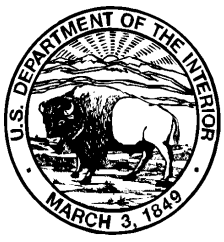
Programmer's Documentation for MODFLOW-96, an update to the U.S. Geological Survey Modular Finite-Difference Ground-Water Flow Model

---

U.S. GEOLOGICAL SURVEY

Open-File Report 96-486

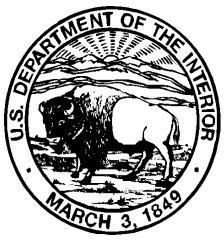


Programmer's Documentation for MODFLOW-96, an update to the U.S. Geological Survey Modular Finite-Difference Ground-Water Flow Model

by Arlen W. Harbaugh and Michael G. McDonald

U.S. GEOLOGICAL SURVEY

Open-File Report 96-486



Reston, Virginia
1996

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 20192

Copies of this report can be
purchased from:

U.S. Geological Survey
Branch of Information Services
Box 25286
Denver, CO 80225-0286
(303) 202-4700

PREFACE

This report presents a revised version of 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.

The computer program is available through the World-Wide Web at address

http://h2o.usgs.gov/software/

or by anonymous ftp file transfer from directory /pub/software/ground_water/modflow at Internet address

h2o.usgs.gov

The computer program is also available on diskette for the cost of processing from:

U.S. Geological Survey
NWIS Program Office
437 National Center
Reston, VA 20192
Telephone: (703) 648-5695

CONTENTS

Page

Abstract 1
Introduction 1
MAIN program 2
Basic Package 7
Module BAS5DF 7
Module BAS5AL 11
Module BAS5RP 13
Module BAS5ST 18
Module BAS5AD 19
Module BAS5FM 20
Module BAS5OC 21
Module BAS5OT 25
Module SBAS5D 26
Module SBAS5H 31
Module SBAS5I 35
Module SBAS5J 41
Module SBAS5T 45
Module SBAS5V 46
Module SBAS5N 51
Module SBAS5L 55
Module SBAS5O 58
Block-Centered Flow Package 62
Module BCF5AL 63
Module BCF5RP 65
Module BCF5AD 67
Module BCF5FM 68
Module SBCF5N 70
Module SBCF5H 72
Module SBCF5C 75
Module SBCF5A 76
Module SBCF5L 77
Module SBCF5U 78
Module SBCF5B 79
Module SBCF5F 86
Module SBCF5S 93
River Package 98
Module RIV5AL 98
Module RIV5RP 101
Module RIV5FM 103
Module RIV5BD 104

CONTENTS (Cont.)

Page

Recharge Package 110
Module RCH5AL 110
Module RCH5RP 111
Module RCH5FM 112
Module RCH5BD 113
Well Package 119
Module WEL5AL 119
Module WEL5RP 122
Module WEL5FM 124
Module WEL5BD 125
Drain Package 131
Module DRN5AL 131
Module DRN5RP 134
Module DRN5FM 136
Module DRN5BD 137
Evapotranspiration Package 143
Module EVT5AL 143
Module EVT5RP 144
Module EVT5FM 146
Module EVT5BD 147
General-Head Boundary Package 153
Module GHB5AL 153
Module GHB5RP 156
Module GHB5FM 158
Module GHB5BD 159
Strongly Implicit Procedure Package 165
Module SIP5AL 165
Module SIP5RP 166
Module SIP5AP 167
Module SSIP5P 171
Module SSIP5I 172
Slice-Successive Overrelaxation Package 174
Module SOR5AL 174
Module SOR5RP 175
Module SOR5AP 176
Module SSOR5B 179
Utility Modules 181
Module UBUDSV 181
Module ULASAV 182
Module ULAPRS 183

Page

Module ULAPRW 185
Module UCOLNO 187
Module U2DREL 188
Module U2DINT 194
Module U1DREL 200
Module URWORD 204
Module UBDSV1 209
Module UBDSV2 211
Module UBDSVA 213
Module UBDSV3 215
Module ULASV2 218
References 220

PROGRAMMER'S DOCUMENTATION FOR MODFLOW-96, AN UPDATE TO THE U.S. GEOLOGICAL SURVEY MODULAR FINITE-DIFFERENCE GROUND-WATER FLOW MODEL (MODFLOW)

By Arlen W. Harbaugh[1] and Michael G. McDonald[2]

ABSTRACT

A number of changes have been made to the U.S. Geological Survey modular finite-difference ground-water flow model, which is commonly known as MODFLOW. This report provides programmer's documentation for the revised code, which is called MODFLOW-96 to distinguish it from other versions of MODFLOW. A complete listing of the MODFLOW-96 code is provided. The listings are organized by package and by module within a package. For each package the major changes to the code are described. The changes to many modules are nonstructural, and for these modules, there is no module documentation provided in this report other than the comments in the code. The original module documentation is still generally valid for these modules. For new modules and those modules that have structural changes, a narrative and list of variables are provided along with the code listing. User's documentation is provided in a separate report.

INTRODUCTION

Revisions to the U.S. Geological Survey modular ground-water flow model, which is called MODFLOW, (McDonald and Harbaugh, 1988) are described in Harbaugh and McDonald (1996). To distinguish the two versions of the model, the 1988 version is called here MODFLOW-88, and the 1996 revised version is called MODFLOW-96. This report provides programmer's documentation for MODFOW-96.

The complete MODFLOW-96 code is listed below. The listings are organized by package and by module within a package. For each package the major changes to the code are described. The changes to many modules are nonstructural, and for these modules, there is no module documentation provided in this report other than the comments in the code. The module documentation in McDonald and Harbaugh (1988) is still generally valid for these modules. Examples of nonstructural changes are conversion of READ statements to optionally use free-format input, conversion of formats to limit the width of output in the listing file to 80 columns, increasing the maximum number of model layers to 200, and the revision of declarations of character variables. For new modules and those modules that have structural changes, a narrative and list of variables are provided along with the code listing.

---

[1]U.S. Geological Survey, Reston, VA
[2]McDonald Morrissey Associates, Reston, VA

Each list of variables for a module defines the variable range much as originally defined in McDonald and Harbaugh (1988, p. 1-7). "Module" variables are used within a single module. "Package" variables are used in more than one module of a package, but not outside of a package. "Global" variables are variables that are used by more than one package, or have the potential to be used by more than one package. This definition of "global" data represents a subtle change compared to MODFLOW-88. In MODFLOW-88, global meant that a variable was used in more than one package. In MODFLOW-96, a variable does not have to actually be used in multiple packages in order to be declared global. The new definition makes it possible to define a varible as global based on its potential for being used in multiple packages rather than basing the decision only on actual use. Thus, a few variables that are used by only one package in the basic model are designated as global in MODFLOW-96. An example is HEADNG, the simulation title. Although the BAS Package is the only package in the basic model that uses HEADNG, it is possible that additional future packages might wish to use HEADNG.

Each module in MODFLOW has a 6-character name. One of the characters is the version number. The version number for all modules in MODFLOW-88 is 1; however, many users have modified the code. If a module is modified, the version number should be changed. To distinguish MODFLOW-96 from MODFLOW-88 and its past derivatives, all modules in MODFLOW-96 have been given a version number of 5. This version number is likely to be higher than the version number of existing modules although there is no way to know this for sure.

MAIN PROGRAM

The MAIN program has been changed to incorporate modifications of the calling arguments for the various modules. The unit number for the name file (99) is defined rather than defining unit numbers for the BAS Package input file and the listing file. A prompt for the name file has been added. Also, the capability to obtain the names of one or more name files from the file modflow.bf was added. If modflow.bf exists, there are no user prompts.

A new array, CUNIT, has been added to the MAIN program. CUNIT is a character array that is used by module SBAS5O to associate the primary options being used in a simulation with elements in the IUNIT array. In MODFLOW-88, a user had to know which IUNIT element corresponded to a particular option because the IUNIT elements were directly read from the Basic Package file. In MODFLOW-96, the user does not need to know the IUNIT element. Instead, the user must know a one to four character option identifier, called the file type in the input instructions for the name file, in order to activate an option. These option identifiers are defined in the CUNIT array through a DATA statement in the MAIN program. Elements in CUNIT and IUNIT directly correspond. For example, if a particular option identifier is defined in the seventh element of the CUNIT array, then the seventh element of the IUNIT array also corresponds to that option. Module SBAS5O will put the file unit for that option, which the user specifies in the name file, into IUNIT(7). Thus, to add a new option (generally a package) to the MAIN program, one needs to find an unused element in CUNIT (unused values are blank strings), and modify the CUNIT data statement to set that CUNIT element equal to the identifier for the new option. Then statements should be added in the appropriate places in the MAIN program to call the option subroutines whenever the IUNIT value for the new option is greater than 0. If an option is not being used, its IUNIT value will be 0.

```
C     ******************************************************************
C     MAIN CODE FOR U.S. GEOLOGICAL SURVEY MODULAR MODEL -- MODFLOW-96
C           BY MICHAEL G. MCDONALD AND ARLEN W. HARBAUGH
C     MODFLOW-88 documented in:
C           McDonald, M.G. and Harbaugh, A.W., 1988, A modular
C           three-dimensional finite-difference ground-water flow
C           model: U.S. Geological Survey Techniques of Water
C           Resources Investigations, Book 6, Chapter A1, 586 p.
C     MODFLOW-96 documented in:
C           Harbaugh, A.W. and McDonald, M.G., 1996, User's
C           documentation for the U.S. Geological Survey modular
C           finite-difference ground-water flow model: U.S. Geological
C           Survey Open-File Report 96-485
C-----VERSION 0950 23MAY1996 MAIN
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C1------SPECIFY THE SIZE OF THE X ARRAY.  TO CHANGE THE SIZE OF THE
C1------X ARRAY, CHANGE VALUE OF LENX IN THE NEXT STATEMENT.
      PARAMETER (LENX=1500000)
      COMMON X(LENX)
      COMMON /FLWCOM/LAYCON(200)
      CHARACTER*16 VBNM(40)
      CHARACTER*80 HEADNG(2)
      DIMENSION VBVL(4,40),IUNIT(40)
      DOUBLE PRECISION DUMMY
      EQUIVALENCE (DUMMY,X(1))
      CHARACTER*20 CHEDFM,CDDNFM
      CHARACTER*80 FNAME
      LOGICAL EXISTS
      CHARACTER*4 CUNIT(40)
      DATA CUNIT/'BCF ','WEL ','DRN ','RIV ','EVT ','    ','GHB ',
     1           'RCH ','SIP ','    ','SOR ','OC  ','    ','    ',
     2           '    ','    ','    ','    ','    ','    ','    ',
     3           '    ','    ','    ','    ','    ','    ','    ',
     4           '    ','    ','    ','    ','    ','    ','    ',
     5           '    ','    ','    ','    ','    ','    ','    '/
C     ------------------------------------------------------------------
      INUNIT=99
      IBUNIT=98
      IBOUTS=97
      IBATCH=0
      INQUIRE(FILE='modflow.bf',EXIST=EXISTS)
      IF(EXISTS) THEN
         IBATCH=1
         OPEN(UNIT=IBUNIT,FILE='modflow.bf',STATUS='OLD')
         OPEN(UNIT=IBOUTS,FILE='modbatch.rpt')
         WRITE(IBOUTS,*) ' USGS MODFLOW MODEL BATCH-MODE REPORT'
      END IF
C
C2------OPEN FILE OF FILE NAMES.
50    IF(IBATCH.GT.0) THEN
         READ(IBUNIT,'(A)',END=500) FNAME
         IF(FNAME.EQ.' ') GO TO 50
         WRITE(IBOUTS,'(1X,/1X,A)') FNAME
      ELSE
         WRITE(*,*) ' Enter the name of the NAME FILE:'
         READ(*,'(A)') FNAME
      END IF
      INQUIRE(FILE=FNAME,EXIST=EXISTS)
      IF(.NOT.EXISTS) THEN
         IF(IBATCH.GT.0) THEN
            WRITE(IBOUTS,*) ' Specified name file does not exist.'
            WRITE(IBOUTS,*) ' Processing will continue with the next ',
     1                      'name file in modflow.bf.'
         ELSE
            WRITE(*,*) ' File does not exist'
         END IF
         GO TO 50
      END IF
      OPEN(UNIT=INUNIT,FILE=FNAME,STATUS='OLD')
C
C3------DEFINE PROBLEM--ROWS,COLUMNS,LAYERS,STRESS PERIODS,PACKAGES.
      CALL BAS5DF(ISUM,HEADNG,NPER,ITMUNI,TOTIM,NCOL,NROW,NLAY,
     1        NODES,INBAS,IOUT,IUNIT,CUNIT,INUNIT,IXSEC,ICHFLG,IFREFM)
C
C4------ALLOCATE SPACE IN "X" ARRAY.
      CALL BAS5AL(ISUM,LENX,LCHNEW,LCHOLD,LCIBOU,LCCR,LCCC,LCCV,
     1              LCHCOF,LCRHS,LCDELR,LCDELC,LCSTRT,LCBUFF,LCIOFL,
```

```
     2              INBAS,ISTRT,NCOL,NROW,NLAY,IOUT,IAPART,IFREFM)
      IF(IUNIT(1).GT.0) CALL BCF5AL(ISUM,LENX,LCSC1,LCHY,
     1     LCBOT,LCTOP,LCSC2,LCTRPY,IUNIT(1),ISS,
     2     NCOL,NROW,NLAY,IOUT,IBCFCB,LCWETD,IWDFLG,LCCVWD,
     3     WETFCT,IWETIT,IHDWET,HDRY,IAPART,IFREFM)
      IF(IUNIT(2).GT.0) CALL WEL5AL(ISUM,LENX,LCWELL,MXWELL,NWELLS,
     1                 IUNIT(2),IOUT,IWELCB,NWELVL,IWELAL,IFREFM)
      IF(IUNIT(3).GT.0) CALL DRN5AL(ISUM,LENX,LCDRAI,NDRAIN,MXDRN,
     1                 IUNIT(3),IOUT,IDRNCB,NDRNVL,IDRNAL,IFREFM)
      IF(IUNIT(4).GT.0) CALL RIV5AL(ISUM,LENX,LCRIVR,MXRIVR,NRIVER,
     1            IUNIT(4),IOUT,IRIVCB,NRIVVL,IRIVAL,IFREFM)
      IF(IUNIT(5).GT.0) CALL EVT5AL(ISUM,LENX,LCIEVT,LCEVTR,LCEXDP,
     1            LCSURF,NCOL,NROW,NEVTOP,IUNIT(5),IOUT,IEVTCB,IFREFM)
      IF(IUNIT(7).GT.0) CALL GHB5AL(ISUM,LENX,LCBNDS,NBOUND,MXBND,
     1            IUNIT(7),IOUT,IGHBCB,NGHBVL,IGHBAL,IFREFM)
      IF(IUNIT(8).GT.0) CALL RCH5AL(ISUM,LENX,LCIRCH,LCRECH,NRCHOP,
     1            NCOL,NROW,IUNIT(8),IOUT,IRCHCB,IFREFM)
      IF(IUNIT(9).GT.0) CALL SIP5AL(ISUM,LENX,LCEL,LCFL,LCGL,LCV,
     1          LCHDCG,LCLRCH,LCW,MXITER,NPARM,NCOL,NROW,NLAY,
     2          IUNIT(9),IOUT,IFREFM)
      IF(IUNIT(11).GT.0) CALL SOR5AL(ISUM,LENX,LCA,LCRES,LCHDCG,LCLRCH,
     1       LCIEQP,MXITER,NCOL,NLAY,NSLICE,MBW,IUNIT(11),IOUT,IFREFM)
C
C5------IF THE "X" ARRAY IS NOT BIG ENOUGH THEN STOP.
      IF(ISUM-1.GT.LENX) STOP
C
C6------READ AND PREPARE INFORMATION FOR ENTIRE SIMULATION.
      CALL BAS5RP(X(LCIBOU),X(LCHNEW),X(LCSTRT),X(LCHOLD),
     1       ISTRT,INBAS,HEADNG,NCOL,NROW,NLAY,VBVL,X(LCIOFL),
     2       IUNIT(12),IHEDFM,IDDNFM,IHEDUN,IDDNUN,IOUT,IPEROC,ITSOC,
     3       CHEDFM,CDDNFM,IBDOPT,IXSEC,LBHDSV,LBDDSV,IFREFM)
      IF(IUNIT(1).GT.0) CALL BCF5RP(X(LCIBOU),X(LCHNEW),X(LCSC1),
     1          X(LCHY),X(LCCR),X(LCCC),X(LCCV),X(LCDELR),
     2     X(LCDELC),X(LCBOT),X(LCTOP),X(LCSC2),X(LCTRPY),IUNIT(1),
     3     ISS,NCOL,NROW,NLAY,IOUT,X(LCWETD),IWDFLG,X(LCCVWD))
      IF(IUNIT(9).GT.0) CALL SIP5RP(NPARM,MXITER,ACCL,HCLOSE,X(LCW),
     1          IUNIT(9),IPCALC,IPRSIP,IOUT,IFREFM)
      IF(IUNIT(11).GT.0) CALL SOR5RP(MXITER,ACCL,HCLOSE,IUNIT(11),
     1         IPRSOR,IOUT,IFREFM)
C
C7------SIMULATE EACH STRESS PERIOD.
      DO 300 KPER=1,NPER
      KKPER=KPER
C
C7A-----READ STRESS PERIOD TIMING INFORMATION.
      CALL BAS5ST(NSTP,DELT,TSMULT,PERTIM,KKPER,INBAS,IOUT,IFREFM)
C
C7B-----READ AND PREPARE INFORMATION FOR STRESS PERIOD.
      IF(IUNIT(2).GT.0) CALL WEL5RP(X(LCWELL),NWELLS,MXWELL,IUNIT(2),
     1           IOUT,NWELVL,IWELAL,IFREFM)
      IF(IUNIT(3).GT.0) CALL DRN5RP(X(LCDRAI),NDRAIN,MXDRN,IUNIT(3),
     1                 IOUT,NDRNVL,IDRNAL,IFREFM)
      IF(IUNIT(4).GT.0) CALL RIV5RP(X(LCRIVR),NRIVER,MXRIVR,IUNIT(4),
     1            IOUT,NRIVVL,IRIVAL,IFREFM)
      IF(IUNIT(5).GT.0) CALL EVT5RP(NEVTOP,X(LCIEVT),X(LCEVTR),
     1            X(LCEXDP),X(LCSURF),X(LCDELR),X(LCDELC),NCOL,NROW,
     1            IUNIT(5),IOUT,IFREFM)
      IF(IUNIT(7).GT.0) CALL GHB5RP(X(LCBNDS),NBOUND,MXBND,IUNIT(7),
     1              IOUT,NGHBVL,IGHBAL,IFREFM)
      IF(IUNIT(8).GT.0) CALL RCH5RP(NRCHOP,X(LCIRCH),X(LCRECH),
     1            X(LCDELR),X(LCDELC),NROW,NCOL,IUNIT(8),IOUT,IFREFM)
C
C7C-----SIMULATE EACH TIME STEP.
      DO 200 KSTP=1,NSTP
      KKSTP=KSTP
C
C7C1----CALCULATE TIME STEP LENGTH. SET HOLD=HNEW..
      CALL BAS5AD(DELT,TSMULT,TOTIM,PERTIM,X(LCHNEW),X(LCHOLD),KKSTP,
     1            NCOL,NROW,NLAY)
      IF(IUNIT(1).GT.0) CALL BCF5AD(X(LCIBOU),X(LCHOLD),X(LCBOT),
     1             X(LCWETD),IWDFLG,ISS,NCOL,NROW,NLAY)
C
C7C2----ITERATIVELY FORMULATE AND SOLVE THE EQUATIONS.
      DO 100 KITER=1,MXITER
      KKITER=KITER
C
C7C2A---FORMULATE THE FINITE DIFFERENCE EQUATIONS.
      CALL BAS5FM(X(LCHCOF),X(LCRHS),NODES)
      IF(IUNIT(1).GT.0) CALL BCF5FM(X(LCHCOF),X(LCRHS),X(LCHOLD),
     1          X(LCSC1),X(LCHNEW),X(LCIBOU),X(LCCR),X(LCCC),X(LCCV),
```

```
     2          X(LCHY),X(LCTRPY),X(LCBOT),X(LCTOP),X(LCSC2),
     3          X(LCDELR),X(LCDELC),DELT,ISS,KKITER,KKSTP,KKPER,NCOL,
     4          NROW,NLAY,IOUT,X(LCWETD),IWDFLG,X(LCCVWD),WETFCT,
     5          IWETIT,IHDWET,HDRY,X(LCBUFF))
      IF(IUNIT(2).GT.0) CALL WEL5FM(NWELLS,MXWELL,X(LCRHS),X(LCWELL),
     1           X(LCIBOU),NCOL,NROW,NLAY,NWELVL)
      IF(IUNIT(3).GT.0) CALL DRN5FM(NDRAIN,MXDRN,X(LCDRAI),X(LCHNEW),
     1         X(LCHCOF),X(LCRHS),X(LCIBOU),NCOL,NROW,NLAY,NDRNVL)
      IF(IUNIT(4).GT.0) CALL RIV5FM(NRIVER,MXRIVR,X(LCRIVR),X(LCHNEW),
     1           X(LCHCOF),X(LCRHS),X(LCIBOU),NCOL,NROW,NLAY,NRIVVL)
      IF(IUNIT(5).GT.0) CALL EVT5FM(NEVTOP,X(LCIEVT),X(LCEVTR),
     1           X(LCEXDP),X(LCSURF),X(LCRHS),X(LCHCOF),X(LCIBOU),
     1           X(LCHNEW),NCOL,NROW,NLAY)
      IF(IUNIT(7).GT.0) CALL GHB5FM(NBOUND,MXBND,X(LCBNDS),X(LCHCOF),
     1           X(LCRHS),X(LCIBOU),NCOL,NROW,NLAY,NGHBVL)
      IF(IUNIT(8).GT.0) CALL RCH5FM(NRCHOP,X(LCIRCH),X(LCRECH),
     1           X(LCRHS),X(LCIBOU),NCOL,NROW,NLAY)
C
C7C2B---MAKE ONE CUT AT AN APPROXIMATE SOLUTION.
      IF(IUNIT(9).GT.0) CALL SIP5AP(X(LCHNEW),X(LCIBOU),X(LCCR),X(LCCC),
     1     X(LCCV),X(LCHCOF),X(LCRHS),X(LCEL),X(LCFL),X(LCGL),X(LCV),
     2     X(LCW),X(LCHDCG),X(LCLRCH),NPARM,KKITER,HCLOSE,ACCL,ICNVG,
     3     KKSTP,KKPER,IPCALC,IPRSIP,MXITER,NSTP,NCOL,NROW,NLAY,NODES,
     4     IOUT)
      IF(IUNIT(11).GT.0) CALL SOR5AP(X(LCHNEW),X(LCIBOU),X(LCCR),
     1     X(LCCC),X(LCCV),X(LCHCOF),X(LCRHS),X(LCA),X(LCRES),X(LCIEQP),
     2     X(LCHDCG),X(LCLRCH),KKITER,HCLOSE,ACCL,ICNVG,KKSTP,KKPER,
     3     IPRSOR,MXITER,NSTP,NCOL,NROW,NLAY,NSLICE,MBW,IOUT)
C
C7C2C---IF CONVERGENCE CRITERION HAS BEEN MET STOP ITERATING.
      IF(ICNVG.EQ.1) GO TO 110
  100 CONTINUE
      KITER=MXITER
  110 CONTINUE
C
C7C3----DETERMINE WHICH OUTPUT IS NEEDED.
      CALL BAS5OC(NSTP,KKSTP,ICNVG,X(LCIOFL),NLAY,IBUDFL,ICBCFL,
     1   IHDDFL,IUNIT(12),IOUT,KKPER,IPEROC,ITSOC,IBDOPT,IXSEC,IFREFM)
C
C7C4----CALCULATE BUDGET TERMS. SAVE CELL-BY-CELL FLOW TERMS.
      MSUM=1
C7C4A---THE ORIGINAL BCF BUDGET MODULE HAS BEEN REPLACED BY THREE
C7C4A---SUBMODULES: SBCF5S, SBCF5F, AND SBCF5B .
      IF(IUNIT(1).GT.0) THEN
         CALL SBCF5S(VBNM,VBVL,MSUM,X(LCHNEW),X(LCIBOU),X(LCHOLD),
     1     X(LCSC1),X(LCTOP),X(LCSC2),DELT,ISS,NCOL,NROW,NLAY,KKSTP,
     2     KKPER,IBCFCB,ICBCFL,X(LCBUFF),IOUT,PERTIM,TOTIM)
         CALL SBCF5F(VBNM,VBVL,MSUM,X(LCHNEW),X(LCIBOU),X(LCCR),
     1     X(LCCC),X(LCCV),X(LCTOP),DELT,NCOL,NROW,NLAY,KKSTP,KKPER,
     2     IBCFCB,X(LCBUFF),IOUT,ICBCFL,PERTIM,TOTIM,ICHFLG)
         IBDRET=0
         IC1=1
         IC2=NCOL
         IR1=1
         IR2=NROW
         IL1=1
         IL2=NLAY
         DO 155 IDIR=1,3
         CALL SBCF5B(X(LCHNEW),X(LCIBOU),X(LCCR),X(LCCC),X(LCCV),
     1      X(LCTOP),NCOL,NROW,NLAY,KKSTP,KKPER,IBCFCB,X(LCBUFF),
     2      IOUT,ICBCFL,DELT,PERTIM,TOTIM,IDIR,IBDRET,ICHFLG,
     3      IC1,IC2,IR1,IR2,IL1,IL2)
155      CONTINUE
      END IF
      IF(IUNIT(2).GT.0) CALL WEL5BD(NWELLS,MXWELL,VBNM,VBVL,MSUM,
     1     X(LCWELL),X(LCIBOU),DELT,NCOL,NROW,NLAY,KKSTP,KKPER,IWELCB,
     1     ICBCFL,X(LCBUFF),IOUT,PERTIM,TOTIM,NWELVL,IWELAL)
      IF(IUNIT(3).GT.0) CALL DRN5BD(NDRAIN,MXDRN,VBNM,VBVL,MSUM,
     1     X(LCDRAI),DELT,X(LCHNEW),NCOL,NROW,NLAY,X(LCIBOU),KKSTP,
     2     KKPER,IDRNCB,ICBCFL,X(LCBUFF),IOUT,PERTIM,TOTIM,NDRNVL,
     3     IDRNAL)
      IF(IUNIT(4).GT.0) CALL RIV5BD(NRIVER,MXRIVR,X(LCRIVR),X(LCIBOU),
     1     X(LCHNEW),NCOL,NROW,NLAY,DELT,VBVL,VBNM,MSUM,KKSTP,KKPER,
     2     IRIVCB,ICBCFL,X(LCBUFF),IOUT,PERTIM,TOTIM,NRIVVL,IRIVAL)
      IF(IUNIT(5).GT.0) CALL EVT5BD(NEVTOP,X(LCIEVT),X(LCEVTR),
     1     X(LCEXDP),X(LCSURF),X(LCIBOU),X(LCHNEW),NCOL,NROW,NLAY,
     2     DELT,VBVL,VBNM,MSUM,KKSTP,KKPER,IEVTCB,ICBCFL,X(LCBUFF),IOUT,
     3     PERTIM,TOTIM)
      IF(IUNIT(7).GT.0) CALL GHB5BD(NBOUND,MXBND,VBNM,VBVL,MSUM,
```

```
     1     X(LCBNDS),DELT,X(LCHNEW),NCOL,NROW,NLAY,X(LCIBOU),KKSTP,
     2     KKPER,IGHBCB,ICBCFL,X(LCBUFF),IOUT,PERTIM,TOTIM,NGHBVL,
     3     IGHBAL)
      IF(IUNIT(8).GT.0) CALL RCH5BD(NRCHOP,X(LCIRCH),X(LCRECH),
     1     X(LCIBOU),NROW,NCOL,NLAY,DELT,VBVL,VBNM,MSUM,KKSTP,KKPER,
     2     IRCHCB,ICBCFL,X(LCBUFF),IOUT,PERTIM,TOTIM)
C
C7C5---PRINT AND OR SAVE HEADS AND DRAWDOWNS. PRINT OVERALL BUDGET.
      CALL BAS5OT(X(LCHNEW),X(LCSTRT),ISTRT,X(LCBUFF),X(LCIOFL),
     1     MSUM,X(LCIBOU),VBNM,VBVL,KKSTP,KKPER,DELT,PERTIM,TOTIM,
     2     ITMUNI,NCOL,NROW,NLAY,ICNVG,IHDDFL,IBUDFL,IHEDFM,IHEDUN,
     3     IDDNFM,IDDNUN,IOUT,CHEDFM,CDDNFM,IXSEC,LBHDSV,LBDDSV)
C
C7C6----IF ITERATION FAILED TO CONVERGE THEN STOP.
      IF(ICNVG.EQ.0) STOP
  200 CONTINUE
  300 CONTINUE
C
C8------END OF SIMULATION
      IF(IBATCH.GT.0) THEN
         WRITE(IBOUTS,*) ' Normal termination of simulation.'
         DO 400 I=1,IBOUTS-1
            INQUIRE(UNIT=I,OPENED=EXISTS)
            IF(EXISTS) CLOSE(I)
  400    CONTINUE
         GO TO 50
      END IF
  500 STOP
C
      END
```

BASIC PACKAGE

Many of the changes in the Basic Package (BAS) are for the purpose of implementing the word method for controlling output. Three new submodules, SBAS5J, SBAS5N and SBAS5L, have been created, and some other modules have been modified, in order to implement the new method of output control. Other significant changes to the BAS Package are for the purpose of opening files and improving input and output for cross-sectional models. Submodule SBAS5O has been added to open files.

Module BAS5DF

Narrative for Module BAS5DF

The BAS5DF module defines and sets key model parameters. It does so in the following order:

0. Call module SBAS5O to open files and assign IUNIT values. IUNIT values determine which of the major options are active.

1. Print the name of the program.

2. Read and print a heading.

3. Read as text a line that contains the number of layers, rows, columns, stress periods, and code ITMUNI. Wait to decode this until it is known if free or fixed format is used.

4. Read the options record and call the URWORD module to parse the record for recognized options. This record was previously used to specify IUNIT values, but IUNIT values are now defined in step 0. Recognized options are "XSECTION" to indicate that a cross section is being simulated along a row, "CHTOCH" to indicate that flow between adjacent constant-head cells should be calculated, and "FREE" to indicate that free format should be used for reading data that are not read by the array reading utility modules. If unrecognized information is found, such as would occur if an old dataset with IUNIT values were being read, then the information is ignored.

5. Read the number of layers, rows, columns, stress periods, and code ITMUNI from the line that was previously read as text. Use fixed or free format as indicated by flag IFREFM. ITMUNI is a code that indicates the time units of model data. It does not affect model calculations of head or flow; it is used when printing the amount of elapsed time (see the input instructions for the codes).

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

7. Select and print a message showing the time units and other options.

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

9. RETURN.

BAS5DF

```
      SUBROUTINE BAS5DF(ISUM,HEADNG,NPER,ITMUNI,TOTIM,NCOL,NROW,
     1    NLAY,NODES,INBAS,IOUT,IUNIT,CUNIT,INUNIT,IXSEC,ICHFLG,IFREFM)
C
C-----VERSION 1030 20FEB1996 BAS5DF
C     ******************************************************************
C     DEFINE KEY MODEL PARAMETERS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*80 HEADNG(2)
      DIMENSION IUNIT(40)
      CHARACTER*4 CUNIT(40)
      CHARACTER*80 LINE1,LINE2
C     ------------------------------------------------------------------
C0------OPEN FILES AND ASSIGN IUNIT VALUES.
      CALL SBAS5O(INUNIT,INBAS,IOUT,IUNIT,CUNIT)
C
C1------PRINT THE NAME OF THE PROGRAM.
      WRITE(IOUT,1)
    1 FORMAT('1',33X,'MODFLOW',/6X,'U.S. GEOLOGICAL SURVEY MODULAR',
     1       ' FINITE-DIFFERENCE GROUND-WATER FLOW MODEL')
C
C2------READ AND PRINT A HEADING.
      READ(INBAS,'(A)') HEADNG(1)
      READ(INBAS,'(A)') HEADNG(2)
      WRITE(IOUT,'(1X,/1X,A)') HEADNG(1)
      WRITE(IOUT,'(1X,A)') HEADNG(2)
C
C3------READ LINE SPECIFYING NUMBER OF LAYERS,ROWS,COLUMNS,STRESS
C3------PERIODS AND UNITS OF TIME CODE, BUT DON'T DECODE UNTIL IT IS
C3------DETERMINED THAT FREE OR FIXED FORMAT IS BEING USED.
      READ(INBAS,'(A)') LINE1
C
C4------READ OPTIONS RECORD AND LOOK FOR OPTIONS
      READ(INBAS,'(A)') LINE2
      IXSEC=0
      ICHFLG=0
      IFREFM=0
      LLOC=1
    5 CALL URWORD(LINE2,LLOC,ISTART,ISTOP,1,N,R,IOUT,INBAS)
      IF(LINE2(ISTART:ISTOP).EQ.'XSECTION' .AND. NROW.EQ.1) THEN
         IXSEC=1
      ELSE IF(LINE2(ISTART:ISTOP).EQ.'CHTOCH') THEN
         ICHFLG=1
      ELSE IF(LINE2(ISTART:ISTOP).EQ.'FREE') THEN
         IFREFM=1
         WRITE(IOUT,6)
    6    FORMAT (1X,'THE FREE FORMAT OPTION HAS BEEN SELECTED')
      END IF
      IF(LLOC.LT.80) GO TO 5
C
C5------READ NUMBER OF LAYERS, ROWS, COLUMNS, STRESS PERIODS, AND
C5------ITMUNI USING FREE OR FIXED FORMAT.
      IF(IFREFM.EQ.0) THEN
         READ(LINE1,'(5I10)') NLAY,NROW,NCOL,NPER,ITMUNI
      ELSE
         LLOC=1
         CALL URWORD(LINE1,LLOC,ISTART,ISTOP,2,NLAY,R,IOUT,INBAS)
         CALL URWORD(LINE1,LLOC,ISTART,ISTOP,2,NROW,R,IOUT,INBAS)
         CALL URWORD(LINE1,LLOC,ISTART,ISTOP,2,NCOL,R,IOUT,INBAS)
         CALL URWORD(LINE1,LLOC,ISTART,ISTOP,2,NPER,R,IOUT,INBAS)
         CALL URWORD(LINE1,LLOC,ISTART,ISTOP,2,ITMUNI,R,IOUT,INBAS)
      END IF
C
C6------PRINT # OF LAYERS, ROWS, COLUMNS AND STRESS PERIODS.
      WRITE(IOUT,7) NLAY,NROW,NCOL
    7 FORMAT(1X,I4,' LAYERS',I10,' ROWS',I10,' COLUMNS')
      WRITE(IOUT,8) NPER
    8 FORMAT(1X,I3,' STRESS PERIOD(S) IN SIMULATION')
C
C7------SELECT AND PRINT A MESSAGE SHOWING TIME UNITS AND OTHER OPTIONS.
      IF(ITMUNI.LT.0 .OR. ITMUNI.GT.5) ITMUNI=0
      IF(ITMUNI.EQ.0) THEN
         WRITE(IOUT,9)
    9    FORMAT(1X,'MODEL TIME UNITS ARE UNDEFINED')
      ELSE IF(ITMUNI.EQ.1) THEN
```

```
         WRITE(IOUT,11)
   11    FORMAT(1X,'MODEL TIME UNIT IS SECONDS')
      ELSE IF(ITMUNI.EQ.2) THEN
         WRITE(IOUT,21)
   21    FORMAT(1X,'MODEL TIME UNIT IS MINUTES')
      ELSE IF(ITMUNI.EQ.3) THEN
         WRITE(IOUT,31)
   31    FORMAT(1X,'MODEL TIME UNIT IS HOURS')
      ELSE IF(ITMUNI.EQ.4) THEN
         WRITE(IOUT,41)
   41    FORMAT(1X,'MODEL TIME UNIT IS DAYS')
      ELSE
         WRITE(IOUT,51)
   51    FORMAT(1X,'MODEL TIME UNIT IS YEARS')
      END IF
      IF(IXSEC.NE.0) WRITE(IOUT,61)
   61 FORMAT(1X,'CROSS SECTION OPTION IS SPECIFIED')
      IF(ICHFLG.NE.0) WRITE(IOUT,62)
   62 FORMAT(1X,'CALCULATE FLOW BETWEEN ADJACENT CONSTANT-HEAD CELLS')
C
C8------INITIALIZE TOTAL ELAPSED TIME COUNTER STORAGE ARRAY COUNTER
C8------AND CALCULATE NUMBER OF CELLS.
      TOTIM=0.
      ISUM=1
      NODES=NCOL*NROW*NLAY
C
C9------RETURN
      RETURN
      END
```

List of Variables for Module BAS5DF

Variable	Range	Definition
CUNIT	Package	CHARACTER*4(40), Names of major options corresponding to elements within the IUNIT array.
HEADNG	Global	CHARACTER*80(2), Heading printed in output to identify problem.
ICHFLG	Global	Flag for flow between constant-head cells: = 0, flow between constant-head cells is not calculated. ≠ 0, flow between constant-head cells is calculated..
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
INBAS	Package	Primary input unit for the Basic (BAS) Package.
INUNIT	Package	Unit for reading from the name file.
IOUT	Global	Unit number for writing to the listing file.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ISUM	Global	Index of the lowest element in the X array which has not yet been allocated.
ITMUNI	Global	Code for time units being used: 0 - undefined 1 - seconds 2 - minutes 3 - hours 4 - days 5 - years
IUNIT	Package	DIMENSION(40), Primary input units for the major model options.
IXSEC	Global	Cross section flag: = 0, the model is not a 1-row cross section. ≠ 0, the model is a 1-row cross section.
LINE1	Module	CHARACTER*80, Third line of the BAS input file, from which NLAY, NROW, NCOL, NPER, and ITMUNI are read.
LINE2	Module	CHARACTER*80, Fourth line of the BAS input file, which is scanned for option words.
LLOC	Module	Location within LINE1 and LINE2 where URWORD starts looking for a word.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NODES	Global	The number of cells (nodes) in the grid.
NPER	Global	The number of stress periods in the simulation.
NROW	Global	The number of rows in the grid.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.
TOTIM	Global	Elapsed time in the simulation.

Module BAS5AL

```
      SUBROUTINE BAS5AL(ISUM,LENX,LCHNEW,LCHOLD,LCIBOU,LCCR,LCCC,LCCV,
     1            LCHCOF,LCRHS,LCDELR,LCDELC,LCSTRT,LCBUFF,LCIOFL,INBAS,
     2            ISTRT,NCOL,NROW,NLAY,IOUT,IAPART,IFREFM)
C-----VERSION 1334 20FEB1996 BAS5AL
C     ******************************************************************
C     ALLOCATE SPACE FOR BASIC MODEL ARRAYS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------PRINT A MESSAGE IDENTIFYING THE PACKAGE.
      WRITE(IOUT,1)INBAS
    1 FORMAT(1X,/1X,'BAS5 -- BASIC MODEL PACKAGE, VERSION 5, 1/1/95',
     2' INPUT READ FROM UNIT',I3)
C
C2------READ & PRINT FLAG IAPART (RHS & BUFFER SHARE SPACE?) AND
C2------FLAG ISTRT (SHOULD STARTING HEADS BE KEPT FOR DRAWDOWN?).
      IF(IFREFM.EQ.0) THEN
         READ(INBAS,'(2I10)') IAPART,ISTRT
      ELSE
         READ(INBAS,*) IAPART,ISTRT
      END IF
      IF(IAPART.NE.0) WRITE(IOUT,2)
    2 FORMAT(1X,
     1    'ARRAYS RHS AND BUFF WILL HAVE SEPARATE MEMORY ALLOCATIONS')
      IF(IAPART.EQ.0) WRITE(IOUT,3)
    3 FORMAT(1X,'ARRAYS RHS AND BUFF WILL SHARE MEMORY')
      IF(ISTRT.NE.0) WRITE(IOUT,4)
    4 FORMAT(1X,'INITIAL HEAD WILL BE KEPT THROUGHOUT THE SIMULATION')
      IF(ISTRT.EQ.0) WRITE(IOUT,5)
    5 FORMAT(1X,'INITIAL HEAD WILL NOT BE KEPT THROUGHOUT THE',
     1 ' SIMULATION, WHICH MEANS',/1X,'DRAWDOWN CANNOT BE CALCULATED')
C
C3------STORE LOCATION OF FIRST UNALLOCATED SPACE IN X.
      ISOLD=ISUM
      NRCL=NROW*NCOL*NLAY
C
C4------ALLOCATE SPACE FOR ARRAYS.
      LCHNEW=ISUM
      ISUM=ISUM+2*NRCL
      LCHOLD=ISUM
      ISUM=ISUM+NRCL
      LCIBOU=ISUM
      ISUM=ISUM+NRCL
      LCCR=ISUM
      ISUM=ISUM+NRCL
      LCCC=ISUM
      ISUM=ISUM+NRCL
      LCCV=ISUM
      ISUM=ISUM+NROW*NCOL*(NLAY-1)
      LCHCOF=ISUM
      ISUM=ISUM+NRCL
      LCRHS=ISUM
      ISUM=ISUM+NRCL
      LCDELR=ISUM
      ISUM=ISUM+NCOL
      LCDELC=ISUM
      ISUM=ISUM+NROW
      LCIOFL=ISUM
      ISUM=ISUM+NLAY*4
C
C5------IF BUFFER AND RHS SHARE SPACE THEN LCBUFF=LCRHS.
      LCBUFF=LCRHS
      IF(IAPART.EQ.0) GO TO 50
      LCBUFF=ISUM
      ISUM=ISUM+NRCL
C
C6------IF STRT WILL BE SAVED THEN ALLOCATE SPACE.
   50 LCSTRT=ISUM
      IF(ISTRT.NE.0) ISUM=ISUM+NRCL
      ISP=ISUM-ISOLD
C
C7------PRINT AMOUNT OF SPACE USED.
      WRITE(IOUT,6) ISP
    6 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY BAS')
```

```
      ISUM1=ISUM-1
      WRITE(IOUT,7) ISUM1,LENX
    7 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,8)
    8 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C
C8------RETURN
      RETURN
C
      END
```

Module BAS5RP

Narrative for Module BAS5RP

This module reads and prepares data for the BAS Package. It reads the boundary array (IBOUND), reads initial values for head (HNEW), copies initial head to array STRT if initial heads are to be kept throughout the simulation, initializes the volumetric-budget accumulators (VBVL), and sets up the Output Control System. Initial heads are read even for steady-state problems because the iterative solvers require an estimate of head to start the solution process. This module also reads a head value, HNOFLO, to which head is set at no-flow cells. Because head at no-flow cells is not part of any model calculations, HNOFLO has no effect on the calculated head at variable-head cells or on any model calculation. The only purpose of HNOFLO is for making inactive cells stand out when head is printed (e.g., 0.0 or 9999.99) or stored in a disk file for use by another program.

The IBOUND codes are as follows.

IBOUND Code	Meaning
negative	constant-head cell
zero	inactive (no-flow) cell
positive	variable-head cell

Module BAS1RP performs its functions as follows:

1. Print the simulation title and calculate the number of cells in a layer.

2. Read the boundary array (IBOUND). If a cross section, read a single 2-D array for the cross section. If not a cross section, read multiple 2-D arrays -- one for each layer.

3. Read and print the head value to be printed for no-flow cells (HNOFLO).

4. Read the initial heads. The initial heads are read as single precision values into the array HOLD and converted to double precision as they are moved into HNEW. If a cross section, read a single 2-D array for the cross section. If not a cross section, read multiple 2-D arrays -- one for each layer.

5. Copy the initial heads (and convert to double precision) from HOLD into HNEW. At no-flow cells, set HNEW equal to HNOFLO.

6. If the initial heads must be kept throughout the simulation, copy them from HOLD to STRT.

7. Initialize volumetric-budget accumulators.

8. Call submodule SBAS5I to initialize the Output Control System.

9. RETURN.

BAS5RP

```
      SUBROUTINE BAS5RP(IBOUND,HNEW,STRT,HOLD,ISTRT,INBAS,HEADNG,NCOL,
     1    NROW,NLAY,VBVL,IOFLG,INOC,IHEDFM,IDDNFM,IHEDUN,IDDNUN,IOUT,
     2    IPEROC,ITSOC,CHEDFM,CDDNFM,IBDOPT,IXSEC,LBHDSV,LBDDSV,IFREFM)
C-----VERSION 1345 20FEB1996 BAS5RP
C     ******************************************************************
C     READ AND INITIALIZE BASIC MODEL ARRAYS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*80 HEADNG(2)
      CHARACTER*24 ANAME(2)
      DOUBLE PRECISION HNEW,HNOFLO
      DIMENSION HNEW(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),
     1     STRT(NCOL,NROW,NLAY),HOLD(NCOL,NROW,NLAY),
     2     VBVL(4,40),IOFLG(NLAY,4)
      CHARACTER*20 CHEDFM,CDDNFM
C
      DATA ANAME(1) /'          BOUNDARY ARRAY'/
      DATA ANAME(2) /'            INITIAL HEAD'/
C     ------------------------------------------------------------------
C
C1------PRINT SIMULATION TITLE, CALCULATE # OF CELLS IN A LAYER.
      WRITE(IOUT,'(''1'',/1X,A)') HEADNG(1)
      WRITE(IOUT,'(1X,A)') HEADNG(2)
C
C2------READ BOUNDARY ARRAY(IBOUND) ONE LAYER AT A TIME.
      IF(IXSEC.EQ.0) THEN
         DO 100 K=1,NLAY
         KK=K
         CALL U2DINT(IBOUND(1,1,KK),ANAME(1),NROW,NCOL,KK,INBAS,IOUT)
  100    CONTINUE
      ELSE
         CALL U2DINT(IBOUND(1,1,1),ANAME(1),NLAY,NCOL,-1,INBAS,IOUT)
      END IF
C
C3------READ AND PRINT HEAD VALUE TO BE PRINTED FOR NO-FLOW CELLS.
      IF(IFREFM.EQ.0) THEN
         READ(INBAS,'(F10.0)') TMP
      ELSE
         READ(INBAS,*) TMP
      END IF
      HNOFLO=TMP
      WRITE(IOUT,3) TMP
    3 FORMAT(1X,/1X,'AQUIFER HEAD WILL BE SET TO ',1PG11.5,
     1       ' AT ALL NO-FLOW NODES (IBOUND=0).')
C
C4------READ INITIAL HEADS.
      IF(IXSEC.EQ.0) THEN
         DO 300 K=1,NLAY
         KK=K
         CALL U2DREL(HOLD(1,1,KK),ANAME(2),NROW,NCOL,KK,INBAS,IOUT)
  300    CONTINUE
      ELSE
         CALL U2DREL(HOLD(1,1,1),ANAME(2),NLAY,NCOL,-1,INBAS,IOUT)
      END IF
C
C5------COPY INITIAL HEADS FROM HOLD TO HNEW.
      DO 400 K=1,NLAY
      DO 400 I=1,NROW
      DO 400 J=1,NCOL
      HNEW(J,I,K)=HOLD(J,I,K)
      IF(IBOUND(J,I,K).EQ.0) HNEW(J,I,K)=HNOFLO
  400 CONTINUE
C
C6------IF STARTING HEADS ARE TO BE SAVED THEN COPY HOLD TO STRT.
      IF(ISTRT.EQ.0) GO TO 590
      DO 500 K=1,NLAY
      DO 500 I=1,NROW
      DO 500 J=1,NCOL
      STRT(J,I,K)=HOLD(J,I,K)
  500 CONTINUE
C
C7------INITIALIZE VOLUMETRIC BUDGET ACCUMULATORS TO ZERO.
  590 ZERO=0.
      DO 600 I=1,40
      DO 600 J=1,4
```

```
      VBVL(J,I)=ZERO
  600 CONTINUE
C
C8------SET UP OUTPUT CONTROL.
      CALL SBAS5I(NLAY,ISTRT,IOFLG,INOC,IOUT,IHEDFM,IDDNFM,IHEDUN,
     1   IDDNUN,IPEROC,ITSOC,CHEDFM,CDDNFM,IBDOPT,LBHDSV,LBDDSV,IFREFM)
C
C9------RETURN
 1000 RETURN
      END
```

List of Variables for Module BAS5RP

Variable	Range	Definition
ANAME	Module	CHARACTER*24(2), Labels that identify the boundary and initial head arrays.
CDDNFM	Package	CHARACTER*20, Format that a user can specify for writing drawdown in a disk file.
CHEDFM	Package	CHARACTER*20, Format that a user can specify for writing head in a disk file.
HEADNG	Global	CHARACTER*80(2), Heading printed in output to identify problem.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
HNOFLO	Module	User specified value for head in cells that are no flow at the start of the simulation (Double Precision).
HOLD	Global	DIMENSION (NCOL,NROW,NLAY), Head at the start of the current time step.
I	Module	Index for rows.
IBDOPT	Package	Flag indicating how cell-by-cell budget data will be written: 1 - All budget terms will be written as a 3-D arrays. 2 - The form for writing budget data will be selected in order to minimize the amount of disk space.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
IDDNFM	Package	Code that specifies the format for printing drawdown.
IDDNUN	Package	Unit number for saving drawdown in a disk file.
IHEDFM	Package	Code that specifies the format for printing head.
IHEDUN	Package	Unit number for saving head in a disk file.
INBAS	Package	Input unit number for the Basic Package.
INOC	Package	Input unit number for the Output Control Option.
IOFLG	Package	DIMENSION (NLAY,4), Flags to control printing and saving of head and drawdown for each layer.
IOUT	Global	Unit number for writing to the listing file.
IPEROC	Package	For alphabetic output control, the stress period at which the next output is requested. IPEROC=-1 for numeric output control.
ISTRT	Global	Flag for keeping initial heads throughout the simulation: = 0, initial heads will not be kept in memory after the simulation starts. ≠ 0, initial heads will be kept in memory throughout the simulation so that drawdown can be calculated.
ITSOC	Package	For alphabetic output control, the time step at which the next output is requested. ITSOC=-1 for numeric output control.
IXSEC	Global	Cross section flag: = 0, the model is not a 1-row cross section. ≠ 0, the model is a 1-row cross section.

J	Module	Index for columns.
K	Module	Index for layers.
KK	Module	Temporary variable set equal to K. KK is used as the calling argument to subroutines in order to avoid problems with some compilers if a DO loop index is an argument.
LBDDSV	Package	Label flag for saving drawdown in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
LBHDSV	Package	Label flag for saving head in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
STRT	Global	DIMENSION (NCOL,NROW,NLAY), Initial (starting) head.
TMP	Module	Single-precision equivalent of HNOFLO.
VBVL	Global	DIMENSION (4,40), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
ZERO	Module	The constant 0.

