Direct Solution Package Based on Alternating Diagonal Ordering for the U.S. Geological Survey Modular Finite-Difference Ground-Water Flow Model

by Arlen W. Harbaugh

U.S. GEOLOGICAL SURVEY
Open File Report 95-288

Reston, Virginia
1995

U.S. DEPARTMENT OF THE INTERIOR
BRUCE BABBITT, *Secretary*

U.S. GEOLOGICAL SURVEY
Gordon P. Eaton, *Director*

For additional information
write to:

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

Copies of this report can be
purchased from:

USGS ESIC -- Open-File Report Section
Box 25286, MS 517
Denver Federal Center
Denver, CO 80225
(303) 236-7476

PREFACE

This report presents an equation solver for the U.S. Geological Survey (USGS) modular finite-difference ground-water flow model, commonly known as MODFLOW. The program has been tested by using it for a variety of model simulations, but it is possible that other applications could reveal errors. Users are requested to notify the USGS if errors are found in this report or the program.

Although this program has been used by the USGS, no warranty, expressed or implied, is made by the USGS or the United States Government as to the accuracy and functioning of the program and related program material nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the USGS in connection therewith.

Copies of the source computer program on diskette are available for the cost of processing from:

U.S. Geological Survey
NWIS Program Office
437 National Center
Reston, VA 22092
Telephone: (703) 648-6978

The source program must be compiled using a Fortran compiler in order to make it run on a particular computer. Modifications to the program may be required in order to make the program work on computers other than the kind on which it was developed, which is the Data General Unix workstation.

(page v follows)

CONTENTS

ILLUSTRATIONS

Page

Figure

TABLES

Page

Table

Direct Solution Package Based on Alternating Diagonal Ordering for the U.S. Geological Survey Modular Finite-Difference Ground-Water Flow Model

By Arlen W. Harbaugh

ABSTRACT

This report documents a direct solver for MODFLOW, the U.S. Geological Survey modular three-dimensional finite-difference ground-water flow model. Included in the report are input instructions, a description of concepts and implementation, and detailed program documentation. Although direct solvers require more memory and typically require more computational effort than iterative solvers, a direct solver may execute faster than an iterative solver in some situations. A direct solver is the most efficient when solving small, linear problems. The direct solver uses Gaussian elimination with an alternating diagonal equation numbering scheme that is more efficient than the standard method of equation numbering.

INTRODUCTION

The U.S. Geological Survey (USGS) modular three-dimensional finite-difference ground-water flow model, commonly called MODFLOW, was designed to incorporate multiple solvers (McDonald and Harbaugh, 1988). A solver is used to solve the simultaneous equations that result from the application of the finite-difference approximation. There is a need for more than one solver because no single solver is well suited for solving the simultaneous equations for all problems. Frequently, multiple solvers can correctly solve the problem, but the execution speed and the amount of memory used are different. Rarely, it may be difficult to find a solver that can correctly solve a particular problem, so having many solvers to choose from increases the chance that a solution can be obtained.

For each time step, the finite-difference approximation results in a set of simultaneous equations of the form (McDonald and Harbaugh, 1988, p. 2-26):

$$[A]\{h\}=\{q\} \qquad (1)$$

where

[A] is a matrix of coefficients of head,

{h} is a vector of head values, and

{q} is a vector of constant terms.

Note that a steady-state simulation consists of one time step in which the storage terms are 0.

There are two broad types of solvers -- direct and iterative. A direct solver solves the set of simultaneous equations through an algorithm of algebraic manipulations that theoretically would produce an exact solution. The solution is not exact in most situations, however, because some error results from a computer's finite precision. Gaussian elimination is a widely known method of direct solution. Iterative solvers make multiple approximate solutions of the equations being solved. Each approximate solution is called an iteration, and the process is designed such that successive iterations improve the accuracy. An iterative solver that is working properly is said to converge upon the correct answer. The goal is to minimize the number of iterations required to obtain a solution having acceptable accuracy.

An additional complication is that MODFLOW in many situations formulates a non-linear flow equation, which results in non-linear simultaneous equations. Non-linear equations are generally more difficult to solve than linear equations. MODFLOW was designed to incorporate Picard iteration (Remson and others, 1971, p. 232) as a means to deal with non linearities. When the flow equation is non linear, some of the terms in [A] and {q} vary according to the value of head. In Picard iteration, the head-dependent terms in [A] and {q} for an iteration are calculated using the head calculated during the previous iteration, and the resulting simultaneous equations are solved as if they were linear. Hopefully, the head calculated in the new iteration is closer to the correct head. If so, then the correctness of the head-dependent terms will be improved in the next iteration. Further iterations should successively improve the accuracy of head and the head-dependent terms.

Iterative solvers are widely used in finite-difference ground-water flow models because iterative solvers execute faster in most situations and always use less memory than direct solvers. In today's computers, memory is not as much of a consideration as it used to be. Although today's computers are also much faster than older computers, execution time is still an important consideration in many modeling situations. Regardless of how long it takes to run a model, there is usually a direct benefit from having that model run as fast as possible. The comparative execution time between iterative and direct solvers depends on the linearity of the equations and the problem size.

For small, linear problems, a direct solver can often obtain a solution faster than an iterative solver. Larson (1978) concluded that a direct solver could be competitive with iterative solvers for two-dimensional problems with approximately 3000 variable-head cells. For iterative solvers, the execution time for a single iteration is proportional to the number of equations; however, there is normally a small increase in the required iterations as the number of equations increases, which causes slightly greater than linear growth in total execution time as the number of equations increases. For direct solution using Gaussian elimination, execution time is approximately proportional to the product of the number of equations and the square of the product of the two smallest grid dimensions. This means that execution time grows more rapidly with grid size for a direct solver than for the iterative solvers (unless the grid is expanding only in the direction of the largest dimension, which would mean that the two smallest grid dimensions stay constant).

The addition of Picard iteration to a direct solver considerably lengthens the execution time because each iteration requires the same computational time as required for one linear solution. Note that although the addition of Picard iteration to a direct solver for the purpose of solving non-linear equations means that the solver is no longer strictly speaking a direct solver, in this paper such a solver will still be called a direct solver. For a solver that is already iterative, Picard iteration can typically be easily incorporated into the iterative scheme. The non-linear terms will likely cause an iterative solver to converge less rapidly, depending on the degree of non linearity, but the effect on execution time generally is less than for direct solvers. In some situations, non-linear terms will cause an iterative solver to fail to converge, in which case a direct solver may be an attractive alternative in spite of the increased execution time that results from Picard iteration.

For transient simulations, an equation in the form of equation (1) must be solved for each time step. The relative efficiency of a direct solver compared to an iterative solver can be improved for transient problems if [A] is constant with time. The improved efficiency results because a large part of the work of Gaussian elimination does not have to be repeated once it has been done for the first time step. [A] will be constant with time if the problem is linear, if coefficients of aquifer head for stresses (for example, riverbed conductance) are constant with time, and if a constant time-step length is used.

This report documents a direct solver, called the D4 solver, for MODFLOW. MODFLOW consists of relatively independent groups of code called packages (McDonald and Harbaugh, 1988, p. 3-4). Each solver is a package, and the new solver is named the DE4 Package. Specific implementation details for the DE4 Package are described in the Module Documentation section. The implementation of the D4 solver in MODFLOW is similar to the implementation of the D4 solver in the USGS two-dimensional finite-difference ground-water flow model (Larson, 1978) and in the USGS three-dimensional finite-difference heat and solute transport model (Kipp, 1987).

The Input Instructions section is meant for anyone who might use the D4 solver. Using the D4 solver is as easy as using any of the other solvers. The Conceptualization and Implementation and the Module Documentation sections are meant for readers knowledgeable about direct solution methods.

CONCEPTUALIZATION AND IMPLEMENTATION

For solution, equation (1) is rewritten as an equation in which the unknown values are the change in head from one solution to the next. To facilitate this, a superscript will be used to indicate a particular solution in a sequence of solutions. If equation (1) is written for solution m, and $[A^m]\{h^{m-1}\}$ is subtracted from both sides, the result is

$$[A^m]\{\xi^m\} = \{b^m\} \tag{2}$$

where

$\{\xi^m\}$ is the change in head, $\{h^m\} - \{h^{m-1}\}$, and

$\{b^m\}$ is $\{q^m\} - [A^m]\{h^{m-1}\}$.

The head for solution m is therefore calculated as

$$\{h^m\} = \{h^{m-1}\} + \{\xi^m\} .$$

For the first solution of a simulation (m=1), $\{h^{m-1}\}$ is defined to be the initial head, which is specified as part of the input data for MODFLOW.

Gaussian elimination (Dahlquist and Bjorck, 1974, p. 147) is used for the direct solver described in this report. In Gaussian elimination, equation (2) is modified systematically so that all elements of [A] that are on the left side of the diagonal are transformed to zero, which is called upper triangular form. In this report, when [A] is modified in the process of Gaussian elimination, [A] is said to be "eliminated." Note that {b} is also modified in the process of Gaussian elimination. The elimination of [A] and the modification of {b} are usually viewed as a single process; however, for reasons that will be explained below, it is useful to consider these two aspects of Gaussian elimination separately. Once Gaussian elimination has been performed, the values of ξ are solved for using a method called back substitution (Dahlquist and Bjorck, 1974, p. 147). In back substitution, the values of ξ are solved for in reverse order starting with the last value. Because [A] is in upper triangular form and reverse solution order is used, each individual equation represented by matrix equation (2) contains only one unknown value of ξ that can be readily solved for.

The efficiency (computer time and memory) of Gaussian elimination is affected by the way that equations are ordered (or numbered). The standard method of ordering equations for a two-dimensional grid is shown in figure 1. Price and Coats (1974) found that the ordering scheme called D4 required much less computer execution time and memory than for standard ordering. They used this name because this is a diagonal ordering scheme, and it was the fourth scheme that they tried. D4 ordering for a two-dimensional grid proceeds along alternating diagonal lines, which is also illustrated in figure 1. For three dimensions, diagonal planes are used rather than lines (Price and Coats, 1974, p. 300).

Col. 1 Col. 5

Row 1	1	2	3	4	5
	6	7	8	9	10
	11	12	13	14	15
	16	17	18	19	20
	21	22	23	24	25
Row 6	26	27	28	29	30

Standard Ordering

Col. 1 Col. 5

1	16	2	18	5
17	3	19	6	22
4	20	7	23	10
21	8	24	11	27
9	25	12	28	14
26	13	29	15	30

D4 Ordering

Figure 1. -- Standard ordering and D4 ordering for a 2-dimensional grid with 6 rows and 5 columns

The form of [A] that results from standard ordering is shown in figure 2, and the form of [A] for D4 ordering is shown in figure 3. For D4 ordering, the upper part of [A] is already in upper triangular form, so Gaussian elimination is not required for this part prior to back substitution. The lower part of [A] reduces to a much more compact form after Gaussian elimination proceeds far enough to convert the original non-zero lower-left coefficients of [A] to zero. The reduced form of the lower part of [A], which is designated as [AL], is also shown in figure 3. The work to construct and eliminate [AL] is significantly less than for eliminating the [A] that results from standard ordering. Price and Coats (1974, p. 302) found that the work to solve the flow equations using D4 ordering is 0.2 to 0.5 of the work with standard ordering. The required memory is approximately half of what is required for standard ordering.

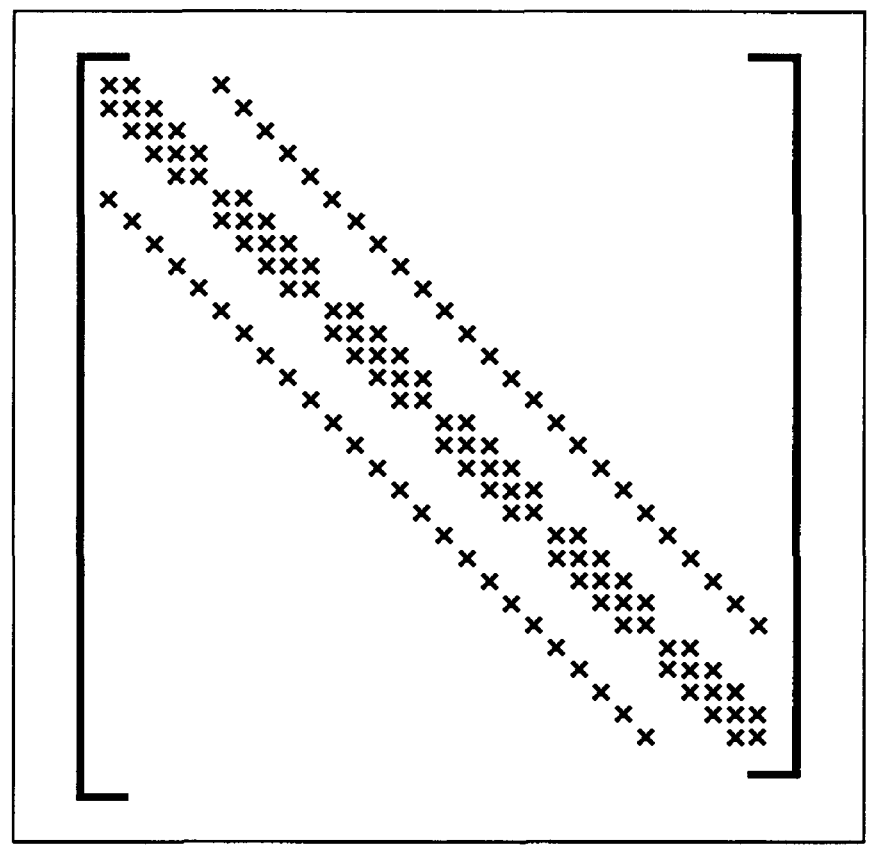

Figure 2. -- Non-zero coefficients in [A] using standard ordering for a grid with 6 rows and 5 columns.

