where α_{ii} is the main diagonal in matrix u.
IB	Index to the off diagonal of the expanded A matrix.
ID	Index for the main diagonal of the expanded A matrix.
IRBW	Number of elements in semi-bandwidth of A that are factored, including the main diagonal.
MR	Amount of storage required for composing upper-triangular matrix A.
NP	Number of terms in JPT.
NR	Amount of storage required for composing upper-triangular matrix A.

Subroutine INITB

NA	Length of vector storage associated with program vector A for storing terms used to form coefficients of matrix equations, computed as (MBWC+1) × NNDS.
NB	Length of vector storage needed to store element areas in program vector AR, computed as 2 × NELS.

Subroutine INITB (continued)

Variable	Definition
NBCZ	Number of head-dependent (Cauchy-type) boundary zones.
NC	Amount of computer storage needed for reduced-matrix A, factored into an upper-triangular matrix computed as MBW × (NNDS-NHDS-MBW) + (MBW × (MBW+1))/2.
ND	Amount of storage needed for the pointer vector JPT computed as (MBWC-1) × NNDS.
NE	Amount of storage needed for the ND vector that stores node numbers of each element, computed as 4 × NELS.
NF	Amount of computer storage needed for right side of matrix equations (1) through (5) computed as the greater value of NNDS-NHDS or NELS.
TIME	Total simulation time.
TITLE	Title of simulation and description of scale change for length terms.

Subroutine SETB

IB, IE, IB1, IE1	Starting (IB and IB1) and ending (IE) locations in general storage vector G for writing terms to storage files.
IBND	Bandwidth of reduced-matrix A.
JB, JE	Indexes for combining incidences of two elements.
MAXND	Largest node number in element for computing reduced-matrix bandwidth.
MINND	Smallest node number in element for computing reduced-matrix bandwidth.
NDC	Index for node numbers in vector ND.

Subroutine SETB (continued)

Variable	Definition
NE	Counter for elements.
NTMP	Current value of reduced-matrix bandwidth.

Table 25.-- *Variable names by subroutine for direct-solution method*

Main program variable	Subroutine BAND	INITB	SETB
G		G	
G(IAA)	A		A
G(IATA)	AT		
G(IBA)	B		
G(IJPA)	JPT		
G(INA)	IN		IN
G(INDA)			ND
IBND	IBND		IBND
NBCZ		NBCZ	
TIME		TIME	
TITLE		TITLE	

Iterative, Conjugate-Gradient Method

Subroutine INITCG

Variable	Definition
NA	Length of vector storage associated with program vector A for storing terms that form coefficients of matrix equations, computed as (MBWC+1) × NNDS.
NB	Length of vector storage associated with vector AR for storing element areas, computed as 2 × NELS.
NBCZ	Number of head-dependent (Cauchy-type) boundary zones.
NC	Length of vector storage associated with upper-triangular matrix U resulting from incomplete factorization, computed as NA+(MBWC+3) × (NNDS-NHDS).
ND	Length of vector storage associated with pointer vector JPT, computed as (MBWC-1) × NNDS.
NE	Length of vector storage associated with vector ND for storing node numbers of each element, computed as 4 × NELS.
NF	Length of vector storage for B vector which stores computed head changes, computed as NNDS-NHDS.
NG	Amount of storage (= NF) used to allocate space for X, P, and R in subroutine MICCG.
TIME	Total simulation time.
TITLE	Title of simulation and description of scale change for length terms.
TOL	Closure tolerance for MICCG solution.

Subroutine MICCG

AF(ID+J)	Term $\tilde{U}$ ij = $\tilde{U}$ij/ $\tilde{\alpha}_{ii}$, of equation (282) in Cooley (1992).

Subroutine MICCG (continued)

Variable	Definition
AF(KD)	Main-diagonal term $\tilde{\alpha}_{ii}$ of $\tilde{\underline{U}}$ computed by equation (278) in Cooley (1992).
AF(KD+L)	Off-diagonal term $\tilde{u}_{ij}$, of $\tilde{\underline{U}}$ computed by equation (274) in Cooley (1992).
AF(LD)	Main-diagonal term $\tilde{\alpha}_{ii}$, of $\tilde{\underline{U}}$ computed by equation (278) in Cooley (1992).
B(I)	Scaled-head change for iteration [length].
C	Factor $\tilde{u}_{ki}/\tilde{a}_{kk}$ for formulating coefficients of $\tilde{\underline{U}}$.
ID	Index to main-diagonal element of $\tilde{\underline{U}}$, α_{ii}.
IT	Iteration number.
KNT	Counter for iteration number.
NP	Index for pointer vector JPT.
PIV	Factor $1/\tilde{\alpha}_{kk}$ for formulating coefficients of $\tilde{\underline{U}}$.
PMAX	Largest scaled displacement for iteration.
PX	$\underline{s}_k^T \underline{r}_k$ of equation (270) in Cooley (1992).
P(I)	Vectors p of equation (271) in Cooley (1992).
RMAX	Largest scaled residual, $\lvert r^k \rvert / \alpha_{ii}$, of equation (289) in Cooley (1992).
RX	$\underline{s}_k^T \underline{r}_k$ of equation (271) in Cooley (1992).
S	β_k of equation (271) in Cooley (1992).

Subroutine MICCG (continued)

Variable	Definition
TMPA	Temporary variable used to compute f_{ij} of equation (277) in Cooley (1992) for row-sum agreement, scaled displacement W P(I), and scaled residual $\lvert r_i^k \rvert / a_{ii}$ for row-sum agreement.
W	Scaling factor relating flow-balance residuals from incomplete factorization of $\underline{A}$ to those computed with $\underline{A}$.
X(K)	y_i of equation (283) in Cooley (1992).
X(M)	y_i and s of equation (283) in Cooley (1992).
X(N)	y_i of equation (283) in Cooley (1992).

Subroutine SETG

IB	Starting location in general-storage vector G for element area.
IB1	Starting location in general-storage vector G for element
IE	Ending location in general-storage vector G for element area.
IE1	Ending location in general-storage vector G for element incidences.

Table 26.-- *Variable names by subroutine for iterative, conjugate-gradient method*