Module BAS5ST

```
      SUBROUTINE BAS5ST(NSTP,DELT,TSMULT,PERTIM,KPER,INBAS,IOUT,IFREFM)
C
C
C-----VERSION 1418 20FEB1996 BAS5ST
C     ******************************************************************
C     SETUP TIME PARAMETERS FOR NEW TIME PERIOD
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------READ AND WRITE LENGTH OF STRESS PERIOD, NUMBER OF TIME STEPS AND
C1------TIME STEP MULTIPLIER.
      IF(IFREFM.EQ.0) THEN
         READ(INBAS,'(F10.0,I10,F10.0)') PERLEN,NSTP,TSMULT
      ELSE
         READ(INBAS,*) PERLEN,NSTP,TSMULT
      END IF
      WRITE (IOUT,1) KPER,PERLEN,NSTP,TSMULT
    1 FORMAT('1',/28X,'STRESS PERIOD NO.',I4,', LENGTH =',G15.7,/
     1            28X,46('-'),//
     2            30X,'NUMBER OF TIME STEPS =',I6,//
     3            31X,'MULTIPLIER FOR DELT =',F10.3)
C
C1A-----STOP IF NSTP LE 0, PERLEN LE 0., OR TSMULT LE 0.
      IF(NSTP.LE.0) THEN
         WRITE(IOUT,2)
    2    FORMAT(1X,/1X,'THERE MUST BE AT LEAST ONE TIME STEP')
         STOP
      END IF
      ZERO=0.
      IF(PERLEN.LE.ZERO) THEN
         WRITE(IOUT,3)
    3    FORMAT(1X,/1X,'PERLEN MUST BE GREATER THAN 0.0')
         STOP
      END IF
      IF(TSMULT.LE.ZERO) THEN
         WRITE(IOUT,4)
    4    FORMAT(1X,/1X,'TSMULT MUST BE GREATER THAN 0.0')
         STOP
      END IF
C
C2------CALCULATE THE LENGTH OF THE FIRST TIME STEP.
C
C2A-----ASSUME TIME STEP MULTIPLIER IS EQUAL TO ONE.
      DELT=PERLEN/FLOAT(NSTP)
C
C2B-----IF TIME STEP MULTIPLIER IS NOT ONE THEN CALCULATE FIRST
C2B-----TERM OF GEOMETRIC PROGRESSION.
      ONE=1.
      IF(TSMULT.NE.ONE) DELT=PERLEN*(ONE-TSMULT)/(ONE-TSMULT**NSTP)
C
C3------PRINT THE LENGTH OF THE FIRST TIME STEP.
      WRITE (IOUT,9) DELT
    9 FORMAT(1X,/28X,'INITIAL TIME STEP SIZE =',G15.7)
C
C4------INITIALIZE PERTIM (ELAPSED TIME WITHIN STRESS PERIOD).
      PERTIM=0.
C
C5------RETURN
      RETURN
      END
```

Module BAS5AD

```
      SUBROUTINE BAS5AD(DELT,TSMULT,TOTIM,PERTIM,HNEW,HOLD,KSTP,
     1                  NCOL,NROW,NLAY)
C
C-----VERSION 1412 22FEB1982 BAS5AD
C
C     ******************************************************************
C     ADVANCE TO NEXT TIME STEP
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW
C
      DIMENSION HNEW(NCOL,NROW,NLAY), HOLD(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------IF NOT FIRST TIME STEP THEN CALCULATE TIME STEP LENGTH.
      IF(KSTP.NE.1) DELT=TSMULT*DELT
C
C2------ACCUMULATE ELAPSED TIME IN SIMULATION(TOTIM) AND IN THIS
C2------STRESS PERIOD(PERTIM).
      TOTIM=TOTIM+DELT
      PERTIM=PERTIM+DELT
C
C3------COPY HNEW TO HOLD.
      DO 10 K=1,NLAY
      DO 10 I=1,NROW
      DO 10 J=1,NCOL
   10 HOLD(J,I,K)=HNEW(J,I,K)
C
C4------RETURN
      RETURN
      END
```

Module BAS5FM

```
      SUBROUTINE BAS5FM(HCOF,RHS,NODES)
C
C
C-----VERSION 1412 02JULY1993 BAS5FM
C     ******************************************************************
C     SET HCOF=RHS=0.
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION HCOF(NODES),RHS(NODES)
C     ------------------------------------------------------------------
C
C1------FOR EACH CELL INITIALIZE HCOF AND RHS ACCUMULATORS.
      ZERO=0.
      DO 100 I=1,NODES
      HCOF(I)=ZERO
      RHS(I)=ZERO
  100 CONTINUE
C
C2------RETURN
      RETURN
      END
```

Module BAS5OC

Narrative for Module BAS5OC

Module BAS5OC is called every time step to set flags used by the budget and output procedures that determine which data are written to the listing file and other files. The flags are numeric codes. These flags can be defined using two methods. The original method reads the numeric codes for these flags while the new method reads alphabetic words that indicate how the flags should be set. A few new options, which can only be invoked through the word form of output control, have also been added.

Individual flags are IHDDFL which indicates that head or drawdown is to be printed or recorded, IBUDFL which indicates that the overall budget should be printed, and ICBCFL which indicates that cell-by-cell flow terms should be calculated and printed or recorded. A table of flags, IOFLG, has four flags for each layer. They correspond to the four options: print heads, print drawdown, save heads, and save drawdown. The flags in IOFLG are used in conjunction with the flag IHDDFL. If IHDDFL is set, IOFLG is used to determine head and drawdown on a layer-by-layer basis. If IHDDFL is not set, heads and drawdown are not printed or saved and IOFLG is ignored.

Module BAS1OC performs its functions as follows:

1. Determine if the Output Control Option is inactive. If inactive, the twelfth element of the IUNIT array will be less than or equal to 0. IUNIT(12) is passed to BAS1OC as variable INOC. If output control is inactive, set flags for default output. Flags IBUDFL and IHDDFL are set only for the last time step of each stress period. Go to 1000 after assigning flags.

2. Output control is active; check to see if using word input. If so, IPEROC will be greater than or equal to 0. (IPEROC is set by module SBAS1I through module BAS5RP.) If using word input, call module SBAS1N to set the flags. Upon return, go to 8.

3. Use the original method for defining output control flags. INOC is the unit number for reading output control data. Read and print INCODE and flags IHDDFL, IBUDFL, and ICBCFL.

4. The code INCODE gives the user several options for specifying IOFLG. Determine whether INCODE is less than zero, equal to zero, or greater than zero. Go to 5, 6, or 7.

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

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

7. INCODE is greater than zero. Read and print IOFLG table for all layers unless the model is a cross section. If a cross section, read IOFLG for a single layer, which will apply to the cross section.

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

9. RETURN.

BAS5OC

```
      SUBROUTINE BAS5OC(NSTP,KSTP,ICNVG,IOFLG,NLAY,IBUDFL,ICBCFL,
     1      IHDDFL,INOC,IOUT,KPER,IPEROC,ITSOC,IBDOPT,IXSEC,IFREFM)
C
C-----VERSION 1340 20FEB1996 BAS5OC
C     ******************************************************************
C     OUTPUT CONTROLLER FOR HEAD, DRAWDOWN, AND BUDGET
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION IOFLG(NLAY,4)
C     ------------------------------------------------------------------
C
C1------TEST UNIT NUMBER (INOC (INOC=IUNIT(12))) TO SEE IF
C1------OUTPUT CONTROL IS ACTIVE.  IF NOT, SET DEFAULTS AND RETURN.
      IF(INOC.LE.0) THEN
         IHDDFL=0
         IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP)IHDDFL=1
         IBUDFL=0
         IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP)IBUDFL=1
         ICBCFL=0
         GO TO 1000
      END IF
C
C2------OUTPUT CONTROL IS ACTIVE.  IF IPEROC >= 0, READ OUTPUT FLAGS
C2------USING ALPHABETIC INPUT STRUCTURE.
      IF(IPEROC.GE.0) THEN
         CALL SBAS5N(IOFLG,NLAY,IHDDFL,IBUDFL,ICBCFL,IPEROC,ITSOC,KPER,
     1           KSTP,INOC,IOUT,IBDOPT)
         GO TO 600
      END IF
C
C3------READ AND PRINT OUTPUT FLAGS AND CODE FOR DEFINING IOFLG USING
C3------THE ORIGINAL NUMERIC INPUT STRUCTURE.
      IF(IFREFM.EQ.0) THEN
         READ(INOC,'(4I10)') INCODE,IHDDFL,IBUDFL,ICBCFL
      ELSE
         READ(INOC,*) INCODE,IHDDFL,IBUDFL,ICBCFL
      END IF
      WRITE(IOUT,3) IHDDFL,IBUDFL,ICBCFL
    3 FORMAT(1X,/1X,'HEAD/DRAWDOWN PRINTOUT FLAG =',I2,
     1    5X,'TOTAL BUDGET PRINTOUT FLAG =',I2,
     2   /1X,'CELL-BY-CELL FLOW TERM FLAG =',I2)
      IF(ICBCFL.NE.0) ICBCFL=IBDOPT
C
C4------DECODE INCODE TO DETERMINE HOW TO SET FLAGS IN IOFLG.
      IF(INCODE) 100,200,300
C
C5------USE IOFLG FROM LAST TIME STEP.
  100 WRITE(IOUT,101)
  101 FORMAT(1X,'REUSING PREVIOUS VALUES OF IOFLG')
      GO TO 600
C
C6------READ IOFLG FOR LAYER 1 AND ASSIGN SAME TO ALL LAYERS
  200 IF(IFREFM.EQ.0) THEN
         READ(INOC,'(4I10)') (IOFLG(1,M),M=1,4)
      ELSE
         READ(INOC,*) (IOFLG(1,M),M=1,4)
      END IF
      DO 210 K=1,NLAY
      IOFLG(K,1)=IOFLG(1,1)
      IOFLG(K,2)=IOFLG(1,2)
      IOFLG(K,3)=IOFLG(1,3)
      IOFLG(K,4)=IOFLG(1,4)
  210 CONTINUE
      WRITE(IOUT,211) (IOFLG(1,M),M=1,4)
  211 FORMAT(1X,/1X,'OUTPUT FLAGS FOR ALL LAYERS ARE THE SAME:'/
     1   1X,'  HEAD    DRAWDOWN  HEAD  DRAWDOWN'/
     2   1X,'PRINTOUT  PRINTOUT  SAVE    SAVE'/
     3   1X,34('-')/1X,I5,I10,I8,I8)
      GO TO 600
C
C7------READ IOFLG IN ENTIRETY -- IF CROSS SECTION, READ ONLY ONE VALUE.
  300 IF(IXSEC.EQ.0) THEN
         DO 301 K=1,NLAY
         IF(IFREFM.EQ.0) THEN
            READ(INOC,'(4I10)') (IOFLG(K,M),M=1,4)
```

```
         ELSE
            READ(INOC,*) (IOFLG(K,M),M=1,4)
         END IF
  301    CONTINUE
         WRITE(IOUT,302) 'OUTPUT FLAGS FOR EACH LAYER:','LAYER'
  302    FORMAT(1X,/1X,A,/
     1   1X,'         HEAD    DRAWDOWN  HEAD  DRAWDOWN'/
     2   1X,A,'  PRINTOUT  PRINTOUT  SAVE    SAVE'/
     3   1X,41('-'))
         WRITE(IOUT,303) (K,(IOFLG(K,M),M=1,4),K=1,NLAY)
  303    FORMAT(1X,I4,I8,I10,I8,I8)
      ELSE
         IF(IFREFM.EQ.0) THEN
            READ(INOC,'(4I10)') (IOFLG(1,M),M=1,4)
         ELSE
            READ(INOC,*) (IOFLG(1,M),M=1,4)
         END IF
         WRITE(IOUT,302) 'OUTPUT FLAGS FOR CROSS SECTION:','     '
         WRITE(IOUT,304) (IOFLG(1,M),M=1,4)
  304    FORMAT(1X,I12,I10,I8,I8)
      END IF
C
C8------THE LAST STEP IN A STRESS PERIOD AND STEPS WHERE ITERATIVE
C8------PROCEDURE FAILED TO CONVERGE GET A VOLUMETRIC BUDGET.
  600 IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP) IBUDFL=1
C
C9------RETURN
 1000 RETURN
C
      END
```

List of Variables for Module BAS5OC

Variable	Range	Definition
IBDOPT	Package	Flag indicating how cell-by-cell budget data will be written: 1 - All budget terms will be written as a 3-D arrays. 2 - The form for writing budget data will be selected in order to minimize the amount of disk space.
IBUDFL	Package	Volumetric budget print flag: = 0, volumetric budget is not printed for the current time step. ≠ 0, volumetric budget is printed for the current time step.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
ICNVG	Global	Flag that is set equal to one when the iteration procedure has converged.
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
IHDDFL	Package	Flag for using IOFLG in the current time step: = 0, regardless of IOFLG values, no head or drawdown values are printed or saved. ≠ 0, IOFLG values are used to determine if head and drawdown are printed and saved.
INCODE	Module	Flag for how IOFLG values are defined in the current time step: < 0, reuse values for IOFLG from the previous time step. = 0, read IOFLG for 1 layer, and the same values for all layers. > 0, read IOFLG values for each layer.
INOC	Package	Input unit number for the Output Control Option.
IOFLG	Package	DIMENSION (NLAY,4), Flags to control printing and saving of head and drawdown for each layer: (N,1) ≠ 0, head will be printed for layer N. (N,2) ≠ 0, drawdown will be printed for layer N. (N,3) ≠ 0, head will be saved for layer N. (N,4) ≠ 0, drawdown will be saved for layer N.
IOUT	Global	Unit number for writing to the listing file.
IPEROC	Package	For alphabetic output control, the stress period at which the next output is requested. IPEROC=-1 for numeric output control.
ITSOC	Package	For alphabetic output control, the time step at which the next output is requested. ITSOC=-1 for numeric output control.
IXSEC	Global	Cross section flag: = 0, the model is not a 1-row cross section. ≠ 0, the model is a 1-row cross section.
K	Module	Index for layers.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
M	Module	Index for the 2nd subscript in IOFLG.
NLAY	Global	The number of layers in the grid.
NSTP	Global	The number of time steps in the current stress period.

Module BAS5OT

```
      SUBROUTINE BAS5OT(HNEW,STRT,ISTRT,BUFF,IOFLG,MSUM,IBOUND,VBNM,
     1  VBVL,KSTP,KPER,DELT,PERTIM,TOTIM,ITMUNI,NCOL,NROW,NLAY,ICNVG,
     2  IHDDFL,IBUDFL,IHEDFM,IHEDUN,IDDNFM,IDDNUN,IOUT,CHEDFM,CDDNFM,
     3  IXSEC,LBHDSV,LBDDSV)
C-----VERSION 1647 29OCT1992 BAS5OT
C     ******************************************************************
C     OUTPUT TIME, VOLUMETRIC BUDGET, HEAD, AND DRAWDOWN
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM)
      DOUBLE PRECISION HNEW
      DIMENSION HNEW(NCOL,NROW,NLAY),STRT(NCOL,NROW,NLAY),
     1          VBVL(4,MSUM),IOFLG(NLAY,4),IBOUND(NCOL,NROW,NLAY),
     2          BUFF(NCOL,NROW,NLAY)
      CHARACTER*20 CHEDFM,CDDNFM
C     ------------------------------------------------------------------
C
C1------CLEAR PRINTOUT FLAG (IPFLG)
      IPFLG=0
C
C2------IF ITERATIVE PROCEDURE FAILED TO CONVERGE PRINT MESSAGE
      IF(ICNVG.EQ.0) WRITE(IOUT,1) KSTP,KPER
    1 FORMAT(1X,/11X,'****FAILED TO CONVERGE IN TIME STEP',I3,
     1       ' OF STRESS PERIOD',I3,'****')
C
C3------IF HEAD AND DRAWDOWN FLAG (IHDDFL) IS SET WRITE HEAD AND
C3------DRAWDOWN IN ACCORDANCE WITH FLAGS IN IOFLG.
      IF(IHDDFL.EQ.0) GO TO 100
C
      CALL SBAS5H(HNEW,BUFF,IOFLG,KSTP,KPER,NCOL,NROW,NLAY,IOUT,
     1    IHEDFM,IHEDUN,IPFLG,PERTIM,TOTIM,CHEDFM,IXSEC,LBHDSV,IBOUND)
      CALL SBAS5D(HNEW,BUFF,IOFLG,KSTP,KPER,NCOL,NROW,NLAY,IOUT,IDDNFM,
     1 IDDNUN,STRT,ISTRT,IBOUND,IPFLG,PERTIM,TOTIM,CDDNFM,IXSEC,
     2 LBDDSV)
C
C4------PRINT TOTAL BUDGET IF REQUESTED
  100 IF(IBUDFL.EQ.0) GO TO 120
      CALL SBAS5V(MSUM,VBNM,VBVL,KSTP,KPER,IOUT)
      IPFLG=1
C
C5------END PRINTOUT WITH TIME SUMMARY AND FORM FEED IF ANY PRINTOUT
C5------WILL BE PRODUCED.
  120 IF(IPFLG.EQ.0) RETURN
      CALL SBAS5T(KSTP,KPER,DELT,PERTIM,TOTIM,ITMUNI,IOUT)
      WRITE(IOUT,101)
  101 FORMAT('1')
C
C6------RETURN
      RETURN
      END
```

Module SBAS5D

Narrative for Module SBAS5D

Module SBAS5D is called by module BAS5OT to calculate and write drawdown for every cell in specified layers of the grid. The module is called at the end of each time step if the head and drawdown flag (IHDDFL) is set. It calculates drawdown only if the user has specified that initial heads should be kept throughout the simulation.

The layers for which drawdown is to be written are determined by the settings of flags in the table named IOFLG. In IOFLG, there are four flags for each layer. If the second flag is set, drawdown is written to the listing file (i.e. printed). If the fourth flag is set, drawdown is written on unit IDDNUN.

Module SBAS5D performs its functions in the following order:

1. For each layer, do steps 2-4.

2. If flags indicate that drawdown is not needed for this layer, go on to the next layer.

3. Drawdown is needed for this layer. Test flag ISTRT to see if initial heads were kept throughout the simulation. If initial heads were not kept, write a message to that effect and STOP.

4. Initial heads were kept. Calculate drawdown for this layer.

5. Check if the model is a cross section. If not a cross section, for each layer check if drawdown is to be written to the listing file. If so, call module ULAPRS or ULAPRW, depending on the format requested (IDDNFM), to write drawdown.

    A. If a cross section, check if drawdown is to be written to the listing file. If so, call module ULAPRS or ULAPRW, depending on the format requested (IDDNFM), to write drawdown.

6. Check if there is a positive unit number (IDDNUN) for saving drawdown in a separate disk file. If not, go to step 7. If IDDNUN is positive, check if the model is a cross section. If not a cross section, for each layer check if drawdown is to be written to unit IDDNUN. If so, call module ULASAV or ULASV2, depending on the format requested (CDDNFM), to write drawdown.

    A. If a cross section, check if drawdown is to be written to unit IDDNUN. If so, call module ULASAV or ULASV2, depending on the format requested (CDDNFM), to write drawdown.

7. RETURN.

```
      SUBROUTINE SBAS5D(HNEW,BUFF,IOFLG,KSTP,KPER,NCOL,NROW,
     1    NLAY,IOUT,IDDNFM,IDDNUN,STRT,ISTRT,IBOUND,IPFLG,
     2    PERTIM,TOTIM,CDDNFM,IXSEC,LBDDSV)
C-----VERSION 1648 25JUNE1993 SBAS5D
C     ******************************************************************
C     CALCULATE, PRINT, AND SAVE DRAWDOWNS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 TEXT
      DOUBLE PRECISION HNEW,SSTRT
      DIMENSION HNEW(NCOL,NROW,NLAY),IOFLG(NLAY,4),
     1     BUFF(NCOL,NROW,NLAY),STRT(NCOL,NROW,NLAY),
     1     IBOUND(NCOL,NROW,NLAY)
      CHARACTER*20 CDDNFM
C
      DATA TEXT /'        DRAWDOWN'/
C     ------------------------------------------------------------------
C
C1------FOR EACH LAYER CALCULATE DRAWDOWN IF PRINT OR SAVE IS REQUESTED.
      DO 59 K=1,NLAY
C
C2------IS DRAWDOWN NEEDED FOR THIS LAYER?
      KL=K
      IF(IXSEC.NE.0) KL=1
      IF(IOFLG(KL,2).EQ.0 .AND. IOFLG(KL,4).EQ.0) GO TO 59
C
C3------DRAWDOWN IS NEEDED. WERE INITIAL HEADS KEPT?
      IF(ISTRT.EQ.0) THEN
         WRITE(IOUT,52)
   52    FORMAT(1X,/1X,'CANNOT CALCULATE DRAWDOWN BECAUSE INITIAL HEAD',
     1   ' WAS NOT KEPT AFTER THE'/1X,
     2   'SIMULATION STARTED.  SEE "ISTRT" PARAMETER IN BAS INPUT.')
         STOP
      END IF
C
C4------CALCULATE DRAWDOWN FOR THE LAYER.
      DO 58 I=1,NROW
      DO 58 J=1,NCOL
      BUFF(J,I,K)=HNEW(J,I,K)
      SSTRT=STRT(J,I,K)
      IF(IBOUND(J,I,K).NE.0) BUFF(J,I,K)=SSTRT-HNEW(J,I,K)
   58 CONTINUE
   59 CONTINUE
C
C5------FOR EACH LAYER: DETERMINE IF DRAWDOWN SHOULD BE PRINTED.
C5------IF SO THEN CALL ULAPRS OR ULAPRW TO PRINT DRAWDOWN.
      IF(IXSEC.EQ.0) THEN
        DO 69 K=1,NLAY
        KK=K
        IF(IOFLG(K,2).EQ.0) GO TO 69
        IF(IDDNFM.LT.0) CALL ULAPRS(BUFF(1,1,K),TEXT,KSTP,KPER,
     1                  NCOL,NROW,KK,-IDDNFM,IOUT)
        IF(IDDNFM.GE.0) CALL ULAPRW(BUFF(1,1,K),TEXT,KSTP,KPER,
     1                 NCOL,NROW,KK,IDDNFM,IOUT)
        IPFLG=1
   69   CONTINUE
C
C5A-----PRINT DRAWDOWN FOR CROSS SECTION.
      ELSE
        IF(IOFLG(1,2).NE.0) THEN
          IF(IDDNFM.LT.0) CALL ULAPRS(BUFF,TEXT,KSTP,KPER,
     1                NCOL,NLAY,-1,-IDDNFM,IOUT)
          IF(IDDNFM.GE.0) CALL ULAPRW(BUFF,TEXT,KSTP,KPER,
     1                NCOL,NLAY,-1,IDDNFM,IOUT)
          IPFLG=1
        END IF
      END IF
C
C6------FOR EACH LAYER: DETERMINE IF DRAWDOWN SHOULD BE SAVED.
C6------IF SO THEN CALL A ULASAV OR ULASV2 TO RECORD DRAWDOWN.
      IFIRST=1
      IF(IDDNUN.LE.0) GO TO 80
      IF(IXSEC.EQ.0) THEN
        DO 79 K=1,NLAY
        KK=K
```

```
         IF(IOFLG(K,4).EQ.0) GO TO 79
         IF(IFIRST.EQ.1) WRITE(IOUT,74) IDDNUN,KSTP,KPER
   74    FORMAT(1X,/1X,'DRAWDOWN WILL BE SAVED ON UNIT',I4,
     1       ' AT END OF TIME STEP',I3,', STRESS PERIOD',I3)
         IFIRST=0
         IF(CDDNFM.EQ.' ') THEN
            CALL ULASAV(BUFF(1,1,K),TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                  NROW,KK,IDDNUN)
         ELSE
            CALL ULASV2(BUFF(1,1,K),TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                  NROW,KK,IDDNUN,CDDNFM,LBDDSV,IBOUND(1,1,K))
         END IF
   79    CONTINUE
C
C6A-----SAVE DRAWDOWN FOR CROSS SECTION.
       ELSE
         IF(IOFLG(1,4).NE.0) THEN
           WRITE(IOUT,74) IDDNUN,KSTP,KPER
           IF(CDDNFM.EQ.' ') THEN
              CALL ULASAV(BUFF,TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                  NLAY,-1,IDDNUN)
           ELSE
              CALL ULASV2(BUFF,TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                   NLAY,-1,IDDNUN,CDDNFM,LBDDSV,IBOUND)
           END IF
         END IF
       END IF
C
C7------RETURN.
   80 RETURN
      END
```

List of Variables for Module SBAS5D

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
CDDNFM	Package	CHARACTER*20, Format that a user can specify for writing drawdown in a disk file.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
I	Module	Index for rows.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IDDNFM	Package	Code that specifies the format for printing drawdown.
IDDNUN	Package	Unit number for saving drawdown in a disk file.
IFIRST	Module	Flag used when printing a notice that drawdown is being saved; the flag prevents multiple notices being printed.
IOFLG	Package	DIMENSION (NLAY,4), Flags to control printing and saving of head and drawdown for each layer: (N,1) ≠ 0, head will be printed for layer N. (N,2) ≠ 0, drawdown will be printed for layer N. (N,3) ≠ 0, head will be saved for layer N. (N,4) ≠ 0, drawdown will be saved for layer N.
IOUT	Global	Unit number for writing to the listing file.
IPFLG	Package	Flag that indicates if head, drawdown, or budget has been printed this time step; if so, the flag is nonzero.
ISTRT	Global	Flag for keeping initial heads throughout the simulation: = 0, initial heads will not be kept in memory after the simulation starts. ≠ 0, initial heads will be kept in memory throughout the simulation so that drawdown can be calculated.
IXSEC	Global	Cross section flag: = 0, the model is not a 1-row cross section. ≠ 0, the model is a 1-row cross section.
J	Module	Index for columns.
K	Module	Index for layers.
KK	Module	Temporary variable set equal to K. KK is used as the calling argument to subroutines in order to avoid problems with some compilers if a DO loop index is an argument.
KL	Module	Index for layers. KL=K when not a cross section, and KL=1 when a cross section.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
LBDDSV	Package	Label flag for saving drawdown in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.

NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
SSTRT	Module	Double precision equivalent of STRT at a cell.
STRT	Global	DIMENSION (NCOL,NROW,NLAY), Initial (starting) head.
TEXT	Module	CHARACTER*16, Label that identifies drawdown when printed or saved.
TOTIM	Global	Elapsed time in the simulation.

Module SBAS5H

Narrative for Module SBAS5H

Module SBAS5H is called by module BAS5OT to write head for every cell in specified layers of the grid. The module is called at the end of each time step if the head and drawdown flag (IHDDFL) is set.

The layers for which head is to be written are determined by the settings of flags in the table named IOFLG. In IOFLG, there are four flags for each layer. If the second flag is set, head is written to the listing file (i.e. printed). If the fourth flag is set, head is written on unit IHEDUN.

Module SBAS5H performs its functions in the following order:

1. For each layer, do steps 2-3.

2. If flags indicate that head is not needed for this layer, go on to the next layer.

3. Head is needed for this layer. Move head to BUFF for this layer.

4. Check if the model is a cross section. If not a cross section, for each layer check if head is to be written to the listing file. If so, call module ULAPRS or ULAPRW, depending on the format requested (IHEDFM), to write head.

    A. If a cross section, check if head is to be written to the listing file. If so, call module ULAPRS or ULAPRW, depending on the format requested (IHEDFM), to write head.

5. Check if there is a positive unit number (IHEDUN) for saving head in a separate disk file. If not, go to step 6. If IHEDUN is positive, check if the model is a cross section. If not a cross section, for each layer check if head is to be written to unit IHEDUN. If so, call module ULASAV or ULASV2, depending on the format requested (CHEDFM), to write head.

    A. If a cross section, check if head is to be written to unit IHEDUN. If so, call module ULASAV or ULASV2, depending on the format requested (CHEDFM), to write head.

6. RETURN.

SBAS5H

```
      SUBROUTINE SBAS5H(HNEW,BUFF,IOFLG,KSTP,KPER,NCOL,NROW,NLAY,IOUT,
     1   IHEDFM,IHEDUN,IPFLG,PERTIM,TOTIM,CHEDFM,IXSEC,LBHDSV,IBOUND)
C
C-----VERSION 1647 18OCT1993 SBAS5H
C     ******************************************************************
C     PRINT AND RECORD HEADS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 TEXT
      DOUBLE PRECISION HNEW
      DIMENSION HNEW(NCOL,NROW,NLAY),IOFLG(NLAY,4),BUFF(NCOL,NROW,NLAY),
     1   IBOUND(NCOL,NROW,NLAY)
      CHARACTER*20 CHEDFM
C
      DATA TEXT /'            HEAD'/
C     ------------------------------------------------------------------
C
C1------FOR EACH LAYER MOVE HNEW TO BUFF IF PRINT OR SAVE IS REQUESTED.
      DO 59 K=1,NLAY
C
C2------IS HEAD NEEDED FOR THIS LAYER?
      KL=K
      IF(IXSEC.NE.0) KL=1
      IF(IOFLG(KL,1).EQ.0 .AND. IOFLG(KL,3).EQ.0) GO TO 59
C
C3------MOVE HNEW TO BUFF FOR THE LAYER.
      DO 58 I=1,NROW
      DO 58 J=1,NCOL
      BUFF(J,I,K)=HNEW(J,I,K)
   58 CONTINUE
   59 CONTINUE
C
C4------FOR EACH LAYER: DETERMINE IF HEAD SHOULD BE PRINTED.
C4------IF SO THEN CALL ULAPRS OR ULAPRW TO PRINT HEAD.
      IF(IXSEC.EQ.0) THEN
        DO 69 K=1,NLAY
        KK=K
        IF(IOFLG(K,1).EQ.0) GO TO 69
        IF(IHEDFM.LT.0) CALL ULAPRS(BUFF(1,1,K),TEXT,KSTP,KPER,
     1               NCOL,NROW,KK,-IHEDFM,IOUT)
        IF(IHEDFM.GE.0) CALL ULAPRW(BUFF(1,1,K),TEXT,KSTP,KPER,
     1               NCOL,NROW,KK,IHEDFM,IOUT)
        IPFLG=1
   69   CONTINUE
C
C4A-----PRINT HEAD FOR CROSS SECTION.
      ELSE
        IF(IOFLG(1,1).NE.0) THEN
          IF(IHEDFM.LT.0) CALL ULAPRS(BUFF,TEXT,KSTP,KPER,
     1                NCOL,NLAY,-1,-IHEDFM,IOUT)
          IF(IHEDFM.GE.0) CALL ULAPRW(BUFF,TEXT,KSTP,KPER,
     1                NCOL,NLAY,-1,IHEDFM,IOUT)
          IPFLG=1
        END IF
      END IF
C
C5------FOR EACH LAYER: DETERMINE IF HEAD SHOULD BE SAVED ON DISK.
C5------IF SO THEN CALL ULASAV OR ULASV2 TO SAVE HEAD.
      IFIRST=1
      IF(IHEDUN.LE.0) GO TO 80
      IF(IXSEC.EQ.0) THEN
        DO 79 K=1,NLAY
        KK=K
        IF(IOFLG(K,3).EQ.0) GO TO 79
        IF(IFIRST.EQ.1) WRITE(IOUT,74) IHEDUN,KSTP,KPER
   74   FORMAT(1X,/1X,'HEAD WILL BE SAVED ON UNIT',I4,
     1      ' AT END OF TIME STEP',I3,', STRESS PERIOD',I3)
        IFIRST=0
        IF(CHEDFM.EQ.' ') THEN
           CALL ULASAV(BUFF(1,1,K),TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                 NROW,KK,IHEDUN)
        ELSE
           CALL ULASV2(BUFF(1,1,K),TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                 NROW,KK,IHEDUN,CHEDFM,LBHDSV,IBOUND(1,1,K))
        END IF
```

```
   79   CONTINUE
C
C5A-----SAVE HEAD FOR CROSS SECTION.
      ELSE
        IF(IOFLG(1,3).NE.0) THEN
          WRITE(IOUT,74) IHEDUN,KSTP,KPER
          IF(CHEDFM.EQ.' ') THEN
             CALL ULASAV(BUFF,TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                 NLAY,-1,IHEDUN)
          ELSE
             CALL ULASV2(BUFF,TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                   NLAY,-1,IHEDUN,CHEDFM,LBHDSV,IBOUND)
          END IF
        END IF
      END IF
C
C6------RETURN.
   80 RETURN
      END
```

List of Variables for Module SBAS5H

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
CHEDFM	Package	CHARACTER*20, Format that a user can specify for writing head in a disk file.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
I	Module	Index for rows.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cellIFIRSTModuleFlag used when printing a notice that head is being saved; the flag prevents multiple notices being printed.
IHEDFM	Package	Code that specifies the format for printing head.
IHEDUN	Package	Unit number for saving head in a disk file.
IOFLG	Package	DIMENSION (NLAY,4), Flags to control printing and saving of head and drawdown for each layer: (N,1) ≠ 0, head will be printed for layer N. (N,2) ≠ 0, drawdown will be printed for layer N. (N,3) ≠ 0, head will be saved for layer N. (N,4) ≠ 0, drawdown will be saved for layer N.
IOUT	Global	Unit number for writing to the listing file.
IPFLG	Package	Flag that indicates if head, drawdown, or budget has been printed this time step; if so, the flag is nonzero.
IXSEC	Global	Cross section flag: = 0, the model is not a 1-row cross section. ≠ 0, the model is a 1-row cross section.
J	Module	Index for columns.
K	Module	Index for layers.
KK	Module	Temporary variable set equal to K. KK is used as the calling argument to subroutines in order to avoid problems with some compilers if a DO loop index is an argument.
KL	Module	Index for layers. KL=K when not a cross section, and KL=1 when a cross section.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
LBHDSV	Package	Label flag for saving head in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
TEXT	Module	CHARACTER*16, Label that identifies head when printed or saved.
TOTIM	Global	Elapsed time in the simulation.

Module SBAS5I

Narrative for Module SBAS5I

Module SBAS5I initializes parameters for controlling the output of model results. If the Output Control option is inactive, the formats for printing head and drawdown are set to default values, and flags are set so that heads and drawdowns will be printed for all layers. If output control is active, the formats for printing and the unit numbers for recording head and drawdown are read.

A table named IOFLG contains one entry for each layer in the grid. Each entry consists of four flags corresponding to four operations: (1) head print (write to the listing file), (2) drawdown print, (3) write head to a separate file, and (4) write drawdown to a separate file. The module BAS5OT examines the table and, for each layer, performs only the operations for which the corresponding flags are set (equal to one). SBAS5I sets values for IOFLG if output control is inactive. If output control is active, the flags in IOFLG are read at each time step by module BAS5OC.

There are two methods of controlling output -- the original method that uses numeric codes and a new method that uses alphabetic words. SBAS5I detects which method is being used and acts accordingly.

Module SBAS5I performs its functions as follows:

1. Assign default values to various variables.

2. Test IUNIT (12), which is passed to this module as variable INOC, to see if it is positive. If INOC is positive, then the Output Control option is active and Output Control data will be read from the unit number contained in INOC.

    A. INOC is not positive, so the Output Control option is inactive. Print a message that tells what the defaults are.

    B. Set IOFLG values to default values; head will be printed for all layers. Also, if initial head is kept throughout the simulation, drawdown will be printed for all layers. Go to step 5.

3. Output Control is active. Read first record and use URWORD to parse the first word.

4. Test for numeric method of Output Control. If the numeric method is used, then the first word will not be "PERIOD", "HEAD", "DRAWDOWN", or "COMPACT".

    A. The numeric method is being used. For free format, use URWORD to parse the four numeric parameters on the first record: the head-print format code (IHEDFM), the drawdown-print format code (IDDNFM), the unit number to record heads (IHEDUN), and the unit number to record drawdown (IDDNUN). Also print these values. Set IPEROC and ITSOC to -1 to indicate to other modules that the numeric method of specifying Output Control is being used.

    B. The alphabetic word method of specifying Output Control data is being used. Call

Module SBAS5J to process the initial output control records.

5. RETURN.

SBAS5I

```
      SUBROUTINE SBAS5I(NLAY,ISTRT,IOFLG,INOC,IOUT,IHEDFM,IDDNFM,IHEDUN,
     1   IDDNUN,IPEROC,ITSOC,CHEDFM,CDDNFM,IBDOPT,LBHDSV,LBDDSV,IFREFM)
C
C-----VERSION 1520 20FEB1996 SBAS5I
C     ******************************************************************
C     SET UP OUTPUT CONTROL.
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION IOFLG(NLAY,4)
      CHARACTER*20 CHEDFM,CDDNFM
      CHARACTER*80 LINE
C     ------------------------------------------------------------------
C
C1------ASSIGN DEFAULT VALUES.
      CHEDFM=' '
      CDDNFM=' '
      IHEDFM=0
      IDDNFM=0
      IHEDUN=0
      IDDNUN=0
      IBDOPT=1
      LBHDSV=0
      LBDDSV=0
C
C2------TEST OUTPUT CONTROL INPUT UNIT TO SEE IF OUTPUT CONTROL IS
C2------ACTIVE.
      IF(INOC.LE.0) THEN
C
C2A-----OUTPUT CONTROL IS INACTIVE. PRINT A MESSAGE LISTING DEFAULTS.
         WRITE(IOUT, 41)
   41    FORMAT(1X,/1X,'DEFAULT OUTPUT CONTROL',/1X,
     1  'THE FOLLOWING OUTPUT COMES AT THE END OF EACH STRESS PERIOD:')
         WRITE(IOUT, 42)
   42    FORMAT(1X,'TOTAL VOLUMETRIC BUDGET')
         WRITE(IOUT, 43)
   43    FORMAT(1X,10X,'HEAD')
         IF(ISTRT.NE.0) WRITE(IOUT, 44)
   44    FORMAT(1X,10X,'DRAWDOWN')
C
C2B-----SET DEFAULT FLAGS IN IOFLG SO THAT HEAD (AND DRAWDOWN) IS
C2B-----PRINTED FOR EVERY LAYER.
         ID=0
         IF(ISTRT.NE.0) ID=1
         DO 80 K=1,NLAY
         IOFLG(K,1)=1
         IOFLG(K,2)=ID
         IOFLG(K,3)=0
         IOFLG(K,4)=0
   80    CONTINUE
         GO TO 1000
      END IF
C
C3------OUTPUT CONTROL IS ACTIVE.  READ FIRST RECORD AND DECODE FIRST
C3------WORD.  MUST USE URWORD IN CASE FIRST WORD IS ALPHABETIC.
      READ(INOC,'(A)') LINE
      LLOC=1
      CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
C
C4------TEST FOR NUMERIC OUTPUT CONTROL.  FIRST WORD WILL NOT BE
C4------"PERIOD", "HEAD", "DRAWDOWN", OR "COMPACT".
      IF(LINE(ISTART:ISTOP).NE.'PERIOD' .AND. LINE(ISTART:ISTOP).NE.
     1     'HEAD' .AND. LINE(ISTART:ISTOP).NE.'DRAWDOWN' .AND.
     2     LINE(ISTART:ISTOP).NE.'COMPACT') THEN
C4A-----NUMERIC OUTPUT CONTROL.  DECODE THE INITIAL RECORD ACCORDINGLY.
         WRITE(IOUT,102)
  102    FORMAT(1X,/1X,'OUTPUT CONTROL IS SPECIFIED EVERY TIME STEP')
         IF(IFREFM.EQ.0) THEN
            READ(LINE,'(4I10)') IHEDFM,IDDNFM,IHEDUN,IDDNUN
         ELSE
            LLOC=1
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IHEDFM,R,IOUT,INOC)
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IDDNFM,R,IOUT,INOC)
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IHEDUN,R,IOUT,INOC)
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IDDNUN,R,IOUT,INOC)
         END IF
```

```
        WRITE(IOUT,103) IHEDFM,IDDNFM
  103   FORMAT(1X,'HEAD PRINT FORMAT CODE IS',I4,
     1     '    DRAWDOWN PRINT FORMAT CODE IS',I4)
        WRITE(IOUT,104) IHEDUN,IDDNUN
  104   FORMAT(1X,'HEADS WILL BE SAVED ON UNIT',I4,
     1     '    DRAWDOWNS WILL BE SAVED ON UNIT',I4)
        IPEROC=-1
        ITSOC=-1
      ELSE
C4B-----ALPHABETIC OUTPUT CONTROL.  CALL MODULE TO READ INITIAL RECORDS.
        CALL SBAS5J(INOC,IOUT,IHEDFM,IDDNFM,IHEDUN,IDDNUN,
     1         IPEROC,ITSOC,CHEDFM,CDDNFM,IBDOPT,LBHDSV,LBDDSV,
     2         LINE,LLOC,ISTART,ISTOP)
      END IF
C
C5------RETURN.
 1000 RETURN
      END
```

List of Variables for Module SBAS5I

Variable	Range	Definition
CDDNFM	Package	CHARACTER*20, Format that a user can specify for writing drawdown in a disk file.
CHEDFM	Package	CHARACTER*20, Format that a user can specify for writing head in a disk file.
IBDOPT	Package	Flag indicating how cell-by-cell budget data will be written: 1 - All budget terms will be written as a 3-D arrays. 2 - The form for writing budget data will be selected in order to minimize the amount of disk space.
ID	Module	Flag; 0 if ISTRT is 0, 1 if ISTRT is not 0.
IDDNFM	Package	Code that specifies the format for printing drawdown.
IDDNUN	Package	Unit number for saving drawdown in a disk file.
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
IHEDFM	Package	Code that specifies the format for printing head.
IHEDUN	Package	Unit number for saving head in a disk file.
INOC	Package	Input unit number for the Output Control Option.
IOFLG	Package	DIMENSION (NLAY,4), Flags to control printing and saving of head and drawdown for each layer: (N,1) ≠ 0, head will be printed for layer N. (N,2) ≠ 0, drawdown will be printed for layer N. (N,3) ≠ 0, head will be saved for layer N. (N,4) ≠ 0, drawdown will be saved for layer N.
IOUT	Global	Unit number for writing to the listing file.
IPEROC	Package	For alphabetic output control, the stress period at which the next output is requested. IPEROC=-1 for numeric output control.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ISTRT	Global	Flag for keeping initial heads throughout the simulation: = 0, initial heads will not be kept in memory after the simulation starts. ≠ 0, initial heads will be kept in memory throughout the simulation so that drawdown can be calculated.
ITSOC	Package	For alphabetic output control, the time step at which the next output is requested. ITSOC=-1 for numeric output control.
K	Module	Index for layers.
LBDDSV	Package	Label flag for saving drawdown in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
LBHDSV	Package	Label flag for saving head in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
LINE	Module	CHARACTER*80, A line read from the output control input file, which is scanned for option words.
LLOC	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.

N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NLAY	Global	The number of layers in the grid.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

Module SBAS5J

Narrative for Module SBAS5J

Module SBAS5J reads initial alphabetic-word records from the Output Control file. Module SBAS5J performs its functions as follows:

1. Write a message saying that the new form Output Control is being used, and set initial values for IPEROC and ITSOC. IPEROC is the stress period and ITSOC is the time step within that stress period at which output is desired. These are set to such high values that they will not result in output unless the user explicitly uses a "PERIOD" command to change these.

2. Go sequentially through the following steps to decode a record. If an error is detected, go to step 5 to print an error message.

   A. Look for "PERIOD". If found, decode IPEROC and ITSOC, and go to step 4. This is the last of the initial records.

   B. Look for "HEAD PRINT FORMAT" or "HEAD SAVE FORMAT". If found, decode the format, and go to step 3.

   C. Look for "DRAWDOWN PRINT FORMAT or "DRAWDOWN SAVE FORMAT". If found, decode the format, and go to step 3.

   D. Look for "COMPACT BUDGET FILES". Actually, only "COMPACT" is checked for because there is only one option that starts with "COMPACT". If found, set IBDOPT=2, and go to step 3.

   E. No valid command has been found, so there is an error. Go to step 5.

3. A record has been decoded. Read another record, get the first word using URWORD, and go back to step 2 to process the record.

4. RETURN.

5. There has been an error decoding records. Print an error message and STOP.

SBAS5J

```
      SUBROUTINE SBAS5J(INOC,IOUT,IHEDFM,IDDNFM,IHEDUN,IDDNUN,
     1         IPEROC,ITSOC,CHEDFM,CDDNFM,IBDOPT,LBHDSV,LBDDSV,
     2         LINE,LLOC,ISTART,ISTOP)
C
C-----VERSION 1433 3JAN1995 SBAS5J
C     ****************************************************************
C     READ INITIAL ALPHABETIC OUTPUT CONTROL RECORDS.
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      CHARACTER*20 CHEDFM,CDDNFM
      CHARACTER*80 LINE
C     ----------------------------------------------------------------
C
C1------ALPHABETIC OUTPUT CONTROL.  WRITE MESSAGE AND SET INITIAL VALUES
C1------FOR IPEROC AND ITSOC.
      WRITE(IOUT,91)
   91 FORMAT(1X,/1X,'OUTPUT CONTROL IS SPECIFIED ONLY AT TIME STEPS',
     1    ' FOR WHICH OUTPUT IS DESIRED')
      IPEROC=9999
      ITSOC=9999
C
C2------LOOK FOR ALPHABETIC WORDS:
C2A-----LOOK FOR "PERIOD", WHICH INDICATES THE END OF INITIAL OUTPUT
C2A-----CONTROL DATA.  IF FOUND, DECODE THE PERIOD NUMBER AND TIME
C2A-----STEP NUMBER FOR LATER USE.
  100 IF(LINE(ISTART:ISTOP).EQ.'PERIOD') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IPEROC,R,IOUT,INOC)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
         IF(LINE(ISTART:ISTOP).NE.'STEP') GO TO 2000
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,ITSOC,R,IOUT,INOC)
         WRITE(IOUT,101) IHEDFM,IDDNFM
  101    FORMAT(1X,'HEAD PRINT FORMAT CODE IS',I4,
     1        '    DRAWDOWN PRINT FORMAT CODE IS',I4)
         WRITE(IOUT,102) IHEDUN,IDDNUN
  102    FORMAT(1X,'HEADS WILL BE SAVED ON UNIT',I4,
     1        '    DRAWDOWNS WILL BE SAVED ON UNIT',I4)
         GO TO 1000
C
C2B-----LOOK FOR "HEAD PRINT FORMAT" AND "HEAD SAVE FORMAT".  IF
C2B-----FOUND, SET APPROPRIATE FLAGS.
      ELSE IF(LINE(ISTART:ISTOP).EQ.'HEAD') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
         IF(LINE(ISTART:ISTOP).EQ.'PRINT') THEN
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
            IF(LINE(ISTART:ISTOP).NE.'FORMAT') GO TO 2000
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IHEDFM,R,IOUT,INOC)
         ELSE IF(LINE(ISTART:ISTOP).EQ.'SAVE') THEN
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
            IF(LINE(ISTART:ISTOP).EQ.'UNIT') THEN
               CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IHEDUN,R,IOUT,
     1            INOC)
            ELSE IF(LINE(ISTART:ISTOP).EQ.'FORMAT') THEN
               CALL URWORD(LINE,LLOC,ISTART,ISTOP,0,N,R,IOUT,INOC)
               CHEDFM=LINE(ISTART:ISTOP)
               WRITE(IOUT,103) CHEDFM
  103          FORMAT(1X,'HEADS WILL BE SAVED WITH FORMAT: ',A)
               CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
               IF(LINE(ISTART:ISTOP).EQ.'LABEL') THEN
                  LBHDSV=1
                  WRITE(IOUT,104)
  104             FORMAT(1X,'SAVED HEADS WILL BE LABELED')
               END IF
            ELSE
               GO TO 2000
            END IF
         ELSE
            GO TO 2000
         END IF
C
C2C-----LOOK FOR "DRAWDOWN PRINT FORMAT" AND "DRAWDOWN SAVE FORMAT".
C2C-----IF FOUND, SET APPROPRIATE FLAGS
      ELSE IF(LINE(ISTART:ISTOP).EQ.'DRAWDOWN') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
         IF(LINE(ISTART:ISTOP).EQ.'PRINT') THEN
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
```

```
            IF(LINE(ISTART:ISTOP).NE.'FORMAT') GO TO 2000
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IDDNFM,R,IOUT,INOC)
         ELSE IF(LINE(ISTART:ISTOP).EQ.'SAVE') THEN
            CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
            IF(LINE(ISTART:ISTOP).EQ.'UNIT') THEN
               CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IDDNUN,R,IOUT,
     1                     INOC)
            ELSE IF(LINE(ISTART:ISTOP).EQ.'FORMAT') THEN
               CALL URWORD(LINE,LLOC,ISTART,ISTOP,0,N,R,IOUT,INOC)
               CDDNFM=LINE(ISTART:ISTOP)
               WRITE(IOUT,105) CDDNFM
  105          FORMAT(1X,'DRAWDOWN WILL BE SAVED WITH FORMAT: ',A)
               CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
               IF(LINE(ISTART:ISTOP).EQ.'LABEL') THEN
                  LBDDSV=1
                  WRITE(IOUT,106)
  106             FORMAT(1X,'SAVED DRAWDOWN WILL BE LABELED')
               END IF
            ELSE
               GO TO 2000
            END IF
         ELSE
            GO TO 2000
         END IF
C
C2D-----LOOK FOR "COMPACT BUDGET FILES" -- "COMPACT" IS SUFFICIENT.
C2D-----IF FOUND, SET APPROPRIATE FLAG.
      ELSE IF(LINE(ISTART:ISTOP).EQ.'COMPACT') THEN
         IBDOPT=2
         WRITE(IOUT,107)
  107    FORMAT(1X,'COMPACT CELL-BY-CELL BUDGET FILES WILL BE WRITTEN')
C
C2E-----ERROR IF UNRECOGNIZED WORD.
      ELSE
         GO TO 2000
      END IF
C
C3------FINISHED READING A RECORD.  READ NEXT RECORD, IGNORING BLANK
C3------LINES.  GO BACK AND DECODE IT.
  110 READ(INOC,'(A)',END=1000) LINE
      IF(LINE.EQ.' ') GO TO 110
      LLOC=1
      CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
      GO TO 100
C
C4------RETURN.
 1000 RETURN
C
C5------ERROR DECODING INPUT DATA.
 2000 WRITE(IOUT,2001) LINE
 2001 FORMAT(1X,/1X,'ERROR READING OUTPUT CONTROL INPUT DATA:'/1X,A80)
      STOP
      END
```

List of Variables for Module SBAS5J

Variable	Range	Definition
CDDNFM	Package	CHARACTER*20, Format that a user can specify for writing drawdown in a disk file.
CHEDFM	Package	CHARACTER*20, Format that a user can specify for writing head in a disk file.
IBDOPT	Package	Flag indicating how cell-by-cell budget data will be written: 1 - All budget terms will be written as a 3-D arrays. 2 - The form for writing budget data will be selected in order to minimize the amount of disk space.
IDDNFM	Package	Code that specifies the format for printing drawdown.
IDDNUN	Package	Unit number for saving drawdown in a disk file.
IHEDFM	Package	Code that specifies the format for printing head.
IHEDUN	Package	Unit number for saving head in a disk file.
INOC	Package	Input unit number for the Output Control Option.
IOUT	Global	Unit number for writing to the listing file.
IPEROC	Package	For alphabetic output control, the stress period at which the next output is requested. IPEROC=-1 for numeric output control.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ITSOC	Package	For alphabetic output control, the time step at which the next output is requested. ITSOC=-1 for numeric output control.
LBDDSV	Package	Label flag for saving drawdown in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
LBHDSV	Package	Label flag for saving head in a formatted file. = 0, do not write a label in the file. ≠ 0, write a label in the file.
LINE	Module	CHARACTER*80, A line read from the output control input file, which is scanned for option words.
LLOC	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

Module SBAS5T

```
      SUBROUTINE SBAS5T(KSTP,KPER,DELT,PERTIM,TOTIM,ITMUNI,IOUT)
C
C
C-----VERSION 0959 22JUNE1992 SBAS5T
C     ******************************************************************
C     PRINT SIMULATION TIME
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
      WRITE(IOUT,199) KSTP,KPER
  199 FORMAT(1X,///10X,'TIME SUMMARY AT END OF TIME STEP',I3,
     1     ' IN STRESS PERIOD',I3)
C
C1------USE TIME UNIT INDICATOR TO GET FACTOR TO CONVERT TO SECONDS.
      ZERO=0.
      CNV=ZERO
      IF(ITMUNI.EQ.1) CNV=1.
      IF(ITMUNI.EQ.2) CNV=60.
      IF(ITMUNI.EQ.3) CNV=3600.
      IF(ITMUNI.EQ.4) CNV=86400.
      IF(ITMUNI.EQ.5) CNV=31557600.
C
C2------IF FACTOR=0 THEN TIME UNITS ARE NON-STANDARD.
      IF(CNV.NE.ZERO) GO TO 100
C
C2A-----PRINT TIMES IN NON-STANDARD TIME UNITS.
      WRITE(IOUT,301) DELT,PERTIM,TOTIM
  301 FORMAT(21X,'     TIME STEP LENGTH =',G15.6/
     1       21X,'   STRESS PERIOD TIME =',G15.6/
     2       21X,'TOTAL SIMULATION TIME =',G15.6)
C
C2B-----RETURN
      RETURN
C
C3------CALCULATE LENGTH OF TIME STEP & ELAPSED TIMES IN SECONDS.
  100 DELSEC=CNV*DELT
      TOTSEC=CNV*TOTIM
      PERSEC=CNV*PERTIM
C
C4------CALCULATE TIMES IN MINUTES,HOURS,DAYS AND YEARS.
      SIXTY=60.
      HRDAY=24.
      DAYYR=365.25
      DELMN=DELSEC/SIXTY
      DELHR=DELMN/SIXTY
      DELDY=DELHR/HRDAY
      DELYR=DELDY/DAYYR
      TOTMN=TOTSEC/SIXTY
      TOTHR=TOTMN/SIXTY
      TOTDY=TOTHR/HRDAY
      TOTYR=TOTDY/DAYYR
      PERMN=PERSEC/SIXTY
      PERHR=PERMN/SIXTY
      PERDY=PERHR/HRDAY
      PERYR=PERDY/DAYYR
C
C5------PRINT TIME STEP LENGTH AND ELAPSED TIMES IN ALL TIME UNITS.
      WRITE(IOUT,200)
  200 FORMAT(19X,' SECONDS     MINUTES      HOURS',7X,
     1    'DAYS        YEARS'/20X,59('-'))
      WRITE (IOUT,201) DELSEC,DELMN,DELHR,DELDY,DELYR
  201 FORMAT(1X,'  TIME STEP LENGTH',1P,5G12.5)
      WRITE(IOUT,202) PERSEC,PERMN,PERHR,PERDY,PERYR
  202 FORMAT(1X,'STRESS PERIOD TIME',1P,5G12.5)
      WRITE(IOUT,203) TOTSEC,TOTMN,TOTHR,TOTDY,TOTYR
  203 FORMAT(1X,'         TOTAL TIME',1P,5G12.5)
C
C6------RETURN
      RETURN
      END
```

Module SBAS5V

Narrative for Module SBAS5V

Module SBAS5V prints the total volumetric budget for a model simulation. The individual components of the budget are calculated by component-of-flow packages and passed to SBAS5V in array VBVL. For each budget component there are four values: rates in and out for the current time step, and cumulative volumes in and out for the entire simulation up to the current time. SBAS5V calculates total inflow and total outflow, and prints the percent discrepancy. SBAS5V performs its functions as follows:

1. Calculate the number of budget components. MSUM contains the next unused location in the VBVL array, so there are MSUM-1 actual budget components.