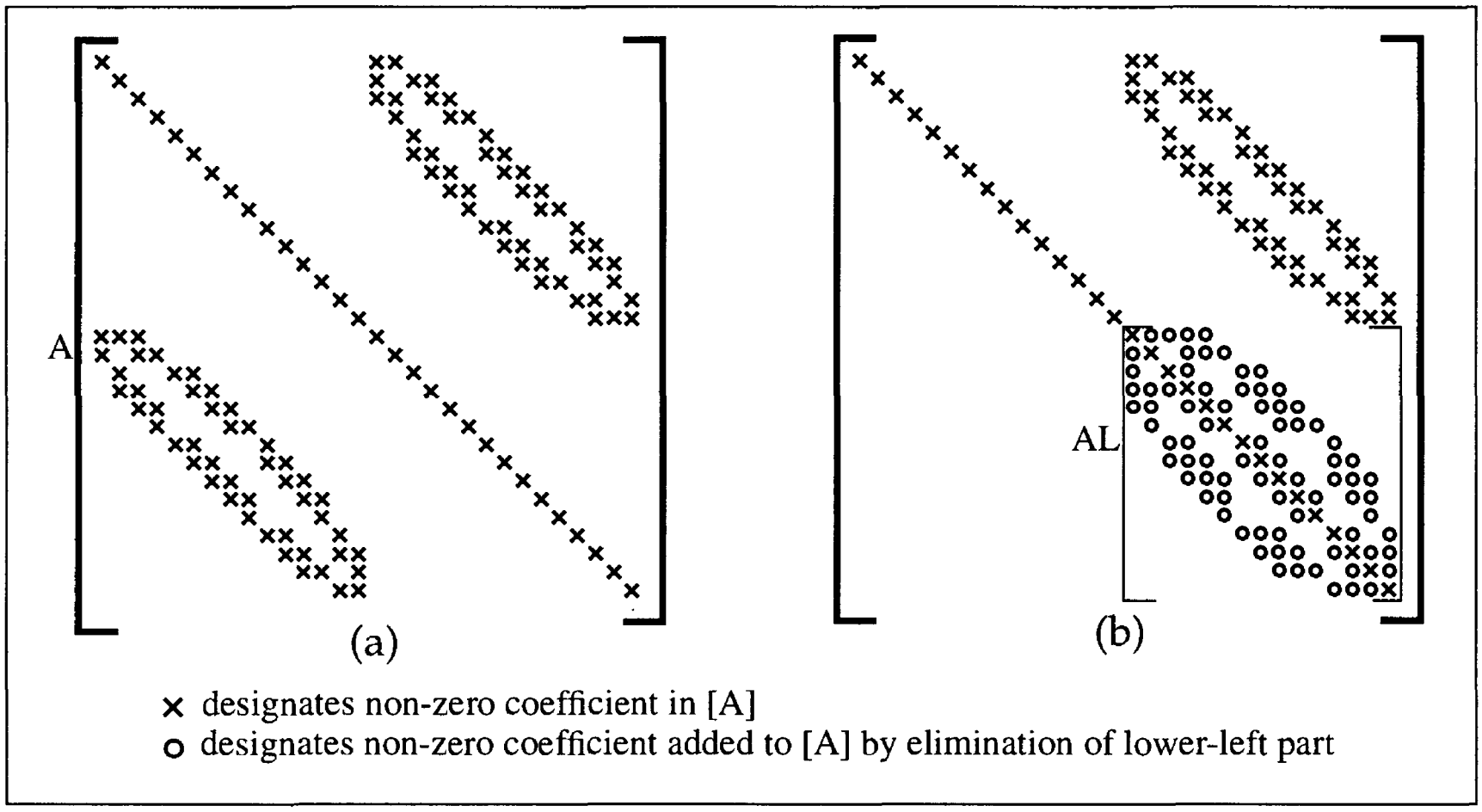

Figure 3. -- Non-zero coefficients in [A] using D4 ordering; (a) shows [A] prior to elimination, and (b) shows [A] after elimination has proceeded to the point that [AL] is created.

The following steps describe the process for solving equation (2) using the D4 direct solution method:

1. Calculate {b}.
2. If [A] has changed since the last solution, eliminate [A]. (This includes partial elimination of [A] to construct [AL] followed by elimination of [AL]).
3. Modify {b} to correspond to the eliminated [A].
4. Back substitute to obtain ξ.
5. Add ξ to the previous value of h to obtain the new value of h.

Notice that step 2 is done only if [A] has changed since the last solution. The computational work for step 2 is more than the total of the work for all the other steps, so efficiency is significantly improved in situations in which step 2 can be skipped most of the time. As described in the Introduction, [A] will be constant during any part of a transient simulation in which time steps have a constant length, coefficients of head for stress terms remain constant, and the flow equations are linear. {b} changes every time a new solution is required, however, so {b} must be modified to correspond to the eliminated [A] every time a new solution is obtained.

There is a further situation for which [A] remains constant for multiple solutions. Steps 2-4 are performed in single precision to minimize computer execution time and memory; however, this can lead to significant computer truncation errors in some situations, especially for large problems. That is, the calculated solution may not be sufficiently accurate. Because the finite-difference equations are formulated with head change as the unknown, successive iterations in which [A] is held constant and {b} is recalculated in double precision will improve accuracy. Whenever the error in head is large, the {b} calculated for the next iteration will be large, which will result in a new ξ that should reduce the error. As the solution becomes closer to the correct answer, the {b} will become smaller and smaller. Each of these iterations will be relatively fast because [A] is held constant and the elimination of [A] (step 2) is skipped. This type of iteration is called internal iteration because it is done entirely within the D4 solver package. The D4 solver for MODFLOW was designed so that internal iteration can be used when needed, as indicated by a large water budget error.

As already mentioned in the Introduction, direct solvers require iteration to solve non-linear equations. For this situation, the user can specify that multiple Picard iterations should occur each time step. The six solution steps are repeated for each Picard iteration, including the elimination of [A]; thus, the solver is least efficient in this situation. Within the D4 solver, Picard iterations are called external iterations.

Note that there is no need to have internal and external iterations in the same simulation. External iterations are required only when there are non linearities. Thus, if the problem is linear, the only reason for iteration would be to avoid computer truncation error, and this can be dealt with entirely through internal iteration. If external iteration is required because of non linearities, then {b} is recalculated in double precision for each external iteration. These repeated double precision calculations of {b} will reduce errors caused by the single precision calculations without the need for internal iteration.

For all problems, advantage is gained because [A] is symmetric in MODFLOW. As a result, [AL] is also symmetric. Computer storage and execution time are significantly less than they would be if these matrices were asymmetric.

MEMORY USAGE

The D4 solver uses a large amount of space in MODFLOW's X array. (See McDonald and Harbaugh [1988, p. 3-22] for information about how the X array is used in MODFLOW). The required space in X can be estimated as

$$N + 8n + (W+1)n/2$$

where

N is the total number of cells in the model grid (the product of the rows, columns, and layers),

n is the number of equations to be solved (N minus the no-flow and constant-head cells), and

W is the band width, which is approximately the product of the two smallest grid dimensions.

The D4 solver prints the actual number of elements used when it runs.

PERFORMANCE MEASUREMENTS

Several test simulations were made using the D4 and Preconditioned Conjugate-Gradient (PCG) (Hill, 1990) solvers. Performance comparisons are shown in table 1. (Details of the test simulations are described in the Appendix.) The execution time in the table is the total time spent in a solver, which does not include time for reading data, formulating equations, calculating the volumetric budget, or writing output. All execution times are for a Data General Aviion 300 workstation.[1] The solvers were adjusted in order to make both solutions for each simulation type have approximately the same accuracy as indicated by MODFLOW's budget summary. The execution times are presented in order to demonstrate the variation in relative performance that can occur between the two solvers depending on the size and linearity of the problem. However, the relative performance of the solvers for other kinds of problems may differ considerably. For a specific problem, the most reliable way to determine the optimum solver is to try all the available solvers.

---

[1]The use of product and firm names in this publication is for descriptive purposes only and does not imply endorsement by the U.S. Government.

Table 1. -- Comparison of D4 and PCG solvers

Simulation Type	Layers, rows, columns	Number of D4 solutions, eliminations of [A]	Total PCG iterations	D4 execution time (sec.)	PCG execution time (sec.)
A - Steady state, linear	2, 20, 30	2, 1	23	2.3	3.1
B - Steady state, non linear	2, 20, 30	4, 4	38	8.2	5.5
C - Transient, linear, constant time step	2, 20, 30	20, 1	108	6.9	15.2
D - Transient, non linear	2, 20, 30	30, 30	199	61.0	30.4
E - Steady state, linear	4, 40, 60	2, 1	44	226.5	49.2

Notice in table 1 that when D4 is used for Simulations A, C, and E, [A]is eliminated only once even though there are multiple solutions. Simulations A and E are steady state, which requires the solution of a single flow equation. For these simulations, two solutions (internal iterations) are performed, and the only purpose for the second solution is to improve accuracy. The second solution requires a relatively small amount of time because [A] does not require elimination the second time. Simulation C is transient, and the flow equations must be solved ten times -- one for each time step. However, [A] is the same for each time step because the problem is linear, the length of each time step is the same, and the stress parameters are constant throughout the single stress period. Therefore, [A] is eliminated only for the first time step. For each time step, two internal iterations are again performed in order to improve accuracy. The result is that a total of 20 solutions are calculated in Simulation C, but [A] is eliminated only once.

INPUT INSTRUCTIONS

The DE4 Package reads its input data from the file-unit number specified in IUNIT(10). Refer to McDonald and Harbaugh (1988, p. 3-25) for a description of the use of file-unit numbers in MODFLOW.

Input to the DE4 Package is defined using two numbered "items." Each item consists of several parameters that are specified in one record. Descriptions of the parameters follow the list of items. The format of the parameters is free format, which means that the exact location of parameter values in a record is not predefined. The parameter values must be specified in the indicated order, and adjacent values must be separated by one or more spaces, or by a comma surrounded optionally by any number of spaces. A zero value must be explicitly specified as zero; that is, zero cannot be specified by using blanks. All parameters are integers except for HCLOSE, which is a real number; integer values must not include a decimal point, but real numbers can. The items are read once at the beginning of the simulation:

1. ITMX MXUP MXLOW MXBW
2. IFREQ MUTD4 ACCL HCLOSE IPRD4

ITMX is the maximum number of iterations each time step. Specify ITMX = 1 if iteration is not desired. Ideally iteration would not be required for direct solution; however, it is necessary to iterate if the flow equation is non linear (see explanation for IFREQ=3) or if computer precision limitations result in inaccurate calculations as indicated by a large water budget error. For a non-linear flow equation, each iteration is equally time consuming because [A] is changed each iteration and Gaussian elimination is required after each change. This is called external iteration. For a linear equation, iteration is significantly faster because [A] is changed at most once per time step; thus, Gaussian elimination is required at most once per time step. This is called internal iteration.

MXUP is the maximum number of equations in the upper part of the equations to be solved. This value impacts the amount of memory used by the DE4 Package. If specified as 0, the program will calculate MXUP as half the number of cells in the model, which is an upper limit. The actual number of equations in the upper part will be less than half the number of cells whenever there are no-flow and constant-head cells because flow equations are not formulated for these cells. The DE4 Package prints the actual number of equations in the upper part when it runs. The printed value can be used for MXUP in future runs in order to minimize memory usage.

MXLOW is the maximum number of equations in the lower part of equations to be solved. This value impacts the amount of memory used by the DE4 Package. If specified as 0, the program will calculate MXLOW as half the number of cells in the model, which is an upper limit. The actual number of equations in the lower part will be less than half the number of cells whenever there are no-flow and constant-head cells because flow equations are not formulated for these cells. The DE4 Package prints the actual number of equations in the lower part when it runs. The printed value can be used for MXLOW in future runs in order to minimize memory usage.