Main program variable	Subroutine		
	INITCG	MICCG	SETCG
G	G		
G(IAA)		A	A
G(IAFA)		AF	
G(IBA)		B	
G(IJPA)		JPT	
G(INA)		IN	
G(IPA)		P	
G(IRA)		R	
G(IXA)		X	
IAFA	IAFA		
IPA	IPA		
IRA	IRA		
IXA	IXA		
NBCZ	NBCZ		
TIME	TIME		
TITLE	TITLE		
TOL	TOL	TOL	

```
      SUBROUTINE BAND(A,AT,B,JPT,IN,IBND)
C        SOLVES MATRIX EQUATION FOR HEAD CHANGE VECTOR, B, BY USING
C        TRIANGULAR DECOMPOSITION
      DIMENSION A(1),AT(1),B(1)
      DIMENSION JPT(1),IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
C *** DEFINE INDICES ***
      N=NNDS-NHDS
      MR=MBWC*N+1
      NR=IBND*(N-IBND)+(IBND*(IBND+1))/2
      IF(MR.GT.NR) NR=MR-1
      MBM1=MBWC-1
      NP=MBM1*NEQ
      NN=NEQ+1
      IBM1=IBND-1
C *** TRANSFORM CONDENSED COEFFICIENT MATRIX, A, INTO ROTATED
C     BAND-MATRIX, A ***
      DO 110 I=1,NEQ
      NP=NP-MBM1
      NN=NN-1
      K=IN(NN)
      IF(K.LT.0) GO TO 110
      IRBW=MIN0(IBND,N-K+1)
      DO 90 J=1,IRBW
   90 AT(J)=0.
      MR=MR-MBWC
      IF(A(MR).GT.1.E-20) GO TO 93
      A(MR)=1.E30
      B(K)=0.
   93 AT(1)=A(MR)
      DO 100 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 100,103,95
   95 AT(M-K+1)=A(MR+J)
  100 CONTINUE
  103 NR=NR-IRBW
      DO 105 J=1,IRBW
  105 A(NR+J)=AT(J)
  110 CONTINUE
C *** FACTOR A INTO PRODUCT OF UPPER TRIANGULAR, DIAGONAL, AND LOWER
C     TRIANGULAR MATRICES, ALL KEPT IN A ***
      NM1=N-1
      ID=NR+1
      DO 140 I=1,NM1
      C1=1./A(ID)
      LD=ID
      L=I
      IRBW=MIN0(IBM1,N-I)
      DO 130 J=1,IRBW
      L=L+1
      LD=LD+MIN0(IBND,N-L+2)
      IB=ID+J
```

```
            IF(A(IB)) 115,130,115
  115 C=A(IB)*C1
      LB=LD-1
      DO 120 K=J,IRBW
            LB=LB+1
      IF(A(ID+K)) 119,120,119
  119 A(LB)=A(LB)-C*A(ID+K)
  120 CONTINUE
      A(IB)=C
  130 CONTINUE
  140 ID=ID+IRBW+1
C *** REDUCE RIGHT-HAND SIDE, B ***
      ID=NR+1
      DO 160 I=1,NM1
      C=B(I)
      B(I)=C/A(ID)
      IRBW=MIN0(IBM1,N-I)
      DO 150 L=1,IRBW
      K=I+L
  150 B(K)=B(K)-A(ID+L)*C
  160 ID=ID+IRBW+1
C *** BACK-SOLVE FOR HEAD CHANGE, B ***
      B(N)=B(N)/A(ID)
      DO 180 I=1,NM1
      IRBW=MIN0(IBM1,I)
      ID=ID-IRBW-1
      L=N-I
      SUM=0.
      DO 170 J=1,IRBW
  170 SUM=SUM-A(ID+J)*B(L+J)
  180 B(L)=B(L)+SUM
      RETURN
      END
```

```
      SUBROUTINE CBADEQ(A,B,CH,CBQ,GMA,IN)
C        ADDS COEFFICIENTS FORMED IN CBFMEQ INTO MATRIX DIAGONAL ANI
C        RIGHT-HAND-SIDE VECTOR
      DIMENSION A(1),B(1),CH(1),CBQ(1),GMA(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      L=0
      DO 10 I=1,NEQ
      IF(GMA(I).LT.1.E-30) GO TO 10
      L=L+1
      K=IN(I)
      IF(K.LT.0) GO TO 10
      NME=MBWC*(K-1)+1
      A(NME)=A(NME)+CH(L)
      B(K)=B(K)+CBQ(L)
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CBADWT(DH,A,B,CH,CBQ,GMA,IN)
         ADDS COEFFICIENTS FORMED IN CBFMEQ INTO MATRIX DIAGONAL AND
         RIGHT-HAND-SIDE VECTOR FOR WATER-TABLE CORRECTOR STEP
      DIMENSION DH(1),A(1),B(1),CH(1),CBQ(1),GMA(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      L=0
      DO 10 I=1,NEQ
      IF(GMA(I).LT.1.E-30) GO TO 10
      L=L+1
      K=IN(I)
      IF(K.LT.0) GO TO 10
      NME=MBWC*(K-1)+1
      A(NME)=A(NME)+CH(L)
      B(K)=B(K)+CBQ(L)-CH(L)*DH(K)
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CBCHG(HR,DHR,TIME,ISTP)
C        READS AND COMPUTES NEW VALUES FOR SOURCE-BED HEAD FOR TRANS
C        LEAKAGE
      DIMENSION HR(1),DHR(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (2I5)
    4 FORMAT (1H1,10X,40HCHANGES IN VALUES OF SOURCE-BED HEAD FOR/1H
     1,2X,57HLEAKAGE INVOLVING TRANSIENT EFFECTS FROM AQUITARD STORAGE
     2/1H ,29H  BEGINNING ON TIME STEP NO. ,I4,4H AT ,G11.5
     3,11H TIME UNITS/1H ,35X,10HSOURCE-BED/1H ,19X,4HNODE,15X,4HHEAD)
    6 FORMAT (1H ,17X,I5,13X,G11.5)
    8 FORMAT (I5,F10.0)
C *** READ NUMBER OF CHANGES AND TIME STEP WHERE CHANGES ARE TO BE
C      MADE AGAIN ***
      READ(IIN,2) N,NCBCH
      WRITE(IOUT,4) ISTP,TIME
      WF=2./3.
      DO 30 I=1,N
C *** READ AND WRITE NEW VALUES ***
      READ(IIN,8) J,HRJ
      WRITE(IOUT,6) J,HRJ
C *** FORM HEAD-CHANGE AND NEW-HEAD VECTORS ***
      DHR(J)=HRJ-HR(J)
   30 HR(J)=WF*DHR(J)+HR(J)
      RETURN
      END
```

```
      SUBROUTINE CBFMCO(A,AR,GMA,ALF,AC,BTA,BC,WVCN,ND,NCBZ)
         FORMS COEFFICIENTS FOR TRANSIENT LEAKAGE
      DIMENSION A(1),AR(1),GMA(1),ALF(1),AC(1),BTA(1),BC(1),WVCN(1)
      DIMENSION ND(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (1H0,15X,40HPARAMETERS FOR TRANSIENT LEAKAGE BY ZONE/1H
     1,10X,5HFIRST,5X,6HNO. OF,4X,18HAQUITARD HYDRAULIC,3X
     2,17HAQUITARD SPECIFIC/1H ,6H  ZONE,3X,7HEL. NO.,3X,8HELEMENTS,6X
     3,12HCONDUCTIVITY,11X,7HSTORAGE)
    4 FORMAT (3I5,2F10.0)
    6 FORMAT (1H ,I5,4X,I5,5X,I5,10X,G11.5,8X,G11.5)
C *** DEFINE COEFFICIENTS FOR SERIES APPROXIMATIONS ***
      ALF(1)=13.656
      ALF(2)=436.53
      ALF(3)=49538.
      AC(1)=.26484
      AC(2)=.060019
      AC(3)=.008474
      BTA(1)=10.764
      BTA(2)=19.805
      BC(1)=-.25754
      BC(2)=.090873
      WRITE(IOUT,2)
C *** INITIALIZE NODAL EFFECTIVE AQUITARD HYDRAULIC CONDUCTIVITY ***
      DO 20 I=1,NNDS
   20 WVCN(I)=0.
C *** BEGIN ZONAL LOOP ***
      DO 50 I=1,NCBZ
C *** READ AND WRITE ZONAL DATA ***
      READ(IIN,4) L,NBE,NO,VCON,SPST
      WRITE(IOUT,6) L,NBE,NO,VCON,SPST
C *** BEGIN ELEMENT LOOP WITHIN ZONE ***
      NEND=NBE+NO-1
      NDC=4*(NBE-1)+1
      NTE=2*(NBE-1)
      DO 40 J=NBE,NEND
C *** BEGIN NODAL LOOP WITHIN ELEMENT ***
      NE=1
      IF(ND(NDC+3).GT.0) NE=2
      DO 30 IE=1,NE
C *** COMPUTE EFFECTIVE AQUITARD HYDRAULIC CONDUCTIVITY AND EFFECTIVE
C     AQUITARD SPECIFIC STORAGE ***
      NA=ND(NDC)
      NB=ND(NDC+IE)
      NC=ND(NDC+IE+1)
      TEVC=VCON*AR(NTE+IE)
      WVCN(NA)=WVCN(NA)+TEVC
      WVCN(NB)=WVCN(NB)+TEVC
      WVCN(NC)=WVCN(NC)+TEVC
      TESA=SPST*AR(NTE+IE)
      GMA(NA)=GMA(NA)+TESA
      GMA(NB)=GMA(NB)+TESA
      GMA(NC)=GMA(NC)+TESA
```

```
   30 CONTINUE
      NTE=NTE+2
   40 NDC=NDC+4
   50 CONTINUE
C *** COMPUTE GAMMA ***
      IW=MBWC+1
      NVL=2
      DO 60 I=1,NNDS
      IF(GMA(I).GT.1.E-30) GMA(I)=A(NVL)*A(NVL)/(GMA(I)*WVCN(I))
   60 NVL=NVL+IW
      RETURN
      END
```

```
      SUBROUTINE CBFMEQ(A,CH,CBQ,DHR,CBTQ,GMA,ALF,AC,BTA,BC,DT,IN)
C        FORMS COEFFICIENTS FOR TRANSIENT LEAKAGE
      DIMENSION A(1),CH(1),CBQ(1),DHR(1),CBTQ(1),GMA(1),ALF(1),AC(1)
     1,BTA(1),BC(1)
      DIMENSION IN(1)
      DIMENSION QOM1(3),QOM2(2)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      IW=MBWC+1
      NVL=2
      NQ=0
      L=0
      DO 50 I=1,NNDS
      IF(GMA(I).LT.1.E-30) GO TO 50
      DTD=GMA(I)*DT
C *** COMPUTE M1 SERIES AND EXPONENTIALLY REDUCE TRANSIENT-LEAKAGE
C     FLUX DUE TO TIME VARIANCE OF AQUIFER HEAD ***
      SM1=0.
      DO 30 NT=1,3
      XP=0.
      TMPA=ALF(NT)*DTD
      IF(TMPA.LT.50.) XP=EXP(-TMPA)
      SM1=SM1+AC(NT)*(1.-XP)
      NQ=NQ+1
      QOM1(NT)=CBTQ(NQ)
      CBTQ(NQ)=XP*CBTQ(NQ)
   30 CONTINUE
C *** COMPUTE NEW TRANSIENT-LEAKAGE FLUX DUE TO TIME VARIANCE OF
C     SOURCE-BED HEAD ***
      DO 40 NT=1,2
      XP=0.
      TMPA=BTA(NT)*DTD
      IF(TMPA.LT.50.) XP=EXP(-TMPA)
      NQ=NQ+1
      QOM2(NT)=CBTQ(NQ)
      CBTQ(NQ)=XP*CBTQ(NQ)+BC(NT)*(1.-XP)*DHR(I)/DTD
   40 CONTINUE
C *** COMPUTE COEFFICIENTS TO ADD INTO MATRIX DIAGONAL AND RIGHT-HAND-
C     SIDE VECTOR ***
      L=L+1
      CH(L)=SM1*A(NVL)/DTD
      CBQ(L)=(2.*(CBTQ(NQ-1)+CBTQ(NQ)-CBTQ(NQ-4)-CBTQ(NQ-3)-CBTQ(NQ-2))
     1+QOM2(1)+QOM2(2)-QOM1(1)-QOM1(2)-QOM1(3))*A(NVL)/3.
   50 NVL=NVL+IW
      RETURN
      END
```

```
      SUBROUTINE CBHRXT(HR,DHR)
C        EXTRAPOLATES SOURCE-BED HEADS AT ALL NODES FROM THE MEAN PO
C        IN THE TIME STEP TO THE END OF THE TIME STEP, THEN ZEROS TH
C        HEAD CHANGES
      DIMENSION HR(1),DHR(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      TMPA=1./3.
      DO 10 I=1,NNDS
      HR(I)=TMPA*DHR(I)+HR(I)
      DHR(I)=0.
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CBINIT(G,ICHA,ICQA,IDHRA,ICTQA,IGMA,IALFA,IACA,IBTA
     1,IBCA,NCBZ)
C        READS PROBLEM SPECIFICATION, DEFINES AND INITIALIZES VARIABLES
C        FOR TRANSIENT LEAKAGE ROUTINES
      DIMENSION G(1)
      COMMON/GDIM/ISUM
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (1H0,40HTRANSIENT LEAKAGE FROM AQUITARD STORAGE:/1H
     1,38HNOW G MUST BE DIMENSIONED TO AT LEAST ,I6)
    4 FORMAT (2I5)
    6 FORMAT (1H0,54HNO. OF AQUITARD PROPERTY ZONES (NCBZ).............
     1 = ,I5/
     $1H ,54HMAXIMUM NUMBER OF TRANSIENT LEAKAGE NODES (MCBN)... = ,I5)
C *** READ AND WRITE NUMBER OF AQUITARD PROPERTY ZONES AND TRANSIENT
C     LEAKAGE NODES ***
      READ(IIN,4) NCBZ,MCBN
      WRITE(IOUT,6) NCBZ,MCBN
C *** DEFINE ADDRESSES OF NEW ARRAYS WITHIN G ***
      ICHA=ISUM
      ISUM=ISUM+MCBN
      ICQA=ISUM
      ISUM=ISUM+MCBN
      IDHRA=ISUM
      ISUM=ISUM+NNDS
      ICTQA=ISUM
      ISUM=ISUM+5*MCBN
      IGMA=ISUM
      ISUM=ISUM+NNDS
      IALFA=ISUM
      ISUM=ISUM+3
      IACA=ISUM
      ISUM=ISUM+3
      IBTA=ISUM
      ISUM=ISUM+2
      IBCA=ISUM
      ISUM=ISUM+2
C *** WRITE NEW SIZE OF G ***
      WRITE(IOUT,2) ISUM
C *** INITIALIZE NEW ELEMENTS OF G ***
      DO 10 I=ICHA,ISUM
   10 G(I)=0.
      RETURN
      END
```

```
      SUBROUTINE CBTQC(B,DHB,CBTQ,GMA,ALF,AC,DT,IN)
C        COMPUTES NEW TRANSIENT-LEAKAGE FLUX DUE TO TIME VARIANCE OF
C        AQUIFER HEAD
      DIMENSION B(1),DHB(1),CBTQ(1),GMA(1),ALF(1),AC(1)
      DIMENSION IN(1)
      DIMENSION XP(3)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      DTM=3./(2.*DT)
      NQ=-4
      DO 30 I=1,NNDS
      IF(GMA(I).LT.1.E-30) GO TO 30
C *** COMPUTE EXPONENTIAL TERMS FOR M1 SERIES ***
      DTD=GMA(I)*DT
      DO 10 NT=1,3
      XP(NT)=0.
      TMPA=ALF(NT)*DTD
      IF(TMPA.LT.50.) XP(NT)=EXP(-TMPA)
   10 CONTINUE
C *** COMPUTE NEW TRANSIENT-LEAKAGE FLUX ***
      NQ=NQ+5
      K=IN(I)
      IF(K.LT.0) GO TO 20
      TMPA=DTM*B(K)/GMA(I)
      GO TO 25
   20 TMPA=DTM*DHB(-K)/GMA(I)
   25 CBTQ(NQ)=CBTQ(NQ)+AC(1)*(1.-XP(1))*TMPA
      CBTQ(NQ+1)=CBTQ(NQ+1)+AC(2)*(1.-XP(2))*TMPA
      CBTQ(NQ+2)=CBTQ(NQ+2)+AC(3)*(1.-XP(3))*TMPA
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE COCHG(Q,AR,H,HR,DHB,HK,HL,ALPH,QBND,CFDK,CFDL,TIME,KQB
     1,LQB,ND,IN,ISTP)
         READS AND COMPUTES NEW VALUES OF TIME-VARIANT QUANTITIES
      DIMENSION Q(1),AR(1),H(1),HR(1),DHB(1),HK(1),HL(1),ALPH(1),QBND(1)
     1,CFDK(1),CFDL(1)
      DIMENSION KQB(1),LQB(1),ND(1),IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** FORMAT LIST ***
    2 FORMAT (16I5)
    4 FORMAT (1H1,14X,32HCHANGES IN VALUES OF POINT FLOWS/1H
     1,29H  BEGINNING ON TIME STEP NO. ,I4,4H AT ,G11.5,11H TIME UNITS
     2/1H ,24X,3HOLD,11X,3HNEW/1H ,15X,15HNODE  DISCHARGE,5X
     3,9HDISCHARGE)
    6 FORMAT (I5,4F10.0)
    8 FORMAT (1H ,13X,I5,2(3X,G11.5))
   10 FORMAT (1H1,12X,38HCHANGES IN VALUES OF DISTRIBUTED FLOWS/1H
     1,29H  BEGINNING ON TIME STEP NO. ,I4,4H AT ,G11.5,11H TIME UNITS
     2/1H ,16X,4HBEG.,3X,3HNO.,3X,8HOLD UNIT,6X,8HNEW UNIT/1H ,10X
     3,27HZONE   EL.  ELS.  DISCHARGE,5X,9HDISCHARGE)
   12 FORMAT (3I5,2F10.0)
   14 FORMAT (1H ,7X,3(1X,I5),2(3X,G11.5))
   16 FORMAT (1H1,12X,37HCHANGES IN VALUES OF CAUCHY-TYPE DATA/1H
     1,29H  BEGINNING ON TIME STEP NO. ,I4,4H AT ,G11.5,11H TIME UNITS
     2/1H ,6X,8HEL. SIDE,3X,8HNEW UNIT,6X,8HEXTERNAL,6X,8HEXTERNAL
     3/1H ,9X,3HNO.,5X,9HDISCHARGE,6X,6HHEAD A,8X,6HHEAD B)
   18 FORMAT (1H ,6X,I5,2X,3(3X,G11.5))
   20 FORMAT (1H1,13X,35HCHANGES IN VALUES OF SPECIFIED HEAD/1H
     1,29H  BEGINNING ON TIME STEP NO. ,I4,4H AT ,G11.5,11H TIME UNITS
     2/1H ,19X,4HNODE,15X,4HHEAD)
   22 FORMAT (1H ,17X,I5,13X,G11.5)
   24 FORMAT (1H1,12X,36HCHANGES IN VALUES OF SOURCE-BED HEAD/1H
     1,29H  BEGINNING ON TIME-STEP NO. ,I4,4H AT ,G11.5,11H TIME UNITS
     2/1H ,19X,4HNODE,15X,4HHEAD)
C *** FOR POINT SOURCES AND SINKS: ***
      IF(NWCH.NE.ISTP) GO TO 70
C *** (1) READ NUMBER OF CHANGES AND TIME STEP WHERE CHANGES ARE TO BE
C     MADE AGAIN ***
      READ(IIN,2) N,NWCH
      WRITE(IOUT,4) ISTP,TIME
      DO 60 I=1,N
C *** (2) READ AND WRITE OLD AND NEW VALUES ***
      READ(IIN,6) J,QOLD,QNEW
      WRITE(IOUT,8) J,QOLD,QNEW
C *** (3) ADD VALUES INTO TOTAL INFLOW OR OUTFLOW PER UNIT TIME FOR
C     FLOW BALANCE ***
      IF(QOLD.GT.0.) GO TO 30
      WQO=WQO-QOLD
      GO TO 40
   30 WQI=WQI-QOLD
   40 IF(QNEW.GT.0.) GO TO 50
      WQO=WQO+QNEW
      GO TO 60
   50 WQI=WQI+QNEW
```

```
C *** (4) ADD VALUES INTO Q-VECTOR ***
   60 Q(J)=Q(J)+QNEW-QOLD
C *** FOR DISTRIBUTED RECHARGE OR DISCHARGE: ***
   70 IF(NQCH.NE.ISTP) GO TO 152
C *** (1) READ NUMBER OF CHANGES AND TIME STEP WHERE CHANGES ARE TO BE
C      MADE AGAIN ***
      READ(IIN,2) N,NQCH
      WRITE(IOUT,10) ISTP,TIME
C *** (2) READ ELEMENT AREAS AND NODE NUMBERS ***
      REWIND ITB
      M1=2*NELS
      M2=4*NELS
      READ(ITB) (AR(I),I=1,M1),(ND(I),I=1,M2)
      TMPA=0.
      TMPB=0.
      DO 150 I=1,N
C *** (3) READ AND WRITE OLD AND NEW VALUES ***
      READ(IIN,12) L,NBE,NO,QOLD,QNEW
      WRITE(IOUT,14) L,NBE,NO,QOLD,QNEW
C *** (4) BEGIN ELEMENT LOOP ***
      NEND=NBE+NO-1
      NDC=4*(NBE-1)+1
      NTE=2*(NBE-1)
      DO 140 J=NBE,NEND
      NE=1
      IF(ND(NDC+3).GT.0) NE=2
      DO 130 IE=1,NE
      AREA=AR(NTE+IE)
C *** (5) ADD VALUES INTO TOTAL INFLOW OR OUTFLOW PER UNIT TIME FOR
C      FLOW BALANCE ***
      IF(QOLD.GT.0.) GO TO 90
      TMPB=TMPB-QOLD*AREA
      GO TO 100
   90 TMPA=TMPA-QOLD*AREA
  100 IF(QNEW.GT.0.) GO TO 110
      TMPB=TMPB+QNEW*AREA
      GO TO 120
  110 TMPA=TMPA+QNEW*AREA
  120 DQ=(QNEW-QOLD)*AREA
C *** (6) ADD VALUES INTO Q-VECTOR ***
      NA=ND(NDC)
      NB=ND(NDC+IE)
      NC=ND(NDC+IE+1)
      Q(NA)=Q(NA)+DQ
      Q(NB)=Q(NB)+DQ
      Q(NC)=Q(NC)+DQ
  130 CONTINUE
      NTE=NTE+2
  140 NDC=NDC+4
  150 CONTINUE
C *** (7) MODIFY TOTAL INFLOW AND OUTFLOW PER UNIT TIME FOR FLOW
C      BALANCE ***
      DQI=3.*TMPA+DQI
      DQO=3.*TMPB+DQO
C *** FOR SOURCE-BED HEADS: ***
```

```
  152 IF(NHRCH.NE.ISTP) GO TO 160
C *** (1) READ NUMBER OF CHANGES AND TIME STEP WHERE CHANGES ARE TO BE
C     MADE AGAIN ***
      READ(IIN,2) N,NHRCH
      WRITE(IOUT,24) ISTP,TIME
      DO 156 I=1,N
C *** (2) READ AND WRITE NEW VALUES ***
      READ (IIN,6) J,HR(J)
      WRITE(IOUT,22) J,HR(J)
  156 CONTINUE
C *** FOR CAUCHY-TYPE BOUNDARIES: ***
  160 IF(NBQCH.NE.ISTP) GO TO 210
C *** (1) READ NUMBER OF CHANGES AND TIME STEP WHERE CHANGES ARE TO BE
C     MADE AGAIN ***
      READ(IIN,2) N,NBQCH
      WRITE(IOUT,16) ISTP,TIME
      DO 200 I=1,N
C *** (2) READ AND WRITE NEW VALUES OF KNOWN FLOW AND KNOWN HEADS ***
      READ(IIN,6) J,QNEW,HK(J),HL(J)
      WRITE(IOUT,18) J,QNEW,HK(J),HL(J)
C *** (3) COMPUTE NEW QBND AND ADD FLOWS INTO Q-VECTOR ***
  195 K=KQB(J)
      L=LQB(J)
      QOLD=QBND(J)
      IF(K.LT.0) GO TO 197
      QBND(J)=QNEW/ALPH(J)
      GO TO 198
  197 K=-K
      QBND(J)=QNEW
  198 Q(K)=Q(K)+(QBND(J)-QOLD)*CFDK(J)
  200 Q(L)=Q(L)+(QBND(J)-QOLD)*CFDL(J)
C *** FOR SPECIFIED-HEAD BOUNDARIES: ***
  210 IF(NHCH.NE.ISTP) RETURN
C *** (1) READ NUMBER OF CHANGES AND TIME STEP WHERE CHANGES ARE TO BE
C     MADE AGAIN ***
      READ(IIN,2) N,NHCH
      WRITE(IOUT,20) ISTP,TIME
      WF=2./3.
      DO 220 I=1,N
C *** (2) READ AND WRITE NEW VALUES ***
      READ(IIN,6) J,HB
      WRITE(IOUT,22) J,HB
C *** (3) FORM HEAD-CHANGE AND NEW-HEAD VECTORS ***
      K=-IN(J)
      DHB(K)=WF*(HB-H(J))
  220 H(J)=DHB(K)+H(J)
      RETURN
      END
```

```
      SUBROUTINE DATIN(TITLE,XG,YG,H,DHB,HR,HK,HL,ALPH,QBND,Q,KQB
     1,LQB,IN,IDZ,IDS,NBCZ)
C         READS DATA, SETS UP SOME INITIAL ARRAYS, AND COMPUTES SOME
C         FLOW-BALANCE COMPONENTS FOR BASIC VERSION OF PROGRAM
      DIMENSION TITLE(20),XG(1),YG(1),H(1),DHB(1),HR(1),HK(1),HL(1)
     1,ALPH(1),QBND(1),Q(1)
      DIMENSION KQB(1),LQB(1),IN(1),IDZ(1),IDS(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** FORMAT LIST ***
    2 FORMAT (I5,7F10.0)
    4 FORMAT (1H0,28X,17HNODAL COORDINATES/1H ,2X,2(4HNODE,5X,7HX COORD
     1,8X,7HY COORD,7X))
    5 FORMAT (1H0,10X,41HNODAL COORDINATES READ, OUTPUT SUPPRESSED)
    6 FORMAT (1H0,29X,13HINITIAL HEADS/1H ,2X,3(4HNODE,8X,4HHEAD,9X))
    7 FORMAT (1H0,10X,37HINITIAL HEADS READ, OUTPUT SUPPRESSED)
    8 FORMAT (1H0,27X,16HSOURCE BED HEADS/1H ,2X,3(4HNODE,8X,4HHEAD,9X))
    9 FORMAT (1H0,10X,40HSOURCE-BED HEADS READ, OUTPUT SUPPRESSED)
   10 FORMAT (1H0,35HSCALE CHANGE FOR NODAL COORDINATES:)
   12 FORMAT (20A4)
   14 FORMAT (1H ,20A4)
   16 FORMAT (8F10.0)
   22 FORMAT (1H0,6X,11HPOINT FLOWS/7H   NODE,5X,9HDISCHARGE)
   23 FORMAT (1H0,10X,35HPOINT FLOWS READ, OUTPUT SUPPRESSED)
   24 FORMAT (1H ,I5,6X,G11.5)
   28 FORMAT (16I5)
   32 FORMAT (1H0,17X,42HCAUCHY-TYPE BOUNDARY DATA BY BOUNDARY ZONE)
   33 FORMAT (1H0,10X,46HCAUCHY-TYPE BOUNDARIES READ, OUTPUT SUPPRESSED)
   34 FORMAT (3I5,4F10.0)
   35 FORMAT (1H0,5X,5HZONE ,I5,4X,9HCONTAINS ,I5,15H BOUNDARY SIDES
     1/1H ,5H SIDE,2X,18HBOUNDARY  BOUNDARY,13X,9HSPECIFIED,5X
     2,8HEXTERNAL,5X,8HEXTERNAL/1H ,5H  NO.,3X,6HNODE A,4X,6HNODE B,3X
     3,5HALPHA,9X,4HFLOW,8X,6HHEAD A,7X,6HHEAD B)
   36 FORMAT (1H ,I4,3X,I5,5X,I5,1X,4(2X,G11.5))
   38 FORMAT (1H0,4X,15HSPECIFIED HEADS/1H ,13X,8HBOUNDARY/7H   NODE,9X
     1,4HHEAD)
   39 FORMAT(1H0,10X,39HSPECIFIED HEADS READ, OUTPUT SUPPRESSED)
   40 FORMAT (1H ,I5,7X,G11.5)
   44 FORMAT (1H0,43H** AXI-SYMMETRIC RADIAL COORDINATES USED **)
   46 FORMAT (1H0,23H** STEADY-STATE FLOW **)
C *** READ INDICATORS FOR RADIAL FLOW, SCALING, AND
C     STEADY-STATE FLOW ***
      READ(IIN,28) IRAD,IUNIT,ISTD
      IF(IRAD.GT.0) WRITE(IOUT,44)
      IF(ISTD.GT.0) WRITE(IOUT,46)
C *** READ AND WRITE NODAL SCALE CHANGE, IF ANY ***
      SCALE=1.
      IF(IUNIT.LE.0) GO TO 60
      WRITE(IOUT,10)
      READ(IIN,12) (TITLE(I),I=1,20)
      WRITE(IOUT,14) (TITLE(I),I=1,20)
      READ(IIN,16) SCALE
```

```
*** READ INDICATOR VARIABLES FOR SUPPRESSING PRINTOUT OF
    INITIAL CONDITIONS ***
 60 READ(IIN,28) IPXY,IPH,IPHR,IPQW,IPCB,IPHB,IPND
C *** READ AND WRITE NODAL COORDINATES AND INITIAL HEADS ***
    DO 70 J=1,NNDS
    READ(IIN,2) I,XG(I),YG(I),H(I),HR(I)
 70 CONTINUE
    IF(IPXY.GT.0) GO TO 71
    WRITE(IOUT,4)
    CALL PRTOB(XG,YG,NNDS)
    GO TO 72
 71 WRITE(IOUT,5)
 72 IF(IPH.GT.0) GO TO 73
    WRITE(IOUT,6)
    CALL PRTOA(H,NNDS)
    GO TO 74
 73 WRITE(IOUT,7)
 74 IF(IPHR.GT.0) GO TO 75
    WRITE(IOUT,8)
    CALL PRTOA(HR,NNDS)
    GO TO 76
 75 WRITE(IOUT,9)
C *** FOR POINT SOURCES AND SINKS: ***
 76 WQI=0.
    WQO=0.
    IF(NWELS.LE.0) GO TO 100
    IF(IPQW.GT.0) GO TO 77
    WRITE(IOUT,22)
    GO TO 78
 77 WRITE(IOUT,23)
 78 DO 90 I=1,NWELS
C *** (1) READ AND WRITE VALUES ***
    READ(IIN,2) J,QWEL
    IF(IPQW.EQ.0) WRITE(IOUT,24) J,QWEL