2. Clear the four accumulators for total rates and volumes.

3. Accumulate the total rates and volumes by summing over all budget components.

4. Print a title, which includes the stress period and time step.

5. Print all the inflow components followed by the total inflow. In order to make it easier to compare magnitudes of values, print the numbers with an aligned decimal point where possible. The biggest value that can fit in the F17.4 format is: `999999999999.9999`
When values cannot be printed with sufficient accuracy using this format, use an exponential format used -- format (1PE17.4).

6. Print all the outflow components followed by the total outflow.

7. Calculate the difference between inflow and outflow and the percent difference.

   A. Calculate the difference between inflow and outflow rates and the absolute value of the difference. The absolute value is used in step 8 in order to determine the format for printing the value.

   B. Calculate the difference in rates as a percent of the average rate.

   C. Calculate the difference between inflow and outflow volumes and the absolute value of the difference. The absolute value is used in step 8 in order to determine the format for printing the value.

   D. Calculate the difference in volumes as a percent of the average volume.

8. Print differences and percent discrepancies. Differences can be negative, so space for the negative sign must be allowed for in the format. The biggest magnitude value that can fit in the F17.4 format is: `-99999999999.9999`

9. RETURN.

SBAS5V

```
      SUBROUTINE SBAS5V(MSUM,VBNM,VBVL,KSTP,KPER,IOUT)
C
C
C-----VERSION 1448 28APRIL1994 SBAS5V
C     ******************************************************************
C     PRINT VOLUMETRIC BUDGET
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM)
      DIMENSION VBVL(4,MSUM)
      CHARACTER*17 VAL1,VAL2
C     ------------------------------------------------------------------
C
C1------DETERMINE NUMBER OF INDIVIDUAL BUDGET ENTRIES.
      MSUM1=MSUM-1
      IF(MSUM1.LE.0) RETURN
C
C2------CLEAR RATE AND VOLUME ACCUMULATORS.
      ZERO=0.
      TWO=2.
      HUND=100.
      BIGVL1=9.99999E11
      BIGVL2=9.99999E10
      SMALL=0.1
      TOTRIN=ZERO
      TOTROT=ZERO
      TOTVIN=ZERO
      TOTVOT=ZERO
C
C3------ADD RATES AND VOLUMES (IN AND OUT) TO ACCUMULATORS.
      DO 100 L=1,MSUM1
      TOTRIN=TOTRIN+VBVL(3,L)
      TOTROT=TOTROT+VBVL(4,L)
      TOTVIN=TOTVIN+VBVL(1,L)
      TOTVOT=TOTVOT+VBVL(2,L)
  100 CONTINUE
C
C4------PRINT TIME STEP NUMBER AND STRESS PERIOD NUMBER.
      WRITE(IOUT,260) KSTP,KPER
      WRITE(IOUT,265)
C
C5------PRINT INDIVIDUAL INFLOW RATES AND VOLUMES AND THEIR TOTALS.
      DO 200 L=1,MSUM1
      IF(VBVL(1,L).NE.ZERO .AND.
     1       (VBVL(1,L).GE.BIGVL1 .OR. VBVL(1,L).LT.SMALL)) THEN
         WRITE(VAL1,'(1PE17.4)') VBVL(1,L)
      ELSE
         WRITE(VAL1,'(F17.4)') VBVL(1,L)
      END IF
      IF(VBVL(3,L).NE.ZERO .AND.
     1       (VBVL(3,L).GE.BIGVL1 .OR. VBVL(3,L).LT.SMALL)) THEN
         WRITE(VAL2,'(1PE17.4)') VBVL(3,L)
      ELSE
         WRITE(VAL2,'(F17.4)') VBVL(3,L)
      END IF
      WRITE(IOUT,275) VBNM(L),VAL1,VBNM(L),VAL2
  200 CONTINUE
      IF(TOTVIN.NE.ZERO .AND.
     1      (TOTVIN.GE.BIGVL1 .OR. TOTVIN.LT.SMALL)) THEN
         WRITE(VAL1,'(1PE17.4)') TOTVIN
      ELSE
         WRITE(VAL1,'(F17.4)') TOTVIN
      END IF
      IF(TOTRIN.NE.ZERO .AND.
     1      (TOTRIN.GE.BIGVL1 .OR. TOTRIN.LT.SMALL)) THEN
         WRITE(VAL2,'(1PE17.4)') TOTRIN
      ELSE
         WRITE(VAL2,'(F17.4)') TOTRIN
      END IF
      WRITE(IOUT,286) VAL1,VAL2
C
C6------PRINT INDIVIDUAL OUTFLOW RATES AND VOLUMES AND THEIR TOTALS.
      WRITE(IOUT,287)
      DO 250 L=1,MSUM1
      IF(VBVL(2,L).NE.ZERO .AND.
```

```
     1       (VBVL(2,L).GE.BIGVL1 .OR. VBVL(2,L).LT.SMALL)) THEN
           WRITE(VAL1,'(1PE17.4)') VBVL(2,L)
         ELSE
           WRITE(VAL1,'(F17.4)') VBVL(2,L)
         END IF
         IF(VBVL(4,L).NE.ZERO .AND.
     1       (VBVL(4,L).GE.BIGVL1 .OR. VBVL(4,L).LT.SMALL)) THEN
           WRITE(VAL2,'(1PE17.4)') VBVL(4,L)
         ELSE
           WRITE(VAL2,'(F17.4)') VBVL(4,L)
         END IF
         WRITE(IOUT,275) VBNM(L),VAL1,VBNM(L),VAL2
  250 CONTINUE
         IF(TOTVOT.NE.ZERO .AND.
     1      (TOTVOT.GE.BIGVL1 .OR. TOTVOT.LT.SMALL)) THEN
           WRITE(VAL1,'(1PE17.4)') TOTVOT
         ELSE
           WRITE(VAL1,'(F17.4)') TOTVOT
         END IF
         IF(TOTROT.NE.ZERO .AND.
     1      (TOTROT.GE.BIGVL1 .OR. TOTROT.LT.SMALL)) THEN
           WRITE(VAL2,'(1PE17.4)') TOTROT
         ELSE
           WRITE(VAL2,'(F17.4)') TOTROT
         END IF
         WRITE(IOUT,298) VAL1,VAL2
C
C7------CALCULATE THE DIFFERENCE BETWEEN INFLOW AND OUTFLOW.
C
C7A-----CALCULATE DIFFERENCE BETWEEN RATE IN AND RATE OUT.
         DIFFR=TOTRIN-TOTROT
         ADIFFR=ABS(DIFFR)
C
C7B-----CALCULATE PERCENT DIFFERENCE BETWEEN RATE IN AND RATE OUT.
         PDIFFR=ZERO
         AVGRAT=(TOTRIN+TOTROT)/TWO
         IF(AVGRAT.NE.ZERO) PDIFFR=HUND*DIFFR/AVGRAT
C
C7C-----CALCULATE DIFFERENCE BETWEEN VOLUME IN AND VOLUME OUT.
         DIFFV=TOTVIN-TOTVOT
         ADIFFV=ABS(DIFFV)
C
C7D-----GET PERCENT DIFFERENCE BETWEEN VOLUME IN AND VOLUME OUT.
         PDIFFV=ZERO
         AVGVOL=(TOTVIN+TOTVOT)/TWO
         IF(AVGVOL.NE.ZERO) PDIFFV=HUND*DIFFV/AVGVOL
C
C8------PRINT DIFFERENCES AND PERCENT DIFFERENCES BETWEEN INPUT
C8------AND OUTPUT RATES AND VOLUMES.
         IF(ADIFFV.NE.ZERO .AND.
     1      (ADIFFV.GE.BIGVL2 .OR. ADIFFV.LT.SMALL)) THEN
           WRITE(VAL1,'(1PE17.4)') DIFFV
         ELSE
           WRITE(VAL1,'(F17.4)') DIFFV
         END IF
         IF(ADIFFR.NE.ZERO .AND.
     1      (ADIFFR.GE.BIGVL2 .OR. ADIFFR.LT.SMALL)) THEN
           WRITE(VAL2,'(1PE17.4)') DIFFR
         ELSE
           WRITE(VAL2,'(F17.4)') DIFFR
         END IF
         WRITE(IOUT,299) VAL1,VAL2
         WRITE(IOUT,300) PDIFFV,PDIFFR
C
C9------RETURN.
         RETURN
C
C    ---FORMATS
C
  260 FORMAT('1',/2X,'VOLUMETRIC BUDGET FOR ENTIRE MODEL AT END OF'
     1,' TIME STEP',I3,' IN STRESS PERIOD',I3/2X,77('-'))
  265 FORMAT(1X,/5X,'CUMULATIVE VOLUMES',6X,'L**3',7X
     1,'RATES FOR THIS TIME STEP',6X,'L**3/T'/5X,18('-'),17X,24('-')
     2//11X,'IN:',38X,'IN:'/11X,'---',38X,'---')
  275 FORMAT(1X,3X,A16,' =',A17,6X,A16,' =',A17)
  286 FORMAT(1X,/12X,'TOTAL IN =',A,14X,'TOTAL IN =',A)
  287 FORMAT(1X,/10X,'OUT:',37X,'OUT:'/10X,4('-'),37X,4('-'))
  298 FORMAT(1X,/11X,'TOTAL OUT =',A,13X,'TOTAL OUT =',A)
  299 FORMAT(1X,/12X,'IN - OUT =',A,14X,'IN - OUT =',A)
  300 FORMAT(1X,/1X,'PERCENT DISCREPANCY =',F15.2
```

```
     1,5X,'PERCENT DISCREPANCY =',F15.2,///)
C
      END
```

List of Variables for Module SBAS5V

Variable	Range	Definition
ADIFFR	Module	The absolute value of DIFFR.
ADIFFV	Module	The absolute value of DIFFV.
AVGRAT	Module	The average of the sum of all inflow rates and the sum of all outflow rates.
AVGVOL	Module	The average of the sum of all inflow volumes and the sum of all outflow volumes.
BIGVL1	Module	The constant 9.99999E11.
BIGVL2	Module	The constant 9.99999E10.
DIFFR	Module	The sum of all inflow rates minus the sum of all outflow rates.
DIFFV	Module	The sum of all inflow volumes minus the sum of all outflow volumes.
HUND	Module	The constant 100.
IOUT	Global	Unit number for writing to the listing file.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
L	Module	Index for flow terms.
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.
MSUM1	Module	MSUM - 1.
PDIFFR	Module	DIFFR as a percent of AVGRAT.
PDIFFV	Module	DIFFV as a percent of AVGVOL.
SMALL	Module	The constant 0.1.
TOTRIN	Module	Accumulator for the inflow rates.
TOTROT	Module	Accumulator for the outflow rates.
TOTVIN	Module	Accumulator for the inflow volumes.
TOTVOT	Module	Accumulator for the outflow volumes.
TWO	Module	The constant 2.
VAL1	Module	CHARACTER*17, Character
VAL2	Module	CHARACTER*17,
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
ZERO	Module	The constant 0.

Module SBAS5N

Narrative for Module SBAS5N

Module SBAS5N sets output flags for time steps if the alphabetic-word method of Output Control is being used. SBAS5N is called by BAS5OC every time step. When the simulation time (time step and stress period) is equal to the time for which output has been requested, SBAS5N reads records from the Output Control file. The records specify various output options, and the appropriate flags are set so that module BAS5OT can output the requested data. Output times are specified by "PERIOD" records in the Output Control data. The first output time is read by SBAS5I. Subsequent output times are read by SBAS5N. Module SBAS5N performs its functions as follows:

1. Check if output time precedes simulation time. This can happen if output times are not entered in order of increasing time or if a nonexistent time step is entered for a stress period. For example, output might be entered for time step 3 of a stress period that has only two time steps. If this happens, print a message and set output time equal to the current simulation time.

2. Set all output flags to off.

3. Output time is not equal to current simulation time. Write a message stating that there will be no output and RETURN.

4. Output time is equal to current simulation time; do the following:

    A. Read a record from Output Control data. If record is blank, read another. If record is not blank, do the following:

        1. Look for "PERIOD". If found, this indicates the end of output options for this stress period. Set the new output time and RETURN.

        2. Not a "PERIOD" record; look for a "PRINT" record. If found, look for "BUDGET", "HEAD", or "DRAWDOWN". If found, set appropriate flags, and go to step 4B.. If not found, there is an error -- go to step 6.

        3. Not a "PRINT" record; look for a "SAVE" record. If found, look for "BUDGET", "HEAD", or "DRAWDOWN". If found, set appropriate flags, and go to step 4B. If not found, there is an error -- go to step 6.

        4. No recognized command was found, so there is an error. Go to step 6.

    B. Go back to step 4A, and read and process another record.

5. An end of file was found when reading Ouput Control data. Thus, there can be no more output after the current time step. Set the output time to stress period 9999 and time step 9999 so that the simulation time will never reach the output time.

6. Error when decoding Output Control data. Print a messaage and STOP.

SBAS5N

```
      SUBROUTINE SBAS5N(IOFLG,NLAY,IHDDFL,IBUDFL,ICBCFL,IPEROC,ITSOC,
     1            KPER,KSTP,INOC,IOUT,IBDOPT)
C
C-----VERSION 0932 14FEB1994 SBAS5N
C     ******************************************************************
C     SET OUTPUT FLAGS USING ALPHABETIC OUTPUT CONTROL INPUT STRUCTURE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION IOFLG(NLAY,4)
      CHARACTER*80 LINE
C     ------------------------------------------------------------------
C
C1------ERROR IF OUTPUT CONTROL TIME STEP PRECEDES CURRENT SIMULATION
C1------TIME STEP.
      IF((IPEROC.LT.KPER).OR.(IPEROC.EQ.KPER .AND. ITSOC.LT.KSTP)) THEN
         WRITE(IOUT,5) IPEROC,ITSOC,KPER,KSTP
    5    FORMAT(1X,/1X,'OUTPUT CONTROL WAS SPECIFIED FOR A NONEXISTENT',
     1   ' TIME STEP',/
     2   1X,'OR OUTPUT CONTROL DATA ARE NOT ENTERED IN ASCENDING ORDER',
     3   /1X,'OUTPUT CONTROL STRESS PERIOD',I3,'   TIME STEP',I3,/
     4   1X,'MODEL STRESS PERIOD',I3,'   TIME STEP',I3,/
     5   1X,'APPLYING THE SPECIFIED OUTPUT CONTROL TO THE CURRENT TIME',
     6   ' STEP')
         IPEROC=KPER
         ITSOC=KSTP
      END IF
C
C2------CLEAR I/O FLAGS.
      IHDDFL=0
      IBUDFL=0
      ICBCFL=0
      DO 10 I=1,4
      DO 10 K=1,NLAY
      IOFLG(K,I)=0
10    CONTINUE
C
C3------IF OUTPUT CONTROL TIME STEP DOES NOT MATCH SIMULATION TIME STEP,
C3------WRITE MESSAGE THAT THERE IS NO OUTPUT CONTROL THIS TIME STEP,
C3------AND RETURN.
      IF(IPEROC.NE.KPER .OR. ITSOC.NE.KSTP) THEN
         WRITE(IOUT,11) KPER,KSTP
11       FORMAT(1X,/1X,'NO OUTPUT CONTROL FOR STRESS PERIOD',I3,
     1                '   TIME STEP',I3)
         RETURN
      END IF
C
C4------OUTPUT CONTROL TIME STEP MATCHES SIMULATION TIME STEP.
      WRITE(IOUT,12) IPEROC,ITSOC
12    FORMAT(1X,/1X,'OUTPUT CONTROL FOR STRESS PERIOD',I3,
     1               '   TIME STEP',I3)
C
C4A-----OUTPUT CONTROL MATCHES SIMULATION TIME.  READ NEXT OUTPUT
C4A-----RECORD; SKIP ANY BLANK LINES.
50    READ(INOC,'(A)',END=1000) LINE
      IF(LINE.EQ.' ') GO TO 50
C
C4A1----LOOK FOR "PERIOD", WHICH TERMINATES OUTPUT CONTROL FOR CURRENT
C4A1----TIME STEP.  IF FOUND, DECODE TIME STEP FOR NEXT OUTPUT.
      LLOC=1
      CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
      IF(LINE(ISTART:ISTOP).EQ.'PERIOD') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IPEROC,R,IOUT,INOC)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
         IF(LINE(ISTART:ISTOP).NE.'STEP') GO TO 2000
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,ITSOC,R,IOUT,INOC)
         RETURN
C
C4A2----LOOK FOR "PRINT", WHICH MAY REFER TO "BUDGET", "HEAD", OR
C4A2----"DRAWDOWN".
      ELSE IF(LINE(ISTART:ISTOP).EQ.'PRINT') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
         IF(LINE(ISTART:ISTOP).EQ.'BUDGET') THEN
            WRITE(IOUT,53)
53          FORMAT(4X,'PRINT BUDGET')
            IBUDFL=1
```

```
         ELSE IF(LINE(ISTART:ISTOP).EQ.'HEAD') THEN
            CALL SBAS5L(1,LINE,LLOC,IOFLG,NLAY,IOUT,'PRINT HEAD',
     1              INOC)
            IHDDFL=1
         ELSE IF(LINE(ISTART:ISTOP).EQ.'DRAWDOWN') THEN
            CALL SBAS5L(2,LINE,LLOC,IOFLG,NLAY,IOUT,
     1              'PRINT DRAWDOWN',INOC)
            IHDDFL=1
         ELSE
            GO TO 2000
         END IF
C
C4A3----LOOK FOR "SAVE", WHICH MAY REFER TO "BUDGET", "HEAD", OR
C4A3----"DRAWDOWN".
      ELSE IF(LINE(ISTART:ISTOP).EQ.'SAVE') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INOC)
         IF(LINE(ISTART:ISTOP).EQ.'BUDGET') THEN
            WRITE(IOUT,57)
57          FORMAT(4X,'SAVE BUDGET')
            ICBCFL=IBDOPT
         ELSE IF(LINE(ISTART:ISTOP).EQ.'HEAD') THEN
            CALL SBAS5L(3,LINE,LLOC,IOFLG,NLAY,IOUT,'SAVE HEAD',INOC)
            IHDDFL=1
         ELSE IF(LINE(ISTART:ISTOP).EQ.'DRAWDOWN') THEN
            CALL SBAS5L(4,LINE,LLOC,IOFLG,NLAY,IOUT,'SAVE DRAWDOWN',
     1          INOC)
            IHDDFL=1
         ELSE
            GO TO 2000
         END IF
C
C4A4----WHEN NO KNOWN ALPHABETIC WORDS ARE FOUND, THERE IS AN ERROR.
      ELSE
         GO TO 2000
C
C4B-----AFTER SUCCESSFULLY DECODING ONE RECORD, READ ANOTHER.
      END IF
      GO TO 50
C
C5------END OF FILE WHILE READING AN OUTPUT CONTROL RECORD, SO THERE
C5------WILL BE NO FURTHER OUTPUT.  SET IPEROC AND ITSOC HIGH ENOUGH
C5------THAT THE MODEL TIME WILL NEVER MATCH THEM.
1000  IPEROC=9999
      ITSOC=9999
      RETURN
C
C6------ERROR DECODING ALPHABETIC INPUT STRUCTURE.
2000  WRITE(IOUT,2001) LINE
2001  FORMAT(1X,/1X,'ERROR READING OUTPUT CONTROL INPUT DATA:'/1X,A80)
      STOP
      END
```

List of Variables for Module SBAS5N

Variable	Range	Definition
I	Module	Index going from 1 to 4.
IBDOPT	Package	Flag indicating how cell-by-cell budget data will be written: 1 - All budget terms will be written as a 3-D arrays. 2 - The form for writing budget data will be selected in order to minimize the amount of disk space.
IBUDFL	Package	Volumetric budget print flag: = 0, volumetric budget is not printed for the current time step. ≠ 0, volumetric budget is printed for the current time step.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IHDDFL	Package	Flag for using IOFLG in the current time step: = 0, regardless of IOFLG values, no head or drawdown values are printed or saved. ≠ 0, IOFLG values are used to determine if head and drawdown are printed and saved.
INOC	Package	Input unit number for the Output Control Option.
IOFLG	Package	DIMENSION (NLAY,4), Flags to control printing and saving of head and drawdown for each layer.
IOUT	Global	Unit number for writing to the listing file.
IPEROC	Package	For alphabetic output control, the stress period at which the next output is requested. IPEROC=-1 for numeric output control.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ITSOC	Package	For alphabetic output control, the time step at which the next output is requested. ITSOC=-1 for numeric output control.
K	Module	Index for layers.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
LINE	Module	CHARACTER*80, A line read from the output control input file, which is scanned for option words.
LLOC	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NLAY	Global	The number of layers in the grid.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

Module SBAS5L

Narrative for Module SBAS5L

When using alphabetic-word Output Control, Module SBAS5L decodes layer numbers for the commands for printing and saving drawdown and head. The user can specify a list of layers, or no specific layers can be specified. If no layers are specified, then all layers are assumed. Module SBAS5L performs its functions as follows:

1. Initialize the counter for the number of layers for which output is specified (NSET) to 0.

2. When SBAS5L is called, a command to print or save drawdown or head has been read and partially parsed. The layer numbers are the only remaining part to be parsed. Call URWORD to continue parsing the command in order to attempt to get an integer layer number. Test that the returned number is within the range of valid layer numbers. If so, keep track of how many layers have been specified, set the appropriate IOFLG value, and try to get another layer number. If the layer number is not valid, continue to step 3. URWORD will return 0 as a layer number if it finds a word, such as a comment, that is not an integer.

3. All valid layer numbers have been obtained from the record. Test to see if there were any layers specified. If no layers were specifieded, then all layers are assumed, and IOFLG for all layers is set. Write a message to this effect.

4. If one or more individual layers were specified, then write the layer numbers.

5. RETURN.

SBAS5L

```
      SUBROUTINE SBAS5L(IPOS,LINE,LLOC,IOFLG,NLAY,IOUT,LABEL,INOC)
C
C
C-----VERSION 1453 14FEB1994 SBAS5L
C     ****************************************************************
C     WHEN USING ALPHABETIC OUTPUT CONTROL, DECODE LAYER
C     NUMBERS FOR PRINTING OR SAVING HEAD OR DRAWDOWN
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      DIMENSION IOFLG(NLAY,4)
      CHARACTER*80 LINE
      CHARACTER*(*) LABEL
      DIMENSION LAYER(200)
C     ----------------------------------------------------------------
C
C1------INITIALIZE COUNTER FOR NUMBER OF LAYERS FOR WHICH OUTPUT IS
C1------SPECIFIED.
      NSET=0
C
C2------CHECK FOR A VALID LAYER NUMBER.  WHEN FOUND, SET FLAG AND
C2------REPEAT.
10    CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,L,R,-1,INOC)
      IF(L.GT.0 .AND. L.LE.NLAY) THEN
         NSET=NSET+1
         LAYER(NSET)=L
         IOFLG(L,IPOS)=1
         GO TO 10
      END IF
C
C3------DONE CHECKING FOR LAYER NUMBERS.  IF NO LAYER NUMBERS WERE
C3------FOUND, SET FLAGS FOR ALL LAYERS.
      IF(NSET.EQ.0) THEN
         DO 110 K=1,NLAY
         IOFLG(K,IPOS)=1
110      CONTINUE
         WRITE(IOUT,111) LABEL
111      FORMAT(4X,A,' FOR ALL LAYERS')
C
C4------IF ONE OR MORE LAYER NUMBERS WERE FOUND, PRINT THE NUMBERS.
      ELSE
         WRITE(IOUT,112) LABEL,(LAYER(M),M=1,NSET)
112      FORMAT(4X,A,' FOR LAYERS:',(1X,15I3))
      END IF
C
C5------RETURN.
      RETURN
      END
```

List of Variables for Module SBAS5L

Variable	Range	Definition
INOC	Package	Input unit number for the Output Control Option.
IOFLG	Package	DIMENSION (NLAY,4), Flags to control printing and saving of head and drawdown for each layer: (N,1) $\neq 0$, head will be printed for layer N. (N,2) $\neq 0$, drawdown will be printed for layer N. (N,3) $\neq 0$, head will be saved for layer N. (N,4) $\neq 0$, drawdown will be saved for layer N.
IOUT	Global	Unit number for writing to the listing file.
IPOS	Module	Integer code passed to SBAS5L that is the 2nd index to IOFLG, which must have a value in the range 1 to 4.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
K	Module	Index for layers.
L	Module	Integer value returned by Module URWORD, which represents a layer number.
LABEL	Module	CHARACTER*(*), Label for the kind of output that is being processed by SBAS5L.
LAYER	Module	DIMENSION (200), List of layers for which output is desired.
LINE	Module	CHARACTER*80, A line that has been read from the output control input file and is passed to SBAS5L. SBAS5L scans this line for layer numbers.
LLOC	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.
M	Module	Index within array LAYER.
NLAY	Global	The number of layers in the grid.
NSET	Module	The number of layers for which output is specified.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

Module SBAS5O

Narrative for Module SBAS5O

Module SBAS5O opens files for MODFLOW. The only file that SBAS5O does not open is the file, called the name file, that contains the names of other files. The name file must be opened prior to entering SBAS5O; the MAIN program opens the name file. SBAS5O also sets values of the IUNIT array. The IUNIT array specifies which major options are active and the unit numbers from which their data are read. Thus, to open a file for a major option, three pieces of information are required: the option name, a unit number, and a file name. Module SBAS5O performs its functions as follows:

1. Initialize data. Specifically, set all file units to 0. Also set a flag, ILIST, to 0 to indicate that the listing file is not open. Until this file is open, error messages are written to unit "*", which is a processor-determined unit that is preconnected. After the listing file is opened, all error messages are sent to that unit.

2. Read a record from the name file. If the record is blank, read another record. If column 1 contains #, the line is a comment. Print the comment line if the listing file is open, and then read the next record.

3. Parse the first two fields of the line: the file type and the unit number.

4. Check for a valid file type as follows:

   A. The first entry in the name file must be for type "LIST". Check for this type if the listing file is not yet open. If the first entry is not file type "LIST", print a message and stop.

   B. Not the first entry; check for file type "BAS".

   C. Not "BAS"; check for type "UNFORMATTED", which indicates an unformatted file that is not a file for a major option.

   D. Not "UNFORMATTED"; check for type "FORMATTED", which indicates a formatted file that is not a file for a major option.

   E. Not "FORMATTED"; check for a major option. If not a major option, print an error message and STOP.

5. A valid file type has been found. Determine file name and the access method (direct or sequential). Unless this is the listing file, write a message stating the name of the file being opened. Open the file.

6. If the file is the listing file, it is now OK to go write the listing file's name in the listing file. Go back to step 2 to process the next record in the name file.

7. The end of the name file has been reached. Check to be sure that at least a listing file and the "BAS" file have been opened. If so, close the name file and RETURN. If not, print a message and STOP.

SBAS5O

```
      SUBROUTINE SBAS5O(INUNIT,INBAS,IOUT,IUNIT,CUNIT)
C
C-----VERSION 0943 18MAR1996 SBAS5O
C     *****************************************************************
C     OPEN FILES.
C     *****************************************************************
C
C        SPECIFICATIONS:
C     -----------------------------------------------------------------
      DIMENSION IUNIT(40)
      CHARACTER*4 CUNIT(40)
      CHARACTER*80 LINE
      CHARACTER*11 FMTARG
C     -----------------------------------------------------------------
C
C1------INITIALIZE CONSTANTS.  ILIST IS SET TO 1 ONCE THE LISTING
C1------FILE HAS BEEN OPENED; UNTIL THEN ERROR MESSAGES ARE WRITTEN
C1------ TO "*" UNIT.
      INBAS=0
      IOUT=0
      ILIST=0
      DO 5 I=1,40
      IUNIT(I)=0
5     CONTINUE
C
C2------READ A LINE; IGNORE BLANK LINES AND PRINT COMMENT LINES.
10    READ(INUNIT,'(A)',END=1000) LINE
      IF(LINE.EQ.' ') GO TO 10
      IF(LINE(1:1).EQ.'#') THEN
        IF(ILIST.NE.0) WRITE(IOUT,'(A)') LINE
        GO TO 10
      END IF
C
C3------DECODE THE FILE TYPE AND UNIT NUMBER.
      LLOC=1
      CALL URWORD(LINE,LLOC,ITYP1,ITYP2,1,N,R,IOUT,INUNIT)
      CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IU,R,IOUT,INUNIT)
C
C4------CHECK FOR A VALID FILE TYPE.
      FMTARG='FORMATTED'
C
C4A-----FIRST ENTRY MUST BE FILE-TYPE "LIST".
      IF(ILIST.EQ.0) THEN
         IF(LINE(ITYP1:ITYP2).NE.'LIST') THEN
            WRITE(*,*) ' FIRST ENTRY IN NAME FILE MUST BE "LIST".'
            STOP
         END IF
         IOUT=IU
C
C4B-----CHECK FOR "BAS" FILE TYPE.
      ELSE IF(LINE(ITYP1:ITYP2).EQ.'BAS') THEN
         INBAS=IU
C
C4C-----CHECK FOR "UNFORMATTED" FILE TYPE.
      ELSE IF(LINE(ITYP1:ITYP2).EQ.'DATA(BINARY)') THEN
         FMTARG='UNFORMATTED'
C
C4D-----CHECK FOR "FORMATTED FILE TYPE.
      ELSE IF(LINE(ITYP1:ITYP2).EQ.'DATA') THEN
         FMTARG='FORMATTED'
C
C4E-----CHECK FOR MAJOR OPTIONS.
      ELSE
         DO 20 I=1,40
            IF(LINE(ITYP1:ITYP2).EQ.CUNIT(I)) THEN
               IUNIT(I)=IU
               GO TO 30
            END IF
20       CONTINUE
         WRITE(IOUT,21) LINE(ITYP1:ITYP2)
21       FORMAT(1X,'ILLEGAL FILE TYPE IN NAME FILE: ',A)
         STOP
30       CONTINUE
      END IF
C
C5------DETERMINE FILE NAME AND THE ACCESS METHOD (DIRECT OR
C5------SEQUENTIAL).  WRITE THE FILE NAME IF THE FILE IS NOT THE
```

```
C5------LISTING FILE.  THEN OPEN THE FILE.
      CALL URWORD(LINE,LLOC,INAM1,INAM2,0,N,R,IOUT,INUNIT)
      CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,INUNIT)
      IF(LINE(ISTART:ISTOP).EQ.'DIRECT') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IRECL,R,IOUT,INUNIT)
         IF(ILIST.NE.0) WRITE(IOUT,35) LINE(INAM1:INAM2),
     1        LINE(ITYP1:ITYP2),IU,IRECL
35       FORMAT(1X,/1X,'OPENING ',A,/
     1     1X,'FILE TYPE:',A,'   UNIT',I4,'   DIRECT ACCESS',I10)
         OPEN(UNIT=IU,FILE=LINE(INAM1:INAM2),FORM=FMTARG,
     1         ACCESS='DIRECT',RECL=IRECL)
      ELSE
         IF(ILIST.NE.0) WRITE(IOUT,36) LINE(INAM1:INAM2),
     1        LINE(ITYP1:ITYP2),IU
36       FORMAT(1X,/1X,'OPENING ',A,/
     1     1X,'FILE TYPE:',A,'   UNIT',I4)
         OPEN(UNIT=IU,FILE=LINE(INAM1:INAM2),FORM=FMTARG,
     1         ACCESS='SEQUENTIAL')
      END IF
C
C6------IF THE OPENED FILE IS THE LISTING FILE, WRITE ITS NAME.
C6------GO BACK AND READ NEXT RECORD.
      IF(ILIST.EQ.0) WRITE(IOUT,37) LINE(INAM1:INAM2),IU
37    FORMAT(1X,'LISTING FILE: ',A,/25X,'UNIT',I4)
      ILIST=1
      GO TO 10
C
C7------END OF NAME FILE.  RETURN PROVIDED THAT LISTING FILE AND BAS
C7------FILES HAVE BEEN OPENED.
1000  IF(ILIST.EQ.0) THEN
         WRITE(*,*) ' NAME FILE IS EMPTY.'
         STOP
      ELSE IF(INBAS.EQ.0) THEN
         WRITE(IOUT,*) ' BAS PACKAGE FILE HAS NOT BEEN OPENED.'
         STOP
      END IF
      CLOSE(UNIT=INUNIT)
      RETURN
C
      END
```

List of Variables for Module SBAS5O

Variable	Range	Definition
CUNIT	Package	CHARACTER*4(40), Names of major options corresponding to elements within the IUNIT array.
FMTARG	Module	CHARACTER*11, The value of the argument for the "FORM=" specifier in the OPEN statement for a file being opened.
I	Module	Index for IUNIT.
ILIST	Module	A flag that is set to non zero once the listing file has been opened.
INAM1	Module	Starting location within LINE of the first character of the name of the file to be opened.
INAM2	Module	Starting location within LINE of the last character of the name of the file to be opened.
INBAS	Package	Primary input unit for the Basic (BAS) Package.
INUNIT	Package	Unit for reading from the name file.
IOUT	Global	Unit number for writing to the listing file.
IRECL	Module	The record length of a direct access file.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ITYP1	Module	Starting locating within LINE of the first character of the file type.
ITYP2	Module	Starting locating within LINE of the last character of the file type.
IU	Module	The unit number of a file to be opened.
IUNIT	Package	DIMENSION(40), Primary input units for the major model options.
LINE	Module	CHARACTER*80, A line that has been read from the name file.
LLOC	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

Budget calculations have been changed to make increased use of double precision. Also, the modular structure of the budget calculations has been changed. Originally, the MAIN program made a single call to the BCF budget module (BCF1BD). This module calculated five budget terms -- flow from storage, flow from constant-head cells, and three terms for flow between adjacent cells. The budget is now calculated using three submodules that are directly called from the MAIN program. Thus, there is not a BCF5BD module. SBCF5F calculates constant-head flow, and SBCF5S calculates flow from storage. The calculations of flow between adjacent cells are in a single module, SBCF5B, which is called three times. The primary purpose for calling the submodules from the MAIN is to make the values they calculate more readily available for use by other modules that might be added to MODFLOW. Note that the calculation of flows between adjacent cells is not a part of the overall volumetric budget calculated by MODFLOW; the values are calculated solely for use elsewhere.

Modules SBCF5F and SBCF5B have the ability to calculate flow between two adjacent constant-head cells, which was previously not an option in the original MODFLOW. Although this flow is not part of MODFLOW's solution of head in a simulation, this flow may be of use in transport models.

Module BCF5AL

```
      SUBROUTINE BCF5AL(ISUM,LENX,LCSC1,LCHY,LCBOT,LCTOP,LCSC2,LCTRPY,
     1  IN,ISS,NCOL,NROW,NLAY,IOUT,IBCFCB,LCWETD,IWDFLG,LCCVWD,
     2  WETFCT,IWETIT,IHDWET,HDRY,IAPART,IFREFM)
C
C-----VERSION 1431 20FEB1996 BCF5AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR BLOCK-CENTERED FLOW PACKAGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      COMMON /FLWCOM/LAYCON(200)
      COMMON /FLWAVG/LAYAVG(200)
      CHARACTER*12 AVGNAM(4)
      DATA AVGNAM/'HARMONIC    ','ARITHMETIC  ',
     1            'LOGARITHMIC ','*UNCONFINED*'/
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE
      WRITE(IOUT,1) IN
    1 FORMAT(1X,/1X,'BCF5 -- BLOCK-CENTERED FLOW PACKAGE, VERSION 5',
     1', 9/1/93',' INPUT READ FROM UNIT',I3)
C
C2------READ AND PRINT ISS (STEADY-STATE FLAG), IBCFCB (FLAG FOR
C2------PRINTING OR UNIT# FOR RECORDING CELL-BY-CELL FLOW TERMS), HDRY
C2------(HEAD AT CELLS THAT CONVERT TO DRY), AND WETTING PARAMETERS.
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(2I10,F10.0,I10,F10.0,2I10)')
     1             ISS,IBCFCB,HDRY,IWDFLG,WETFCT,IWETIT,IHDWET
      ELSE
         READ(IN,*) ISS,IBCFCB,HDRY,IWDFLG,WETFCT,IWETIT,IHDWET
      END IF
      IF(ISS.EQ.0) WRITE(IOUT,3)
    3 FORMAT(1X,'TRANSIENT SIMULATION')
      IF(ISS.NE.0) WRITE(IOUT,4)
    4 FORMAT(1X,'STEADY-STATE SIMULATION')
      IF(IBCFCB.LT.0) WRITE(IOUT,8)
    8 FORMAT(1X,'CONSTANT-HEAD CELL-BY-CELL FLOWS WILL BE PRINTED',
     1     ' WHEN ICBCFL IS NOT 0')
      IF(IBCFCB.GT.0) WRITE(IOUT,9) IBCFCB
    9 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE SAVED ON UNIT',I3)
      WRITE(IOUT,11) HDRY
   11 FORMAT(1X,'HEAD AT CELLS THAT CONVERT TO DRY=',G13.5)
      IF(IWDFLG.NE.0) GO TO 35
      WRITE(IOUT,12)
   12 FORMAT(1X,'WETTING CAPABILITY IS NOT ACTIVE')
      GO TO 39
C
   35 WRITE(IOUT,36)
   36 FORMAT(1X,'WETTING CAPABILITY IS ACTIVE')
      IF(IWETIT.LE.0) IWETIT=1
      WRITE(IOUT,37)WETFCT,IWETIT
   37 FORMAT(1X,'WETTING FACTOR=',F10.5,
     1     '  WETTING ITERATION INTERVAL=',I4)
      WRITE(IOUT,38)IHDWET
   38 FORMAT(1X,'FLAG THAT SPECIFIES THE EQUATION TO USE FOR HEAD',
     1    ' AT WETTED CELLS=',I4)
C
C3------STOP THE SIMULATION IF THERE ARE MORE THAN 200 LAYERS.
   39 IF(NLAY.LE.200) GO TO 50
      WRITE(IOUT,41)
   41 FORMAT(1X,/1X,'YOU HAVE SPECIFIED MORE THAN 200 MODEL LAYERS'/1X,
     1 'SPACE IS RESERVED FOR A MAXIMUM OF 200 LAYERS IN ARRAYS LAYCON',
     2 ' AND LAYAVG')
      STOP
C
C4------READ LAYCON & PRINT TITLE FOR LAYCON TABLE.
   50 IF(IFREFM.EQ.0) THEN
         READ(IN,'(40I2)') (LAYCON(I),I=1,NLAY)
      ELSE
         READ(IN,*) (LAYCON(I),I=1,NLAY)
      END IF
      WRITE(IOUT,52)
   52 FORMAT(1X,5X,'LAYER  LAYER-TYPE CODE     INTERBLOCK T',
     1      /1X,5X,44('-'))
C
C5------LOOP THROUGH LAYERS CALCULATING LAYAVG, PRINTING THE LAYER-TYPE
```

```
C5------CODE, AND COUNTING LAYERS THAT NEED TOP & BOT ARRAYS.
      NBOT=0
      NTOP=0
      DO 100 I=1,NLAY
      IF(LAYCON(I).EQ.30 .OR. LAYCON(I).EQ.32) LAYCON(I)=LAYCON(I)-10
      INAM=LAYCON(I)/10
      LAYAVG(I)=INAM*10
      IF(LAYAVG(I).LT.0 .OR. LAYAVG(I).GT.30) THEN
         WRITE(IOUT,53) LAYAVG(I)
   53    FORMAT(1X,'INVALID INTERBLOCK T CODE:',I4)
         STOP
      END IF
      LAYCON(I)=LAYCON(I)-LAYAVG(I)
      L=LAYCON(I)
      INAM=INAM+1
      WRITE(IOUT,55) I,L,LAYAVG(I),AVGNAM(INAM)
   55 FORMAT(1X,I9,I13,I11,' -- ',A)
      IF(LAYCON(I).LT.0 .OR. LAYCON(I).GT.3) THEN
         WRITE(IOUT,56) LAYCON(I)
   56    FORMAT(1X,'INVALID LAYER TYPE:',I4)
         STOP
      END IF
C
C5A-----ONLY THE TOP LAYER CAN BE UNCONFINED(LAYCON=1).
      IF(L.NE.1 .OR. I.EQ.1) GO TO 70
      WRITE(IOUT,57)
   57 FORMAT(1X,/1X,'LAYER TYPE 1 IS ONLY ALLOWED IN TOP LAYER')
      STOP
C
C5B-----LAYER TYPES 1 AND 3 NEED A BOTTOM. ADD 1 TO KB.
   70 IF(L.EQ.1 .OR. L.EQ.3) NBOT=NBOT+1
C
C5C-----LAYER TYPES 2 AND 3 NEED A TOP. ADD 1 TO KT.
      IF(L.EQ.2 .OR. L.EQ.3) NTOP=NTOP+1
C
C5D-----IF LAYAVG=30, BUFF MUST BE SEPARATE FROM RHS (IAPART NOT 0).
      IF(IAPART.EQ.0 .AND. LAYAVG(I).EQ.30) THEN
         WRITE(IOUT,75)
   75    FORMAT(1X,'IAPART IN BAS PACKAGE MUST BE NONZERO',
     1      ' WHEN INTERBLOCK T IS *UNCONFINED*')
         STOP
      END IF
  100 CONTINUE
C
C
C6------COMPUTE THE NUMBER OF CELLS IN THE ENTIRE GRID AND IN ONE LAYER.
      NRC=NROW*NCOL
      ISIZ=NRC*NLAY
C
C7------ALLOCATE SPACE FOR ARRAYS.
      ISOLD=ISUM
      LCSC1=ISUM
      IF(ISS.EQ.0) ISUM=ISUM+ISIZ
      LCSC2=ISUM
      IF(ISS.EQ.0) ISUM=ISUM+NRC*NTOP
      LCTRPY=ISUM
      ISUM=ISUM+NLAY
      LCBOT=ISUM
      ISUM=ISUM+NRC*NBOT
      LCHY=ISUM
      ISUM=ISUM+NRC*NBOT
      LCTOP=ISUM
      ISUM=ISUM+NRC*NTOP
      LCWETD=ISUM
      IF(IWDFLG.NE.0)ISUM=ISUM+NRC*NBOT
      LCCVWD=ISUM
      IF(IWDFLG.NE.0)ISUM=ISUM+NRC*(NLAY-1)
C
C8------PRINT THE AMOUNT OF SPACE USED BY THE BCF PACKAGE.
      ISP=ISUM-ISOLD
      WRITE(IOUT,101) ISP
  101 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY BCF')
      ISUM1=ISUM-1
      WRITE(IOUT,102) ISUM1,LENX
  102 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,103)
  103 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C9------RETURN.
      RETURN
      END
```

Module BCF5RP

```
      SUBROUTINE BCF5RP(IBOUND,HNEW,SC1,HY,CR,CC,CV,DELR,DELC,BOT,TOP,
     1 SC2,TRPY,IN,ISS,NCOL,NROW,NLAY,IOUT,WETDRY,IWDFLG,CVWD)
C
C-----VERSION 0917 17JULY1992 BCF5RP
C     ******************************************************************
C     READ AND INITIALIZE DATA FOR BLOCK-CENTERED FLOW PACKAGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*24 ANAME(11)
      DOUBLE PRECISION HNEW
C
      DIMENSION HNEW(NCOL,NROW,NLAY),SC1(NCOL,NROW,NLAY),
     1    HY(NCOL,NROW,NLAY),CR(NCOL,NROW,NLAY),CC(NCOL,NROW,NLAY),
     2    CV(NCOL,NROW,NLAY),DELR(NCOL),DELC(NROW),BOT(NCOL,NROW,NLAY),
     3    TOP(NCOL,NROW,NLAY),SC2(NCOL,NROW,NLAY),TRPY(NLAY),
     4    IBOUND(NCOL,NROW,NLAY),WETDRY(NCOL,NROW,NLAY),
     5    CVWD(NCOL,NROW,NLAY)
C
      COMMON /FLWCOM/LAYCON(200)
C
      DATA ANAME(1) /'    PRIMARY STORAGE COEF'/
      DATA ANAME(2) /'    TRANSMIS. ALONG ROWS'/
      DATA ANAME(3) /'   HYD. COND. ALONG ROWS'/
      DATA ANAME(4) /'VERT HYD COND /THICKNESS'/
      DATA ANAME(5) /'                 BOTTOM'/
      DATA ANAME(6) /'                    TOP'/
      DATA ANAME(7) /'  SECONDARY STORAGE COEF'/
      DATA ANAME(8) /'COLUMN TO ROW ANISOTROPY'/
      DATA ANAME(9) /'                   DELR'/
      DATA ANAME(10)/'                   DELC'/
      DATA ANAME(11)/'        WETDRY PARAMETER'/
C     ------------------------------------------------------------------
C
C1------READ TRPY,DELR,DELC.
      CALL U1DREL(TRPY,ANAME(8),NLAY,IN,IOUT)
      CALL U1DREL(DELR,ANAME(9),NCOL,IN,IOUT)
      CALL U1DREL(DELC,ANAME(10),NROW,IN,IOUT)
C
C2------READ ALL PARAMETERS FOR EACH LAYER.
      KT=0
      KB=0
      DO 200 K=1,NLAY
      KK=K
C
C2A-----FIND ADDRESS OF EACH LAYER IN THREE DIMENSION ARRAYS.
      IF(LAYCON(K).EQ.1 .OR. LAYCON(K).EQ.3) KB=KB+1
      IF(LAYCON(K).EQ.2 .OR. LAYCON(K).EQ.3) KT=KT+1
C
C2B-----READ PRIMARY STORAGE COEFFICIENT INTO ARRAY SC1 IF TRANSIENT.
      IF(ISS.EQ.0)CALL U2DREL(SC1(1,1,K),ANAME(1),NROW,NCOL,KK,IN,IOUT)
C
C2C-----READ TRANSMISSIVITY INTO ARRAY CC IF LAYER TYPE IS 0 OR 2.
      IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.1) GO TO 100
      CALL U2DREL(CC(1,1,K),ANAME(2),NROW,NCOL,KK,IN,IOUT)
      GO TO 110
C
C2D-----READ HYDRAULIC CONDUCTIVITY(HY) AND BOTTOM ELEVATION(BOT)
C2D-----IF LAYER TYPE IS 1 OR 3.
  100 CALL U2DREL(HY(1,1,KB),ANAME(3),NROW,NCOL,KK,IN,IOUT)
      CALL U2DREL(BOT(1,1,KB),ANAME(5),NROW,NCOL,KK,IN,IOUT)
C
C2E-----READ VERTICAL HYCOND/THICK INTO ARRAY CV IF NOT BOTTOM LAYER;
C2E-----MULTIPLIED BY CELL AREA TO CONVERT TO CONDUCTANCE LATER.
  110 IF(K.EQ.NLAY) GO TO 120
      CALL U2DREL(CV(1,1,K),ANAME(4),NROW,NCOL,KK,IN,IOUT)
C
C2F-----READ SECONDARY STORAGE COEFFICIENT INTO ARRAY SC2 IF TRANSIENT
C2F-----AND LAYER TYPE IS 2 OR 3.
  120 IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.2) GO TO 130
      IF(ISS.EQ.0)CALL U2DREL(SC2(1,1,KT),ANAME(7),NROW,NCOL,KK,IN,IOUT)
C
C2G-----READ TOP ELEVATION(TOP) IF LAYER TYPE IS 2 OR 3.
      CALL U2DREL(TOP(1,1,KT),ANAME(6),NROW,NCOL,KK,IN,IOUT)
C
C2H-----READ WETDRY CODES IF LAYER TYPE IS 1 OR 3 AND WETTING
```

```
C2H-----CAPABILITY HAS BEEN INVOKED (IWDFLG NOT 0).
  130 IF(LAYCON(K).NE.3.AND.LAYCON(K).NE.1)GO TO 200
      IF(IWDFLG.EQ.0)GO TO 200
      CALL U2DREL(WETDRY(1,1,KB),ANAME(11),NROW,NCOL,KK,IN,IOUT)
  200 CONTINUE
C
C3------PREPARE AND CHECK BCF DATA.
      CALL SBCF5N(HNEW,IBOUND,SC1,SC2,CR,CC,CV,HY,TRPY,DELR,DELC,ISS,
     1         NCOL,NROW,NLAY,IOUT,WETDRY,IWDFLG,CVWD)
C
C4------RETURN
      RETURN
      END
```

Module BCF5AD

```
      SUBROUTINE BCF5AD(IBOUND,HOLD,BOT,WETDRY,IWDFLG,ISS,
     1                  NCOL,NROW,NLAY)
C
C-----VERSION 1659 30OCT1992 BCF5AD
C     ******************************************************************
C     SET HOLD TO BOT WHENEVER A WETTABLE CELL IS DRY
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C
      DIMENSION IBOUND(NCOL,NROW,NLAY),HOLD(NCOL,NROW,NLAY),
     1          BOT(NCOL,NROW,NLAY),WETDRY(NCOL,NROW,NLAY)
C
      COMMON /FLWCOM/LAYCON(200)
C     ------------------------------------------------------------------
C
C1------RETURN IF STEADY STATE OR IF NOT USING WETTING CAPABILITY
      IF(IWDFLG.EQ.0 .OR. ISS.NE.0) RETURN
C
C2------LOOP THROUGH ALL LAYERS TO SET HOLD=BOT IF A WETTABLE CELL IS DRY
      ZERO=0.
      KB=0
      DO 100 K=1,NLAY
C
C2A-----SKIP LAYERS THAT CANNOT CONVERT BETWEEN WET AND DRY
      IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.1) GO TO 100
      KB=KB+1
      DO 90 I=1,NROW
      DO 90 J=1,NCOL
C
C2B-----SKIP CELLS THAT ARE CURRENTLY WET OR ARE NOT WETTABLE
      IF(IBOUND(J,I,K).NE.0) GO TO 90
      IF(WETDRY(J,I,KB).EQ.ZERO) GO TO 90
C
C2C-----SET HOLD=BOT
      HOLD(J,I,K)=BOT(J,I,KB)
   90 CONTINUE
  100 CONTINUE
C
C3------RETURN
      RETURN
      END
```