MXBW is the maximum band width plus 1 of the [AL] matrix. This value impacts the amount of memory used by the DE4 Package. If specified as 0, the program will calculate MXBW as the product of the two smallest grid dimensions plus 1, which is an upper limit. The DE4 Package prints the actual band width plus 1 when it runs. The printed value can be used for MXBW in future runs in order to minimize memory usage.

IFREQ is a flag indicating the frequency at which coefficients in [A] change. This affects the efficiency of solution; significant work can be avoided if it is known that [A] remains constant all or part of the time.

IFREQ = 1 indicates that the flow equations are linear and that coefficients of simulated head for all stress terms are constant for all stress periods. To meet the linearity requirement, all model layers must be confined (which is specified in the Block-Centered Flow Package by setting parameter LAYCON equal to 0 for all layers), and there must be no formulations that change based upon head (such as seepage from a river changing from head dependent flow to a constant flow when head drops below the bottom of the riverbed). Examples of coefficients of simulated head for stress terms are riverbed conductance, drain conductance, maximum evapotranspiration rate, evapotranspiration extinction depth, and general-head boundary conductance.

IFREQ = 2 indicates that the flow equations are linear, but coefficients of simulated head for some stress terms may change at the start of each stress period. (See IFREQ=1 for information about linear equations.) Examples of coefficients of simulated head for stress terms are riverbed conductance, drain conductance, maximum evapotranspiration rate, evapotranspiration extinction depth, and general-head boundary conductance. For a simulation consisting of only one stress period, IFREQ=2 has the same meaning as IFREQ=1.

IFREQ = 3 indicates that a non-linear flow equation is being solved, which means that some terms in [A] depend on simulated head. Examples of head-dependent terms in [A] are transmissivity for water-table layers, which is based on saturated thickness; flow terms for rivers, drains, and evapotranspiration if they convert between head dependent flow and constant flow; and the change in storage coefficient when a cell converts between confined and unconfined. When a non-linear flow equation is being solved, external iteration (ITMX>1) is normally required in order to accurately approximate the non linearities. Note that when non linearities caused by water-table calculations are part of a simulation, there are not necessarily any obvious signs in the output from a simulation that does not use external iteration to indicate that iteration is needed. In particular, the budget error may be acceptably small without iteration even though there is significant error in head because of non linearity. To understand this, consider the water-table correction for transmissivity. Each iteration a new transmissivity is calculated based on the previous head. Then the flow equations are solved, and a budget is computed using the new head with the same transmissivities. No budget discrepancy results because heads are correct for the transmissivity being used at this point; however, the new heads may mean that there should be a significant change in transmissivity. The new transmissivity will not be calculated unless there is another iteration. Therefore, when one or more layers is under water-table conditions, iteration

should always be tried. The maximum change in head each iteration (printed by DE4 when IPRD4 = 1 and MUTD4 = 0) provides an indication of the impact of all non linearities.

MUTD4 is a flag that indicates the quantity of information that is printed when convergence information is printed for a time step.

MUTD4 = 0 indicates that the number of iterations in the time step and the maximum head change each iteration are printed.

MUTD4 = 1 indicates that only the number of iterations in the time step is printed.

MUTD4 = 2 indicates no information is printed.

ACCL is a multiplier for the computed head change for each iteration. Normally this value is 1. A value greater than 1 may be useful for improving the rate of convergence when using external iteration to solve non-linear problems (IFREQ=3). ACCL should always be 1 for linear problems. When ITMX=1, ACCL is changed to 1 regardless of the input value; however, a value must always be specified.

HCLOSE is the head change closure criterion. If iterating (ITMX > 1), iteration stops when the absolute value of head change at every node is less than or equal to HCLOSE. HCLOSE is not used if not iterating, but a value must always be specified.

IPRD4 is the time step interval for printing out convergence information when iterating (ITMX > 1). For example, if IPRD4 is 2, convergence information is printed every other time step. A value must always be specified even if not iterating.

MODULE DOCUMENTATION

The DE4 Package consists of six modules (subroutines). Each module is designated as version 5, which distinguishes this version from earlier prototypes. The modules are:

DE45AL -- Allocates space in the X array for all arrays needed by the DE4 Package, and determines the direction for equation ordering.

DE45RP -- Reads the control information for the DE4 Package.

DE45AP -- Performs one external iteration using the D4 solution method. There may be multiple internal iterations depending on user specifications. The major steps are:

1. Calculate and load coefficients into arrays, including {b}. Order the equations if [A] has changed since the last solution because the number of variable-head cells may have changed. (Module SDE45N is called to do the ordering.)
2. Eliminate [A] if it has changed since the last solution. (Module SDE45S is called to do this.)
3. Modify {b} to correspond to the eliminated [A]. (Module SDE45S is called to do this.)
4. Back substitute to obtain the head change. (Module SDE45S is called to do this.)
5. Add the head change to the previous value of head to obtain the new head.
6. Keep track of external and internal iterations. Call Module SDE45P to print head change for each iteration when required.

SDE45N -- Orders the equations.

SDE45P -- Prints maximum head change each iteration.

SDE45S -- Performs Gaussian elimination, modification of {b}, and back substitution.

Three statements, which are shown below, must be added to MODFLOW's Main program (McDonald and Harbaugh, 1988, p. 3-29) in order to incorporate the D4 solver into MODFLOW. These statements call the appropriate D4 modules whenever IUNIT(10) is greater than 0. Thus, D4 will not be used if IUNIT(10) is less than or equal to 0. The IUNIT array is read by the Basic Package. An IUNIT element other than 10 can be substituted provided that it is unused by any other package. To use a different element, replace 10 with the desired element in all five places where IUNIT(10) is found in the three statements.

Add the following statement immediately following the statement that calls the SIP Package Allocate Module (in the section of code between comments C4 and C5):

```
      IF(IUNIT(10).GT.0) CALL DE45AL(ISUM,LENX,LCAU,LCAL,LCIUPP,
     1            LCIEQP,LCD4B,LCLRCH,LCHDCG,
     2            MXUP,MXLOW,MXEQ,MXBW,IUNIT(10),ITMX,ID4DIR,
     3            NCOL,NROW,NLAY,IOUT,ID4DIM)
```

Add the following statement immediately following the statement that calls the SIP Package Read and Prepare Module (in the section of code between comments C6 and C7).

```
      IF(IUNIT(10).GT.0) CALL DE45RP(IUNIT(10),MXITER,NITER,ITMX,
     1             ACCL,HCLOSE,IFREQ,IPRD4,IOUT,MUTD4)
```

Add the following statement immediately following the statement that calls the SIP Package Approximate Module (in the section of code between comments C7C2B and C7C2C):

```
      IF(IUNIT(10).GT.0) CALL DE45AP(X(LCHNEW),X(LCIBOU),X(LCAU),
     1  X(LCAL),X(LCIUPP),X(LCIEQP),X(LCD4B),MXUP,MXLOW,MXEQ,MXBW,
     2  X(LCCR),X(LCCC),X(LCCV),X(LCHCOF),X(LCRHS),ACCL,KKITER,ITMX,
     3  MXITER,NITER,HCLOSE,IPRD4,ICNVG,NCOL,NROW,NLAY,IOUT,X(LCLRCH),
     4  X(LCHDCG),IFREQ,KKSTP,KKPER,DELT,NSTP,ID4DIR,ID4DIM,MUTD4)
```

Module DE45AL

Narrative for Module DE45AL

Module DE45AL allocates space in the X array for all arrays needed by the DE4 Package. Space is required for the equation number of each model cell, IEQPNT; B; the non-zero coefficients for the upper part of equations, AU; the [AL] matrix, AL; and the equation number of each coefficient in AU, IUPPNT.

1. Print a message identifying the DE4 Package.

2. Calculate default values for MXUP, MXLOW, and MXBW based on grid dimensions, and set the value of parameters ID4DIR and ID4DIM. ID4DIR can have a value from 1-6, which indicates the relative size of NCOL, NROW, and NLAY (e.g. ID4DIR=1 indicates NCOL≥NROW ≥NLAY). This is required in order to determine the maximum band width and the optimum way to order equations. ID4DIM is 5 if the model is two dimensional and 7 if the model is three dimensional; this is used to dimension arrays AU and IUPPNT.

3. Read and print ITMX, MXUP, MXLOW, and MXBW. These must be defined in order to determine how much space should be allocated. For each 0 or negative value, substitute the default value.

4. Allocate space in the X array. Save the value of the X array pointer (ISUM) in ISOLD prior to allocating space for DE4.

5. Calculate and print the space used in the X array. The space used by DE4 is ISUM-ISOLD. The total space used by all packages so far is ISUM-1.

6. Return.

Listing for Module DE45AL

```
      SUBROUTINE DE45AL(ISUM,LENX,LCAU,LCAL,LCIUPP,LCIEQP,LCD4B,LCLRCH,
     1            LCHDCG,MXUP,MXLOW,MXEQ,MXBW,INDE4,ITMX,ID4DIR,NCOL,
     2            NROW,NLAY,IOUT,ID4DIM)
C-----VERSION 10JAN1995 DE45AL
C     ******************************************************************
C     ALLOCATE STORAGE IN X ARRAY FOR D4 ARRAYS
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------PRINT A MESSAGE IDENTIFYING DE4 PACKAGE.
      WRITE(IOUT,1) INDE4
    1 FORMAT(1X,/1X,'DE45 -- D4 DIRECT SOLUTION PACKAGE',
     1    ', VERSION 5, 1/10/95  INPUT READ FROM UNIT',I3)
C
C2------CALCULATE DEFAULT VALUES FOR MXUP, MXLOW, AND MXBW.
C2------ALSO SET VALUES FOR ID4DIR (DIRECTION OF EQUATION ORDERING) AND
C2------ID4DIM (MAXIMUM NUMBER OF HEAD COEFFICIENTS FOR ONE EQUATION).
C2------ID4DIM=5 for a 2-D problem; ID4DIM=7 for 3-D.
      NODES=NCOL*NROW*NLAY
      NHALFU=(NODES-1)/2 +1
      NHALFL=NODES-NHALFU
      ID4DIM=7
      IF(NLAY.LE.NCOL .AND. NLAY.LE.NROW) THEN
         IF(NLAY.EQ.1) ID4DIM=5
         IF(NCOL.GE.NROW) THEN
            ID4DIR=1
            NBWGRD=NROW*NLAY+1
         ELSE
            ID4DIR=2
            NBWGRD=NCOL*NLAY+1
         END IF
      ELSE IF(NROW.LE.NCOL .AND. NROW.LE.NLAY) THEN
         IF(NROW.EQ.1) ID4DIM=5
         IF(NCOL.GE.NLAY) THEN
            ID4DIR=3
            NBWGRD=NROW*NLAY+1
         ELSE
            ID4DIR=4
            NBWGRD=NROW*NCOL+1
         END IF
      ELSE
         IF(NCOL.EQ.1) ID4DIM=5
         IF(NROW.GE.NLAY) THEN
            ID4DIR=5
            NBWGRD=NCOL*NLAY+1
         ELSE
            ID4DIR=6
            NBWGRD=NCOL*NROW+1
         END IF
      END IF
C
C3------READ AND PRINT ITMX, MXUP, MXLOW, MXBW.  FOR ANY ZERO OR
C3------NEGATIVE VALUES, SUBSTITUE THE DEFAULT VALUE.
      READ(INDE4,*) ITMX,MXUP,MXLOW,MXBW
      IF(ITMX.LT.1) ITMX=1
      WRITE(IOUT,3) ITMX
    3 FORMAT(1X,'MAXIMUM ITERATIONS (EXTERNAL OR INTERNAL) =',I3)
      IF(MXUP.LT.1) MXUP=NHALFU
      IF(MXLOW.LT.1) MXLOW=NHALFL
      MXEQ=MXUP+MXLOW
      IF(MXBW.LT.1) MXBW=NBWGRD
      WRITE(IOUT,4) MXUP,MXLOW,MXBW
    4 FORMAT(1X,'MAXIMUM EQUATIONS IN UPPER PART OF [A]:',I7,/
     1       1X,'MAXIMUM EQUATIONS IN LOWER PART OF [A]:',I7,/
     2       1X,'MAXIMUM BAND WIDTH OF [AL] PLUS 1:',I5)
C
```

```
C4------ALLOCATE SPACE FOR THE DE4 ARRAYS.
      ISOLD=ISUM
      LCAU=ISUM
      ISUM=ISUM+MXUP*ID4DIM
      LCIUPP=ISUM
      ISUM=ISUM+MXUP*ID4DIM
      LCAL=ISUM
      ISUM=ISUM+MXLOW*MXBW
      LCIEQP=ISUM
      ISUM=ISUM+NODES
      LCD4B=ISUM
      ISUM=ISUM+MXEQ
      LCLRCH=ISUM
      ISUM=ISUM+ITMX*3
      LCHDCG=ISUM
      ISUM=ISUM+ITMX
      ID4SP=ISUM-ISOLD
C
C5------CALCULATE AND PRINT THE SPACE USED IN THE X ARRAY.
      WRITE(IOUT,5) ID4SP
    5 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY DE4')
      ISUM1=ISUM-1
      WRITE(IOUT,6) ISUM1,LENX
    6 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,7)
    7 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C6------RETURN.
      RETURN
      END
```

