CHAPTER 3
Design of the Ground-Water Flow Process

Procedures

To facilitate writing an easily understood program, the Ground-Water Flow (GWF) Process is broken into pieces called “procedures.” Figure 3–1 is a flowchart that shows the order in which procedures are invoked to create a program that solves the ground-water flow equation. The procedures are shown as rectangles in figure 3–1. Other processes can be broken into procedures and combined in the flowchart in a similar manner, but the scope of this report is limited to GWF.

The period of simulation is divided into a series of "stress periods" within which specified stress data are constant. Each stress period, in turn, is divided into a series of time steps. The system of finite-difference equations of the form of equation 2–26 is formulated and solved to yield the head at each node at the end of each time step. Iterative solution methods are used to solve for the heads for each time step. Thus, the program includes three nested loops: a stress-period loop, within which there is a time-step loop, which in turn contains an iteration loop.

Prior to entering the stress loop, the Allocate and Read (AR) Procedure is executed. The AR Procedure pertains to the simulation as a whole and performs a number of setup functions. The number of cells in the grid for the problem to be simulated is determined as well as the hydrologic options and the solution method. Memory is allocated for all aspects of the simulation. Data that do not vary within stress periods are read. These data include the following: the cell dimensions and time information, boundary conditions, initial heads (starting heads), aquifer hydraulic properties, and control information required by the specified solution scheme. Certain preliminary calculations also are made in this procedure to prepare data for further processing.

Within the stress period loop, the first procedure is termed the Stress (ST) Procedure. This procedure advances to a new stress period. The Read and Prepare (RP) Procedure reads and processes all data that pertain to a stress period, such as pumping rates and areal recharge. The time-step loop is then entered; in the Advance (AD) Procedure, the length of the time step is calculated and the heads for the start of the time step are initialized. The AD Procedure also performs other processing that must be done every time step.

The iteration loop contains the Formulate (FM) Procedure, which determines the conductances and coefficients for each node as required by equation 2–26, and the Approximate Procedure (AP), which approximates a solution to the system of linear equations for head. The FM Procedure is called prior to each solver iteration so that the conductances and coefficients in the flow equation can be changed based on the latest approximate head solution. Iteration proceeds until closure is achieved or until a specified maximum number of allowable iterations are reached.

At the end of the iteration loop, the Output Control (OC) Procedure determines which computed information, such as heads, budget terms, and cell-by-cell flow terms, will be output. In the Water Budget (BD) Procedure, budget entries are calculated and cell-by-cell flow terms are printed or recorded, as explained in a subsequent section. In the Output (OT) Procedure, the specified computed information is printed or recorded.

After all time steps are completed for all stress periods, the Deallocate (DA) Procedure releases allocated memory.