Module BCF5FM

```
      SUBROUTINE BCF5FM(HCOF,RHS,HOLD,SC1,HNEW,IBOUND,CR,CC,CV,HY,TRPY,
     1                  BOT,TOP,SC2,DELR,DELC,DELT,ISS,KITER,KSTP,KPER,
     2                  NCOL,NROW,NLAY,IOUT,WETDRY,IWDFLG,CVWD,
     3                  WETFCT,IWETIT,IHDWET,HDRY,BUFF)
C-----VERSION 1500 29JUNE1993 BCF5FM
C     ******************************************************************
C     ADD LEAKAGE CORRECTION AND STORAGE TO HCOF AND RHS, AND CALCULATE
C     CONDUCTANCE AS REQUIRED
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW
C
      DIMENSION HCOF(NCOL,NROW,NLAY),RHS(NCOL,NROW,NLAY),
     1    HOLD(NCOL,NROW,NLAY),SC1(NCOL,NROW,NLAY),HNEW(NCOL,NROW,NLAY),
     2    IBOUND(NCOL,NROW,NLAY),CR(NCOL,NROW,NLAY),
     3    CC(NCOL,NROW,NLAY),CV(NCOL,NROW,NLAY),HY(NCOL,NROW,NLAY),
     4    TRPY(NLAY),BOT(NCOL,NROW,NLAY),TOP(NCOL,NROW,NLAY),DELR(NCOL),
     5    DELC(NROW),SC2(NCOL,NROW,NLAY),WETDRY(NCOL,NROW,NLAY),
     6    CVWD(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
C
      COMMON /FLWCOM/LAYCON(200)
C     ------------------------------------------------------------------
      KB=0
      KT=0
      ONE=1.
      TLED=ONE/DELT
C
C1------FOR EACH LAYER: IF T VARIES CALCULATE HORIZONTAL CONDUCTANCES
      DO 100 K=1,NLAY
      KK=K
      IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.2) KT=KT+1
C
C1A-----IF LAYER TYPE IS NOT 1 OR 3 THEN SKIP THIS LAYER.
      IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.1) GO TO 100
      KB=KB+1
C
C1B-----FOR LAYER TYPES 1 & 3 CALL SBCF5H TO CALCULATE
C1B-----HORIZONTAL CONDUCTANCES.
      CALL SBCF5H(HNEW,IBOUND,CR,CC,CV,HY,TRPY,DELR,DELC,BOT,TOP,
     1   KK,KB,KT,KITER,KSTP,KPER,NCOL,NROW,NLAY,IOUT,WETDRY,IWDFLG,
     2   CVWD,WETFCT,IWETIT,IHDWET,HDRY,BUFF)
  100 CONTINUE
C
C2------IF THE SIMULATION IS TRANSIENT ADD STORAGE TO HCOF AND RHS
      IF(ISS.NE.0) GO TO 201
      KT=0
      DO 200 K=1,NLAY
C
C3------SEE IF THIS LAYER IS CONVERTIBLE OR NON-CONVERTIBLE.
      IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.2) GO TO 150
C4------NON-CONVERTIBLE LAYER, SO USE PRIMARY STORAGE
      DO 140 I=1,NROW
      DO 140 J=1,NCOL
      IF(IBOUND(J,I,K).LE.0) GO TO 140
      RHO=SC1(J,I,K)*TLED
      HCOF(J,I,K)=HCOF(J,I,K)-RHO
      RHS(J,I,K)=RHS(J,I,K)-RHO*HOLD(J,I,K)
  140 CONTINUE
      GO TO 200
C
C5------A CONVERTIBLE LAYER, SO CHECK OLD AND NEW HEADS TO DETERMINE
C5------WHEN TO USE PRIMARY AND SECONDARY STORAGE
  150 KT=KT+1
      DO 180 I=1,NROW
      DO 180 J=1,NCOL
C
C5A-----IF THE CELL IS EXTERNAL THEN SKIP IT.
      IF(IBOUND(J,I,K).LE.0) GO TO 180
      TP=TOP(J,I,KT)
      RHO2=SC2(J,I,KT)*TLED
      RHO1=SC1(J,I,K)*TLED
C
C5B-----FIND STORAGE FACTOR AT START OF TIME STEP.
      SOLD=RHO2
      IF(HOLD(J,I,K).GT.TP) SOLD=RHO1
C
```

```
C5C-----FIND STORAGE FACTOR AT END OF TIME STEP.
      HTMP=HNEW(J,I,K)
      SNEW=RHO2
      IF(HTMP.GT.TP) SNEW=RHO1
C
C5D-----ADD STORAGE TERMS TO RHS AND HCOF.
      HCOF(J,I,K)=HCOF(J,I,K)-SNEW
      RHS(J,I,K)=RHS(J,I,K) - SOLD*(HOLD(J,I,K)-TP) - SNEW*TP
C
  180 CONTINUE
C
  200 CONTINUE
C
C6------FOR EACH LAYER DETERMINE IF CORRECTION TERMS ARE NEEDED FOR
C6------FLOW DOWN INTO PARTIALLY SATURATED LAYERS.
  201 KT=0
      DO 300 K=1,NLAY
C
C7------SEE IF CORRECTION IS NEEDED FOR LEAKAGE FROM ABOVE.
      IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.2) GO TO 250
      KT=KT+1
      IF(K.EQ.1) GO TO 250
C
C7A-----FOR EACH CELL MAKE THE CORRECTION IF NEEDED.
      DO 220 I=1,NROW
      DO 220 J=1,NCOL
C
C7B-----IF THE CELL IS EXTERNAL(IBOUND<=0) THEN SKIP IT.
      IF(IBOUND(J,I,K).LE.0) GO TO 220
      HTMP=HNEW(J,I,K)
C
C7C-----IF HEAD IS ABOVE TOP THEN CORRECTION NOT NEEDED
      IF(HTMP.GE.TOP(J,I,KT)) GO TO 220
C
C7D-----WITH HEAD BELOW TOP ADD CORRECTION TERMS TO RHS.
      RHS(J,I,K)=RHS(J,I,K) + CV(J,I,K-1)*(TOP(J,I,KT)-HTMP)
  220 CONTINUE
C
C8------SEE IF THIS LAYER MAY NEED CORRECTION FOR LEAKAGE TO BELOW.
  250 IF(K.EQ.NLAY) GO TO 300
      IF(LAYCON(K+1).NE.3 .AND. LAYCON(K+1).NE.2) GO TO 300
      KTT=KT+1
C
C8A-----FOR EACH CELL MAKE THE CORRECTION IF NEEDED.
      DO 280 I=1,NROW
      DO 280 J=1,NCOL
C
C8B-----IF CELL IS EXTERNAL (IBOUND<=0) THEN SKIP IT.
      IF(IBOUND(J,I,K).LE.0) GO TO 280
C
C8C-----IF HEAD IN THE LOWER CELL IS LESS THAN TOP ADD CORRECTION
C8C-----TERM TO RHS.
      HTMP=HNEW(J,I,K+1)
      IF(HTMP.LT.TOP(J,I,KTT)) RHS(J,I,K)=RHS(J,I,K)
     1                         - CV(J,I,K)*(TOP(J,I,KTT)-HTMP)
  280 CONTINUE
  300 CONTINUE
C
C9------RETURN
      RETURN
      END
```

Module SBCF5N

```
      SUBROUTINE SBCF5N(HNEW,IBOUND,SC1,SC2,CR,CC,CV,HY,TRPY,DELR,DELC,
     1    ISS,NCOL,NROW,NLAY,IOUT,WETDRY,IWDFLG,CVWD)
C
C-----VERSION 1456 29JUNE1993 SBCF5N
C     ******************************************************************
C     INITIALIZE AND CHECK BCF DATA
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C
      DOUBLE PRECISION HNEW,HCNV
C
      DIMENSION HNEW(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY)
     1    ,SC1(NCOL,NROW,NLAY),CR(NCOL,NROW,NLAY)
     2    ,CC(NCOL,NROW,NLAY),CV(NCOL,NROW,NLAY)
     3    ,HY(NCOL,NROW,NLAY),TRPY(NLAY),DELR(NCOL),DELC(NROW)
     4    ,SC2(NCOL,NROW,NLAY),WETDRY(NCOL,NROW,NLAY)
     5    ,CVWD(NCOL,NROW,NLAY)
C
      COMMON /FLWCOM/LAYCON(200)
      COMMON /FLWAVG/LAYAVG(200)
C     ------------------------------------------------------------------
C
C1------MULTIPLY VERTICAL LEAKANCE BY AREA TO MAKE CONDUCTANCE.
      ZERO=0.
      IF(NLAY.EQ.1) GO TO 20
      K1=NLAY-1
      DO 10 K=1,K1
      DO 10 I=1,NROW
      DO 10 J=1,NCOL
      CV(J,I,K)=CV(J,I,K)*DELR(J)*DELC(I)
   10 CONTINUE
C
C2------IF WETTING CAPABILITY IS ACTIVATED, SAVE CV IN CVWD FOR USE WHEN
C2------WETTING CELLS.
      IF(IWDFLG.EQ.0) GO TO 20
      DO 15 K=1,K1
      DO 15 I=1,NROW
      DO 15 J=1,NCOL
      CVWD(J,I,K)=CV(J,I,K)
   15 CONTINUE
C
C3------IF IBOUND=0, SET CV=0 AND CC=0.
   20 DO 30 K=1,NLAY
      DO 30 I=1,NROW
      DO 30 J=1,NCOL
      IF(IBOUND(J,I,K).NE.0) GO TO 30
      IF(K.NE.NLAY) CV(J,I,K)=ZERO
      IF(K.NE.1) CV(J,I,K-1)=ZERO
      CC(J,I,K)=ZERO
   30 CONTINUE
C
C4------INSURE THAT EACH ACTIVE CELL HAS AT LEAST ONE NON-ZERO
C4------TRANSMISSIVE PARAMETER.
      HCNV=888.88
      KB=0
      DO 60 K=1,NLAY
      IF(LAYCON(K).EQ.1 .OR. LAYCON(K).EQ.3) GO TO 50
C
C4A-----WHEN LAYER TYPE IS 0 OR 2, TRANSMISSIVITY OR CV MUST BE NONZERO.
      DO 45 I=1,NROW
      DO 45 J=1,NCOL
      IF(IBOUND(J,I,K).EQ.0) GO TO 45
      IF(CC(J,I,K).NE.ZERO) GO TO 45
      IF(K.EQ.NLAY) GO TO 41
      IF(CV(J,I,K).NE.ZERO) GO TO 45
   41 IF(K.EQ.1) GO TO 42
      IF(CV(J,I,K-1).NE.ZERO) GO TO 45
   42 IBOUND(J,I,K)=0
      HNEW(J,I,K)=HCNV
      WRITE(IOUT,43) K,I,J
   43 FORMAT(1X,'NODE (LAYER,ROW,COL)',3I4,
     1       ' ELIMINATED BECAUSE ALL CONDUCTANCES TO NODE ARE 0')
   45 CONTINUE
      GO TO 60
C
```

```
C4B-----WHEN LAYER TYPE IS 1 OR 3, HY OR CV MUST BE NONZERO.
   50 KB=KB+1
      DO 59 I=1,NROW
      DO 59 J=1,NCOL
C
C4B1----IF WETTING CAPABILITY IS ACTIVE, CHECK CVWD.
      IF(IWDFLG.EQ.0) GO TO 55
      IF(WETDRY(J,I,KB).EQ.ZERO) GO TO 55
      IF(K.EQ.NLAY) GO TO 51
      IF(CVWD(J,I,K).NE.ZERO) GO TO 59
   51 IF(K.EQ.1) GO TO 57
      IF(CVWD(J,I,K-1).NE.ZERO) GO TO 59
      GO TO 57
C
C4B2----WETTING CAPABILITY IS INACTIVE, SO CHECK CV AT ACTIVE CELLS.
   55 IF(IBOUND(J,I,K).EQ.0) GO TO 59
      IF(K.EQ.NLAY) GO TO 56
      IF(CV(J,I,K).NE.ZERO) GO TO 59
   56 IF(K.EQ.1) GO TO 57
      IF(CV(J,I,K-1).NE.ZERO) GO TO 59
C
C4B3----CHECK HYDRAULIC CONDUCTIVITY.
   57 IF(HY(J,I,KB).NE.ZERO) GO TO 59
C
C4B4----HY AND CV ARE ALL 0, SO CONVERT CELL TO NO FLOW.
      IBOUND(J,I,K)=0
      HNEW(J,I,K)=HCNV
      IF(IWDFLG.NE.0) WETDRY(J,I,KB)=ZERO
      WRITE(IOUT,43) K,I,J
   59 CONTINUE
   60 CONTINUE
C
C5------CALCULATE HOR. CONDUCTANCE(CR AND CC) FOR CONSTANT T LAYERS.
      DO 70 K=1,NLAY
      KK=K
      IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.1) GO TO 70
      IF(LAYAVG(K).EQ.0) THEN
         CALL SBCF5C(CR,CC,TRPY,DELR,DELC,KK,NCOL,NROW,NLAY)
      ELSE IF(LAYAVG(K).EQ.10) THEN
         CALL SBCF5A(CR,CC,TRPY,DELR,DELC,KK,NCOL,NROW,NLAY)
      ELSE
         CALL SBCF5L(CR,CC,TRPY,DELR,DELC,KK,NCOL,NROW,NLAY)
      END IF
   70 CONTINUE
C
C6------IF TRANSIENT, LOOP THROUGH LAYERS AND CALCULATE STORAGE
C6------CAPACITY.
      IF(ISS.NE.0) GO TO 100
      KT=0
      DO 90 K=1,NLAY
C
C6A-----MULTIPLY PRIMARY STORAGE COEFFICIENT BY DELR & DELC TO GET
C6A-----PRIMARY STORAGE CAPACITY.
      DO 80 I=1,NROW
      DO 80 J=1,NCOL
      SC1(J,I,K)=SC1(J,I,K)*DELR(J)*DELC(I)
   80 CONTINUE
C
C6B-----IF LAYER IS CONF/UNCONF MULTIPLY SECONDARY STORAGE COEFFICIENT
C6B-----BY DELR AND DELC TO GET SECONDARY STORAGE CAPACITY(SC2).
      IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.2) GO TO 90
      KT=KT+1
      DO 85 I=1,NROW
      DO 85 J=1,NCOL
      SC2(J,I,KT)=SC2(J,I,KT)*DELR(J)*DELC(I)
   85 CONTINUE
   90 CONTINUE
C
C7------RETURN.
  100 RETURN
      END
```

Module SBCF5H

```
      SUBROUTINE SBCF5H(HNEW,IBOUND,CR,CC,CV,HY,TRPY,DELR,DELC
     1,BOT,TOP,K,KB,KT,KITER,KSTP,KPER,NCOL,NROW,NLAY,IOUT
     2,WETDRY,IWDFLG,CVWD,WETFCT,IWETIT,IHDWET,HDRY,BUFF)
C-----VERSION 1501 29JUNE1993 SBCF5H
C     ******************************************************************
C     COMPUTE CONDUCTANCE FOR ONE LAYER FROM SATURATED THICKNESS AND
C     HYDRAULIC CONDUCTIVITY
C     ******************************************************************
C
C      SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW,HD,BBOT,TTOP
C
      DIMENSION HNEW(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY)
     1,CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY), CV(NCOL,NROW,NLAY)
     2,HY(NCOL,NROW,NLAY), TRPY(NLAY), DELR(NCOL), DELC(NROW)
     3,BOT(NCOL,NROW,NLAY),TOP(NCOL,NROW,NLAY),WETDRY(NCOL,NROW,NLAY)
     4,CVWD(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
      CHARACTER*3 ACNVRT
      DIMENSION ICNVRT(5),JCNVRT(5),ACNVRT(5)
C
      COMMON /FLWCOM/LAYCON(200)
      COMMON /FLWAVG/LAYAVG(200)
C     ------------------------------------------------------------------
C
C1------LOOP THROUGH EACH CELL IN LAYER AND CALCULATE TRANSMISSIVITY AT
C1------EACH ACTIVE CELL.
      ZERO=0.
      NCNVRT=0
      IHDCNV=0
      ITFLG=1
      IF(IWDFLG.NE.0) ITFLG=MOD(KITER,IWETIT)
      DO 200 I=1,NROW
      DO 200 J=1,NCOL
C
C2------IF CELL IS ACTIVE, THEN SKIP TO CODE THAT CALCULATES SATURATED
C2------THICKNESS.
      IF(IBOUND(J,I,K).NE.0) GO TO 20
C
C3------DETERMINE IF THE CELL CAN CONVERT BETWEEN CONFINED AND
C3------UNCONFINED.  IF NOT, SKIP TO CODE THAT SETS TRANSMISSIVITY TO 0.
      IF(ITFLG.NE.0) GO TO 6
      IF(WETDRY(J,I,KB).EQ.ZERO)GO TO 6
      WD=WETDRY(J,I,KB)
      IF(WD.LT.ZERO) WD=-WD
      TURNON=BOT(J,I,KB)+WD
C
C3A-----CHECK HEAD IN CELL BELOW TO SEE IF WETTING THRESHOLD HAS BEEN
C3A-----REACHED.
      IF(K.EQ.NLAY)GO TO 2
      HTMP=HNEW(J,I,K+1)
      IF(IBOUND(J,I,K+1).GT.0.AND.HTMP.GE.TURNON)GO TO 9
C
C3B-----CHECK HEAD IN ADJACENT HORIZONTAL CELLS TO SEE IF WETTING
C3B-----THRESHOLD HAS BEEN REACHED.
    2 IF(WETDRY(J,I,KB).LT.ZERO) GO TO 6
      IF(J.EQ.1)GO TO 3
      HTMP=HNEW(J-1,I,K)
      IF(IBOUND(J-1,I,K).GT.0.AND.IBOUND(J-1,I,K).NE.30000.AND.
     1                           HTMP.GE.TURNON)GO TO 9
    3 IF(J.EQ.NCOL)GO TO 4
      HTMP=HNEW(J+1,I,K)
      IF(IBOUND(J+1,I,K).GT.0.AND.HTMP.GE.TURNON)GO TO 9
    4 IF(I.EQ.1)GO TO 5
      HTMP=HNEW(J,I-1,K)
      IF(IBOUND(J,I-1,K).GT.0.AND.IBOUND(J,I-1,K).NE.30000.AND.
     1                            HTMP.GE.TURNON)GO TO 9
    5 IF(I.EQ.NROW)GO TO 6
      HTMP=HNEW(J,I+1,K)
      IF(IBOUND(J,I+1,K).GT.0.AND.HTMP.GE.TURNON)GO TO 9
C
C3C-----CELL IS DRY AND STAYS DRY.  SET TRANSMISSIVITY TO 0, SET
C3C-----SATURATED THICKNESS (BUFF) TO 0, AND SKIP TO THE NEXT CELL.
    6 CC(J,I,K)=ZERO
      IF(LAYAVG(K).EQ.30) BUFF(J,I,K)=ZERO
      GO TO 200
C
```

```
C4------CELL BECOMES WET.  SET INITIAL HEAD AND VERTICAL CONDUCTANCE.
    9 IF(IHDWET.NE.0) HNEW(J,I,K)=BOT(J,I,KB)+WETFCT*WD
      IF(IHDWET.EQ.0) HNEW(J,I,K)=BOT(J,I,KB)+WETFCT*(HTMP-BOT(J,I,KB))
      IF(K.EQ.NLAY) GO TO 12
      IF(IBOUND(J,I,K+1).NE.0) CV(J,I,K)= CVWD(J,I,K)
   12 IF(K.EQ.1) GO TO 14
      IF(IBOUND(J,I,K-1).NE.0) CV(J,I,K-1)= CVWD(J,I,K-1)
   14 IBOUND(J,I,K)=30000
C
C4A-----PRINT MESSAGE SAYING CELL HAS BEEN CONVERTED TO WET.
      NCNVRT=NCNVRT+1
      ICNVRT(NCNVRT)=I
      JCNVRT(NCNVRT)=J
      ACNVRT(NCNVRT)='WET'
      IF(NCNVRT.LT.5) GO TO 20
         IF(IHDCNV.EQ.0) WRITE(IOUT,17) KITER,K,KSTP,KPER
   17    FORMAT(1X,/1X,'CELL CONVERSIONS FOR ITER.=',I3,'  LAYER=',
     1    I3,'  STEP=',I3,'  PERIOD=',I3,'   (ROW,COL)')
         IHDCNV=1
         WRITE(IOUT,18) (ACNVRT(L),ICNVRT(L),JCNVRT(L),L=1,NCNVRT)
   18    FORMAT(1X,3X,5(A,'(',I3,',',I3,')   '))
         NCNVRT=0
C
C5------CALCULATE SATURATED THICKNESS.
   20 HD=HNEW(J,I,K)
      BBOT=BOT(J,I,KB)
      IF(LAYCON(K).EQ.1) GO TO 50
      TTOP=TOP(J,I,KT)
      IF(HD.GT.TTOP) HD=TTOP
   50 THCK=HD-BBOT
C
C6------CHECK TO SEE IF SATURATED THICKNESS IS GREATER THAN ZERO.
      IF(THCK.LE.ZERO) GO TO 100
C
C6A-----IF SATURATED THICKNESS>0 THEN EITHER CALCULATE TRANSMISSIVITY
C6A-----AS HYDRAULIC CONDUCTIVITY TIMES SATURATED THICKNESS OR STORE
C6A-----K IN CC AND SATURATED THICKNESS IN BUFF.
      IF(LAYAVG(K).EQ.30) THEN
         CC(J,I,K)=HY(J,I,KB)
         BUFF(J,I,K)=THCK
      ELSE
         CC(J,I,K)=THCK*HY(J,I,KB)
      END IF
      GO TO 200
C
C6B-----WHEN SATURATED THICKNESS < 0, PRINT A MESSAGE AND SET
C6B-----TRANSMISSIVITY, IBOUND, AND VERTICAL CONDUCTANCE =0
  100 NCNVRT=NCNVRT+1
      ICNVRT(NCNVRT)=I
      JCNVRT(NCNVRT)=J
      ACNVRT(NCNVRT)='DRY'
      IF(NCNVRT.LT.5) GO TO 150
         IF(IHDCNV.EQ.0) WRITE(IOUT,17) KITER,K,KSTP,KPER
         IHDCNV=1
         WRITE(IOUT,18) (ACNVRT(L),ICNVRT(L),JCNVRT(L),L=1,NCNVRT)
         NCNVRT=0
  150 HNEW(J,I,K)=HDRY
      CC(J,I,K)=ZERO
      IF(IBOUND(J,I,K).GE.0) GO TO 160
         WRITE(IOUT,151)
  151    FORMAT(1X,/1X,'CONSTANT-HEAD CELL WENT DRY',
     1           ' -- SIMULATION ABORTED')
         WRITE(IOUT,152) K,I,J,KITER,KSTP,KPER
  152    FORMAT(1X,'LAYER=',I2,'   ROW=',I3,'   COLUMN=',I3,
     1     '   ITERATION=',I3,'   TIME STEP=',I3,'   STRESS PERIOD=',I3)
         STOP
  160 IBOUND(J,I,K)=0
      IF(K.LT.NLAY) CV(J,I,K)=ZERO
      IF(K.GT.1) CV(J,I,K-1)=ZERO
  200 CONTINUE
C
C7------PRINT ANY REMAINING CELL CONVERSIONS NOT YET PRINTED
      IF(NCNVRT.EQ.0) GO TO 203
         IF(IHDCNV.EQ.0) WRITE(IOUT,17) KITER,K,KSTP,KPER
         IHDCNV=1
         WRITE(IOUT,18) (ACNVRT(L),ICNVRT(L),JCNVRT(L),L=1,NCNVRT)
         NCNVRT=0
C
C8------CHANGE IBOUND VALUE FOR CELLS THAT CONVERTED TO WET THIS
C8------ITERATION FROM 30000 to 1.
```

```
  203 IF(IWDFLG.EQ.0) GO TO 210
      DO 205 I=1,NROW
      DO 205 J=1,NCOL
      IF(IBOUND(J,I,K).EQ.30000) IBOUND(J,I,K)=1
  205 CONTINUE
C
C9------COMPUTE HORIZONTAL BRANCH CONDUCTANCES FROM TRANSMISSIVITY.
  210 IF(LAYAVG(K).EQ.0) THEN
         CALL SBCF5C(CR,CC,TRPY,DELR,DELC,K,NCOL,NROW,NLAY)
      ELSE IF(LAYAVG(K).EQ.10) THEN
         CALL SBCF5A(CR,CC,TRPY,DELR,DELC,K,NCOL,NROW,NLAY)
      ELSE IF(LAYAVG(K).EQ.20) THEN
         CALL SBCF5L(CR,CC,TRPY,DELR,DELC,K,NCOL,NROW,NLAY)
      ELSE
         CALL SBCF5U(CR,CC,TRPY,DELR,DELC,BUFF,K,NCOL,NROW,NLAY)
      END IF
C
C10-----RETURN.
      RETURN
      END
```

Module SBCF5C

```
      SUBROUTINE SBCF5C(CR,CC,TRPY,DELR,DELC,K,NCOL,NROW,NLAY)
C
C
C-----VERSION 1512 02JULY1993 SBCF5C
C     ******************************************************************
C     COMPUTE BRANCH CONDUCTANCE USING HARMONIC MEAN OF BLOCK
C     CONDUCTANCES -- BLOCK TRANSMISSIVITY IS IN CC UPON ENTRY
C     ******************************************************************
C
C      SPECIFICATIONS:
C     ------------------------------------------------------------------
C
      DIMENSION CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY)
     2   , TRPY(NLAY), DELR(NCOL), DELC(NROW)
C
C     ------------------------------------------------------------------
      ZERO=0.
      TWO=2.
      YX=TRPY(K)*TWO
C
C1------FOR EACH CELL CALCULATE BRANCH CONDUCTANCES FROM THAT CELL
C1------TO THE ONE ON THE RIGHT AND THE ONE IN FRONT.
      DO 40 I=1,NROW
      DO 40 J=1,NCOL
      T1=CC(J,I,K)
C
C2------IF T=0 THEN SET CONDUCTANCE EQUAL TO 0. GO ON TO NEXT CELL.
      IF(T1.NE.ZERO) GO TO 10
      CR(J,I,K)=ZERO
      GO TO 40
C
C3------IF THIS IS NOT THE LAST COLUMN(RIGHTMOST) THEN CALCULATE
C3------BRANCH CONDUCTANCE IN THE ROW DIRECTION (CR) TO THE RIGHT.
   10 IF(J.EQ.NCOL) GO TO 30
      T2=CC(J+1,I,K)
      CR(J,I,K)=TWO*T2*T1*DELC(I)/(T1*DELR(J+1)+T2*DELR(J))
C
C4------IF THIS IS NOT THE LAST ROW(FRONTMOST) THEN CALCULATE
C4------BRANCH CONDUCTANCE IN THE COLUMN DIRECTION (CC) TO THE FRONT.
   30 IF(I.EQ.NROW) GO TO 40
      T2=CC(J,I+1,K)
      CC(J,I,K)=YX*T2*T1*DELR(J)/(T1*DELC(I+1)+T2*DELC(I))
   40 CONTINUE
C
C5------RETURN
      RETURN
      END
```

Module SBCF5A

```
      SUBROUTINE SBCF5A(CR,CC,TRPY,DELR,DELC,K,NCOL,NROW,NLAY)
C
C-----VERSION 02JULY1993 SBCF5A
C     ****************************************************************
C-------COMPUTE CONDUCTANCE USING ARITHMETIC MEAN TRANSMISSIVITY
C-------ACTIVATED BY LAYAVG=10
C     ****************************************************************
C
C      SPECIFICATIONS:
C     ----------------------------------------------------------------
C
      DIMENSION CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY)
     2   , TRPY(NLAY), DELR(NCOL), DELC(NROW)
C
C     ----------------------------------------------------------------
      ZERO=0.
      YX=TRPY(K)
C
C1------FOR EACH CELL CALCULATE BRANCH CONDUCTANCES FROM THAT CELL
C1------TO THE ONE ON THE RIGHT AND THE ONE IN FRONT.
      DO 40 I=1,NROW
      DO 40 J=1,NCOL
      T1=CC(J,I,K)
C
C2------IF T=0 THEN SET CONDUCTANCE EQUAL TO 0. GO ON TO NEXT CELL.
      IF(T1.NE.ZERO) GO TO 10
      CR(J,I,K)=ZERO
      GO TO 40
C
C3------IF THIS IS NOT THE LAST COLUMN(RIGHTMOST) THEN CALCULATE
C3------BRANCH CONDUCTANCE IN THE ROW DIRECTION (CR) TO THE RIGHT.
   10 IF(J.EQ.NCOL) GO TO 30
      T2=CC(J+1,I,K)
C3A-----ARITHMETIC MEAN INTERBLOCK TRANSMISSIVITY
      IF(T2.EQ.ZERO) THEN
         CR(J,I,K)=ZERO
      ELSE
         CR(J,I,K)=DELC(I)*(T1+T2)/(DELR(J+1)+DELR(J))
      END IF
C
C4------IF THIS IS NOT THE LAST ROW(FRONTMOST) THEN CALCULATE
C4------BRANCH CONDUCTANCE IN THE COLUMN DIRECTION (CC) TO THE FRONT.
   30 IF(I.EQ.NROW) GO TO 40
      T2=CC(J,I+1,K)
      IF(T2.EQ.ZERO) THEN
         CC(J,I,K)=ZERO
      ELSE
         CC(J,I,K)=YX*DELR(J)*(T1+T2)/(DELC(I+1)+DELC(I))
      END IF
   40 CONTINUE
C
C5------RETURN
      RETURN
      END
```

Module SBCF5L

```
      SUBROUTINE SBCF5L(CR,CC,TRPY,DELR,DELC,K,NCOL,NROW,NLAY)
C
C-----VERSION 02JULY1993 SBCF5L
C     ****************************************************************
C-------COMPUTE CONDUCTANCE USING LOGARITHMIC MEAN TRANSMISSIVITY
C-------ACTIVATED BY LAYAVG=20
C     ****************************************************************
C
C      SPECIFICATIONS:
C     ----------------------------------------------------------------
C
      DIMENSION CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY)
     2   , TRPY(NLAY), DELR(NCOL), DELC(NROW)
C
C     ----------------------------------------------------------------
      ZERO=0.
      TWO=2.
      HALF=0.5
      FRAC1=1.005
      FRAC2=0.995
      YX=TRPY(K)*TWO
C
C1------FOR EACH CELL CALCULATE BRANCH CONDUCTANCES FROM THAT CELL
C1------TO THE ONE ON THE RIGHT AND THE ONE IN FRONT.
      DO 40 I=1,NROW
      DO 40 J=1,NCOL
      T1=CC(J,I,K)
C
C2------IF T=0 THEN SET CONDUCTANCE EQUAL TO 0. GO ON TO NEXT CELL.
      IF(T1.NE.ZERO) GO TO 10
      CR(J,I,K)=ZERO
      GO TO 40
C
C3------IF THIS IS NOT THE LAST COLUMN(RIGHTMOST) THEN CALCULATE
C3------BRANCH CONDUCTANCE IN THE ROW DIRECTION (CR) TO THE RIGHT.
   10 IF(J.EQ.NCOL) GO TO 30
      T2=CC(J+1,I,K)
      IF(T2.EQ.ZERO) THEN
C3A-----SET TO ZERO AND EXIT IF T2 IS ZERO
         CR(J,I,K)=ZERO
         GO TO 30
      END IF
C3B-----LOGARITHMIC MEAN INTERBLOCK TRANSMISSIVITY
      RATIO=T2/T1
      IF(RATIO.GT.FRAC1.OR.RATIO.LT.FRAC2) THEN
         T=(T2-T1)/LOG(RATIO)
      ELSE
         T=HALF*(T1+T2)
      END IF
      CR(J,I,K)=TWO*DELC(I)*T/(DELR(J+1)+DELR(J))
C
C4------IF THIS IS NOT THE LAST ROW(FRONTMOST) THEN CALCULATE
C4------BRANCH CONDUCTANCE IN THE COLUMN DIRECTION (CC) TO THE FRONT.
   30 IF(I.EQ.NROW) GO TO 40
      T2=CC(J,I+1,K)
      IF(T2.EQ.ZERO) THEN
         CC(J,I,K)=ZERO
         GO TO 40
      END IF
      RATIO=T2/T1
      IF(RATIO.GT.FRAC1.OR.RATIO.LT.FRAC2) THEN
         T=(T2-T1)/LOG(RATIO)
      ELSE
         T=HALF*(T1+T2)
      END IF
      CC(J,I,K)=YX*DELR(J)*T/(DELC(I+1)+DELC(I))
   40 CONTINUE
C
C5------RETURN
      RETURN
      END
```

Module SBCF5U

```
      SUBROUTINE SBCF5U(CR,CC,TRPY,DELR,DELC,BUFF,K,NCOL,NROW,NLAY)
C
C-----VERSION 02JULY1993 SBCF5U
C     ***************************************************************
C-------COMPUTE CONDUCTANCE USING ARITHMETIC MEAN SATURATED THICKNESS
C-------AND LOGARITHMIC MEAN HYDRAULIC CONDUCTIVITY
C-------NODE HYDRAULIC CONDUCTIVITY IS IN CC,
C-------NODE SATURATED THICKNESS IS IN BUFF
C-------ACTIVATED BY LAYAVG=30
C     ***************************************************************
C
C      SPECIFICATIONS:
C     ---------------------------------------------------------------
C
      DIMENSION CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY)
     2   , TRPY(NLAY), DELR(NCOL), DELC(NROW)
     3   , BUFF(NCOL,NROW,NLAY)
C
C     ---------------------------------------------------------------
      ZERO=0.
      HALF=0.5
      FRAC1=1.005
      FRAC2=0.995
      YX=TRPY(K)
C
C1------FOR EACH CELL CALCULATE BRANCH CONDUCTANCES FROM THAT CELL
C1------TO THE ONE ON THE RIGHT AND THE ONE IN FRONT.
      DO 40 I=1,NROW
      DO 40 J=1,NCOL
      T1=CC(J,I,K)
C
C2------IF T=0 THEN SET CONDUCTANCE EQUAL TO 0. GO ON TO NEXT CELL.
      IF(T1.NE.ZERO) GO TO 10
      CR(J,I,K)=ZERO
      GO TO 40
C
C3------IF THIS IS NOT THE LAST COLUMN(RIGHTMOST) THEN CALCULATE
C3------BRANCH CONDUCTANCE IN THE ROW DIRECTION (CR) TO THE RIGHT.
   10 IF(J.EQ.NCOL) GO TO 30
      T2=CC(J+1,I,K)
      IF(T2.EQ.ZERO) THEN
C3A-----SET TO ZERO AND EXIT IF T2 IS ZERO
         CR(J,I,K)=ZERO
         GO TO 30
      END IF
C3B-----LOGARITHMIC MEAN HYDRAULIC CONDUCTIVITY
      RATIO=T2/T1
      IF(RATIO.GT.FRAC1.OR.RATIO.LT.FRAC2) THEN
         T=(T2-T1)/LOG(RATIO)
      ELSE
         T=HALF*(T1+T2)
      END IF
C3C-----MULTIPLY LOGARITHMIC K BY ARITHMETIC SAT THICK
      CR(J,I,K)=DELC(I)*T*(BUFF(J,I,K)+BUFF(J+1,I,K))
     *               /(DELR(J+1)+DELR(J))
C
C4------IF THIS IS NOT THE LAST ROW(FRONTMOST) THEN CALCULATE
C4------BRANCH CONDUCTANCE IN THE COLUMN DIRECTION (CC) TO THE FRONT.
   30 IF(I.EQ.NROW) GO TO 40
      T2=CC(J,I+1,K)
      IF(T2.EQ.ZERO) THEN
         CC(J,I,K)=ZERO
         GO TO 40
      END IF
      RATIO=T2/T1
      IF(RATIO.GT.FRAC1.OR.RATIO.LT.FRAC2) THEN
         T=(T2-T1)/LOG(RATIO)
      ELSE
         T=HALF*(T1+T2)
      END IF
      CC(J,I,K)=YX*DELR(J)*T*(BUFF(J,I,K)+BUFF(J,I+1,K))
     *            /(DELC(I+1)+DELC(I))
   40 CONTINUE
C
C5------RETURN
      RETURN
      END
```

Module SBCF5B

Narrative for Module SBCF5B

Module SBCF5B computes flow between adjacent cells in a subregion of the model grid. It does so in three passes; across columns, across rows, and across layers. SBCF5B performs its functions as follows:

1. Set IBD flag if cell-by-cell flows will be written to disk. If IBCFCB is greater than 0, IBD will be set equal to ICBCFL. ICBCFL is set by the Output Control Option.

2. Set the subregion equal to the entire grid if values will be saved in a file.

3. If the direction code (IDIR) is not 1, then go to step 4. Direction 1 indicates flow should be calculated across columns. If there is only 1 column, RETURN because flow cannot be calculated unless there are at least 2 columns.

   A. If not saving values in a file, then set the subregion to the region indicated by calling arguments IL1,IL2,IR1,IR2,IC1,IC2. The flow for the subregion includes flow that crosses the boundary between the outside and inside of the region. Clear the buffer.

   B. Calculate flow through the right face, but avoid attempting to calculate flow out of the right edge of the grid (J2 must be less than NCOL). When ICHFLG is 0, do not calculate flow between 2 constant-head cells (or any combination of constant-head and no-flow cells). When ICHFLG is not 0, do not calculate flow when either cell is no flow. This check is unnecessary because conductance is 0 when a cell is no-flow, but the check saves unnecessary computation time. Flow is calculated as conductance times head difference.

   C. Record the buffer if indicated by the budget flag (IBD). If IBD is 1, call UBUDSV; if IBD is 2, call UBDSV1. RETURN.

4. If the direction code (IDIR) is not 2, then go to step 5. Direction 2 indicates flow should be calculated across rows. If there is only 1 row, RETURN because flow cannot be calculated unless there are at least 2 rows.

   A. If not saving values in a file, then set the subregion to the region indicated by calling arguments IL1,IL2,IR1,IR2,IC1,IC2. Clear the buffer.

   B. Calculate flow through the front face, but avoid attempting to calculate flow out of the front edge of the grid.

   C. Record the buffer if indicated by the budget flag (IBD). If IBD is 1, call UBUDSV; if IBD is 2, call UBDSV1. RETURN.

5. Direction code (IDIR) is not 1 or 2, so it assumed to be 3. Direction 3 indicates flow should be calculated across layers. If there is only 1 layer, RETURN because flow cannot be calculated unless there are at least 2 layers.

A. If not saving values in a file, then set the subregion to the region indicated by calling arguments IL1,IL2,IR1,IR2,IC1,IC2. Clear the buffer.

B. Calculate flow through the lower face, but avoid attempting to calculate flow out of the bottom of the grid. The head in the lower aquifer must be compared to its TOP elevation for confined / unconfined cells in order to limit flow when the lower cell is unconfined.

C. Record the buffer if indicated by the budget flag (IBD). If IBD is 1, call UBUDSV; if IBD is 2, call UBDSV1. RETURN.

```
      SUBROUTINE SBCF5B(HNEW,IBOUND,CR,CC,CV,TOP,NCOL,NROW,NLAY,KSTP,
     1      KPER,IBCFCB,BUFF,IOUT,ICBCFL,DELT,PERTIM,TOTIM,
     2      IDIR,IBDRET,ICHFLG,IC1,IC2,IR1,IR2,IL1,IL2)
C
C-----VERSION 1308 28JUNE1993 SBCF5B
C     ******************************************************************
C     COMPUTE FLOW BETWEEN ADJACENT CELLS IN A SUBREGION OF THE GRID
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 TEXT(3)
      DOUBLE PRECISION HNEW,HD
C
      DIMENSION HNEW(NCOL,NROW,NLAY), IBOUND(NCOL,NROW,NLAY),
     1     CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY),
     2     CV(NCOL,NROW,NLAY), TOP(NCOL,NROW,NLAY),
     3     BUFF(NCOL,NROW,NLAY)
C
      COMMON /FLWCOM/LAYCON(200)
C
      DATA TEXT(1),TEXT(2),TEXT(3)
     1 /'FLOW RIGHT FACE ','FLOW FRONT FACE ','FLOW LOWER FACE '/
C     ------------------------------------------------------------------
C
C1------IF CELL-BY-CELL FLOWS WILL BE SAVED IN A FILE, SET FLAG IBD.
C1------RETURN IF FLOWS ARE NOT BEING SAVED OR RETURNED.
      ZERO=0.
      IBD=0
      IF(IBCFCB.GT.0) IBD=ICBCFL
      IF(IBD.EQ.0 .AND. IBDRET.EQ.0) RETURN
C
C2------SET THE SUBREGION EQUAL TO THE ENTIRE GRID IF VALUES ARE BEING
C2------SAVED IN A FILE.
      IF(IBD.NE.0) THEN
         K1=1
         K2=NLAY
         I1=1
         I2=NROW
         J1=1
         J2=NCOL
      END IF
C
C3------TEST FOR DIRECTION OF CALCULATION;  IF NOT ACROSS COLUMNS, GO TO
C3------STEP 4.  IF ONLY 1 COLUMN, RETURN.
      IF(IDIR.NE.1) GO TO 405
      IF(NCOL.EQ.1) RETURN
C
C3A-----CALCULATE FLOW ACROSS COLUMNS (THROUGH RIGHT FACE).  IF NOT
C3A-----SAVING IN A FILE, SET THE SUBREGION.  CLEAR THE BUFFER.
      IF(IBD.EQ.0) THEN
         K1=IL1
         K2=IL2
         I1=IR1
         I2=IR2
         J1=IC1-1
         IF(J1.LT.1) J1=1
         J2=IC2
      END IF
      DO 310 K=K1,K2
      DO 310 I=I1,I2
      DO 310 J=J1,J2
      BUFF(J,I,K)=ZERO
  310 CONTINUE
C
C3B-----FOR EACH CELL CALCULATE FLOW THRU RIGHT FACE & STORE IN BUFFER.
      IF(J2.EQ.NCOL) J2=J2-1
      DO 400 K=K1,K2
      DO 400 I=I1,I2
      DO 400 J=J1,J2
      IF(ICHFLG.EQ.0) THEN
         IF((IBOUND(J,I,K).LE.0) .AND. (IBOUND(J+1,I,K).LE.0)) GO TO 400
      ELSE
         IF((IBOUND(J,I,K).EQ.0) .OR. (IBOUND(J+1,I,K).EQ.0)) GO TO 400
      END IF
      HDIFF=HNEW(J,I,K)-HNEW(J+1,I,K)
      BUFF(J,I,K)=HDIFF*CR(J,I,K)
```

```
  400 CONTINUE
C
C3C-----RECORD CONTENTS OF BUFFER AND RETURN.
      IF(IBD.EQ.1)
     1   CALL UBUDSV(KSTP,KPER,TEXT(1),IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
      IF(IBD.EQ.2) CALL UBDSV1(KSTP,KPER,TEXT(1),IBCFCB,BUFF,NCOL,NROW,
     1     NLAY,IOUT,DELT,PERTIM,TOTIM,IBOUND)
      RETURN
C
C4------TEST FOR DIRECTION OF CALCULATION;  IF NOT ACROSS ROWS, GO TO
C4------STEP 5.  IF ONLY 1 ROW, RETURN.
  405 IF(IDIR.NE.2) GO TO 505
      IF(NROW.EQ.1) RETURN
C
C4A-----CALCULATE FLOW ACROSS ROWS (THROUGH FRONT FACE).  IF NOT SAVING
C4A-----IN A FILE, SET THE SUBREGION.  CLEAR THE BUFFER.
      IF(IBD.EQ.0) THEN
         K1=IL1
         K2=IL2
         I1=IR1-1
         IF(I1.LT.1) I1=1
         I2=IR2
         J1=IC1
         J2=IC2
      END IF
      DO 410 K=K1,K2
      DO 410 I=I1,I2
      DO 410 J=J1,J2
      BUFF(J,I,K)=ZERO
  410 CONTINUE
C
C4B-----FOR EACH CELL CALCULATE FLOW THRU FRONT FACE & STORE IN BUFFER.
      IF(I2.EQ.NROW) I2=I2-1
      DO 500 K=K1,K2
      DO 500 I=I1,I2
      DO 500 J=J1,J2
      IF(ICHFLG.EQ.0) THEN
         IF((IBOUND(J,I,K).LE.0) .AND. (IBOUND(J,I+1,K).LE.0)) GO TO 500
      ELSE
         IF((IBOUND(J,I,K).EQ.0) .OR. (IBOUND(J,I+1,K).EQ.0)) GO TO 500
      END IF
      HDIFF=HNEW(J,I,K)-HNEW(J,I+1,K)
      BUFF(J,I,K)=HDIFF*CC(J,I,K)
  500 CONTINUE
C
C4C-----RECORD CONTENTS OF BUFFER AND RETURN.
      IF(IBD.EQ.1)
     1   CALL UBUDSV(KSTP,KPER,TEXT(2),IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
      IF(IBD.EQ.2) CALL UBDSV1(KSTP,KPER,TEXT(2),IBCFCB,BUFF,NCOL,NROW,
     1     NLAY,IOUT,DELT,PERTIM,TOTIM,IBOUND)
      RETURN
C
C5------DIRECTION OF CALCULATION IS ACROSS LAYERS BY ELIMINATION.  IF
C5------ONLY 1 LAYER, RETURN.
  505 IF(NLAY.EQ.1) RETURN
C
C5A-----CALCULATE FLOW ACROSS LAYERS (THROUGH LOWER FACE).  IF NOT
C5A-----SAVING IN A FILE, SET THE SUBREGION.  CLEAR THE BUFFER.
      IF(IBD.EQ.0) THEN
         K1=IL1-1
         IF(K1.LT.1) K1=1
         K2=IL2
         I1=IR1
         I2=IR2
         J1=IC1
         J2=IC2
      END IF
      DO 510 K=K1,K2
      DO 510 I=I1,I2
      DO 510 J=J1,J2
      BUFF(J,I,K)=ZERO
  510 CONTINUE
C
C5B-----FOR EACH CELL CALCULATE FLOW THRU LOWER FACE & STORE IN BUFFER.
      IF(K2.EQ.NLAY) K2=K2-1
      KT=0
      DO 600 K=1,K2
      IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.2) KT=KT+1
      IF(K.LT.K1) GO TO 600
      DO 590 I=I1,I2
```

```
      DO 590 J=J1,J2
      IF(ICHFLG.EQ.0) THEN
         IF((IBOUND(J,I,K).LE.0) .AND. (IBOUND(J,I,K+1).LE.0)) GO TO 590
      ELSE
         IF((IBOUND(J,I,K).EQ.0) .OR. (IBOUND(J,I,K+1).EQ.0)) GO TO 590
      END IF
      HD=HNEW(J,I,K+1)
      IF(LAYCON(K+1).NE.3 .AND. LAYCON(K+1).NE.2) GO TO 580
      TMP=HD
      IF(TMP.LT.TOP(J,I,KT+1)) HD=TOP(J,I,KT+1)
  580 HDIFF=HNEW(J,I,K)-HD
      BUFF(J,I,K)=HDIFF*CV(J,I,K)
  590 CONTINUE
  600 CONTINUE
C
C5C-----RECORD CONTENTS OF BUFFER AND RETURN.
      IF(IBD.EQ.1)
     1   CALL UBUDSV(KSTP,KPER,TEXT(3),IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
      IF(IBD.EQ.2) CALL UBDSV1(KSTP,KPER,TEXT(3),IBCFCB,BUFF,NCOL,NROW,
     1     NLAY,IOUT,DELT,PERTIM,TOTIM,IBOUND)
      RETURN
      END
```

List of Variables for Module SBCF5B

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
CC	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) is conductance between cells (J,I,K) and (J,I+1,K).
CR	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) is conductance between cells (J,I,K) and (J+1,I,K).
CV	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the vertical direction. CV(J,I,K) is conductance between cells (J,I,K) and (J,I,K+1). Although CV is dimensioned to the size of the grid, space exists for only NLAY-1 layers.
DELT	Global	Length of the current time step.
HD	Module	Temporary value for head.
HDIFF	Module	Head difference between two adjacent nodes.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
I	Module	Index for rows.
I1	Module	First row of region in which flow is calculated.
I2	Module	Last row of region in which flow is calculated.
IBCFCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0 or < 0, cell-by-cell budget flow will not be saved or written to the listing file.
IBD	Module	Cell-by-cell budget flag, which is a composit of IBCFCB and ICBCFL: = 0, budget will not be saved for the current time step. = 1, budget will be saved by Module UBUDSV for the current time step. = 2, budget will be saved by Module UBDSV1.
IBDRET	Module	Flag that is non zero when there is a user-specified subregion.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IC1	Module	User-specified first column of subregion in which flow is to be calculated.
IC2	Module	User-specified last column of subregion in which flow is to be calculated.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.

ICHFLG	Global	Flag for flow between constant-head cells: = 0, flow between constant-head cells is not calculated. ≠ 0, flow between constant-head cells is calculated.
IDIR	Module	Code for which flow term is being calculated: 1 - across columns. 2 - across rows. 3 - across layers.
IL1	Module	User-specified first layer of subregion in which flow is to be calculated.
IL2	Module	User-specified last layer of subregion in which flow is to be calculated.
IOUT	Global	Unit number for writing to the listing file.
IR1	Module	User-specified first row of subregion in which flow is to be calculated.
IR2	Module	User-specified last row of subregion in which flow is to be calculated.
J	Module	Index for columns.
J1	Module	First column of region in which flow is calculated.
J2	Module	Last column of region in which flow is calculated.
K	Module	Index for layers.
K1	Module	First layer of region in which flow is calculated.
K2	Module	Last layer of region in which flow is calculated.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
KT	Module	Layer index for TOP array.
LAYCON	Package	DIMENSION (200), Layer-type code: 0 - Layer is always confined. 1 - Layer is always unconfined. 2 - Layer is convertible between confined and unconfined, but transmissivity is constant. 3 - Layer is convertible between confined and unconfined, and transmissivity is not constant.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
TEXT	Module	CHARACTER*16(3), Labels that identify the flow terms.
TMP	Module	Single precision equivalent of HD.
TOP	Package	DIMENSION (NCOL,NROW,NLAY), Elevation of the top of cells. Although TOP is dimensioned to the size of the grid, space exists only for cells that can convert between confined and unconfined.
TOTIM	Global	Elapsed time in the simulation.
ZERO	Module	The constant 0.

Module SBCF5F

Narrative for Module SBCF5F

Module SBCF5F computes flow from constant-head cells. SBCF5F performs its functions as follows.

1. Set flag IBD to indicate how cell-by-cell flows will be written. If ICBCFL is 0, no flows are written (IBD=0). If ICBCFL is not 0, flows are written as follows:
   If IBCFCB is less than 0, flows are written to the listing file (IBD=-1).
   If IBCFCB is greater than 0 and ICBCFL is 1, flows are written to unit IBCFCB as a 3-D array (IBD=1).
   If IBCFCB is greater than 0 and ICBCFL is 2, flows are written to unit IBCFCB as a list (IBD=2).

2. Set total flow rates in and out of the model to 0. Also, set the flag that controls the printing of a label when cell-by-cell flows are printed in the listing.

3. Clear the 3-D buffer that is used to store flow rates for each cell in the grid.

   3A. If cell-by-cell flows are to be written as a list, call UBDSV2 to write header information.

4. Loop through each cell in the grid calculating constant-head flow. Keep track of the layer index for the TOP array for confined / unconfined layers.

5. Skip the cell if it is not constant head.

6. Clear values for flow through the 6 faces of the cell.

7. Calculate flow through the left face. Comments A-C appear only in this section, but they apply in similar manner to section 8-12.

   A. If there is no flow across this face, skip to the next face. No flow occurs at the edge of the grid, when the adjacent cell is no flow, and when both the adjacent cell is constant head and ICHFLG=0.

   B. Calculate flow through the face as head difference times conductance.

   C. Accumulate total constant-head flow. In order to accumulate inflows separately from outflows, check the sign of the flow. When flow is negative, subtract flow from total outflow. When flow is positive, add flow to total inflow.

8. Calculate flow through the right face.

9. Calculate flow through the back face.

10. Calculate flow through the front face.

11. Calculate flow through the upper face.

12. Calculate flow through the lower face.

13. Calculate the total constant-head flow for the cell by adding the flow for the 6 faces. Store the flow in the cell-by-cell buffer.

14. If IBD is -1, print the flow for the cell in the listing file. Print the label for constant-head flow only once.

15. If IBD is 2, call UBDSVA to save flow to a file as a list. This is the end of the loop that is invoked for each river reach.

16. If IBD is 1, call UBUDSV to save flow as a 3-D array.

17. Move total flow rates and volumes into the global array of budget terms for the model budget. Also, define the name of the river budget term, and increment the budget term counter.