List of Variables for Module DE45AL

Variable	Range	Definition
ID4DIM	Package	A parameter used to dimension arrays AU and IUPPNT: 5 -- for a 2-dimensional grid. 7 -- for a 3-dimensional grid.
ID4DIR	Package	Flag that indicates the relative size of NCOL, NROW, and NLAY: 1 -- NLAY≤NROW≤NCOL 2 -- NLAY≤NCOL<NROW 3 -- NROW<NLAY≤NCOL 4 -- NROW≤NCOL<NLAY 5 -- NCOL<NLAY≤NROW 6 -- NCOL<NROW<NLAY
ID4SP	Module	Number of elements in the X array allocated for the D4 solver.
INDE4	Package	Primary unit number from which input for this package will be read.
IOUT	Global	Primary unit number for all printed output.
ISOLD	Module	Value of ISUM upon entry to this module.
ISUM	Global	Element number of the lowest element in the X array that has not yet been allocated. When space is allocated in the X array, the size of the allocation is added to ISUM.
ISUM1	Module	ISUM-1
ITMX	Package	The maximum number of iterations (internal or external) allowed for one time step.
LCAL	Package	Location in the X array of the first element of array AL.
LCAU	Package	Location in the X array of the first element of array AU.
LCD4B	Package	Location in the X array of the first element of array B.
LCHDCG	Package	Location in the X array of the first element of array HDCG.
LCIEQP	Package	Location in the X array of the first element of array IEQPNT.
LCIUPP	Package	Location in the X array of the first element of array IUPPNT.
LCLRCH	Package	Location in the X array of the first element of array LRCH.
LENX	Global	Number of elements in the X array. This should always be equal to the dimension of X specified in the MAIN program.
MXBW	Package	The maximum allowed value for the band width plus one.
MXEQ	Package	MXUP+MXLOW
MXLOW	Package	The maximum allowed number of equations in the lower part of [A].
MXUP	Package	The maximum allowed number of equations in the upper part of [A].
NBWGRD	Module	One plus the product of the two smallest grid dimensions. This is the maximum possible value for band width plus one.
NCOL	Global	Number of columns in the grid.
NHALFL	Module	NODES-NHALFU -- the maximum possible number of equations in the lower part of [A].
NHALFU	Module	Half the number of nodes in the grid, rounded up. This is the maximum possible number of equations in the upper part of [A].
NLAY	Global	Number of layers in the grid.
NODES	Module	Number of nodes in the grid (NCOL*NROW*NLAY).
NROW	Global	Number of rows in the grid.

Module DE45RP

Narrative for Module DE45RP

Module DE45RP reads data for the DE4 Package. Variables to control iteration are calculated. NITER is the maximum number of internal iterations, and MXITER is the maximum number of external iterations.

1. Define values for constant variables ONE and ZERO. Read the remaining data record. If ACCL is less than or equal to zero, substitute a value of 1. If IPRD4 is less than or equal to zero, substitute a time interval of 999 time steps. Print a message and stop the simulation if IFREQ is not in the range of 1-3.

2. Check to see if there is iteration. If ITMX >1, there is iteration, and step 3 is executed. Otherwise, there is no iteration, and step 4 is executed.

3. There is iteration. The type of iteration (internal or external) depends on the linearity of the problem as indicated by IFREQ. If IFREQ=3, the problem is non linear, and step 3A is executed. Otherwise, the problem is linear, and step 3B is executed.
   A. The problem is non linear, so iteration is external. Set MXITER=ITMX and NITER=1, and print a message stating that iteration is external. Skip to step 3C.
   B. The problem is linear, so iteration is internal. Set MXITER=1 and NITER=ITMX, and print a message stating that iteration is internal. Continue to step 3C.
   C. Print values that control iteration -- ITMX, ACCL, HCLOSE, and IPRD4. If MUTD4 is 1 or 2, print a message explaining how this limits the printing of convergence information. Skip to step 5.

4. There is no iteration. Set MXITER=1, NITER=1, and ACCL=1.0. Print a message stating that there is no iteration. If MUTD4 is 2, print a message explaining that head change will not be printed.

5. Print a message about frequency at which [A] is eliminated. This is useful because it affects efficiency of solution.

6. Return.

Listing for Module DE45RP

```
      SUBROUTINE DE45RP(INDE4,MXITER,NITER,ITMX,ACCL,HCLOSE,
     1             IFREQ,IPRD4,IOUT,MUTD4)
C-----VERSION 29SEPT1994 DE45RP
C     ******************************************************************
C     READ DATA FOR DE4
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------READ IFREQ, MUTD4, ACCL, HCLOSE, AND IPRD4
      ONE=1.
      ZERO=0.
      READ(INDE4,*) IFREQ,MUTD4,ACCL,HCLOSE,IPRD4
      IF(ACCL.LE.ZERO) ACCL=ONE
      IF(IPRD4.LE.0) IPRD4=999
      IF(MUTD4.LT.0 .OR. MUTD4.GT.2) MUTD4=0
      IF(IFREQ.LT.1 .OR. IFREQ.GT.3) THEN
         WRITE(IOUT,11) IFREQ
   11    FORMAT(1H0,'INVALID VALUE FOR IFREQ PARAMETER:',I8)
         STOP
      END IF
C
C2------CHECK TO SEE IF THERE IS ITERATION (ITMX>1).
      IF(ITMX.GT.1) THEN
C
C3------THERE IS ITERATION -- DETERMINE TYPE OF ITERATION BASED ON IFREG
C3------VALUE.
         IF(IFREQ.EQ.3) THEN
C
C3A-----EXTERNAL ITERATION -- SET ITERATION VARIABLES AND PRINT A
C3A-----MESSAGE.
            MXITER=ITMX
            NITER=1
            WRITE(IOUT,51)
   51       FORMAT(1H0,20X,'SOLUTION BY D4 DIRECT SOLVER WITH EXTERNAL',
     1        ' ITERATION'/21X,52('-')/)
C
C3B-----INTERNAL ITERATION -- SET ITERATION VARIABLES AND PRINT A
C3B-----MESSAGE.
         ELSE
            MXITER=1
            NITER=ITMX
            WRITE(IOUT,81)
   81       FORMAT(1H0,20X,'SOLUTION BY D4 DIRECT SOLVER WITH INTERNAL',
     1        ' ITERATION'/21X,52('-')/)
         END IF
C
C3C-----PRINT ITERATION INFORMATION.
         WRITE(IOUT,91) ITMX,ACCL,HCLOSE,IPRD4
   91    FORMAT(1X,'MAXIMUM ITERATIONS =',I3/
     1          1X,'RELAXATION-ACCELERATION PARAMETER =',F10.6/
     2          1X,'HEAD CHANGE CRITERION FOR CLOSURE =',G15.5/
     3          1X,'D4 PRINTOUT INTERVAL =',I4)
         IF(MUTD4.EQ.1) WRITE(IOUT,92)
   92    FORMAT(1X,
     1   'CONVERGENCE PRINTOUT WILL SHOW ONLY THE NUMBER OF ITERATIONS')
         IF(MUTD4.EQ.2) WRITE(IOUT,93)
   93    FORMAT(1X,'CONVERGENCE PRINTOUT WILL BE SUPPRESSED')
C
C4------NO ITERATION -- SET ITERATION VARIABLES AND PRINT A MESSAGE.
      ELSE
         MXITER=1
         NITER=1
         ACCL=ONE
         WRITE(IOUT,94)
   94    FORMAT(1H0,20X,
     1   'SOLUTION BY D4 DIRECT SOLVER WITH NO ITERATION'/21X,46('-')/)
```

```
          IF(MUTD4.EQ.2) WRITE(IOUT,95)
   95     FORMAT(1X,'PRINTOUT OF MAXIMUM HEAD CHANGE WILL BE SUPPRESSED')
        END IF
C
C5------PRINT MESSAGE ABOUT FREQUENCY AT WHICH [A] IS ELIMINATED.
        IF(IFREQ.EQ.3) WRITE(IOUT,102)
  102   FORMAT(1X,'NON-LINEAR PROBLEM -- [A] MATRIX ELIMINATED',
       1       ' EVERY TIME EQUATIONS ARE REFORMULATED')
        IF(IFREQ.NE.3) WRITE(IOUT,103) IFREQ
  103   FORMAT(1X,'LINEAR PROBLEM -- [A] MATRIX ELIMINATED ONLY',
       1       ' WHEN IT CHANGES -- IFREQ=',I1)
C
C6------RETURN.
        RETURN
        END
```

List of Variables for Module DE45RP

Variable	Range	Definition
ACCL	Package	Multiplier for the computed head change for each iteration.
HCLOSE	Package	Head change closure criterion.
IFREQ	Package	Flag indicating the frequency at which [A] changes. 1 -- [A] changes only when the time step size changes. 2 -- [A] can change at the start of each stress period and when the time step size changes. 3 -- a non-linear flow equation is being solved, which means that [A] can change each time module DE45AP is called.
INDE4	Package	Primary unit number from which input for this package will be read.
IOUT	Global	Primary unit number for all printed output.
IPRD4	Package	The time step interval for printing out convergence information when iterating (ITMX > 1). For example, if IPRD4 is 2, convergence information is printed every other time step.
ITMX	Package	The maximum number of iterations (internal or external) allowed for one time step.
MUTD4	Package	Flag that indicates the quantity of information that is printed when convergence information is printed for a time step. 0 -- the number of iterations in the time step and the maximum head change each iteration are printed. 1 -- only the number of iterations in the time step is printed. 2 -- no information is printed.
MXITER	Global	The maximum number of external iterations in a time step.
NITER	Package	The maximum number of internal iterations in a time step.
ONE	Module	The constant 1.
ZERO	Module	The constant 0.

Module DE45AP

Narrative for Module DE45AP

Module DE45AP performs one external iteration using the D4 solution method. There may be multiple internal iterations depending on user specifications. This module is complex even though submodules are called to do part of the work. {b} is calculated in double precision; the set of simultaneous equations is solved using single precision.

1. Initialize some variables, including ITYPE, which indicates if [A] requires Gaussian elimination. The need for elimination depends on whether [A] has changed since the last solution. The user-specified IFREQ indicates the frequency at which stress and conductance terms in [A] change. In addition, the time step length is used to indicate if the storage terms in [A] are changing. Elimination is required (ITYPE=0) unless one of the conditions is met:
   IFREQ=0 -- this can only be set under program control. When this value is set, [A] is not eliminated regardless of any other parameters.
   IFREQ=1 and it's not the first time step of the first stress period and the current time step length is equal to the length of the previous time step.
   IFREQ=2 and it's not the first time step of the current stress period and the current time step length is equal to the length of the previous time step.

2. This is the start of the loop for an internal iteration. Initialize variables that track the maximum head change.

3. Skip to step 5 if [A] does not require Gaussian elimination.

4. [A] requires Gaussian elimination.
   A. Call Module SDE45N to order equations. After equations are ordered, check to see if the actual numbers of equations in the upper and lower halves of [A] are greater than the allocated sizes. If so, print an error message and stop.
   B. Initialize all the off diagonal values of the upper matrix to 0 These values are stored as packed values in array AU. Array IUPPNT keeps track of the equation number for each of the values.

5. Loop through all cells calculating coefficients and loading arrays for solution. In all situations, B is calculated. When [A] requires elimination, then IUPPNT and the diagonal members of AU and AL are loaded. Precision is carefully controlled so that B is calculated in double precision.
   A. For each cell, there can be conductance coefficients to neighboring cells. Loop through the 6 neighboring cells accumulating components of {b}, saving the conductance coefficient, and saving the equation number of the neighboring cell. Then calculate {b} for the cell and store in B.

B. Check if [A] requires elimination. If so, check if the cell is in the upper or lower part of [A] by comparing the cell number to NUP, the number of equations in the upper part of [A]. For a cell in the lower part, load the diagonal of [A] into AL(1,n). For upper-part cells, load up to 7 components of AU. The conductances to variable-head and constant-head neighboring cells are stored in positions 2-7. These neighboring conductances are stored in order of increasing equation number, which depends on the direction of ordering. There are 6 possible directions of ordering; IDIR contains 6 sets of orders. The order number, ID4DIR, has already been determined in Module DE45AL based on grid dimensions. AU is packed from the left side; thus, any unused locations are on the right side. For each conductance loaded into AU, the corresponding equation number is loaded into IUPPNT. After the neighboring conductances have been loaded into AU, the number of neighboring conductances that were loaded is stored in IUPPNT(1,n), and the diagonal component of [A] is stored in AU(1,n).