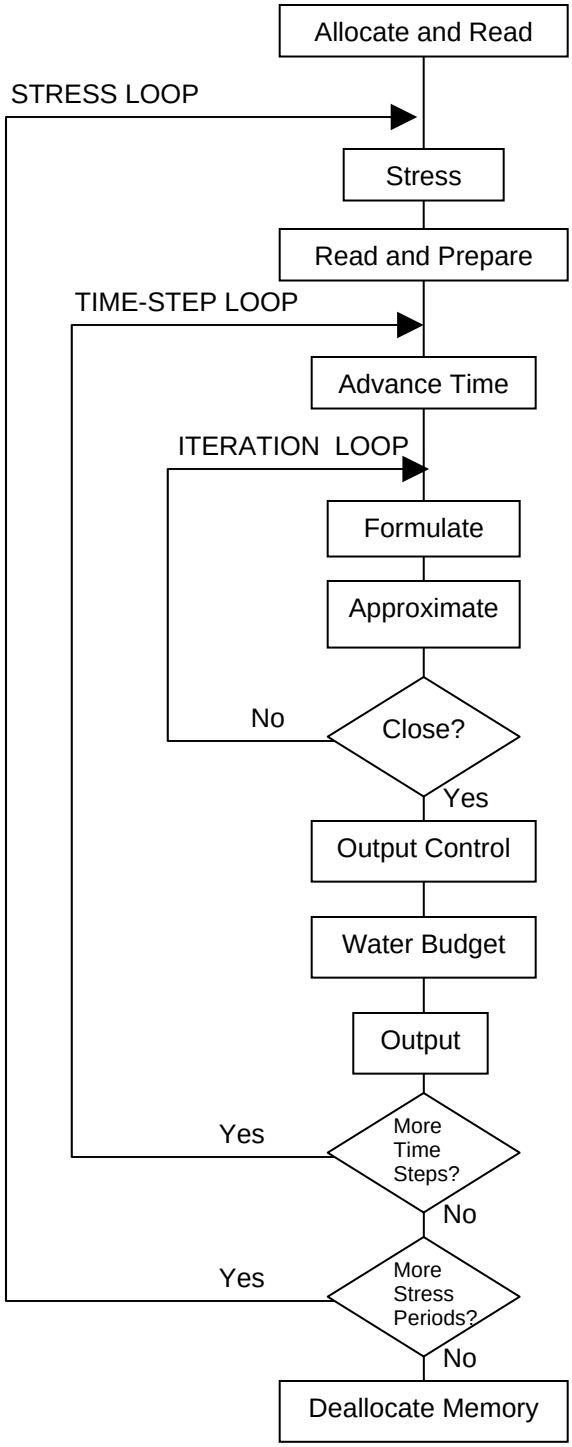

Figure 3–1. Flowchart of program to simulate ground-water flow.

Packages

Although the procedures are fundamental components of the computer program, a hydrologist using the program will likely want to think of the program in terms of its capabilities for solving hydrologic problems. For this purpose, the program is divided into packages. The hydrologist will naturally think about the way the ground-water flow equation is constructed. Accordingly, the various parts of the code that deal with defining the ground-water flow equation are divided into packages that are called hydrologic packages. There are two types of hydrologic packages. The first type is the internal flow package, which simulates flow between adjacent cells. The second type is the stress package, which simulates an individual kind of stress (such as rivers, wells, and recharge). Although hydrologists would generally prefer not to have to deal with how the simultaneous equations that result from the finite-difference approximation are solved, the reality is that equation solution is not straightforward and must be dealt with when using a ground-water model. MODFLOW incorporates multiple solution methods, and each one is termed a solver package. Logically, the hydrologic and solver packages represent the entire work of the program; however, an additional component is needed to control the program. This is called the Basic Package.

Table 1 lists the various packages of GWF that are documented in this publication, the three-character abbreviation used for each package, and the package category. Table 1 does not show anything about the programming, but shows the capabilities of the program. The hydrologic packages calculate the coefficients of the finite-difference equation for each cell. The Block-Centered Flow (BCF) and Layer-Property Flow (LPF) Packages are both internal flow packages that provide alternate approaches to formulate the internal flow terms. The Horizontal Flow Barrier (HFB) Package is a supplementary internal flow package that works with BCF and LPF to modify conductances to simulate a barrier between nodes. Each of the stress packages formulates the coefficients describing a particular external or boundary flow; for example, the River Package calculates the coefficients describing flow between a cell and a surface river. The solver packages are those that implement algorithms for solution of the systems of finite-difference equations. This documentation describes three solver packages, one incorporating the Strongly Implicit Procedure of solution, another utilizing Preconditioned Conjugate Gradient method, and one using a direct solution method. The only package that does not fit into the hydrologic or solver categories is the Basic Package, which addresses a variety of tasks in support of the entire process.

Table 3–1. List of Packages for simulating ground-water flow.