18. RETURN.

```
      SUBROUTINE SBCF5F(VBNM,VBVL,MSUM,HNEW,IBOUND,CR,CC,CV,TOP,DELT,
     1         NCOL,NROW,NLAY,KSTP,KPER,IBCFCB,BUFF,IOUT,ICBCFL,
     2         PERTIM,TOTIM,ICHFLG)
C-----VERSION 1315 18DEC1992 SBCF5F
C     ******************************************************************
C     COMPUTE FLOW FROM CONSTANT-HEAD CELLS
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM),TEXT
      DOUBLE PRECISION HNEW,HD,CHIN,CHOUT,XX1,XX2,XX3,XX4,XX5,XX6
C
      DIMENSION HNEW(NCOL,NROW,NLAY), IBOUND(NCOL,NROW,NLAY),
     1     CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY),
     2     CV(NCOL,NROW,NLAY), VBVL(4,MSUM),
     3     TOP(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
C
      COMMON /FLWCOM/LAYCON(200)
C
      DATA TEXT /'   CONSTANT HEAD'/
C     ------------------------------------------------------------------
C
C1------SET IBD TO INDICATE IF CELL-BY-CELL BUDGET VALUES WILL BE SAVED.
      IBD=0
      IF(IBCFCB.LT.0 .AND. ICBCFL.NE.0) IBD=-1
      IF(IBCFCB.GT.0) IBD=ICBCFL
C
C2------CLEAR BUDGET ACCUMULATORS.
      ZERO=0.
      CHIN=ZERO
      CHOUT=ZERO
      IBDLBL=0
C
C3------CLEAR BUFFER.
      DO 5 K=1,NLAY
      DO 5 I=1,NROW
      DO 5 J=1,NCOL
      BUFF(J,I,K)=ZERO
5     CONTINUE
C
C3A-----IF SAVING CELL-BY-CELL FLOW IN A LIST, COUNT CONSTANT-HEAD
C3A-----CELLS AND WRITE HEADER RECORDS.
      IF(IBD.EQ.2) THEN
         NCH=0
         DO 7 K=1,NLAY
         DO 7 I=1,NROW
         DO 7 J=1,NCOL
         IF(IBOUND(J,I,K).LT.0) NCH=NCH+1
7        CONTINUE
         CALL UBDSV2(KSTP,KPER,TEXT,IBCFCB,NCOL,NROW,NLAY,
     1           NCH,IOUT,DELT,PERTIM,TOTIM,IBOUND)
      END IF
C
C4------LOOP THROUGH EACH CELL AND CALCULATE FLOW INTO MODEL FROM EACH
C4------CONSTANT-HEAD CELL.
      KT=0
      DO 200 K=1,NLAY
      LC=LAYCON(K)
      IF(LC.EQ.3 .OR. LC.EQ.2) KT=KT+1
      DO 200 I=1,NROW
      DO 200 J=1,NCOL
C
C5------IF CELL IS NOT CONSTANT HEAD SKIP IT & GO ON TO NEXT CELL.
      IF (IBOUND(J,I,K).GE.0)GO TO 200
C
C6------CLEAR VALUES FOR FLOW RATE THROUGH EACH FACE OF CELL.
      X1=ZERO
      X2=ZERO
      X3=ZERO
      X4=ZERO
      X5=ZERO
      X6=ZERO
C
C7------CALCULATE FLOW THROUGH THE LEFT FACE.
C7------COMMENTS A-C APPEAR ONLY IN THE SECTION HEADED BY COMMENT 7,
C7------BUT THEY APPLY IN A SIMILAR MANNER TO SECTIONS 8-12.
C
```

```
C7A-----IF THERE IS NO FLOW TO CALCULATE THROUGH THIS FACE, THEN GO ON
C7A-----TO NEXT FACE.  NO FLOW OCCURS AT THE EDGE OF THE GRID, TO AN
C7A-----ADJACENT NO-FLOW CELL, OR TO AN ADJACENT CONSTANT-HEAD CELL
C7A-----WHEN ICHFLG IS 0.
      IF(J.EQ.1) GO TO 30
      IF(IBOUND(J-1,I,K).EQ.0) GO TO 30
      IF(ICHFLG.EQ.0 .AND. IBOUND(J-1,I,K).LT.0) GO TO 30
C
C7B-----CALCULATE FLOW THROUGH THIS FACE INTO THE ADJACENT CELL.
      HDIFF=HNEW(J,I,K)-HNEW(J-1,I,K)
      X1=HDIFF*CR(J-1,I,K)
      XX1=X1
C
C7C-----ACCUMULATE POSITIVE AND NEGATIVE FLOW.
      IF (X1) 10,30,20
   10 CHOUT=CHOUT-XX1
      GO TO 30
   20 CHIN=CHIN+XX1
C
C8------CALCULATE FLOW THROUGH THE RIGHT FACE.
   30 IF(J.EQ.NCOL) GO TO 60
      IF(IBOUND(J+1,I,K).EQ.0) GO TO 60
      IF(ICHFLG.EQ.0 .AND. IBOUND(J+1,I,K).LT.0) GO TO 60
      HDIFF=HNEW(J,I,K)-HNEW(J+1,I,K)
      X2=HDIFF*CR(J,I,K)
      XX2=X2
      IF(X2)40,60,50
   40 CHOUT=CHOUT-XX2
      GO TO 60
   50 CHIN=CHIN+XX2
C
C9------CALCULATE FLOW THROUGH THE BACK FACE.
   60 IF(I.EQ.1) GO TO 90
      IF (IBOUND(J,I-1,K).EQ.0) GO TO 90
      IF(ICHFLG.EQ.0 .AND. IBOUND(J,I-1,K).LT.0) GO TO 90
      HDIFF=HNEW(J,I,K)-HNEW(J,I-1,K)
      X3=HDIFF*CC(J,I-1,K)
      XX3=X3
      IF(X3) 70,90,80
   70 CHOUT=CHOUT-XX3
      GO TO 90
   80 CHIN=CHIN+XX3
C
C10-----CALCULATE FLOW THROUGH THE FRONT FACE.
   90 IF(I.EQ.NROW) GO TO 120
      IF(IBOUND(J,I+1,K).EQ.0) GO TO 120
      IF(ICHFLG.EQ.0 .AND. IBOUND(J,I+1,K).LT.0) GO TO 120
      HDIFF=HNEW(J,I,K)-HNEW(J,I+1,K)
      X4=HDIFF*CC(J,I,K)
      XX4=X4
      IF (X4) 100,120,110
  100 CHOUT=CHOUT-XX4
      GO TO 120
  110 CHIN=CHIN+XX4
C
C11-----CALCULATE FLOW THROUGH THE UPPER FACE.
  120 IF(K.EQ.1) GO TO 150
      IF (IBOUND(J,I,K-1).EQ.0) GO TO 150
      IF(ICHFLG.EQ.0 .AND. IBOUND(J,I,K-1).LT.0) GO TO 150
      HD=HNEW(J,I,K)
      IF(LC.NE.3 .AND. LC.NE.2) GO TO 122
      TMP=HD
      IF(TMP.LT.TOP(J,I,KT)) HD=TOP(J,I,KT)
  122 HDIFF=HD-HNEW(J,I,K-1)
      X5=HDIFF*CV(J,I,K-1)
      XX5=X5
      IF(X5) 130,150,140
  130 CHOUT=CHOUT-XX5
      GO TO 150
  140 CHIN=CHIN+XX5
C
C12-----CALCULATE FLOW THROUGH THE LOWER FACE.
  150 IF(K.EQ.NLAY) GO TO 180
      IF(IBOUND(J,I,K+1).EQ.0) GO TO 180
      IF(ICHFLG.EQ.0 .AND. IBOUND(J,I,K+1).LT.0) GO TO 180
      HD=HNEW(J,I,K+1)
      IF(LAYCON(K+1).NE.3 .AND. LAYCON(K+1).NE.2) GO TO 152
      TMP=HD
      IF(TMP.LT.TOP(J,I,KT+1)) HD=TOP(J,I,KT+1)
  152 HDIFF=HNEW(J,I,K)-HD
```

```
      X6=HDIFF*CV(J,I,K)
      XX6=X6
      IF(X6) 160,180,170
  160 CHOUT=CHOUT-XX6
      GO TO 180
  170 CHIN=CHIN+XX6
C
C13-----SUM THE FLOWS THROUGH SIX FACES OF CONSTANT HEAD CELL, AND
C13-----STORE SUM IN BUFFER.
 180  RATE=X1+X2+X3+X4+X5+X6
      BUFF(J,I,K)=RATE
C
C14-----PRINT THE FLOW FOR THE CELL IF REQUESTED.
      IF(IBD.LT.0) THEN
         IF(IBDLBL.EQ.0) WRITE(IOUT,899) TEXT,KPER,KSTP
  899    FORMAT(1X,/1X,A,'   PERIOD',I3,'   STEP',I3)
         WRITE(IOUT,900) K,I,J,RATE
  900    FORMAT(1X,'LAYER',I3,'   ROW',I4,'   COL',I4,
     1        '   RATE',1PG15.6)
         IBDLBL=1
      END IF
C
C15-----IF SAVING CELL-BY-CELL FLOW IN LIST, WRITE FLOW FOR CELL.
      IF(IBD.EQ.2) CALL UBDSVA(IBCFCB,NCOL,NROW,J,I,K,RATE,IBOUND,NLAY)
  200 CONTINUE
C
C16-----IF SAVING CELL-BY-CELL FLOW IN 3-D ARRAY, WRITE THE ARRAY.
      IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,
     1                  IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
C
C17-----SAVE TOTAL CONSTANT HEAD FLOWS AND VOLUMES IN VBVL TABLE
C17-----FOR INCLUSION IN BUDGET. PUT LABELS IN VBNM TABLE.
      CIN=CHIN
      COUT=CHOUT
      VBVL(1,MSUM)=VBVL(1,MSUM)+CIN*DELT
      VBVL(2,MSUM)=VBVL(2,MSUM)+COUT*DELT
      VBVL(3,MSUM)=CIN
      VBVL(4,MSUM)=COUT
      VBNM(MSUM)=TEXT
      MSUM=MSUM+1
C
C18-----RETURN.
      RETURN
      END
```

List of Variables for Module SBCF5F

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
CC	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) is conductance between cells (J,I,K) and (J,I+1,K).
CHIN	Module	Accumulator for flow into the modeled area.
CHOUT	Module	Accumulator for flow out of the modeled area.
CIN	Module	Single precision equivalent of CHIN.
COUT	Module	Single precision equivalent of CHOUT.
CR	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) is conductance between cells (J,I,K) and (J+1,I,K).
CV	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the vertical direction. CV(J,I,K) is conductance between cells (J,I,K) and (J,I,K+1). Although CV is dimensioned to the size of the grid, space exists for only NLAY-1 layers.
DELT	Global	Length of the current time step.
HD	Module	Temporary value for head.
HDIFF	Module	Head difference between two adjacent nodes.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
I	Module	Index for rows.
IBCFCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
IBD	Module	Cell-by-cell budget flag, which is a composit of IBCFCB and ICBCFL: = -1, budget will be printed in the listing file. = 0, budget will not be saved for the current time step. = 1, budget will be saved by Module UBUDSV for the current time step. = 2, budget will be saved by Modules UBDSV2 and UBDSVA.
IBDLBL	Module	Flag used when printing cell-by-cell budget values in the listing file so that the budget label is printed only once.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.

ICHFLG	Global	Flag for flow between constant-head cells: = 0, flow between constant-head cells is not calculated. ≠ 0, flow between constant-head cells is calculated.
IOUT	Global	Unit number for writing to the listing file.
J	Module	Index for columns.
K	Module	Index for layers.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
KT	Module	Layer index for TOP array.
LAYCON	Package	DIMENSION (200), Layer-type code: 0 - Layer is always confined. 1 - Layer is always unconfined. 2 - Layer is convertible between confined and unconfined, but transmissivity is constant. 3 - Layer is convertible between confined and unconfined, and transmissivity is not constant.
LC	Module	LAYCON(K).
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.
NCH	Module	Accumulator for the total number of constant-head cells.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
RATE	Module	The sum of flows through the 6 faces of a constant-head cell.
TEXT	Module	Label that identifies constant-head flow.
TMP	Module	Single precision equivalent of HD.
TOP	Package	DIMENSION (NCOL,NROW,NLAY), Elevation of the top of cells. Although TOP is dimensioned to the size of the grid, space exists only for cells that can convert between confined and unconfined.
TOTIM	Global	Elapsed time in the simulation.
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
X1	Module	Flow through the left face of a constant-head cell.
X2	Module	Flow through the right face of a constant-head cell.
X3	Module	Flow through the back face of a constant-head cell.
X4	Module	Flow through the front face of a constant-head cell.
X5	Module	Flow through the upper face of a constant-head cell.
X6	Module	Flow through the lower face of a constant-head cell.
XX1	Module	Double precision equivalent of X1.
XX2	Module	Double precision equivalent of X2.
XX3	Module	Double precision equivalent of X3.
XX4	Module	Double precision equivalent of X4.
XX5	Module	Double precision equivalent of X5.
XX6	Module	Double precision equivalent of X6.
ZERO	Module	The constant 0.

Module SBCF5S

Narrative for Module SBCF5S

Module SBCF5S calculates flow from storage. Rate of inflow, rate of outflow, volume of inflow, and volume of outflow are calculated for the overall budget. For cell-by-cell flows, the net inflow is calculated for each cell, with outflow being a negative value. Budget totals are accumulated in double precision in order to avoid truncation when summing storage for large models; however, storage for each cell is calculated as a single precision value because HOLD (head from the previous time step) is stored as a single precision value. Module SBCF5S performs its functions as follows:

1. RETURN if steady state (ISS not 0).

2. Initialize budget accumulators, STOIN and STOUT, and calculate 1./DELT.

3. Set IBD flag if cell-by-cell flows will be written to disk. If IBCFCB is greater than 0, IBD will be set equal to ICBCFL. ICBCFL is set by the Output Control Option.

4. Clear the buffer used for storing cell-by-cell flows.

5. Loop through each cell in the grid calculating storage flow. Keep track of the layer index for the TOP array for confined/unconfined layers.

6. Skip no-flow and constant-head cells. Also, convert head at current cell to single precision.

7. Check the layer-type code to see if there is one storage capacity (always confined or unconfined) or two (convertible between confined and unconfined).

   A. There are two storage capacities; calculate storage as a sum of confined and unconfined parts.

   B. There is one storage capacity; calculate storage.

8. Store storage in cell-by-cell buffer, and accumulate total storage. In order to accumulate inflows separately from outflows, check the sign of the flow. When flow is negative, subtract flow from total outflow. When flow is positive, add flow to total inflow.

9. Call the appropriate utility module to save cell-by-cell budget data, depending on the value of IBD.

10. Store total rates, and accumulate total volumes in VBVL. Put title in VBNM, and increment the budget counter, MSUM.

11. RETURN.

SBCF5S

```
      SUBROUTINE SBCF5S(VBNM,VBVL,MSUM,HNEW,IBOUND,HOLD,SC1,
     1   TOP,SC2,DELT,ISS,NCOL,NROW,NLAY,KSTP,KPER,IBCFCB,
     2   ICBCFL,BUFF,IOUT,PERTIM,TOTIM)
C-----VERSION 1257 28JUNE1993 SBCF5S
C     ******************************************************************
C     COMPUTE STORAGE BUDGET FLOW TERM FOR BCF.
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM),TEXT
      DOUBLE PRECISION HNEW,STOIN,STOUT,SSTRG
C
      DIMENSION HNEW(NCOL,NROW,NLAY), IBOUND(NCOL,NROW,NLAY),
     1   HOLD(NCOL,NROW,NLAY),SC1(NCOL,NROW,NLAY),VBVL(4,MSUM),
     2   SC2(NCOL,NROW,NLAY),TOP(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
C
      COMMON /FLWCOM/LAYCON(200)
C
      DATA TEXT /'         STORAGE'/
C     ------------------------------------------------------------------
C
C1------RETURN IF STEADY STATE.
      IF(ISS.NE.0) RETURN
C
C2------INITIALIZE BUDGET ACCUMULATORS AND 1/DELT.
      ZERO=0.
      STOIN=ZERO
      STOUT=ZERO
      ONE=1.
      TLED=ONE/DELT
C
C3------IF CELL-BY-CELL FLOWS WILL BE SAVED, SET FLAG IBD.
      IBD=0
      IF(IBCFCB.GT.0) IBD=ICBCFL
C
C4------CLEAR BUFFER.
      DO 210 K=1,NLAY
      DO 210 I=1,NROW
      DO 210 J=1,NCOL
      BUFF(J,I,K)=ZERO
210   CONTINUE
C
C5------LOOP THROUGH EVERY CELL IN THE GRID.
      KT=0
      DO 300 K=1,NLAY
      LC=LAYCON(K)
      IF(LC.EQ.3 .OR. LC.EQ.2) KT=KT+1
      DO 300 I=1,NROW
      DO 300 J=1,NCOL
C
C6------SKIP NO-FLOW AND CONSTANT-HEAD CELLS.
      IF(IBOUND(J,I,K).LE.0) GO TO 300
      HSING=HNEW(J,I,K)
C
C7-----CHECK LAYER TYPE TO SEE IF ONE STORAGE CAPACITY OR TWO.
      IF(LC.NE.3 .AND. LC.NE.2) GO TO 285
C
C7A----TWO STORAGE CAPACITIES.
      TP=TOP(J,I,KT)
      RHO2=SC2(J,I,KT)*TLED
      RHO1=SC1(J,I,K)*TLED
      SOLD=RHO2
      IF(HOLD(J,I,K).GT.TP) SOLD=RHO1
      SNEW=RHO2
      IF(HSING.GT.TP) SNEW=RHO1
      STRG=SOLD*(HOLD(J,I,K)-TP) + SNEW*TP - SNEW*HSING
      GO TO 288
C
C7B----ONE STORAGE CAPACITY.
  285 RHO=SC1(J,I,K)*TLED
      STRG=RHO*HOLD(J,I,K) - RHO*HSING
C
C8-----STORE CELL-BY-CELL FLOW IN BUFFER AND ADD TO ACCUMULATORS.
  288 BUFF(J,I,K)=STRG
      SSTRG=STRG
      IF(STRG) 292,300,294
```

```
  292 STOUT=STOUT-SSTRG
      GO TO 300
  294 STOIN=STOIN+SSTRG
C
  300 CONTINUE
C
C9-----IF IBD FLAG IS SET RECORD THE CONTENTS OF THE BUFFER.
      IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,
     1                      IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
      IF(IBD.EQ.2) CALL UBDSV1(KSTP,KPER,TEXT,IBCFCB,
     1            BUFF,NCOL,NROW,NLAY,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C10-----ADD TOTAL RATES AND VOLUMES TO VBVL & PUT TITLE IN VBNM.
      SIN=STOIN
      SOUT=STOUT
      VBVL(1,MSUM)=VBVL(1,MSUM)+SIN*DELT
      VBVL(2,MSUM)=VBVL(2,MSUM)+SOUT*DELT
      VBVL(3,MSUM)=SIN
      VBVL(4,MSUM)=SOUT
      VBNM(MSUM)=TEXT
      MSUM=MSUM+1
C
C11----RETURN.
      RETURN
      END
```

List of Variables for Module SBCF5S

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
DELT	Global	Length of the current time step.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
HOLD	Global	DIMENSION (NCOL,NROW,NLAY), Head at the start of the current time step.
HSING	Module	Single precision equivalent of HNEW.
I	Module	Index for rows.
IBCFCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. < 0 or = 0, cell-by-cell budget flow will not be saved or written to the listing file.
IBD	Module	Cell-by-cell budget flag, which is a composit of IBCFCB and ICBCFL: = 0, budget will not be saved for the current time step. = 1, budget will be saved by Module UBUDSV for the current time step. = 2, budget will be saved by Module UBDSV1.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IOUT	Global	Unit number for writing to the listing file.
ISS	Package	Steady-state flag: = 0, simulation is transient. ≠ 0, simulation is steady state.
J	Module	Index for columns.
K	Module	Index for layers.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
KT	Module	Layer index for TOP array.
LAYCON	Package	DIMENSION (200), Layer-type code: 0 - Layer is always confined. 1 - Layer is always unconfined. 2 - Layer is convertible between confined and unconfined, but transmissivity is constant. 3 - Layer is convertible between confined and unconfined, and transmissivity is not constant.
LC	Module	LAYCON(K).
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.

NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
ONE	Module	The constant 1.
PERTIM	Global	Elapsed time during the current stress period.
RHO	Module	Storage capacity divided by DELT for layers that are always confined or always unconfined.
RHO1	Module	Confined storage capacity divided by DELT for convertible layers.
RHO2	Module	Unconfined storage capacity divided by DELT for convertible layers.
SC1	Package	DIMENSION (NCOL,NROW,NLAY), Primary storage capacity.
SC2	Package	DIMENSION (NCOL,NROW,NLAY), Secondary storage capacity. Although SC2 is dimensioned to the size of the grid, space exists only for cells that can convert between confined and unconfined.
SIN	Module	Single precision equivalent of STOIN.
SNEW	Module	Storage capacity divided by DELT at the end of the time step for convertible layers.
SOLD	Module	Storage capacity divided by DELT at the start of the time step for convertible layers.
SOUT	Module	Single precision equivalent of STOUT.
SSTRG	Module	Double precision equivalent of STRG.
STOIN	Module	Accumulator for storage flow into the modeled area.
STOUT	Module	Accumulator for storage flow out of the modeled area.
STRG	Module	Flow from storage into the modeled area for a cell (negative for outflow).
TEXT	Module	Label that identifies the storage budget term.
TLED	Module	1./DELT.
TOP	Package	DIMENSION (NCOL,NROW,NLAY), Elevation of the top of cells. Although TOP is dimensioned to the size of the grid, space exists only for cells that can convert between confined and unconfined.
TOTIM	Global	Elapsed time in the simulation.
TP	Module	TOP(J,I,KT).
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
ZERO	Module	The constant 0.

River Package

Budget calculations in the River (RIV) Package have been changed to double precision. In addition, the ability to read extra data parameters for each river reach has been added.

Module RIV5AL

Narrative for Module RIV5AL

Module RIV5AL allocates space in the X array for river data. RIV5AL performs its functions as follows:

1. Write a message identifying the package. Also, initialize the number of river reaches to 0.

2. Read the first record from the RIV file into a buffer so it can be parsed by URWORD. Decode the maximum number of river reaches (MXRIVR) in any stress period and the river cell-by-cell budget flag (IRIVCB) using URWORD or READ depending on the free format flag (IFREFM).

3. Check for alphabetic options. "AUXILIARY" or "AUX" indicates an extra data parameter is to be read. "CBCALLOCATE" or "CBC" indicates that memory will be allocated to store the flow rate into the model from each river reach. Keep track of the total number of data values for each river reach in NRIVVL, which is needed in order to calculate the required space in the X array.

4. Allocate space in the X array for the RIVR array. Set LCRIVR equal to ISUM, which is the lowest unused location in X. Add the size of RIVR to ISUM.

5. Print the number of elements in the X array used by the RIV Package and the total space used in the X array.

6. RETURN.

```
      SUBROUTINE RIV5AL(ISUM,LENX,LCRIVR,MXRIVR,NRIVER,IN,IOUT,IRIVCB,
     1        NRIVVL,IRIVAL,IFREFM)
C
C-----VERSION 1445 20FEB1996 RIV5AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR RIVERS
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      COMMON /RIVCOM/RIVAUX(5)
      CHARACTER*16 RIVAUX
      CHARACTER*80 LINE
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE AND INITIALIZE NRIVER.
      WRITE(IOUT,1)IN
    1 FORMAT(1X,/1X,'RIV5 -- RIVER PACKAGE, VERSION 5, 9/1/93',
     1' INPUT READ FROM UNIT',I3)
      NRIVER=0
C
C2------READ MAXIMUM NUMBER OF RIVER REACHES AND UNIT OR FLAG FOR
C2------CELL-BY-CELL FLOW TERMS.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(2I10)') MXRIVR,IRIVCB
         LLOC=21
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,MXRIVR,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IRIVCB,R,IOUT,IN)
      END IF
      WRITE(IOUT,3) MXRIVR
    3 FORMAT(1X,'MAXIMUM OF',I5,' RIVER REACHES')
      IF(IRIVCB.LT.0) WRITE(IOUT,7)
    7 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE PRINTED WHEN ICBCFL NOT 0')
      IF(IRIVCB.GT.0) WRITE(IOUT,8) IRIVCB
    8 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE SAVED ON UNIT',I3)
C
C3------READ AUXILIARY PARAMETERS AND CBC ALLOCATION OPTION.
      IRIVAL=0
      NAUX=0
   10 CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
      IF(LINE(ISTART:ISTOP).EQ.'CBCALLOCATE' .OR.
     1   LINE(ISTART:ISTOP).EQ.'CBC') THEN
         IRIVAL=1
         WRITE(IOUT,11)
   11    FORMAT(1X,'MEMORY IS ALLOCATED FOR CELL-BY-CELL BUDGET TERMS')
         GO TO 10
      ELSE IF(LINE(ISTART:ISTOP).EQ.'AUXILIARY' .OR.
     1        LINE(ISTART:ISTOP).EQ.'AUX') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
         IF(NAUX.LT.5) THEN
            NAUX=NAUX+1
            RIVAUX(NAUX)=LINE(ISTART:ISTOP)
            WRITE(IOUT,12) RIVAUX(NAUX)
   12       FORMAT(1X,'AUXILIARY RIVER PARAMETER: ',A)
         END IF
         GO TO 10
      END IF
      NRIVVL=6+NAUX+IRIVAL
C
C4------ALLOCATE SPACE IN THE X ARRAY FOR THE RIVR ARRAY.
      LCRIVR=ISUM
      ISP=NRIVVL*MXRIVR
      ISUM=ISUM+ISP
C
C5------PRINT AMOUNT OF SPACE USED BY RIVER PACKAGE.
      WRITE (IOUT,14)ISP
   14 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY RIV')
      ISUM1=ISUM-1
      WRITE(IOUT,15)ISUM1,LENX
   15 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,16)
   16 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C6------RETURN.
      RETURN
      END
```

List of Variables for Module RIV5AL

Variable	Range	Definition
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
IN	Package	Primary unit number from which input for this package is read.
IOUT	Global	Unit number for writing to the listing file.
IRIVAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in RIVR array to return budget values. ≠ 0, memory has been allocated in RIVR array to return budget values.
IRIVCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
ISP	Module	Number of elements allocated in the X array by this package.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ISUM	Global	Index of the lowest element in the X array which has not yet been allocated.
ISUM1	Module	ISUM - 1.
LCRIVR	Package	Location in the X array of the first element of array RIVR.
LENX	Global	The number of elements in the X array. LENX is defined in a PARAMETER statement in the MAIN program.
LINE	Module	CHARACTER*80, contents of a record that has been read from the package input file. LINE is parsed by URWORD.
LLOC	Module	Index that tells URWORD where to start looking for a word within LINE.
MXRIVR	Package	The maximum number of river reaches active at one time.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NAUX	Module	Counter for the number of auxiliary river parameters.
NRIVER	Package	Number of river reaches active in the current stress period.
NRIVVL	Package	The size of the first dimension of the RIVR array; that is, RIVR has dimensions of (NRIVVL,MXRIVR).
R	Module	Argument place holder for calls to URWORD in which the argument is unused.
RIVAUX	Package	CHARACTER*16(5), names of auxiliary parameters.

Module RIV5RP

```
      SUBROUTINE RIV5RP(RIVR,NRIVER,MXRIVR,IN,IOUT,NRIVVL,IRIVAL,IFREFM)
C
C-----VERSION 1449 20FEB1996 RIV5RP
C     ******************************************************************
C     READ RIVER HEAD, CONDUCTANCE AND BOTTOM ELEVATION
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION RIVR(NRIVVL,MXRIVR)
      COMMON /RIVCOM/RIVAUX(5)
      CHARACTER*16 RIVAUX
      CHARACTER*151 LINE
C     ------------------------------------------------------------------
C
C1------READ ITMP (NUMBER OF RIVER REACHES OR FLAG TO REUSE DATA).
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(I10)') ITMP
      ELSE
         READ(IN,*) ITMP
      END IF
C
C2------TEST ITMP.
      IF(ITMP.GE.0) GO TO 50
C
C2A-----IF ITMP <0 THEN REUSE DATA FROM LAST STRESS PERIOD.
      WRITE(IOUT,7)
    7 FORMAT(1X,/1X,'REUSING RIVER REACHES FROM LAST STRESS PERIOD')
      GO TO 260
C
C3------IF ITMP=> ZERO THEN IT IS THE NUMBER OF RIVER REACHES.
   50 NRIVER=ITMP
C
C4------IF NRIVER>MXRIVR THEN STOP.
      IF(NRIVER.LE.MXRIVR)GO TO 100
      WRITE(IOUT,99)NRIVER,MXRIVR
   99 FORMAT(1X,/1X,'NRIVER(',I4,') IS GREATER THAN MXRIVR(',I4,')')
C
C4A-----ABNORMAL STOP.
      STOP
C
C5------PRINT NUMBER OF RIVER REACHES IN THIS STRESS PERIOD.
  100 WRITE(IOUT,101)NRIVER
  101 FORMAT(1X,//1X,I5,' RIVER REACHES')
C
C6------IF THERE ARE NO RIVER REACHES THEN RETURN.
      IF(NRIVER.EQ.0) GO TO 260
C
C7------READ AND PRINT DATA FOR EACH RIVER REACH.
      NAUX=NRIVVL-6-IRIVAL
      MAXAUX=NRIVVL-IRIVAL
      IF(NAUX.GT.0) THEN
         WRITE(IOUT,103) (RIVAUX(JJ),JJ=1,NAUX)
         WRITE(IOUT,104) ('------------------',JJ=1,NAUX)
      ELSE
         WRITE(IOUT,103)
         WRITE(IOUT,104)
      END IF
  103 FORMAT(1X,/1X,'LAYER   ROW   COL     STAGE   CONDUCTANCE   ',
     1          'BOT. ELEV.  REACH NO.',:5(2X,A))
  104 FORMAT(1X,65('-'),5A)
      DO 250 II=1,NRIVER
C7A-----READ THE REQUIRED DATA WITH FIXED OR FREE FORMAT.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(3I10,3F10.0)') K,I,J,(RIVR(JJ,II),JJ=4,6)
         LLOC=61
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,K,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,I,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,J,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,RIVR(4,II),IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,RIVR(5,II),IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,RIVR(6,II),IOUT,IN)
      END IF
```

```
C7B-----READ ANY AUXILIARY DATA WITH FREE FORMAT, AND PRINT ALL VALUES.
      IF(NAUX.GT.0) THEN
         DO 110 JJ=1,NAUX
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,RIVR(JJ+6,II),IOUT,IN)
  110    CONTINUE
         WRITE (IOUT,115) K,I,J,RIVR(4,II),RIVR(5,II),RIVR(6,II),II,
     1         (RIVR(JJ,II),JJ=7,MAXAUX)
      ELSE
         WRITE (IOUT,115) K,I,J,RIVR(4,II),RIVR(5,II),RIVR(6,II),II
      END IF
  115 FORMAT(1X,I4,I7,I6,G13.4,G12.4,G12.4,I8,:5(2X,G16.5))
      RIVR(1,II)=K
      RIVR(2,II)=I
      RIVR(3,II)=J
  250 CONTINUE
C
C8------RETURN.
  260 RETURN
      END
```

Module RIV5FM

```
      SUBROUTINE RIV5FM(NRIVER,MXRIVR,RIVR,HNEW,HCOF,RHS,IBOUND,
     1                  NCOL,NROW,NLAY,NRIVVL)
C
C-----VERSION 0950 16JULY1992 RIV5FM
C     ******************************************************************
C     ADD RIVER TERMS TO RHS AND HCOF
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW,RRBOT
      DIMENSION RIVR(NRIVVL,MXRIVR),HNEW(NCOL,NROW,NLAY),
     1          HCOF(NCOL,NROW,NLAY),RHS(NCOL,NROW,NLAY),
     2          IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C
C1------IF NRIVER<=0 THERE ARE NO RIVERS. RETURN.
      IF(NRIVER.LE.0)RETURN
C
C2------PROCESS EACH CELL IN THE RIVER LIST.
      DO 100 L=1,NRIVER
C
C3------GET COLUMN, ROW, AND LAYER OF CELL CONTAINING REACH.
      IL=RIVR(1,L)
      IR=RIVR(2,L)
      IC=RIVR(3,L)
C
C4------IF THE CELL IS EXTERNAL SKIP IT.
      IF(IBOUND(IC,IR,IL).LE.0)GO TO 100
C
C5------SINCE THE CELL IS INTERNAL GET THE RIVER DATA.
      HRIV=RIVR(4,L)
      CRIV=RIVR(5,L)
      RBOT=RIVR(6,L)
      RRBOT=RBOT
C
C6------COMPARE AQUIFER HEAD TO BOTTOM OF STREAM BED.
      IF(HNEW(IC,IR,IL).LE.RRBOT)GO TO 96
C
C7------SINCE HEAD>BOTTOM ADD TERMS TO RHS AND HCOF.
      RHS(IC,IR,IL)=RHS(IC,IR,IL)-CRIV*HRIV
      HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-CRIV
      GO TO 100
C
C8------SINCE HEAD<BOTTOM ADD TERM ONLY TO RHS.
   96 RHS(IC,IR,IL)=RHS(IC,IR,IL)-CRIV*(HRIV-RBOT)
  100 CONTINUE
C
C9------RETURN
      RETURN
      END
```

Module RIV5BD

Narrative for Module RIV5BD

RIV5BD calculates flow rates and volumes for water moving between the ground-water flow system and rivers. RIV5BD performs its functions as follows:

1. Initialize values. Total flow rates in and out of the model are set to 0. Flag IBD is set to indicate how cell-by-cell flows will be written. If ICBCFL is 0, no flows are written (IBD=0). If ICBCFL is not 0, flows are written as follows:

   If IRIVCB is less than 0, flows are written to the listing file (IBD=-1).
   If IRIVCB is greater than 0 and ICBCFL is 1, flows are written to unit IRIVCB as a 3-D array (IBD=1).
   If IRIVCB is greater than 0 and ICBCFL is 2, flows are written to unit IRIVCB as a list (IBD=2).

2. If cell-by-cell flows are to be written as a list, call UBDSV2 to write header information.

3. Clear the 3-D buffer that is used to store flow rates for each cell in the grid. Even if these values are not written to a file, they are returned by RIV5BD for possible use by other modules.

4. If there are no river reaches, skip to step 7.

5. Loop through each river reach calculating flow.

   A. Get layer, row, and column locations for river reach, and initialize the flow rate to 0.

   B. Skip to step 5L if the cell is no-flow or constant-head, which means there is no flow to the river for this reach.

   C. Get river parameters from the RIVR array, and define variables. RRBOT is the double precision value of the elevation of the bottom of the riverbed. Precision of values is carefully controlled in order that calculations are done using the same precision that the solvers use and to avoid mixed mode expressions.

   D. Compare head at the node containing the river reach to the elevation of the bottom of the riverbed.

   E. Head at the node is greater than elevation of the bottom of the riverbed. Calculate a head-dependent flow. Flow is CRIV(HRIV-HNEW); however, this is calculated as CRIV*HRIV - CRIV*HNEW so that budget calculations will be formulated like the flow equation is formulated. This can be important for simulations that can barely be solved due to a computer's limited precision.

   F. Head at the node is less than or equal to the elevation of the bottom of the riverbed. Calculate a constant flow that is independent of head at the node.

   G. Write the flow rate to the listing file if IBD is less than 0. Before writing the flow for the first reach, write a heading.

H. Add the flow rate to the 3-D buffer.

I. In order to accumulate inflows separately from outflows, check the sign of the flow.

J. Flow is negative, which indicates flow into the river. Subtract flow from total outflow.

K. Flow is positive, which indicates flow into the model node. Add flow to total inflow.

L. If IBD is 2, call UBDSVA to save flow to a file. If flow is being returned in the RIVR array, copy flow to RIVR. This is the end of the loop that is invoked for each river reach.

6. If IBD is 1, call UBUDSV to save flow as a 3-D array.

7. Move total flow rates and volumes into the global array of budget terms for the model budget. Also, define the name of the river budget term.

8. Increment the budget term counter.

9. RETURN.

```
      SUBROUTINE RIV5BD(NRIVER,MXRIVR,RIVR,IBOUND,HNEW,
     1      NCOL,NROW,NLAY,DELT,VBVL,VBNM,MSUM,KSTP,KPER,IRIVCB,
     2      ICBCFL,BUFF,IOUT,PERTIM,TOTIM,NRIVVL,IRIVAL)
C-----VERSION 1422 05APRIL1993 RIV5BD
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR RIVERS
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM),TEXT
      DOUBLE PRECISION HNEW,HHNEW,CHRIV,RRBOT,CCRIV,RATIN,RATOUT,RRATE
      DIMENSION RIVR(NRIVVL,MXRIVR),IBOUND(NCOL,NROW,NLAY),
     1          HNEW(NCOL,NROW,NLAY),VBVL(4,MSUM),BUFF(NCOL,NROW,NLAY)
C
      DATA TEXT /'   RIVER LEAKAGE'/
C     ------------------------------------------------------------------
C
C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATORS (RATIN AND RATOUT).
      ZERO=0.
      RATIN=ZERO
      RATOUT=ZERO
      IBD=0
      IF(IRIVCB.LT.0 .AND. ICBCFL.NE.0) IBD=-1
      IF(IRIVCB.GT.0) IBD=ICBCFL
      IBDLBL=0
C
C2------IF CELL-BY-CELL FLOWS WILL BE SAVED AS A LIST, WRITE HEADER.
      IF(IBD.EQ.2) CALL UBDSV2(KSTP,KPER,TEXT,IRIVCB,NCOL,NROW,NLAY,
     1          NRIVER,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C3------CLEAR THE BUFFER.
      DO 50 IL=1,NLAY
      DO 50 IR=1,NROW
      DO 50 IC=1,NCOL
      BUFF(IC,IR,IL)=ZERO
50    CONTINUE
C
C4------IF NO REACHES, SKIP FLOW CALCULATIONS.
      IF(NRIVER.EQ.0)GO TO 200
C
C5------LOOP THROUGH EACH RIVER REACH CALCULATING FLOW.
      DO 100 L=1,NRIVER
C
C5A-----GET LAYER, ROW & COLUMN OF CELL CONTAINING REACH.
      IL=RIVR(1,L)
      IR=RIVR(2,L)
      IC=RIVR(3,L)
      RATE=ZERO
C
C5B-----IF CELL IS NO-FLOW OR CONSTANT-HEAD MOVE ON TO NEXT REACH.
      IF(IBOUND(IC,IR,IL).LE.0)GO TO 99
C
C5C-----GET RIVER PARAMETERS FROM RIVER LIST.
      HRIV=RIVR(4,L)
      CRIV=RIVR(5,L)
      RBOT=RIVR(6,L)
      RRBOT=RBOT
      HHNEW=HNEW(IC,IR,IL)
C
C5D-----COMPARE HEAD IN AQUIFER TO BOTTOM OF RIVERBED.
      IF(HHNEW.GT.RRBOT) THEN
C
C5E-----AQUIFER HEAD > BOTTOM THEN RATE=CRIV*(HRIV-HNEW).
         CCRIV=CRIV
         CHRIV=CRIV*HRIV
         RRATE=CHRIV - CCRIV*HHNEW
         RATE=RRATE
C
C5F-----AQUIFER HEAD < BOTTOM THEN RATE=CRIV*(HRIV-RBOT).
      ELSE
         RATE=CRIV*(HRIV-RBOT)
         RRATE=RATE
      END IF
C
```

```
C5G-----PRINT THE INDIVIDUAL RATES IF REQUESTED(IRIVCB<0).
      IF(IBD.LT.0) THEN
         IF(IBDLBL.EQ.0) WRITE(IOUT,61) TEXT,KPER,KSTP
   61    FORMAT(1X,/1X,A,'   PERIOD',I3,'   STEP',I3)
         WRITE(IOUT,62) L,IL,IR,IC,RATE
   62    FORMAT(1X,'REACH',I4,'   LAYER',I3,'   ROW',I4,'   COL',I4,
     1        '   RATE',1PG15.6)
         IBDLBL=1
      END IF
C
C5H------ADD RATE TO BUFFER.
      BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+RATE
C
C5I-----SEE IF FLOW IS INTO AQUIFER OR INTO RIVER.
      IF(RATE)94,99,96
C
C5J-----AQUIFER IS DISCHARGING TO RIVER SUBTRACT RATE FROM RATOUT.
   94 RATOUT=RATOUT-RRATE
      GO TO 99
C
C5K-----AQUIFER IS RECHARGED FROM RIVER; ADD RATE TO RATIN.
   96 RATIN=RATIN+RRATE
C
C5L-----IF SAVING CELL-BY-CELL FLOWS IN LIST, WRITE FLOW.  OR IF
C5L-----RETURNING THE FLOW IN THE RIVR ARRAY, COPY FLOW TO RIVR.
   99 IF(IBD.EQ.2) CALL UBDSVA(IRIVCB,NCOL,NROW,IC,IR,IL,RATE,IBOUND,
     1                         NLAY)
      IF(IRIVAL.NE.0) RIVR(NRIVVL,L)=RATE
  100 CONTINUE
C
C6------IF CELL-BY-CELL FLOW WILL BE SAVED AS A 3-D ARRAY,
C6------CALL UBUDSV TO SAVE THEM.
      IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IRIVCB,BUFF,NCOL,NROW,
     1                         NLAY,IOUT)
C
C7------MOVE RATES,VOLUMES & LABELS INTO ARRAYS FOR PRINTING.
  200 RIN=RATIN
      ROUT=RATOUT
      VBVL(3,MSUM)=RIN
      VBVL(4,MSUM)=ROUT
      VBVL(1,MSUM)=VBVL(1,MSUM)+RIN*DELT
      VBVL(2,MSUM)=VBVL(2,MSUM)+ROUT*DELT
      VBNM(MSUM)=TEXT
C
C8------INCREMENT BUDGET TERM COUNTER.
      MSUM=MSUM+1
C
C9------RETURN.
      RETURN
      END
```

List of Variables for Module RIV5BD

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
CCRIV	Module	Double precision equivalent of CRIV.
CHRIV	Module	CRIV times HRIV.
CRIV	Module	Riverbed conductance for a river reach.
DELT	Global	Length of the current time step.
HHNEW	Module	HNEW(IC,IR,IL).
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
HRIV	Module	Stage in a river reach.
IBD	Module	Cell-by-cell budget flag, which is a composit of IRIVCB and ICBCFL: = -1, budget will be printed in the listing file. = 0, budget will not be saved or printed. = 1, budget will be saved by Module UBUDSV. = 2, budget will be saved by Modules UBDSV2 and UBDSVA..
IBDLBL	Module	Flag used when printing cell-by-cell budget values in the listing file so that the budget label is printed only once.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IC	Module	Index for columns.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IL	Module	Index for layers.
IOUT	Global	Unit number for writing to the listing file.
IR	Module	Index for rows.
IRIVAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in RIVR array to return budget values. ≠ 0, memory has been allocated in RIVR array to return budget values.
IRIVCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
L	Module	Index for river reaches.
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.

MXRIVR	Package	The maximum number of river reaches active at one time.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NRIVER	Package	Number of river reaches active in the current stress period.
NRIVVL	Package	The size of the first dimension of the RIVR array; that is, RIVR has dimensions of (NRIVVL,MXRIVR).
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
RATE	Module	Flow from a river reach into a cell.
RATIN	Module	Accumulator for flow into the model from rivers
RATOUT	Module	Accumulator for flow out of the model to rivers.
RBOT	Module	Elevation of the bottom of the riverbed for a river reach.
RIN	MOdule	Single precision equivalent of RATIN.
RIVR	Module	DIMENSION (NRIVVL,MXRIVR), For each river reach: layer, row, column, river head, riverbed conductance, elevation of the bottom of the riverbed, optional auxiliary parameters, and optional cell-by-cell budget flow.
ROUT	Module	Single precision equivalent of RATOUT.
RRATE	Module	Double precision equivalent of RATE.
RRBOT	Module	Double precision equivalent of RBOT.
TEXT	Module	CHARACTER*16, Label that identifies river budget data.
TOTIM	Global	Elapsed time in the simulation.
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
ZERO	Module	The constant 0.

Budget calculations in the Recharge (RCH) Package have been changed to double precision.

Module RCH5AL

```
      SUBROUTINE RCH5AL(ISUM,LENX,LCIRCH,LCRECH,NRCHOP,
     1                  NCOL,NROW,IN,IOUT,IRCHCB,IFREFM)
C
C-----VERSION 1512 20FEB1996 RCH5AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR RECHARGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE.
      WRITE(IOUT,1)IN
    1 FORMAT(1X,/1X,'RCH5 -- RECHARGE PACKAGE, VERSION 5, 6/1/95',
     1' INPUT READ FROM UNIT',I3)
C
C2------READ NRCHOP AND IRCHCB.
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(2I10)') NRCHOP,IRCHCB
      ELSE
         READ(IN,*) NRCHOP,IRCHCB
      END IF
C
C3------CHECK TO SEE THAT OPTION IS LEGAL.
      IF(NRCHOP.GE.1.AND.NRCHOP.LE.3)GO TO 200
C
C3A-----IF ILLEGAL PRINT A MESSAGE AND ABORT SIMULATION
      WRITE(IOUT,8)
    8 FORMAT(1X,'ILLEGAL OPTION CODE. SIMULATION ABORTING')
      STOP
C
C4------IF OPTION IS LEGAL PRINT OPTION CODE.
  200 IRK=ISUM
      IF(NRCHOP.EQ.1) WRITE(IOUT,201)
  201 FORMAT(1X,'OPTION 1 -- RECHARGE TO TOP LAYER')
      IF(NRCHOP.EQ.2) WRITE(IOUT,202)
  202 FORMAT(1X,'OPTION 2 -- RECHARGE TO ONE SPECIFIED NODE IN EACH',
     1       ' VERTICAL COLUMN')
      IF(NRCHOP.EQ.3) WRITE(IOUT,203)
  203 FORMAT(1X,'OPTION 3 -- RECHARGE TO HIGHEST ACTIVE NODE IN EACH',
     1       ' VERTICAL COLUMN')
C
C5------IF CELL-BY-CELL FLOWS ARE TO BE SAVED, THEN PRINT UNIT NUMBER.
      IF(IRCHCB.GT.0) WRITE(IOUT,204) IRCHCB
  204 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE SAVED ON UNIT',I3)
C
C6------ALLOCATE SPACE FOR THE RECHARGE ARRAY(RECH).
      LCRECH=ISUM
      ISUM=ISUM+NCOL*NROW
C
C7------IF OPTION 2 OR 3, ALLOCATE SPACE FOR INDICATOR ARRAY(IRCH)
      LCIRCH=ISUM
      IF(NRCHOP.EQ.2 .OR. NRCHOP.EQ.3) ISUM=ISUM+NCOL*NROW
C
C8------CALCULATE AND PRINT AMOUNT OF SPACE USED BY RECHARGE.
      IRK=ISUM-IRK
      WRITE(IOUT,4)IRK
    4 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY RCH')
      ISUM1=ISUM-1
      WRITE(IOUT,5)ISUM1,LENX
    5 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX)WRITE(IOUT,6)
    6 FORMAT(1X,'   ***X ARRAY MUST BE MADE LARGER***')
C
C9------RETURN
      RETURN
      END
```

RCH5RP

```
      SUBROUTINE RCH5RP(NRCHOP,IRCH,RECH,DELR,DELC,NROW,NCOL,
     1                  IN,IOUT,IFREFM)
C
C-----VERSION 1514 20FEB1996 RCH5RP
C     ******************************************************************
C     READ RECHARGE RATES
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*24 ANAME(2)
      DIMENSION IRCH(NCOL,NROW),RECH(NCOL,NROW),DELR(NCOL),DELC(NROW)
C
      DATA ANAME(1) /'    RECHARGE LAYER INDEX'/
      DATA ANAME(2) /'                RECHARGE'/
C     ------------------------------------------------------------------
C
C1------READ FLAGS SHOWING WHETHER DATA IS TO BE REUSED.
      IF(NRCHOP.EQ.2) THEN
         IF(IFREFM.EQ.0) THEN
            READ(IN,'(2I10)') INRECH,INIRCH
         ELSE
            READ(IN,*) INRECH,INIRCH
         END IF
      ELSE
         IF(IFREFM.EQ.0) THEN
            READ(IN,'(I10)') INRECH
         ELSE
            READ(IN,*) INRECH
         END IF
      END IF
C
C2------TEST INRECH TO SEE WHERE RECH IS COMING FROM.
      IF(INRECH.GE.0)GO TO 32
C
C2A-----IF INRECH<0 THEN REUSE RECHARGE ARRAY FROM LAST STRESS PERIOD
      WRITE(IOUT,3)
    3 FORMAT(1X,/1X,'REUSING RECH FROM LAST STRESS PERIOD')
      GO TO 55
C
C3------IF INRECH=>0 THEN CALL U2DREL TO READ RECHARGE RATE.
   32 CALL U2DREL(RECH,ANAME(2),NROW,NCOL,0,IN,IOUT)
C
C4------MULTIPLY RECHARGE RATE BY CELL AREA TO GET VOLUMETRIC RATE.
      DO 50 IR=1,NROW
      DO 50 IC=1,NCOL
      RECH(IC,IR)=RECH(IC,IR)*DELR(IC)*DELC(IR)
   50 CONTINUE
C
C5------IF NRCHOP=2 THEN A LAYER INDICATOR ARRAY IS NEEDED.
  55  IF (NRCHOP.NE.2)GO TO 60
C
C6------IF INIRCH<0 THEN REUSE LAYER INDICATOR ARRAY.
      IF(INIRCH.GE.0)GO TO 58
      WRITE(IOUT,2)
    2 FORMAT(1X,/1X,'REUSING IRCH FROM LAST STRESS PERIOD')
      GO TO 60
C
C7------IF INIRCH=>0 CALL U2DINT TO READ LAYER IND ARRAY(IRCH)
   58 CALL U2DINT(IRCH,ANAME(1),NROW,NCOL,0,IN,IOUT)
C
C8------RETURN
   60 RETURN
      END
```

RCH5FM

```
      SUBROUTINE RCH5FM(NRCHOP,IRCH,RECH,RHS,IBOUND,NCOL,
     1                       NROW,NLAY)
C
C-----VERSION 1404 12MAY1987 RCH5FM
C     ******************************************************************
C     SUBTRACT RECHARGE FROM RHS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION IRCH(NCOL,NROW),RECH(NCOL,NROW),
     1          RHS(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------IF NRCHOP IS 1 RECHARGE IS IN TOP LAYER. LAYER INDEX IS 1.
      IF(NRCHOP.NE.1) GO TO 15
C
      DO 10 IR=1,NROW
      DO 10 IC=1,NCOL
C
C1A-----IF CELL IS EXTERNAL THERE IS NO RECHARGE INTO IT.
      IF(IBOUND(IC,IR,1).LE.0)GO TO 10
C
C1B-----SUBTRACT RECHARGE RATE FROM RIGHT-HAND-SIDE.
      RHS(IC,IR,1)=RHS(IC,IR,1)-RECH(IC,IR)
   10 CONTINUE
      GO TO 100
C
C2------IF OPTION IS 2 THEN RECHARGE IS INTO LAYER IN INDICATOR ARRAY
   15 IF(NRCHOP.NE.2)GO TO 25
      DO 20 IR=1,NROW
      DO 20 IC=1,NCOL
C
C2A-----LAYER INDEX IS IN INDICATOR ARRAY.
      IL=IRCH(IC,IR)
C
C2B-----IF THE CELL IS EXTERNAL THERE IS NO RECHARGE INTO IT.
      IF(IBOUND(IC,IR,IL).LE.0)GO TO 20
C
C2C-----SUBTRACT RECHARGE FROM RIGHT-HAND-SIDE.
      RHS(IC,IR,IL)=RHS(IC,IR,IL)-RECH(IC,IR)
   20 CONTINUE
      GO TO 100
C
C3------IF OPTION IS 3 RECHARGE IS INTO HIGHEST INTERNAL CELL.
   25 IF(NRCHOP.NE.3)GO TO 100
C        CANNOT PASS THROUGH CONSTANT HEAD NODE
      DO 30 IR=1,NROW
      DO 30 IC=1,NCOL
      DO 28 IL=1,NLAY
C
C3A-----IF CELL IS CONSTANT HEAD MOVE ON TO NEXT HORIZONTAL LOCATION.
      IF(IBOUND(IC,IR,IL).LT.0) GO TO 30
C
C3B-----IF CELL IS INACTIVE MOVE DOWN A LAYER.
      IF (IBOUND(IC,IR,IL).EQ.0)GO TO 28
C
C3C-----SUBTRACT RECHARGE FROM RIGHT-HAND-SIDE.
      RHS(IC,IR,IL)=RHS(IC,IR,IL)-RECH(IC,IR)
      GO TO 30
   28 CONTINUE
   30 CONTINUE
  100 CONTINUE
C
C4------RETURN
      RETURN
      END
```

Module RCH5BD

Narrative for Module RCH5BD

RCH5BD calculates flow rates and volumes for water added to the ground-water flow system from areal recharge. RCH5BD performs its functions as follows:

1. Set total flow rates in and out of the model to 0.

2. Clear the 3-D buffer that is used to store flow rates for each cell in the grid. Even if these values are not written to a file, they are returned by RCH5BD for possible use by other modules. Flag IBD is set to indicate how cell-by-cell flows will be written. If IRCHCB is less than or equal to 0, no flows are written (IBD=0). If IRCHCB is greater than 0, then IBD is set equal to ICBCFL. The value of ICBCFL, which is specified in the Output Control Option, determines how flows are written as follows:

    0 -- flows are not written
    1 -- flows are written as a 3-D array to unit IRCHCB.
    2 -- flows are written to unit IRCHCB as a 1-layer array.

3. If the recharge option (NRCHOP) is 1, recharge is to layer 1. Loop through all cells in layer 1 and accumulate recharge. If NRCHOP is not 1, skip to step 4.

    A. If cell is no flow or constant head, skip that cell. Otherwise, set Q equal to the recharge flow.

    B. Add recharge to the 3-D cell-by-cell budget array, BUFF.

    C. Add positive recharge to RATIN and negative recharge to RATOUT. Skip to step 6 when done with the loop.

4. If the recharge option (NRCHOP) is 2, recharge is to layers specified in the IRCH indicator array. Loop through all rows and columns and accumulate recharge. If NRCHOP is not 2, skip to step 5.

    A. Get the layer number for a row and column location from IRCH.

    B. If cell is no flow or constant head, skip that cell. Otherwise, set Q equal to the recharge flow.

    C. Add recharge to the 3-D cell-by-cell budget array, BUFF.

    D. Add positive recharge to RATIN and negative recharge to RATOUT. Skip to step 6 when done with the loop.

5. The recharge option is 3 by elimination, which means that recharge is to the highest cell in each vertical column that is not no flow. Recharge will not pass through a constant-head cell. Loop through each horizontal cell.