6. If [A] does not require elimination, skip to step 7.
   A. [A] requires elimination, determine the band width plus 1, NBW. This is done by finding the minimum and maximum displacement from the diagonal of the off diagonal terms in AU. Because off-diagonal terms are placed in order of increasing equation number, the minimum displacement is determined by IUPPNT(mineq,n), where mineq is 2; and the maximum is determined by IUPPNT(maxeq,n), where maxeq is IUPPNT(1,n). Loop through all upper part cells to find the overall minimum and maximum displacements. NBW is the maximum minus the minimum plus 1.
   B. Compare the bandwidth plus one, number of upper-part equations, and number of lower-part equations to their previous values. If any are different, print them. Initially, the previous values are defined to be 0 in a DATA statement.
   C. Stop if the bandwidth plus 1 is greater than what the user specified as a maximum.
   D. Set the off-diagonal part of AL=0.

7. Call Module SDE45S to solve the equations for head change. Head change is returned in B.

8. Loop through all cells calculating head as the previous head plus the product of ACCL and head change. ACCL is the user-specified acceleration/relaxation parameter, which is usually 1. Also, find the location where the absolute value of head change is a maximum, and store this location and the value of head change. Precision is carefully controlled to maintain head as double precision while head change is single precision.

9. Skip to step 11 if NITER (the number of internal iterations) is 1.

10. Iterating internally because NITER>1. Save the location of the maximum of the absolute value of head change and the head change at that location. Check for convergence by comparing the maximum absolute value of head change to the user-specified HCLOSE.
    A. After setting ITYPE=1 to avoid elimination of [A], go back to step 2 to do another internal iteration if convergence has not occurred and more iterations are allowed.

B. Internal iteration is done either because of convergence or failure to converge within the required number of iterations. In either case, print convergence information unless MUTD4=2. If MUTD4=2, skip to step 10B4.
   1. Print a blank line if the current time step is the first step in a stress period. This is for ease of reading the printout.
   2. Print the number of iterations.
   3. Print a table of maximum head-change values and locations if MUTD4=0 and any of the following conditions are true: convergence failed, the current time step is the last step of the current stress period, or the head-printout interval (IPRD4) has been reached.
   4. Return.

11. Internal iteration is not being used (NITER=1), so check if external iteration is being used. If so, skip to step 13.

12. There is no iteration, so convergence is assumed by default. Set the convergence flag. If the user-specified flag, MUTD4, is not 2 print the location where the absolute value of head change was a maximum and the value of head change. Return.

13. Iterating externally because MXITER>1. Save the location of the maximum of the absolute value of head change and the head change at that location. Check for convergence by comparing the maximum absolute value of head change to the user-specified HCLOSE.
   A. Return if convergence has not occurred and more iterations are allowed.
   B. External iteration is done either because of convergence or failure to converge within the required number of iterations. In either case, print convergence information unless MUTD4 is 2. If MUTD4=2, skip to step 13B4.
      1. Print a blank line if the current time step is the first step in a stress period. This is for ease of reading the printout.
      2. Print the number of iterations.
      3. Print a table of maximum head-change values and locations if MUTD4=0 and any of the following conditions are true: convergence failed, the current time step is the last step of the current stress period, or the head-printout interval (IPRD4) has been reached.
      4. Return.

Listing for Module DE45AP

```
      SUBROUTINE DE45AP(HNEW,IBOUND,AU,AL,IUPPNT,IEQPNT,B,MXUP,MXLOW,
     1  MXEQ,MXBW,CR,CC,CV,HCOF,RHS,ACCL,KITER,ITMX,MXITER,NITER,HCLOSE,
     2  IPRD4,ICNVG,NCOL,NROW,NLAY,IOUT,LRCH,HDCG,IFREQ,KSTP,
     3  KPER,DELT,NSTP,ID4DIR,ID4DIM,MUTD4)
C-----VERSION 29SEPT1994 DE45AP
C     ******************************************************************
C     SOLVE FINITE-DIFFERENCE EQUATIONS FOR ONE EXTERNAL ITERATION.
C     MULTIPLE SOLUTIONS ARE MADE WHEN INTERNALLY ITERATING.
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION HNEW(NCOL,NROW,NLAY),AU(ID4DIM,MXUP),AL(MXBW,MXLOW),
     1          IEQPNT(NCOL,NROW,NLAY),IUPPNT(ID4DIM,MXUP),B(MXEQ),
     2          CR(NCOL,NROW,NLAY),CC(NCOL,NROW,NLAY),
     3          CV(NCOL,NROW,NLAY),HCOF(NCOL,NROW,NLAY),
     4          RHS(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),LRCH(3,ITMX),
     5          HDCG(ITMX)
C
      DOUBLE PRECISION HNEW,EE,COND,RR,DDH
C
      DIMENSION CND(6),IEQ(6),IDIR(6,6)
      SAVE DELTL,NBWL,NUPL,NLOWL
      DATA IDIR/1,2,3,4,5,6,
     1          2,1,3,4,6,5,
     2          1,3,2,5,4,6,
     3          3,1,2,5,6,4,
     4          2,3,1,6,4,5,
     5          3,2,1,6,5,4/
C
      DATA DELTL,NBWL,NUPL,NLOWL/0.,0,0,0/
C     ------------------------------------------------------------------
C
C1------INITIALIZE VARIABLES AND SET FLAG THAT INDICATES IF [A] REQUIRES
C1------ELIMINATION.
      ZERO=0.
      ICNVG=0
      NIT=0
      ITYPE=0
      IF(IFREQ.EQ.0) THEN
         ITYPE=1
      ELSE IF(IFREQ.EQ.2) THEN
         IF(KSTP.NE.1 .AND. DELT.EQ.DELTL) ITYPE=1
      ELSE IF(IFREQ.EQ.1) THEN
         IF((KPER.NE.1 .OR. KSTP.NE.1) .AND. DELT.EQ.DELTL) ITYPE=1
      END IF
      DELTL=DELT
C
C2------DO ONE INTERNAL ITERATION.
   10 NIT=NIT+1
      BIGA=ZERO
      BIG=ZERO
      IBIG=0
      JBIG=0
      KBIG=0
C
C3------GO TO STATEMENT 100 IF [A] DOES NOT REQUIRE ELIMINATION.
      IF (ITYPE.EQ.1) GO TO 100
C
C4------[A] REQUIRES ELIMINATION.
C4A-----CALL MODULE SDE45N TO ORDER EQUATIONS.  IF MXUP OR MXLOW ARE
C4A-----TOO SMALL, PRINT A MESSAGE AND STOP.
      CALL SDE45N(IEQPNT,IBOUND,NCOL,NROW,NLAY,ID4DIR,NUP,NLOW,NEQ)
      IF(NUP.GT.MXUP .OR. NLOW.GT.MXLOW) THEN
         WRITE(IOUT,41) NUP,NLOW
   41    FORMAT(1X,'INSUFFICIENT MEMORY FOR DE4 SOLVER:',/
     1          1X,'MXUP MUST BE AT LEAST',I8,/
     1          1X,'MXLOW MUST BE AT LEAST',I8)
         STOP
```

```
      END IF
C
C4B-----INITIALIZE AU.
      DO 50 I=1,NUP
      DO 50 J=2,ID4DIM
      AU(J,I)=ZERO
   50 CONTINUE
C
C5------LOOP THROUGH ALL CELLS CALCULATING COEFFICIENTS AND LOADING
C5------ARRAYS FOR SOLUTION.  THE RESIDUAL MUST ALWAYS BE CALCULATED
C5------AND LOADED IN B; IUPPNT, AU, AND AL(1,n) ARE CALCULATED AND
C5------LOADED ONLY IF [A] REQUIRES ELIMINATION.
  100 DO 310 K=1,NLAY
      DO 310 I=1,NROW
      DO 310 J=1,NCOL
      IR=IEQPNT(J,I,K)
      IF(IR.EQ.0) GO TO 310
C
C5A-----CALCULATE AND LOAD B.
      DO 110 N=1,6
      IEQ(N)=0
  110 CONTINUE
      RR=RHS(J,I,K)
      EE=HCOF(J,I,K)
      IF(J.NE.1) THEN
         CLF=CR(J-1,I,K)
         COND=CLF
         RR=RR-COND*HNEW(J-1,I,K)
         EE=EE-COND
         CND(1)=CLF
         IEQ(1)=IEQPNT(J-1,I,K)
      END IF
      IF(I.NE.1) THEN
         CBK=CC(J,I-1,K)
         COND=CBK
         RR=RR-COND*HNEW(J,I-1,K)
         EE=EE-COND
         CND(2)=CBK
         IEQ(2)=IEQPNT(J,I-1,K)
END IF
IF(K.NE.1) THEN
         CUP=CV(J,I,K-1)
         COND=CUP
         RR=RR-COND*HNEW(J,I,K-1)
         EE=EE-COND
         CND(3)=CUP
         IEQ(3)=IEQPNT(J,I,K-1)
END IF
IF(K.NE.NLAY) THEN
         CDN=CV(J,I,K)
         COND=CDN
         RR=RR-COND*HNEW(J,I,K+1)
         EE=EE-COND
         CND(4)=CDN
         IEQ(4)=IEQPNT(J,I,K+1)
END IF
IF(I.NE.NROW) THEN
         CFR=CC(J,I,K)
         COND=CFR
         RR=RR-COND*HNEW(J,I+1,K)
         EE=EE-COND
         CND(5)=CFR
         IEQ(5)=IEQPNT(J,I+1,K)
END IF
IF(J.NE.NCOL) THEN
         CRT=CR(J,I,K)
         COND=CRT
         RR=RR-COND*HNEW(J+1,I,K)
         EE=EE-COND
         CND(6)=CRT
         IEQ(6)=IEQPNT(J+1,I,K)
END IF
```

```
      B(IR)=RR-EE*HNEW(J,I,K)
C
C5B-----IF [A] REQUIRES ELIMINATION, LOAD AL(1,N) AND AU
      IF (ITYPE.EQ.1) GO TO 310
      IF (IR.GT.NUP) THEN
         IRR=IR-NUP
         AL(1,IRR)=EE
      ELSE
         N=1
         DO 305 II=1,6
            M=IDIR(II,ID4DIR)
            L=IEQ(M)
            IF(L.NE.0) THEN
               N=N+1
               IUPPNT(N,IR)=L
               AU(N,IR)=CND(M)
            END IF
  305    CONTINUE
         AU(1,IR)=EE
         IUPPNT(1,IR)=N
      END IF
  310 CONTINUE
C
C6------IF [A] DOES NOT REQUIRE ELIMINATION, SKIP TO STATEMENT 380.
      IF(ITYPE.EQ.1) GO TO 380
C6A-----[A] REQUIRES ELIMINATION -- DETERMINE BAND WIDTH + 1.
      MNN=999999
      MXN=0
      DO 350 I=1,NUP
      L=IUPPNT(1,I)
      IF(L.LT.2) GO TO 350
      N=IUPPNT(2,I)-I
      IF(N.LT.MNN) MNN=N
      N=IUPPNT(L,I)-I
      IF(N.GT.MXN) MXN=N
  350 CONTINUE
      NBW=MXN-MNN+1
C
C6B-----WRITE BAND WIDTH + 1 AND NUMBER OF EQUATIONS IF ANY HAVE CHANGED.
      IF(NUP.NE.NUPL .OR. NLOW.NE.NLOWL .OR. NBW.NE.NBWL) THEN
         WRITE(IOUT,351) NUP,NLOW,NBW
  351    FORMAT(1X,/1X,I7,' UPPER PART EQS.',
     1      I10,' LOWER PART EQS.    BAND WIDTH + 1 =',I5)
         NUPL=NUP
         NLOWL=NLOW
         NBWL=NBW
      END IF
C
C6C-------STOP IF BAND WIDTH EXCEEDS USER-SPECIFIED SIZE.
      IF(NBW.GT.MXBW) THEN
         WRITE(IOUT,353) NBW
  353    FORMAT(1X,'INSUFFICIENT MEMORY FOR DE4 SOLVER:',/
     1          1X,'MXBW MUST BE AT LEAST',I5)
         STOP
      END IF
C
C6D-------INITIALIZE OFF DIAGONAL PART OF AL.
      DO 360 I=1,NLOW
      DO 360 J=2,NBW
      AL(J,I)=ZERO
  360 CONTINUE
C
C7------CALL MODULE SDE45S TO SOLVE EQUATIONS FOR HEAD CHANGE.
  380 CALL SDE45S(AU,AL,IUPPNT,B,NUP,NLOW,NEQ,MXBW,NBW,ITYPE,ID4DIM)
C
C8------CALCULATE NEW HEAD FROM HEAD CHANGE AND FIND MAXIMUM CHANGE.
      DO 400 K=1,NLAY
      DO 400 I=1,NROW
      DO 400 J=1,NCOL
      L=IEQPNT(J,I,K)
      IF(L.EQ.0) GO TO 400
      DH=ACCL*B(L)
```

```
      TCHK=ABS(DH)
      IF(TCHK.GT.BIGA) THEN
         BIGA=TCHK
         BIG=DH
         IBIG=I
         JBIG=J
         KBIG=K
      END IF
      DDH=DH
      HNEW(J,I,K)=HNEW(J,I,K)+DDH
  400 CONTINUE
C
C9------IF THE NUMBER OF INTERNAL ITERATIONS IS 1, GO TO STATEMENT 500
      IF(NITER.EQ.1) GO TO 500
C
C10-----THE NUMBER OF INTERNAL ITERATIONS IS GREATER THAN 1, SO MUST BE
C10-----ITERATING INTERNALLY.  KEEP TRACK OF MAXIMUM HEAD CHANGE AND
C10-----CHECK FOR CONVERGENCE.
      LRCH(1,NIT)=KBIG
      LRCH(2,NIT)=IBIG
      LRCH(3,NIT)=JBIG
      HDCG(NIT)=BIG
      IF(ABS(BIG).LE.HCLOSE) ICNVG=1
C
C10A----IF NOT CONVERGED AND MAXIMUM ITERATIONS HAS NOT BEEN REACHED,
C10A----GO BACK AND DO ANOTHER INTERNAL ITERATION.  SET ITYPE=1 TO
C10A----AVOID REFORMULATION OF [A].
      ITYPE=1
      IF(ICNVG.EQ.0 .AND. NIT.NE.NITER) GO TO 10
C
C10B----INTERNAL ITERATION IS DONE EITHER BECAUSE CONVERGENCE IS REACHED
C10B----OR BECAUSE MAX. ITERATIONS EXCEEDED.  PRINT CONVERGENCE
C10B----INFORMATION UNLESS IMUTD4=2.
      IF(MUTD4.NE.2) THEN
C10B1---PRINT A BLANK LINE IF THIS IS THE FIRST TIME STEP.
         IF(KSTP.EQ.1) WRITE(IOUT,601)
C10B2---PRINT NUMBER OF ITERATIONS.
         WRITE(IOUT,751) NIT,KSTP,KPER
  751    FORMAT(1X,I5,' INTERNAL ITERATIONS FOR TIME STEP',I4,
     1        ' IN STRESS PERIOD',I3)
C10B3---IF MUTD4=0 AND
C10B3---IF FAILED TO CONVERGE OR LAST TIME STEP IN STRESS PERIOD OR
C10B3---IPRD4 INTERVAL IS MET, CALL MODULE SDE45P TO PRINT HEAD CHANGE.
         IF((MUTD4.EQ.0) .AND.
     1      (ICNVG.EQ.0 .OR. KSTP.EQ.NSTP .OR. MOD(KSTP,IPRD4).EQ.0))
     2         CALL SDE45P(HDCG,LRCH,NIT,IOUT)
      END IF
C10B4---RETURN.
      RETURN
C
C11-----THERE ARE NO INTERNAL ITERATIONS, SO THERE MUST EITHER BE
C11-----EXTERNAL ITERATION OR NO ITERATION.  IF THERE IS EXTERNAL
C11-----ITERATION, GO TO STATEMENT 600.
  500 IF(MXITER.NE.1) GO TO 600
C
C12-----NO ITERATION (NITER=1 AND MXITER=1).  SET FLAG TO INDICATE
C12-----CONVERGENCE HAS OCCURRED, AND PRINT MAXIMUM HEAD CHANGE UNLESS
C12-----MUTD4=2.
      ICNVG=1
      IF(MUTD4.NE.2) WRITE(IOUT,501) KSTP,KPER,BIG,KBIG,IBIG,JBIG
  501 FORMAT(1X,/1X,'MAXIMUM HEAD CHANGE IN TIME STEP',I3,
     1       ' OF STRESS PERIOD',I3,' =',G15.5,
     2       ' AT LAYER =',I3,', ROW =',I3,', COL=',I3)
      RETURN
C
C13-----EXTERNAL ITERATION.  KEEP TRACK OF MAXIMUM HEAD CHANGE AND SET
C13-----CONVERGENCE FLAG IF CONVERGENCE OCCURRED.
  600 LRCH(1,KITER)=KBIG
      LRCH(2,KITER)=IBIG
      LRCH(3,KITER)=JBIG
      HDCG(KITER)=BIG
      IF(ABS(BIG).LE.HCLOSE) ICNVG=1
```

```
C13A----RETURN IF NO CONVERGENCE AND MAX. ITERATIONS NOT EXCEEDED.
      IF(ICNVG.EQ.0 .AND. KITER.NE.MXITER) RETURN
C13B----EXTERNAL ITERATION IS DONE EITHER BECAUSE CONVERGENCE IS REACHED
C13B----OR BECAUSE MAX. ITERATIONS EXCEEDED.  PRINT CONVERGENCE
C13B----INFORMATION UNLESS IMUTD4=2.
      IF(MUTD4.NE.2) THEN
C13B1---PRINT A BLANK LINE IF THIS IS THE FIRST TIME STEP.
         IF(KSTP.EQ.1) WRITE(IOUT,601)
  601    FORMAT(1X)
C13B2---PRINT NUMBER OF ITERATIONS.
         WRITE(IOUT,602) KITER,KSTP,KPER
  602    FORMAT(1X,I5,' EXTERNAL ITERATIONS FOR TIME STEP',I4,
     1        ' IN STRESS PERIOD',I3)
C13B3---IF MUTD4=0 AND
C13B3---IF FAILED TO CONVERGE OR LAST TIME STEP IN STRESS PERIOD OR
C13B3---IPRD4 INTERVAL IS MET, CALL MODULE SDE45P TO PRINT HEAD CHANGE.
         IF((MUTD4.EQ.0) .AND.
     1      (ICNVG.EQ.0 .OR. KSTP.EQ.NSTP .OR. MOD(KSTP,IPRD4).EQ.0))
     2         CALL SDE45P(HDCG,LRCH,KITER,IOUT)
      END IF
C13B4---Return.
      RETURN
C
      END
```