Package Name	Abbreviation	Package Category
Basic	BAS	Program Control
Block-Centered Flow	BCF	Hydrologic/Internal
Layer-Property Flow	LPF	Hydrologic/Internal
Horizontal Flow Barrier	HFB	Hydrologic/Internal
Well	WEL	Hydrologic/Stress
Recharge	RCH	Hydrologic/Stress
River	RIV	Hydrologic/Stress
General-Head Boundary	GHB	Hydrologic/Stress
Drain	DRN	Hydrologic/Stress
Evapotranspiration	EVT	Hydrologic/Stress
Strongly Implicit Procedure	SIP	Solver
Preconditioned Conjugate Gradient	PCG	Solver
Direct Solution	DE4	Solver

Primary Subroutines

As described earlier in this chapter, figure 3–1 provides a flow chart for the overall structure of MODFLOW for solving the ground-water flow equation. The flow chart indicates the sequence in which procedures must be invoked to build and solve the ground-water flow equation. To construct a program following this flow chart, each procedure could be implemented as a single subroutine. From a programming perspective, this would be the most direct way to construct the program; however, the subroutines would be quite large, and there is benefit from further subdividing the work into smaller subroutines.

There are many ways the code could be subdivided. The approach used for MODFLOW is to divide the code into pieces that can be combined into both procedures and packages as desired. Accordingly, a primary subroutine is defined as the code within a procedure for one package. A procedure can be constructed by grouping all the primary subroutines that comprise the procedure. A package can be constructed by grouping all the primary subroutines that comprise the package.

Using this approach, the MAIN Program is simply an organized sequence of call statements to the primary subroutines. The MAIN Program is organized according to procedures in the flow chart (fig. 3–1). Each procedure is invoked through calls to multiple primary subroutines — one for each package represented by the procedure. Most of the packages are optional depending on the problem being simulated. Accordingly, the calls to the subroutines of optional packages are invoked through "IF" tests that determine whether a primary subroutine is required. Thus, the MAIN Program does not itself do the work of simulation, but merely calls the various primary subroutines in the proper sequence to do that work.

The classification of GWF Process primary subroutines by procedure and by package is illustrated in figure 3–2. The horizontal rows in figure 3–2 correspond to procedures, and the vertical columns correspond to packages. An "X" is entered in each block of the matrix for which a primary subroutine exists; absence of an "X" indicates that the procedure in question is not required in the indicated package.

	BAS	BCF	LPF	HFB	WEL	RCH	RIV	GHB	DRN	EVT	SIP	PCG	DE4
Allocate and Read (AR)	X	X	X	X	X	X	X	X	X	X	X	X	X
Stress (ST)	X												
Read and Prepare (RP)					X	X	X	X	X	X			
Advance (AD)	X												
Formulate (FM)	X	X	X	X	X	X	X	X	X	X			
Approximate (AP)											X	X	X
Output Control (OC)	X												
Water Budget (BD)		X	X		X	X	X	X	X	X			
Output (OT)	X												
Deallocate (DA)	X	X	X	X	X	X	X	X	X	X	X	X	X

Figure 3–2. GWF Process primary subroutines classified by procedure and package.

The primary subroutines are named according to a convention that indicates the process, package, and the procedure to which they belong. The first three characters designate the process. The fourth character is a process version number. The next three characters designate the package, the following character is a package version number, and the last two indicate the procedure. For example, in figure 3–2, a GWF primary subroutine is indicated

for the Well Package and Allocate and Read Procedure. This subroutine is designated as GWF2WEL7AR—the first three letters, GWF, indicate the Ground-Water Flow Process; the fifth through seventh letters, WEL, indicate that the subroutine is part of the Well Package; the last two letters, AR, indicate that it performs the Allocate and Read Procedure in that package. Thus, this subroutine is one that deals with the simulation of specified withdrawal or injection, as through wells, and its particular functions are to allocate the space in computer memory used to store well data and to read well data that are constant during the simulation. The number 2 in the fourth place of the name is a process version number. In this report, the GWF Process is designated as version 2. The number 7 appearing in the eighth place of the subroutine name is a package version number. If the package is modified to effect improvements, a different integer would be used in this place to distinguish the modified package from other versions. McDonald and Harbaugh (1988) designated all packages as version 1. Many model users have made modifications to various packages, so there are model packages with versions greater than 1. In this report, all packages are designated as version 7 in order to minimize the chance of duplicating a version number used by others.