A. For each vertical column, the layer in which recharge is applied will be stored in array IRCH. Initialize IRCH to 1. Loop through all cells in a vertical column.

B. If the cell is constant head, recharge is assumed to be combined with the constant-head source. Move to next horizontal location.

C. If cell is no flow, skip down to next lower cell.

D. Cell is variable head; set Q equal to the recharge flow. Add recharge to the 3-D cell-by-cell budget array, BUFF. Put the layer number of the cell that receives the recharge in IRCH.

E. Add positive recharge to RATIN and negative recharge to RATOUT. Move to next horizontal location.

6. Call the appropriate utility module to write cell-by-cell flows; call UBUDSV for a 3-D array and UBDSV3 for 1-layer array.

7. Move total flow rates into the global array of budget terms for the model budget.

8. Add the total flow volumes for the time step to the global array of budget terms for the model budget.

9. Define the name of the recharge budget term.

10. Increment the budget term counter.

11. RETURN.

RCH5BD

```
      SUBROUTINE RCH5BD(NRCHOP,IRCH,RECH,IBOUND,NROW,NCOL,NLAY,
     1    DELT,VBVL,VBNM,MSUM,KSTP,KPER,IRCHCB,ICBCFL,BUFF,IOUT,
     2    PERTIM,TOTIM)
C-----VERSION 1519 18DEC1992 RCH5BD
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR RECHARGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION RATIN,RATOUT,QQ
      CHARACTER*16 VBNM(MSUM),TEXT
      DIMENSION IRCH(NCOL,NROW),RECH(NCOL,NROW),
     1          IBOUND(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY),
     2          VBVL(4,MSUM)
      DATA TEXT /'        RECHARGE'/
C     ------------------------------------------------------------------
C
C1------CLEAR THE RATE ACCUMULATORS.
      ZERO=0.
      RATIN=ZERO
      RATOUT=ZERO
C
C2------CLEAR THE BUFFER & SET FLAG FOR SAVING CELL-BY-CELL FLOW TERMS.
      DO 2 IL=1,NLAY
      DO 2 IR=1,NROW
      DO 2 IC=1,NCOL
      BUFF(IC,IR,IL)=ZERO
2     CONTINUE
      IBD=0
      IF(IRCHCB.GT.0) IBD=ICBCFL
C
C3------IF NRCHOP=1 RECH GOES INTO LAYER 1. PROCESS EACH HORIZONTAL
C3------CELL LOCATION.
      IF(NRCHOP.NE.1) GO TO 15
      DO 10 IR=1,NROW
      DO 10 IC=1,NCOL
C
C3A-----IF CELL IS EXTERNAL THEN DO NOT DO BUDGET FOR IT.
      IF(IBOUND(IC,IR,1).LE.0)GO TO 10
      Q=RECH(IC,IR)
      QQ=Q
C
C3B-----ADD RECH TO BUFF.
      BUFF(IC,IR,1)=Q
C
C3C-----IF RECH POSITIVE ADD IT TO RATIN ELSE ADD IT TO RATOUT.
      IF(Q) 8,10,7
    7 RATIN=RATIN+QQ
      GO TO 10
    8 RATOUT=RATOUT-QQ
   10 CONTINUE
      GO TO 100
C
C4------IF NRCHOP=2 RECH IS IN LAYER SHOWN IN INDICATOR ARRAY(IRCH).
C4------PROCESS HORIZONTAL CELL LOCATIONS ONE AT A TIME.
   15 IF(NRCHOP.NE.2) GO TO 24
      DO 20 IR=1,NROW
      DO 20 IC=1,NCOL
C
C4A-----GET LAYER INDEX FROM INDICATOR ARRAY(IRCH).
      IL=IRCH(IC,IR)
C
C4B-----IF CELL IS EXTERNAL DO NOT CALCULATE BUDGET FOR IT.
      IF(IBOUND(IC,IR,IL).LE.0)GO TO 20
      Q=RECH(IC,IR)
      QQ=Q
C
C4C-----ADD RECHARGE TO BUFFER.
      BUFF(IC,IR,IL)=Q
C
C4D-----IF RECHARGE IS POSITIVE ADD TO RATIN ELSE ADD IT TO RATOUT.
      IF(Q) 18,20,17
   17 RATIN=RATIN+QQ
      GO TO 20
   18 RATOUT=RATOUT-QQ
   20 CONTINUE
      GO TO 100
```

```
C
C5------OPTION=3; RECHARGE IS INTO HIGHEST CELL IN A VERTICAL COLUMN
C5------THAT IS NOT NO FLOW.  PROCESS HORIZONTAL CELL LOCATIONS ONE
C5------AT A TIME.
24    DO 30 IR=1,NROW
      DO 29 IC=1,NCOL
C
C5A-----INITIALIZE IRCH TO 1, AND LOOP THROUGH CELLS IN A VERTICAL
C5A-----COLUMN TO FIND WHERE TO PLACE RECHARGE.
      IRCH(IC,IR)=1
      DO 28 IL=1,NLAY
C
C5B-----IF CELL IS CONSTANT HEAD MOVE ON TO NEXT HORIZONTAL LOCATION.
      IF(IBOUND(IC,IR,IL).LT.0) GO TO 29
C
C5C-----IF CELL IS INACTIVE MOVE DOWN TO NEXT CELL.
      IF (IBOUND(IC,IR,IL).EQ.0) GO TO 28
C
C5D-----CELL IS VARIABLE HEAD, SO APPLY RECHARGE TO IT.  ADD RECHARGE TO
C5D-----BUFFER, AND STORE LAYER NUMBER IN IRCH.
      Q=RECH(IC,IR)
      QQ=Q
      BUFF(IC,IR,IL)=Q
      IRCH(IC,IR)=IL
C
C5E-----IF RECH IS POSITIVE ADD IT TO RATIN ELSE ADD IT TO RATOUT.
      IF(Q) 27,29,26
   26 RATIN=RATIN+QQ
      GO TO 29
   27 RATOUT=RATOUT-QQ
      GO TO 29
28    CONTINUE
29    CONTINUE
30    CONTINUE
C
C
C6------IF CELL-BY-CELL FLOW TERMS SHOULD BE SAVED, CALL APPROPRIATE
C6------UTILITY MODULE TO WRITE THEM.
100   IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IRCHCB,BUFF,NCOL,NROW,
     1                         NLAY,IOUT)
      IF(IBD.EQ.2) CALL UBDSV3(KSTP,KPER,TEXT,IRCHCB,BUFF,IRCH,NRCHOP,
     1                  NCOL,NROW,NLAY,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C7------MOVE TOTAL RECHARGE RATE INTO VBVL FOR PRINTING BY BAS1OT.
      ROUT=RATOUT
      RIN=RATIN
      VBVL(4,MSUM)=ROUT
      VBVL(3,MSUM)=RIN
C
C8------ADD RECHARGE FOR TIME STEP TO RECHARGE ACCUMULATOR IN VBVL.
      VBVL(2,MSUM)=VBVL(2,MSUM)+ROUT*DELT
      VBVL(1,MSUM)=VBVL(1,MSUM)+RIN*DELT
C
C9------MOVE BUDGET TERM LABELS TO VBNM FOR PRINT BY MODULE BAS_OT.
      VBNM(MSUM)=TEXT
C
C10-----INCREMENT BUDGET TERM COUNTER.
      MSUM=MSUM+1
C
C11-----RETURN
      RETURN
      END
```

List of Variables for Module RCH5BD

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
DELT	Global	Length of the current time step.
IBD	Module	Cell-by-cell budget flag, which is a composit of IRCHCB and ICBCFL: = 0, budget will not be saved or printed. = 1, budget will be saved by Module UBUDSV. = 2, budget will be saved by Module UBDSV3.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IC	Module	Index for columns.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IL	Module	Index for layers.
IOUT	Global	Unit number for writing to the listing file.
IR	Module	Index for rows.
IRCH	Package	DIMENSION (NCOL,NROW), Layer number to which recharge will be applied if the recharge option (NRCHOP) is 2.
IRCHCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. < 0 or = 0, cell-by-cell budget flow will not be saved or written to the listing file.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NRCHOP	Package	Recharge option: = 1, recharge is to the top layer. = 2, layer number for recharge for each horizontal cell location is specified in IRCH. = 3, recharge is to the uppermost variable-head or constant-head cell at each horizontal cell location.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
Q	Module	Flow into a cell from recharge.
QQ	Module	Double precision equivalent of Q.
RATIN	Module	Accumulator for flow into the model from recharge.
RATOUT	Module	Accumulator for flow out of the model from recharge.
RECH	Package	DIMENSION (NCOL,NROW), Recharge flow rate.
RIN	Module	Single precision equivalent of RATIN.

ROUT	Module	Single precision equivalent of RATOUT.
TEXT	Module	CHARACTER*16, Label that identifies recharge budget data.
TOTIM	Global	Elapsed time in the simulation.
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
ZERO	Module	The constant 0.

Well Package

Budget calculations in the Well (WEL) Package have been changed to double precision. In addition, the ability to read extra data parameters for each well has been added.

Module WEL5AL

Narrative for Module WEL5AL

Module WEL5AL allocates space in the X array for well data. WEL5AL performs its functions as follows:

1. Write a message identifying the package. Also, initialize the number of wells to 0.

2. Read the first record from the WEL file into a buffer so it can be parsed by URWORD. Decode the maximum number of wells (MXWELL) in any stress period and the well cell-by-cell budget flag (IWELCB) using URWORD or READ depending on the free format flag (IFREFM).

3. Check for alphabetic options. "AUXILIARY" or "AUX" indicates an extra data parameter is to be read. "CBCALLOCATE" or "CBC" indicates that memory will be allocated to store the flow rate into the model from each well. Keep track of the total number of data values for each well in NWELVL, which is needed in order to calculate the required space in the X array.

4. Allocate space in the X array for the WELL array. Set LCWELL equal to ISUM, which is the lowest unused location in X. Add the size of WELL to ISUM.

5. Print the number of elements in the X array used by the WEL Package and the total space used in the X array.

6. RETURN.

```
      SUBROUTINE WEL5AL(ISUM,LENX,LCWELL,MXWELL,NWELLS,IN,IOUT,IWELCB,
     1        NWELVL,IWELAL,IFREFM)
C
C-----VERSION 0820 21FEB1996 WEL5AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR WELL PACKAGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      COMMON /WELCOM/WELAUX(5)
      CHARACTER*16 WELAUX
      CHARACTER*80 LINE
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE AND INITIALIZE NWELLS.
      WRITE(IOUT,1)IN
    1 FORMAT(1X,/1X,'WEL5 -- WELL PACKAGE, VERSION 5, 9/1/93',
     1' INPUT READ FROM UNIT',I3)
      NWELLS=0
C
C2------READ MAXIMUM NUMBER OF WELLS AND UNIT OR FLAG FOR
C2------CELL-BY-CELL FLOW TERMS.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(2I10)') MXWELL,IWELCB
         LLOC=21
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,MXWELL,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IWELCB,R,IOUT,IN)
      END IF
      WRITE(IOUT,3) MXWELL
    3 FORMAT(1X,'MAXIMUM OF',I5,' WELLS')
      IF(IWELCB.LT.0) WRITE(IOUT,7)
    7 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE PRINTED WHEN ICBCFL NOT 0')
      IF(IWELCB.GT.0) WRITE(IOUT,8) IWELCB
    8 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE SAVED ON UNIT',I3)
C
C3------READ AUXILIARY PARAMETERS AND CBC ALLOCATION OPTION.
      IWELAL=0
      NAUX=0
   10 CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
      IF(LINE(ISTART:ISTOP).EQ.'CBCALLOCATE' .OR.
     1   LINE(ISTART:ISTOP).EQ.'CBC') THEN
         IWELAL=1
         WRITE(IOUT,11)
   11    FORMAT(1X,'MEMORY IS ALLOCATED FOR CELL-BY-CELL BUDGET TERMS')
         GO TO 10
      ELSE IF(LINE(ISTART:ISTOP).EQ.'AUXILIARY' .OR.
     1        LINE(ISTART:ISTOP).EQ.'AUX') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
         IF(NAUX.LT.5) THEN
            NAUX=NAUX+1
            WELAUX(NAUX)=LINE(ISTART:ISTOP)
            WRITE(IOUT,12) WELAUX(NAUX)
   12       FORMAT(1X,'AUXILIARY WELL PARAMETER: ',A)
         END IF
         GO TO 10
      END IF
      NWELVL=4+NAUX+IWELAL
C
C4------ALLOCATE SPACE IN THE X ARRAY FOR THE WELL ARRAY.
      LCWELL=ISUM
      ISP=NWELVL*MXWELL
      ISUM=ISUM+ISP
C
C5------PRINT NUMBER OF SPACES IN X ARRAY USED BY WELL PACKAGE.
      WRITE(IOUT,14) ISP
   14 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY WEL')
      ISUM1=ISUM-1
      WRITE(IOUT,15) ISUM1,LENX
   15 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,16)
   16 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C6------RETURN
      RETURN
      END
```

List of Variables for Module WEL5AL

Variable	Range	Definition
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
IN	Package	Primary unit number from which input for this package is read.
IOUT	Global	Unit number for writing to the listing file.
ISP	Module	Number of elements allocated in the X array by this package.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ISUM	Global	Index of the lowest element in the X array which has not yet been allocated.
ISUM1	Module	ISUM - 1.
IWELAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in WELL array to return budget values. ≠ 0, memory has been allocated in WELL array to return budget values.
IWELCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
LCWELL	Package	Location in the X array of the first element of array WELL.
LENX	Global	The number of elements in the X array. LENX is defined in a PARAMETER statement in the MAIN program.
LINE	Module	CHARACTER*80, contents of a record that has been read from the package input file. LINE is parsed by URWORD.
LLOC	Module	Index that tells URWORD where to start looking for a word within LINE.
MXWELL	Package	The maximum number of wells active at one time.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NAUX	Module	Counter for the number of auxiliary well parameters.
NWELLS	Package	Number of wells active in the current stress period.
NWELVL	Package	The size of the first dimension of the WELL array; that is, WELL has dimensions of (NWELVL,MXWELL).
R	Module	Argument place holder for calls to URWORD in which the argument is unused.
WELAUX	Package	CHARACTER*16(5), names of auxiliary parameters.

Module WEL5RP

```
      SUBROUTINE WEL5RP(WELL,NWELLS,MXWELL,IN,IOUT,NWELVL,IWELAL,IFREFM)
C
C-----VERSION 0823 21FEB1996 WEL5RP
C     ******************************************************************
C     READ NEW WELL LOCATIONS AND STRESS RATES
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION WELL(NWELVL,MXWELL)
      COMMON /WELCOM/WELAUX(5)
      CHARACTER*16 WELAUX
      CHARACTER*151 LINE
C     ------------------------------------------------------------------
C
C1------READ ITMP(NUMBER OF WELLS OR FLAG SAYING REUSE WELL DATA).
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(I10)') ITMP
      ELSE
         READ(IN,*) ITMP
      END IF
      IF(ITMP.GE.0) GO TO 50
C
C1A-----IF ITMP LESS THAN ZERO REUSE DATA. PRINT MESSAGE AND RETURN.
      WRITE(IOUT,6)
    6 FORMAT(1X,/1X,'REUSING WELLS FROM LAST STRESS PERIOD')
      RETURN
C
C1B-----ITMP=>0.  SET NWELLS EQUAL TO ITMP.
   50 NWELLS=ITMP
      IF(NWELLS.LE.MXWELL) GO TO 100
C
C2------NWELLS>MXWELL.  PRINT MESSAGE. STOP.
      WRITE(IOUT,99) NWELLS,MXWELL
   99 FORMAT(1X,/1X,'NWELLS(',I4,') IS GREATER THAN MXWELL(',I4,')')
      STOP
C
C3------PRINT NUMBER OF WELLS IN CURRENT STRESS PERIOD.
  100 WRITE (IOUT,101) NWELLS
  101 FORMAT(1X,//1X,I5,' WELLS')
C
C4------IF THERE ARE NO ACTIVE WELLS IN THIS STRESS PERIOD THEN RETURN.
      IF(NWELLS.EQ.0) GO TO 260
C
C5------READ AND PRINT DATA FOR EACH WELL.
      NAUX=NWELVL-4-IWELAL
      MAXAUX=NWELVL-IWELAL
      IF(NAUX.GT.0) THEN
         WRITE(IOUT,103) (WELAUX(JJ),JJ=1,NAUX)
         WRITE(IOUT,104) ('-----------------',JJ=1,NAUX)
      ELSE
         WRITE(IOUT,103)
         WRITE(IOUT,104)
      END IF
  103 FORMAT(1X,/
     1       1X,'LAYER   ROW   COL   STRESS RATE   WELL NO.',:5(2X,A))
  104 FORMAT(1X,42('-'),5A)
      DO 250 II=1,NWELLS
C5A-----READ THE REQUIRED DATA WITH FIXED OR FREE FORMAT.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(3I10,F10.0)') K,I,J,WELL(4,II)
         LLOC=41
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,K,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,I,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,J,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,WELL(4,II),IOUT,IN)
      END IF
C5B-----READ ANY AUXILIARY DATA WITH FREE FORMAT, AND PRINT ALL VALUES.
      IF(NAUX.GT.0) THEN
         DO 110 JJ=1,NAUX
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,WELL(JJ+4,II),IOUT,IN)
  110    CONTINUE
         WRITE (IOUT,115) K,I,J,WELL(4,II),II,
     1         (WELL(JJ,II),JJ=5,MAXAUX)
```

```
      ELSE
         WRITE (IOUT,115) K,I,J,WELL(4,II),II
      END IF
  115 FORMAT(1X,I4,I7,I6,G15.5,I7,:5(2X,G16.5))
      WELL(1,II)=K
      WELL(2,II)=I
      WELL(3,II)=J
  250 CONTINUE
C
C6------RETURN
  260 RETURN
      END
```

Module WEL5FM

```
      SUBROUTINE WEL5FM(NWELLS,MXWELL,RHS,WELL,IBOUND,
     1        NCOL,NROW,NLAY,NWELVL)
C
C-----VERSION 1101 28AUG1992 WEL5FM
C
C     ******************************************************************
C     SUBTRACT Q FROM RHS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION RHS(NCOL,NROW,NLAY),WELL(NWELVL,MXWELL),
     1            IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C1------IF NUMBER OF WELLS <= 0 THEN RETURN.
      IF(NWELLS.LE.0) RETURN
C
C2------PROCESS EACH WELL IN THE WELL LIST.
      DO 100 L=1,NWELLS
      IR=WELL(2,L)
      IC=WELL(3,L)
      IL=WELL(1,L)
      Q=WELL(4,L)
C
C2A-----IF THE CELL IS INACTIVE THEN BYPASS PROCESSING.
      IF(IBOUND(IC,IR,IL).LE.0) GO TO 100
C
C2B-----IF THE CELL IS VARIABLE HEAD THEN SUBTRACT Q FROM
C       THE RHS ACCUMULATOR.
      RHS(IC,IR,IL)=RHS(IC,IR,IL)-Q
  100 CONTINUE
C
C3------RETURN
      RETURN
      END
```

Module WEL5BD

Narrative for Module WEL5BD

WEL5BD calculates flow rates and volumes for water moving between the ground-water flow system and wells. WEL5BD performs its functions as follows:

1. Initialize values. Total flow rates in and out of the model are set to 0. Flag IBD is set to indicate how cell-by-cell flows will be written. If ICBCFL is 0, no flows are written (IBD=0). If ICBCFL is not 0, flows are written as follows:
   If IWELCB is less than 0, flows are written to the listing file (IBD=-1).
   If IWELCB is greater than 0 and ICBCFL is 1, flows are written to unit IWELCB as a 3-D array (IBD=1).
   If IWELCB is greater than 0 and ICBCFL is 2, flows are written to unit IWELCB as a list (IBD=2).

2. If cell-by-cell flows are to be written as a list, call UBDSV2 to write header information.

3. Clear the 3-D buffer that is used to store flow rates for each cell in the grid. Even if these values are not written to a file, they are returned by WEL5BD for possible use by other modules.

4. If there are no wells, skip to step 7.

5. Loop through each well calculating flow.

   A. Get layer, row, and column locations for well, and initialize the flow rate to 0.

   B. Skip to step 5I if the cell is no-flow or constant-head, which means there is no well flow.

   C. Get flow rate from the WELL array.

   D. Write the flow rate to the listing file if IBD is less than 0. Before writing the flow for the first well, write a heading.

   E. Add the flow rate to the 3-D buffer.

   F. In order to accumulate inflows separately from outflows, check the sign of the flow.

   G. Flow is positive, which indicates flow from well into the model node. Add flow to total inflow.

   H. Flow is negative, which indicates flow into the well (pumping). Subtract flow from total outflow.

   I. If IBD is 2, call UBDSVA to save flow to a file. If flow is being returned in the WELL array, copy flow to WELL. This is the end of the loop that is invoked for each well.

6. If IBD is 1, call UBUDSV to save flow as a 3-D array.

7. Move total flow rates and volumes into the global array of budget terms for the model budget. Also, define the name of the well budget term.

8. Increment the budget term counter.

9. RETURN.

```
      SUBROUTINE WEL5BD(NWELLS,MXWELL,VBNM,VBVL,MSUM,WELL,IBOUND,DELT,
     1        NCOL,NROW,NLAY,KSTP,KPER,IWELCB,ICBCFL,BUFF,IOUT,
     2        PERTIM,TOTIM,NWELVL,IWELAL)
C-----VERSION 1120 16APRIL1993 WEL5BD
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR WELLS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM),TEXT
      DIMENSION VBVL(4,MSUM),WELL(NWELVL,MXWELL),IBOUND(NCOL,NROW,NLAY),
     1          BUFF(NCOL,NROW,NLAY)
      DOUBLE PRECISION RATIN,RATOUT,QQ
      DATA TEXT /'           WELLS'/
C     ------------------------------------------------------------------
C
C1------CLEAR RATIN AND RATOUT ACCUMULATORS, AND SET CELL-BY-CELL
C1------BUDGET FLAG.
      ZERO=0.
      RATIN=ZERO
      RATOUT=ZERO
      IBD=0
      IF(IWELCB.LT.0 .AND. ICBCFL.NE.0) IBD=-1
      IF(IWELCB.GT.0) IBD=ICBCFL
      IBDLBL=0
C
C2-----IF CELL-BY-CELL FLOWS WILL BE SAVED AS A LIST, WRITE HEADER.
      IF(IBD.EQ.2) CALL UBDSV2(KSTP,KPER,TEXT,IWELCB,NCOL,NROW,NLAY,
     1          NWELLS,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C3------CLEAR THE BUFFER.
      DO 50 IL=1,NLAY
      DO 50 IR=1,NROW
      DO 50 IC=1,NCOL
      BUFF(IC,IR,IL)=ZERO
50    CONTINUE
C
C4------IF THERE ARE NO WELLS, DO NOT ACCUMULATE FLOW.
      IF(NWELLS.EQ.0) GO TO 200
C
C5------LOOP THROUGH EACH WELL CALCULATING FLOW.
      DO 100 L=1,NWELLS
C
C5A-----GET LAYER, ROW & COLUMN OF CELL CONTAINING WELL.
      IR=WELL(2,L)
      IC=WELL(3,L)
      IL=WELL(1,L)
      Q=ZERO
C
C5B-----IF THE CELL IS NO-FLOW OR CONSTANT_HEAD, IGNORE IT.
      IF(IBOUND(IC,IR,IL).LE.0)GO TO 99
C
C5C-----GET FLOW RATE FROM WELL LIST.
      Q=WELL(4,L)
      QQ=Q
C
C5D-----PRINT FLOW RATE IF REQUESTED.
      IF(IBD.LT.0) THEN
         IF(IBDLBL.EQ.0) WRITE(IOUT,61) TEXT,KPER,KSTP
   61    FORMAT(1X,/1X,A,'   PERIOD',I3,'   STEP',I3)
         WRITE(IOUT,62) L,IL,IR,IC,Q
   62    FORMAT(1X,'WELL',I4,'   LAYER',I3,'   ROW',I4,'   COL',I4,
     1        '   RATE',1PG15.6)
         IBDLBL=1
      END IF
C
C5E-----ADD FLOW RATE TO BUFFER.
      BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+Q
C
C5F-----SEE IF FLOW IS POSITIVE OR NEGATIVE.
      IF(Q) 90,99,80
C
C5G-----FLOW RATE IS POSITIVE (RECHARGE). ADD IT TO RATIN.
   80 RATIN=RATIN+QQ
      GO TO 99
C
```

```
C5H-----FLOW RATE IS NEGATIVE (DISCHARGE). ADD IT TO RATOUT.
   90 RATOUT=RATOUT-QQ
C
C5I-----IF CELL-BY-CELL FLOWS ARE BEING SAVED AS A LIST, WRITE FLOW.
C5I-----OR IF RETURNING THE FLOW IN THE WELL ARRAY, COPY FLOW TO WELL.
   99 IF(IBD.EQ.2) CALL UBDSVA(IWELCB,NCOL,NROW,IC,IR,IL,Q,IBOUND,NLAY)
      IF(IWELAL.NE.0) WELL(NWELVL,L)=Q
  100 CONTINUE
C
C6------IF CELL-BY-CELL FLOWS WILL BE SAVED AS A 3-D ARRAY,
C6------CALL UBUDSV TO SAVE THEM.
      IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IWELCB,BUFF,NCOL,NROW,
     1                         NLAY,IOUT)
C
C7------MOVE RATES, VOLUMES & LABELS INTO ARRAYS FOR PRINTING.
  200 RIN=RATIN
      ROUT=RATOUT
      VBVL(3,MSUM)=RIN
      VBVL(4,MSUM)=ROUT
      VBVL(1,MSUM)=VBVL(1,MSUM)+RIN*DELT
      VBVL(2,MSUM)=VBVL(2,MSUM)+ROUT*DELT
      VBNM(MSUM)=TEXT
C
C8------INCREMENT BUDGET TERM COUNTER(MSUM).
      MSUM=MSUM+1
C
C9------RETURN
      RETURN
      END
```

List of Variables for Module WEL5BD

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
DELT	Global	Length of the current time step.
IBD	Module	Cell-by-cell budget flag, which is a composit of IWELCB and ICBCFL: = -1, budget will be printed in the listing file. = 0, budget will not be saved or printed. = 1, budget will be saved by Module UBUDSV. = 2, budget will be saved by Modules UBDSV2 and UBDSVA.
IBDLBL	Module	Flag used when printing cell-by-cell budget values in the listing file so that the budget label is printed only once.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IC	Module	Column index.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IL	Module	Layer index.
IOUT	Global	Unit number for writing to the listing file.
IR	Module	Row index.
IWELAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in WELL array to return budget values. ≠ 0, memory has been allocated in WELL array to return budget values.
IWELCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
L	Module	Index for wells.
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.
MXWELL	Package	The maximum number of wells active at one time.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
NWELLS	Package	Number of wells active in the current stress period.
NWELVL	Package	The size of the first dimension of the WELL array; that is, WELL has dimensions of (NWELVL,MXWELL).

PERTIM	Global	Elapsed time during the current stress period.
Q	Module	Recharge rate from a well.
QQ	Module	DOuble precision equivalent of Q.
RATIN	Module	Accumulator for flow into the model from wells.
RATOUT	Module	Accumulator for flow out of the model from wells.
RIN	Module	Single precision equivalent of RATIN.
ROUT	Module	Single precision equivalent of RATOUT.
TEXT	Module	CHARACTER*16, Label that identifies well budget data.
TOTIM	Global	Elapsed time in the simulation.
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
WELL	Package	DIMENSION (NWELVL,MXWELL), For each well: layer, row, column, recharge rate, optional auxiliary parameters, and optional cell-by-cell budget flow.
ZERO	Module	The constant 0.

Drain Package

Budget calculations in the Drain (DRN) Package have been changed to double precision. In addition, the ability to read extra data parameters for each drain has been added.

Module DRN5AL

Narrative for Module DRN5AL

Module DRN5AL allocates space in the X array for drain data. DRN5AL performs its functions as follows:

1. Write a message identifying the package. Also, initialize the number of drains to 0.

2. Read the first record from the DRN file into a buffer so it can be parsed by URWORD. Decode the maximum number of drains (MXDRN) in any stress period and the drain cell-by-cell budget flag (IDRNCB) using URWORD or READ depending on the free format flag (IFREFM).

3. Check for alphabetic options. "AUXILIARY" or "AUX" indicates an extra data parameter is to be read. "CBCALLOCATE" or "CBC" indicates that memory will be allocated to store the flow rate into the model from each drain. Keep track of the total number of data values for each drain in NDRNVL, which is needed in order to calculate the required space in the X array.

4. Allocate space in the X array for the DRAI array. Set LCDRAI equal to ISUM, which is the lowest unused location in X. Add the size of DRAI to ISUM.

5. Print the number of elements in the X array used by the DRN Package and the total space used in the X array.

6. RETURN.

```
      SUBROUTINE DRN5AL(ISUM,LENX,LCDRAI,NDRAIN,MXDRN,IN,IOUT,IDRNCB,
     1        NDRNVL,IDRNAL,IFREFM)
C
C-----VERSION 0841 21FEB1996 DRN5AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR DRAIN PACKAGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      COMMON /DRNCOM/DRNAUX(5)
      CHARACTER*16 DRNAUX
      CHARACTER*80 LINE
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE AND INITIALIZE NDRAIN.
      WRITE(IOUT,1)IN
    1 FORMAT(1X,/1X,'DRN5 -- DRAIN PACKAGE, VERSION 5, 9/1/93',
     1' INPUT READ FROM UNIT',I3)
      NDRAIN=0
C
C2------READ MAXIMUM NUMBER OF DRAINS AND UNIT OR FLAG FOR
C2------CELL-BY-CELL FLOW TERMS.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(2I10)') MXDRN,IDRNCB
         LLOC=21
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,MXDRN,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IDRNCB,R,IOUT,IN)
      END IF
      WRITE(IOUT,3) MXDRN
    3 FORMAT(1X,'MAXIMUM OF',I5,' DRAINS')
      IF(IDRNCB.LT.0) WRITE(IOUT,7)
    7 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE PRINTED WHEN ICBCFL NOT 0')
      IF(IDRNCB.GT.0) WRITE(IOUT,8) IDRNCB
    8 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE SAVED ON UNIT',I3)
C
C3------READ AUXILIARY PARAMETERS AND CBC ALLOCATION OPTION.
      IDRNAL=0
      NAUX=0
   10 CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
      IF(LINE(ISTART:ISTOP).EQ.'CBCALLOCATE' .OR.
     1   LINE(ISTART:ISTOP).EQ.'CBC') THEN
         IDRNAL=1
         WRITE(IOUT,11)
   11    FORMAT(1X,'MEMORY IS ALLOCATED FOR CELL-BY-CELL BUDGET TERMS')
         GO TO 10
      ELSE IF(LINE(ISTART:ISTOP).EQ.'AUXILIARY' .OR.
     1        LINE(ISTART:ISTOP).EQ.'AUX') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
         IF(NAUX.LT.5) THEN
            NAUX=NAUX+1
            DRNAUX(NAUX)=LINE(ISTART:ISTOP)
            WRITE(IOUT,12) DRNAUX(NAUX)
   12       FORMAT(1X,'AUXILIARY DRAIN PARAMETER: ',A)
         END IF
         GO TO 10
      END IF
      NDRNVL=5+NAUX+IDRNAL
C
C4------ALLOCATE SPACE IN THE X ARRAY FOR THE DRAI ARRAY.
      LCDRAI=ISUM
      ISP=NDRNVL*MXDRN
      ISUM=ISUM+ISP
C
C5------PRINT AMOUNT OF SPACE USED BY DRAIN PACKAGE.
      WRITE(IOUT,14) ISP
   14 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY DRN')
      ISUM1=ISUM-1
      WRITE(IOUT,15) ISUM1,LENX
   15 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,16)
   16 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C6------RETURN.
      RETURN
      END
```

List of Variables for Module DRN5AL

Variable	Range	Definition
DRNAUX	Package	CHARACTER*16(5), names of auxiliary parameters.
IDRNAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in DRAI array to return budget values. ≠ 0, memory has been allocated in DRAI array to return budget values.
IDRNCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
IN	Package	Primary unit number from which input for this package is read.
IOUT	Global	Unit number for writing to the listing file.
ISP	Module	Number of elements allocated in the X array by this package.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ISUM	Global	Index of the lowest element in the X array which has not yet been allocated.
ISUM1	Module	ISUM - 1.
LCDRAI	Package	Location in the X array of the first element of array DRAI.
LENX	Global	The number of elements in the X array. LENX is defined in a PARAMETER statement in the MAIN program.
LINE	Module	CHARACTER*80, contents of a record that has been read from the package input file. LINE is parsed by URWORD.
LLOC	Module	Index that tells URWORD where to start looking for a word within LINE.
MXDRN	Package	The maximum number of drains active at one time.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NAUX	Module	Counter for the number of auxiliary drain parameters.
NDRAIN	Package	Number of drains active in the current stress period.
NDRNVL	Package	The size of the first dimension of the DRAI array; that is, DRAI has dimensions of (NDRNVL,MXDRN).
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

DRN5RP

```
      SUBROUTINE DRN5RP(DRAI,NDRAIN,MXDRN,IN,IOUT,NDRNVL,IDRNAL,IFREFM)
C
C-----VERSION 0845 21FEB1996 DRN5RP
C     ******************************************************************
C     READ DRAIN LOCATIONS, ELEVATIONS, AND CONDUCTANCES
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION DRAI(NDRNVL,MXDRN)
      COMMON /DRNCOM/DRNAUX(5)
      CHARACTER*16 DRNAUX
      CHARACTER*151 LINE
C     ------------------------------------------------------------------
C
C1------READ ITMP (NUMBER OF DRAIN CELLS OR FLAG TO REUSE DATA).
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(I10)') ITMP
      ELSE
         READ(IN,*) ITMP
      END IF
C
C2------TEST ITMP.
      IF(ITMP.GE.0) GO TO 50
C
C2A-----IF ITMP<0 THEN REUSE DATA FROM LAST STRESS PERIOD.
      WRITE(IOUT,7)
    7 FORMAT(1X,/1X,'REUSING DRAINS FROM LAST STRESS PERIOD')
      RETURN
C
C3------IF ITMP=>0 THEN IT IS THE NUMBER OF DRAINS.
   50 NDRAIN=ITMP
      IF(NDRAIN.LE.MXDRN) GO TO 100
C
C4------IF NDRAIN>MXDRN THEN STOP.
      WRITE(IOUT,99) NDRAIN,MXDRN
   99 FORMAT(1X,/1X,'NDRAIN(',I4,') IS GREATER THAN MXDRN(',I4,')')
      STOP
C
C5------PRINT NUMBER OF DRAINS IN THIS STRESS PERIOD.
  100 WRITE(IOUT,101) NDRAIN
  101 FORMAT(1X,//1X,I5,' DRAINS')
C
C6------IF THERE ARE NO DRAINS THEN RETURN.
      IF(NDRAIN.EQ.0) GO TO 260
C
C7------READ AND PRINT DATA FOR EACH DRAIN.
      NAUX=NDRNVL-5-IDRNAL
      MAXAUX=NDRNVL-IDRNAL
      IF(NAUX.GT.0) THEN
         WRITE(IOUT,103) (DRNAUX(JJ),JJ=1,NAUX)
         WRITE(IOUT,104) ('----------------',JJ=1,NAUX)
      ELSE
         WRITE(IOUT,103)
         WRITE(IOUT,104)
      END IF
  103 FORMAT(1X,/1X,'LAYER   ROW   COL   ELEVATION   CONDUCTANCE   ',
     1          'DRAIN NO.',:5(2X,A))
  104 FORMAT(1X,55('-'),5A)
      DO 250 II=1,NDRAIN
C7A-----READ THE REQUIRED DATA WITH FIXED OR FREE FORMAT.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(3I10,2F10.0)') K,I,J,(DRAI(JJ,II),JJ=4,5)
         LLOC=51
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,K,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,I,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,J,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,DRAI(4,II),IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,DRAI(5,II),IOUT,IN)
      END IF
C7B-----READ ANY AUXILIARY DATA WITH FREE FORMAT, AND PRINT ALL VALUES.
      IF(NAUX.GT.0) THEN
         DO 110 JJ=1,NAUX
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,DRAI(JJ+5,II),IOUT,IN)
```

```
  110    CONTINUE
         WRITE (IOUT,115) K,I,J,DRAI(4,II),DRAI(5,II),II,
     1         (DRAI(JJ,II),JJ=6,MAXAUX)
       ELSE
         WRITE (IOUT,115) K,I,J,DRAI(4,II),DRAI(5,II),II
       END IF
  115 FORMAT(1X,I4,I7,I6,G13.4,G14.4,I8,:5(2X,G16.5))
       DRAI(1,II)=K
       DRAI(2,II)=I
       DRAI(3,II)=J
  250 CONTINUE
C
C8------RETURN.
  260 RETURN
C
       END
```

DRN5FM

```
      SUBROUTINE DRN5FM(NDRAIN,MXDRN,DRAI,HNEW,HCOF,RHS,IBOUND,
     1              NCOL,NROW,NLAY,NDRNVL)
C
C-----VERSION 1050 16JULY1992 DRN5FM
C     ******************************************************************
C     ADD DRAIN FLOW TO SOURCE TERM
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW,EEL
C
      DIMENSION DRAI(NDRNVL,MXDRN),HNEW(NCOL,NROW,NLAY),
     1         RHS(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),
     1         HCOF(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------IF NDRAIN<=0 THERE ARE NO DRAINS. RETURN.
      IF(NDRAIN.LE.0) RETURN
C
C2------PROCESS EACH CELL IN THE DRAIN LIST.
      DO 100 L=1,NDRAIN
C
C3------GET COLUMN, ROW AND LAYER OF CELL CONTAINING DRAIN.
      IL=DRAI(1,L)
      IR=DRAI(2,L)
      IC=DRAI(3,L)
C
C4-------IF THE CELL IS EXTERNAL SKIP IT.
      IF(IBOUND(IC,IR,IL).LE.0) GO TO 100
C
C5-------IF THE CELL IS INTERNAL GET THE DRAIN DATA.
      EL=DRAI(4,L)
      EEL=EL
C
C6------IF HEAD IS LOWER THAN DRAIN THEN SKIP THIS CELL.
      IF(HNEW(IC,IR,IL).LE.EEL) GO TO 100
C
C7------HEAD IS HIGHER THAN DRAIN. ADD TERMS TO RHS AND HCOF.
      C=DRAI(5,L)
      HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-C
      RHS(IC,IR,IL)=RHS(IC,IR,IL)-C*EL
  100 CONTINUE
C
C8------RETURN.
      RETURN
      END
```

Module DRN5BD

Narrative for Module DRN5BD

DRN5BD calculates flow rates and volumes for water moving from the ground-water flow system to drains. DRN5BD performs its functions as follows:

1. Initialize values. Total flow rate out of the model is set to 0. Flag IBD is set to indicate how cell-by-cell flows will be written. If ICBCFL is 0, no flows are written (IBD=0). If ICBCFL is not 0, flows are written as follows:
   If IDRNCB is less than 0, flows are written to the listing file (IBD=-1).
   If IDRNCB is greater than 0 and ICBCFL is 1, flows are written to unit IDRNCB as a 3-D array (IBD=1).
   If IDRNCB is greater than 0 and ICBCFL is 2, flows are written to unit IDRNCB as a list (IBD=2).

2. If cell-by-cell flows are to be written as a list, call UBDSV2 to write header information.

3. Clear the 3-D buffer that is used to store flow rates for each cell in the grid. Even if these values are not written to a file, they are returned by DRN5BD for possible use by other modules.

4. If there are no drains, skip to step 7.

5. Loop through each drain calculating flow.

   A. Get layer, row, and column locations for drain, and initialize the flow rate to 0.

   B. Skip to step 5G if the cell is no-flow or constant-head, which means there is no flow to this drain.

   C. Get drain parameters from the DRAI array, and define variables. EEL is the double precision value of the drain elevation. Precision of values is carefully controlled in order that calculations are done using the same precision that the solvers use and to avoid mixed mode expressions.

   D. If head at the node is greater than the drain elevation, then calculate a head-dependent flow. Flow is C(EL-HNEW); however, this is calculated as C*EL - C*HNEW so that budget calculations will be formulated like the flow equation is formulated. This can be important for simulations that can barely be solved due to a computer's limited precision. Subtract flow for this drain from total drain outflow (flow is negative, which indicates flow into the drain). If head at the node is less than or equal to the drain elevation, flow is 0; so no calculations are made

   E. Write the flow rate to the listing file if IBD is less than 0. Before writing the flow for the first drain, write a heading.

   F. Add the flow rate to the 3-D buffer.

G. If IBD is 2, call UBDSVA to save flow to a file. If returning flow in the DRAI array, copy flow to DRAI. This is the end of the loop that is invoked for each drain.

6. If IBD is 1, call UBUDSV to save flow as a 3-D array.

7. Move total flow rates and volume into the global array of budget terms for the model budget. Also, define the name of the drain budget term.

8. Increment the budget term counter.

9. RETURN.

DRN5BD

```
      SUBROUTINE DRN5BD(NDRAIN,MXDRN,VBNM,VBVL,MSUM,DRAI,DELT,HNEW,
     1        NCOL,NROW,NLAY,IBOUND,KSTP,KPER,IDRNCB,ICBCFL,BUFF,IOUT,
     2        PERTIM,TOTIM,NDRNVL,IDRNAL)
C-----VERSION 1052 06APRIL1993 DRN5BD
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR DRAINS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM),TEXT
      DOUBLE PRECISION HNEW,HHNEW,EEL,CC,CEL,RATOUT,QQ
C
      DIMENSION VBVL(4,MSUM),DRAI(NDRNVL,MXDRN),HNEW(NCOL,NROW,NLAY),
     1          IBOUND(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
C
      DATA TEXT /'          DRAINS'/
C     ------------------------------------------------------------------
C
C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATOR (RATOUT).
      ZERO=0.
      RATOUT=ZERO
      IBD=0
      IF(IDRNCB.LT.0 .AND. ICBCFL.NE.0) IBD=-1
      IF(IDRNCB.GT.0) IBD=ICBCFL
      IBDLBL=0
C
C2------IF CELL-BY-CELL FLOWS WILL BE SAVED AS A LIST, WRITE HEADER.
      IF(IBD.EQ.2) CALL UBDSV2(KSTP,KPER,TEXT,IDRNCB,NCOL,NROW,NLAY,
     1          NDRAIN,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C3------CLEAR THE BUFFER.
      DO 50 IL=1,NLAY
      DO 50 IR=1,NROW
      DO 50 IC=1,NCOL
      BUFF(IC,IR,IL)=ZERO
50    CONTINUE
C
C4------IF THERE ARE NO DRAINS THEN DO NOT ACCUMULATE DRAIN FLOW.
      IF(NDRAIN.LE.0) GO TO 200
C
C5------LOOP THROUGH EACH DRAIN CALCULATING FLOW.
      DO 100 L=1,NDRAIN
C
C5A-----GET LAYER, ROW & COLUMN OF CELL CONTAINING REACH.
      IL=DRAI(1,L)
      IR=DRAI(2,L)
      IC=DRAI(3,L)
      Q=ZERO
C
C5B-----IF CELL IS NO-FLOW OR CONSTANT-HEAD, IGNORE IT.
      IF(IBOUND(IC,IR,IL).LE.0) GO TO 99
C
C5C-----GET DRAIN PARAMETERS FROM DRAIN LIST.
      EL=DRAI(4,L)
      EEL=EL
      C=DRAI(5,L)
      HHNEW=HNEW(IC,IR,IL)
C
C5D-----IF HEAD HIGHER THAN DRAIN, CALCULATE Q=C*(EL-HHNEW).
C5D-----SUBTRACT Q FROM RATOUT.
      IF(HHNEW.GT.EEL) THEN
         CC=C
         CEL=C*EL
         QQ=CEL - CC*HHNEW
         Q=QQ
         RATOUT=RATOUT-QQ
      END IF
C
C5E-----PRINT THE INDIVIDUAL RATES IF REQUESTED(IDRNCB<0).
      IF(IBD.LT.0) THEN
         IF(IBDLBL.EQ.0) WRITE(IOUT,61) TEXT,KPER,KSTP
   61    FORMAT(1X,/1X,A,'   PERIOD',I3,'   STEP',I3)
         WRITE(IOUT,62) L,IL,IR,IC,Q
   62    FORMAT(1X,'DRAIN',I4,'   LAYER',I3,'   ROW',I4,'   COL',I4,
     1          '   RATE',1PG15.6)
         IBDLBL=1
      END IF
```

```
C
C5F-----ADD Q TO BUFFER.
      BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+Q
C
C5G-----IF SAVING CELL-BY-CELL FLOWS IN A LIST, WRITE FLOW.  OR IF
C5G-----RETURNING THE FLOW IN THE DRAI ARRAY, COPY FLOW TO DRAI.
   99 IF(IBD.EQ.2) CALL UBDSVA(IDRNCB,NCOL,NROW,IC,IR,IL,Q,IBOUND,NLAY)
      IF(IDRNAL.NE.0) DRAI(NDRNVL,L)=Q
  100 CONTINUE
C
C6------IF CELL-BY-CELL FLOW WILL BE SAVED AS A 3-D ARRAY,
C6------CALL UBUDSV TO SAVE THEM.
      IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IDRNCB,BUFF,NCOL,NROW,
     1                         NLAY,IOUT)
C
C7------MOVE RATES,VOLUMES & LABELS INTO ARRAYS FOR PRINTING.
  200 ROUT=RATOUT
      VBVL(3,MSUM)=ZERO
      VBVL(4,MSUM)=ROUT
      VBVL(2,MSUM)=VBVL(2,MSUM)+ROUT*DELT
      VBNM(MSUM)=TEXT
C
C8------INCREMENT BUDGET TERM COUNTER.
      MSUM=MSUM+1
C
C9------RETURN.
      RETURN
      END
```

List of Variables for Module DRN5BD

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
C	Module	Conductance of a drain.
CC	Module	Double precision equivalent of C.
CEL	Module	C times EL.
DELT	Global	Length of the current time step.
DRAI	Package	DIMENSION (NDRNVL,MXDRN), For each drain: layer, row, column, elevation, conductance, optional auxiliary parameters, and optional cell-by-cell budget flow.
EEL	Module	Double precision equivalent of EL.
EL	Module	Elevation of the drain.
HHNEW	Module	Double precision equivalent of HNEW(IC,IR,IL).
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
IBD	Module	Cell-by-cell budget flag, which is a composit of IDRNCB and ICBCFL: = -1, budget will be printed in the listing file. = 0, budget will not be saved or printed. = 1, budget will be saved by Module UBUDSV. = 2, budget will be saved by Modules UBDSV2 and UBDSVA.
IBDLBL	Module	Flag used when printing cell-by-cell budget values in the listing file so that the budget label is printed only once.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IC	Module	Index for columns.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IDRNAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in DRAI array to return budget values. ≠ 0, memory has been allocated in DRAI array to return budget values.
IDRNCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
IL	Module	Index for layers.
IOUT	Global	Unit number for writing to the listing file.
IR	Module	Index for rows.

KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
L	Module	Index for drains.
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.
MXDRN	Package	The maximum number of drains active at one time.
NCOL	Global	The number of columns in the grid.
NDRAIN	Package	Number of drains active in the current stress period.
NDRNVL	Package	The size of the first dimension of the DRAI array; that is, DRAI has dimensions of (NDRNVL,MXDRN).
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
Q	Module	Flow rate from a drain into a model cell.
QQ	Module	Double precision equivalent of Q.
RATOUT	Module	Accumulator for flow out of the model into drains.
ROUT	Module	Single precision equivalent of RATOUT.
TEXT	Module	CHARACTER*16, Label that identifies drain budget data.
TOTIM	Global	Elapsed time in the simulation.
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
ZERO	Module	The constant 0.

Budget calculations in the Evapotranspiration (EVT) Package have been changed to double precision.