List of Variables for Module DE45AP

Variable	Range	Definition
ACCL	Package	Multiplier for the computed head change for each iteration.
AL	Package	DIMENSION (MXBW,MXLOW), the matrix [AL].
AU	Package	DIMENSION (ID4DIM,MXUP), the upper part of [A].
B	Package	DIMENSION (MXEQ), the vector {b}.
BIG	Module	The head change (including the sign) at the cell that has the largest absolute value of head change for an iteration.
BIGA	Module	The largest absolute value of head change for an iteration.
CBK	Module	Conductance from the node for which the flow equation is being formulated to the adjacent node that is one row back.
CC	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction.
CDN	Module	Conductance from the node for which the flow equation is being formulated to the adjacent node that is one layer down.
CFR	Module	Conductance from the node for which the flow equation is being formulated to the adjacent node that is one row forward.
CLF	Module	Conductance from the node for which the flow equation is being formulated to the adjacent node that is one column back.
CND	Module	DIMENSION (6), Conductance to the 6 nodes that are adjacent to the node for which the flow equation is being formulated.
COND	Module	Double precision equivalent of conductance, which is used to accumulate EE and RR.
CR	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction.
CRT	Module	Conductance from the node for which the flow equation is being formulated to the adjacent node that is one column forward.
CUP	Module	Conductance from the node for which the flow equation is being formulated to the adjacent node that is one layer up.
CV	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the vertical direction. Although CV is dimensioned to the size of the grid, space exists only for NLAY-1 layers.
DDH	Module	Double precision form of DH.
DELT	Global	Length of the current time step.
DELTL	Module	Length of the previous time step.
DH	Module	The head change at a cell for an iteration.
EE	Module	Accumulator for all the coefficients of head at the cell for which the flow equation is being defined. This is the diagonal term in AU and AL.
HCLOSE	Package	The head change closure criterion.
HCOF	Global	DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the finite-difference equation.
HDCG	Package	DIMENSION (MXITER), Maximum head change for each iteration.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell.
I	Module	Do-loop index.
IBIG	Module	Row index for the cell that has the largest absolute value of head change.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell: < 0 -- constant-head cell = 0 -- no-flow cell > 0 -- variable-head cell

Variable	Range	Definition
ICNVG	Global	Flag that is set to one when convergence has occurred.
ID4DIM	Package	A parameter used to dimension arrays AU and IUPPNT: 5 -- for a 2-dimensional grid. 7 -- for a 3-dimensional grid.
ID4DIR	Package	Flag that indicates the relative size of NCOL, NROW, and NLAY: 1 -- NLAY≤NROW≤NCOL 2 -- NLAY≤NCOL<NROW 3 -- NROW<NLAY≤NCOL 4 -- NROW≤NCOL<NLAY 5 -- NCOL<NLAY≤NROW 6 -- NCOL<NROW<NLAY
IDIR	Module	DIMENSION (6,6), Contains 6 sets of order numbers for storing coefficients in AU. These relate to the 6 ways that equations are ordered depending on grid dimensions. The IDIR order number are defined so that coefficients in AU are stored in the order of increasing equation number.
IEQ	Module	DIMENSION (6), The equation number for the coefficients being stored in AU.
IEQPNT	Package	DIMENSION (NCOL,NROW,NLAY), D4 order number for model cells.
IFREQ	Package	Flag indicating the frequency at which [A] changes. 0 -- [A] has not changed since the previous solution. 1 -- [A] changes only when the time step size changes. 2 -- [A] can change at the start of each stress period and when the time step size changes. 3 -- a non-linear flow equation is being solved, which means that [A] can change each time module DE45AP is called.
II	Module	Do-loop index.
IOUT	Global	Primary unit number for all printed output.
IPRD4	Package	The time step interval for printing out convergence information when iterating (ITMX > 1). For example, if IPRD4 is 2, convergence information is printed every other time step.
IR	Module	Temporary storage for IEQPNT(J,I,K), which is the D4 order number for the cell whose flow equation is being formulated.
IRR	Module	IR-NUP, which is the row in AL that corresponds to D4 order number IR.
ITMX	Package	The maximum number of iterations (internal or external) allowed for one time step.
ITYPE	Package	Flag that indicates if [A] requires elimination; 0 indicates elimination is required, and 1 indicates elimination is not required.
IUPPNT	Package	DIMENSION (ID4DIM,MXUP), Contains the number of off-diagonal coefficients and the equation numbers of the off-diagonal coefficients that are stored in AU
J	Module	Do-loop index.
JBIG	Module	Column index for the cell that has the largest absolute value of head change.
K	Module	Do-loop index.
KBIG	Module	Layer index for the cell that has the largest absolute value of head change.
KITER	Global	External iteration counter. Reset at the start of each time step.
Variable	Range	Definition
KPER	Global	Stress period counter.

KSTP	Global	Time step counter. Reset at the start of each stress period.
L	Module	Temporary storage for array elements.
LRCH	Package	DIMENSION (3,MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.
M	Module	Temporary storage for array elements.
MNN	Module	The minimum offset from the diagonal of the first off-diagonal term in each row of AU.
MUTD4	Package	Flag that indicates the quantity of information that is printed when convergence information is printed for a time step. 0 -- the number of iterations in the time step and the maximum head change each iteration are printed. 1 -- only the number of iterations in the time step is printed. 2 -- no information is printed.
MXBW	Package	The maximum allowed value for the band width plus one.
MXEQ	Package	MXUP+MXLOW
MXITER	Global	Maximum allowed external iterations.
MXLOW	Package	The maximum allowed number of equations in the lower part of [A].
MXN	Module	The maximum offset from the diagonal of the last off-diagonal term in each row of AU.
MXUP	Package	The maximum allowed number of equations in the upper part of [A].
N	Module	Temporary storage.
NBW	Package	The bandwidth plus one of [AL].
NBWL	Package	Value of NBW from previous call to this module.
NCOL	Global	Number of columns in the grid.
NEQ	Package	The total number of equations to be solved.
NIT	Module	Counter for the number of internal iterations.
NITER	Package	The maximum number of internal iterations in a time step.
NLAY	Global	Number of layers in the grid.
NLOW	Package	The number of equations in the lower part of [A].
NLOWL	Module	Value of NLOW from previous call to this module.
NROW	Global	Number of rows in the grid.
NSTP	Global	Number of time steps in the current stress period.
NUP	Package	The number of equations in the upper part of [A].
NUPL	Module	Value of NUP from previous call to this module.
RHS	Global	DIMENSION (NCOL,NROW,NLAY), Right-hand side of the finite-difference equation.
RR	Module	Accumulator for terms in {b} that are not coefficients of head at the cell for which the flow equation is being formulated.
TCHK	Module	The absolute value of head change at a cell for an iteration.
ZERO	Module	The constant 0.