When primary subroutines become so large that they are difficult to understand, they are broken into smaller secondary subroutines. All of these subroutines start with the letter “S.” Utility subroutines are further used to keep the program as easy to understand as possible. Utility subroutines implement functionality that is required by many packages. When a secondary or utility subroutine is called, it is logically viewed as being part of the calling subroutine. Specific secondary and utility subroutines are documented later in this report.

In summary, the program is broken into primary subroutines. These primary subroutines can be grouped according to the procedures indicated in figure 3–1 in order to construct a computer program. Primary subroutines can also be grouped by "packages," where a package (for example, the River Package, the Well Package, or the Strongly Implicit Procedure Package) includes those subroutines required to incorporate a particular hydrologic process or solution algorithm into the simulation. In terms of understanding the operation of the model, both of these two methods of grouping subroutines are useful. The package classification, for example, indicates which subroutines will be active in a given simulation. Subroutines are called by the MAIN Program only if they are part of a package that is required in the simulation. On the other hand, the procedure classification defines the specific function of a subroutine in relation to the structure of the computer program. For example, several subroutines whose function is to allocate space and read data are grouped under the Allocate and Read Procedure; each of these subroutines allocates the space required for use in a single package. If few options or features are specified, relatively few packages are involved in the simulation, and the Allocate and Read Procedure is handled by a relatively small number of subroutines. As the options specified by the user increase, more packages enter the simulation, and more subroutines are called to complete the space allocation task.

Computing Flow Equation Terms

For cells that are designated as always being confined, the conductance terms for cell-to-cell flow (CC, CR, and CV of equation 2–26) are computed at the beginning of the simulation in the AR subroutine of the internal flow package. These terms will remain constant throughout the simulation for confined cells. For cells that are (or may become) unconfined, the conductance terms are computed in the FM Procedure at each solution iteration because the conductances depend upon saturated thickness, which may change at each iteration. The various conductance terms are ultimately passed to a solver package, where the matrix equations (eq. 2–27) are solved.

The coefficient $HCOF_{i,j,k}$ and the term $RHS_{i,j,k}$ of equation 2–26 are formulated anew at each iteration for all active nodes in the grid. This formulation is done progressively as the FM subroutine for each package calculates and adds terms for the particular hydrologic process associated with that package. At the beginning of each iteration, the values of $HCOF_{i,j,k}$ and $RHS_{i,j,k}$ are set to zero throughout the grid. The internal flow package then adds the term

$$\frac{-SS_{i,j,k}\,\Delta r_j\,\Delta c_i\,\Delta v_k}{t - t^{m-1}}$$

to $HCOF_{i,j,k}$ at each node, and adds the term

$$-SS_{i,j,k}\,\Delta r_j\,\Delta c_i\,\Delta v_k\,\frac{h_{i,j,k}^{m-1}}{t\;-t^{m-1}}$$

to $RHS_{i,j,k}$ at each node. Packages that specify external inflow (negative value indicates outflow) using expressions of the form $P_{i,j,k}h_{i,j,k}+Q_{i,j,k}$ add the term $P_{i,j,k}$ to $HCOF_{i,j,k}$ and the term $-Q_{i,j,k}h_s$ to $RHS_{i,j,k}$ for the affected cells. This process continues until each package specified by the user has added its contribution to HCOF and RHS at each indicated cell of the grid. The HCOF and RHS values are then transferred to the solver package, together with the conductances (CC, CR, and CV) and the heads at the beginning of the time step. The solver package sums the six conductance terms and the value of HCOF for each cell to create a single coefficient of $h_{i,j,k}$ (corresponding to the term in parentheses in equation 2–26), and carries out one iteration of the solution procedure.

Adding and Modifying Packages

The structure of GWF has been designed in such a way that the packages are as independent as possible. This facilitates making modifications of all types. New packages can be added and the subroutines of an existing package can be modified without affecting other packages. The method of incorporating stresses through the HCOF and RHS terms allows new stress packages to be used along with the original stress packages. Development of a new package involves creating the primary subroutines for each of the procedures involved and modifying the MAIN Program to call those subroutines in proper sequence. Two general constraints limit what can be done with new packages. First, only one internal flow package that formulates the internal flow terms (the BCF and LPF Packages in this report), and second, only one solver package can be in use in a single simulation. For example, a new internal flow package utilizing a point centered approach could be added, but using the new package at the same time as the BCF or LPF Package would not be possible (or make sense). Likewise, new solvers can be added as long as only one is used at a time.