Module EVT5AL

```
      SUBROUTINE EVT5AL(ISUM,LENX,LCIEVT,LCEVTR,LCEXDP,LCSURF,
     1                  NCOL,NROW,NEVTOP,IN,IOUT,IEVTCB,IFREFM)
C
C-----VERSION 0957 21FEB1996 EVT5AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR EVAPOTRANSPIRATION
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE.
      WRITE(IOUT,1)IN
    1 FORMAT(1X,/1X,'EVT5 -- EVAPOTRANSPIRATION PACKAGE, VERSION 5,',
     1     ' 9/1/93',' INPUT READ FROM UNIT',I3)
C
C2------READ ET OPTION (NEVTOP) AND UNIT OR FLAG FOR CELL-BY-CELL FLOW
C2------TERMS (IEVTCB).
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(2I10)') NEVTOP,IEVTCB
      ELSE
         READ(IN,*) NEVTOP,IEVTCB
      END IF
C
C3------CHECK TO SEE THAT ET OPTION IS LEGAL.
      IF(NEVTOP.GE.1.AND.NEVTOP.LE.2)GO TO 200
C
C3A-----IF ILLEGAL PRINT A MESSAGE & ABORT SIMULATION.
      WRITE(IOUT,8)
    8 FORMAT(1X,'ILLEGAL ET OPTION CODE. SIMULATION ABORTING')
      STOP
C
C4------IF THE OPTION IS LEGAL THEN PRINT THE OPTION CODE.
  200 IF(NEVTOP.EQ.1) WRITE(IOUT,201)
  201 FORMAT(1X,'OPTION 1 -- EVAPOTRANSPIRATION FROM TOP LAYER')
      IF(NEVTOP.EQ.2) WRITE(IOUT,202)
  202 FORMAT(1X,'OPTION 2 -- EVAPOTRANSPIRATION FROM ONE SPECIFIED',
     1      ' NODE IN EACH VERTICAL COLUMN')
      IRK=ISUM
C
C5------IF CELL-BY-CELL FLOWS ARE TO BE SAVED, THEN PRINT UNIT NUMBER.
      IF(IEVTCB.GT.0) WRITE(IOUT,203) IEVTCB
  203 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE SAVED ON UNIT',I3)
C
C6------ALLOCATE SPACE FOR THE ARRAYS EVTR, EXDP AND SURF.
      LCEVTR=ISUM
      ISUM=ISUM+NCOL*NROW
      LCEXDP=ISUM
      ISUM=ISUM+NCOL*NROW
      LCSURF=ISUM
      ISUM=ISUM+NCOL*NROW
C
C7------IF OPTION 2 THEN ALLOCATE SPACE FOR THE INDICATOR ARRAY(IEVT)
      LCIEVT=ISUM
      IF(NEVTOP.NE.2)GO TO 300
      ISUM=ISUM+NCOL*NROW
C
C8------CALCULATE & PRINT AMOUNT OF SPACE USED BY ET PACKAGE.
  300 IRK=ISUM-IRK
      WRITE(IOUT,4)IRK
    4 FORMAT(1X,I10,' ELEMENTS OF X ARRAY ARE USED BY EVT')
      ISUM1=ISUM-1
      WRITE(IOUT,5)ISUM1,LENX
    5 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX)WRITE(IOUT,6)
    6 FORMAT(1X,'   ***X ARRAY MUST BE MADE LARGER***')
C
C9------RETURN.
      RETURN
      END
```

Module EVT5RP

```
      SUBROUTINE EVT5RP(NEVTOP,IEVT,EVTR,EXDP,SURF,DELR,DELC,
     1                  NCOL,NROW,IN,IOUT,IFREFM)
C
C-----VERSION 1001 21FEB1996 EVT5RP
C     ******************************************************************
C     READ EVAPOTRANSPIRATION DATA
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*24 ANAME(4)
      DIMENSION IEVT(NCOL,NROW),EVTR(NCOL,NROW),EXDP(NCOL,NROW),
     1          SURF(NCOL,NROW),DELR(NCOL),DELC(NROW)
C
      DATA ANAME(1) /'          ET LAYER INDEX'/
      DATA ANAME(2) /'            ET SURFACE'/
      DATA ANAME(3) /' EVAPOTRANSPIRATION RATE'/
      DATA ANAME(4) /'        EXTINCTION DEPTH'/
C     ------------------------------------------------------------------
C
C1------READ FLAGS SHOWING WHETHER DATA IS TO BE REUSED.
      IF(NEVTOP.EQ.2) THEN
         IF(IFREFM.EQ.0) THEN
            READ(IN,'(4I10)') INSURF,INEVTR,INEXDP,INIEVT
         ELSE
            READ(IN,*) INSURF,INEVTR,INEXDP,INIEVT
         END IF
      ELSE
         IF(IFREFM.EQ.0) THEN
            READ(IN,'(3I10)') INSURF,INEVTR,INEXDP
         ELSE
            READ(IN,*) INSURF,INEVTR,INEXDP
         END IF
      END IF
C
C2------TEST INSURF TO SEE WHERE SURFACE ELEVATION COMES FROM.
      IF(INSURF.GE.0)GO TO 32
C
C2A-----IF INSURF<0 THEN REUSE SURFACE ARRAY FROM LAST STRESS PERIOD
      WRITE(IOUT,3)
    3 FORMAT(1X,/1X,'REUSING SURF FROM LAST STRESS PERIOD')
      GO TO 35
C
C3------IF INSURF=>0 THEN CALL MODULE U2DREL TO READ SURFACE.
   32 CALL U2DREL(SURF,ANAME(2),NROW,NCOL,0,IN,IOUT)
C
C4------TEST INEVTR TO SEE WHERE MAX ET RATE COMES FROM.
   35 IF(INEVTR.GE.0)GO TO 37
C
C4A-----IF INEVTR<0 THEN REUSE MAX ET RATE.
      WRITE(IOUT,4)
    4 FORMAT(1X,/1X,'REUSING EVTR FROM LAST STRESS PERIOD')
      GO TO 45
C
C5------IF INEVTR=>0 CALL MODULE U2DREL TO READ MAX ET RATE.
   37 CALL U2DREL(EVTR,ANAME(3),NROW,NCOL,0,IN,IOUT)
C
C6------MULTIPLY MAX ET RATE BY CELL AREA TO GET VOLUMETRIC RATE
      DO 40 IR=1,NROW
      DO 40 IC=1,NCOL
      EVTR(IC,IR)=EVTR(IC,IR)*DELR(IC)*DELC(IR)
   40 CONTINUE
C
C7------TEST INEXDP TO SEE WHERE EXTINCTION DEPTH COMES FROM
   45 IF(INEXDP.GE.0)GO TO 47
C
C7A-----IF INEXDP<0 REUSE EXTINCTION DEPTH FROM LAST STRESS PERIOD
      WRITE(IOUT,5)
    5 FORMAT(1X,/1X,'REUSING EXDP FROM LAST STRESS PERIOD')
      GO TO 48
C
C8------IF INEXDP=>0 CALL MODULE U2DREL TO READ EXTINCTION DEPTH
   47 CALL U2DREL(EXDP,ANAME(4),NROW,NCOL,0,IN,IOUT)
C
C9------IF OPTION(NEVTOP) IS 2 THEN WE NEED AN INDICATOR ARRAY.
  48  IF(NEVTOP.NE.2)GO TO 50
C
```

```
C10------IF INIEVT<0 THEN REUSE LAYER INDICATOR ARRAY.
      IF(INIEVT.GE.0)GO TO 49
      WRITE(IOUT,2)
    2 FORMAT(1X,/1X,'REUSING IEVT FROM LAST STRESS PERIOD')
      GO TO 50
C
C11------IF INIEVT=>0 THEN CALL MODULE U2DINT TO READ INDICATOR ARRAY.
   49 CALL U2DINT(IEVT,ANAME(1),NROW,NCOL,0,IN,IOUT)
C
C12-----RETURN
   50 RETURN
      END
```

Module EVT5FM

```
      SUBROUTINE EVT5FM(NEVTOP,IEVT,EVTR,EXDP,SURF,RHS,HCOF,
     1                  IBOUND,HNEW,NCOL,NROW,NLAY)
C
C-----VERSION 1616 16JULY1992 EVT5FM
C     ******************************************************************
C        ADD EVAPOTRANSPIRATION TO RHS AND HCOF
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW,HH,SS,XX,DD
      DIMENSION IEVT(NCOL,NROW),EVTR(NCOL,NROW),EXDP(NCOL,NROW),
     1          SURF(NCOL,NROW),RHS(NCOL,NROW,NLAY),
     2          HCOF(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),
     3          HNEW(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------PROCESS EACH HORIZONTAL CELL LOCATION
      DO 10 IR=1,NROW
      DO 10 IC=1,NCOL
C
C2------SET THE LAYER INDEX EQUAL TO 1      .
      IL=1
C
C3------IF OPTION 2 IS SPECIFIED THEN GET LAYER INDEX FROM IEVT ARRAY
      IF(NEVTOP.EQ.2)IL=IEVT(IC,IR)
C
C4------IF THE CELL IS EXTERNAL IGNORE IT.
      IF(IBOUND(IC,IR,IL).LE.0)GO TO 10
      C=EVTR(IC,IR)
      S=SURF(IC,IR)
      SS=S
      HH=HNEW(IC,IR,IL)
C
C5------IF AQUIFER HEAD IS GREATER THAN OR EQUAL TO SURF, ET IS CONSTANT
      IF(HH.LT.SS) GO TO 5
C
C5A-----SUBTRACT -EVTR FROM RHS
      RHS(IC,IR,IL)=RHS(IC,IR,IL) + C
      GO TO 10
C
C6------IF DEPTH TO WATER>=EXTINCTION DEPTH THEN ET IS 0
    5 DD=SS-HH
      X=EXDP(IC,IR)
      XX=X
      IF(DD.GE.XX)GO TO 10
C
C7------LINEAR RANGE. ADD ET TERMS TO BOTH RHS AND HCOF.
      RHS(IC,IR,IL)=RHS(IC,IR,IL)+C-C*S/X
      HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-C/X
   10 CONTINUE
C
C8------RETURN
      RETURN
      END
```

Module EVT5BD

Narrative for Module EVT5BD

EVT5BD calculates flow rates and volumes of water removed from the ground-water flow system by evapotranspiration. EVT5BD performs its functions as follows:

1. Set total flow rate out of the model to 0.

2. Clear the 3-D buffer that is used to store flow rates for each cell in the grid. Even if these values are not written to a file, they are returned by EVT5BD for possible use by other modules. Flag IBD is set to indicate how cell-by-cell flows will be written. If IEVTCB is less than or equal to 0, no flows are written (IBD=0). If IEVTCB is greater than 0, then IBD is set equal to ICBCFL. The value of ICBCFL, which is specified in the Output Control Option, determines how flows are written as follows:

    0 -- flows are not written
    1 -- flows are written as a 3-D array to unit IEVTCB.
    2 -- flows are written to unit IEVTCB as a 1-layer array.

3. Loop through all rows and columns and accumulate evapotranspiration.

4. Set the layer index to 1, which is the correct value when the evapotranspiration (NEVTOP) is 1.

5. If the evapotranspiration option is 2, set the layer index to the value of the layer indicator array, IEVT.

6. If cell is no flow or constant head, skip this cell. Otherwise, get parameters needed to calculate the evapotranspiration rate.

7. If aquifer head is less than the evapotranspiration surface, skip to step 8. If aquifer head is greater than or equal to the evapotranspiration surface, then set the evapotranspiration rate to the maximum rate and go to step 10.

8. Calculate the distance of the water level below the evapotranspiration surface. If this distance is greater than the extinction depth, ET is 0; skip this cell.

9. ET is in the linear range; calculate the rate using the linear formula.

10. Add evapotranspiration to RATOUT.

11. Add evapotranspiration to the 3-D cell-by-cell budget array, BUFF.

12. Call the appropriate utility module to write cell-by-cell flows; UBUDSV for a 3-D array and UBDSV3 for 1-layer array.

13. Move total flow rates into the global array of budget terms for the model budget.

14. Add the total flow volume for the time step to the global array of budget terms for the model budget.

15. Define the name of the recharge budget term.

16. Increment the budget term counter.

17. RETURN.

EVT5BD

```
      SUBROUTINE EVT5BD(NEVTOP,IEVT,EVTR,EXDP,SURF,IBOUND,HNEW,
     1           NCOL,NROW,NLAY,DELT,VBVL,VBNM,MSUM,KSTP,KPER,
     2           IEVTCB,ICBCFL,BUFF,IOUT,PERTIM,TOTIM)
C-----VERSION 0829 18DEC1992 EVT5BD
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR EVAPOTRANSPIRATION
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM),TEXT
      DOUBLE PRECISION HNEW,RATOUT,QQ,HH,SS,DD,XX,HHCOF,RRHS
      DIMENSION IEVT(NCOL,NROW),EVTR(NCOL,NROW),EXDP(NCOL,NROW),
     1          SURF(NCOL,NROW),IBOUND(NCOL,NROW,NLAY),
     2          VBVL(4,MSUM),HNEW(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
C
      DATA TEXT /'              ET'/
C     ------------------------------------------------------------------
C
C1------CLEAR THE RATE ACCUMULATOR.
      ZERO=0.
      RATOUT=ZERO
C
C2------SET CELL-BY-CELL BUDGET SAVE FLAG (IBD) AND CLEAR THE BUFFER.
      IBD=0
      IF(IEVTCB.GT.0) IBD=ICBCFL
      DO 2 IL=1,NLAY
      DO 2 IR=1,NROW
      DO 2 IC=1,NCOL
      BUFF(IC,IR,IL)=ZERO
    2 CONTINUE
C
C3------PROCESS EACH HORIZONTAL CELL LOCATION.
      DO 10 IR=1,NROW
      DO 10 IC=1,NCOL
C
C4------SET THE LAYER INDEX EQUAL TO 1.
      IL=1
C
C5------IF OPTION 2 IS SPECIFIED THEN GET LAYER INDEX FROM IEVT ARRAY.
      IF(NEVTOP.EQ.2)IL=IEVT(IC,IR)
C
C6------IF CELL IS EXTERNAL THEN IGNORE IT.
      IF(IBOUND(IC,IR,IL).LE.0)GO TO 10
      C=EVTR(IC,IR)
      S=SURF(IC,IR)
      SS=S
      HH=HNEW(IC,IR,IL)
C
C7------IF AQUIFER HEAD => SURF,SET Q=MAX ET RATE.
      IF(HH.LT.SS) GO TO 7
      QQ=-C
      GO TO 9
C
C8------IF DEPTH=>EXTINCTION DEPTH, ET IS 0.
    7 X=EXDP(IC,IR)
      XX=X
      DD=SS-HH
      IF(DD.GE.XX)GO TO 10
C
C9------LINEAR RANGE.  Q=-EVTR*(HNEW-(SURF-EXDP))/EXDP, WHICH IS
C9------FORMULATED AS Q= -HNEW*EVTR/EXDP + (EVTR*SURF/EXDP -EVTR).
      HHCOF=-C/X
      RRHS=(C*S/X)-C
      QQ=HH*HHCOF+RRHS
C
C10-----ACCUMULATE TOTAL FLOW RATE.
    9 Q=QQ
      RATOUT=RATOUT-QQ
C
C11-----ADD Q TO BUFFER.
      BUFF(IC,IR,IL)=Q
   10 CONTINUE
C
C12-----IF CELL-BY-CELL FLOW TO BE SAVED, CALL APPROPRIATE UTILITY
C12-----MODULE SAVE THEM.
      IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IEVTCB,BUFF,NCOL,NROW,
```

```
     1                            NLAY,IOUT)
      IF(IBD.EQ.2) CALL UBDSV3(KSTP,KPER,TEXT,IEVTCB,BUFF,IEVT,NEVTOP,
     1                   NCOL,NROW,NLAY,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C13-----MOVE TOTAL ET RATE INTO VBVL FOR PRINTING BY BAS1OT.
      ROUT=RATOUT
      VBVL(3,MSUM)=ZERO
      VBVL(4,MSUM)=ROUT
C
C14-----ADD ET(ET_RATE TIMES STEP LENGTH) TO VBVL.
      VBVL(2,MSUM)=VBVL(2,MSUM)+ROUT*DELT
C
C15-----MOVE BUDGET TERM LABELS TO VBNM FOR PRINT BY MODULE BAS1OT.
      VBNM(MSUM)=TEXT
C
C16-----INCREMENT BUDGET TERM COUNTER.
      MSUM=MSUM+1
C
C17-----RETURN.
      RETURN
      END
```

List of Variables for Module EVT5BD

Variable	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
C	Module	Maximum evapotranspiration rate at a cell.
DD	Module	SS-HH.
DELT	Global	Length of the current time step.
EVTR	Package	DIMENSION (NCOL,NROW), Maximum evapotranspiration rate.
EXDP	Package	DIMENSION (NCOL,NROW), Extinction depth.
HH	Module	Double precision equivalent of HNEW(IC,IR,IL).
HHCOF	Module	-C/X.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
IBD	Module	Cell-by-cell budget flag for this package: = 0, budget will not be saved or printed. = 1, budget will be saved by Module UBUDSV. = 2, budget will be saved by Modules UBDSV2 and UBDSVA.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IC	Module	Index for columns.
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IEVT	Package	DIMENSION (NCOL,NROW), Layer number to which evapotrnspiration will be applied if the evapotranspiration option (NEVTOP) is 2.
IEVTCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. < 0 or = 0, cell-by-cell budget flow will not be saved or written to the listing file.
IL	Module	Index for layers.
IOUT	Global	Unit number for writing to the listing file.
IR	Module	Index for rows.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.
NCOL	Global	The number of columns in the grid.
NEVTOP	Package	Evapotranspiration option: = 1, evapotranspiration is to the top layer. = 2, layer number for evapotranspiration for each horizontal cell location is specified in IRCH.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.

PERTIM	Global	Elapsed time during the current stress period.
Q	Module	Rate of flow from evapotranspiration into a model cell.
QQ	Module	Double precision equivalent of Q
RATOUT	Module	Accumulator for flow from evapotranspiration out of the model.
ROUT	Module	Single precision equivalent of RATOUT.
RRHS	Module	(C*S/X)-C
S	Module	Elevation of the evapotranspiration surface for a cell.
SS	Module	Double precision equivalent of S.
SURF	Package	DIMENSION (NCOL,NROW), Elevation of the evapotranspiration surface.
TEXT	Module	CHARACTER*16, Label that identifies the evapotranspiration budget data.
TOTIM	Global	Elapsed time in the simulation.
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
X	Module	Extinction depth for a cell.
XX	Module	Double precision equivalent of X.
ZERO	Module	The constant 0.

General-Head Boundary Package

Budget calculations in the General-Head Boundary (GHB) Package have been changed to double precision. In addition, the ability to read extra data parameters for each boundary has been added.

Module GHB5AL

Narrative for Module GHB5AL

Module GHB5AL allocates space in the X array for boundary data. GHB5AL performs its functions as follows:

1. Write a message identifying the package. Also, initialize the number of boundaries to 0.

2. Read the first record from the GHB file into a buffer so it can be parsed by URWORD. Decode the maximum number of boundaries (MXBND) in any stress period and the GHB cell-by-cell budget flag (IGHBCB) using URWORD or READ depending on the free format flag (IFREFM).

3. Check for alphabetic options. "AUXILIARY" or "AUX" indicates an extra data parameter is to be read. "CBCALLOCATE" or "CBC" indicates that memory will be allocated to store the flow rate into the model from each boundary. Keep track of the total number of data values for each boundary in NGHBVL, which is needed in order to calculate the required space in the X array.

4. Allocate space in the X array for the BNDS array. Set LCBNDS equal to ISUM, which is the lowest unused location in X. Add the size of BNDS to ISUM.

5. Print the number of elements in the X array used by the GHB Package and the total space used in the X array.

6. RETURN.

```
      SUBROUTINE GHB5AL(ISUM,LENX,LCBNDS,NBOUND,MXBND,IN,IOUT,IGHBCB,
     1        NGHBVL,IGHBAL,IFREFM)
C
C-----VERSION 0943 21FEB1996 GHB5AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR HEAD-DEPENDENT BOUNDARIES
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      COMMON /GHBCOM/GHBAUX(5)
      CHARACTER*16 GHBAUX
      CHARACTER*80 LINE
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE AND INITIALIZE # OF GENERAL HEAD BOUNDS.
      WRITE(IOUT,1)IN
1     FORMAT(1X,/1X,'GHB5 -- GHB PACKAGE, VERSION 5, 9/1/93',
     1' INPUT READ FROM UNIT',I3)
      NBOUND=0
C
C2------READ MAXIMUM NUMBER OF BOUNDS AND UNIT OR FLAG FOR
C2------CELL-BY-CELL FLOW TERMS.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(2I10)') MXBND,IGHBCB
         LLOC=21
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,MXBND,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,IGHBCB,R,IOUT,IN)
      END IF
      WRITE(IOUT,3) MXBND
    3 FORMAT(1X,'MAXIMUM OF',I5,' HEAD-DEPENDENT BOUNDARY NODES')
      IF(IGHBCB.LT.0) WRITE(IOUT,7)
    7 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE PRINTED WHEN ICBCFL NOT 0')
      IF(IGHBCB.GT.0) WRITE(IOUT,8) IGHBCB
    8 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE SAVED ON UNIT',I3)
C
C3------READ AUXILIARY PARAMETERS AND CBC ALLOCATION OPTION.
      IGHBAL=0
      NAUX=0
   10 CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
      IF(LINE(ISTART:ISTOP).EQ.'CBCALLOCATE' .OR.
     1   LINE(ISTART:ISTOP).EQ.'CBC') THEN
         IGHBAL=1
         WRITE(IOUT,11)
   11    FORMAT(1X,'MEMORY IS ALLOCATED FOR CELL-BY-CELL BUDGET TERMS')
         GO TO 10
      ELSE IF(LINE(ISTART:ISTOP).EQ.'AUXILIARY' .OR.
     1        LINE(ISTART:ISTOP).EQ.'AUX') THEN
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,1,N,R,IOUT,IN)
         IF(NAUX.LT.5) THEN
            NAUX=NAUX+1
            GHBAUX(NAUX)=LINE(ISTART:ISTOP)
            WRITE(IOUT,12) GHBAUX(NAUX)
   12       FORMAT(1X,'AUXILIARY BOUNDARY PARAMETER: ',A)
         END IF
         GO TO 10
      END IF
      NGHBVL=5+NAUX+IGHBAL
C
C4------ALLOCATE SPACE IN THE X ARRAY FOR THE BNDS ARRAY.
      LCBNDS=ISUM
      ISP=NGHBVL*MXBND
      ISUM=ISUM+ISP
C
C5------PRINT AMOUNT OF SPACE USED BY THE GHB PACKAGE.
      WRITE(IOUT,14) ISP
   14 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY GHB')
      ISUM1=ISUM-1
      WRITE(IOUT,15) ISUM1,LENX
   15 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,16)
   16 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C6------RETURN.
      RETURN
      END
```

List of Variables for Module GHB5AL

Variable	Range	Definition
GHBAUX	Package	CHARACTER*16(5), names of auxiliary parameters.
IFREFM	Global	Flag indicating if data should be read using free or fixed format: = 0, fixed format ≠ 0, free format
IGHBAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in BNDS array to return budget values. ≠ 0, memory has been allocated in BNDS array to return budget values.
IGHBCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
IN	Package	Primary unit number from which input for this package is read.
IOUT	Global	Unit number for writing to the listing file.
ISP	Module	Number of elements allocated in the X array by this package.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
ISUM	Global	Index of the lowest element in the X array which has not yet been allocated.
ISUM1	Module	ISUM - 1.
LCBNDS	Package	Location in the X array of the first element of array BNDS.
LENX	Global	The number of elements in the X array. LENX is defined in a PARAMETER statement in the MAIN program.
LINE	Module	CHARACTER*80, contents of a record that has been read from the package input file. LINE is parsed by URWORD.
LLOC	Module	Index that tells URWORD where to start looking for a word within LINE.
MXBND	Package	The maximum number of boundaries active at one time.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NAUX	Module	Counter for the number of auxiliary boundary parameters.
NBOUND	Package	Number of boundaries active in the current stress period.
NGHBVL	Package	The size of the first dimension of the BNDS array; that is, BNDS has dimensions of (NGHBVL,MXBND).
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

GHB5RP

```
      SUBROUTINE GHB5RP(BNDS,NBOUND,MXBND,IN,IOUT,NGHBVL,IGHBAL,IFREFM)
C
C-----VERSION 0946 21FEB1996 GHB5RP
C     ******************************************************************
C     READ DATA FOR GHB
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION BNDS(NGHBVL,MXBND)
      COMMON /GHBCOM/GHBAUX(5)
      CHARACTER*16 GHBAUX
      CHARACTER*151 LINE
C     ------------------------------------------------------------------
C
C1------READ ITMP (# OF GENERAL HEAD BOUNDS OR FLAG TO REUSE DATA).
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(I10)') ITMP
      ELSE
         READ(IN,*) ITMP
      END IF
C
C2------TEST ITMP
      IF(ITMP.GE.0) GO TO 50
C
C2A-----IF ITMP<0 THEN REUSE DATA FROM LAST STRESS PERIOD.
      WRITE(IOUT,7)
    7 FORMAT(1X,/1X,'REUSING HEAD-DEPENDENT BOUNDS FROM LAST STRESS',
     1      ' PERIOD')
      GO TO 260
C
C3------IF ITMP=>0 THEN IT IS THE # OF GENERAL HEAD BOUNDS.
   50 NBOUND=ITMP
C
C4------IF MAX NUMBER OF BOUNDS IS EXCEEDED THEN STOP.
      IF(NBOUND.LE.MXBND) GO TO 100
      WRITE(IOUT,99) NBOUND,MXBND
   99 FORMAT(1X,/1X,'NBOUND(',I4,') IS GREATER THAN MXBND(',I4,')')
C
C4A-----ABNORMAL STOP.
      STOP
C
C5------PRINT # OF GENERAL HEAD BOUNDS THIS STRESS PERIOD.
  100 WRITE(IOUT,101) NBOUND
  101 FORMAT(1X,//1X,I5,' HEAD-DEPENDENT BOUNDARY NODES')
C
C6------IF THERE ARE NO GENERAL HEAD BOUNDS THEN RETURN.
      IF(NBOUND.EQ.0) GO TO 260
C
C7------READ & PRINT DATA FOR EACH GENERAL HEAD BOUNDARY.
      NAUX=NGHBVL-5-IGHBAL
      MAXAUX=NGHBVL-IGHBAL
      IF(NAUX.GT.0) THEN
         WRITE(IOUT,103) (GHBAUX(JJ),JJ=1,NAUX)
         WRITE(IOUT,104) ('------------------',JJ=1,NAUX)
      ELSE
         WRITE(IOUT,103)
         WRITE(IOUT,104)
      END IF
  103 FORMAT(1X,/1X,'LAYER   ROW   COL   ELEVATION   CONDUCTANCE   ',
     1           'BOUND NO.',:5(2X,A))
  104 FORMAT(1X,55('-'),5A)
      DO 250 II=1,NBOUND
C7A-----READ THE REQUIRED DATA WITH FIXED OR FREE FORMAT.
      READ(IN,'(A)') LINE
      IF(IFREFM.EQ.0) THEN
         READ(LINE,'(3I10,2F10.0)') K,I,J,(BNDS(JJ,II),JJ=4,5)
         LLOC=51
      ELSE
         LLOC=1
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,K,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,I,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,2,J,R,IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,BNDS(4,II),IOUT,IN)
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,BNDS(5,II),IOUT,IN)
      END IF
C7B-----READ ANY AUXILIARY DATA WITH FREE FORMAT, AND PRINT ALL VALUES.
```

```
      IF(NAUX.GT.0) THEN
         DO 110 JJ=1,NAUX
         CALL URWORD(LINE,LLOC,ISTART,ISTOP,3,N,BNDS(JJ+5,II),IOUT,IN)
  110    CONTINUE
         WRITE (IOUT,115) K,I,J,BNDS(4,II),BNDS(5,II),II,
     1         (BNDS(JJ,II),JJ=6,MAXAUX)
      ELSE
         WRITE (IOUT,115) K,I,J,BNDS(4,II),BNDS(5,II),II
      END IF
  115 FORMAT(1X,I4,I7,I6,G13.4,G14.4,I8,:5(2X,G16.5))
      BNDS(1,II)=K
      BNDS(2,II)=I
      BNDS(3,II)=J
  250 CONTINUE
C
C8------RETURN.
  260 RETURN
      END
```

```
      SUBROUTINE GHB5FM(NBOUND,MXBND,BNDS,HCOF,RHS,IBOUND,
     1              NCOL,NROW,NLAY,NGHBVL)
C
C-----VERSION 1352 28AUG1992 GHB5FM
C     ******************************************************************
C     ADD GHB TERMS TO RHS AND HCOF
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION BNDS(NGHBVL,MXBND),HCOF(NCOL,NROW,NLAY),
     1         RHS(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------IF NBOUND<=0 THEN THERE ARE NO GENERAL HEAD BOUNDS. RETURN.
      IF(NBOUND.LE.0) RETURN
C
C2------PROCESS EACH ENTRY IN THE GENERAL HEAD BOUND LIST (BNDS).
      DO 100 L=1,NBOUND
C
C3------GET COLUMN, ROW AND LAYER OF CELL CONTAINING BOUNDARY.
      IL=BNDS(1,L)
      IR=BNDS(2,L)
      IC=BNDS(3,L)
C
C4------IF THE CELL IS EXTERNAL THEN SKIP IT.
      IF(IBOUND(IC,IR,IL).LE.0) GO TO 100
C
C5------SINCE THE CELL IS INTERNAL GET THE BOUNDARY DATA.
      HB=BNDS(4,L)
      C=BNDS(5,L)
C
C6------ADD TERMS TO RHS AND HCOF.
      HCOF(IC,IR,IL)=HCOF(IC,IR,IL)-C
      RHS(IC,IR,IL)=RHS(IC,IR,IL)-C*HB
  100 CONTINUE
C
C7------RETURN.
      RETURN
      END
```

Module GHB5BD

Narrative for Module GHB5BD

GHB5BD calculates flow rates and volumes for water moving between the ground-water flow system and general head-dependent boundaries. GHB5BD performs its functions as follows:

1. Initialize values. Total flow rates in and out of the model are set to 0. Flag IBD is set to indicate how cell-by-cell flows will be written. If ICBCFL is 0, no flows are written (IBD=0). If ICBCFL is not 0, flows are written as follows:
   If IGHBCB is less than 0, flows are written to the listing file (IBD=-1).
   If IGHBCB is greater than 0 and ICBCFL is 1, flows are written to unit IGHBCB as a 3-D array (IBD=1).
   If IGHBCB is greater than 0 and ICBCFL is 2, flows are written to unit IGHBCB as a list (IBD=2).

2. If cell-by-cell flows are to be written as a list, call UBDSV2 to write header information.

3. Clear the 3-D buffer that is used to store flow rates for each cell in the grid. Even if these values are not written to a file, they are returned by GHB5BD for possible use by other modules.

4. If there are no boundaries, skip to step 7.

5. Loop through each boundary calculating flow.

   A. Get layer, row, and column locations for boundary, and initialize the flow rate to 0.

   B. Skip to step 5J if the cell is no-flow or constant-head, which means there is no flow from this boundary.

   C. Get boundary parameters from the BNDS array, and define variables. CC is the double precision value of the conductance of the boundary. Precision of values is carefully controlled in order that calculations are done using the same precision that the solvers use and to avoid mixed mode expressions.

   D. Calculate the head-dependent flow. Flow is C(HB-HNEW); however, this is calculated as C*HB - C*HNEW so that budget calculations will be formulated like the flow equation is formulated. This can be important for simulations that can barely be solved due to a computer's limited precision.

   E. Write the flow rate to the listing file if IBD is less than 0. Before writing the flow for the first boundary, write a heading.

   F. Add the flow rate to the 3-D buffer.

   G. In order to accumulate inflows separately from outflows, check the sign of the flow.

   H. Flow is negative, which indicates flow into the boundary. Subtract flow from total outflow.

I. Flow is positive, which indicates flow into the model node. Add flow to total inflow.

J. If IBD is 2, call UBDSVA to save flow to a file. If flow is being returned in the BNDS array, copy flow to BNDS. This is the end of the loop that is invoked for each boundary.

6. If IBD is 1, call UBUDSV to save flow as a 3-D array.

7. Move total flow rates and volumes into the global array of budget terms for the model budget. Also, define the name of the boundary budget term.

8. Increment the budget term counter.

9. RETURN.

GHB5BD

```
      SUBROUTINE GHB5BD(NBOUND,MXBND,VBNM,VBVL,MSUM,BNDS,DELT,HNEW,
     1   NCOL,NROW,NLAY,IBOUND,KSTP,KPER,IGHBCB,ICBCFL,BUFF,IOUT,
     2   PERTIM,TOTIM,NGHBVL,IGHBAL)
C-----VERSION 1410 07APRIL1993 GHB5BD
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR GHB
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 VBNM(MSUM),TEXT
      DOUBLE PRECISION HNEW,CC,CHB,RATIN,RATOUT,RRATE
      DIMENSION VBVL(4,MSUM),BNDS(NGHBVL,MXBND),HNEW(NCOL,NROW,NLAY),
     1          IBOUND(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
C
      DATA TEXT /' HEAD DEP BOUNDS'/
C     ------------------------------------------------------------------
C
C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATORS (RATIN AND RATOUT).
      ZERO=0.
      RATOUT=ZERO
      RATIN=ZERO
      IBD=0
      IF(IGHBCB.LT.0 .AND. ICBCFL.NE.0) IBD=-1
      IF(IGHBCB.GT.0) IBD=ICBCFL
      IBDLBL=0
C
C2------IF CELL-BY-CELL FLOWS WILL BE SAVED AS A LIST, WRITE HEADER.
      IF(IBD.EQ.2) CALL UBDSV2(KSTP,KPER,TEXT,IGHBCB,NCOL,NROW,NLAY,
     1          NBOUND,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C3------CLEAR THE BUFFER.
      DO 50 IL=1,NLAY
      DO 50 IR=1,NROW
      DO 50 IC=1,NCOL
      BUFF(IC,IR,IL)=ZERO
50    CONTINUE
C
C4------IF NO BOUNDARIES, SKIP FLOW CALCULATIONS.
      IF(NBOUND.EQ.0) GO TO 200
C
C5------LOOP THROUGH EACH BOUNDARY CALCULATING FLOW.
      DO 100 L=1,NBOUND
C
C5A-----GET LAYER, ROW AND COLUMN OF EACH GENERAL HEAD BOUNDARY.
      IL=BNDS(1,L)
      IR=BNDS(2,L)
      IC=BNDS(3,L)
      RATE=ZERO
C
C5B-----IF CELL IS NO-FLOW OR CONSTANT-HEAD, THEN IGNORE IT.
      IF(IBOUND(IC,IR,IL).LE.0) GO TO 99
C
C5C-----GET PARAMETERS FROM BOUNDARY LIST.
      HB=BNDS(4,L)
      C=BNDS(5,L)
      CC=C
C
C5D-----CALCULATE THE FOW RATE INTO THE CELL.
      CHB=C*HB
      RRATE=CHB - CC*HNEW(IC,IR,IL)
      RATE=RRATE
C
C5E-----PRINT THE INDIVIDUAL RATES IF REQUESTED(IGHBCB<0).
      IF(IBD.LT.0) THEN
         IF(IBDLBL.EQ.0) WRITE(IOUT,61) TEXT,KPER,KSTP
   61    FORMAT(1X,/1X,A,'   PERIOD',I3,'   STEP',I3)
         WRITE(IOUT,62) L,IL,IR,IC,RATE
   62    FORMAT(1X,'BOUNDARY',I4,'   LAYER',I3,'   ROW',I4,'   COL',I4,
     1       '   RATE',1PG15.6)
         IBDLBL=1
      END IF
C
C5F-----ADD RATE TO BUFFER.
      BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+RATE
C
```

```
C5G-----SEE IF FLOW IS INTO AQUIFER OR OUT OF AQUIFER.
      IF(RATE)94,99,96
C
C5H------FLOW IS OUT OF AQUIFER SUBTRACT RATE FROM RATOUT.
94    RATOUT=RATOUT-RRATE
      GO TO 99
C
C5I-----FLOW IS INTO AQIFER; ADD RATE TO RATIN.
96    RATIN=RATIN+RRATE
C
C5J-----IF SAVING CELL-BY-CELL FLOWS IN LIST, WRITE FLOW.  OR IF
C5J-----RETURNING THE FLOW IN THE BNDS ARRAY, COPY FLOW TO BNDS.
99    IF(IBD.EQ.2) CALL UBDSVA(IGHBCB,NCOL,NROW,IC,IR,IL,RATE,IBOUND,
     1                         NLAY)
      IF(IGHBAL.NE.0) BNDS(NGHBVL,L)=RATE
100   CONTINUE
C
C6------IF CELL-BY-CELL TERMS WILL BE SAVED AS A 3-D ARRAY, THEN CALL
C6------UTILITY MODULE UBUDSV TO SAVE THEM.
      IF(IBD.EQ.1) CALL UBUDSV(KSTP,KPER,TEXT,IGHBCB,BUFF,NCOL,NROW,
     1                           NLAY,IOUT)
C
C7------MOVE RATES, VOLUMES AND LABELS INTO ARRAYS FOR PRINTING.
  200 RIN=RATIN
      ROUT=RATOUT
      VBVL(3,MSUM)=RIN
      VBVL(1,MSUM)=VBVL(1,MSUM)+RIN*DELT
      VBVL(4,MSUM)=ROUT
      VBVL(2,MSUM)=VBVL(2,MSUM)+ROUT*DELT
      VBNM(MSUM)=TEXT
C
C8------INCREMENT THE BUDGET TERM COUNTER.
      MSUM=MSUM+1
C
C9------RETURN.
      RETURN
      END
```

List of Variables for GHB5BD

Variable	Range	Definition
BNDS	Package	DIMENSION (NGHBVL,MXBND), For each boundary: layer, row, column, conductance, optional auxiliary parameters, and optional cell-by-cell budget flow.
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
C	Module	Conductance of a boundary.
CC	Module	Double precision equivalent of C.
CHB	Module	C*HB.
DELT	Global	Length of the current time step.
HB	Module	Head for a boundary.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
IBD	Module	Cell-by-cell budget flag, which is a composit of IGHBCB and ICBCFL: = -1, budget will be printed in the listing file. = 0, budget will not be saved or printed. = 1, budget will be saved by Module UBUDSV. = 2, budget will be saved by Modules UBDSV2 and UBDSVA.
IBDLBL	Module	Flag used when printing cell-by-cell budget values in the listing file so that the budget label is printed only once.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell
IC	Module	Index for columns
ICBCFL	Global	Flag for saving or printing cell-by-cell flow terms: = 0, cell-by-cell flow terms will not be saved or printed for the current time step. ≠ 0, cell-by-cell flow terms will be saved or printed for the current time step.
IGHBAL	Package	Flag for allocation of memory for returning cell-by-cell flows: = 0, memory has not been allocated in BNDS array to return budget values. ≠ 0, memory has been allocated in BNDS array to return budget values.
IGHBCB	Package	Cell-by-cell budget flag for this package: > 0, unit number for saving cell-by-cell budget flow whenever ICBCFL is set. = 0, cell-by-cell budget flow will not be saved or written to the listing file. < 0, cell-by-cell budget flow will be written to the listing file whenever ICBCFL is set.
IL	Module	Index for layers.
IOUT	Global	Unit number for writing to the listing file.
IR	Module	Index for rows.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
L	Module	Index for boundaries.
MSUM	Global	Counter for budget terms stored in VBNM and VBVL.

MXBND	Package	The maximum number of boundaries active at one time.
NBOUND	Package	Number of boundaries active in the current stress period.
NCOL	Global	The number of columns in the grid.
NGHBVL	Package	The size of the first dimension of the BNDS array; that is, BNDS has dimensions of (NGHBVL,MXBND).
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
RATE	Module	Flow from a boundary into a cell.
RATIN	Module	Accumulator for flow from boundaries into the model.
RATOUT	Module	Accumulator for flow from the model out to boundaries.
RIN	Module	Single precision equivalent of RATIN.
ROUT	Module	Single precision equivalent of RATOUT.
RRATE	Module	Double precision equivalent of RATE.
TEXT	Module	CHARACTER*16, Label that identifies boundary budget data.
TOTIM	Global	Elapsed time in the simulation.
VBNM	Global	CHARACTER*16(MSUM), Labels for terms in the volumetric budget.
VBVL	Global	DIMENSION (4,MSUM), Flows for the volumetric budget. For budget term N, the values in VBVL are: (1,N) Volume into the flow system during the simulation. (2,N) Volume out of the flow system during the simulation. (3,N) Rate into the flow system for the current time step. (4,N) Rate out of the flow system for the current time step.
ZERO	Module	The constant 0.

In Module SIP5AP, a check has been added for division by 0. If detected, a message is printed telling the location of the cell that is causing the attempt to divide by 0. By knowing which cell is causing the problem, the user can examine the data for that cell to determine the cause. Previously the divide by 0 error would cause the program to abort with little indication of the cause.

Module SIP5AL

```
      SUBROUTINE SIP5AL(ISUM,LENX,LCEL,LCFL,LCGL,LCV,LCHDCG,LCLRCH,
     1       LCW,MXITER,NPARM,NCOL,NROW,NLAY,IN,IOUT,IFREFM)
C
C-----VERSION 0812 21FEB1996 SIP5AL
C
C     ******************************************************************
C     ALLOCATE STORAGE IN THE X ARRAY FOR SIP ARRAYS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------PRINT A MESSAGE IDENTIFYING SIP PACKAGE
      WRITE(IOUT,1)IN
    1 FORMAT(1X,
     1   /1X,'SIP5 -- STRONGLY IMPLICIT PROCEDURE SOLUTION PACKAGE',
     2   /20X,'VERSION 5, 9/1/93',' INPUT READ FROM UNIT',I3)
C
C2------READ AND PRINT MXITER AND NPARM
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(2I10)') MXITER,NPARM
      ELSE
         READ(IN,*) MXITER,NPARM
      END IF
      WRITE(IOUT,3) MXITER,NPARM
    3 FORMAT(1X,'MAXIMUM OF',I4,' ITERATIONS ALLOWED FOR CLOSURE'/
     1       1X,I2,' ITERATION PARAMETERS')
C
C3------ALLOCATE SPACE FOR THE SIP ARRAYS
      ISOLD=ISUM
      NRC=NROW*NCOL
      ISIZ=NRC*NLAY
      LCEL=ISUM
      ISUM=ISUM+ISIZ
      LCFL=ISUM
      ISUM=ISUM+ISIZ
      LCGL=ISUM
      ISUM=ISUM+ISIZ
      LCV=ISUM
      ISUM=ISUM+ISIZ
      LCHDCG=ISUM
      ISUM=ISUM+MXITER
      LCLRCH=ISUM
      ISUM=ISUM+3*MXITER
      LCW=ISUM
      ISUM=ISUM+NPARM
C
C4------CALCULATE AND PRINT THE SPACE USED IN THE X ARRAY
      ISP=ISUM-ISOLD
      WRITE(IOUT,4) ISP
    4 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY SIP')
      ISUM1=ISUM-1
      WRITE(IOUT,5) ISUM1,LENX
    5 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,6)
    6 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C5------RETURN
      RETURN
      END
```

Module SIP5RP

```
      SUBROUTINE SIP5RP(NPARM,MXITER,ACCL,HCLOSE,W,IN,IPCALC,IPRSIP,
     1                  IOUT,IFREFM)
C
C-----VERSION 0812 21FEB1996 SIP5RP
C
C     ******************************************************************
C     READ DATA FOR SIP
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION W(NPARM)
C     ------------------------------------------------------------------
C
C1------READ ACCL,HCLOSE,WSEED,IPCALC,IPRSIP
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(2F10.0,I10,F10.0,I10)')
     1             ACCL,HCLOSE,IPCALC,WSEED,IPRSIP
      ELSE
         READ(IN,*) ACCL,HCLOSE,IPCALC,WSEED,IPRSIP
      END IF
      ZERO=0.
      IF(ACCL.EQ.ZERO) ACCL=1.
C
C2------PRINT DATA VALUES JUST READ
      WRITE(IOUT,100)
  100 FORMAT(1X,///10X,'SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE'
     1/10X,43('-'))
      WRITE(IOUT,115) MXITER
  115 FORMAT(1X,'MAXIMUM ITERATIONS ALLOWED FOR CLOSURE =',I9)
      WRITE(IOUT,120) ACCL
  120 FORMAT(1X,16X,'ACCELERATION PARAMETER =',G15.5)
      WRITE(IOUT,125) HCLOSE
  125 FORMAT(1X,5X,'HEAD CHANGE CRITERION FOR CLOSURE =',E15.5)
      IF(IPRSIP.LE.0)IPRSIP=999
      WRITE(IOUT,130) IPRSIP
  130 FORMAT(1X,5X,'SIP HEAD CHANGE PRINTOUT INTERVAL =',I9)
C
C3------CHECK IF SPECIFIED VALUE OF WSEED SHOULD BE USED OR IF
C3------SEED SHOULD BE CALCULATED
      IF(IPCALC.EQ.0) GO TO 150
C
C3A-----CALCULATE SEED & ITERATION PARAMETERS PRIOR TO 1ST ITERATION
      WRITE(IOUT,140)
  140 FORMAT(1X,/5X,'CALCULATE ITERATION PARAMETERS FROM MODEL',
     1' CALCULATED WSEED')
      GO TO 1000
C
C3B-----USE SPECIFIED VALUE OF WSEED
C3B-----CALCULATE AND PRINT ITERATION PARAMETERS
  150 ONE=1.
      P1=-ONE
      P2=NPARM-1
      DO 160 I=1,NPARM
      P1=P1+ONE
  160 W(I)=ONE-WSEED**(P1/P2)
      WRITE(IOUT,161) NPARM,WSEED,(W(J),J=1,NPARM)
  161 FORMAT(1X,/1X,I5,' ITERATION PARAMETERS CALCULATED FROM',
     1     ' SPECIFIED WSEED =',F11.8,' :'//(1X,5E13.6))
C
C4------RETURN
 1000 RETURN
      END
```

Module SIP5AP

```
      SUBROUTINE SIP5AP(HNEW,IBOUND,CR,CC,CV,HCOF,RHS,EL,FL,GL,V,
     1      W,HDCG,LRCH,NPARM,KITER,HCLOSE,ACCL,ICNVG,KSTP,KPER,
     2      IPCALC,IPRSIP,MXITER,NSTP,NCOL,NROW,NLAY,NODES,IOUT)
C-----VERSION 1402 09APRIL1993 SIP5AP
C
C     ******************************************************************
C     SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE -- 1 ITERATION
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW,DITPAR,AC,HHCOF,RRHS,XI,DZERO,DONE,RES
      DOUBLE PRECISION Z,B,D,E,F,H,S,AP,TP,CP,GP,UP,RP
      DOUBLE PRECISION ZHNEW,BHNEW,DHNEW,FHNEW,HHNEW,SHNEW
      DOUBLE PRECISION AL,BL,CL,DL,ELNCL,FLNCL,GLNCL
      DOUBLE PRECISION ELNRL,FLNRL,GLNRL,ELNLL,FLNLL,GLNLL
      DOUBLE PRECISION VNRL,VNCL,VNLL,ELXI,FLXI,GLXI,VN
C
      DIMENSION HNEW(NODES), IBOUND(NODES), CR(NODES), CC(NODES),
     1  CV(NODES), HCOF(NODES), RHS(NODES), EL(NODES), FL(NODES),
     2  GL(NODES), V(NODES), W(NPARM), HDCG(MXITER), LRCH(3,MXITER)
C     ------------------------------------------------------------------
C
C1------CALCULATE ITERATION PARAMETERS IF FLAG IS SET.  THEN
C1------CLEAR THE FLAG SO THAT CALCULATION IS DONE ONLY ONCE.
      IF(IPCALC.NE.0)
     1   CALL SSIP5I(CR,CC,CV,IBOUND,NPARM,W,NCOL,NROW,NLAY,IOUT)
      IPCALC=0
C
C2------ASSIGN VALUES TO FIELDS THAT ARE CONSTANT DURING AN ITERATION
      ZERO=0.
      DZERO=0.
      DONE=1.
      AC=ACCL
      NRC=NROW*NCOL
      NTH=MOD(KITER-1,NPARM)+1
      DITPAR=W(NTH)
C
C3------INITIALIZE VARIABLE THAT TRACKS MAXIMUM HEAD CHANGE DURING
C3------THE ITERATION
      BIGG=ZERO
C
C4------CLEAR SIP WORK ARRAYS.
      DO 100 I=1,NODES
      EL(I)=ZERO
      FL(I)=ZERO
      GL(I)=ZERO
  100 V(I)=ZERO
C
C5------SET NORMAL/REVERSE EQUATION ORDERING FLAG (1 OR -1) AND
C5------CALCULATE INDEXES DEPENDENT ON ORDERING
      IDIR=1
      IF(MOD(KITER,2).EQ.0)IDIR=-1
      IDNRC=IDIR*NRC
      IDNCOL=IDIR*NCOL
C
C6------STEP THROUGH CELLS CALCULATING INTERMEDIATE VECTOR V
C6------USING FORWARD SUBSTITUTION
      DO 150 K=1,NLAY
      DO 150 I=1,NROW
      DO 150 J=1,NCOL
C
C6A-----SET UP CURRENT CELL LOCATION INDEXES.  THESE ARE DEPENDENT
C6A-----ON THE DIRECTION OF EQUATION ORDERING.
      IF(IDIR.LE.0)GO TO 120
      II=I
      JJ=J
      KK=K
      GO TO 122
  120 II=NROW-I+1
      JJ=J
      KK=NLAY-K+1
C
C6B-----CALCULATE 1 DIMENSIONAL SUBSCRIPT OF CURRENT CELL AND
C6B-----SKIP CALCULATIONS IF CELL IS NOFLOW OR CONSTANT HEAD
  122 N=JJ+(II-1)*NCOL+(KK-1)*NRC
      IF(IBOUND(N).LE.0)GO TO 150
```

```
C
C6C-----CALCULATE 1 DIMENSIONAL SUBSCRIPTS FOR LOCATING THE 6
C6C-----SURROUNDING CELLS
      NRN=N+IDNCOL
      NRL=N-IDNCOL
      NCN=N+1
      NCL=N-1
      NLN=N+IDNRC
      NLL=N-IDNRC
C
C6D-----CALCULATE 1 DIMENSIONAL SUBSCRIPTS FOR CONDUCTANCE TO THE 6
C6D-----SURROUNDING CELLS.  THESE DEPEND ON ORDERING OF EQUATIONS.
      IF(IDIR.LE.0)GO TO 124
      NCF=N
      NCD=NCL
      NRB=NRL
      NRH=N
      NLS=N
      NLZ=NLL
      GO TO 126
  124 NCF=N
      NCD=NCL
      NRB=N
      NRH=NRN
      NLS=NLN
      NLZ=N
C
C6E-----ASSIGN VARIABLES IN MATRICES A & U INVOLVING ADJACENT CELLS
C6E1----NEIGHBOR IS 1 ROW BACK
  126 B=DZERO
      ELNRL=DZERO
      FLNRL=DZERO
      GLNRL=DZERO
      BHNEW=DZERO
      VNRL=DZERO
      IF(I.EQ.1) GO TO 128
      B=CC(NRB)
      ELNRL=EL(NRL)
      FLNRL=FL(NRL)
      GLNRL=GL(NRL)
      BHNEW=B*HNEW(NRL)
      VNRL=V(NRL)
C
C6E2----NEIGHBOR IS 1 ROW AHEAD
  128 H=DZERO
      HHNEW=DZERO
      IF(I.EQ.NROW) GO TO 130
      H=CC(NRH)
      HHNEW=H*HNEW(NRN)
C
C6E3----NEIGHBOR IS 1 COLUMN BACK
  130 D=DZERO
      ELNCL=DZERO
      FLNCL=DZERO
      GLNCL=DZERO
      DHNEW=DZERO
      VNCL=DZERO
      IF(J.EQ.1) GO TO 132
      D=CR(NCD)
      ELNCL=EL(NCL)
      FLNCL=FL(NCL)
      GLNCL=GL(NCL)
      DHNEW=D*HNEW(NCL)
      VNCL=V(NCL)
C
C6E4----NEIGHBOR IS 1 COLUMN AHEAD
  132 F=DZERO
      FHNEW=DZERO
      IF(J.EQ.NCOL) GO TO 134
      F=CR(NCF)
      FHNEW=F*HNEW(NCN)
C
C6E5----NEIGHBOR IS 1 LAYER BEHIND
  134 Z=DZERO
      ELNLL=DZERO
      FLNLL=DZERO
      GLNLL=DZERO
      ZHNEW=DZERO
      VNLL=DZERO
      IF(K.EQ.1) GO TO 136
```

```
      Z=CV(NLZ)
      ELNLL=EL(NLL)
      FLNLL=FL(NLL)
      GLNLL=GL(NLL)
      ZHNEW=Z*HNEW(NLL)
      VNLL=V(NLL)
C
C6E6----NEIGHBOR IS 1 LAYER AHEAD
  136 S=DZERO
      SHNEW=DZERO
      IF(K.EQ.NLAY) GO TO 138
      S=CV(NLS)
      SHNEW=S*HNEW(NLN)
C
C6E7----CALCULATE THE NEGATIVE SUM OF ALL CONDUCTANCES TO NEIGHBORING
C6E7----CELLS
  138 E=-Z-B-D-F-H-S
C
C6F-----CALCULATE COMPONENTS OF THE UPPER AND LOWER MATRICES, WHICH
C6F-----ARE THE FACTORS OF MATRIX (A+B)
      AL=Z/(DONE+DITPAR*(ELNLL+FLNLL))
      BL=B/(DONE+DITPAR*(ELNRL+GLNRL))
      CL=D/(DONE+DITPAR*(FLNCL+GLNCL))
      AP=AL*ELNLL
      CP=BL*ELNRL
      GP=CL*FLNCL
      RP=CL*GLNCL
      TP=AL*FLNLL
      UP=BL*GLNRL
      HHCOF=HCOF(N)
      DL=E+HHCOF+DITPAR*(AP+TP+CP+GP+UP+RP)-AL*GLNLL-BL*FLNRL-CL*ELNCL
      IF(DL.EQ.DZERO) THEN
         WRITE(IOUT,139) KK,II,JJ
139      FORMAT(1X,/1X,'DIVIDE BY 0 IN SIP AT LAYER',I3,',  ROW',I4,
     1   ',  COLUMN',I4,/
     2   1X,'THIS CAN OCCUR WHEN A CELL IS CONNECTED TO THE REST OF',/
     3   1X,'THE MODEL THROUGH A SINGLE CONDUCTANCE BRANCH.  CHECK',/
     4   1X,'FOR THIS SITUATION AT THE INDICATED CELL.')
         STOP
      END IF
      EL(N)=(F-DITPAR*(AP+CP))/DL
      FL(N)=(H-DITPAR*(TP+GP))/DL
      GL(N)=(S-DITPAR*(RP+UP))/DL
C
C6G-----CALCULATE THE RESIDUAL
      RRHS=RHS(N)
      RES=RRHS-ZHNEW-BHNEW-DHNEW-E*HNEW(N)-HHCOF*HNEW(N)-FHNEW-HHNEW
     1      -SHNEW
C
C6H-----CALCULATE THE INTERMEDIATE VECTOR V
      V(N)=(AC*RES-AL*VNLL-BL*VNRL-CL*VNCL)/DL
C
  150 CONTINUE
C
C7------STEP THROUGH EACH CELL AND SOLVE FOR HEAD CHANGE BY BACK
C7------SUBSTITUTION
      DO 160 K=1,NLAY
      DO 160 I=1,NROW
      DO 160 J=1,NCOL
C
C7A-----SET UP CURRENT CELL LOCATION INDEXES.  THESE ARE DEPENDENT
C7A-----ON THE DIRECTION OF EQUATION ORDERING.
      IF(IDIR.LT.0) GO TO 152
      KK=NLAY-K+1
      II=NROW-I+1
      JJ=NCOL-J+1
      GO TO 154
  152 KK=K
      II=I
      JJ=NCOL-J+1
C
C7B-----CALCULATE 1 DIMENSIONAL SUBSCRIPT OF CURRENT CELL AND
C7B-----SKIP CALCULATIONS IF CELL IS NOFLOW OR CONSTANT HEAD
  154 N=JJ+(II-1)*NCOL+(KK-1)*NRC
      IF(IBOUND(N).LE.0)GO TO 160
C
C7C-----CALCULATE 1 DIMENSIONAL SUBSCRIPTS FOR THE 3 NEIGHBORING CELLS
C7C-----BEHIND (RELATIVE TO THE DIRECTION OF THE BACK SUBSTITUTION
C7C-----ORDERING) THE CURRRENT CELL.
      NC=N+1
```

```
      NR=N+IDNCOL
      NL=N+IDNRC
C
C7D-----BACK SUBSTITUTE, STORING HEAD CHANGE IN ARRAY V IN PLACE OF
C7D-----INTERMEDIATE FORWARD SUBSTITUTION VALUES.
      ELXI=DZERO
      FLXI=DZERO
      GLXI=DZERO
      IF(JJ.NE.NCOL) ELXI=EL(N)*V(NC)
      IF(I.NE.1) FLXI=FL(N)*V(NR)
      IF(K.NE.1) GLXI=GL(N)*V(NL)
      VN=V(N)
      V(N)=VN-ELXI-FLXI-GLXI
C
C7E-----GET THE ABSOLUTE HEAD CHANGE. IF IT IS MAX OVER GRID SO FAR.
C7E-----THEN SAVE IT ALONG WITH CELL INDICES AND HEAD CHANGE.
      TCHK=ABS(V(N))
      IF (TCHK.LE.BIGG) GO TO 155
      BIGG=TCHK
      BIG=V(N)
      IB=II
      JB=JJ
      KB=KK
C
C7F-----ADD HEAD CHANGE THIS ITERATION TO HEAD FROM THE PREVIOUS
C7F-----ITERATION TO GET A NEW ESTIMATE OF HEAD.
  155 XI=V(N)
      HNEW(N)=HNEW(N)+XI
C
  160 CONTINUE
C
C8------STORE THE LARGEST ABSOLUTE HEAD CHANGE (THIS ITERATION) AND
C8------AND ITS LOCATION.
      HDCG(KITER)=BIG
      LRCH(1,KITER)=KB
      LRCH(2,KITER)=IB
      LRCH(3,KITER)=JB
      ICNVG=0
      IF(BIGG.LE.HCLOSE) ICNVG=1
C
C9------IF END OF TIME STEP, PRINT # OF ITERATIONS THIS STEP
      IF(ICNVG.EQ.0 .AND. KITER.NE.MXITER) GO TO 600
      IF(KSTP.EQ.1) WRITE(IOUT,500)
  500 FORMAT(1X)
      WRITE(IOUT,501) KITER,KSTP,KPER
  501 FORMAT(1X,I5,' ITERATIONS FOR TIME STEP',I4,' IN STRESS PERIOD',
     1        I3)
C
C10-----PRINT HEAD CHANGE EACH ITERATION IF PRINTOUT INTERVAL IS REACHED
      IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP .OR. MOD(KSTP,IPRSIP).EQ.0)
     1      CALL SSIP5P(HDCG,LRCH,KITER,MXITER,IOUT)
C
C11-----RETURN
600   RETURN
C
      END
```