Module SDE45N

Narrative for Module SDE45N

Module SDE45N orders the equations. There are 6 possible ways to order based on the relative sizes of the grid dimensions. Variable ID4DIR has already been set by Module DE45AL to a value in the range of 1-6 based on grid dimensions.

1. Calculate the maximum plane number, which is the sum of the 3 grid dimensions. A plane is defined as the set of all cells for which the sum of the cell indices equals the plane number. Thus, the minimum plane number is 3. Initialize IEQPNT, which will hold the equation number for each model cell. If a cell is constant head or no flow, IEQPNT will be 0.
2. Skip to one of 6 sections of code depending on the value of ID4DIR.
3. ID4DIR=1, which indicates NCOL≥NROW≥NLAY.
   A. Equations are ordered in alternate diagonal planes. Number the odd plane numbers first, starting with plane 3 and proceeding in order to the maximum odd plane number. Within a plane, cells are numbered in the order of decreasing layer number (this index is varied least rapidly), decreasing row number (this index is varied the 2nd least rapidly), and increasing column number (most rapidly varied index) . Set NUP equal to the number of equations in the upper part of equations, which is all equations in the odd planes.
   B. Repeat the same ordering process for even planes. Set NLOW equal to the number of equations in the lower part of equations, which is all equations in the even planes. Return.
4. ID4DIR=2, which indicates NROW>NCOL≥NLAY. Number as described for step 3, except change the order of indices to correspond to the relative size of the grid dimensions. That is, cells are numbered in the order of decreasing layer number, decreasing column number, and increasing row number.
5. ID4DIR=3, which indicates NCOL≥NLAY>NROW. Number as described for step 3, except change the order of indices to correspond to the relative size of the grid dimensions. That is, cells are numbered in the order of decreasing row number, decreasing layer number, and increasing column number.
6. ID4DIR=4, which indicates NLAY>NCOL≥NROW. Number as described for step 3, except change the order of indices to correspond to the relative size of the grid dimensions. That is, cells are numbered in the order of decreasing row number, decreasing column number, and increasing layer number.
7. ID4DIR=5, which indicates NROW≥NLAY>NCOL. Number as described for step 3, except change the order of indices to correspond to the relative size of the grid dimensions. That is, cells are numbered in the order of decreasing column number, decreasing layer number, and increasing row number.
8. ID4DIR=6, which indicates NLAY>NROW>NCOL. Number as described for step 3, except change the order of indices to correspond to the relative size of the grid dimensions. That is, cells are numbered in the order of decreasing column number, decreasing row number, and increasing layer number.

Listing for Module SDE45N

```
      SUBROUTINE SDE45N(IEQPNT,IBOUND,NCOL,NROW,NLAY,ID4DIR,NUP,NLOW,
     1           NEQ)
C
C-----VERSION 29SEPT1994 SDE45N
C     ******************************************************************
C     ORDER EQUATIONS USING D4 ORDERING
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION IEQPNT(NCOL,NROW,NLAY), IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------CALCULATE MAXIMUM PLANE NUMBER AND INITIALIZE EQUATION POINTERS.
      NPLANE=NCOL+NROW+NLAY
      DO 20 K=1,NLAY
      DO 20 I=1,NROW
      DO 20 J=1,NCOL
      IEQPNT(J,I,K)=0
   20 CONTINUE
      NEQ=0
C
C2------ORDER EQUATIONS BASED ON DIRECTION FLAG, ID4DIR
C2------Ordering is done as described by Price, H.S. and Coats, K.H.,
C2------1974,Direct methods in reservoir simulation: Soc. Petrol. Eng.
C2------Jour., June 1974, p. 295-308.
      GO TO (100,200,300,400,500,600) ID4DIR
      STOP
C
C3------DIRECTION 1 -- NCOL>or=NROW>or=NLAY
C3A-----Order equations with odd plane numbers.
  100 DO 130 N=3,NPLANE,2
      K1=N-2
      IF(K1.GT.NLAY) K1=NLAY
      K2=N-NCOL-NROW
      IF(K2.LT.1) K2=1
      DO 130 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NROW) I1=NROW
      I2=N-K-NCOL
      IF(I2.LT.1) I2=1
      DO 130 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(J,I,K).LE.0) GO TO 130
      NEQ=NEQ+1
      IEQPNT(J,I,K)=NEQ
  130 CONTINUE
      NUP=NEQ
C
C3B-----Order equations with even plane numbers.
      DO 140 N=4,NPLANE,2
      K1=N-2
      IF(K1.GT.NLAY) K1=NLAY
      K2=N-NCOL-NROW
      IF(K2.LT.1) K2=1
      DO 140 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NROW) I1=NROW
      I2=N-K-NCOL
      IF(I2.LT.1) I2=1
      DO 140 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(J,I,K).LE.0) GO TO 140
      NEQ=NEQ+1
      IEQPNT(J,I,K)=NEQ
  140 CONTINUE
      NLOW=NEQ-NUP
      RETURN
C
```

```
C4------DIRECTION 2 NROW>NCOL>or=NLAY
C4A-----Order equations with odd plane numbers.
  200 DO 230 N=3,NPLANE,2
      K1=N-2
      IF(K1.GT.NLAY) K1=NLAY
      K2=N-NCOL-NROW
      IF(K2.LT.1) K2=1
      DO 230 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NCOL) I1=NCOL
      I2=N-K-NROW
      IF(I2.LT.1) I2=1
      DO 230 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(I,J,K).LE.0) GO TO 230
      NEQ=NEQ+1
      IEQPNT(I,J,K)=NEQ
  230 CONTINUE
      NUP=NEQ
C
C4B-----Order equations with even plane numbers.
      DO 240 N=4,NPLANE,2
      K1=N-2
      IF(K1.GT.NLAY) K1=NLAY
      K2=N-NCOL-NROW
      IF(K2.LT.1) K2=1
      DO 240 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NCOL) I1=NCOL
      I2=N-K-NROW
      IF(I2.LT.1) I2=1
      DO 240 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(I,J,K).LE.0) GO TO 240
      NEQ=NEQ+1
      IEQPNT(I,J,K)=NEQ
  240 CONTINUE
      NLOW=NEQ-NUP
      RETURN
C
C-------DIRECTION 3 NCOL>or=NLAY>NROW
C5A-----Order equations with odd plane numbers.
  300 DO 330 N=3,NPLANE,2
      K1=N-2
      IF(K1.GT.NROW) K1=NROW
      K2=N-NCOL-NLAY
      IF(K2.LT.1) K2=1
      DO 330 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NLAY) I1=NLAY
      I2=N-K-NCOL
      IF(I2.LT.1) I2=1
      DO 330 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(J,K,I).LE.0) GO TO 330
      NEQ=NEQ+1
      IEQPNT(J,K,I)=NEQ
  330 CONTINUE
      NUP=NEQ
C
C5B-----Order equations with even plane numbers.
      DO 340 N=4,NPLANE,2
      K1=N-2
      IF(K1.GT.NROW) K1=NROW
      K2=N-NCOL-NLAY
      IF(K2.LT.1) K2=1
      DO 340 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NLAY) I1=NLAY
      I2=N-K-NCOL
      IF(I2.LT.1) I2=1
      DO 340 I=I1,I2,-1
```

```
      J=N-K-I
      IF(IBOUND(J,K,I).LE.0) GO TO 340
      NEQ=NEQ+1
      IEQPNT(J,K,I)=NEQ
  340 CONTINUE
      NLOW=NEQ-NUP
      RETURN
C
C6------DIRECTION 4 NLAY>NCOL>or=NROW
C6A-----Order equations with odd plane numbers.
  400 DO 430 N=3,NPLANE,2
      K1=N-2
      IF(K1.GT.NROW) K1=NROW
      K2=N-NCOL-NLAY
      IF(K2.LT.1) K2=1
      DO 430 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NCOL) I1=NCOL
      I2=N-K-NLAY
      IF(I2.LT.1) I2=1
      DO 430 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(I,K,J).LE.0) GO TO 430
      NEQ=NEQ+1
      IEQPNT(I,K,J)=NEQ
  430 CONTINUE
      NUP=NEQ
C
C6B-----Order equations with even plane numbers.
      DO 440 N=4,NPLANE,2
      K1=N-2
      IF(K1.GT.NROW) K1=NROW
      K2=N-NCOL-NLAY
      IF(K2.LT.1) K2=1
      DO 440 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NCOL) I1=NCOL
      I2=N-K-NLAY
      IF(I2.LT.1) I2=1
      DO 440 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(I,K,J).LE.0) GO TO 440
      NEQ=NEQ+1
      IEQPNT(I,K,J)=NEQ
  440 CONTINUE
      NLOW=NEQ-NUP
      RETURN
C
C7------DIRECTION 5 NROW>or=NLAY>NCOL
C7A-----Order equations with odd plane numbers.
  500 DO 530 N=3,NPLANE,2
      K1=N-2
      IF(K1.GT.NCOL) K1=NCOL
      K2=N-NROW-NLAY
      IF(K2.LT.1) K2=1
      DO 530 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NLAY) I1=NLAY
      I2=N-K-NROW
      IF(I2.LT.1) I2=1
      DO 530 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(K,J,I).LE.0) GO TO 530
      NEQ=NEQ+1
      IEQPNT(K,J,I)=NEQ
  530 CONTINUE
      NUP=NEQ
C
C7B-----Order equations with even plane numbers.
      DO 540 N=4,NPLANE,2
      K1=N-2
      IF(K1.GT.NCOL) K1=NCOL
```

```
      K2=N-NROW-NLAY
      IF(K2.LT.1) K2=1
      DO 540 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NLAY) I1=NLAY
      I2=N-K-NROW
      IF(I2.LT.1) I2=1
      DO 540 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(K,J,I).LE.0) GO TO 540
      NEQ=NEQ+1
      IEQPNT(K,J,I)=NEQ
  540 CONTINUE
      NLOW=NEQ-NUP
      RETURN
C
C-------DIRECTION 6 NLAY>NROW>NCOL
C8A-----Order equations with odd plane numbers.
  600 DO 630 N=3,NPLANE,2
      K1=N-2
      IF(K1.GT.NCOL) K1=NCOL
      K2=N-NROW-NLAY
      IF(K2.LT.1) K2=1
      DO 630 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NROW) I1=NROW
      I2=N-K-NLAY
      IF(I2.LT.1) I2=1
      DO 630 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(K,I,J).LE.0) GO TO 630
      NEQ=NEQ+1
      IEQPNT(K,I,J)=NEQ
  630 CONTINUE
      NUP=NEQ
C
C8B-----Order equations with even plane numbers.
      DO 640 N=4,NPLANE,2
      K1=N-2
      IF(K1.GT.NCOL) K1=NCOL
      K2=N-NROW-NLAY
      IF(K2.LT.1) K2=1
      DO 640 K=K1,K2,-1
      I1=N-K-1
      IF(I1.GT.NROW) I1=NROW
      I2=N-K-NLAY
      IF(I2.LT.1) I2=1
      DO 640 I=I1,I2,-1
      J=N-K-I
      IF(IBOUND(K,I,J).LE.0) GO TO 640
      NEQ=NEQ+1
      IEQPNT(K,I,J)=NEQ
  640 CONTINUE
      NLOW=NEQ-NUP
      RETURN
C
      END
```

List of Variables for Module SDE45N

Variable	Range	Definition
I	Module	Index for the second most rapidly changing index when numbering in a plane. Also, an index for rows.
I1	Module	Starting value for I index.
I2	Module	Ending value for I index.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell: < 0 -- constant-head cell = 0 -- no-flow cell > 0 -- variable-head cell
ID4DIR	Package	Flag that indicates the relative size of NCOL, NROW, and NLAY: 1 -- NLAY≤NROW≤NCOL 2 -- NLAY≤NCOL<NROW 3 -- NROW<NLAY≤NCOL 4 -- NROW≤NCOL<NLAY 5 -- NCOL<NLAY≤NROW 6 -- NCOL<NROW<NLAY
IEQPNT	Package	DIMENSION (NCOL,NROW,NLAY), Equation number for model cells.
J	Module	Index for the most rapidly changing index when numbering in a plane. Also, an index for columns.
K	Module	Index for the least rapidly changing index when numbering in a plane. Also, an index for layers.
K1	Module	Starting value for K index.
K2	Module	Ending value for K index.
N	Module	Index for plane numbers.
NCOL	Global	Number of columns in the grid.
NEQ	Package	The total number of equations to be solved.
NLAY	Global	Number of layers in the grid.
NLOW	Package	The number of equations in the lower part of [A].
NPLANE	Module	The maximum plane number, which is the sum of NCOL, NROW, and NLAY.
NROW	Global	Number of rows in the grid.
NUP	Package	The number of equations in the upper part of [A].