C *** (2) COMPUTE TOTAL INFLOW OR OUTFLOW PER UNIT TIME
C     FOR FLOW BALANCE ***
    IF(QWEL.GT.0.) GO TO 85
    WQO=WQO+QWEL
    GO TO 90
 85 WQI=WQI+QWEL
C *** (3) ADD VALUES INTO Q-VECTOR ***
 90 Q(J)=Q(J)+QWEL
C *** READ AND WRITE DATA FOR CAUCHY-TYPE BOUNDARIES ***
100 IF(NQBND.LE.0) GO TO 130
    IF(IPCB.GT.0) GO TO 101
    WRITE(IOUT,32)
    GO TO 102
101 WRITE(IOUT,33)
102 DO 128 NZ=1,NBCZ
    READ(IIN,28) KZ,NOS,IZIN
    IF(IPCB.EQ.0) WRITE(IOUT,35) KZ,NOS
    IDZ(NZ)=KZ
    IDS(KZ)=NOS
    IF(IZIN.GT.0) GO TO 124
    DO 120 I=1,NOS
```

```
      READ(IIN,34) J,KQB(J),LQB(J),ALPH(J),QBND(J),HK(J),HL(J)
      IF(IPCB.EQ.0) WRITE(IOUT,36) J,KQB(J),LQB(J),ALPH(J),QBND(J)
     1,HK(J),HL(J)
  120 CONTINUE
      GO TO 128
  124 READ(IIN,16) ALPHZ,QBNZ
      DO 126 I=1,NOS
      READ(IIN,34) J,KQB(J),LQB(J),HK(J),HL(J)
      ALPH(J)=ALPHZ
      QBND(J)=QBNZ
      IF(IPCB.EQ.0) WRITE(IOUT,36) J,KQB(J),LQB(J),ALPH(J),QBND(J)
     1,HK(J),HL(J)
  126 CONTINUE
  128 CONTINUE
C *** INITIALIZE NODAL POINTER ARRAY, IN
  130 DO 134 I=1,NNDS
  134 IN(I)=I
      IN(NNDS+1)=0
      NEQ=NNDS
C *** FOR SPECIFIED-HEAD BOUNDARIES: ***
      IF(NHDS.LE.0) RETURN
      WF=1.
      IF(ISTD.LT.1) WF=2./3.
      IF(IPHB.GT.0) GO TO 135
      WRITE(IOUT,38)
      GO TO 136
  135 WRITE(IOUT,39)
  136 DO 140 I=1,NHDS
C *** (1) READ AND WRITE VALUES ***
      READ(IIN,2) J,HB
      IF(IPHB.EQ.0) WRITE(IOUT,40) J,HB
C *** (2) MODIFY IN TO EXCLUDE SPECIFIED HEAD NODES FROM
C     MATRIX EQUATION ***
      DO 138 K=J,NNDS
  138 IF(IN(K).GT.0) IN(K)=IN(K)-1
      IN(J)=-I
C *** (3) DEFINE HEAD-CHANGE AND MEAN-HEAD VECTORS ***
      DHB(I)=WF*(HB-H(J))
      H(J)=DHB(I)+H(J)
  140 CONTINUE
C *** COMPUTE NUMBER OF SPECIFIED HEADS AT END OF HEAD VECTOR ***
      NLHS=0
      J=NNDS+1
      DO 150 I=1,NNDS
      J=J-1
      IF(IN(J).GT.0) GO TO 160
  150 NLHS=NLHS+1
C *** COMPUTE REDUCED NUMBER OF EQUATIONS ***
  160 NEQ=NNDS-NLHS
      RETURN
      END
```

```
      SUBROUTINE DATOUT(H,DT,TIME,ISTP)
         UPDATES TIME AND WRITES HEAD VECTOR AT END OF TIME STEP
      DIMENSION H(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
   10 FORMAT (1H1,9X,25HOUTPUT FOR TIME STEP NO. ,I3,4H AT ,G11.5
     1,11H TIME UNITS)
   12 FORMAT (1H0,19X,33HCOMPUTED VALUES OF HYDRAULIC HEAD
     1/1H ,3(6H  NODE,8X,4HHEAD,7X))
      TIME=TIME+DT
      WRITE(IOUT,10) ISTP,TIME
      WRITE(IOUT,12)
      CALL PRTOA(H,NNDS)
      RETURN
      END
```

```
      SUBROUTINE EXTRAP(H,DHB,B,IN)
C        EXTRAPOLATES HEADS AT ALL NODES FROM THE MEAN POINT IN THE T  F
C        STEP TO THE END OF THE TIME STEP, THEN ZEROES THE HEAD CHAN
      DIMENSION H(1),DHB(1),B(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      DO 30 I=1,NNDS
      J=IN(I)
      IF(J.GT.0) GO TO 20
      J=-J
      H(I)=.5*DHB(J)+H(I)
      DHB(J)=0.
      GO TO 30
   20 H(I)=.5*B(J)+H(I)
      B(J)=0.
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FMCO(XG,YG,A,Q,AR,ALPH,QBND,CFDK,CFDL,ND,IZN,JPT,KQB
     1,LQB)
         READS ELEMENT DATA AND FORMS COEFFICIENT ARRAYS FOR MATRIX
         EQUATION
      DIMENSION XG(1),YG(1),A(1),Q(1),AR(1),ALPH(1),QBND(1),CFDK(1)
     1,CFDL(1)
      DIMENSION ND(1),IZN(1),JPT(1),KQB(1),LQB(1)
      DIMENSION XL(4),YL(4),TFL(3),NDID(4)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** FORMAT LIST ***
    2 FORMAT (1H0,31X,18HPARAMETERS BY ZONE/1H ,31X,8HROTATION,6X
     1,8HAQUITARD,5X,7HSTORAGE,4X,8HRECHARGE/7H   ZONE,2X,8HX TRANS.
     2,4X,8HY TRANS.,5X,5HANGLE,6X,10HHYD. COND.,2X,11HCOEFFICIENT,4X
     3,4HRATE)
    4 FORMAT (2I5,6F10.0)
    6 FORMAT (1H ,I5,1X,6(1X,G11.5))
    8 FORMAT (16I5)
   10 FORMAT (1H0,18X,12HELEMENT DATA/9H  ELEMENT,2X,6HNODE 1,2X
     1,6HNODE 2,2X,6HNODE 3,2X,6HNODE 4,3X,4HZONE)
   11 FORMAT (1H0,8X,36HELEMENT DATA READ, OUTPUT SUPPRESSED)
   12 FORMAT (1H ,I6,5I8)
C *** FOR CAUCHY-TYPE BOUNDARIES: ***
      IF(NQBND.LE.0) GO TO 35
      CA=.5*SCALE
      DO 30 J=1,NQBND
  *** (1) FORM COEFFICIENTS ***
      K=KQB(J)
      L=LQB(J)
      TMPA=XG(K)-XG(L)
      TMPB=YG(K)-YG(L)
      DIST=CA*(TMPA*TMPA+TMPB*TMPB)**.5
      TMPA=DIST
      TMPB=DIST
      IF(IRAD.LE.0) GO TO 20
      TMPA=DIST*(2.*XG(K)+XG(L))/3.
      TMPB=DIST*(2.*XG(L)+XG(K))/3.
   20 ALF=ALPH(J)
      QB=QBND(J)
      IF(ALF.LT.1.E-30) GO TO 24
      CFDK(J)=ALF*TMPA
      CFDL(J)=ALF*TMPB
      QBND(J)=QB/ALF
      GO TO 26
   24 CFDK(J)=TMPA
      CFDL(J)=TMPB
      KQB(J)=-K
C *** (2) ADD KNOWN-FLUX COMPONENTS INTO Q-VECTOR ***
   26 Q(K)=Q(K)+QB*TMPA
   30 Q(L)=Q(L)+QB*TMPB
C *** INITIALIZE POINTER ARRAY, JPT, AND VARIABLES FOR ZONAL LOOP ***
   35 MBM1=MBWC-1
```

```
      N=MBM1*NNDS
      L=NNDS+1
      DO 40 I=1,N
   40 JPT(I)=L
      DQI=0.
      DQO=0.
      NDC=0
      NTE=0
      CA=.5
      CB=SCALE*SCALE/6.
      IW=MBWC+1
      KNT=0
C *** BEGIN ZONAL LOOP ***
      WRITE(IOUT,2)
      DO 150 IZ=1,NZNS
C *** READ AND WRITE DATA FOR EACH ZONE ***
      READ(IIN,4) KZ,NO,XTR,YTR,ANG,VLC,STR,QD
      WRITE(IOUT,6) KZ,XTR,YTR,ANG,VLC,STR,QD
C *** SCALE DATA ***
      XTR=XTR*CA
      YTR=YTR*CA
      VLC=VLC*CB
      STR=STR*CB
      QD=QD*CB
C *** COMPUTE SINE AND COSINE OF ROTATION ANGLE ***
      IF(ANG.LE.0.) GO TO 42
      ANG=.01745329*ANG
      SN=SIN(ANG)
      CS=COS(ANG)
C *** BEGIN ELEMENT LOOP WITHIN ZONE ***
   42 DO 140 I=1,NO
C *** READ NODE NUMBERS FOR EACH ELEMENT IN ZONE ***
      READ(IIN,8) IEL,(ND(NDC+J),J=1,4)
      KNT=KNT+1
      IZN(KNT)=KZ
      NE=2
      N=4
      IF(ND(NDC+4).GT.0) GO TO 45
      NE=1
      N=3
C *** COMPUTE LOCAL, ROTATED COORDINATES ***
   45 DO 50 J=1,N
      K=ND(NDC+J)
      XL(J)=XG(K)
   50 YL(J)=YG(K)
      IF(ANG.LT.1.E-20) GO TO 60
      DO 55 J=1,N
      TMPA=XL(J)
      TMPB=YL(J)
      XL(J)=TMPA*CS+TMPB*SN
   55 YL(J)=-TMPA*SN+TMPB*CS
C *** BEGIN NODAL LOOP WITHIN ELEMENT ***
   60 NT=NDC+1
      DO 135 IE=1,NE
C *** COMPUTE COEFFICIENTS FOR COORDINATE FUNCTIONS ***
```

```
      NA=ND(NT)
      NB=ND(NT+IE)
      NC=ND(NT+IE+1)
      XNA=XL(1)
      YNA=YL(1)
      XNB=XL(IE+1)
      YNB=YL(IE+1)
      XNC=XL(IE+2)
      YNC=YL(IE+2)
      BJ=YNB-YNC
      BK=YNC-YNA
      BL=YNA-YNB
      CJ=XNC-XNB
      CK=XNA-XNC
      CL=XNB-XNA
      AREA=BJ*CK-BK*CJ
      AR(NTE+IE)=CB*AREA
C *** COMPUTE ELEMENT CONTRIBUTION TO DISTRIBUTED RECHARGE-DISCHARGE ***
      TEQ=QD*AREA
C *** COMPUTE TOTAL DISTRIBUTED RECHARGE OR DISCHARGE PER UNIT
C     TIME FOR FLOW BALANCE ***
      IF(QD.GT.0.) GO TO 65
      DQO=DQO+TEQ
      GO TO 70
   65 DQI=DQI+TEQ
C *** ADD DISTRIBUTED RECHARGE-DISCHARGE INTO Q-VECTOR ***
   70 Q(NA)=Q(NA)+TEQ
      Q(NB)=Q(NB)+TEQ
      Q(NC)=Q(NC)+TEQ
C *** COMPUTE ELEMENT CONTRIBUTIONS TO CAPACITANCE, LEAKANCE,
      AND FLOW DIVERGENCE ***
      TESJ=STR*AREA
      TESK=TESJ
      TESL=TESJ
      XT=XTR/AREA
      YT=YTR/AREA
      IF(IRAD.LE.0) GO TO 75
      TESJ=TESJ*.25*(2.*XNA+XNB+XNC)
      TESK=TESK*.25*(2.*XNB+XNA+XNC)
      TESL=TESL*.25*(2.*XNC+XNA+XNB)
      TMPA=(XNA+XNB+XNC)/3.
      XT=XT*TMPA
      YT=YT*TMPA
   75 TEL=VLC*AREA
      TXJ=XT*BJ
      TYJ=YT*CJ
      TXL=XT*BL
      TYL=YT*CL
      TFL(1)=TXJ*BK+TYJ*CK
      TFL(2)=TXL*BK+TYL*CK
      TFL(3)=TXJ*BL+TYJ*CL
C *** ADD ELEMENT CONTRIBUTIONS FOR CAPACITANCE AND LEAKANCE INTO
C     A-MATRIX ***
      ICA=IW*(NA-1)+1
      ICB=IW*(NB-1)+1
      ICC=IW*(NC-1)+1
```

```
      A(ICA)=A(ICA)+TESJ
      A(ICB)=A(ICB)+TESK
      A(ICC)=A(ICC)+TESL
      A(ICA+1)=A(ICA+1)+TEL
      A(ICB+1)=A(ICB+1)+TEL
      A(ICC+1)=A(ICC+1)+TEL
C *** ADD ELEMENT CONTRIBUTIONS FOR FLOW DIVERGENCE INTO A-MATRIX AND
C     FORM JPT ***
      NDID(1)=NA
      NDID(2)=NB
      NDID(3)=NC
      NDID(4)=NA
      DO 130 J=1,3
      NA=NDID(J)
      NB=NDID(J+1)
      IF(NB.GT.NA) GO TO 80
      NA=NB
      NB=NDID(J)
   80 M=IW*(NA-1)+1
      M1=MBM1*(NA-1)
      K=0
      DO 90 KK=1,MBM1
      K=K+1
      L=JPT(M1+K)
      IF(L-NB) 90,100,110
   90 CONTINUE
  100 A(M+K+1)=A(M+K+1)+TFL(J)
      GO TO 130
  110 KK=MBM1-K
      L=MBWC
      DO 120 LL=1,KK
      L=L-1
      JPT(M1+L)=JPT(M1+L-1)
  120 A(M+L+1)=A(M+L)
      JPT(M1+K)=NB
      A(M+K+1)=TFL(J)
  130 CONTINUE
  135 CONTINUE
      NTE=NTE+2
  140 NDC=NDC+4
  150 CONTINUE
C *** WRITE ELEMENT NODE AND ZONE NUMBERS ***
      IF(IPND.GT.0) GO TO 162
      WRITE(IOUT,10)
      NDC=0
      DO 160 I=1,NELS
      WRITE(IOUT,12) I,(ND(NDC+J),J=1,4),IZN(I)
      IZN(I)=0
  160 NDC=NDC+4
      GO TO 166
  162 WRITE(IOUT,11)
      DO 164 I=1,NELS
  164 IZN(I)=0
C *** ADJUST TOTAL RECHARGE AND DISCHARGE PER UNIT TIME ***
  166 DQI=3.*DQI
      DQO=3.*DQO
      RETURN
      END
```

```
      SUBROUTINE FMECWT(H,DH,DHB,HR,HK,HL,CFDK,CFDL,A,AD,Q,B,THK,DTK
     1,TOP,ASY,DT,JPT,IN,KQB,LQB)
C        FORMS CONDENSED COEFFICIENT MATRIX, A, AND RIGHT-HAND-SIDE
C        VECTOR, B, FOR WATER-TABLE CORRECTOR STEP
      DIMENSION H(1),DH(1),DHB(1),HR(1),HK(1),HL(1),CFDK(1),CFDL(1)
     1,A(1),AD(1),Q(1),B(1),THK(1),DTK(1),TOP(1),ASY(1)
      DIMENSION JPT(1),IN(1),KQB(1),LQB(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** FORMAT LIST ***
    1 FORMAT (1H0,5HNODE ,I5,20H PREDICTED TO BE DRY/1H ,3X
     1,38HPREDICTED AQUIFER THICKNESS AT NODE = ,G11.5/1H ,3X
     2,19HNET FLOW AT NODE = ,G11.5)
    2 FORMAT (1H ,3X,30HNET FLUX FROM NODE REDUCED TO ,G11.5)
      WF=2./3.
      IW=MBWC+1
      NC=1
      DO 25 I=1,NNDS
C *** INITIALIZE  DTK-VECTOR ***
      DTK(I)=0.
      J=IN(I)
      IF(J.GT.0) GO TO 12
C *** COMPUTE DTK-VECTOR AT SPECIFIED-HEAD NODES ***
      HO=H(I)-DHB(-J)
      DHP=1.5*DHB(-J)
      HP=HO+DHP
      IF(HO.LT.TOP(I)) GO TO 8
      IF(HP.GT.TOP(I)) GO TO 25
      DTK(I)=HP-TOP(I)
      GO TO 20
    8 IF(HP.LT.TOP(I)) GO TO 10
      DTK(I)=TOP(I)-HO
      GO TO 20
   10 DTK(I)=DHP
      GO TO 20
C *** COMPUTE OLD AND PREDICTED HEADS AT ACTIVE NODES ***
   12 HO=H(I)-DH(J)
      DHP=1.5*DH(J)
      HP=HO+DHP
C *** INITIALIZE MATRIX DIAGONAL, AD, AND B-VECTOR ***
      AD(J)=0.
      B(J)=0.
      IF(HO.LT.TOP(I)) GO TO 14
      IF(HP.GE.TOP(I)) GO TO 25
C *** FOR ACTIVE NODES CONVERTING FROM CONFINED TO UNCONFINED
C     CONDITIONS: ***
C *** (1) COMPUTE PREDICTED HEAD AND DTK-VECTOR FOR NODES THAT ARE
C     SATURATED ***
      HP=(A(NC)/ASY(I))*(HP-TOP(I))+TOP(I)
      DHP=HP-HO
      DH(J)=WF*DHP
      H(I)=HO+DH(J)
      DTK(I)=HP-TOP(I)
C *** (2) MODIFY B-VECTOR AND DEFINE CAPACITANCE TERM, A, AS ASY ***
      B(J)=B(J)+(ASY(I)-A(NC))*(TOP(I)-HO)/DT
```

```
      A(NC)=ASY(I)
C *** (3) COMPUTE DTK-VECTOR AND IDENTIFY AND REDUCE DISCHARGE AT NC
C     THAT ARE PREDICTED TO GO DRY ***
      IF(THK(I)+DTK(I).GT.0.) GO TO 20
      THKP=THK(I)+DTK(I)
      WRITE(IOUT,1) I,THKP,Q(I)
      DTK(I)=-THK(I)
      IF(Q(I).GE.0.) GO TO 20
      Q(I)=.5*Q(I)
      WQO=WQO-Q(I)
      WRITE(IOUT,2) Q(I)
      GO TO 20
C *** COMPUTE DTK-VECTOR AND MODIFY B-VECTOR AT ACTIVE NODES
C     CONVERTING FROM UNCONFINED TO CONFINED CONDITIONS ***
   14 IF(HP.LT.TOP(I)) GO TO 16
      DTK(I)=TOP(I)-HO
      B(J)=B(J)+(A(NC)-ASY(I))*DTK(I)/DT
      GO TO 20
C *** FOR UNCONFINED ACTIVE NODES: ***
C *** (1) DEFINE CAPACITANCE TERM, A, AS ASY ***
   16 A(NC)=ASY(I)
C *** (2) COMPUTE DTK-VECTOR AND IDENTIFY AND REDUCE DISCHARGE AT
C     NODES THAT ARE DRY ***
      IF(THK(I)+DHP.GT.0.) GO TO 18
      DTK(I)=0.
      IF(THK(I).GT.0.) DTK(I)=-THK(I)
      THKP=THK(I)+DHP
      WRITE(IOUT,1) I,THKP,Q(I)
      IF(Q(I).GE.0..OR.DHP.GE.0.) GO TO 20
      Q(I)=.5*Q(I)
      WQO=WQO-Q(I)
      WRITE(IOUT,2) Q(I)
      GO TO 20
   18 DTK(I)=DHP
      IF(THK(I).LT.0.) DTK(I)=DHP+THK(I)
C *** MODIFY DTK-VECTOR AND DEFINE PH-VECTOR ***
   20 DTK(I)=WF*DTK(I)
   25 NC=NC+IW
C *** INITIALIZE FOR NODAL ASSEMBLY LOOP ***
      DTM=3./(2.*DT)
      MBM1=MBWC-1
      C1=9./16.
      C2=8./9.
      C3=1./16.
      NC=1
      NME=1
      NP=0
      DO 80 I=1,NNDS
      NVL=NC+1
      ND=NVL
C *** COMPUTE A-MATRIX AND B-VECTOR EXCEPT FOR BOUNDARY CONDITION
C     CONTRIBUTIONS ***
      THKI=THK(I)
      IF(THKI.LT.0.) THKI=0.
      K=IN(I)
```

```
      IF(K.LT.0) GO TO 64
      TMPA=A(NC)*DTM
      B(K)=B(K)-TMPA*DH(K)+A(NVL)*(HR(I)-H(I))+Q(I)
      AD(K)=AD(K)+TMPA+A(NVL)
      DO 50 J=1,MBM1
      L=JPT(NP+J)
      IF(L.GT.NNDS) GO TO 60
      THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      TMPA=C1*(DTK(I)+DTK(L)+C2*(THKI+THKL))*A(ND+J)
      AD(K)=AD(K)-TMPA
      M=IN(L)
      IF(M.GT.0) GO TO 40
      M=-M
      B(K)=B(K)-(C3*(DTK(I)+DTK(L))*(DHB(M)-DH(K))
     1+.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(L)-H(I)))*A(ND+J)
      A(NME+J)=0.
      GO TO 50
   40 TMPB=(C3*(DTK(I)+DTK(L))*(DH(M)-DH(K))
     1+.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(L)-H(I)))*A(ND+J)
      B(K)=B(K)-TMPB
      B(M)=B(M)+TMPB
      A(NME+J)=TMPA
      AD(M)=AD(M)-TMPA
   50 CONTINUE
   60 NME=NME+MBWC
      GO TO 70
   64 K=-K
      DO 66 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 66,70,65
   65 THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      B(M)=B(M)-(C3*(DTK(I)+DTK(L))*(DHB(K)-DH(M))
     1+.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(I)-H(L)))*A(ND+J)
      AD(M)=AD(M)-C1*(DTK(I)+DTK(L)+C2*(THKI+THKL))*A(ND+J)
   66 CONTINUE
   70 NC=NC+IW
   80 NP=NP+MBM1
      N=NNDS-NHDS
      NME=1
      J=0
      DO 85 I=1,N
      A(NME)=AD(I)
      J=J+1
   85 NME=NME+MBWC
C *** ADD CONTRIBUTIONS FROM CAUCHY-TYPE BOUNDARIES INTO A AND B ***
      IF(NQBND.LE.0) RETURN
      DO 130 I=1,NQBND
      K=KQB(I)
      IF(K.LT.0) GO TO 130
      L=LQB(I)
      KE=IN(K)
      IF(KE.LT.0) GO TO 100
      M=MBWC*(KE-1)+1
```

```
      A(M)=A(M)+CFDK(I)
      B(KE)=B(KE)+CFDK(I)*(HK(I)-H(K))
  100 LE=IN(L)
      IF(LE.LT.0) GO TO 130
      N=MBWC*(LE-1)+1
      A(N)=A(N)+CFDL(I)
      B(LE)=B(LE)+CFDL(I)*(HL(I)-H(L))
  130 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FMEPWT(H,HR,HK,HL,CFDK,CFDL,A,AD,Q,B,THK,TOP,ASY,DT,JPT
     1,IN,KQB,LQB)
C        FORMS CONDENSED COEFFICIENT MATRIX, A, AND RIGHT-HAND-SIDE
C        VECTOR, B, FOR WATER-TABLE PREDICTOR STEP
      DIMENSION H(1),HR(1),HK(1),HL(1),CFDK(1),CFDL(1),A(1),AD(1),Q(1)
     1,B(1),THK(1),TOP(1),ASY(1)
      DIMENSION JPT(1),IN(1),KQB(1),LQB(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      IW=MBWC+1
      NC=1
      DO 10 I=1,NEQ
      J=IN(I)
      IF(J.LT.0) GO TO 10
C *** INITIALIZE MATRIX DIAGONAL, AD, AND B-VECTOR ***
      AD(J)=0.
      B(J)=0.
C *** DEFINE CAPACITANCE FOR WATER-TABLE NODES AS THE NODAL SPECIFIC
C     YIELD COEFFICIENTS, ASY ***
      IF(H(I).LE.TOP(I)) A(NC)=ASY(I)
   10 NC=NC+IW
C *** INITIALIZE FOR NODAL ASSEMBLY LOOP ***
      DTM=3./(2.*DT)
      MBM1=MBWC-1
      NC=1
      NME=1
      NP=0
C *** COMPUTE A-MATRIX AND B-VECTOR EXCEPT FOR BOUNDARY-CONDITION
C     CONTRIBUTIONS ***
      DO 80 I=1,NNDS
      NVL=NC+1
      ND=NVL
      THKI=THK(I)
      IF(THKI.LT.0.) THKI=0.
      K=IN(I)
      IF(K.LT.0) GO TO 64
      B(K)=B(K)+A(NVL)*(HR(I)-H(I))+Q(I)
      AD(K)=AD(K)+A(NC)*DTM+A(NVL)
      DO 50 J=1,MBM1
      L=JPT(NP+J)
      IF(L.GT.NNDS) GO TO 60
      THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      TMPA=.5*(THKI+THKL)*A(ND+J)
      TMPB=TMPA*(H(L)-H(I))
      B(K)=B(K)-TMPB
      AD(K)=AD(K)-TMPA
      M=IN(L)
      IF(M.GT.0) GO TO 40
      A(NME+J)=0.
      GO TO 50
   40 B(M)=B(M)+TMPB
      A(NME+J)=TMPA
      AD(M)=AD(M)-TMPA
   50 CONTINUE
   60 NME=NME+MBWC
      GO TO 70
```

```
   64 DO 66 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 66,70,65
   65 THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      TMPA=.5*(THKI+THKL)*A(ND+J)
      B(M)=B(M)-TMPA*(H(I)-H(L))
      AD(M)=AD(M)-TMPA
   66 CONTINUE
   70 NC=NC+IW
   80 NP=NP+MBM1
      N=NNDS-NHDS
      NME=1
      DO 85 I=1,N
      A(NME)=AD(I)
   85 NME=NME+MBWC
C *** ADD CONTRIBUTIONS FROM CAUCHY-TYPE BOUNDARIES INTO A AND B ***
      IF(NQBND.LE.0) RETURN
      DO 130 I=1,NQBND
      K=KQB(I)
      IF(K.LT.0) GO TO 130
      L=LQB(I)
      KE=IN(K)
      IF(KE.LT.0) GO TO 100
      M=MBWC*(KE-1)+1
      A(M)=A(M)+CFDK(I)
      B(KE)=B(KE)+CFDK(I)*(HK(I)-H(K))
  100 LE=IN(L)
      IF(LE.LT.0) GO TO 130
      N=MBWC*(LE-1)+1
      A(N)=A(N)+CFDL(I)
      B(LE)=B(LE)+CFDL(I)*(HL(I)-H(L))
  130 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FMEQ(H,HR,HK,HL,CFDK,CFDL,A,AD,Q,B,DT,JPT,IN,KQB,LQB)
C        FORMS CONDENSED COEFFICIENT MATRIX, A, AND RIGHT-HAND-SIDE
C        VECTOR, B
      DIMENSION H(1),HR(1),HK(1),HL(1),CFDK(1),CFDL(1),A(1),AD(1),Q(1)
     1,B(1)
      DIMENSION JPT(1),IN(1),KQB(1),LQB(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
C *** INITIALIZE MATRIX DIAGONAL, AD, AND B-VECTOR ***
      N=NNDS-NHDS
      DO 20 I=1,N
      AD(I)=0.
   20 B(I)=0.
C *** INIALIZE FOR NODAL LOOP ***
      DTM=3./(2.*DT)
      MBM1=MBWC-1
      IW=MBWC+1
      NC=1
      NME=1
      NP=0
C *** COMPUTE A-MATRIX AND B-VECTOR EXCEPT FOR BOUNDARY-CONDITION
C     CONTRIBUTIONS ***
      DO 80 I=1,NNDS
      NVL=NC+1
      ND=NVL
      K=IN(I)
      IF(K.LT.0) GO TO 64
      B(K)=B(K)+A(NVL)*(HR(I)-H(I))+Q(I)
      AD(K)=AD(K)+A(NC)*DTM+A(NVL)
      DO 50 J=1,MBM1
      L=JPT(NP+J)
      IF(L.GT.NNDS) GO TO 60
      TMPB=A(ND+J)*(H(L)-H(I))
      B(K)=B(K)-TMPB
      AD(K)=AD(K)-A(ND+J)
      M=IN(L)
      IF(M.GT.0) GO TO 40
      A(NME+J)=0.
      GO TO 50
   40 B(M)=B(M)+TMPB
      A(NME+J)=A(ND+J)
      AD(M)=AD(M)-A(ND+J)
   50 CONTINUE
   60 NME=NME+MBWC
      GO TO 70
   64 DO 66 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 66,70,65
   65 B(M)=B(M)-A(ND+J)*(H(I)-H(L))
      AD(M)=AD(M)-A(ND+J)
   66 CONTINUE
   70 NC=NC+IW
   80 NP=NP+MBM1
      NME=1
      DO 85 I=1,N
      A(NME)=AD(I)
```

```
   85 NME=NME+MBWC
C *** ADD CONTRIBUTIONS FROM CAUCHY-TYPE BOUNDARIES INTO A AND B ***
      IF(NQBND.LE.0) RETURN
      DO 130 I=1,NQBND
      K=KQB(I)
      IF(K.LT.0) GO TO 130
      L=LQB(I)
      KE=IN(K)
      IF(KE.LT.0) GO TO 100
      M=MBWC*(KE-1)+1
      A(M)=A(M)+CFDK(I)
      B(KE)=B(KE)+CFDK(I)*(HK(I)-H(K))
  100 LE=IN(L)
      IF(LE.LT.0) GO TO 130
      N=MBWC*(LE-1)+1
      A(N)=A(N)+CFDL(I)
      B(LE)=B(LE)+CFDL(I)*(HL(I)-H(L))
  130 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GNBAL(H,DH,DHB,R,GC,HRK,HRL,ZRK,ZRL,ZP,VQK,VQL,DT,IN,KR
     1,LR,KP,NBNC,NPNB)
C        COMPUTES FLOW-BALANCE COMPONENTS FOR NONLINEAR CAUCHY-TYPE
C        BOUNDARIES AND NONLINEAR POINT SINKS
      DIMENSION H(1),DH(1),DHB(1),R(1),GC(1),HRK(1),HRL(1),ZRK(1),ZRL(1)
     1,ZP(1),VQK(1),VQL(1)
      DIMENSION IN(1),KR(1),LR(1),KP(1),NL(2)
      DIMENSION HR(2),ZR(2),QR(2)
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
C *** SET VOLUMETRIC-FLOW RATES TO ZERO ***