Steady-State Simulations

The program structure for simulating ground-water flow illustrated in figure 3–1 was described assuming that a transient problem is being solved; however, the same structure also works for steady-state simulations. A steady-state simulation is represented by a single stress period having a single time step with the storage term set to zero. Thus a single set of simultaneous equations will be solved. All procedures within the program structure are required much as they are for a transient problem. Setting the number and length of stress periods and time steps is the responsibility of the Basic Package. The length of the stress period and time step will not affect the head solution because the time derivative is not calculated in a steady-state problem. Setting the storage term to zero is the responsibility of the internal flow package. Most other packages need not "know" that a simulation is steady state.

Simulations also can be mixed transient and steady state because each stress period can be designated transient or steady state. Thus, a simulation can start with a steady-state stress period and continue with one or more transient stress periods.

Units of Length and Time

The GWF Process formulates the ground-water flow equation without using prescribed length and time units. Any consistent units of length and time can be used when specifying the input data for a simulation. This capability gives a certain amount of freedom to the user, but care must be exercised to avoid mixing units. The program cannot detect the use of inconsistent units. For example, if hydraulic conductivity is entered in units of feet per day and pumpage as cubic meters per second, the program will run, but the results will be meaningless. Other processes

generally are expected to work with consistent length and time units; however, other processes could conceivably place restrictions on which units are supported.

The user can set flags that specify the length and time units (see the input instructions for the Discretization File), which may be useful in various parts of MODFLOW. For example, the Basic Package will label the table of simulation time with time units if the time units are specified by the time-units flag (ITMUNI). If the time units are not specified, the program still runs, but the table of simulation time does not indicate the time units. A length-unit flag (LENUNI) is also included in MODFLOW. Situations in other processes may require that the length or time units be specified. In such situations, the input instructions will state the requirements. Remember that specifying the unit flags does not enforce consistent use of units. The user must insure that consistent units are used in all input data.

Model Grid and Aquifer Boundaries

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

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

A finite-difference equation of the form of equation 2–26 is formulated for each variable-head cell in the grid. For constant-head cells, no equation is formulated; however, the equation for each variable-head cell adjacent to a constant-head cell contains a term describing flow to and from the constant-head cell. For no-flow cells, no equation is formulated, and no term appears in the equation of any adjacent cell for flow to or from the no-flow cell; thus no flow is simulated across the interval between a no-flow cell and any adjacent cell.

Constant-head cells, no-flow cells, and variable-head cells are distinguished from one another in the model through the IBOUND variable, which contains one value for each cell in the grid. The entry in IBOUND for a given cell indicates the type of cell according to the following convention:

If $IBOUND_{i,j,k} < 0$, then cell i,j,k is constant head.

If $IBOUND_{i,j,k} = 0$, then cell i,j,k is no flow.

If $IBOUND_{i,j,k} > 0$, then cell i,j,k is variable head.

The IBOUND codes are initially specified by the user. If necessary, the codes are adjusted so that they are consistent with other data specified by the user and with intermediate results. For example, cells that are specified as active but are given hydraulic conductivity values equal to zero are changed to no-flow cells by the program.

Model Input and Output

MODFLOW is designed to permit model input to be gathered, as it is needed, from many different files. Likewise, results from the model calculations can be written to a number of files. The Listing File is a key file to which model output is written. As MODFLOW runs, information is written to the Listing File, including much of the input data (as a record of the simulation) and calculated results. An overview of input and output is provided here. Details about the files used by each package are provided in the chapters documenting the packages. Also, the next chapter provides details about the ways data are read from files.