Module SDE45P

Narrative for Module SDE45P

Module SDE45P prints the location of the maximum absolute value of head change and the actual head change at this location for each iteration of a time step. This module is so short that there are no numbered comments.

Listing for Module SDE45P

```
      SUBROUTINE SDE45P(HDCG,LRCH,NUMIT,IOUT)
C
C
C-----VERSION 29SEPT1994 SDE45P
C     ******************************************************************
C     PRINT MAXIMUM HEAD CHANGE DURING EACH D4 ITERATION FOR A TIME STEP
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION HDCG(NUMIT), LRCH(3,NUMIT)
C     ------------------------------------------------------------------
C
      WRITE(IOUT,5)
    5 FORMAT(1X,/1X,'MAXIMUM HEAD CHANGE FOR EACH ITERATION:',/
     1    1X,/1X,3('  HEAD CHANGE  LAY,ROW,COL'),/1X,79('-'))
      WRITE (IOUT,10) (HDCG(J),(LRCH(I,J),I=1,3),J=1,NUMIT)
   10 FORMAT((2X,3(2X,1PG10.3,' (',I3,',',I3,',',I3,')')))
      WRITE(IOUT,11)
   11 FORMAT(1H0)
C
      RETURN
      END
```

List of Variables for Module SDE45P

Variable	Range	Definition
HDCG	Package	DIMENSION (MXITER), Maximum head change for each iteration.
I	Module	Index for the three grid coordinates showing where the maximum head change occurs.
IOUT	Global	Primary unit number for all printed output.
J	Module	Index for iterations.
LRCH	Package	DIMENSION (3,MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.
NUMIT	Module	The number of iterations for which the maximum head change is printed.

Module SDE45S

Narrative for Module SDE45S

Module SDE45S performs Gaussian elimination of the [A] matrix, modification of {b}, and back substitution to solve for head change. D4 ordering and symmetry are assumed. The coefficients of [A] are stored in two arrays, AU and AL, which correspond to the upper and lower parts of [A] as defined for D4 ordering (Price and Coats, 1974). Prior to application of Gaussian elimination, AU contains a diagonal coefficient for each upper equation plus up to six off-diagonal coefficients. IUPPNT contains the number of off-diagonal coefficients and the equation numbers of the off-diagonal coefficients that are stored in AU. AL contains a diagonal coefficient for each lower equation plus space for all the off-diagonal coefficients that will be created when the original left-side coefficients of the lower part of [A] are eliminated using the upper part of [A]. Once [AL] is created, Gaussian elimination is applied to make it upper triangular. The elimination of [A] is separated from the corresponding modification of {b} so that the elimination can be skipped if [A] has not changed since the previous solution. This saves a major part of the solution process.

1. Define variables for frequently used constants. ZERO and ONE facilitate conversion to double precision should that be desired. By declaring all real variables double precision, ZERO and ONE will automatically convert to double precision.

2. Skip to step 3 if Gaussian elimination of [A] is not required.
   A. Eliminate the left side of the lower part of [A], and fill [AL] with the resulting coefficients.
   B. Eliminate [AL] so that it is in upper triangular form.

3. Modify {b} to correspond to the eliminated [A].
   A. Modify {b} due to the elimination step 2A, which filled [AL].
   B. Modify {b} due to the elimination of [AL], which was step 2B.

4. Back substitute to solve for the head change for the lower part of the equations.

5. Back substitute to solve for the head change for the upper part of the equations.

6. Return.

Listing for Module SDE45S

```
      SUBROUTINE SDE45S(AU,AL,IUPPNT,B,NUP,NLOW,NEQ,MXBW,NBW,ITYPE,
     1               ID4DIM)
C
C-----VERSION 29SEPT1994 SDE45S
C     ****************************************************************
C     SOLVE EQUATIONS USING GAUSS ELIMINATION ASSUMING D4 ORDERING
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      DIMENSION AU(ID4DIM,NUP),AL(MXBW,NLOW),IUPPNT(ID4DIM,NUP),B(NEQ)
C     ----------------------------------------------------------------
C
C1------DEFINE CONSTANTS.
      ONE=1.
      ZERO=0.
      NLOWM1=NLOW-1
C
C2------DON'T ELIMINATE UNLESS NECESSARY.
      IF (ITYPE.EQ.1) GO TO 80
C
C2A-----MUST ELIMINATE -- DO THIS IN TWO PARTS.
C2A-----ELIMINATE THE LEFT SIDE OF THE LOWER PART OF [A] TO FILL [AL].
      DO 40 I=1,NUP
      JJ=IUPPNT(1,I)
      C1=ONE/AU(1,I)
      DO 30 J=2,JJ
      LR=IUPPNT(J,I)
      L=LR-NUP
      C=AU(J,I)*C1
      DO 20 K=J,JJ
      KL=IUPPNT(K,I)-LR+1
      AL(KL,L)=AL(KL,L)-C*AU(K,I)
   20 CONTINUE
      AU(J,I)=C
   30 CONTINUE
   40 CONTINUE
C
C2B-----ELIMINATE [AL].
      DO 70 I=1,NLOWM1
      L=I
      C1=ONE/AL(1,I)
      DO 60 J=2,NBW
      L=L+1
      IF (AL(J,I).EQ.ZERO) GO TO 60
      C=AL(J,I)*C1
      KL=0
      DO 50 K=J,NBW
      KL=KL+1
      IF (AL(K,I).NE.ZERO) AL(KL,L)=AL(KL,L)-C*AL(K,I)
   50 CONTINUE
      AL(J,I)=C
   60 CONTINUE
   70 CONTINUE
C
C3------B MUST ALWAYS BE MODIFIED -- MODIFY B IN TWO PARTS.
C3A-----MODIFY B DUE TO ELIMINATION TO FILL [AL].
   80 DO 100 I=1,NUP
      JJ=IUPPNT(1,I)
      DO 90 J=2,JJ
      LR=IUPPNT(J,I)
      B(LR)=B(LR)-AU(J,I)*B(I)
   90 CONTINUE
      B(I)=B(I)/AU(1,I)
  100 CONTINUE
C
C3B-----MODIFY B DUE TO ELIMINATION OF [AL].
      DO 120 I=1,NLOWM1
      IR=I+NUP
```

```
      LR=IR
      DO 110 J=2,NBW
      LR=LR+1
      IF (AL(J,I).NE.ZERO) B(LR)=B(LR)-AL(J,I)*B(IR)
  110 CONTINUE
      B(IR)=B(IR)/AL(1,I)
  120 CONTINUE
C
C4------BACK SUBSTITUTE LOWER PART.
      B(NEQ)=B(NEQ)/AL(1,NEQ-NUP)
      DO 140 I=1,NLOWM1
      K=NEQ-I
      KL=K-NUP
      L=K
      DO 130 J=2,NBW
      L=L+1
      IF (AL(J,KL).NE.ZERO) B(K)=B(K)-AL(J,KL)*B(L)
  130 CONTINUE
  140 CONTINUE
C
C5------BACK SUBSTITUTE UPPER PART.
      DO 160 I=1,NUP
      K=NUP+1-I
      JJ=IUPPNT(1,K)
      DO 150 J=2,JJ
      L=IUPPNT(J,K)
      B(K)=B(K)-AU(J,K)*B(L)
  150 CONTINUE
  160 CONTINUE
C
C6------RETURN.
      RETURN
      END
```

List of Variables for Module SDE45S

Variable	Range	Definition
AL	Package	DIMENSION (MXBW,NLOW), the matrix [AL].
AU	Package	DIMENSION (ID4DIM,NUP), the upper part of [A].
B	Package	DIMENSION (NEQ), the vector {b}.
C	Module	Temporary storage for terms used in elimination.
C1	Module	Temporary storage for terms used in elimination.
I	Module	Loop index.
ID4DIM	Package	A parameter used to dimension arrays AU and IUPPNT: 5 -- for a 2-dimensional grid. 7 -- for a 3-dimensional grid.
IR	Module	I+NUP.
ITYPE	Package	Flag that indicates if [A] requires elimination; 0 indicates elimination is required, and 1 indicates elimination is not required.
IUPPNT	Package	DIMENSION (ID4DIM,NUP), Contains the number of off-diagonal coefficients and the equation numbers of the off-diagonal coefficients that are stored in AU
J	Module	Loop index.
JJ	Module	Maximum index for some loops where J is the loop index.
K	Module	Loop index.
KL	Module	First subscript in [AL] in some situations.
L	Module	Second subscript for [AL] in some situations.
LR	Module	Subscript for B in some situations.
MXBW	Package	The maximum allowed value for the band width plus one.
NBW	Package	The bandwidth plus one of [AL].
NEQ	Package	The total number of equations to be solved.
NLOW	Package	The number of equations in the lower part of [A].
NLOWM1	Module	NLOW-1
NUP	Package	The number of equations in the upper part of [A].
ONE	Module	The constant 1.
ZERO	Module	The constant 0.

*U.S. G.P.O.:1995-387-030:00093

REFERENCES CITED

Dahlquist, Germund and Bjorck, Ake (translated by Anderson, Ned), 1974, Numerical methods: Englewood Cliffs, New Jersey, Prentice-Hall, 573 p.

Hill, M. C., 1990, Preconditioned conjugate-gradient 2 (PCG2), a computer program for solving ground-water flow equations: U.S. Geological Survey Water-Resources Investigations Report 90-4048, 43 p.

Kipp, K. L. ,1987, HST3D: a computer code for simulation of heat and solute transport in three-dimensional ground-water flow systems, U.S. Geological Survey Water-Resources Investigations Report 86-4095, 517 p.

Larson, S. P., 1978, Direct solution algorithm for the two-dimensional ground-water flow model: U.S. Geological Survey Open-File Report 79-202, 30 p.

McDonald, M. G. and Harbaugh, A. W., 1988, A modular three-dimensional finite-difference ground-water flow model: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 6, Chapter A1, 586 p.

Price, H. S. and Coats, K. H., 1974, Direct methods in reservoir simulation: Society of Petroleum Engineers Journal, June 1974, p. 295-308.

Remson, Irwin, Hornberger, G. M., and Molz, F. J., 1971, Numerical methods in subsurface hydrology: New York, Wiley-Interscience, 389 p.

APPENDIX

This appendix describes the sample problems for which the execution times are shown in table 1. The problems are hypothetical. In all problems, a rectangular aquifer 200 ft thick, 8000 ft wide, and 12000 ft long is simulated. Horizontal hydraulic conductivity is 100 ft/d, and vertical hydraulic conductivity is 1 ft/d. For transient problems, the specific yield is 0.2, and the confined storage coefficient is 0.0.

For problems A-D, the grid is 2 layers, 20 rows, and 30 columns. Each cell is 400 ft square horizontally. For the linear problems, each layer is 100 ft thick. For the non-linear problems, layer 2 is 100 ft thick, and layer 1 is unconfined; the bottom elevation of the unconfined layer is -100.0 ft. For problem E, the grid is 4 layers, 40 rows, and 60 columns. Each cell is 200 ft square horizontally, and each layer is 50 ft thick.

For all problems there is a no-flow boundary on all sides. In addition for problems A-D, a constant-head boundary runs along the entire length of column one in layer 1; the head is 0.0 ft. In problem E, there is a constant-head boundary along columns one and two in layers one and two, and the head is 0.0 ft.

Recharge is applied uniformly to the top layer in all problems at a rate of .0054 ft/d. Ten wells are each pumping 100,000 ft^3/d. The locations of wells are shown below.

Problems A-D			Problem E		
Layer	Row	Column	Layer	Row	Column
1	13	13	2	26	26
1	8	22	2	16	44
2	5	25	4	10	50
2	9	15	4	18	30
2	15	17	4	30	34
2	7	12	4	14	24
2	12	9	4	24	18
1	10	24	2	20	48
1	15	5	2	30	10
1	5	20	2	10	40

The transient problems, C and D, consist of a single stress period of 1000 days, and ten equal-length time steps are used.

Initial head in all problems is 0.0 ft at all cells.

When the D4 solver is used, the head closure criterion is .01 ft. The iteration schemes used for each problem are described in the discussion after table 1.

For the PCG solver, the head closure criterion is .001 ft and the residual closure criterion is 1000 ft^3/d. For linear problems, a single external iteration is used with multiple inner iterations. For non-linear problems, 10 inner iterations are allowed for each external iteration.

*U.S. G.P.O.:1995-387-030:93