      BNQI=0.
      BNQO=0.
      PNQO=0.
C *** COMPUTE WEIGHTING FACTORS FOR GALERKIN-IN-TIME FORMULATION ***
      CA=1./3.
      CB=2./3.
C ***
      IF(NBNC.LT.1) GO TO 65
C *** FORM COEFFICIENTS FOR EACH SIDE J ON A NONLINEAR CAUCHY-TYPE
C     BOUNDARY ***
      DO 60 J=1,NBNC
      HR(1)=HRK(J)
      HR(2)=HRL(J)
      ZR(1)=ZRK(J)
      ZR(2)=ZRL(J)
      NL(1)=KR(J)
      NL(2)=LR(J)
C *** FOR EACH NODE I ON THE SIDE: ***
      DO 50 I=1,2
   *** (1) COMPUTE HEAD AT N LEVEL, HO, AND HEAD AT N+1 LEVEL, HC ***
      L=NL(I)
      LE=IN(L)
      IF(LE.LT.0) GO TO 10
      DHC=DH(LE)
      GO TO 15
   10 DHC=DHB(-LE)
   15 HO=H(L)-DHC
      DHC=1.5*DHC
      HC=HO+DHC
C *** (2) REPRESENT ELEVATIONS, AND HEAD AND ELEVATION DIFFERENCES, BY
C         TEMPORARY (UNSUBSCRIPTED) VARIABLES ***
      ZRI=ZR(I)
      HRI=HR(I)
      DRZ=HRI-ZRI
      DHZ=HO-ZRI
C *** (3) COMPARE HEADS WITH CONTROLLING ELEVATIONS TO IDENTIFY LEAKAGE
C         CASE. COMPUTE HEAD DIFFERENCE, TMPA, FOR VOLUMETRIC-FLOW RATES
C         FOR: ***
      IF(HO.LT.ZRI) GO TO 30
      IF(HC.LT.ZRI) GO TO 20
C *** CASE (1) ***
      TMPA=HRI-H(L)
      GO TO 40
C *** CASE (2) ***
   20 PHI=DHZ/DHC
```

```
      TMPA=DRZ-CA*PHI*PHI*DHZ
      GO TO 40
   30 IF(HC.GT.ZRI) GO TO 35
C *** CASE (4) ***
      TMPA=DRZ
      GO TO 40
C *** CASE (3) ***
   35 PHC=1.+DHZ/DHC
      TMPA=DRZ-CB*PHC*(HC-ZRI)
C *** (4) COMPUTE VOLUMETRIC-FLOW RATE ***
   40 TMPB=GC(J)*TMPA
      QR(I)=TMPB
C *** (5) INCLUDE VOLUMETRIC-FLOW RATE INTO FLOW-BALANCE EQUATION AT
C         SPECIFIED-HEAD NODE ***
      IF(LE.LT.0) R(-LE)=R(-LE)-TMPB
C *** (6) SUM VOLUMETRIC-FLOW RATES ACCORDING TO SIGN, POSITIVE
C         FOR INFLOW ***
      IF(TMPB.LT.0.) GO TO 45
      BNQI=BNQI+TMPB
      GO TO 50
   45 BNQO=BNQO+TMPB
   50 CONTINUE
      VQK(J)=QR(1)
      VQL(J)=QR(2)
   60 CONTINUE
C *** COMPUTE TOTAL VOLUMES TO BE RECHARGED (+) AND DICHARGED (-) ACROSS
C     BOUNDARY FOR ENTIRE SIMULATION ***
      TBNQI=TBNQI+BNQI*DT
      TBNQO=TBNQO+BNQO*DT
C *** INCLUDE VOLUMETRIC-FLOW RATES IN THE FLOW IMBALANCE ***
      ER=ER-BNQI-BNQO
C ***
   65 IF(NPNB.LT.1) RETURN
C *** FORM COEFFICIENTS FOR NONLINEAR POINT SINKS ***
C *** FOR EACH POINT I: ***
      DO 100 I=1,NPNB
      K=KP(I)
      KE=IN(K)
      IF(KE.LT.0) GO TO 70
      DHC=DH(KE)
      GO TO 75
   70 DHC=DHB(-KE)
   75 IP=I+NBNC
C *** (1) COMPUTE HEAD AT N LEVEL, HO, AND HEAD AT N+1 LEVEL, HC ***
      HO=H(K)-DHC
      DHC=1.5*DHC
      HC=HO+DHC
C *** (2) REPRESENT ELEVATION, AND HEAD AND ELEVATION DIFFERENCE, BY
C         TEMPORARY (UNSUBSCRIPTED) VARIABLES ***
      ZPI=ZP(I)
      DHZ=ZPI-HO
C *** (3) COMPARE HEADS WITH CONTROLLING ELEVATION TO IDENTIFY LEAKAGE
C         CASE. COMPUTE HEAD DIFFERENCE, TMPA, FOR VOLUMETRIC-FLOW RATES
C         FOR: ***
      IF(HO.GT.ZPI) GO TO 80
```

```
      IF(HC.LT.ZPI) GO TO 100
C *** CASE (3) ***
      PHC=1.-DHZ/DHC
      TMPA=CB*PHC*(ZPI-HC)
      GO TO 90
   80 IF(HC.GT.ZPI) GO TO 85
C *** CASE (2) ***
      PHI=DHZ/DHC
      TMPA=CA*PHI*PHI*DHZ
      GO TO 90
C *** CASE (1) ***
   85 TMPA=ZPI-H(K)
C *** (4) COMPUTE VOLUMERIC-FLOW RATE ***
   90 TMPB=GC(IP)*TMPA
C *** (5) INCLUDE VOLUMETRIC-FLOW RATE INTO FLOW-BALANCE EQUATION AT
C         SPECIFIED-HEAD NODE ***
      IF(KE.LT.0) R(-KE)=R(-KE)-TMPB
C *** (6) SUM VOLUMETRIC-FLOW RATES ***
      PNQO=PNQO+TMPB
  100 CONTINUE
C *** COMPUTE TOTAL VOLUME DISCHARGED ACROSS POINTS FOR ENTIRE
C     SIMULATION ***
      TPNQO=TPNQO+PNQO*DT
C *** INCLUDE VOLUMETRIC-FLOW RATES IN THE FLOW IMBALANCE ***
      ER=ER-PNQO
      RETURN
      END
```

```
      SUBROUTINE GNBLSS(H,R,GC,HRK,HRL,ZRK,ZRL,ZP,VQK,VQL,DT,IN,KR,LR,KP
     1,NBNC,NPNB)
C         COMPUTES FLOW-BALANCE COMPONENTS FOR NONLINEAR CAUCHY-TYPE
C         BOUNDARIES AND NONLINEAR POINT SINKS FOR STEADY-STATE
C         SIMULATIONS
      DIMENSION H(1),R(1),GC(1),HRK(1),HRL(1),ZRK(1),ZRL(1),ZP(1),VQK(1)
     1,VQL(1)
      DIMENSION IN(1),KR(1),LR(1),KP(1),NL(2)
      DIMENSION HR(2),ZR(2),QR(2)
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
C *** SET VOLUMETRIC-FLOW RATES TO ZERO ***
      BNQI=0.
      BNQO=0.
      PNQO=0.
C ***
      IF(NBNC.LT.1) GO TO 65
C *** FORM COEFFICIENTS FOR EACH SIDE J ON A NONLINEAR CAUCHY-TYPE
C     BOUNDARY ***
      DO 60 J=1,NBNC
      HR(1)=HRK(J)
      HR(2)=HRL(J)
      ZR(1)=ZRK(J)
      ZR(2)=ZRL(J)
      NL(1)=KR(J)
      NL(2)=LR(J)
C *** FOR EACH NODE I ON THE SIDE: ***
      DO 50 I=1,2
      L=NL(I)
C *** (1) REPRESENT HEADS AND CONTROLLING ELEVATION BY TEMPORARY
C         (UNSUBSCRIPTED) VARIABLES ***
      HL=H(L)
      HRI=HR(I)
      ZRI=ZR(I)
C *** (2) COMPARE HEAD WITH CONTROLLING ELEVATION TO IDENTIFY LEAKAGE
C         CASE. COMPUTE HEAD DIFFERENCE, TMPA, FOR VOLUMETRIC-FLOW RATES
C         FOR: ***
      IF(HL.GT.ZRI) GO TO 20
C *** CASE (4) ***
      TMPA=HRI-ZRI
      GO TO 30
C *** CASE (1) ***
   20 TMPA=HRI-HL
C *** (3) COMPUTE VOLUMETRIC-FLOW RATE ***
   30 TMPB=GC(J)*TMPA
      QR(I)=TMPB
C *** (4) INCLUDE VOLUMETRIC-FLOW RATE INTO FLOW-BALANCE EQUATION AT
C         SPECIFIED-HEAD NODE ***
      LE=-IN(L)
      IF(LE.GT.0) R(LE)=R(LE)-TMPB
C *** (5) SUM VOLUMETRIC-FLOW RATES ACCORDING TO SIGN, POSITIVE FOR
C         INFLOW ***
      IF(TMPB.LT.0.) GO TO 40
      BNQI=BNQI+TMPB
      GO TO 50
   40 BNQO=BNQO+TMPB
```

```
   50 CONTINUE
      VQK(J)=QR(1)
      VQL(J)=QR(2)
   60 CONTINUE
C *** COMPUTE TOTAL VOLUMES RECHARGED (+) AND DISCHARGED (-) ACROSS
C     BOUNDARY FOR SIMULATION ***
      TBNQI=BNQI*DT
      TBNQO=BNQO*DT
C *** INCLUDE VOLUMETRIC-FLOW RATES IN THE FLOW IMBALANCE ***
      ER=ER-BNQI-BNQO
C ***
   65 IF(NPNB.LT.1) RETURN
C *** FORM COEFFICIENTS FOR NONLINEAR POINT SINKS ***
C *** FOR EACH POINT I: ***
      DO 80 I=1,NPNB
      K=KP(I)
      IP=I+NBNC
C *** REPRESENT HEAD AND CONTOLLING ELEVATION BY TEMPORARY
C     (UNSUBSCRIPTED) VARIABLES ***
      HK=H(K)
      ZPI=ZP(I)
C *** COMPARE HEAD WITH CONTROLLING ELEVATION TO DETERMINE LEAKAGE
C     CASE ***
      IF(HK.GT.ZPI) GO TO 70
C *** NO LEAKAGE, THUS NO COMPUTATION ***
      GO TO 80
C *** LEAKAGE, THUS COMPUTE VOLUMETRIC-FLOW RATE ***
   70 TMPB=GC(IP)*(ZPI-HK)
   *** INCLUDE VOLUMETRIC-FLOW RATE INTO FLOW-BALANCE EQUATION AT
      SPECIFIED-HEAD NODES ***
      KE=-IN(K)
      IF(KE.GT.0) R(KE)=R(KE)-TMPB
C *** SUM VOLUMETRIC-FLOW RATES ***
      PNQO=PNQO+TMPB
   80 CONTINUE
C *** COMPUTE TOTAL VOLUME DISCHARGED ACROSS BOUNDARY FOR SIMULATION ***
      TPNQO=PNQO*DT
C *** INCLUDE VOLUMETRIC-FLOW RATES INTO FLOW IMBALANCE ***
      ER=ER-PNQO
      RETURN
      END
```

```
      SUBROUTINE GNCHG(HRK,HRL,TIME,KR,LR,ISTP)
C        CHANGES VALUES OF EXTERNAL HEADS FOR NONLINEAR CAUCHY-TYPE
C        BOUNDARIES FOR TIME-VARIANT BOUNDARY CONDITIONS
      DIMENSION HRK(1),HRL(1)
      DIMENSION KR(1),LR(1)
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (16I5)
    4 FORMAT (1H1,13X,34HCHANGES IN VALUES OF EXTERNAL HEAD
     1/1H ,12X,36HFOR NONLINEAR CAUCHY-TYPE BOUNDARIES
     2/1H ,29H  BEGINNING ON TIME STEP NO. ,I4,4H AT ,G11.5
     3,11H TIME UNITS/1H ,15X,4HSIDE,19X,8HEXTERNAL/1H ,16X,3HNO.,8X
     4,4HNODE,9X,4HHEAD)
    6 FORMAT (I5,2F10.0)
    8 FORMAT (1H ,13X,2(I5,7X),G11.5/1H ,25X,I5,7X,G11.5)
C *** READ NUMBER OF BOUNDARIES TO BE CHANGED AND TIME-STEP NUMBER FOR
C     NEXT CHANGE ***
      READ(IIN,2) N,NGNCH
C *** WRITE HEADING ***
      WRITE(IOUT,4) ISTP,TIME
C *** READ AND WRITE VALUES OF NEW EXTERNAL HEADS ***
      DO 20 I=1,N
      READ(IIN,6) J,HRK(J),HRL(J)
      WRITE(IOUT,8) J,KR(J),HRK(J),LR(J),HRL(J)
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GNCORR(H,DH,A,B,GC,HRK,HRL,ZRK,ZRL,ZP,IN,KR,LR,KP
     1,NBNC,NPNB)
C        FORMS COEFFICIENTS FOR NONLINEAR CAUCHY-TYPE BOUNDARIES AND
C        NONLINEAR POINT SINKS; ADDS TERMS TO MATRIX EQUATION FOR
C        THE CORRECTOR
      DIMENSION H(1),DH(1),A(1),B(1),GC(1),HRK(1),HRL(1),ZRK(1),ZRL(1)
     1,ZP(1)
      DIMENSION IN(1),KR(1),LR(1),KP(1)
      DIMENSION HR(2),ZR(2),NL(2)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
C *** COMPUTE WEIGHTING FACTORS FOR GALERKIN-IN-TIME FORMULATION ***
      CA=1./3.
      CB=2./3.
C ***
      IF(NBNC.LT.1) GO TO 110
C *** FORM COEFFICIENTS FOR EACH SIDE J ON A NONLINEAR CAUCHY-TYPE
C     BOUNDARY ***
      DO 100 J=1,NBNC
      HR(1)=HRK(J)
      HR(2)=HRL(J)
      ZR(1)=ZRK(J)
      ZR(2)=ZRL(J)
      NL(1)=KR(J)
      NL(2)=LR(J)
C *** FOR EACH NODE I ON THE SIDE: ***
      DO 50 I=1,2
      L=NL(I)
      LE=IN(L)
C *** (1) BYPASS SPECIFIED-HEAD NODES ***
      IF(LE.LT.0) GO TO 50
      N=MBWC*(LE-1)+1
C *** (2) REPRESENT HEADS, ELEVATIONS, AND DIFFERENCES BY TEMPORARY
C          (UNSUBSCRIPTED) VARIABLES ***
      HO=H(L)-DH(LE)
      DHP=1.5*DH(LE)
      HP=HO+DHP
      ZRI=ZR(I)
      HRI=HR(I)
      DRZ=HRI-ZRI
      DHZ=HO-ZRI
C *** (3) COMPARE HEADS WITH CONTROLLING ELEVATIONS TO IDENTIFY LEAKAGE
C          CASE. ADD LEAKAGE COEFFICIENT TO MAIN DIAGONAL, IF APPLICABLE,
C          AND COMPUTE HEAD DIFFERENCE, TMPA, FOR RIGHT-SIDE TERM
C          FOR: ***
      IF(HO.LT.ZRI) GO TO 30
      IF(HP.LT.ZRI) GO TO 20
C *** CASE (1) ***
      A(N)=A(N)+GC(J)
      TMPA=HRI-H(L)
      GO TO 40
C *** CASE (2) ***
   20 PHI=DHZ/DHP
      TMPA=DRZ-CA*PHI*PHI*DHZ
      GO TO 40
   30 IF(HP.GT.ZRI) GO TO 35
C *** CASE (4) ***
```

```
      TMPA=DRZ
      GO TO 40
C *** CASE (3) ***
   35 PHC=1.+DHZ/DHP
      A(N)=A(N)+PHC*GC(J)
      TMPA=DRZ-CB*PHC*(HP-ZRI)
C *** (4) COMPUTE RIGHT-SIDE TERM AND ADD TO B-VECTOR ***
   40 B(LE)=B(LE)+GC(J)*TMPA
   50 CONTINUE
  100 CONTINUE
C ***
  110 IF(NPNB.LT.1) RETURN
C *** FORM COEFFICIENTS FOR NONLINEAR POINT SINKS ***
C *** FOR EACH POINT I: ***
      DO 150 I=1,NPNB
      K=KP(I)
      KE=IN(K)
C *** (1) BYPASS SPECIFIED-HEAD NODES ***
      IF(KE.LT.0) GO TO 150
      N=MBWC*(KE-1)+1
      IP=I+NBNC
C *** (2) REPRESENT HEADS, ELEVATION, AND DIFFERENCE BY TEMPORARY
C         (UNSUBSCRIPTED) VARIABLES ***
      HO=H(K)-DH(KE)
      DHP=1.5*DH(KE)
      HP=HO+DHP
      ZPI=ZP(I)
      DHZ=ZPI-HO
C *** (3) COMPARE HEADS WITH CONTROLLING ELEVATION TO IDENTIFY LEAKA
C         CASE. ADD LEAKAGE COEFFICIENT TO MAIN DIAGONAL, IF APPLICA.
C         AND COMPUTE HEAD DIFFERENCE, TMPA, FOR RIGHT-SIDE TERM
C         FOR: ***
      IF(HO.GT.ZPI) GO TO 120
      IF(HP.LT.ZPI) GO TO 150
C *** CASE (3) ***
      PHC=1.-DHZ/DHP
      A(N)=A(N)+GC(IP)*PHC
      TMPA=CB*PHC*(ZPI-HP)
      GO TO 140
  120 IF(HP.GT.ZPI) GO TO 130
C *** CASE (2) ***
      PHI=DHZ/DHP
      TMPA=CA*PHI*PHI*DHZ
      GO TO 140
C *** CASE (1) ***
  130 A(N)=A(N)+GC(IP)
      TMPA=ZPI-H(K)
C *** (4) COMPUTE RIGHT-SIDE TERM AND ADD TO B-VECTOR ***
  140 B(KE)=B(KE)+GC(IP)*TMPA
  150 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GNFMCO(XG,YG,GC,HRK,HRL,ZRK,ZRL,ZP,KR,LR,KP,INLZ,INLS
     1,NBNC,NPNB,NLCZ)
C        READS AND WRITES DATA AND COMPUTES LEAKAGE COEFFICIENTS FOR
C        NONLINEAR CAUCHY-TYPE BOUNDARIES AND NONLINEAR POINT SINKS
      DIMENSION XG(1),YG(1),GC(1),HRK(1),HRL(1),ZRK(1),ZRL(1),ZP(1)
      DIMENSION KR(1),LR(1),KP(1),INLZ(1),INLS(1)
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/SCLE/SCALE
C *** FORMAT LIST ***
    2 FORMAT (1H0,13X,52HNONLINEAR CAUCHY-TYPE BOUNDARY DATA BY BOUNDARY
     1 ZONE)
    3 FORMAT (1H0,5X,5HZONE ,I5,4X,9HCONTAINS ,I5,15H BOUNDARY SIDES
     1/1H ,7X,4HSIDE,6X,7HLEAKAGE,19X,8HEXTERNAL,4X,11HCONTROLLING/1H
     2,8X,3HNO.,4X,11HCOEFFICIENT,7X,4HNODE,8X,4HHEAD,7X,9HELEVATION)
    4 FORMAT (F10.0)
    5 FORMAT (1H0,8X,50HNONLINEAR CAUCHY-TYPE DATA READ, OUTPUT SUPPRESS
     1ED)
    6 FORMAT (3I5,5F10.0)
    7 FORMAT (1H ,2(5X,I5,6X,G11.5),3X,G11.5/1H ,32X,I5,3X,2(3X,G11.5))
    8 FORMAT (1H0,16X,25HNONLINEAR POINT SINK DATA/1H ,6X,5HPOINT,5X
     1,4HNODE,8X,7HLEAKAGE,6X,11HCONTROLLING/1H ,7X,3HNO.,7X,3HNO.,6X
     2,11HCOEFFICIENT,5X,9HELEVATION)
    9 FORMAT(1H0,8X,48HNONLINEAR POINT SINK DATA READ,OUTPUT SUPPRESSED)
   10 FORMAT (2I5,2F10.0)
   12 FORMAT (1H ,4X,I5,5X,I5,2X,2(5X,G11.5))
C *** COMPUTE FACTORING AND SCALING TERM FOR BOUNDARY LENGTHS ***
      CA=.5*SCALE
C *** READ INDICATOR VARIABLES FOR SUPRESSING PRINTOUT OF NONLINEAR-
C     BOUNDARY DATA ***
      READ(IIN,6) IPNC,IPNP
      IF(NBNC.LT.1) GO TO 30
C *** FOR NONLINEAR CAUCHY-TYPE BOUNDARIES: ***
      IF(IPNC.GT.0) GO TO 14
      WRITE(IOUT,2)
      GO TO 15
   14 WRITE(IOUT,5)
C *** (1) READ AND WRITE DATA DEFINING BOUNDARY SIDE I ***
   15 DO 19 NZ=1,NLCZ
      READ(IIN,6) KZ,NOS,IZIN
      IF(IPNC.EQ.0) WRITE(IOUT,3) KZ,NOS
      INLZ(NZ)=KZ
      INLS(NZ)=NOS
      IF(IZIN.GT.0) GO TO 17
      DO 16 I=1,NOS
      READ(IIN,6) J,KR(J),LR(J),GC(J),HRK(J),HRL(J),ZRK(J),ZRL(J)
      IF(IPNC.EQ.0) WRITE(IOUT,7) J,GC(J),KR(J),HRK(J),ZRK(J),LR(J)
     1,HRL(J),ZRL(J)
   16 CONTINUE
      GO TO 19
   17 READ(IIN,4) GCZ
      DO 18 I=1,NOS
      READ(IIN,6) J,KR(J),LR(J),HRK(J),HRL(J),ZRK(J),ZRL(J)
      GC(J)=GCZ
      IF(IPNC.EQ.0) WRITE(IOUT,7) J,GC(J),KR(J),HRK(J),ZRK(J),LR(J)
     1,HRL(J),ZRL(J)
   18 CONTINUE
```

```
   19   CONTINUE
C *** (2) COMPUTE LENGTH OF BOUNDARY SIDE ***
        DO 20 J=1,NBNC
        K=KR(J)
        L=LR(J)
        TMPA=XG(K)-XG(L)
        TMPB=YG(K)-YG(L)
        DIST=CA*(TMPA*TMPA+TMPB*TMPB)**.5
C *** (3) COMPUTE LEAKAGE COEFFICIENT ***
   20   GC(J)=GC(J)*DIST
C ***
   30   IF(NPNB.LT.1) RETURN
C *** FOR NONLINEAR POINT SINKS: ***
        IF(IPNP.GT.0) GO TO 32
        WRITE(IOUT,8)
        GO TO 34
   32   WRITE(IOUT,9)
C *** (1) READ AND WRITE DATA DEFINING THE BOUNDARY ***
   34   DO 40 J=1,NPNB
        READ(IIN,10) I,KP(I),GCP,ZP(I)
        IF(IPNP.EQ.0) WRITE(IOUT,12) I,KP(I),GCP,ZP(I)
C *** (2) STORE LEAKAGE COEFFICIENT IN GC ***
        GC(I+NBNC)=GCP
   40   CONTINUE
        RETURN
        END
```

```
      SUBROUTINE GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA
     1,INZA,INSA,NBNC,NPNB,NLCZ)
C        READS DATA, ASSIGNS STORAGE, AND INITIALIZES STORAGE LOCATIONS
C        TO ZERO FOR NONLINEAR CAUCHY-TYPE BOUNDARIES AND NONLINEAR
C        POINT SINKS
      DIMENSION G(1)
      COMMON/GDIM/ISUM
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (16I5)
    4 FORMAT (1H0,54HNO. OF NONLINEAR CAUCHY-TYPE BOUNDARIES (NBNC).....
     $ = ,I5/
     $1H ,54HNO. OF NONLINEAR CAUCHY-TYPE BOUNDARY ZONES (NLCZ). = ,I5/
     $1H ,54HNO. OF NONLINEAR POINT SINKS (NPNB)..... = ,I5)
    6 FORMAT (1H0,64HNONLINEAR CAUCHY-TYPE BOUNDARIES AND (OR) NONLINEAR
     1 POINT SINKS:/1H ,38HNOW G MUST BE DIMENSIONED TO AT LEAST ,I6)
C *** READ AND WRITE NUMBER OF NONLINEAR CAUCHY-TYPE BOUNDARIES AND
C     NONLINEAR POINT SINKS ***
      READ(IIN,2) NBNC,NLCZ,NPNB
      WRITE(IOUT,4) NBNC,NLCZ,NPNB
C *** DEFINE ADDRESSES OF NEW ARRAYS WITHIN G ***
      IGCA=ISUM
      ISUM=ISUM+NBNC+NPNB
      IHRK=ISUM
      ISUM=ISUM+NBNC
      IHRL=ISUM
      ISUM=ISUM+NBNC
      IZRK=ISUM
      ISUM=ISUM+NBNC
      IZRL=ISUM
      ISUM=ISUM+NBNC
      IKRA=ISUM
      ISUM=ISUM+NBNC
      ILRA=ISUM
      ISUM=ISUM+NBNC
      INZA=ISUM
      ISUM=ISUM+NLCZ
      INSA=ISUM
      ISUM=ISUM+NLCZ
      IZPA=ISUM
      ISUM=ISUM+NPNB
      IKPA=ISUM
      ISUM=ISUM+NPNB
C *** WRITE SIZE OF G ***
      WRITE(IOUT,6) ISUM
C *** INITIALIZE G ***
      DO 10 I=IGCA,ISUM
   10 G(I)=0.
      RETURN
      END
```

```
      SUBROUTINE GNPRED(H,A,B,GC,HRK,HRL,ZRK,ZRL,ZP,IN,KR,LR,KP,NBNC
     1,NPNB)
C         FORMS COEFFICIENTS FOR NONLINEAR CAUCHY-TYPE BOUNDARIES AND
C         NONLINEAR POINT SINKS; ADDS TERMS TO MATRIX EQUATION FOR THE
C         PREDICTOR
      DIMENSION H(1),A(1),B(1),GC(1),HRK(1),HRL(1),ZRK(1),ZRL(1),ZP(1)
      DIMENSION IN(1),KR(1),LR(1),KP(1)
      DIMENSION HR(2),ZR(2),NL(2)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
C ***
      IF(NBNC.LT.1) GO TO 50
C *** FORM COEFFICIENTS FOR EACH SIDE J ON A NONLINEAR CAUCHY-TYPE
C     BOUNDARY ***
      DO 40 J=1,NBNC
      HR(1)=HRK(J)
      HR(2)=HRL(J)
      ZR(1)=ZRK(J)
      ZR(2)=ZRL(J)
      NL(1)=KR(J)
      NL(2)=LR(J)
C *** FOR EACH NODE I ON THE BOUNDARY: ***
      DO 30 I=1,2
      L=NL(I)
      LE=IN(L)
C *** (1) BYPASS SPECIFIED-HEAD NODES ***
      IF(LE.LT.0) GO TO 30
C *** (2) COMPARE AQUIFER HEAD WITH CONTROLLING ELEVATION TO IDENTIFY
C         LEAKAGE CASE. ADD LEAKAGE COEFFICIENT TO MAIN DIAGONAL, IF
C         APPLICABLE, AND COMPUTE HEAD DIFFERENCE, TMPA, FOR RIGHT SIDE
C         FOR: ***
      IF(H(L).LT.ZR(I)) GO TO 10
      N=MBWC*(LE-1)+1
C *** CASE (1) ***
      A(N)=A(N)+GC(J)
      TMPA=HR(I)-H(L)
      GO TO 20
C *** CASE (4) ***
   10 TMPA=HR(I)-ZR(I)
C *** (3) COMPUTE RIGHT-SIDE TERM AND ADD TO B-VECTOR ***
   20 B(LE)=B(LE)+GC(J)*TMPA
   30 CONTINUE
   40 CONTINUE
C ***
   50 IF(NPNB.LT.1) RETURN
C *** FORM COEFFICIENTS FOR NONLINEAR POINT SINKS ***
C *** FOR EACH POINT I: ***
      DO 60 I=1,NPNB
      IP=I+NBNC
      K=KP(I)
      KE=IN(K)
C *** (1) BYPASS SPECIFIED-HEAD NODES ***
      IF(KE.LT.0) GO TO 60
C *** (2) COMPARE AQUIFER HEAD WITH CONTROLLING ELEVATION TO DETERMINE
C         IF LEAKAGE CASE (1) APPLIES ***
      IF(H(K).LT.ZP(I)) GO TO 60
      N=MBWC*(KE-1)+1
```

```
*** (3) COMPUTE AND ADD TERMS TO MAIN DIAGONAL AND RIGHT SIDE ***
    A(N)=A(N)+GC(IP)
    B(KE)=B(KE)+GC(IP)*(ZP(I)-H(K))
 60 CONTINUE
    RETURN
    END
```

```
      SUBROUTINE HCALC(H,B,IN)
C        COMPUTES MEAN HEAD DURING TIME STEP AT ALL ACTIVE NODES
      DIMENSION H(1),B(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      DO 10 I=1,NEQ
      J=IN(I)
      IF(J.GT.0) H(I)=H(I)+B(J)
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE HCALWT(H,DH,B,IN)
C        COMPUTES MEAN HEAD AND MEAN HEAD CHANGE DURING TIME STEP AT
C        ALL ACTIVE NODES
      DIMENSION H(1),DH(1),B(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      DO 10 I=1,NEQ
      J=IN(I)
      IF(J.LT.0) GO TO 10
      H(I)=H(I)+B(J)
      DH(J)=DH(J)+B(J)
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INITB(TITLE,G,TIME,NBCZ)
C        READS PROBLEM SPECIFICATIONS, DEFINES AND INITIALIZES VARIAP
C        FOR BASIC VERSION OF PROGRAM THAT USES BAND MATRIX SOLVER
      DIMENSION TITLE(20),G(1)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
C *** FORMAT LIST ***
    8 FORMAT (1H1)
   10 FORMAT (20A4)
   12 FORMAT (1H ,20A4)
   14 FORMAT (16I5)
   16 FORMAT (1H0,54HNO. OF ELEMENTS (NELS).............................
     1 = ,I5/
     $1H ,54HNO. OF NODES (NNDS)..................................... = ,I5/
     $1H ,54HMAX. NO. OF TIME STEPS PER STRESS PERIOD (MXSTPS).. = ,I5/
     $1H ,54HNO. OF STRESS PERIODS (NPER)........................ = ,I5/
     $1H ,54HNO. OF ZONES (NZNS)................................. = ,I5/