MODFLOW is further designed to allow the user to control the amount, type, and frequency of information to be output. Much of the output will be written to the Listing File, but some model output can be written to other files. The Listing File is designed to be printed; however, the Listing File may not need to be printed if tools (such as a word processor) are available to examine it using the computer display. The Listing File includes a summary of the input data read for all packages. In addition, the Listing File optionally contains calculated head and drawdown controlled by layer and time step, and the overall volumetric budget controlled by time step. The Listing File also contains information about solver convergence and error messages. Output to other files can include head, drawdown, and cell-by-cell flow terms for use in calculations external to the model or in user-supplied applications such as plotting programs.

The Basic Package reads a file called the Name File, which specifies most of the files that will be used in a simulation. Several files, including the Listing File, are always required whereas other files are optional depending on the simulation. Output Control, which is a major option contained within the Basic Package (and a procedure in the overall flow chart), receives instructions from the user to control the amount and frequency of output. Details about Output Control and the Name File are provided in Chapter 4 (Basic Package).

Within the program, input and output are based on an element of the Fortran language called the file unit number, which identifies the file from which the input is to be read or to which the output is to be written. A connection must be provided between the name of each input or output file and the corresponding unit number. For files included in the Name File, this link is made by directly specifying the file unit number along with the file name.

Volumetric Budget

A summary of all inflows (sources) and outflows (sinks) of water to a region is generally called a water budget. In this report, the water budget is termed a volumetric budget because volumes of water and volumetric flow rates are involved; thus strictly speaking, a volumetric budget is not a mass balance, although this term has been used in other model reports. The model program calculates a water budget for the overall model as a check on the acceptability of the solution, and to provide a summary of the sources and sinks of water to the flow system.

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

change in storage. In the model program, the water budget is calculated independently of the equation solution process, and in this sense may provide independent evidence of a valid solution.

The total budget as printed in the output does not include internal flows between model cells—only flows into or out of the model as a whole. For example, flow to or from rivers, flow to or from constant-head cells, and flow to or from wells are all included in the overall budget terms. Flow into and out of storage is also considered part of the overall budget inasmuch as accumulation in storage effectively removes water from the flow system and storage release effectively adds water to the flow—even though neither process, in itself, involves the transfer of water into or out of the ground-water regime. Each hydrologic package calculates its own contribution to the budget.

For every time step, the budget subroutine of each hydrologic package calculates the rate of flow into and out of the system due to the process simulated by the package. The inflows and outflows for each component of flow are stored separately in a program variable (VBVL). Most packages deal with only one such component of flow, but the internal flow packages (BCF and LPF) deal with two—flow to constant-head cells and flow to storage. In addition to flow, the volumes of water entering and leaving the model during the time step are calculated as the product of flow rate and time-step length. Cumulative volumes, from the beginning of the simulation, are then calculated and stored in VBVL.

The BAS Package uses the inflows, outflows, and cumulative volumes in VBVL to write the budget to the Listing File at the times requested by the model user. When a budget is written, the flow rates for the last time step and cumulative volumes from the beginning of simulation are written for each component of flow. Inflows are written separately from ouflows. Following the convention indicated above, water entering storage is treated as an outflow (that is, as a loss of water from the flow system) while water released from storage is treated as an inflow (that is, a source of water to the flow system). In addition, total inflow and total outflow are written, as well as the difference between total inflow and outflow. The difference is then written as a percentage error, calculated using the formula:

$$D = \frac{100(\mathrm{IN} - \mathrm{OUT})}{(\mathrm{IN} + \mathrm{OUT})/2} ,$$

where

D is the percentage error term,

IN is the total inflow to the system, and

OUT is the total outflow.

If the model equations are solved correctly, the percentage error should be small. In general, flow rates may be taken as an indication of solution validity for the time step to which they apply, while cumulative volumes are an indication of validity for the entire simulation up to the time of the output. The budget is written to the Listing File at the end of each stress period whether requested or not.