     $1H ,54HNO. OF POINT FLOWS (NWELS).......................... = ,I5/
     $1H ,54HNO. OF CAUCHY-TYPE BOUNDARY ELEMENT SIDES (NQBND).. = ,I5/
     $1H ,54HNO. OF CAUCHY-TYPE BOUNDARY ZONES (NBCZ)............ = ,I5/
     $1H ,54HNO. OF SPECIFIED HEADS (NHDS)....................... = ,I5/
     $1H ,54HMAXIMUM CONDENSED BAND WIDTH (MBWC)................. = ,I
     $1H ,54HMAXIMUM MATRIX BAND WIDTH (MBW)..................... = ,
   18 FORMAT (33H0DIMENSION OF G MUST BE AT LEAST ,I6)
C *** DEFINE FILE NUMBERS ***
      IIN=50
      IOUT=60
      ITA=55
      ITB=56
C *** READ AND WRITE PROBLEM TITLE ***
      WRITE(IOUT,8)
      DO 20 I=1,3
      READ(IIN,10) (TITLE(K),K=1,20)
      WRITE(IOUT,12) (TITLE(K),K=1,20)
   20 CONTINUE
C *** READ AND WRITE PROBLEM SPECIFICATIONS ***
      READ(IIN,14) NELS,NNDS,MXSTPS,NPER,NZNS,NWELS,NQBND,NBCZ,NHDS,MBWC
     1,MBW
      WRITE(IOUT,16) NELS,NNDS,MXSTPS,NPER,NZNS,NWELS,NQBND,NBCZ,NHDS
     1,MBWC,MBW
C *** DEFINE ADDRESSES OF ARRAYS WITHIN G ***
      NA=(MBWC+1)*NNDS
      NB=2*NELS
      NC=MBW*(NNDS-NHDS-MBW)+(MBW*(MBW+1))/2
      ND=(MBWC-1)*NNDS
      NE=4*NELS
      NF=NNDS-NHDS
      IF(NELS.GT.NF) NF=NELS
```

```
      ISUM=1
      IAA=ISUM
      ISUM=ISUM+NA
      IARA=ISUM
      ISUM=ISUM+NB
      INDA=ISUM
      ISUM=ISUM+NE
      IXGA=ISUM
      ISUM=ISUM+NNDS
      IYGA=ISUM
      ISUM=ISUM+NNDS
      IF(NC.GE.ISUM) ISUM=NC+1
      IATA=ISUM
      ISUM=ISUM+MBW
      IQA=ISUM
      ISUM=ISUM+NNDS
      IBA=ISUM
      ISUM=ISUM+NF
      IHA=ISUM
      ISUM=ISUM+NNDS
      IHRA=ISUM
      ISUM=ISUM+NNDS
      IHBA=ISUM
      ISUM=ISUM+NHDS
      IALA=ISUM
      ISUM=ISUM+NQBND
      IQBA=ISUM
      ISUM=ISUM+NQBND
      ICKA=ISUM
      ISUM=ISUM+NQBND
      ICLA=ISUM
      ISUM=ISUM+NQBND
      IHKA=ISUM
      ISUM=ISUM+NQBND
      IHLA=ISUM
      ISUM=ISUM+NQBND
      IDZA=ISUM
      ISUM=ISUM+NBCZ
      IDSA=ISUM
      ISUM=ISUM+NBCZ
      IDTA=ISUM
      ISUM=ISUM+MXSTPS
      IJPA=ISUM
      ISUM=ISUM+ND
      INA=ISUM
      ISUM=ISUM+NNDS+1
      IKA=ISUM
      ISUM=ISUM+NQBND
      ILA=ISUM
      ISUM=ISUM+NQBND
C *** WRITE SIZE OF G ***
      WRITE(IOUT,18) ISUM
C *** INITIALIZE G ***
      DO 30 I=1,ISUM
```

```
   30 G(I)=0.
C *** INITIALIZE TIME AND FLOW-BALANCE COMPONENTS ***
      TIME=0.
      TSA=0.
      TWQI=0.
      TWQO=0.
      TDQI=0.
      TDQO=0.
      TLQI=0.
      TLQO=0.
      TBQI=0.
      TBQO=0.
      THBQI=0.
      THBQO=0.
      PNQO=0.
      TPNQO=0.
      VNLQI=0.
      VNLQO=0.
      TNLQI=0.
      TNLQO=0.
      BNQI=0.
      BNQO=0.
      TBNQI=0.
      TBNQO=0.
      TER=0.
      RETURN
      END
```

```
      SUBROUTINE INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
C        READS PROBLEM SPECIFICATIONS, DEFINES AND INITIALIZES VARIABLES
         FOR BASIC VERSION OF PROGRAM THAT USES MICCG MATRIX SOLVER
      DIMENSION TITLE(20),G(1)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
C *** FORMAT LIST ***
    8 FORMAT (1H1)
   10 FORMAT (20A4)
   12 FORMAT (1H ,20A4)
   14 FORMAT (16I5)
   15 FORMAT (F10.0)
   16 FORMAT (1H0,54HNO. OF ELEMENTS (NELS).............................
     1 = ,I5/
     $1H ,54HNO. OF NODES (NNDS)..................................... = ,I5/
     $1H ,54HMAX. NO. OF TIME STEPS PER STRESS PERIOD (MXSTPS).. = ,I5/
     $1H ,54HNO. OF STRESS PERIODS (NPER)............................ = ,I5/
     $1H ,54HNO. OF ZONES (NZNS)..................................... = ,I5/
     $1H ,54HNO. OF POINT FLOWS (NWELS).............................. = ,I5/
     $1H ,54HNO. OF CAUCHY-TYPE BOUNDARY ELEMENT SIDES (NQBND).. = ,I5/
     $1H ,54HNO. OF CAUCHY-TYPE BOUNDARY ZONES (NBCZ)............ = ,I5/
     $1H ,54HNO. OF SPECIFIED HEADS (NHDS)........................... = ,I5/
     $1H ,54HMAXIMUM CONDENSED BAND WIDTH (MBWC)..................... = ,I5/
     $1H ,54HMAXIMUM NO. OF ITERATIONS FOR MICCG (NIT)........... = ,I5/
     $1H ,54HCLOSURE TOLERANCE FOR MICCG (TOL)..................... =
     $,G11.5)
   18 FORMAT (33H0DIMENSION OF G MUST BE AT LEAST ,I6)
C *** DEFINE FILE NUMBERS ***
      IIN=50
      IOUT=60
      ITA=55
      ITB=56
C *** READ AND WRITE PROBLEM TITLE ***
      WRITE(IOUT,8)
      DO 20 I=1,3
      READ(IIN,10) (TITLE(K),K=1,20)
      WRITE(IOUT,12) (TITLE(K),K=1,20)
   20 CONTINUE
C *** READ AND WRITE PROBLEM SPECIFICATIONS ***
      READ(IIN,14) NELS,NNDS,MXSTPS,NPER,NZNS,NWELS,NQBND,NBCZ,NHDS,MBWC
     1,NIT
      READ(IIN,15) TOL
      WRITE(IOUT,16) NELS,NNDS,MXSTPS,NPER,NZNS,NWELS,NQBND,NBCZ,NHDS
     1,MBWC,NIT,TOL
C *** DEFINE ADDRESSES OF ARRAYS WITHIN G ***
      NA=(MBWC+1)*NNDS
      NB=2*NELS
      NC=NA+(MBWC+3)*(NNDS-NHDS)
```

```
ND=(MBWC-1)*NNDS
NE=4*NELS
NF=NNDS-NHDS
NG=NF
IF(NELS.GT.NF) NF=NELS
ISUM=1
IAA=ISUM
ISUM=ISUM+NA
IARA=ISUM
ISUM=ISUM+NB
INDA=ISUM
ISUM=ISUM+NE
IXGA=ISUM
ISUM=ISUM+NNDS
IYGA=ISUM
ISUM=ISUM+NNDS
IAFA=NA+1
IXA=IAFA+NG*MBWC
IPA=IXA+NG
IRA=IPA+NG
IF(NC.GE.ISUM) ISUM=NC+1
IQA=ISUM
ISUM=ISUM+NNDS
IBA=ISUM
ISUM=ISUM+NF
IHA=ISUM
ISUM=ISUM+NNDS
IHRA=ISUM
ISUM=ISUM+NNDS
IHBA=ISUM
ISUM=ISUM+NHDS
IALA=ISUM
ISUM=ISUM+NQBND
IQBA=ISUM
ISUM=ISUM+NQBND
ICKA=ISUM
ISUM=ISUM+NQBND
ICLA=ISUM
ISUM=ISUM+NQBND
IHKA=ISUM
ISUM=ISUM+NQBND
IHLA=ISUM
ISUM=ISUM+NQBND
IDZA=ISUM
ISUM=ISUM+NBCZ
IDSA=ISUM
ISUM=ISUM+NBCZ
IDTA=ISUM
ISUM=ISUM+MXSTPS
IJPA=ISUM
ISUM=ISUM+ND
INA=ISUM
ISUM=ISUM+NNDS+1
IKA=ISUM
ISUM=ISUM+NQBND
```

```
      ILA=ISUM
      ISUM=ISUM+NQBND
C *** WRITE SIZE OF G ***
      WRITE(IOUT,18) ISUM
C *** INITIALIZE G ***
      DO 30 I=1,ISUM
   30 G(I)=0.
C *** INITIALIZE TIME AND FLOW-BALANCE COMPONENTS ***
      TIME=0.
      TSA=0.
      TWQI=0.
      TWQO=0.
      TDQI=0.
      TDQO=0.
      TLQI=0.
      TLQO=0.
      TBQI=0.
      TBQO=0.
      THBQI=0.
      THBQO=0.
      PNQO=0.
      TPNQO=0.
      VNLQI=0.
      VNLQO=0.
      TNLQI=0.
      TNLQO=0.
      BNQI=0.
      BNQO=0.
      TBNQI=0.
      TBNQO=0.
      TER=0.
      RETURN
      END
```

```
      SUBROUTINE MASBAL(H,DHB,HR,HK,HL,QBND,CFDK,CFDL,A,Q,B,R,VQK,VQL,DT
     1,JPT,IN,KQB,LQB)
C         COMPUTES FLOW BALANCE FOR BASIC, LINEAR VERSION OF PROGRAM
      DIMENSION H(1),DHB(1),HR(1),HK(1),HL(1),QBND(1),CFDK(1),CFDL(1)
     1,A(1),Q(1),B(1),R(1),VQK(1),VQL(1)
      DIMENSION JPT(1),IN(1),KQB(1),LQB(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** INITIALIZE FLOW-BALANCE COMPONENTS ***
      IF(NHDS.LE.0) GO TO 15
      DO 10 I=1,NHDS
   10 R(I)=0.
   15 SA=0.
      VLQI=0.
      VLQO=0.
      MBM1=MBWC-1
      IW=MBWC+1
      NC=1
      NP=0
      DTM=3./(2.*DT)
C *** COMPUTE TOTAL ACCUMULATION IN STORAGE PER UNIT TIME AND LEAKAGE
C     ACROSS AN AQUITARD PER UNIT TIME, AND ADD THESE CONTRIBUTIONS
C     AND Q INTO THE NODAL FLUX, R, AT SPECIFIED-HEAD NODES ***
      DO 90 I=1,NNDS
      NVL=NC+1
      ND=NVL
      TMPB=A(NVL)*(HR(I)-H(I))
      IF(TMPB.GT.0.) GO TO 20
      VLQO=VLQO+TMPB
      GO TO 25
   20 VLQI=VLQI+TMPB
   25 K=IN(I)
      IF(K.GT.0) GO TO 35
      K=-K
      TMPA=A(NC)*DTM*DHB(K)
      SA=SA+TMPA
      R(K)=R(K)-Q(I)+TMPA-TMPB
      DO 30 J=1,MBM1
      L=JPT(NP+J)
      IF(L.GT.NNDS) GO TO 80
      TMPA=A(ND+J)*(H(L)-H(I))
      M=-IN(L)
      IF(M.GT.0) R(M)=R(M)-TMPA
   30 R(K)=R(K)+TMPA
      GO TO 80
   35 SA=SA+A(NC)*DTM*B(K)
      DO 70 J=1,MBM1
      L=JPT(NP+J)
      IF(L.GT.NNDS) GO TO 80
      M=-IN(L)
      IF(M.GT.0) R(M)=R(M)+A(ND+J)*(H(I)-H(L))
   70 CONTINUE
   80 NC=NC+IW
   90 NP=NP+MBM1
C *** ADD HEAD-DEPENDENT FLUX INTO TOTAL INFLOW OR OUTFLOW PER UNIT TIME
C     ACROSS CAUCHY-TYPE BOUNDARIES AND INTO R ***
```

```
      BQI=0.
      BQO=0.
      IF(NQBND.LE.0) GO TO 140
      DO 130 I=1,NQBND
      TMPA=0.
      TMPB=0.
      K=KQB(I)
      IF(K.LT.0) GO TO 95
      L=LQB(I)
      M=-IN(K)
      N=-IN(L)
      TMPA=CFDK(I)*(HK(I)-H(K))
      TMPB=CFDL(I)*(HL(I)-H(L))
      IF(M.GT.0) R(M)=R(M)-TMPA
      IF(N.GT.0) R(N)=R(N)-TMPB
   95 TMPA=QBND(I)*CFDK(I)+TMPA
      VQK(I)=TMPA
      IF(TMPA.GT.0.) GO TO 100
      BQO=BQO+TMPA
      GO TO 110
  100 BQI=BQI+TMPA
  110 TMPB=QBND(I)*CFDL(I)+TMPB
      VQL(I)=TMPB
      IF(TMPB.GT.0.) GO TO 120
      BQO=BQO+TMPB
      GO TO 130
  120 BQI=BQI+TMPB
  130 CONTINUE
C *** COMPUTE TOTAL FLOW IMBALANCE PER UNIT TIME ***
  140 ER=SA-WQI-WQO-DQI-DQO-VLQI-VLQO-BQI-BQO
      RETURN
      END
```

```
      SUBROUTINE MASOUT(R,DT,ISTP)
C        COMPUTES TOTAL FLOW BALANCE ACROSS SPECIFIED-HEAD BOUNDARIES
C        AND WRITES TOTAL FLOW-BALANCE COMPONENTS
      DIMENSION R(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
C *** FORMAT LIST ***
    2 FORMAT (1H1,20X,21HWATER BALANCE SUMMARY/1H0,12X,35HVOLUMETRIC RAT
     1ES FOR TIME STEP NO. ,I3)
    4 FORMAT (56H ACCUMULATION OF WATER IN STORAGE..................... =
     1 ,G11.5/
     $56H RECHARGE FROM POINT SOURCES.......................... = ,G11.5/
     $56H DISCHARGE FROM POINT SINKS........................... = ,G11.5/
     $56H RECHARGE FROM DISTRIBUTED SOURCES.................... = ,G11.5/
     $56H DISCHARGE FROM DISTRIBUTED SINKS..................... = ,G11.5/
     $56H RECHARGE FROM STEADY OR TRANSIENT LEAKAGE............ = ,G11.5/
     $56H DISCHARGE FROM STEADY OR TRANSIENT LEAKAGE........... = ,G11.5/
     $56H RECHARGE ACROSS CAUCHY-TYPE BOUNDARIES............... = ,G11.5/
     $56H DISCHARGE ACROSS CAUCHY-TYPE BOUNDARIES.............. = ,G11.5/
     $56H RECHARGE ACROSS SPECIFIED-HEAD BOUNDARIES............ = ,G11.5/
     $56H DISCHARGE ACROSS SPECIFIED-HEAD BOUNDARIES........... = ,G11.5/
     $56H DISCHARGE FROM NONLINEAR POINT SINKS................. = ,G11.5/
     $56H RECHARGE FROM NONLINEAR STEADY LEAKAGE............... = ,G11.5/
     $56H DISCHARGE FROM NONLINEAR STEADY LEAKAGE.............. = ,G11.5/
     $56H RECHARGE FROM NONLINEAR CAUCHY-TYPE BOUNDARIES....... = ,G1
     $56H DISCHARGE FROM NONLINEAR CAUCHY-TYPE BOUNDARIES...... = ,G1
     $56H FLOW IMBALANCE....................................... = ,G1
    6 FORMAT (1H0,10X,43HTOTAL VOLUMES SINCE BEGINNING OF SIMULATION)
C *** COMPUTE TOTAL INFLOW OR OUTFLOW PER UNIT TIME ACROSS SPECIFIED-
C     HEAD BOUNDARIES ***
      HBQO=0.
      HBQI=0.
      IF(NHDS.LE.0) GO TO 170
      DO 160 I=1,NHDS
      IF(R(I).GT.0.) GO TO 150
      HBQO=HBQO+R(I)
      GO TO 160
  150 HBQI=HBQI+R(I)
  160 CONTINUE
C *** MODIFY FLOW IMBALANCE AND COMPUTE TOTAL INFLOW OR OUTFLOW
C     SINCE T=0 ACROSS SPECIFIED-HEAD BOUNDARIES ***
      ER=ER-HBQO-HBQI
      THBQO=THBQO+HBQO*DT
      THBQI=THBQI+HBQI*DT
C *** WRITE TOTAL INFLOWS AND OUTFLOWS PER UNIT TIME ***
  170 WRITE(IOUT,2) ISTP
      WRITE(IOUT,4) SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,HBQI,HBQO
     1,PNQO,VNLQI,VNLQO,BNQI,BNQO,ER
C *** COMPUTE TOTAL INFLOWS AND OUTFLOWS SINCE T=0 ***
      TSA=TSA+SA*DT
      TWQI=TWQI+WQI*DT
```

```
      TWQO=TWQO+WQO*DT
      TDQI=TDQI+DQI*DT
      TDQO=TDQO+DQO*DT
      TLQI=TLQI+VLQI*DT
      TLQO=TLQO+VLQO*DT
      TBQI=TBQI+BQI*DT
      TBQO=TBQO+BQO*DT
      TER=TER+ER*DT
      WRITE(IOUT,6)
C *** WRITE TOTAL INFLOWS AND OUTFLOWS SINCE T=0 ***
      WRITE(IOUT,4) TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TPNQO,TNLQI,TNLQO,TBNQI,TBNQO,TER
      RETURN
      END
```

```
      SUBROUTINE MBALCB(H,DHB,HR,HK,HL,QBND,CFDK,CFDL,A,Q,B,R,VQK,VQL
     1,CH,CBQ,GMA,DT,JPT,IN,KQB,LQB)
C         COMPUTES FLOW BALANCE FOR BASIC, LINEAR VERSION OF PROGRAM
C         WITH TRANSIENT LEAKAGE
      DIMENSION H(1),DHB(1),HR(1),HK(1),HL(1),QBND(1),CFDK(1),CFDL(1)
     1,A(1),Q(1),B(1),R(1),VQK(1),VQL(1),CH(1),CBQ(1),GMA(1)
      DIMENSION JPT(1),IN(1),KQB(1),LQB(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** INITIALIZE FLOW-BALANCE COMPONENTS ***
      IF(NHDS.LE.0) GO TO 15
      DO 10 I=1,NHDS
   10 R(I)=0.
   15 SA=0.
      VLQI=0.
      VLQO=0.
      MBM1=MBWC-1
      IW=MBWC+1
      NC=1
      NP=0
      N=0
      DTM=3./(2.*DT)
C *** COMPUTE TOTAL ACCUMULATION IN STORAGE PER UNIT TIME AND LEAKAGE
C     ACROSS AN AQUITARD PER UNIT TIME, AND ADD THESE CONTRIBUTIONS
C     AND Q INTO THE NODAL FLUX, R, AT SPECIFIED-HEAD NODES ***
      DO 90 I=1,NNDS
      NVL=NC+1
      ND=NVL
      TMPB=A(NVL)*(HR(I)-H(I))
      K=IN(I)
      IF(K.GT.0) GO TO 35
      K=-K
      TMPA=A(NC)*DTM*DHB(K)
      SA=SA+TMPA
      IF(GMA(I).LT.1.E-30) GO TO 20
      N=N+1
      TMPB=TMPB+CBQ(N)-CH(N)*DHB(K)
   20 R(K)=R(K)-Q(I)+TMPA-TMPB
      DO 30 J=1,MBM1
      L=JPT(NP+J)
      IF(L.GT.NNDS) GO TO 75
      TMPA=A(ND+J)*(H(L)-H(I))
      M=-IN(L)
      IF(M.GT.0) R(M)=R(M)-TMPA
   30 R(K)=R(K)+TMPA
      GO TO 75
   35 SA=SA+A(NC)*DTM*B(K)
      IF(GMA(I).LT.1.E-30) GO TO 40
      N=N+1
      TMPB=TMPB+CBQ(N)-CH(N)*B(K)
   40 DO 70 J=1,MBM1
      L=JPT(NP+J)
      IF(L.GT.NNDS) GO TO 75
      M=-IN(L)
      IF(M.GT.0) R(M)=R(M)+A(ND+J)*(H(I)-H(L))
   70 CONTINUE
```

```
   75 IF(TMPB.GT.0.) GO TO 80
      VLQO=VLQO+TMPB
      GO TO 85
   80 VLQI=VLQI+TMPB
   85 NC=NC+IW
   90 NP=NP+MBM1
C *** ADD HEAD-DEPENDENT FLUX INTO TOTAL INFLOW OR OUTFLOW PER UNIT TIME
C     ACROSS CAUCHY-TYPE BOUNDARIES AND INTO R ***
      BQI=0.
      BQO=0.
      IF(NQBND.LE.0) GO TO 140
      DO 130 I=1,NQBND
      TMPA=0.
      TMPB=0.
      K=KQB(I)
      IF(K.LT.0) GO TO 95
      L=LQB(I)
      M=-IN(K)
      N=-IN(L)
      TMPA=CFDK(I)*(HK(I)-H(K))
      TMPB=CFDL(I)*(HL(I)-H(L))
      IF(M.GT.0) R(M)=R(M)-TMPA
      IF(N.GT.0) R(N)=R(N)-TMPB
   95 TMPA=QBND(I)*CFDK(I)+TMPA
      VQK(I)=TMPA
      IF(TMPA.GT.0.) GO TO 100
      BQO=BQO+TMPA
      GO TO 110
  100 BQI=BQI+TMPA
  110 TMPB=QBND(I)*CFDL(I)+TMPB
      VQL(1)=TMPB
      IF(TMPB.GT.0.) GO TO 120
      BQO=BQO+TMPB
      GO TO 130
  120 BQI=BQI+TMPB
  130 CONTINUE
C *** COMPUTE TOTAL FLOW IMBALANCE PER UNIT TIME ***
  140 ER=SA-WQI-WQO-DQI-DQO-VLQI-VLQO-BQI-BQO
      RETURN
      END
```

```
      SUBROUTINE MBALWT(H,DH,DHB,HR,HK,HL,QBND,CFDK,CFDL,A,Q,R,VQL,THK
     1,DTK,TOP,ASY,DT,JPT,IN,KQB,LQB)
C         COMPUTES FLOW BALANCE FOR BASIC, WATER-TABLE AQUIFER VERSIO
C         PROGRAM
      DIMENSION H(1),DH(1),DHB(1),HR(1),HK(1),HL(1),QBND(1),CFDK(1)
     1,CFDL(1),A(1),Q(1),R(1),VQL(1),THK(1),DTK(1),TOP(1),ASY(1)
      DIMENSION JPT(1),IN(1),KQB(1),LQB(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** COMPUTE DTK-VECTOR, AND COMPUTE CAPACITANCE AT STORAGE CONVERSION
C     NODES ***
      WF=2./3.
      NC=1
      IW=MBWC+1
      DO 14 I=1,NNDS
      DTK(I)=0.
      J=IN(I)
      IF(J.GT.0) GO TO 2
      DHC=DHB(-J)
      GO TO 4
    2 DHC=DH(J)
    4 HO=H(I)-DHC
      DHC=1.5*DHC
      HC=HO+DHC
      IF(HO.LT.TOP(I)) GO TO 6
      IF(HC.GE.TOP(I)) GO TO 14
      DTK(I)=HC-TOP(I)
      IF(THK(I)+DTK(I).LT.0.) DTK(I)=-THK(I)
      THET=(TOP(I)-HO)/DHC
      A(NC)=THET*(A(NC)-ASY(I))+ASY(I)
      GO TO 12
    6 IF(HC.LT.TOP(I)) GO TO 8
      DTK(I)=TOP(I)-HO
      THET=DTK(I)/DHC
      A(NC)=THET*(ASY(I)-A(NC))+A(NC)
      GO TO 12
    8 IF(THK(I)+DHC.GT.0.) GO TO 10
      DTK(I)=0.
      IF(THK(I).GT.0.) DTK(I)=-THK(I)
      GO TO 11
   10 DTK(I)=DHC
      IF(THK(I).LT.0.) DTK(I)=DHC+THK(I)
   11 A(NC)=ASY(I)
   12 DTK(I)=WF*DTK(I)
   14 NC=NC+IW
C *** INITIALIZE FLOW-BALANCE COMPONENTS ***
      IF(NHDS.LE.0) GO TO 18
      DO 16 I=1,NHDS
   16 R(I)=0.
   18 SA=0.
      VLQI=0.
      VLQO=0.
      C=1./16.
      MBM1=MBWC-1
      NC=1
      NP=0
```

```
      DTM=3./(2.*DT)
*** COMPUTE TOTAL ACCUMULATION IN STORAGE PER UNIT TIME AND LEAKAGE
      ACROSS AN AQUITARD PER UNIT TIME, AND ADD THESE CONTRIBUTIONS
C     AND Q INTO THE NODAL FLUX, R, AT SPECIFIED-HEAD NODES ***
      DO 90 I=1,NNDS
      NVL=NC+1
      ND=NVL
      TMPB=A(NVL)*(HR(I)-H(I))
      IF(TMPB.GT.0.) GO TO 20
      VLQO=VLQO+TMPB
      GO TO 25
   20 VLQI=VLQI+TMPB
   25 THKI=THK(I)
      IF(THKI.LT.0.) THKI=0.
      K=IN(I)
      IF(K.GT.0) GO TO 35
      K=-K
      TMPA=A(NC)*DTM*DHB(K)
      SA=SA+TMPA
      R(K)=R(K)-Q(I)+TMPA-TMPB
      DO 30 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 26,80,28
   26 THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      M=-M
      TMPA=(.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(L)-H(I))
     1+C*(DTK(I)+DTK(L))*(DHB(M)-DHB(K)))*A(ND+J)
      R(K)=R(K)+TMPA
      R(M)=R(M)-TMPA
      GO TO 30
   28 THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      R(K)=R(K)+(.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(L)-H(I))
     1+C*(DTK(I)+DTK(L))*(DH(M)-DHB(K)))*A(ND+J)
   30 CONTINUE
      GO TO 80
   35 SA=SA+A(NC)*DTM*DH(K)
      DO 70 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 60,80,70
   60 M=-M
      THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      R(M)=R(M)+(.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(I)-H(L))
     1+C*(DTK(I)+DTK(L))*(DH(K)-DHB(M)))*A(ND+J)
   70 CONTINUE
   80 NC=NC+IW
   90 NP=NP+MBM1
C *** ADD HEAD-DEPENDENT FLUX INTO TOTAL INFLOW OR OUTFLOW PER UNIT
C     TIME ACROSS CAUCHY-TYPE BOUNDARIES AND INTO R ***
      BQI=0.
```

```
      BQO=0.
      IF(NQBND.LE.0) GO TO 140
      DO 130 I=1,NQBND
      TMPA=0.
      TMPB=0.
      K=KQB(I)
      IF(K.LT.0) GO TO 95
      L=LQB(I)
      M=-IN(K)
      N=-IN(L)
      TMPA=CFDK(I)*(HK(I)-H(K))
      TMPB=CFDL(I)*(HL(I)-H(L))
      IF(M.GT.0) R(M)=R(M)-TMPA
      IF(N.GT.0) R(N)=R(N)-TMPB
   95 TMPA=QBND(I)*CFDK(I)+TMPA
      DTK(I)=TMPA
      IF(TMPA.GT.0.) GO TO 100
      BQO=BQO+TMPA
      GO TO 110
  100 BQI=BQI+TMPA
  110 TMPB=QBND(I)*CFDL(I)+TMPB
      VQL(I)=TMPB
      IF(TMPB.GT.0.) GO TO 120
      BQO=BQO+TMPB
      GO TO 130
  120 BQI=BQI+TMPB
  130 CONTINUE
C *** COMPUTE TOTAL FLOW IMBALANCE PER UNIT TIME ***
  140 ER=SA-WQI-WQO-DQI-DQO-VLQI-VLQO-BQI-BQO
      RETURN
      END
```

```
      SUBROUTINE MBWTCB(H,DH,DHB,HR,HK,HL,QBND,CFDK,CFDL,A,Q,R,VQL,THK
     1,DTK,TOP,ASY,CH,CBQ,GMA,DT,JPT,IN,KQB,LQB)
C        COMPUTES FLOW BALANCE FOR BASIC, WATER-TABLE AQUIFER VERSION OF
C        PROGRAM WITH TRANSIENT LEAKAGE
      DIMENSION H(1),DH(1),DHB(1),HR(1),HK(1),HL(1),QBND(1),CFDK(1)
     1,CFDL(1),A(1),Q(1),R(1),VQL(1),THK(1),DTK(1),TOP(1),ASY(1),CH(1)
     2,CBQ(1),GMA(1)
      DIMENSION JPT(1),IN(1),KQB(1),LQB(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** COMPUTE DTK-VECTOR, AND COMPUTE CAPACITANCE AT STORAGE CONVERSION
C     NODES ***
      WF=2./3.
      NC=1
      IW=MBWC+1
      DO 14 I=1,NNDS
      DTK(I)=0.
      J=IN(I)
      IF(J.GT.0) GO TO 2
      DHC=DHB(-J)
      GO TO 4
    2 DHC=DH(J)
    4 HO=H(I)-DHC
      DHC=1.5*DHC
      HC=HO+DHC
      IF(HO.LT.TOP(I)) GO TO 6
      IF(HC.GE.TOP(I)) GO TO 14
      DTK(I)=HC-TOP(I)
      IF(THK(I)+DTK(I).LT.0.) DTK(I)=-THK(I)
      THET=(TOP(I)-HO)/DHC
      A(NC)=THET*(A(NC)-ASY(I))+ASY(I)
      GO TO 12
    6 IF(HC.LT.TOP(I)) GO TO 8
      DTK(I)=TOP(I)-HO
      THET=DTK(I)/DHC
      A(NC)=THET*(ASY(I)-A(NC))+A(NC)
      GO TO 12
    8 IF(THK(I)+DHC.GT.0.) GO TO 10
      DTK(I)=0.
      IF(THK(I).GT.0.) DTK(I)=-THK(I)
      GO TO 11
   10 DTK(I)=DHC
      IF(THK(I).LT.0.) DTK(I)=DHC+THK(I)
   11 A(NC)=ASY(I)
   12 DTK(I)=WF*DTK(I)
   14 NC=NC+IW
C *** INITIALIZE FLOW-BALANCE COMPONENTS ***
      IF(NHDS.LE.0) GO TO 18
      DO 16 I=1,NHDS
   16 R(I)=0.
   18 SA=0.
      VLQI=0.
      VLQO=0.
      C=1./16.
      MBM1=MBWC-1
      NC=1
```

```
      NP=0
      N=0
      DTM=3./(2.*DT)
C *** COMPUTE TOTAL ACCUMULATION IN STORAGE PER UNIT TIME AND LEAKAGE
C     ACROSS AN AQUITARD PER UNIT TIME, AND ADD THESE CONTRIBUTIONS
C     AND Q INTO THE NODAL FLUX, R, AT SPECIFIED-HEAD NODES ***
      DO 90 I=1,NNDS
      NVL=NC+1
      ND=NVL
      TMPB=A(NVL)*(HR(I)-H(I))
      THKI=THK(I)
      IF(THKI.LT.0.) THKI=0.
      K=IN(I)
      IF(K.GT.0) GO TO 35
      K=-K
      TMPA=A(NC)*DTM*DHB(K)
      SA=SA+TMPA
      IF(GMA(I).LT.1.E-30) GO TO 20
      N=N+1
      TMPB=TMPB+CBQ(N)-CH(N)*DHB(K)
   20 R(K)=R(K)-Q(I)+TMPA-TMPB
      DO 30 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 26,75,28
   26 THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      M=-M
      TMPA=(.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(L)-H(I))
     1+C*(DTK(I)+DTK(L))*(DHB(M)-DHB(K)))*A(ND+J)
      R(K)=R(K)+TMPA
      R(M)=R(M)-TMPA
      GO TO 30
   28 THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      R(K)=R(K)+(.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(L)-H(I))
     1+C*(DTK(I)+DTK(L))*(DH(M)-DHB(K)))*A(ND+J)
   30 CONTINUE
      GO TO 75
   35 SA=SA+A(NC)*DTM*DH(K)
      IF(GMA(I).LT.1.E-30) GO TO 40
      N=N+1
      TMPB=TMPB+CBQ(N)-CH(N)*DH(K)
   40 DO 70 J=1,MBM1
      L=JPT(NP+J)
      M=IN(L)
      IF(M) 60,75,70
   60 M=-M
      THKL=THK(L)
      IF(THKL.LT.0.) THKL=0.
      R(M)=R(M)+(.5*(DTK(I)+DTK(L)+THKI+THKL)*(H(I)-H(L))
     1+C*(DTK(I)+DTK(L))*(DH(K)-DHB(M)))*A(ND+J)
   70 CONTINUE
   75 IF(TMPB.GT.0.) GO TO 80
```

```
      VLQO=VLQO+TMPB
      GO TO 85
   80 VLQI=VLQI+TMPB
   85 NC=NC+IW
   90 NP=NP+MBM1
C *** ADD HEAD-DEPENDENT FLUX INTO TOTAL INFLOW OR OUTFLOW PER UNIT
C     TIME ACROSS CAUCHY-TYPE BOUNDARIES AND INTO R ***
      BQI=0.
      BQO=0.
      IF(NQBND.LE.0) GO TO 140
      DO 130 I=1,NQBND
      TMPA=0.
      TMPB=0.
      K=KQB(I)
      IF(K.LT.0) GO TO 95
      L=LQB(I)
      M=-IN(K)
      N=-IN(L)
      TMPA=CFDK(I)*(HK(I)-H(K))
      TMPB=CFDL(I)*(HL(I)-H(L))
      IF(M.GT.0) R(M)=R(M)-TMPA
      IF(N.GT.0) R(N)=R(N)-TMPB
   95 TMPA=QBND(I)*CFDK(I)+TMPA
      DTK(I)=TMPA
      IF(TMPA.GT.0.) GO TO 100
      BQO=BQO+TMPA
      GO TO 110
  100 BQI=BQI+TMPA
  110 TMPB=QBND(I)*CFDL(I)+TMPB
      VQL(I)=TMPB
      IF(TMPB.GT.0.) GO TO 120
      BQO=BQO+TMPB
      GO TO 130
  120 BQI=BQI+TMPB
  130 CONTINUE
C *** COMPUTE TOTAL FLOW IMBALANCE PER UNIT TIME ***
  140 ER=SA-WQI-WQO-DQI-DQO-VLQI-VLQO-BQI-BQO
      RETURN
      END
```

```
      SUBROUTINE MICCG(A,AF,X,P,R,B,TOL,JPT,IN)
C        SOLVES MATRIX EQUATION FOR HEAD CHANGE VECTOR, B, BY USING
C        THE MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHC
      DIMENSION A(1),AF(1),X(1),P(1),R(1),B(1)
      DIMENSION JPT(1),IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (23H0SOLUTION CONVERGED IN ,I4,11H ITERATIONS)
    4 FORMAT (32H0SOLUTION FAILED TO CONVERGE IN ,I4,11H ITERATIONS
     1,3X,7HRMAX = ,G11.5)
    6 FORMAT (1H0,14X,42HCURRENT VALUES OF CHANGE IN HYDRAULIC HEAD
     1/1H ,3(6H  NODE,8X,4HHEAD,7X))
      N=NNDS-NHDS
      MBM1=MBWC-1
C *** COMPUTE AUGMENTATION FACTOR, IF NEEDED, FOR DIAGONAL ELEMENTS OF
C     AF MATRIX, WHICH IS THE MATRIX TO BE FACTORED ***
      DEL=.9993333
   10 DEL=1.5*DEL-.499
C *** DEFINE AF MATRIX AS THE A MATRIX WITH THE DIAGONALS AUGMENTED
C     IF NECESSARY ***
      ID=1
      DO 20 I=1,N
      IF(A(ID).GT.1.E-20) GO TO 12
      A(ID)=1.E30
      B(I)=0.
   12 AF(ID)=DEL*A(ID)
      DO 15 J=1,MBM1
   15 AF(ID+J)=A(ID+J)
   20 ID=ID+MBWC
      NEM1=NEQ-1
C *** BEGIN FACTORIZATION OF AF INTO SPARSE PRODUCT OF UPPER TRIANGULAR,
C     DIAGONAL, AND LOWER TRIANGULAR MATRICES, ALL KEPT IN AF ***
      NP=0
      ID=1
      DO 90 I=1,NEM1
      IF(IN(I).LT.0) GO TO 90
C *** DEFINE PIVOT ***
      IF(AF(ID).LE.0.) GO TO 10
      PIV=1./AF(ID)
      DO 70 J=1,MBM1
      KJ=JPT(NP+J)
      K=IN(KJ)
      IF(K) 70,80,25
   25 KD=MBWC*(K-1)+1
C *** DEFINE ROW MULTIPLIER ***
      C=AF(ID+J)*PIV
C *** REDUCE DIAGONAL OF AF ***
      AF(KD)=AF(KD)-C*AF(ID+J)
      LBG=J+1
      IF(LBG.GT.MBM1) GO TO 60
      KJ=MBM1*(KJ-1)
      MBG=1
      DO 50 L=LBG,MBM1
      KL=JPT(NP+L)
      IF(IN(KL)) 50,60,30
```

```
   30 TMPA=C*AF(ID+L)
  *** SEARCH TO DETERMINE IF NODE KL LIES IN THE ROW BEING REDUCED ***
      DO 40 M=MBG,MBM1
      MS=M
      IF(JPT(KJ+M).EQ.KL) GO TO 45
   40 CONTINUE
C *** REDUCE APPROPRIATE DIAGONAL ELEMENTS BY SUBTRACTING FILL-IN IF
C     KL IS NOT IN THE ROW BEING REDUCED ***
      AF(KD)=AF(KD)-TMPA
      LD=MBWC*(IN(KL)-1)+1
      AF(LD)=AF(LD)-TMPA
      GO TO 50
C *** REDUCE OFF-DIAGONAL IF KL IS IN THE ROW BEING REDUCED ***
   45 AF(KD+MS)=AF(KD+MS)-TMPA
      MBG=MS
   50 CONTINUE
   60 AF(ID+J)=C
   70 CONTINUE
   80 ID=ID+MBWC
   90 NP=NP+MBM1
      IF(AF(ID).LE.0.) GO TO 10
C *** DEFINE INITIAL RESIDUAL VECTOR, R, AS THE RIGHT-HAND-SIDE VECTOR,
C     B, OF THE ORIGINAL MATRIX EQUATION, THEN ZERO OUT INITIAL HEAD
C     CHANGE VECTOR, B ***
      DO 95 I=1,N
      R(I)=B(I)
   95 B(I)=0.
C *** BEGIN CONJUGATE GRADIENT ITERATION LOOP ***
      DO 270 KNT=1,NIT
      IT=KNT
C *** READ RESIDUAL VECTOR, R, INTO X ***
      DO 100 I=1,N
  100 X(I)=R(I)
C *** REDUCE RESIDUAL VECTOR, X ***
      ID=1
      NP=0
      DO 130 I=1,NEM1
      M=IN(I)
      IF(M.LT.0) GO TO 130
      DO 110 J=1,MBM1
      K=JPT(NP+J)
      K=IN(K)
      IF(K) 110,120,105
  105 X(K)=X(K)-AF(ID+J)*X(M)
  110 CONTINUE
  120 X(M)=X(M)/AF(ID)
      ID=ID+MBWC
  130 NP=NP+MBM1
C *** BACK-SOLVE FOR FIRST-ORDER DISPLACEMENT VECTOR, X ***
      X(N)=X(N)/AF(ID)
      K=NEQ
      DO 150 I=1,NEM1
      NP=NP-MBM1
      K=K-1
      M=IN(K)
```

```
      IF(M.LT.0) GO TO 150
      ID=ID-MBWC
      DO 140 J=1,MBM1
      L=JPT(NP+J)
      L=IN(L)
      IF(L) 140,150,135
  135 X(M)=X(M)-AF(ID+J)*X(L)
  140 CONTINUE
  150 CONTINUE
C *** FORM INNER PRODUCT OF RESIDUAL AND X VECTORS ***
      RX=0.
      DO 160 I=1,N
  160 RX=RX+R(I)*X(I)
      IF(IT.EQ.1) GO TO 180
C *** COMPUTE ITERATION PARAMETER, BETA ***
      S=RX/RXO
C *** COMPUTE SECOND-ORDER DISPLACEMENT VECTOR, P ***
      DO 170 I=1,N
  170 P(I)=X(I)+S*P(I)
      GO TO 200
  180 DO 190 I=1,N
  190 P(I)=X(I)
  200 DO 210 I=1,N
  210 X(I)=0.
C *** COMPUTE PRODUCT OF MATRIX A AND VECTOR P, AND KEEP PRODUCT
C     IN X ***
      ID=1
      NP=0
      DO 240 I=1,NEQ
      M=IN(I)
      IF(M.LT.0) GO TO 240
      X(M)=X(M)+A(ID)*P(M)
      DO 220 J=1,MBM1
      K=JPT(NP+J)
      K=IN(K)
      IF(K) 220,230,215
  215 X(M)=X(M)+A(ID+J)*P(K)
      X(K)=X(K)+A(ID+J)*P(M)
  220 CONTINUE
  230 ID=ID+MBWC
  240 NP=NP+MBM1
C *** FORM INNER PRODUCT OF P AND X VECTORS ***
      PX=0.
      DO 250 I=1,N
  250 PX=PX+P(I)*X(I)
C *** COMPUTE ITERATION PARAMETER, ALPHA ***
      W=1.
      IF(PX.GT.0.) W=RX/PX
C *** COMPUTE NEW HEAD CHANGE VECTOR, B, AND NEW RESIDUAL VECTOR, R,
C     THEN TEST FOR CONVERGENCE ***
      PMAX=0.
      RMAX=0.
      ID=1
      DO 260 I=1,N
      TMPA=W*P(I)
      B(I)=B(I)+TMPA
```

```
      R(I)=R(I)-W*X(I)
      TMPA=ABS(TMPA)
      IF(TMPA.GT.PMAX) PMAX=TMPA
      TMPA=ABS(R(I)/A(ID))
      IF(TMPA.GT.RMAX) RMAX=TMPA
  260 ID=ID+MBWC
      IF(PMAX.LE.TOL.AND.RMAX.LE.TOL) GO TO 280
      RXO=RX
  270 CONTINUE
C *** WRITE MAXIMUM SCALED RESIDUAL AND CURRENT HEAD CHANGE VECTOR IF
C     SOLUTION DID NOT CONVERGE ***
      WRITE(IOUT,4) NIT,RMAX
      WRITE(IOUT,6)
      CALL PRTOA(B,N)
      STOP
C *** WRITE NUMBER OF ITERATIONS IF SOLUTION CONVERGED ***
  280 WRITE(IOUT,2) IT
      RETURN
      END
```

```
      SUBROUTINE NXTPD(DELT,JPER)
C        READS TIME-STEP DATA FOR NEXT PUMPING PERIOD
      DIMENSION DELT(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,W
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT(16I5)
    4 FORMAT(8F10.0)
    6 FORMAT(1H0,19X,14HSTRESS PERIOD ,I3,19H:   TIME STEP SIZES
     1/1H ,3X,3(3HNO.,6X,7HDELTA T,9X))
C *** READ NUMBER OF TIME STEPS AND TIME-STEP NUMBERS WHERE VALUES OF
C     TIME-VARIANT QUANTITIES ARE CHANGED ***
      READ(IIN,2) NTMP,NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
C *** READ AND WRITE TIME-STEP SIZES IF DIFFERENT FROM LAST PERIOD ***
      IF(NTMP.LE.0) GO TO 10
      NSTEPS=NTMP
      READ(IIN,4) (DELT(I),I=1,NSTEPS)
   10 WRITE(IOUT,6) JPER
      CALL PRTOA(DELT,NSTEPS)
      RETURN
      END
```

```
      SUBROUTINE PRTCBV(VALA,VALB,DT,NVLA,NVLB,INZ,INS,NBZ)
C        PRINTS OUT NODAL VOLUMETRIC FLOW RATES FROM (NONLINEAR) CAUCHY-
C        TYPE BOUNDARIES AND SUMS NODAL RATES BY ZONE.  FLOW RATES FOR
C        NODES K AND L ON BOUNDARIES ARE STORED IN TWO VECTORS, WHICH
C        ARE WRITTEN OUT FOR EACH BOUNDARY SIDE.
      DIMENSION VALA(1),VALB(1)
      DIMENSION NVLA(1),NVLB(1),INZ(1),INS(1)
      COMMON/ITP/IIN,IOUT,ITA,ITB
   20 FORMAT(1H0,5HZONE ,I5/
     $56H VOLUMETRIC RECHARGE RATE..............................  = ,G11.5/
     $56H VOLUMETRIC DISCHARGE RATE.............................  = ,G11.5/
     $56H TOTAL RECHARGE VOLUME.................................  = ,G11.5/
     $56H TOTAL DISCHARGE VOLUME................................  = ,G11.5/
     $56H NET VOLUMETRIC FLOW RATE, POSITIVE FOR RECHARGE.....  = ,G11.5/
     $56H NET VOLUME, POSITIVE FOR ACCUMULATION.................  = ,G11.5)
   22 FORMAT(/1H0,4X
     1,48HVOLUMETRIC FLOW RATES BY BOUNDARY SIDE FOR ZONE ,I5/
     2/1H ,4X,8HBOUNDARY,4X,4HNODE,4X,4HNODE,5X,21HVOLUMETRIC FLOW RATES
     3/1H ,6X,4HSIDE,8X,1HK,7X,1HL,6X,6HNODE K,9X,6HNODE L)
   24 FORMAT(' ',4X,I5,6X,I5,3X,I5,2(4X,G11.5))
      J=0
      JB=1
      DO 150 NZ=1,NBZ
      KZ=INZ(NZ)
      NOS=INS(KZ)
      SMQI=0.
      SMQO=0.
      SMVI=0.
      SMVO=0.
      QNET=0.
      VNET=0.
      DO 130 I=1,NOS
      J=J+1
      VQK=VALA(J)
      VQL=VALB(J)
      IF(VQK.GT.0.) GO TO 100
      SMQO=SMQO+VQK
      GO TO 110
  100 SMQI=SMQI+VQK
  110 IF(VQL.GT.0.) GO TO 120
      SMQO=SMQO+VQL
      GO TO 130
  120 SMQI=SMQI+VQL
  130 CONTINUE
      SMVI=SMQI*DT
      SMVO=SMQO*DT
      QNET=SMQI+SMQO
      VNET=SMVI+SMVO
      WRITE(IOUT,20) KZ,SMQI,SMQO,SMVI,SMVO,QNET,VNET
      WRITE(IOUT,22) KZ
      L=JB
      DO 145 I=1,NOS
      NVA=NVLA(L)
      IF(NVA.LT.0) NVA=-NVA
      WRITE(IOUT,24) L,NVA,NVLB(L),VALA(L),VALB(L)
  145 L=L+1
```

```
      JB=L
  150 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PRTOA(VAL,NO)
C        WRITES VECTOR IN THREE COLUMNS
      DIMENSION VAL(1)
      COMMON/ITP/IIN,IOUT,ITA,ITB
      NR=NO/3
      IF(3*NR.NE.NO) NR=NR+1
      DO 10 K=1,NR
      WRITE(IOUT,20) (L,VAL(L),L=K,NO,NR)
   10 CONTINUE
      RETURN
   20 FORMAT (1H ,3(I5,6X,G11.5,3X))
      END


      SUBROUTINE PRTOB(VALA,VALB,NO)
C        WRITES TWO VECTORS IN TWO DOUBLE COLUMNS
      DIMENSION VALA(1),VALB(1)
      COMMON/ITP/IIN,IOUT,ITA,ITB
      NR=NO/2
      IF(2*NR.NE.NO) NR=NR+1
      DO 10 K=1,NR
      WRITE(IOUT,20) (L,VALA(L),VALB(L),L=K,NO,NR)
   10 CONTINUE
      RETURN
   20 FORMAT (1H ,2(I5,5X,G11.5,4X,G11.5,2X))
      END
```

```
      SUBROUTINE RDTP(A)
C        READS COEFFICIENT ARRAY FROM ITA
      DIMENSION A(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
      IE=(MBWC+1)*NNDS
      REWIND ITA
      READ(ITA) (A(I),I=1,IE)
      RETURN
      END
```

```
      SUBROUTINE SETB(A,IN,ND,IBND)
C        FINDS MATRIX BAND WIDTH FOR BAND MATRIX SOLVER AND WRITES
C        ARRAYS INTO FILES
      DIMENSION A(1)
      DIMENSION IN(1),ND(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (1H0,27HMATRIX BAND WIDTH (IBND) = ,I3)
      M=NNDS-NHDS
      NDC=0
      IBND=0
C *** FIND AND WRITE MAXIMUM MATRIX BAND WIDTH ***
      DO 30 I=1,NELS
      NE=2
      IF(ND(NDC+4).LE.0) NE=1
      JB=1
      JE=3
      DO 25 IE=1,NE
      MINND=M
      MAXND=0
      DO 20 J=JB,JE
      K=ND(NDC+1)
      IF(J.NE.5) K=ND(NDC+J)
      L=IN(K)
      IF(L.LT.0) GO TO 20
      IF(L.GT.MAXND) MAXND=L
      IF(L.LT.MINND) MINND=L
   20 CONTINUE
      NTMP=MAXND-MINND
      IF(NTMP.GT.IBND) IBND=NTMP
      JB=3
   25 JE=5
   30 NDC=NDC+4
      IBND=IBND+1
      WRITE(IOUT,2) IBND
C *** REWIND FILES ITA AND ITB ***
      REWIND ITA
      REWIND ITB
C *** WRITE COEFFICIENT ARRAY TO ITA ***
      IE=(MBWC+1)*NNDS
      WRITE(ITA) (A(I),I=1,IE)
C *** WRITE ELEMENT AREAS AND NODE POINTS TO ITB ***
      IB=IE+1
      IE=IB+2*NELS
      IB1=IE+1
      IE1=IB1+4*NELS
      WRITE(ITB) (A(I),I=IB,IE),(A(I),I=IB1,IE1)
      RETURN
      END
```

```
      SUBROUTINE SETCG(A)
C        WRITES ARRAYS INTO FILES
      DIMENSION A(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** REWIND FILES ITA AND ITB ***
      REWIND ITA
      REWIND ITB
C *** WRITE COEFFICIENT ARRAY TO ITA ***
      IE=(MBWC+1)*NNDS
      WRITE(ITA) (A(I),I=1,IE)
C *** WRITE ELEMENT AREAS AND NODE POINTS TO ITB ***
      IB=IE+1
      IE=IB+2*NELS
      IB1=IE+1
      IE1=IB1+4*NELS
      WRITE(ITB) (A(I),I=IB,IE),(A(I),I=IB1,IE1)
      RETURN
      END
```

```
      SUBROUTINE SWBDMP(B,DSMX,DSP,DSPO,DSPA,RP,ITER)
C        COMPUTES DAMPING PARAMETER AND DAMPS B-VECTOR
      DIMENSION B(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      N=NNDS-NHDS
      DSPA=0.
      DSPO=DSP
      DO 10 I=1,N
C *** COMPUTE MAXIMUM ABSOLUTE VALUE OF B-VECTOR ***
      TMPA=ABS(B(I))
      IF(TMPA.LE.DSPA) GO TO 10
      DSPA=TMPA
      DSP=B(I)
   10 CONTINUE
C *** COMPUTE DAMPING PARAMETER, RP ***
      IF(ITER.LE.1) GO TO 30
      SPR=DSP/(RP*DSPO)
      IF(SPR.LT.-1.) GO TO 20
      RP=(3.+SPR)/(3.+ABS(SPR))
      GO TO 30
   20 RP=.5/ABS(SPR)
   30 IF(RP*DSPA.GT.DSMX) RP=DSMX/DSPA
C *** DAMP B-VECTOR ***
      DO 40 I=1,N
   40 B(I)=RP*B(I)
      RETURN
      END
```

```
      SUBROUTINE SWFMCO(A,THK,TOP,JPT,IN)
C        READS AND WRITES INITIAL AQUIFER THICKNESS AND ELEVATION OF
C        AQUIFER TOP THEN FORMS INITIAL FLOW COEFFICIENTS
      DIMENSION A(1),THK(1),TOP(1)
      DIMENSION JPT(1),IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    4 FORMAT (8F10.0)
    5 FORMAT (16I5)
    6 FORMAT (1H0,24X,25HINITIAL AQUIFER THICKNESS/1H ,3(6H  NODE,5X
     1,9HTHICKNESS,5X))
    7 FORMAT (1H0,8X,49HINITIAL AQUIFER THICKNESS READ, OUTPUT SUPPRESSE
     1D)
    8 FORMAT (1H0,25X,24HELEVATION OF AQUIFER TOP/1H ,3(6H  NODE,5X
     1,9HELEVATION,5X))
    9 FORMAT (1H0,8X,45HAQUIFER TOP ELEVATION READ, OUTPUT SUPPRESSED)
C *** READ AND WRITE INITIAL AQUIFER THICKNESS, THK, AND ELEVATION OF
C     AQUIFER TOP, TOP ***
      READ(IIN,5) IPTK,IPTP
      READ(IIN,4) (THK(I),I=1,NNDS)
      IF(IPTK.GT.0) GO TO 16
      WRITE(IOUT,6)
      CALL PRTOA(THK,NNDS)
      GO TO 17
   16 WRITE(IOUT,7)
   17 READ(IIN,4) (TOP(I),I=1,NNDS)
      IF(IPTP.GT.0) GO TO 18
      WRITE(IOUT,8)
      CALL PRTOA(TOP,NNDS)
      GO TO 19
   18 WRITE(IOUT,9)
C *** COMPUTE INITIAL FLOW COEFFICIENTS, A ***
   19 IW=MBWC+1
      MBM1=MBWC-1
      ND=2
      NP=0
      DO 40 I=1,NNDS
      DO 20 J=1,MBM1
      K=JPT(NP+J)
      IF(IN(K).EQ.0) GO TO 30
   20 A(ND+J)=.5*(THK(K)+THK(I))*A(ND+J)
   30 ND=ND+IW
   40 NP=NP+MBM1
      RETURN
      END
```

```
      SUBROUTINE SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
C        READS PROBLEM SPECIFICATIONS, DEFINES AND INITIALIZES VARIABLES
         FOR STEADY-STATE WATER-TABLE AQUIFER VERSION OF PROGRAM
      DIMENSION G(1)
      COMMON/GDIM/ISUM
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (I5,2F10.0)
    4 FORMAT (1H0,54HMAX. NO. OF WATER-TABLE ITERATIONS (NITSW) ........
     $ = ,I5/
     $1H ,54HCLOSURE TOL. FOR WATER-TABLE ITERATIONS (TOLSW) ... =
     $,G11.5/
     $1H ,54HMAXIMUM ALLOWABLE DISPLACEMENT (DSMX) ............. =
     $,G11.5)
    6 FORMAT (1H0,43HSTEADY-STATE FLOW IN A WATER-TABLE AQUIFER:/1H
     1,38HNOW G MUST BE DIMENSIONED TO AT LEAST ,I6)
C *** READ AND WRITE PROBLEM SPECIFICATIONS ***
      READ(IIN,2) NITSW,TOLSW,DSMX
      WRITE(IOUT,4) NITSW,TOLSW,DSMX
C *** DEFINE INITIAL MAXIMUM HEAD DISPLACEMENT AND DAMPING PARAMETER ***
      DSP=DSMX
      RP=1.
C *** DEFINE ADDRESSES OF ARRAYS WITHIN G ***
      ITKA=ISUM
      ISUM=ISUM+NNDS
      ITPA=ISUM
      ISUM=ISUM+NNDS
C *** WRITE NEW SIZE OF G ***
      WRITE(IOUT,6) ISUM
C *** INITIALIZE NEW ELEMENTS OF G ***
      DO 10 I=ITKA,ISUM
   10 G(I)=0.
      RETURN
      END
```

```
      SUBROUTINE SWTHK(H,A,B,Q,THK,TOP,DSPA,TOLSW,JPT,IN,ITER)
C        COMPUTES CURRENT AQUIFER THICKNESS AND FLOW COEFFICIENT ARRAY
      DIMENSION H(1),A(1),B(1),Q(1),THK(1),TOP(1)
      DIMENSION JPT(1),IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
C *** FORMAT LIST ***
    1 FORMAT (1H0,5HNODE ,I5,23H WENT DRY ON ITERATION ,I3/1H ,3X
     1,38HPREDICTED AQUIFER THICKNESS AT NODE = ,G11.5/1H ,3X
     2,19HNET FLOW AT NODE = ,G11.5)
    2 FORMAT (1H ,3X,30HNET FLUX FROM NODE REDUCED TO ,G11.5)
C *** COMPUTE AQUIFER THICKNESS, THK ***
      DO 20 I=1,NNDS
      J=IN(I)
      IF(J.LT.0) GO TO 20
      HO=H(I)-B(J)
      IF(HO.LT.TOP(I)) GO TO 5
      IF(H(I).GT.TOP(I)) GO TO 20
      THK(I)=THK(I)+H(I)-TOP(I)
      GO TO 20
    5 IF(H(I).LT.TOP(I)) GO TO 10
      THK(I)=THK(I)-HO+TOP(I)
      GO TO 20
   10 THK(I)=THK(I)+B(J)
C *** REDUCE DISCHARGE AT NODES THAT ARE DRY ***
      IF(THK(I).GT.0.) GO TO 20
      WRITE(IOUT,1) I,ITER,THK(I),Q(I)
      IF(Q(I).GE.0..OR.B(J).GE.0.) GO TO 20
      Q(I)=.5*Q(I)
      WQO=WQO-Q(I)
      WRITE(IOUT,2) Q(I)
C *** INCREASE MAXIMUM ABSOLUTE DISPLACEMENT TO PERMIT
C     ANOTHER ITERATION USING REDUCED Q VECTOR ***
      DSPA=TOLSW
   20 CONTINUE
C *** COMPUTE FLOW COEFFICIENT ARRAY, A ***
      IW=MBWC+1
      MBM1=MBWC-1
      ND=2
      NP=0
      DO 50 I=1,NNDS
      THKI=THK(I)
      IF(THKI.LT.0.) THKI=0.
      DO 30 J=1,MBM1
      K=JPT(NP+J)
      IF(IN(K).EQ.0) GO TO 40
      THKK=THK(K)
      IF(THKK.LT.0.) THKK=0.
   30 A(ND+J)=.5*(THKK+THKI)*A(ND+J)
   40 ND=ND+IW
   50 NP=NP+MBM1
      RETURN
      END
```

```
      SUBROUTINE TKOUT(THK)
         WRITES SATURATED THICKNESS VECTOR, THK
      DIMENSION THK(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
    2 FORMAT (1H0,27X,19HSATURATED THICKNESS/1H ,3(6H  NODE,5X
     1,9HTHICKNESS,5X))
      WRITE(IOUT,2)
      CALL PRTOA(THK,NNDS)
      RETURN
      END
```

```
      SUBROUTINE VNBAL(H,DH,DHB,R,TOP,E,HS,DT,IN)