In some situations, calculating flow terms for various subregions of the model is useful. To facilitate such calculations, provision has been made to save flow terms for individual cells in a separate file so they can be used in computations external to the model itself. These individual cell flows are referred to here as "cell-by-cell" flow terms and are of four general types: (1) cell-by-cell stress flows, or flows into or from an individual cell caused by one of the external stresses represented in the model, such as evapotranspiration or recharge; (2) cell-by-cell storage terms, which give the rate of accumulation or depletion of storage in an individual cell; (3) cell-by-cell constant-head flow terms, which give the net flow to or from individual constant-head cells; and (4) internal cell-by-cell flows, which are actually the flows across individual cell faces—that is, between adjacent model cells. These four kinds of cell-by-cell flow terms are discussed further in subsequent paragraphs. To save any of these cell-by-cell terms, two flags in the model input must be set. The input to the Output Control section of the Basic Package indicates the time steps for which cell-by-cell terms are to be saved. In addition, each hydrologic package includes a flag that is set if the cell-by-cell terms computed by that package are to be saved. Thus, if the appropriate flag in the Evapotranspiration Package input is set, cell-by-cell evapotranspiration terms will be saved for each time step for which the saving of cell-by-cell flow is requested through the Output Control Option. Only flow values are saved in

the cell-by-cell files; neither water volumes nor cumulative water volumes are included. The flow dimensions are volume per unit time, where volume and time are in the same units used for all model input data. The cell-by-cell flow values are stored in unformatted form to make the most efficient use of disk space; see the narrative for the budget utility subroutines in Chapter 9 for information on how the data are written to a file.

Three of the four types of cell-by-cell flow terms listed above—storage, constant-head cell, and internal flows—are computed in the internal flow package, and thus fall under the control of a single flag in the input to that package. Thus, all three types are saved in a file if this flag is set, and if the saving of cell-by-cell flow also is requested in Output Control for the time step.

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

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

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

Cell-by-cell stress flows are flow rates into or out of the model, at a particular cell, owing to one particular external stress. For example, the cell-by-cell evapotranspiration term for cell i,j,k would give the flow out of the model by evapotranspiration from cell i,j,k. Cell-by-cell stress flows are considered positive if flow is into the cell, and negative if out of the cell.

Three-Dimensional Model Data

Many terms in the equations presented in this report specify cell location using row, column, and layer indices in that order (usually designated as i,j,k), as is customary in scientific literature. Such terms are stored in the program as three-dimensional variables (called arrays in Fortran). Fortran incorporates the capability to use subscripts to access values (called elements) in variables, and the subscripts are indicated in parentheses after a variable name. The order of subscripts in the Fortran language determines the order of storage in computer memory, which can affect program efficiency. The design of the program is such that the subscripts should be in column, row, and layer order for the best efficiency on most computers; therefore, this order has been used throughout the program. Bear in mind this difference in the subscript ordering when comparing the model program to the equations presented in this report. Typically in the program, J is used for the column subscript, I is used for row, and K is used for layer, so the order of indices is J,I,K.

In many cases, the variable name in the program is the same as the term name used in the equations. For example, there are CR, CC, CV, HCOF, and RHS variables that store the corresponding terms. There are some variables in the model program, however, that have different names than the terms used in equations. For example, h is not used as a variable name primarily because head is not stored for every time step. Rather, two values of head are stored: the head at the end of the new time step (h^m) and the head at the end of the prior time step (h^{m-1}). These

are named HNEW and HOLD, respectively, and these names serve to indicate that these values change as the time step changes. The program moves HNEW to HOLD at the start of each time step. Another head variable in the program is STRT, which is the initial head specified by the user at the start of the simulation (h^0). The documentation for each package specifies any differences between the names of terms in equations and the variable names used in the program.

Memory Allocation and Deallocation

Most variables that are shared among subroutines are declared in Fortran modules. These variables can be accessed in subroutines through Fortran USE statements. The memory required for variables that are referenced using subscripts (arrays) is obtained using Fortran ALLOCATE statements. DEALLOCATE statements are used at the end of the model to release the allocated memory.

The MAIN Program

The MAIN Program controls the order in which the primary subroutines are executed. This occurs with CALL statements that specify, by name, the subroutines to be executed. The arrangement of CALL statements in the MAIN Program reflects the order of procedures shown in the system flow chart (fig. 3–1). The subroutines of a package are called only if the package is being used in a simulation as specified in the Name File as described previously. Details about the MAIN Program are provided in Chapter 9.