C        COMPUTES FLOW-BALANCE COMPONENTS FOR NONLINEAR STEADY-
C        LEAKAGE FUNCTIONS
      DIMENSION H(1),DH(1),DHB(1),R(1),TOP(1),E(1),HS(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
C *** INITIALIZE VOLUMETRIC-FLOW RATES TO ZERO ***
      VNLQI=0.
      VNLQO=0.
C *** COMPUTE WEIGHTING FACTORS FOR GALERKIN-IN-TIME FORMULATION ***
      CA=1./3.
      CB=2./3.
C *** FOR EACH NODE I ON THE LEAKAGE BOUNDARY: ***
      DO 170 I=1,NNDS
C *** (1) COMPUTE HEAD AT N LEVEL, HO, AND HEAD AT N+1 LEVEL, HC ***
      K=IN(I)
      IF(K.LT.0) GO TO 5
      DHC=DH(K)
      GO TO 10
    5 DHC=DHB(-K)
   10 HO=H(I)-DHC
      DHC=1.5*DHC
      HC=HO+DHC
C *** (2) DETERMINE FUNCTION TYPE BY SIGN OF E(I) ***
      IF(E(I)) 15,170,90
C ***
C *** DISCHARGE-ONLY TYPE ***
   15 EI=-E(I)
C *** (3) REPRESENT ELEVATIONS, AND HEAD AND ELEVATION DIFFERENCES, I
C        TEMPORARY (UNSUBSCRIPTED) VARIABLES ***
      ZE=HS(I)
      ZT=TOP(I)
      DEH=ZE-HO
      DHT=HO-ZT
      DET=ZE-ZT
C *** (4) COMPARE HEADS WITH CONTROLLING ELEVATIONS TO IDENTIFY LEAKAGE
C        CASE. COMPUTE HEAD DIFFERENCE, TMPA, FOR VOLUMETRIC FLOW RATES
C        FOR: ***
      IF(HO.GT.ZE) GO TO 30
      IF(HC.LT.ZE) GO TO 170
      IF(HC.LT.ZT) GO TO 20
C *** CASE (9) ***
      DMHT=DEH-DHT
      PHCF=1.-CA*(DMHT*DMHT+DHT*DEH)/(DHC*DHC)
      TMPA=PHCF*DET
      GO TO 150
C *** CASE (6) ***
   20 PHC=1.-DEH/DHC
      TMPA=CB*PHC*(ZE-HC)
      GO TO 150
   30 IF(HO.LT.ZT) GO TO 40
      IF(HC.LT.ZT) GO TO 70
C *** CASE (1) ***
      TMPA=DET
```

```
      GO TO 150
   40 IF(HC.GT.ZT) GO TO 60
      IF(HC.GT.ZE) GO TO 50
C *** CASE (5) ***
      PHE=DEH/DHC
      TMPA=CA*PHE*PHE*DEH
      GO TO 150
C *** CASE (4) ***
   50 TMPA=ZE-H(I)
      GO TO 150
C *** CASE (3) ***
   60 PHT=DHT/DHC
      TMPA=DET-CA*PHT*PHT*DHT
      GO TO 150
   70 IF(HC.GT.ZE) GO TO 80
C *** CASE (8) ***
      DMHT=DEH-DHT
      PHCF=(DMHT*DMHT+DHT*DEH)/(DHC*DHC)
      TMPA=CA*PHCF*DET
      GO TO 150
C *** CASE (2) ***
   80 PHC=1.+DHT/DHC
      TMPA=DET-CB*PHC*(HC-ZT)
      GO TO 150
C ***
C *** STEADY VERTICAL-LEAKAGE TYPE ***
   90 EI=E(I)
C *** (3) REPRESENT CONTROLLING HEAD, AQUIFER TOP ELEVATION, AND
C         DIFFERENCES BY TEMPORARY (UNSUBSCRIPTED) VARIABLES ***
      HA=HS(I)
      ZT=TOP(I)
      DHT=HO-ZT
      DAT=HA-ZT
C *** (4) COMPARE HEADS WITH AQUIFER TOP TO IDENTIFY LEAKAGE CASE.
C         COMPUTE HEAD DIFFERENCE, TMPA, FOR VOLUMETRIC-FLOW RATES
C         FOR: ***
      IF(HO.LT.ZT) GO TO 130
      IF(HC.LT.ZT) GO TO 120
C *** CASE (1) ***
      TMPA=HA-H(I)
      GO TO 150
C *** CASE (2) ***
  120 PHI=DHT/DHC
      TMPA=DAT-CA*PHI*PHI*DHT
      GO TO 150
  130 IF(HC.GT.ZT) GO TO 140
C *** CASE (4) ***
      TMPA=DAT
      GO TO 150
C *** CASE (3) ***
  140 PHC=1.+DHT/DHC
      TMPA=DAT-CB*PHC*(HC-ZT)
C ***
C *** (5) COMPARE VOLUMETRIC-FLOW RATES FOR BOTH TYPES OF NONLINEAR
C         STEADY-LEAKAGE FUNCTIONS ***
```

```
  150 TMPB=EI*TMPA
C *** INCLUDE VOLUMETRIC-FLOW RATE INTO FLOW-BALANCE EQUATION AT
C     SPECIFIED-HEAD NODE ***
      IF(K.LT.0) R(-K)=R(-K)-TMPB
C *** SUM VOLUMETRIC-FLOW RATES ACCORDING TO SIGN, POSITIVE FOR
C     INFLOW ***
      IF(TMPB.LT.0.) GO TO 160
      VNLQI=VNLQI+TMPB
      GO TO 170
  160 VNLQO=VNLQO+TMPB
  170 CONTINUE
C *** COMPUTE TOTAL VOLUMES RECHARGED (+) AND DISCHARGED (-) ACROSS
C     BOUNDARY FOR ENTIRE SIMULATION ***
      TNLQI=TNLQI+VNLQI*DT
      TNLQO=TNLQO+VNLQO*DT
C *** INCLUDE VOLUMETRIC-FLOW RATES IN THE FLOW IMBALANCE ***
      ER=ER-VNLQI-VNLQO
      RETURN
      END
```

```
      SUBROUTINE VNBLSS(H,R,TOP,E,HS,DT,IN)
C        COMPUTES FLOW-BALANCE COMPONENTS FOR NONLINEAR STEADY-LEAKAGE
C        FUNCTIONS FOR STEADY-STATE SIMULATIONS
      DIMENSION H(1),R(1),TOP(1),E(1),HS(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
C *** INITIALIZE VOLUMETRIC FLOW RATES TO ZERO ***
      VNLQI=0.
      VNLQO=0.
C *** FOR EACH NODE I ON THE LEAKAGE BOUNDARY: ***
      DO 90 I=1,NNDS
C *** (1) DETERMINE FUNCTION TYPE BY SIGN OF E(I) ***
      IF(E(I)) 10,90,40
C ***
C *** DISCHARGE-ONLY TYPE ***
   10 EI=-E(I)
C *** (2) REPRESENT HEADS AND ELEVATIONS BY TEMPORARY (UNSUBSCRIPTED)
C        VARIABLES ***
      ZE=HS(I)
      ZT=TOP(I)
      HI=H(I)
C *** (3) COMPARE HEADS WITH CONTROLLING ELEVATIONS TO IDENTIFY
C        LEAKAGE CASE. COMPUTE HEAD DIFFERENCE, TMPA, FOR VOLUMETRIC-
C        FLOW RATES FOR: ***
      IF(HI.GT.ZE) GO TO 20
C *** CASE (7) (NO FORMULATION) ***
      GO TO 90
   20 IF(HI.LT.ZT) GO TO 30
C *** CASE (1) ***
      TMPA=ZE-ZT
      GO TO 60
C *** CASE (4) ***
   30 TMPA=ZE-HI
      GO TO 60
C ***
C *** STEADY VERTICAL-LEAKAGE TYPE ***
   40 EI=E(I)
C *** (2) REPRESENT HEADS AND ELEVATIONS BY TEMPORARY (UNSUBSCRIPTED)
C        VARIABLES ***
      HA=HS(I)
      ZT=TOP(I)
      HI=H(I)
C *** (3) COMPARE HEAD WITH AQUIFER TOP TO IDENTIFY LEAKAGE CASE.
C        COMPARE HEAD DIFFERENCE, TMPA, FOR VOLUMETRIC-FLOW RATES
C        FOR: ***
      IF(HI.GT.ZT) GO TO 50
C *** CASE (4) ***
      TMPA=HA-ZT
      GO TO 60
C *** CASE (1) ***
   50 TMPA=HA-HI
C ***
C *** (4) COMPUTE VOLUMETRIC-FLOW RATE ***
   60 TMPB=EI*TMPA
```

```
C *** (5) SUM VOLUMETRIC-FLOW RATES ACCORDING TO SIGN, POSITIVE
C         FOR INFLOW ***
      IF(TMPB.LT.0.) GO TO 70
      VNLQI=VNLQI+TMPB
      GO TO 80
   70 VNLQO=VNLQO+TMPB
C *** (6) INCLUDE VOLUMETRIC-FLOW RATE INTO FLOW-BALANCE EQUATION AT
C         SPECIFIED-HEAD NODE ***
   80 K=-IN(I)
      IF(K.GT.0) R(K)=R(K)-TMPB
   90 CONTINUE
C *** COMPUTE TOTAL VOLUMES RECHARGED (+) AND DISCHARGED (-) ACROSS
C     BOUNDARY FOR SIMULATION ***
      TNLQI=TNLQI+VNLQI*DT
      TNLQO=TNLQO+VNLQO*DT
C *** INCLUDE VOLUMETRIC-FLOW RATES IN THE FLOW IMBALANCE ***
      ER=ER-VNLQI-VNLQO
      RETURN
      END
```

```
      SUBROUTINE VNCHG(HS,TIME,ISTP)
C        CHANGES VALUES OF CONTROLLING HEAD FOR NONLINEAR STEADY
C        VERTICAL-LEAKAGE FUNCTIONS FOR TIME-VARIANT BOUNDARY CONDITIONS
      DIMENSION HS(1)
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (16I5)
    4 FORMAT (1H1,11X,37HCHANGES IN VALUES OF CONTROLLING HEAD/1H
     1,10X,38HFOR NONLINEAR STEADY-LEAKAGE FUNCTIONS
     2/1H ,29H  BEGINNING ON TIME STEP NO. ,I4,4H AT ,G11.5
     3,11H TIME UNITS/1H ,34X,11HCONTROLLING/1H ,19X,4HNODE,15X,4HHEAD)
    6 FORMAT (I5,F10.0)
    8 FORMAT (1H ,17X,I5,13X,G11.5)
C *** READ NUMBER OF CONTROLLING HEADS TO BE CHANGED AND TIME-STEP
C     NUMBER FOR NEXT CHANGE ***
      READ(IIN,2) N,NVNCH
C *** WRITE HEADING ***
      WRITE(IOUT,4) ISTP,TIME
C *** READ AND WRITE VALUES OF NEW CONTROLLING HEADS ***
      DO 20 I=1,N
      READ(IIN,6) J,HS(J)
      WRITE(IOUT,8) J,HS(J)
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE VNCORR(H,DH,A,B,TOP,E,HS,IN)
C        FORMS COEFFICIENTS FOR NONLINEAR STEADY-LEAKAGE FUNCTIONS;
C        ADDS TERMS TO MATRIX EQUATION FOR THE CORRECTOR
      DIMENSION H(1),DH(1),A(1),B(1),TOP(1),E(1),HS(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      NME=1
C *** COMPUTE WEIGHTING FACTORS FOR GALERKIN-IN-TIME FORMULATION ***
      CA=1./3.
      CB=2./3.
C *** FOR EACH NODE I ON THE LEAKAGE BOUNDARY: ***
      DO 140 I=1,NEQ
      K=IN(I)
C *** (1) BYPASS SPECIFIED-HEAD NODES ***
      IF(K.LT.0) GO TO 140
C *** (2) COMPUTE HEAD AT N LEVEL, HO, CHANGE IN HEAD, DHP, AND
C        PREDICTED HEAD, HP ***
      HO=H(I)-DH(K)
      DHP=1.5*DH(K)
      HP=HO+DHP
C *** (3) DETERMINE FUNCTION TYPE BY SIGN OF E(I) ***
      IF(E(I)) 10,140,90
C ***
C *** DISCHARGE-ONLY TYPE ***
   10 EI=-E(I)
C *** (4) REPRESENT ELEVATIONS, AND HEAD AND ELEVATION DIFFERENCES, BY
C        TEMPORARY (UNSUBSCRIPTED) VARIABLES ***
      ZE=HS(I)
      ZT=TOP(I)
      DEH=ZE-HO
      DHT=HO-ZT
      DET=ZE-ZT
C *** (5) COMPARE HEADS WITH CONTROLLING ELEVATIONS TO IDENTIFY LEAKAGE
C        CASE. ADD LEAKAGE COEFFICIENT TO MAIN DIAGONAL, IF
C        APPLICABLE, AND COMPUTE HEAD DIFFERENCE, TMPA, FOR RIGHT-SIDE
C        TERM FOR: ***
      IF(HO.GT.ZE) GO TO 30
      IF(HP.LT.ZE) GO TO 140
      IF(HP.LT.ZT) GO TO 20
C *** CASE (9) ***
      DMHT=DEH-DHT
      PHCF=1.-CA*(DMHT*DMHT+DHT*DEH)/(DHP*DHP)
      TMPA=PHCF*DET
      GO TO 130
C *** CASE (6) ***
   20 PHC=1.-DEH/DHP
      A(NME)=A(NME)+PHC*EI
      TMPA=CB*PHC*(ZE-HP)
      GO TO 130
   30 IF(HO.LT.ZT) GO TO 40
      IF(HP.LT.ZT) GO TO 70
C *** CASE (1) ***
      TMPA=DET
      GO TO 130
   40 IF(HP.GT.ZT) GO TO 60
      IF(HP.GT.ZE) GO TO 50
```

```
C *** CASE (5) ***
      PHE=DEH/DHP
      TMPA=CA*PHE*PHE*DEH
      GO TO 130
C *** CASE (4) ***
   50 A(NME)=A(NME)+EI
      TMPA=ZE-H(I)
      GO TO 130
C *** CASE (3) ***
   60 PHT=DHT/DHP
      TMPA=DET-CA*PHT*PHT*DHT
      GO TO 130
   70 IF(HP.GT.ZE) GO TO 80
C *** CASE (8) ***
      DMHT=DEH-DHT
      PHCF=(DMHT*DMHT+DHT*DEH)/(DHP*DHP)
      TMPA=CA*PHCF*DET
      GO TO 130
C *** CASE (2) ***
   80 PHC=1.+DHT/DHP
      A(NME)=A(NME)+PHC*EI
      TMPA=DET-CB*PHC*(HP-ZT)
      GO TO 130
C ***
C *** STEADY VERTICAL-LEAKAGE TYPE ***
   90 EI=E(I)
C *** (4) REPRESENT CONTROLLING HEAD, AQUIFER TOP ELEVATION, AND
C         DIFFERENCES BY TEMPORARY (UNSUBSCRIPTED) VARIABLES ***
      HA=HS(I)
      ZT=TOP(I)
      DHT=HO-ZT
      DAT=HA-ZT
C *** (5) COMPARE HEADS WITH AQUIFER TOP TO IDENTIFY LEAKAGE CASE. ADD
C         LEAKAGE COEFFICIENTS TO MAIN DIAGONAL, IF APPLICABLE, AND
C         COMPUTE HEAD DIFFERENCE, TMPA, FOR RIGHT-SIDE TERM FOR: ***
      IF(HO.LT.ZT) GO TO 110
      IF(HP.LT.ZT) GO TO 100
C *** CASE (1) ***
      A(NME)=A(NME)+EI
      TMPA=HA-H(I)
      GO TO 130
C *** CASE (2) ***
  100 PHI=DHT/DHP
      TMPA=DAT-CA*PHI*PHI*DHT
      GO TO 130
  110 IF(HP.GT.ZT) GO TO 120
C *** CASE (4) ***
      TMPA=DAT
      GO TO 130
C *** CASE (3) ***
  120 PHC=1.+DHT/DHP
      A(NME)=A(NME)+PHC*EI
      TMPA=DAT-CB*PHC*(HP-ZT)
C ***
C *** (6) COMPUTE RIGHT-SIDE TERM AND ADD TO B-VECTOR FOR BOTH TYPES (
```

```
C          NONLINEAR STEADY-LEAKAGE FUNCTIONS ***
  130 B(K)=B(K)+EI*TMPA
      NME=NME+MBWC
  140 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE VNFMCO(H,AR,E,HS,ND,NVNZ)
         READS AND WRITES DATA AND COMPUTES LEAKAGE COEFFICIENTS
         FOR NONLINEAR STEADY-LEAKAGE FUNCTIONS
      DIMENSION H(1),AR(1),E(1),HS(1)
      DIMENSION ND(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    6 FORMAT (1H0,15X,14HPARAMETERS FOR/1H ,6X,32HNONLINEAR STEADY LEAKA
     1GE BY ZONE/1H ,10X,5HFIRST,5X,6HNO. OF,5X,11HCOEFFICIENT/1H
     2,6H  ZONE,3X,7HEL. NO.,3X,8HELEMENTS,7X,5HVALUE)
    7 FORMAT (1H0,8X,53HNONLINEAR STEADY-LEAKAGE DATA READ, OUTPUT SUPPR
     1ESSED)
    8 FORMAT (3I5,F10.0)
   10 FORMAT (1H ,I5,4X,I5,5X,I5,7X,G11.5)
   12 FORMAT (8F10.0)
   14 FORMAT (1H0,12X,46HCONTROL ELEVATION FOR NONLINEAR STEADY LEAKAGE
     1/1H ,2X,3(4HNODE,7X,5HVALUE,9X))
   15 FORMAT (1H0,8X,70HCONTROL ELEVATION FOR NONLINEAR STEADY LEAKAGE R
     1EAD, OUTPUT SUPPRESSED)
C *** READ INDICATOR VARIABLES FOR SUPPRESSING OUTPUT OF INITIAL VALUES
C     OF NONLINEAR STEADY-LEAKAGE FUNCTIONS ***
      READ(IIN,8) IPNV,IPHS
C *** FOR EACH ZONE SIMULATING NONLINEAR STEADY LEAKAGE: ***
      IF(IPNV.GT.0) GO TO 16
      WRITE(IOUT,6)
      GO TO 18
   16 WRITE(IOUT,7)
   18 DO 40 I=1,NVNZ
C *** (1) READ AND WRITE ZONE DATA ***
      READ(IIN,8) L,NBE,NO,VNCF
      IF(IPNV.EQ.0) WRITE (IOUT,10) L,NBE,NO,VNCF
C *** (2) COMPUTE ELEMENT AND NODE INDICES ***
      NEND=NBE+NO-1
      NDC=4*(NBE-1)+1
      NTE=2*(NBE-1)
C *** FOR EACH ELEMENT IN ZONE I: ***
      DO 30 J=NBE,NEND
      NE=1
C *** (1) CHECK FOR COMBINED INPUT OF NODE NUMBERS FOR TWO ELEMENTS ***
      IF(ND(NDC+3).GT.0) NE=2
C *** (2) COMPUTE LEAKAGE COEFFICIENT, EC, AND STORE IN E-VECTOR
C     BY NODE ***
      DO 20 IE=1,NE
      NA=ND(NDC)
      NB=ND(NDC+IE)
      NC=ND(NDC+IE+1)
      EC=VNCF*AR(NTE+IE)
      E(NA)=E(NA)+EC
      E(NB)=E(NB)+EC
      E(NC)=E(NC)+EC
   20 CONTINUE
      NTE=NTE+2
   30 NDC=NDC+4
   40 CONTINUE
C *** READ AND WRITE CONTROLLING HEAD (OR ELEVATION) ***
```

```
      READ(IIN,12) (HS(I),I=1,NNDS)
      IF(IPHS.GT.0) GO TO 42
      WRITE(IOUT,14)
      CALL PRTOA(HS,NNDS)
      RETURN
   42 WRITE(IOUT,15)
      RETURN
      END
```

```
      SUBROUTINE VNINIT(G,IECA,IHSA,NVNZ)
         READS DATA, ASSIGNS STORAGE, AND INITIALIZES STORAGE LOCATIONS
         TO ZERO FOR NONLINEAR STEADY-LEAKAGE FUNCTIONS
      DIMENSION G(1)
      COMMON/GDIM/ISUM
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (16I5)
    4 FORMAT (1H0,54HNO. OF NONLINEAR STEADY-LEAKAGE ZONES (NVNZ).......
     1 = ,I5)
    6 FORMAT (1H0,25HNONLINEAR STEADY LEAKAGE:/1H
     1,38HNOW G MUST BE DIMENSIONED TO AT LEAST ,I6)
C *** READ AND WRITE NUMBER OF NONLINEAR STEADY-LEAKAGE ZONES ***
      READ(IIN,2) NVNZ
      WRITE(IOUT,4) NVNZ
C *** DEFINE ADDRESSES OF NEW ARRAYS WITHIN G ***
      IECA=ISUM
      ISUM=ISUM+NNDS
      IHSA=ISUM
      ISUM=ISUM+NNDS
C *** WRITE SIZE OF G ***
      WRITE(IOUT,6) ISUM
C *** INITIALIZE NEW ELEMENTS OF G ***
      DO 10 I=IECA,ISUM
   10 G(I)=0.
      RETURN
      END
```

```
      SUBROUTINE VNPRED(H,A,B,TOP,E,HS,IN)
C        FORMS COEFFICIENTS FOR NONLINEAR STEADY-LEAKAGE FUNCTIONS;
C        ADDS TERMS TO MATRIX EQUATION FOR THE PREDICTOR
      DIMENSION H(1),A(1),B(1),TOP(1),E(1),HS(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      NME=1
C *** FOR EACH NODE I ON THE LEAKAGE BOUNDARY: ***
      DO 40 I=1,NEQ
      K=IN(I)
C *** (1) BYPASS SPECIFIED-HEAD NODES ***
      IF(K.LT.0) GO TO 40
C *** (2) DETERMINE FUNCTION TYPE BY SIGN OF E(I) ***
      IF(E(I)) 5,40,10
C ***
C *** DISCHARGE-ONLY TYPE ***
    5 EI=-E(I)
C *** COMPARE AQUIFER HEAD WITH CONTROLLING ELEVATIONS ***
      IF(H(I).LT.HS(I)) GO TO 35
      IF(H(I).GT.TOP(I)) GO TO 20
      GO TO 15
C ***
C *** STEADY VERTICAL-LEAKAGE TYPE ***
   10 EI=E(I)
C *** COMPARE HEAD WITH AQUIFER TOP ***
      IF(H(I).LT.TOP(I)) GO TO 20
C ***
C *** COMPUTE MAIN DIAGONAL TERM FOR: ***
C *** CASE (4) - DISCHARGE ONLY
C     CASE (1) - STEADY VERTICAL LEAKAGE ***
   15 A(NME)=A(NME)+EI
C *** COMPUTE HEAD DIFFERENCE, TMPA, FOR RIGHT-SIDE TERMS FOR: ***
C *** CASE (4) - DISCHARGE ONLY
C     CASE (1) - STEADY VERTICAL LEAKAGE ***
      TMPA=HS(I)-H(I)
      GO TO 30
C *** CASE (1) - DISCHARGE ONLY ***
C     CASE (4) - STEADY VERTICAL LEAKAGE ***
   20 TMPA=HS(I)-TOP(I)
C *** COMPUTE RIGHT-SIDE LEAKAGE COEFFICIENT AND ADD TO B-VECTOR ***
   30 B(K)=B(K)+EI*TMPA
   35 NME=NME+MBWC
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE WTCCHK(H,DH,B,TOP,IN,ISC)
         CHECKS TO SEE IF A PREDICTED STORAGE CONVERSION DID NOT TAKE
         PLACE OR IF AN UNPREDICTED CONVERSION DID TAKE PLACE
      DIMENSION H(1),DH(1),B(1),TOP(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      ISC=0
      DO 40 I=1,NEQ
      J=IN(I)
      IF(J.LT.0) GO TO 40
      HP=H(I)+.5*DH(J)
      IF(H(I)-DH(J).LT.TOP(I)) GO TO 20
      IF(HP.GE.TOP(I)) GO TO 30
   10 IF(HP+1.5*B(J).GE.TOP(I)) ISC=1
      GO TO 40
   20 IF(HP.LT.TOP(I)) GO TO 10
   30 IF(HP+1.5*B(J).LT.TOP(I)) ISC=1
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE WTFMCO(AR,THK,TOP,ASY,ND)
C        FORMS COEFFICIENTS FOR WATER-TABLE AQUIFER
      DIMENSION AR(1),THK(1),TOP(1),ASY(1)
      DIMENSION ND(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    4 FORMAT (8F10.0)
    5 FORMAT (16I5)
    6 FORMAT (1H0,24X,25HINITIAL AQUIFER THICKNESS/1H ,3(6H  NODE,5X
     1,9HTHICKNESS,5X))
    7 FORMAT (1H0,8X,49HINITIAL AQUIFER THICKNESS READ, OUTPUT SUPPRESSE
     1D)
    8 FORMAT (1H0,25X,24HELEVATION OF AQUIFER TOP/1H ,3(6H  NODE,5X
     1,9HELEVATION,5X))
    9 FORMAT (1H0,8X,45HAQUIFER TOP ELEVATION READ, OUTPUT SUPPRESSED)
   10 FORMAT (1H0,6H  ZONE,5X,3HN0.,5X,8HSPECIFIC/1H ,10X,4HELS.,7X
     1,5HYIELD)
   12 FORMAT (2I5,F10.0)
   14 FORMAT (1H ,I5,3X,I5,4X,G11.5)
C *** READ AND WRITE INITIAL AQUIFER THICKNESS ***
      READ(IIN,5) IPTK,IPTP
      READ(IIN,4) (THK(I),I=1,NNDS)
      IF(IPTK.GT.0) GO TO 16
      WRITE(IOUT,6)
      CALL PRTOA(THK,NNDS)
      GO TO 17
   16 WRITE(IOUT,7)
C *** READ AND WRITE INITIAL ELEVATION OF AQUIFER TOP ***
   17 READ(IIN,4) (TOP(I),I=1,NNDS)
      IF(IPTP.GT.0) GO TO 18
      WRITE(IOUT,8)
      CALL PRTOA(TOP,NNDS)
      GO TO 19
   18 WRITE(IOUT,9)
C *** INITIALIZE NODAL SPECIFIC YIELD COEFFICIENTS ***
   19 DO 20 I=1,NNDS
   20 ASY(I)=0.
C *** BEGIN ZONAL LOOP ***
      NDC=1
      NTE=0
      WRITE(IOUT,10)
      DO 50 IZ=1,NZNS
C *** READ AND WRITE ZONAL SPECIFIC YIELD DATA ***
      READ(IIN,12) KZ,NO,SY
      WRITE(IOUT,14) KZ,NO,SY
C *** BEGIN ELEMENT LOOP WITHIN ZONE ***
      DO 40 I=1,NO
      NE=1
      IF(ND(NDC+3).GT.0) NE=2
      DO 30 IE=1,NE
      NA=ND(NDC)
      NB=ND(NDC+IE)
      NC=ND(NDC+IE+1)
C *** COMPUTE NODAL SPECIFIC YIELD COEFFICIENTS ***
      TESY=SY*AR(NTE+IE)
```

```
      ASY(NA)=ASY(NA)+TESY
      ASY(NB)=ASY(NB)+TESY
30    ASY(NC)=ASY(NC)+TESY
      NTE=NTE+2
40    NDC=NDC+4
50    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE WTINIT(G,IDHA,ITKA,ITPA,ISYA)
C        DEFINES AND INITIALIZES VARIABLES FOR A WATER-TABLE AQUIFER
      DIMENSION G(1)
      COMMON/GDIM/ISUM
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/ITP/IIN,IOUT,ITA,ITB
C *** FORMAT LIST ***
    2 FORMAT (1H0,20HWATER-TABLE AQUIFER:/1H ,38HNOW G MUST BE DIMENSION
     1ED TO AT LEAST ,I6)
C *** DEFINE ADDRESSES OF ARRAYS WITHIN G ***
      IDHA=ISUM
      ISUM=ISUM+NNDS-NHDS
      ITKA=ISUM
      ISUM=ISUM+NNDS
      ITPA=ISUM
      ISUM=ISUM+NNDS
      ISYA=ISUM
      ISUM=ISUM+NNDS
C *** WRITE NEW SIZE OF G ***
      WRITE(IOUT,2) ISUM
C *** INITIALIZE NEW ELEMENTS OF G ***
      DO 10 I=ITKA,ISUM
   10 G(I)=0.
      RETURN
      END
```

```
      SUBROUTINE XTRPWT(H,DH,DHB,THK,TOP,IN,ISTP)
         EXTRAPOLATES HEADS AND THICKNESSES AT ALL NODES TO VALUES AT
         THE END OF THE TIME STEP AND ZEROES HEAD CHANGES
      DIMENSION H(1),DH(1),DHB(1),THK(1),TOP(1)
      DIMENSION IN(1)
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      DO 60 I=1,NNDS
      J=IN(I)
      IF(J.GT.0) GO TO 20
      J=-J
C *** DEFINE HEAD CHANGE, DHC, AND ZERO HEAD CHANGE VECTORS DHB
C     AND DH ***
      DHC=DHB(J)
      DHB(J)=0.
      GO TO 30
   20 DHC=DH(J)
      DH(J)=0.
   30 HO=H(I)-DHC
C *** EXTRAPOLATE HEAD FROM THE MEAN POINT IN THE TIME STEP TO THE END
C     OF THE TIME STEP ***
      H(I)=.5*DHC+H(I)
C *** EXTRAPOLATE THICKNESS FROM THE BEGINNING TO THE END OF THE TIME
C     STEP ***
      IF(HO.LT.TOP(I)) GO TO 40
      IF(H(I).GT.TOP(I)) GO TO 60
      THK(I)=THK(I)+H(I)-TOP(I)
      GO TO 60
   40 IF(H(I).LT.TOP(I)) GO TO 50
      THK(I)=THK(I)-HO+TOP(I)
      GO TO 60
   50 THK(I)=THK(I)+1.5*DHC
   60 CONTINUE
      RETURN
      END
```

List of Main Programs

```
C         *** LMFE1 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) CONFINED FLOW
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
        DIMENSION TITLE(20)
        COMMON/PRIME/G(5000)
        COMMON/GDIM/ISUM
        COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
       1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
        COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
        COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
        COMMON/ITP/IIN,IOUT,ITA,ITB
        COMMON/IPRN/IPND
        COMMON/IND/IRAD,IUNIT,ISTD
        COMMON/SCLE/SCALE
        COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
        COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
       1,THBQO,TER
        COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
        COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
        OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
       1,FORM='FORMATTED')
        OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
       1,FORM='FORMATTED')
        OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
        OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
        CALL INITB(TITLE,G,TIME,NBCZ)
        CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
       1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
       2,G(IDZA),G(IDSA),NBCZ)
        CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
       1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
        CALL SETB(G(IAA),G(INA),G(INDA),IBND)
        DO 20 JP=1,NPER
        JPER=JP
        CALL NXTPD(G(IDTA),JPER)
        DO 10 I=1,NSTEPS
        ISTP=I
        DT=G(IDTA+I-1)
        CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
       1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
       2,ISTP)
        CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
```

```
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      IF(ISTD.LT.1) CALL EXTRAP(G(IHA),G(IHBA),G(IBA),G(INA))
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** LMFE2 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) CONFINED FLOW
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETCG(G(IAA))
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      IF(ISTD.LT.1) CALL EXTRAP(G(IHA),G(IHBA),G(IBA),G(INA))
```

```
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** LMFE3 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) CONFINED FLOW
C         (2) TRANSIENT LEAKAGE
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL CBINIT(G,ICHA,ICQA,IDHRA,ICTQA,IGMA,IALFA,IACA,IBTA,IBCA
     1,NCBZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL CBFMCO(G(IAA),G(IARA),G(IGMA),G(IALFA),G(IACA),G(IBTA)
     1,G(IBCA),G(IYGA),G(INDA),NCBZ)
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NCBCH.EQ.ISTP) CALL CBCHG(G(IHRA),G(IDHRA),TIME,ISTP)
      CALL CBFMEQ(G(IAA),G(ICHA),G(ICQA),G(IDHRA),G(ICTQA),G(IGMA)
     1,G(IALFA),G(IACA),G(IBTA),G(IBCA),DT,G(INA))
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
```

```
      CALL CBADEQ(G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALCB(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),G(ICHA)
     2,G(ICQA),G(IGMA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL CBTQC(G(IBA),G(IHBA),G(ICTQA),G(IGMA),G(IALFA),G(IACA)
     1,DT,G(INA))
      IF(ISTD.LT.1) CALL EXTRAP(G(IHA),G(IHBA),G(IBA),G(INA))
      CALL CBHRXT(G(IHRA),G(IDHRA))
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** LMFE4 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) CONFINED FLOW
C         (2) TRANSIENT LEAKAGE
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE,
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL CBINIT(G,ICHA,ICQA,IDHRA,ICTQA,IGMA,IALFA,IACA,IBTA,IBCA
     1,NCBZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL CBFMCO(G(IAA),G(IARA),G(IGMA),G(IALFA),G(IACA),G(IBTA)
     1,G(IBCA),G(IYGA),G(INDA),NCBZ)
      CALL SETCG(G(IAA))
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NCBCH.EQ.ISTP) CALL CBCHG(G(IHRA),G(IDHRA),TIME,ISTP)
      CALL CBFMEQ(G(IAA),G(ICHA),G(ICQA),G(IDHRA),G(ICTQA),G(IGMA)
     1,G(IALFA),G(IACA),G(IBTA),G(IBCA),DT,G(INA))
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
```

```
      CALL CBADEQ(G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALCB(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),G(ICHA)
     2,G(ICQA),G(IGMA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL CBTQC(G(IBA),G(IHBA),G(ICTQA),G(IGMA),G(IALFA),G(IACA)
     1,DT,G(INA))
      IF(ISTD.LT.1) CALL EXTRAP(G(IHA),G(IHBA),G(IBA),G(INA))
      CALL CBHRXT(G(IHRA),G(IDHRA))
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE1 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
```

```
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE2 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL SETCG(G(IAA))
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
```

```
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE3 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         (3) TRANSIENT LEAKAGE
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZON
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL CBINIT(G,ICHA,ICQA,IDHRA,ICTQA,IGMA,IALFA,IACA,IBTA,IBCA
     1,NCBZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL CBFMCO(G(IAA),G(IARA),G(IGMA),G(IALFA),G(IACA),G(IBTA)
     1,G(IBCA),G(IYGA),G(INDA),NCBZ)
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NCBCH.EQ.ISTP) CALL CBCHG(G(IHRA),G(IDHRA),TIME,ISTP)
      CALL RDTP(G(IAA))
```

```
      CALL CBFMEQ(G(IAA),G(ICHA),G(ICQA),G(IDHRA),G(ICTQA),G(IGMA)
     1,G(IALFA),G(IACA),G(IBTA),G(IBCA),DT,G(INA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL CBADEQ(G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL CBADWT(G(IDHA),G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL CBADWT(G(IDHA),G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBWTCB(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),G(ICHA),G(ICQA),G(IGMA),DT,G(IJPA)
     3,G(INA),G(IKA),G(ILA))
      CALL CBTQC(G(IDHA),G(IHBA),G(ICTQA),G(IGMA),G(IALFA),G(IACA)
     1,DT,G(INA))
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL CBHRXT(G(IHRA),G(IDHRA))
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE4 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         (3) TRANSIENT LEAKAGE
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,MBW
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'\
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTE'
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZON.
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL CBINIT(G,ICHA,ICQA,IDHRA,ICTQA,IGMA,IALFA,IACA,IBTA,IBCA
     1,NCBZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL CBFMCO(G(IAA),G(IARA),G(IGMA),G(IALFA),G(IACA),G(IBTA)
     1,G(IBCA),G(IYGA),G(INDA),NCBZ)
      CALL SETCG(G(IAA))
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NCBCH.EQ.ISTP) CALL CBCHG(G(IHRA),G(IDHRA),TIME,ISTP)
      CALL RDTP(G(IAA))
```

```
      CALL CBFMEQ(G(IAA),G(ICHA),G(ICQA),G(IDHRA),G(ICTQA),G(IGMA)
     1,G(IALFA),G(IACA),G(IBTA),G(IBCA),DT,G(INA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL CBADEQ(G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL CBADWT(G(IDHA),G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL CBADWT(G(IDHA),G(IAA),G(IBA),G(ICHA),G(ICQA),G(IGMA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBWTCB(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),G(ICHA),G(ICQA),G(IGMA),DT,G(IJPA)
     3,G(INA),G(IKA),G(ILA))
      CALL CBTQC(G(IDHA),G(IHBA),G(ICTQA),G(IGMA),G(IALFA),G(IACA)
     1,DT,G(INA))
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL CBHRXT(G(IHRA),G(IDHRA))
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE5 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         (3) NONLINEAR STEADY LEAKAGE
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZON
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NVNCH.EQ.ISTP) CALL VNCHG(G(IHSA),TIME,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
```

```
     2,G(IKA),G(ILA))
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(ITPA),G(IECA),G(IHSA)
     1,DT,G(INA))
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE6 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         (3) NONLINEAR STEADY LEAKAGE
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTE'
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZON.
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      CALL SETCG(G(IAA))
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NVNCH.EQ.ISTP) CALL VNCHG(G(IHSA),TIME,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
```

```
     2,G(IKA),G(ILA))
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(ITPA),G(IECA),G(IHSA)
     1,DT,G(INA))
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE7 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         (3) NONLINEAR CAUCHY-TYPE BOUNDARIES AND (OR) NONLINEAR POINT
C             SINKS
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTE'
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTE
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
    7 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
     1,INSA,NBNC,NPNB,NLCZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
     2,NPNB,NLCZ)
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
```

```
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NGNCH.EQ.ISTP) CALL GNCHG(G(IHRK),G(IHRL),TIME,G(IKRA),G(ILRA)
     1,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL GNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA)
     2,G(IKPA),NBNC,NPNB)
      IF(NLCZ.LT.1) GO TO 9
      WRITE(IOUT,7)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKRA),G(ILRA),G(INZA),G(INSA)
     1,NLCZ)
    9 IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE8 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         (3) NONLINEAR CAUCHY-TYPE BOUNDARIES AND (OR) NONLINEAR POINT
C             SINKS
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTE
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTE
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
    7 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
     1,INSA,NBNC,NPNB,NLCZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
     2,NPNB,NLCZ)
      CALL SETCG(G(IAA))
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
```

```
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NGNCH.EQ.ISTP) CALL GNCHG(G(IHRK),G(IHRL),TIME,G(IKRA),G(ILRA)
     1,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL GNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA)
     2,G(IKPA),NBNC,NPNB)
      IF(NLCZ.LT.1) GO TO 9
      WRITE(IOUT,7)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKRA),G(ILRA),G(INZA),G(INSA)
     1,NLCZ)
    9 IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NLMFE5 COMBINED WITH NLMFE7 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED FLOW
C         (2) AQUIFER STORAGE CONVERSIONS
C         (3) NONLINEAR CAUCHY-TYPE BOUNDARIES AND (OR) NONLINEAR POINT
C             SINKS
C         (4) NONLINEAR STEADY LEAKAGE
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/IPRN/IPND
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTE
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
    7 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
     1,INSA,NBNC,NPNB,NLCZ)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
     2,NPNB,NLCZ)
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
```

```
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHLA)
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NGNCH.EQ.ISTP) CALL GNCHG(G(IHRK),G(IHRL),TIME,G(IKRA),G(ILRA)
     1,ISTP)
      IF(NVNCH.EQ.ISTP) CALL VNCHG(G(IHSA),TIME,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL GNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA)
     2,G(IKPA),NBNC,NPNB)
      IF(NLCZ.LT.1) GO TO 9
      WRITE(IOUT,7)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKRA),G(ILRA),G(INZA),G(INSA)
     1,NLCZ)
    9 CALL VNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(ITPA),G(IECA),G(IHSA)
     1,DT,G(INA))
```

```
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C        *** NLMFE6 COMBINED WITH NLMFE8 ***
C        MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C        IN TWO DIMENSIONS.
C        SIMULATES THE FOLLOWING CONDITIONS:
C        (1) UNCONFINED FLOW
C        (2) AQUIFER STORAGE CONVERSIONS
C        (3) NONLINEAR CAUCHY-TYPE BOUNDARIES AND (OR) NONLINEAR POINT
C            SINKS
C        (4) NONLINEAR STEADY LEAKAGE
C        USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C        TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
    6 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
    7 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL WTINIT(G,IDHA,ITKA,ITPA,ISYA)
      CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
     1,INSA,NBNC,NPNB,NLCZ)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL WTFMCO(G(IARA),G(ITKA),G(ITPA),G(ISYA),G(INDA))
      CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
     2,NPNB,NLCZ)
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      CALL SETCG(G(IAA))
      DO 20 JP=1,NPER
      JPER=JP
      CALL NXTPD(G(IDTA),JPER)
      DO 10 I=1,NSTEPS
```

```
      ISTP=I
      DT=G(IDTA+I-1)
      CALL COCHG(G(IQA),G(IARA),G(IHA),G(IHRA),G(IHBA),G(IHKA),G(IHL
     1,G(IALA),G(IQBA),G(ICKA),G(ICLA),TIME,G(IKA),G(ILA),G(INDA),G(INA)
     2,ISTP)
      IF(NGNCH.EQ.ISTP) CALL GNCHG(G(IHRK),G(IHRL),TIME,G(IKRA),G(ILRA)
     1,ISTP)
      IF(NVNCH.EQ.ISTP) CALL VNCHG(G(IHSA),TIME,ISTP)
      CALL RDTP(G(IAA))
      CALL FMEPWT(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA),G(IAA)
     1,G(IARA),G(IQA),G(IBA),G(ITKA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA)
     2,G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL WTCCHK(G(IHA),G(IDHA),G(IBA),G(ITPA),G(INA),ISC)
      IF(ISC.EQ.0) GO TO 5
      CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL FMECWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(ICKA)
     1,G(ICLA),G(IAA),G(IARA),G(IQA),G(IBA),G(ITKA),G(IYGA),G(ITPA)
     2,G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNCORR(G(IHA),G(IDHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA)
     1,G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
    5 CALL HCALWT(G(IHA),G(IDHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL MBALWT(G(IHA),G(IDHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA)
     1,G(IQBA),G(ICKA),G(ICLA),G(IAA),G(IQA),G(IYGA),G(IXGA),G(ITKA)
     2,G(IARA),G(ITPA),G(ISYA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 8
      WRITE(IOUT,6)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
    8 CALL GNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(IGCA),G(IHRK),G(IHRL)
     1,G(IZRK),G(IZRL),G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA)
     2,G(IKPA),NBNC,NPNB)
      IF(NLCZ.LT.1) GO TO 9
      WRITE(IOUT,7)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKRA),G(ILRA),G(INZA),G(INSA)
     1
```

```
     ,NLCZ)
    9 CALL VNBAL(G(IHA),G(IDHA),G(IHBA),G(IYGA),G(ITPA),G(IECA),G(IHSA)
     1,DT,G(INA))
      IF(ISTD.LT.1) CALL XTRPWT(G(IHA),G(IDHA),G(IHBA),G(ITKA),G(ITPA)
     1,G(INA),ISTP)
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL MASOUT(G(IYGA),DT,ISTP)
   10 CONTINUE
   20 CONTINUE
      STOP
      END
```

```
C         *** NSSFE1 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED, STEADY-STATE FLOW
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/IPRN/IPND
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ..... = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETCG(G(IAA))
      CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
      ISTP=1
      DT=1.
      DO 50 IT=1,NITSW
      ITER=IT
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
```

```
50 CONTINUE
60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
  1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
  2,G(INA),G(IKA),G(ILA))
   WRITE(IOUT,10) ITER,DSPA
   CALL DATOUT(G(IHA),DT,TIME,ISTP)
   CALL TKOUT(G(ITKA))
   IF(NQBND.LT.1) GO TO 62
   WRITE(IOUT,20)
   CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
62 CALL MASOUT(G(IYGA),DT,ISTP)
   STOP
   END
```

```
C         *** NSSFE2 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED, STEADY-STATE FLOW
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ..... = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
      ISTP=1
      DT=1.
      DO 50 IT=1,NITSW
      ITER=IT
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
   50 CONTINUE
```

```
60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
  1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
  2,G(INA),G(IKA),G(ILA))
   WRITE(IOUT,10) ITER,DSPA
   CALL DATOUT(G(IHA),DT,TIME,ISTP)
   CALL TKOUT(G(ITKA))
   IF(NQBND.LT.1) GO TO 62
   WRITE(IOUT,20)
   CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
62 CALL MASOUT(G(IYGA),DT,ISTP)
   STOP
   END
```

```
C         *** NSSFE3 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED, STEADY-STATE FLOW
C         (2) NONLINEAR STEADY VERTICAL LEAKAGE
C         USES MODIFIED, INCOMPLETE-CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/IPRN/IPND
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ..... = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETCG(G(IAA))
      CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      ISTP=1
      DT=1.
      DO 50 IT=1,NITSW
      ITER=IT
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
```

```
      CALL RDTP(G(IAA))
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
   50 CONTINUE
   60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      CALL VNBLSS(G(IHA),G(IYGA),G(ITPA),G(IECA),G(IHSA),DT,G(INA))
      WRITE(IOUT,10) ITER,DSPA
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL TKOUT(G(ITKA))
      IF(NQBND.LT.1) GO TO 62
      WRITE(IOUT,20)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
   62 CALL MASOUT(G(IYGA),DT,ISTP)
      STOP
      END
```

```
C         *** NSSFE4 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED, STEADY-STATE FLOW
C         (2) NONLINEAR STEADY LEAKAGE
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..................... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ................. = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      ISTP=1
      DT=1.
      DO 50 IT=1,NITSW
      ITER=IT
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
```

```
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
   50 CONTINUE
   60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      CALL VNBLSS(G(IHA),G(IYGA),G(ITPA),G(IECA),G(IHSA),DT,G(INA))
      WRITE(IOUT,10) ITER,DSPA
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL TKOUT(G(ITKA))
      IF(NQBND.LT.1) GO TO 62
      WRITE(IOUT,20)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
   62 CALL MASOUT(G(IYGA),DT,ISTP)
      STOP
      END
```

```
C         *** NSSFE5 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED, STEADY-STATE FLOW
C         (2) NONLINEAR CAUCHY-TYPE BOUNDARIES AND (OR) NONLINEAR POINT
C             SINKS
C         USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C         TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED'
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..................... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ................. = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
   22 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
      CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
     1,INSA,NBNC,NPNB,NLCZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETCG(G(IAA))
      CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
      CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
     2,NPNB,NLCZ)
      ISTP=1
      DT=1.
      DO 50 IT=1,NITSW
      ITER=IT
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
```

```
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
   50 CONTINUE
   60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 62
      WRITE(IOUT,20)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
   62 CALL GNBLSS(G(IHA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK),G(IZRL)
     1,G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC
     2,NPNB)
      IF(NLCZ.LT.1) GO TO 64
      WRITE(IOUT,22)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(INZA),G(INSA),NLCZ)
   64 WRITE(IOUT,10) ITER,DSPA
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL TKOUT(G(ITKA))
      CALL MASOUT(G(IYGA),DT,ISTP)
      STOP
      END
```

```
C         *** NSSFE6 ***
C         MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C         IN TWO DIMENSIONS.
C         SIMULATES THE FOLLOWING CONDITIONS:
C         (1) UNCONFINED, STEADY-STATE FLOW
C         (2) NONLINEAR CAUCHY-TYPE BOUNDARIES AND (OR) NONLINEAR POINT
C             SINKS
C         USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C         TO SOLVE MATRIX EQUATIONS.
       DIMENSION TITLE(20)
       COMMON/PRIME/G(5000)
       COMMON/GDIM/ISUM
       COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
      1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
       COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
       COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
       COMMON/ITP/IIN,IOUT,ITA,ITB
       COMMON/IPRN/IPND
       COMMON/IND/IRAD,IUNIT,ISTD
       COMMON/SCLE/SCALE
       COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
       COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
      1,THBQO,TER
       COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
       COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
       OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
      1,FORM='FORMATTED')
       OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
      1,FORM='FORMATTED')
       OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED
       OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..................... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ................. = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
   22 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
       CALL INITB(TITLE,G,TIME,NBCZ)
       CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
       CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
      1,INSA,NBNC,NPNB,NLCZ)
       CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
      1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
      2,G(IDZA),G(IDSA),NBCZ)
       CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
      1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
       CALL SETB(G(IAA),G(INA),G(INDA),IBND)
       CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
       CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
      1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
      2,NPNB,NLCZ)
       ISTP=1
       DT=1.
       DO 50 IT=1,NITSW
       ITER=IT
       CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
```

```
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
   50 CONTINUE
   60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 62
      WRITE(IOUT,20)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
   62 CALL GNBLSS(G(IHA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK),G(IZRL)
     1,G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC
     2,NPNB)
      IF(NLCZ.LT.1) GO TO 64
      WRITE(IOUT,22)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(INZA),G(INSA),NLCZ)
   64 WRITE(IOUT,10) ITER,DSPA
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL TKOUT(G(ITKA))
      CALL MASOUT(G(IYGA),DT,ISTP)
      STOP
      END
```

```
C        *** NSSFE3 COMBINED WITH NSSFE5 ***
C        MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C        IN TWO DIMENSIONS.
C        SIMULATES THE FOLLOWING CONDITIONS:
C        (1) UNCONFINED, STEADY-STATE FLOW
C        (2) NONLINEAR STEADY LEAKAGE
C        (3) NONLINEAR CAUCHY-TYPE BOUNDARIES
C        USES MODIFIED, INCOMPLETE CHOLESKY, CONJUGATE GRADIENT METHOD
C        TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..................... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ................. = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
   22 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
      CALL INITCG(TITLE,G,TIME,TOL,IAFA,IXA,IPA,IRA,NBCZ)
      CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
      CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
     1,INSA,NBNC,NPNB,NLCZ)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETCG(G(IAA))
      CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
      CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
     2,NPNB,NLCZ)
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      ISTP=1
      DT=1.
      DO 50 IT=1,NITSW
```

```
      ITER=IT
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL MICCG(G(IAA),G(IAFA),G(IXA),G(IPA),G(IRA),G(IBA),TOL,G(IJPA)
     1,G(INA))
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
   50 CONTINUE
   60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 62
      WRITE(IOUT,20)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
   62 CALL GNBLSS(G(IHA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK),G(IZRL)
     1,G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC
     2,NPNB)
      IF(NLCZ.LT.1) GO TO 64
      WRITE(IOUT,22)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKRA),G(ILRA),G(INZA),G(INSA)
     1,NLCZ)
   64 CALL VNBLSS(G(IHA),G(IYGA),G(ITPA),G(IECA),G(IHSA),DT,G(INA))
      WRITE(IOUT,10) ITER,DSPA
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL TKOUT(G(ITKA))
      CALL MASOUT(G(IYGA),DT,ISTP)
      STOP
      END
```

```
C        *** NSSFE4 COMBINED WITH NSSFE6 ***
C        MODULAR FINITE-ELEMENT MODEL OF GROUND-WATER FLOW
C        IN TWO DIMENSIONS.
C        SIMULATES THE FOLLOWING CONDITIONS:
C        (1) UNCONFINED, STEADY-STATE FLOW
C        (2) NONLINEAR STEADY LEAKAGE
C        (3) NONLINEAR CAUCHY-TYPE BOUNDARIES
C        USES DIRECT METHOD OF TRIANGULAR DECOMPOSITION
C        TO SOLVE MATRIX EQUATIONS.
      DIMENSION TITLE(20)
      COMMON/PRIME/G(5000)
      COMMON/GDIM/ISUM
      COMMON/ADR/IAA,IARA,IXGA,IYGA,IATA,IQA,IBA,IHA,IHRA,IHBA,IALA,IQBA
     1,ICKA,ICLA,IHKA,IHLA,IDTA,IJPA,INA,INDA,IKA,ILA,IDZA,IDSA
      COMMON/NO/NELS,NNDS,NSTEPS,NPER,NZNS,NWELS,NQBND,NHDS,NEQ,MBWC,NIT
      COMMON/CHG/NWCH,NQCH,NHRCH,NBQCH,NHCH,NCBCH,NVNCH,NGNCH
      COMMON/ITP/IIN,IOUT,ITA,ITB
      COMMON/IPRN/IPND
      COMMON/IND/IRAD,IUNIT,ISTD
      COMMON/SCLE/SCALE
      COMMON/BAL/SA,WQI,WQO,DQI,DQO,VLQI,VLQO,BQI,BQO,ER
      COMMON/TBAL/TSA,TWQI,TWQO,TDQI,TDQO,TLQI,TLQO,TBQI,TBQO,THBQI
     1,THBQO,TER
      COMMON/VNLBL/VNLQI,VNLQO,TNLQI,TNLQO
      COMMON/GNBL/BNQI,BNQO,TBNQI,TBNQO,PNQO,TPNQO
      OPEN (50,FILE='MODFE.DAT',STATUS='OLD',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (60,FILE='MODFE.OUT',STATUS='NEW',ACCESS='SEQUENTIAL'
     1,FORM='FORMATTED')
      OPEN (55,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
      OPEN (56,STATUS='SCRATCH',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
   10 FORMAT (1H1,18X,30HSUMMARY OF CLOSURE INFORMATION/
     $56H NO. OF ITERATIONS TO CLOSE (ITER) ..................... = ,I4/
     $56H MAXIMUM ABSOLUTE DISPLACEMENT (DSPA) ................. = ,G11.5)
   20 FORMAT(/9X,49HSUMMARY OF FLOW AT CAUCHY-TYPE BOUNDARIES BY ZONE)
   22 FORMAT(/4X,59HSUMMARY OF FLOW AT NONLINEAR CAUCHY-TYPE BOUNDARIES
     1BY ZONE)
      CALL INITB(TITLE,G,TIME,NBCZ)
      CALL SWINIT(G,TOLSW,DSMX,DSP,RP,NITSW,ITKA,ITPA)
      CALL GNINIT(G,IGCA,IHRK,IHRL,IZRK,IZRL,IZPA,IKRA,ILRA,IKPA,INZA
     1,INSA,NBNC,NPNB,NLCZ)
      CALL VNINIT(G,IECA,IHSA,NVNZ)
      CALL DATIN(TITLE,G(IXGA),G(IYGA),G(IHA),G(IHBA),G(IHRA)
     1,G(IHKA),G(IHLA),G(IALA),G(IQBA),G(IQA),G(IKA),G(ILA),G(INA)
     2,G(IDZA),G(IDSA),NBCZ)
      CALL FMCO(G(IXGA),G(IYGA),G(IAA),G(IQA),G(IARA),G(IALA),G(IQBA)
     1,G(ICKA),G(ICLA),G(INDA),G(IBA),G(IJPA),G(IKA),G(ILA))
      CALL SETB(G(IAA),G(INA),G(INDA),IBND)
      CALL SWFMCO(G(IAA),G(ITKA),G(ITPA),G(IJPA),G(INA))
      CALL GNFMCO(G(IXGA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(IKRA),G(ILRA),G(IKPA),G(INZA),G(INSA),NBNC
     2,NPNB,NLCZ)
      CALL VNFMCO(G(IHA),G(IARA),G(IECA),G(IHSA),G(INDA),NVNZ)
      ISTP=1
      DT=1.
      DO 50 IT=1,NITSW
```

```
      ITER=IT
      CALL FMEQ(G(IHA),G(IHRA),G(IHKA),G(IHLA),G(ICKA),G(ICLA)
     1,G(IAA),G(IARA),G(IQA),G(IBA),DT,G(IJPA),G(INA),G(IKA),G(ILA))
      CALL GNPRED(G(IHA),G(IAA),G(IBA),G(IGCA),G(IHRK),G(IHRL),G(IZRK)
     1,G(IZRL),G(IZPA),G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC,NPNB)
      CALL VNPRED(G(IHA),G(IAA),G(IBA),G(ITPA),G(IECA),G(IHSA),G(INA))
      CALL BAND(G(IAA),G(IATA),G(IBA),G(IJPA),G(INA),IBND)
      CALL SWBDMP(G(IBA),DSMX,DSP,DSPO,DSPA,RP,ITER)
      CALL HCALC(G(IHA),G(IBA),G(INA))
      CALL RDTP(G(IAA))
      CALL SWTHK(G(IHA),G(IAA),G(IBA),G(IQA),G(ITKA),G(ITPA),DSPA,TOLSW
     1,G(IJPA),G(INA),ITER)
      IF(DSPA.LT.TOLSW) GO TO 60
   50 CONTINUE
   60 CALL MASBAL(G(IHA),G(IHBA),G(IHRA),G(IHKA),G(IHLA),G(IQBA),G(ICKA)
     1,G(ICLA),G(IAA),G(IQA),G(IBA),G(IYGA),G(IARA),G(IXGA),DT,G(IJPA)
     2,G(INA),G(IKA),G(ILA))
      IF(NQBND.LT.1) GO TO 62
      WRITE(IOUT,20)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKA),G(ILA),G(IDZA),G(IDSA),NBCZ)
   62 CALL GNBLSS(G(IHA),G(IYGA),G(IGCA),G(IHRK),G(IHRL),G(IZRK),G(IZRL)
     1,G(IZPA),G(IARA),G(IXGA),DT,G(INA),G(IKRA),G(ILRA),G(IKPA),NBNC
     2,NPNB)
      IF(NLCZ.LT.1) GO TO 64
      WRITE(IOUT,22)
      CALL PRTCBV(G(IARA),G(IXGA),DT,G(IKRA),G(ILRA),G(INZA),G(INSA)
     1,NLCZ)
   64 CALL VNBLSS(G(IHA),G(IYGA),G(ITPA),G(IECA),G(IHSA),DT,G(INA))
      WRITE(IOUT,10) ITER,DSPA
      CALL DATOUT(G(IHA),DT,TIME,ISTP)
      CALL TKOUT(G(ITKA))
      CALL MASOUT(G(IYGA),DT,ISTP)
      STOP
      END
```