Module SSIP5P

```
      SUBROUTINE SSIP5P(HDCG,LRCH,KITER,MXITER,IOUT)
C
C-----VERSION 1534 31OCT1995 SSIP5P
C     ******************************************************************
C     PRINT MAXIMUM HEAD CHANGE FOR EACH ITERATION DURING A TIME STEP
C     ******************************************************************
C
C       SPECIFICATIONS:
C     ------------------------------------------------------------------
C
      DIMENSION HDCG(MXITER), LRCH(3,MXITER)
C     ------------------------------------------------------------------
C
      WRITE(IOUT,5)
5     FORMAT(1X,/1X,'MAXIMUM HEAD CHANGE FOR EACH ITERATION:',/
     1       1X,/1X,5('   HEAD CHANGE'),/
     2          1X,5(' LAYER,ROW,COL')/1X,70('-'))
      NGRP=(KITER-1)/5 +1
      DO 20 K=1,NGRP
         L1=(K-1)*5 +1
         L2=L1+4
         IF(K.EQ.NGRP) L2=KITER
         WRITE(IOUT,10) (HDCG(J),J=L1,L2)
         WRITE(IOUT,11) ((LRCH(I,J),I=1,3),J=L1,L2)
10       FORMAT(1X,5G14.4)
11       FORMAT(1X,5(:' (',I3,',',I3,',',I3,')'))
20    CONTINUE
      WRITE(IOUT,12)
12    FORMAT(1X)
C
      RETURN
C
      END
```

Module SSIP5I

```
      SUBROUTINE SSIP5I(CR,CC,CV,IBOUND,NPARM,W,NCOL,NROW,NLAY,
     1          IOUT)
C
C-----VERSION 1033 22JUNE1992 SSIP5I
C     ******************************************************************
C     CALCULATE AN ITERATION PARAMETER SEED AND USE IT TO CALCULATE SIP
C     ITERATION PARAMETERS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION CR(NCOL,NROW,NLAY),CC(NCOL,NROW,NLAY)
     1       ,CV(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),W(NPARM)
C
      DOUBLE PRECISION DWMIN,AVGSUM
C     ------------------------------------------------------------------
C
C1------CALCULATE CONSTANTS AND INITIALIZE VARIABLES
      ZERO=0.
      ONE=1.
      TWO=2.
      PIEPIE=9.869604
      R=NROW
      C=NCOL
      ZL=NLAY
      CCOL=PIEPIE/(TWO*C*C)
      CROW=PIEPIE/(TWO*R*R)
      CLAY=PIEPIE/(TWO*ZL*ZL)
      WMINMN=ONE
      AVGSUM=ZERO
      NODES=0
C
C2------LOOP THROUGH ALL CELLS, CALCULATING A SEED FOR EACH CELL
C2------THAT IS ACTIVE
      DO 100 K=1,NLAY
      DO 100 I=1,NROW
      DO 100 J=1,NCOL
      IF(IBOUND(J,I,K).LE.0) GO TO 100
C
C2A-----CONDUCTANCE FROM THIS CELL
C2A-----TO EACH OF THE 6 ADJACENT CELLS
      D=ZERO
      IF(J.NE.1) D=CR(J-1,I,K)
      F=ZERO
      IF(J.NE.NCOL) F=CR(J,I,K)
      B=ZERO
      IF(I.NE.1) B=CC(J,I-1,K)
      H=ZERO
      IF(I.NE.NROW) H=CC(J,I,K)
      Z=ZERO
      IF(K.NE.1) Z=CV(J,I,K-1)
      S=ZERO
      IF(K.NE.NLAY) S=CV(J,I,K)
C
C2B-----FIND THE MAXIMUM AND MINIMUM OF THE 2 CONDUCTANCE COEFFICIENTS
C2B-----IN EACH PRINCIPAL COORDINATE DIRECTION
      DFMX=MAX(D,F)
      BHMX=MAX(B,H)
      ZSMX=MAX(Z,S)
      DFMN=MIN(D,F)
      BHMN=MIN(B,H)
      ZSMN=MIN(Z,S)
      IF(DFMN.EQ.ZERO) DFMN=DFMX
      IF(BHMN.EQ.ZERO) BHMN=BHMX
      IF(ZSMN.EQ.ZERO) ZSMN=ZSMX
C
C2C-----CALCULATE A SEED IN EACH PRINCIPAL COORDINATE DIRECTION
      WCOL=ONE
      IF(DFMN.NE.ZERO) WCOL=CCOL/(ONE+(BHMX+ZSMX)/DFMN)
      WROW=ONE
      IF(BHMN.NE.ZERO) WROW=CROW/(ONE+(DFMX+ZSMX)/BHMN)
      WLAY=ONE
      IF(ZSMN.NE.ZERO) WLAY=CLAY/(ONE+(DFMX+BHMX)/ZSMN)
C
C2D-----SELECT THE CELL SEED, WHICH IS THE MINIMUM SEED OF THE 3.
C2D-----SELECT THE MINIMUM SEED OVER THE WHOLE GRID.
      WMIN=MIN(WCOL,WROW,WLAY)
```

```
      WMINMN=MIN(WMINMN,WMIN)
C
C2E-----ADD THE CELL SEED TO THE ACCUMULATOR AVGSUM FOR USE
C2E-----IN GETTING THE AVERAGE SEED.
      DWMIN=WMIN
      AVGSUM=AVGSUM+DWMIN
      NODES=NODES+1
C
  100 CONTINUE
C
C3------CALCULATE THE AVERAGE SEED OF THE CELL SEEDS, AND PRINT
C3------THE AVERAGE AND MINIMUM SEEDS.
      TMP=NODES
      AVGMIN=AVGSUM
      AVGMIN=AVGMIN/TMP
      WRITE(IOUT,101) AVGMIN,WMINMN
  101 FORMAT(1X,/1X,'AVERAGE SEED =',F11.8/1X,'MINIMUM SEED =',F11.8)
C
C4------CALCULATE AND PRINT ITERATION PARAMETERS FROM THE AVERAGE SEED
      P1=-ONE
      P2=NPARM-1
      DO 50 I=1,NPARM
      P1=P1+ONE
   50 W(I)=ONE-AVGMIN**(P1/P2)
      WRITE(IOUT,150) NPARM,(W(J),J=1,NPARM)
  150 FORMAT(1X,/1X,I5,' ITERATION PARAMETERS CALCULATED FROM',
     1       ' AVERAGE SEED:'//(1X,5E13.6))
C
C5------RETURN
      RETURN
      END
```

Slice-Successive Overrelaxation Package

There are no major changes to any modules in the Slice-Successive Overrelaxation Package.

SOR5AL

```
      SUBROUTINE SOR5AL(ISUM,LENX,LCA,LCRES,LCHDCG,LCLRCH,LCIEQP,
     1       MXITER,NCOL,NLAY,NSLICE,MBW,IN,IOUT,IFREFM)
C
C-----VERSION 0816 21FEB1996 SOR5AL
C     ******************************************************************
C     ALLOCATE STORAGE FOR SOR ARRAYS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------PRINT A MESSAGE IDENTIFYING SOR PACKAGE
      WRITE(IOUT,1)IN
    1 FORMAT(1X,
     1  /1X,'SOR5 -- SLICE-SUCCESSIVE OVERRELAXATION SOLUTION PACKAGE',
     2  /20X,'VERSION 5, 9/1/93 INPUT READ FROM UNIT',I3)
C
C2------READ AND PRINT MXITER (MAXIMUM # OF ITERATIONS)
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(I10)') MXITER
      ELSE
         READ(IN,*) MXITER
      END IF
      WRITE(IOUT,3) MXITER
    3 FORMAT(1X,I5,' ITERATIONS ALLOWED FOR SOR CLOSURE')
C
C3------ALLOCATE SPACE FOR THE SOR ARRAYS
      ISOLD=ISUM
      NSLICE=NCOL*NLAY
      MBW=NLAY+1
      LCA=ISUM
      ISUM=ISUM+NSLICE*MBW
      LCRES=ISUM
      ISUM=ISUM+NSLICE
      LCIEQP=ISUM
      ISUM=ISUM+NSLICE
      LCHDCG=ISUM
      ISUM=ISUM+MXITER
      LCLRCH=ISUM
      ISUM=ISUM+3*MXITER
      ISP=ISUM-ISOLD
C
C4------CALCULATE AND PRINT THE SPACE USED IN THE X ARRAY
      WRITE(IOUT,4) ISP
    4 FORMAT(1X,I10,' ELEMENTS IN X ARRAY ARE USED BY SOR')
      ISUM1=ISUM-1
      WRITE(IOUT,5) ISUM1,LENX
    5 FORMAT(1X,I10,' ELEMENTS OF X ARRAY USED OUT OF ',I10)
      IF(ISUM1.GT.LENX) WRITE(IOUT,6)
    6 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C5------RETURN
      RETURN
      END
```

Module SOR5RP

```
      SUBROUTINE SOR5RP(MXITER,ACCL,HCLOSE,IN,IPRSOR,IOUT,IFREFM)
C
C
C-----VERSION 0817 21FEB1996 SOR5RP
C     ******************************************************************
C     READ PARAMETERS FOR SOR
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
C
C1------READ THE ACCELERATION PARAMETER/RELAXATION FACTOR (ACCL) THE
C1------CLOSURE CRITERION (HCLOSE) AND THE NUMBER OF TIME STEPS
C1------BETWEEN PRINTOUTS OF MAXIMUM HEAD CHANGES (IPRSOR).
      IF(IFREFM.EQ.0) THEN
         READ(IN,'(2F10.0,I10)') ACCL,HCLOSE,IPRSOR
      ELSE
         READ(IN,*) ACCL,HCLOSE,IPRSOR
      END IF
      ZERO=0.
      IF(ACCL.EQ.ZERO) ACCL=1.
      IF(IPRSOR.LT.1) IPRSOR=999
C
C2------PRINT ACCL, HCLOSE, IPRSOR
      WRITE(IOUT,100)
  100 FORMAT(1X,///10X,'SOLUTION BY SLICE-SUCCESSIVE OVERRELAXATION'
     1    /10X,43('-'))
      WRITE(IOUT,115) MXITER
  115 FORMAT(1X,'MAXIMUM ITERATIONS ALLOWED FOR CLOSURE =',I9)
      WRITE(IOUT,120) ACCL
  120 FORMAT(1X,16X,'ACCELERATION PARAMETER =',G15.5)
      WRITE(IOUT,125) HCLOSE
  125 FORMAT(1X,5X,'HEAD CHANGE CRITERION FOR CLOSURE =',E15.5)
      WRITE(IOUT,130) IPRSOR
  130 FORMAT(1X,5X,'SOR HEAD CHANGE PRINTOUT INTERVAL =',I9)
C
C3------RETURN
      RETURN
      END
```

Module SOR5AP

```
      SUBROUTINE SOR5AP(HNEW,IBOUND,CR,CC,CV,HCOF,RHS,A,RES,IEQPNT,
     1      HDCG,LRCH,KITER,HCLOSE,ACCL,ICNVG,KSTP,KPER,
     2      IPRSOR,MXITER,NSTP,NCOL,NROW,NLAY,NSLICE,MBW,IOUT)
C-----VERSION 1537 31OCT1995 SOR5AP
C     ******************************************************************
C     SOLUTION BY SLICE-SUCCESSIVE OVERRELAXATION -- 1 ITERATION
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW,DIFF,DP,EE,R,HHCOF,DZERO
C
      DIMENSION HNEW(NCOL,NROW,NLAY), IBOUND(NCOL,NROW,NLAY),
     1   CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY),
     1   CV(NCOL,NROW,NLAY), HCOF(NCOL,NROW,NLAY), RHS(NCOL,NROW,NLAY),
     2   HDCG(MXITER), LRCH(3,MXITER),A(MBW,NSLICE),RES(NSLICE),
     3   IEQPNT(NLAY,NCOL)
C     ------------------------------------------------------------------
C
C1------CALCULATE # OF ELEMENTS IN COMPRESSED MATRIX A AND
C1------INITIALIZE FIELDS TO SAVE LARGEST HEAD CHANGE.
      NA=MBW*NSLICE
      ZERO=0.
      DZERO=0.
      BIG=ZERO
      ABSBIG=ZERO
      IB=0
      JB=0
      KB=0
C
C2------PROCESS EACH SLICE.
      DO 500 I=1,NROW
C
C3------CLEAR A.
      DO 110 J=1,NSLICE
      DO 110 K=1,MBW
  110 A(K,J)=ZERO
C
C4------ASSIGN A SEQUENCE # TO EACH VARIABLE HEAD CELL.
      NEQT=0
      DO 200 J=1,NCOL
      DO 200 K=1,NLAY
      IEQPNT(K,J)=0
      IF(IBOUND(J,I,K).LE.0) GO TO 200
      NEQT=NEQT+1
      IEQPNT(K,J)=NEQT
  200 CONTINUE
C
C5------FOR EACH CELL LOAD MATRIX A AND VECTOR RES.
      DO 300 J=1,NCOL
      DO 300 K=1,NLAY
C
C5A-----IF SEQUENCE # IS 0 (CELL IS EXTERNAL) GO ON TO NEXT CELL.
      NEQ=IEQPNT(K,J)
      IF(NEQ.EQ.0) GO TO 300
C
C5B-----INITIALIZE ACCUMULATORS EE AND R.
      EE=DZERO
      R=RHS(J,I,K)
C
C5C-----IF NODE TO LEFT SUBTRACT TERMS FROM EE AND R.
      IF(J.EQ.1) GO TO 120
      DP=CR(J-1,I,K)
      R=R-DP*HNEW(J-1,I,K)
      EE=EE-DP
C
C5D-----IF NODE TO RIGHT SUBTRACT TERMS FROM EE & R, MOVE COND TO A.
  120 IF(J.EQ.NCOL) GO TO 125
      SP=CR(J,I,K)
      DP=SP
      R=R-DP*HNEW(J+1,I,K)
      EE=EE-DP
      NXT=IEQPNT(K,J+1)
      IF(NXT.GT.0) A(1+NXT-NEQ,NEQ)=SP
C
C5E-----IF NODE TO REAR SUBTRACT TERMS FROM EE AND R.
  125 IF(I.EQ.1) GO TO 130
```

```
      DP=CC(J,I-1,K)
      R=R-DP*HNEW(J,I-1,K)
      EE=EE-DP
C
C5F-----IF NODE TO FRONT SUBTRACT TERMS FROM EE AND R.
  130 IF(I.EQ.NROW) GO TO 132
      DP=CC(J,I,K)
      R=R-DP*HNEW(J,I+1,K)
      EE=EE-DP
C
C5G-----IF NODE ABOVE SUBTRACT TERMS FROM EE AND R.
  132 IF(K.EQ.1) GO TO 134
      DP=CV(J,I,K-1)
      R=R-DP*HNEW(J,I,K-1)
      EE=EE-DP
C
C5H-----IF NODE BELOW SUBTRACT TERMS FROM EE & R AND MOVE COND TO A.
  134 IF(K.EQ.NLAY) GO TO 136
      SP=CV(J,I,K)
      DP=SP
      R=R-DP*HNEW(J,I,K+1)
      EE=EE-DP
      IF(IEQPNT(K+1,J).GT.0) A(2,NEQ)=SP
C
C5I-----MOVE EE INTO A, SUBTRACT EE TIMES LAST HEAD FROM R TO GET RES.
  136 HHCOF=HCOF(J,I,K)
      EE=EE+HHCOF
      A(1,NEQ)=EE
      RES(NEQ)=R-EE*HNEW(J,I,K)
  300 CONTINUE
C
C6------IF NO EQUATIONS GO TO NEXT SLICE, IF ONE EQUATION SOLVE
C6------DIRECTLY, IF 2 EQUATIONS CALL SSOR5B TO SOLVE FOR FIRST
C6------ESTIMATE OF HEAD CHANGE FOR THIS ITERATION.
      IF(NEQT.LT.1) GO TO 500
      IF(NEQT.EQ.1) RES(1)=RES(1)/A(1,1)
      IF(NEQT.GE.2) CALL SSOR5B(A,RES,NEQT,NA,MBW)
C
C7------FOR EACH CELL IN SLICE CALCULATE FINAL HEAD CHANGE THEN HEAD.
      DO 400 J=1,NCOL
      DO 400 K=1,NLAY
      NEQ=IEQPNT(K,J)
      IF(NEQ.EQ.0) GO TO 400
C
C7A-----MULTIPLY FIRST ESTIMATE OF HEAD CHANGE BY RELAX FACTOR TO
C7A-----GET FINAL ESTIMATE OF HEAD CHANGE FOR THIS ITERATION.
      DH=RES(NEQ)*ACCL
      DIFF=DH
C
C7B-----ADD FINAL ESTIMATE TO HEAD FROM LAST ITERATION TO GET HEAD
C7B-----FOR THIS ITERATION.
      HNEW(J,I,K)=HNEW(J,I,K)+DIFF
C
C7C-----SAVE FINAL HEAD CHANGE IF IT IS THE LARGEST.
      ABSDH=ABS(DH)
      IF(ABSDH.LE.ABSBIG) GO TO 400
      ABSBIG=ABSDH
      BIG=DH
      IB=I
      JB=J
      KB=K
  400 CONTINUE
C
C
  500 CONTINUE
C
C8------SAVE LARGEST HEAD CHANGE FOR THIS ITERATION.
      HDCG(KITER)=BIG
      LRCH(1,KITER)=KB
      LRCH(2,KITER)=IB
      LRCH(3,KITER)=JB
C
C9------IF LARGEST HEAD CHANGE IS SMALLER THAN CLOSURE THEN SET
C9------CONVERGE FLAG (ICNVG) EQUAL TO 1.
      ICNVG=0
      IF(ABSBIG.LE.HCLOSE) ICNVG=1
C
C10-----IF NOT CONVERGED AND NOT EXCEDED ITERATIONS THEN RETURN.
      IF(ICNVG.EQ.0 .AND. KITER.NE.MXITER) RETURN
      IF(KSTP.EQ.1) WRITE(IOUT,600)
  600 FORMAT(1X)
C
```

```
C11-----PRINT NUMBER OF ITERATIONS.
      WRITE(IOUT,601) KITER,KSTP,KPER
  601 FORMAT(1X,I5,' ITERATIONS FOR TIME STEP',I4,' IN STRESS PERIOD',
     1       I3)
C
C12-----IF FAILED TO CONVERGE, OR LAST TIME STEP, OR PRINTOUT
C12-----INTERVAL SPECIFIED BY USER IS HERE; THEN PRINT MAXIMUM
C12-----HEAD CHANGES FOR EACH ITERATION.
      IF(ICNVG.NE.0 .AND. KSTP.NE.NSTP .AND. MOD(KSTP,IPRSOR).NE.0)
     1      GO TO 700
      WRITE(IOUT,5)
    5 FORMAT(1X,/1X,'MAXIMUM HEAD CHANGE FOR EACH ITERATION:',/
     1       1X,/1X,5('   HEAD CHANGE'),/
     2         1X,5(' LAYER,ROW,COL')/1X,70('-'))
      NGRP=(KITER-1)/5 +1
      DO 620 K=1,NGRP
         L1=(K-1)*5 +1
         L2=L1+4
         IF(K.EQ.NGRP) L2=KITER
         WRITE(IOUT,618) (HDCG(J),J=L1,L2)
         WRITE(IOUT,619) ((LRCH(I,J),I=1,3),J=L1,L2)
  618    FORMAT(1X,5G14.4)
  619    FORMAT(1X,5(:' (',I3,',',I3,',',I3,')'))
  620 CONTINUE
      WRITE(IOUT,11)
   11 FORMAT(1X)
C
C13-----RETURN.
  700 RETURN
C
      END
```

Module SSOR5B

```
      SUBROUTINE SSOR5B(A,B,N,NA,MBW)
C
C
C-----VERSION 1634 29OCT1992 SSOR5B
C     ****************************************************************
C     SOLVE A SYMMETRIC SET OF EQUATIONS
C        A IS COEFFICIENT MATRIX IN COMPRESSED FORM
C        B IS RIGHT HAND SIDE AND IS REPLACED BY SOLUTION
C        N IS NUMBER OF EQUATIONS TO BE SOLVED
C        MBW IS BANDWIDTH OF A
C        NA IS ONE-DIMENSION SIZE OF A
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      DIMENSION A(NA),B(N)
C     ----------------------------------------------------------------
C
      NM1=N-1
      MBW1=MBW-1
      ID=1-MBW
      ZERO=0.
      ONE=1.
C
C1------SEQUENTIALLY USE EACH OF THE FIRST N-1 ROWS AS
C1------THE PIVOT ROW.
      DO 20 I=1,NM1
C
C2------CALCULATE THE INVERSE OF THE PIVOT.
      ID=ID+MBW
      C1=ONE/A(ID)
      LD=ID
      L=I
C
C3------FOR EACH ROW AFTER THE PIVOT ROW (THE TARGET ROW)
C3------ELIMINATE THE COLUMN CORRESPONDING TO THE PIVOT.
      DO 15 J=1,MBW1
      L=L+1
      IF(L.GT.N) GO TO 20
      IB=ID+J
C
C4------CALCULATE THE FACTOR NEEDED TO ELIMINATE A TERM IN THE
C4------TARGET ROW.
      C=A(IB)*C1
      LD=LD+MBW
      LB=LD-1
C
C5------MODIFY THE REST OF THE TERMS IN THE TARGET ROW.
      DO 10 K=J,MBW1
C
C6------SUBTRACT THE FACTOR TIMES A TERM IN THE PIVOT ROW
C6------FROM THE CORRESPONDING COLUMN IN THE TARGET ROW.
      LB=LB+1
      A(LB)=A(LB)-C*A(ID+K)
   10 CONTINUE
C
C7------MODIFY THE RIGHT SIDE OF THE EQUATION CORRESPONDING
C7------TO THE TARGET ROW.
      B(I+J)=B(I+J)-C*B(I)
   15 CONTINUE
   20 CONTINUE
      ID=ID+MBW
C
C8------SOLVE THE LAST EQUATION.
      B(N)=B(N)/A(ID)
C
C9------WORKING BACKWARDS SOLVE THE REST OF THE EQUATIONS.
      DO 70 I=1,NM1
      ID=ID-MBW
C
C10-----CLEAR THE ACCUMULATOR SUM.
      SUM=ZERO
      L=N-I
      MBW1M=MIN(MBW1,I)
C
C11-----ADD THE KNOWN TERMS IN EQUATION L TO SUM.
      DO 60 J=1,MBW1M
```

```
      SUM=SUM+A(ID+J)*B(L+J)
   60 CONTINUE
C
C12-----SOLVE FOR THE ONE UNKNOWN IN EQUATION L.
      B(L)=(B(L)-SUM)/A(ID)
   70 CONTINUE
C
C13-----RETURN.
      RETURN
      END
```

Utility Modules

Some of the utility modules have been modified to provide additional flexibility. For example, additional formats have been added to the modules that print arrays (ULAPRW and ULAPRS), and the array readers (U1DREL, U2DINT, and U2DREL) have been modified to read control records that optionally use an alphabetic word to control how the array is defined. Calling arguments to existing modules have not been modified so that other packages that make use of them should not require any changes. A new module was added to parse words from within a string of characters (URWORD). New modules were also added to save budget data more compactly (UBDSV1, UBDSV2, UBDSVA, and UBDSV3) and to save formatted layer data (ULASV2).

In the lists of variables for utility modules, the "Argument" variable type has been added. The Argument type refers to dummy subroutine arguments for which the variable type can vary because the utility modules can be called by many packages in differing contexts. For any specific call to a utility module, each actual argument would be one of the standard types, but the type of the actual argument could change from call to call. However, in situations where a dummy argument always becomes a Global actual argument, that variable is specified as Global. Also, the Package variable type is not used in any of the utility modules because the utility modules are not considered to be directly a part of any package. Each utility module stands alone.

Module UBUDSV

```
      SUBROUTINE UBUDSV(KSTP,KPER,TEXT,IBDCHN,BUFF,NCOL,NROW,NLAY,IOUT)
C
C
C-----VERSION 1039 26JUNE1992 UBUDSV
C     ***************************************************************
C     RECORD CELL-BY-CELL FLOW TERMS FOR ONE COMPONENT OF FLOW.
C     ***************************************************************
C
C        SPECIFICATIONS:
C     -----------------------------------------------------------------
      CHARACTER*16 TEXT
      DIMENSION BUFF(NCOL,NROW,NLAY)
C     -----------------------------------------------------------------
C
C1------WRITE AN UNFORMATTED RECORD IDENTIFYING DATA.
      WRITE(IOUT,1) TEXT,IBDCHN,KSTP,KPER
    1 FORMAT(1X,'UBUDSV SAVING "',A16,'" ON UNIT',I3,
     1     ' AT TIME STEP',I3,', STRESS PERIOD',I3)
      WRITE(IBDCHN) KSTP,KPER,TEXT,NCOL,NROW,NLAY
C
C2------WRITE AN UNFORMATTED RECORD CONTAINING VALUES FOR
C2------EACH CELL IN THE GRID.
      WRITE(IBDCHN) BUFF
C
C3------RETURN
      RETURN
      END
```

Module ULASAV

```
      SUBROUTINE ULASAV(BUF,TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                  NROW,ILAY,ICHN)
C
C-----VERSION 1642 12MAY1987 ULASAV
C     ******************************************************************
C     SAVE 1 LAYER ARRAY ON DISK
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*4 TEXT
      DIMENSION BUF(NCOL,NROW),TEXT(4)
C     ------------------------------------------------------------------
C
C1------WRITE AN UNFORMATTED RECORD CONTAINING IDENTIFYING
C1------INFORMATION.
      WRITE(ICHN) KSTP,KPER,PERTIM,TOTIM,TEXT,NCOL,NROW,ILAY
C
C2------WRITE AN UNFORMATTED RECORD CONTAINING ARRAY VALUES
C2------THE ARRAY IS DIMENSIONED (NCOL,NROW)
      WRITE(ICHN) ((BUF(IC,IR),IC=1,NCOL),IR=1,NROW)
C
C3------RETURN
      RETURN
      END
```

Module ULAPRS

```
      SUBROUTINE ULAPRS(BUF,TEXT,KSTP,KPER,NCOL,NROW,ILAY,IPRN,IOUT)
C
C
C-----VERSION 0755 01NOV1995 ULAPRS
C     ******************************************************************
C     PRINT A 1 LAYER ARRAY IN STRIPS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 TEXT
      DIMENSION BUF(NCOL,NROW)
C     ------------------------------------------------------------------
C
C1------MAKE SURE THE FORMAT CODE (IP OR IPRN) IS BETWEEN 1
C1------AND 18.
      IP=IPRN
      IF(IP.LT.1 .OR. IP.GT.18) IP=12
C
C2------DETERMINE THE NUMBER OF VALUES (NCAP) PRINTED ON ONE LINE.
      NCAP=10
      IF(IP.EQ.1) NCAP=11
      IF(IP.EQ.2) NCAP=9
      IF(IP.GT.2 .AND. IP.LT.7) NCAP=15
      IF(IP.GT.6 .AND. IP.LT.12) NCAP=20
C
C3------CALCULATE THE NUMBER OF STRIPS (NSTRIP).
      NCPF=129/NCAP
      IF(IP.GE.13) NCPF=7
      ISP=0
      IF(NCAP.GT.12 .OR. IP.GE.13) ISP=3
      NSTRIP=(NCOL-1)/NCAP + 1
      J1=1-NCAP
      J2=0
C
C4------LOOP THROUGH THE STRIPS.
      DO 2000 N=1,NSTRIP
C
C5------CALCULATE THE FIRST(J1) & THE LAST(J2) COLUMNS FOR THIS STRIP
      J1=J1+NCAP
      J2=J2+NCAP
      IF(J2.GT.NCOL) J2=NCOL
C
C6-------PRINT TITLE ON EACH STRIP DEPENDING ON ILAY
      IF(ILAY.GT.0) THEN
         WRITE(IOUT,1) TEXT,ILAY,KSTP,KPER
    1    FORMAT('1',/2X,A,' IN LAYER',I3,' AT END OF TIME STEP',I3,
     1     ' IN STRESS PERIOD',I3/2X,71('-'))
      ELSE IF(ILAY.LT.0) THEN
         WRITE(IOUT,2) TEXT,KSTP,KPER
    2    FORMAT('1',/2X,A,' FOR CROSS SECTION AT END OF TIME STEP',I3,
     1     ' IN STRESS PERIOD',I3/2X,77('-'))
      END IF
C
C7------PRINT COLUMN NUMBERS ABOVE THE STRIP
      CALL UCOLNO(J1,J2,ISP,NCAP,NCPF,IOUT)
C
C8------LOOP THROUGH THE ROWS PRINTING COLS J1 THRU J2 WITH FORMAT IP
      DO 1000 I=1,NROW
      GO TO(10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,170,
     1      180), IP
C
C------------FORMAT 10G10.3
   10 WRITE(IOUT,11) I,(BUF(J,I),J=J1,J2)
   11 FORMAT(1X,I3,2X,1PG10.3,10(1X,G10.3))
      GO TO 1000
C
C------------FORMAT 8G13.6
   20 WRITE(IOUT,21) I,(BUF(J,I),J=J1,J2)
   21 FORMAT(1X,I3,2X,1PG13.6,8(1X,G13.6))
      GO TO 1000
C
C------------FORMAT 15F7.1
   30 WRITE(IOUT,31) I,(BUF(J,I),J=J1,J2)
   31 FORMAT(1X,I3,1X,15(1X,F7.1))
      GO TO 1000
C
C------------FORMAT 15F7.2
```

```
   40 WRITE(IOUT,41) I,(BUF(J,I),J=J1,J2)
   41 FORMAT(1X,I3,1X,15(1X,F7.2))
      GO TO 1000
C
C------------FORMAT 15F7.3
   50 WRITE(IOUT,51) I,(BUF(J,I),J=J1,J2)
   51 FORMAT(1X,I3,1X,15(1X,F7.3))
      GO TO 1000
C
C------------FORMAT 15F7.4
   60 WRITE(IOUT,61) I,(BUF(J,I),J=J1,J2)
   61 FORMAT(1X,I3,1X,15(1X,F7.4))
      GO TO 1000
C
C------------FORMAT 20F5.0
   70 WRITE(IOUT,71) I,(BUF(J,I),J=J1,J2)
   71 FORMAT(1X,I3,1X,20(1X,F5.0))
      GO TO 1000
C
C------------FORMAT 20F5.1
   80 WRITE(IOUT,81) I,(BUF(J,I),J=J1,J2)
   81 FORMAT(1X,I3,1X,20(1X,F5.1))
      GO TO 1000
C
C------------FORMAT 20F5.2
   90 WRITE(IOUT,91) I,(BUF(J,I),J=J1,J2)
   91 FORMAT(1X,I3,1X,20(1X,F5.2))
      GO TO 1000
C
C------------FORMAT 20F5.3
  100 WRITE(IOUT,101) I,(BUF(J,I),J=J1,J2)
  101 FORMAT(1X,I3,1X,20(1X,F5.3))
      GO TO 1000
C
C------------FORMAT 20F5.4
  110 WRITE(IOUT,111) I,(BUF(J,I),J=J1,J2)
  111 FORMAT(1X,I3,1X,20(1X,F5.4))
      GO TO 1000
C
C------------FORMAT 9G11.4
  120 WRITE(IOUT,121) I,(BUF(J,I),J=J1,J2)
  121 FORMAT(1X,I3,2X,1PG11.4,9(1X,G11.4))
      GO TO 1000
C
C------------FORMAT 10F6.0
  130 WRITE(IOUT,131) I,(BUF(J,I),J=J1,J2)
  131 FORMAT(1X,I3,1X,10(1X,F6.0))
      GO TO 1000
C
C------------FORMAT 10F6.1
  140 WRITE(IOUT,141) I,(BUF(J,I),J=J1,J2)
  141 FORMAT(1X,I3,1X,10(1X,F6.1))
      GO TO 1000
C
C------------FORMAT 10F6.2
  150 WRITE(IOUT,151) I,(BUF(J,I),J=J1,J2)
  151 FORMAT(1X,I3,1X,10(1X,F6.2))
      GO TO 1000
C
C------------FORMAT 10F6.3
  160 WRITE(IOUT,161) I,(BUF(J,I),J=J1,J2)
  161 FORMAT(1X,I3,1X,10(1X,F6.3))
      GO TO 1000
C
C------------FORMAT 10F6.4
  170 WRITE(IOUT,171) I,(BUF(J,I),J=J1,J2)
  171 FORMAT(1X,I3,1X,10(1X,F6.4))
      GO TO 1000
C
C------------FORMAT 10F6.5
  180 WRITE(IOUT,181) I,(BUF(J,I),J=J1,J2)
  181 FORMAT(1X,I3,1X,10(1X,F6.5))
C
 1000 CONTINUE
 2000 CONTINUE
C
C9------RETURN
      RETURN
      END
```

Module ULAPRW

```
      SUBROUTINE ULAPRW(BUF,TEXT,KSTP,KPER,NCOL,NROW,ILAY,IPRN,IOUT)
C
C
C-----VERSION 0758 01NOV1995 ULAPRW
C     ******************************************************************
C     PRINT 1 LAYER ARRAY
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 TEXT
      DIMENSION BUF(NCOL,NROW)
C     ------------------------------------------------------------------
C
C1------PRINT A HEADER DEPENDING ON ILAY
      IF(ILAY.GT.0) THEN
         WRITE(IOUT,1) TEXT,ILAY,KSTP,KPER
    1    FORMAT('1',/2X,A,' IN LAYER',I3,' AT END OF TIME STEP',I3,
     1     ' IN STRESS PERIOD',I3/2X,71('-'))
      ELSE IF(ILAY.LT.0) THEN
         WRITE(IOUT,2) TEXT,KSTP,KPER
    2    FORMAT('1',/2X,A,' FOR CROSS SECTION AT END OF TIME STEP',I3,
     1     ' IN STRESS PERIOD',I3/2X,77('-'))
      END IF
C
C2------MAKE SURE THE FORMAT CODE (IP OR IPRN) IS
C2------BETWEEN 1 AND 13.
    5 IP=IPRN
      IF(IP.LT.1 .OR. IP.GT.18) IP=12
C
C3------CALL THE UTILITY MODULE UCOLNO TO PRINT COLUMN NUMBERS.
      IF(IP.EQ.1) CALL UCOLNO(1,NCOL,0,11,11,IOUT)
      IF(IP.EQ.2) CALL UCOLNO(1,NCOL,0,9,14,IOUT)
      IF(IP.GT.2 .AND. IP.LT.7) CALL UCOLNO(1,NCOL,3,15,8,IOUT)
      IF(IP.GT.6 .AND. IP.LT.12) CALL UCOLNO(1,NCOL,3,20,6,IOUT)
      IF(IP.EQ.12) CALL UCOLNO(1,NCOL,0,10,12,IOUT)
      IF(IP.GE.13 .AND. IP.LE.18) CALL UCOLNO(1,NCOL,3,10,7,IOUT)
C
C4------LOOP THROUGH THE ROWS PRINTING EACH ONE IN ITS ENTIRETY.
      DO 1000 I=1,NROW
      GO TO(10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,170,
     1      180), IP
C
C------------ FORMAT 11G10.3
   10 WRITE(IOUT,11) I,(BUF(J,I),J=1,NCOL)
   11 FORMAT(1X,I3,2X,1PG10.3,10(1X,G10.3):/(5X,11(1X,G10.3)))
      GO TO 1000
C
C------------ FORMAT 9G13.6
   20 WRITE(IOUT,21) I,(BUF(J,I),J=1,NCOL)
   21 FORMAT(1X,I3,2X,1PG13.6,8(1X,G13.6):/(5X,9(1X,G13.6)))
      GO TO 1000
C
C------------ FORMAT 15F7.1
   30 WRITE(IOUT,31) I,(BUF(J,I),J=1,NCOL)
   31 FORMAT(1X,I3,1X,15(1X,F7.1):/(5X,15(1X,F7.1)))
      GO TO 1000
C
C------------ FORMAT 15F7.2
   40 WRITE(IOUT,41) I,(BUF(J,I),J=1,NCOL)
   41 FORMAT(1X,I3,1X,15(1X,F7.2):/(5X,15(1X,F7.2)))
      GO TO 1000
C
C------------ FORMAT 15F7.3
   50 WRITE(IOUT,51) I,(BUF(J,I),J=1,NCOL)
   51 FORMAT(1X,I3,1X,15(1X,F7.3):/(5X,15(1X,F7.3)))
      GO TO 1000
C
C------------ FORMAT 15F7.4
   60 WRITE(IOUT,61) I,(BUF(J,I),J=1,NCOL)
   61 FORMAT(1X,I3,1X,15(1X,F7.4):/(5X,15(1X,F7.4)))
      GO TO 1000
C
C------------ FORMAT 20F5.0
   70 WRITE(IOUT,71) I,(BUF(J,I),J=1,NCOL)
   71 FORMAT(1X,I3,1X,20(1X,F5.0):/(5X,20(1X,F5.0)))
      GO TO 1000
```

```
C
C------------ FORMAT 20F5.1
   80 WRITE(IOUT,81) I,(BUF(J,I),J=1,NCOL)
   81 FORMAT(1X,I3,1X,20(1X,F5.1):/(5X,20(1X,F5.1)))
      GO TO 1000
C
C------------ FORMAT 20F5.2
   90 WRITE(IOUT,91) I,(BUF(J,I),J=1,NCOL)
   91 FORMAT(1X,I3,1X,20(1X,F5.2):/(5X,20(1X,F5.2)))
      GO TO 1000
C
C------------ FORMAT 20F5.3
  100 WRITE(IOUT,101) I,(BUF(J,I),J=1,NCOL)
  101 FORMAT(1X,I3,1X,20(1X,F5.3):/(5X,20(1X,F5.3)))
      GO TO 1000
C
C------------ FORMAT 20F5.4
  110 WRITE(IOUT,111) I,(BUF(J,I),J=1,NCOL)
  111 FORMAT(1X,I3,1X,20(1X,F5.4):/(5X,20(1X,F5.4)))
      GO TO 1000
C
C------------ FORMAT 10G11.4
  120 WRITE(IOUT,121) I,(BUF(J,I),J=1,NCOL)
  121 FORMAT(1X,I3,2X,1PG11.4,9(1X,G11.4):/(5X,10(1X,G11.4)))
      GO TO 1000
C
C------------ FORMAT 10F6.0
  130 WRITE(IOUT,131) I,(BUF(J,I),J=1,NCOL)
  131 FORMAT(1X,I3,1X,10(1X,F6.0):/(5X,10(1X,F6.0)))
      GO TO 1000
C
C------------ FORMAT 10F6.1
  140 WRITE(IOUT,141) I,(BUF(J,I),J=1,NCOL)
  141 FORMAT(1X,I3,1X,10(1X,F6.1):/(5X,10(1X,F6.1)))
      GO TO 1000
C
C------------ FORMAT 10F6.2
  150 WRITE(IOUT,151) I,(BUF(J,I),J=1,NCOL)
  151 FORMAT(1X,I3,1X,10(1X,F6.2):/(5X,10(1X,F6.2)))
      GO TO 1000
C
C------------ FORMAT 10F6.3
  160 WRITE(IOUT,161) I,(BUF(J,I),J=1,NCOL)
  161 FORMAT(1X,I3,1X,10(1X,F6.3):/(5X,10(1X,F6.3)))
      GO TO 1000
C
C------------ FORMAT 10F6.4
  170 WRITE(IOUT,171) I,(BUF(J,I),J=1,NCOL)
  171 FORMAT(1X,I3,1X,10(1X,F6.4):/(5X,10(1X,F6.4)))
      GO TO 1000
C
C------------ FORMAT 10F6.5
  180 WRITE(IOUT,181) I,(BUF(J,I),J=1,NCOL)
  181 FORMAT(1X,I3,1X,10(1X,F6.5):/(5X,10(1X,F6.5)))
C
 1000 CONTINUE
C
C5------RETURN
      RETURN
      END
```

Module UCOLNO

```
      SUBROUTINE UCOLNO(NLBL1,NLBL2,NSPACE,NCPL,NDIG,IOUT)
C
C
C-----VERSION 0934 22JUNE1992 UCOLNO
C     ****************************************************************
C     OUTPUT COLUMN NUMBERS ABOVE A MATRIX PRINTOUT
C        NLBL1 IS THE START COLUMN LABEL (NUMBER)
C        NLBL2 IS THE STOP COLUMN LABEL (NUMBER)
C        NSPACE IS NUMBER OF BLANK SPACES TO LEAVE AT START OF LINE
C        NCPL IS NUMBER OF COLUMN NUMBERS PER LINE
C        NDIG IS NUMBER OF CHARACTERS IN EACH COLUMN FIELD
C        IOUT IS OUTPUT CHANNEL
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      CHARACTER*1 DOT,SPACE,DG,BF
      DIMENSION BF(130),DG(10)
C
      DATA DG(1),DG(2),DG(3),DG(4),DG(5),DG(6),DG(7),DG(8),DG(9),DG(10)/
     1         '0','1','2','3','4','5','6','7','8','9'/
      DATA DOT,SPACE/'.',' '/
C     ----------------------------------------------------------------
C
C1------CALCULATE # OF COLUMNS TO BE PRINTED (NLBL), WIDTH
C1------OF A LINE (NTOT), NUMBER OF LINES (NWRAP).
      WRITE(IOUT,1)
    1 FORMAT(1X)
      NLBL=NLBL2-NLBL1+1
      N=NLBL
      IF(NLBL.GT.NCPL) N=NCPL
      NTOT=NSPACE+N*NDIG
      IF(NTOT.GT.130) GO TO 50
      NWRAP=(NLBL-1)/NCPL + 1
      J1=NLBL1-NCPL
      J2=NLBL1-1
C
C2------BUILD AND PRINT EACH LINE
      DO 40 N=1,NWRAP
C
C3------CLEAR THE BUFFER (BF).
      DO 20 I=1,130
      BF(I)=SPACE
   20 CONTINUE
      NBF=NSPACE
C
C4------DETERMINE FIRST (J1) AND LAST (J2) COLUMN # FOR THIS LINE.
      J1=J1+NCPL
      J2=J2+NCPL
      IF(J2.GT.NLBL2) J2=NLBL2
C5------LOAD THE COLUMN #'S INTO THE BUFFER.
      DO 30 J=J1,J2
      NBF=NBF+NDIG
      I2=J/10
      I1=J-I2*10+1
      BF(NBF)=DG(I1)
      IF(I2.EQ.0) GO TO 30
      I3=I2/10
      I2=I2-I3*10+1
      BF(NBF-1)=DG(I2)
      IF(I3.EQ.0) GO TO 30
      BF(NBF-2)=DG(I3+1)
   30 CONTINUE
C
C6------PRINT THE CONTENTS OF THE BUFFER (I.E. PRINT THE LINE).
      WRITE(IOUT,31) (BF(I),I=1,NBF)
   31 FORMAT(1X,130A1)
C
   40 CONTINUE
C
C7------PRINT A LINE OF DOTS (FOR ESTHETIC PURPOSES ONLY).
   50 NTOT=NTOT
      IF(NTOT.GT.130) NTOT=130
      WRITE(IOUT,51) (DOT,I=1,NTOT)
   51 FORMAT(1X,130A1)
C
C8------RETURN
      RETURN
      END
```

Module U2DREL

Narrative for Module U2DREL

Module U2DREL defines values for a two-dimensional real array. It first reads an "array control record"; then based upon the contents of the control record, it defines the values for all elements in the array. The elements are either all set to the same value, or they are individually read from a file. If array elements are read from a file, the file may be formatted or unformatted. Variable LOCAT is a numeric code that controls all of these options. LOCAT is either directly read from the control record, or it is defined by an alphabetic word in the control record. If one of the allowed words is not found as the first field of the record, then a numeric code is assumed. U2DREL performs its functions as follows:

1. Read the array control record as character data (CHARACTER variable CNTRL).

2. Use URWORD to look for an alphabetic word -- either "CONSTANT', "INTERNAL", EXTERNAL", or "OPEN/CLOSE". If found, set IFREE flag to 1 and set LOCAT to the appropriate value:

"CONSTANT" -- all elements of array are the same value, which is CNSTNT; set LOCAT = 0

"INTERNAL" -- elements of array will be read on same unit as control record: set LOCAT = IN

"EXTERNAL" -- elements of array are read on the unit specified in the following field in the control record: call URWORD to obtain this value and store it in LOCAT.

"OPEN/CLOSE" -- elements of array are read from the file whose name is specified in the following field. Call URWORD to obtain this file name, store its name in FNAME, write its name, set LOCAT equal to NUNOPN (99), and set flag ICLOSE to 1 to indicate that this file should be opened prior to reading and closed after reading.

A. Alphabetic word was not found; set IFREE flag to 0 and read the control record using a fixed format.

3. If free-format control record is being used (IREE not 0), parse the remaining fields. First get the value for CNSTNT, which is always required. If LOCAT is not 0, a format and print code are required. After getting the format, check for an OPEN/CLOSE file; if so, open the file as either formatted or unformatted. If the fomat is 'BINARY', then set LOCAT to a negative value to indicated an unformatted file.

4. Test LOCAT to determine how to define elements in array.

A. LOCAT is 0. Set all elements of array equal to CNSTNT. Print the value, and RETURN.

B. LOCAT > 0. Read formatted records on unit LOCAT. Prior to reading the data, print a heading for the array. There are 3 forms for the header: array name with a layer specifier, array name for a cross section, and array name without any other designation. If format in FMTIN is "(FREE)", then read the data using free format; otherwise, use FMTIN. After reading the array go to step 5.

C. LOCAT < 0. Prior to reading unformatted data, print a heading for the array. There are 3 forms for the header: array name with a layer specifier, array name for a cross section, and array name without any other designation. Read the unformatted records on unit -LOCAT. First read a header record, and then read a record containing all elements in the array.

5. Close the file if ICLOSE is not 0. If CNSTNT is not 0, multiply all elements in array by CNSTNT.

6. If the print code, IPRN, is greater than or equal to 0, call ULAPRW to print array.

7. RETURN.

8. Error reading control record. Write a message, and STOP.

U2DREL

```
      SUBROUTINE U2DREL(A,ANAME,II,JJ,K,IN,IOUT)
C
C
C-----VERSION 1539 22JUNE1993 U2DREL
C     ****************************************************************
C     ROUTINE TO INPUT 2-D REAL DATA MATRICES
C       A IS ARRAY TO INPUT
C       ANAME IS 24 CHARACTER DESCRIPTION OF A
C       II IS NO. OF ROWS
C       JJ IS NO. OF COLS
C       K IS LAYER NO. (USED WITH NAME TO TITLE PRINTOUT --)
C              IF K=0, NO LAYER IS PRINTED
C              IF K<0, CROSS SECTION IS PRINTED)
C       IN IS INPUT UNIT
C       IOUT IS OUTPUT UNIT
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      CHARACTER*24 ANAME
      DIMENSION A(JJ,II)
      CHARACTER*20 FMTIN
      CHARACTER*80 CNTRL
      CHARACTER*16 TEXT
      CHARACTER*80 FNAME
      DATA NUNOPN/99/
C     ----------------------------------------------------------------
C
C1------READ ARRAY CONTROL RECORD AS CHARACTER DATA.
      READ(IN,'(A)') CNTRL
C
C2------LOOK FOR ALPHABETIC WORD THAT INDICATES THAT THE RECORD IS FREE
C2------FORMAT.  SET A FLAG SPECIFYING IF FREE FORMAT OR FIXED FORMAT.
      ICLOSE=0
      IFREE=1
      ICOL=1
      CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,1,N,R,IOUT,IN)
      IF (CNTRL(ISTART:ISTOP).EQ.'CONSTANT') THEN
         LOCAT=0
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'INTERNAL') THEN
         LOCAT=IN
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'EXTERNAL') THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,2,LOCAT,R,IOUT,IN)
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'OPEN/CLOSE') THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,0,N,R,IOUT,IN)
         FNAME=CNTRL(ISTART:ISTOP)
         LOCAT=NUNOPN
         WRITE(IOUT,15) LOCAT,FNAME
   15    FORMAT(1X,/1X,'OPENING FILE ON UNIT',I4,':',/1X,A)
         ICLOSE=1
      ELSE
C
C2A-----DID NOT FIND A RECOGNIZED WORD, SO NOT USING FREE FORMAT.
C2A-----READ THE CONTROL RECORD THE ORIGINAL WAY.
         IFREE=0
         READ(CNTRL,1,ERR=500) LOCAT,CNSTNT,FMTIN,IPRN
    1    FORMAT(I10,F10.0,A20,I10)
      END IF
C
C3------FOR FREE FORMAT CONTROL RECORD, READ REMAINING FIELDS.
      IF(IFREE.NE.0) THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,3,N,CNSTNT,IOUT,IN)
         IF(LOCAT.NE.0) THEN
            CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,1,N,R,IOUT,IN)
            FMTIN=CNTRL(ISTART:ISTOP)
            IF(ICLOSE.NE.0) THEN
               IF(FMTIN.EQ.'(BINARY)') THEN
                  OPEN(UNIT=LOCAT,FILE=FNAME,FORM='UNFORMATTED')
               ELSE
                  OPEN(UNIT=LOCAT,FILE=FNAME)
               END IF
            END IF
            IF(LOCAT.GT.0 .AND. FMTIN.EQ.'(BINARY)') LOCAT=-LOCAT
            CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,2,IPRN,R,IOUT,IN)
         END IF
      END IF
C
```

```
C4------TEST LOCAT TO SEE HOW TO DEFINE ARRAY VALUES.
      IF(LOCAT) 200,50,90
C
C4A-----LOCAT=0; SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN.
   50 DO 80 I=1,II
      DO 80 J=1,JJ
   80 A(J,I)=CNSTNT
      IF(K.GT.0) WRITE(IOUT,2) ANAME,CNSTNT,K
    2 FORMAT(1X,/1X,A,' =',G15.7,' FOR LAYER',I4)
      IF(K.LE.0) WRITE(IOUT,3) ANAME,CNSTNT
    3 FORMAT(1X,/1X,A,' =',G15.7)
      RETURN
C
C4B-----LOCAT>0; READ FORMATTED RECORDS USING FORMAT FMTIN.
   90 IF(K.GT.0) THEN
         WRITE(IOUT,94) ANAME,K,LOCAT,FMTIN
   94    FORMAT(1X,///11X,A,' FOR LAYER',I4,/
     1       1X,'READING ON UNIT',I4,' WITH FORMAT: ',A)
      ELSE IF(K.EQ.0) THEN
         WRITE(IOUT,95) ANAME,LOCAT,FMTIN
   95    FORMAT(1X,///11X,A,/
     1       1X,'READING ON UNIT',I4,' WITH FORMAT: ',A)
      ELSE
         WRITE(IOUT,96) ANAME,LOCAT,FMTIN
   96    FORMAT(1X,///11X,A,' FOR CROSS SECTION',/
     1       1X,'READING ON UNIT',I4,' WITH FORMAT: ',A)
      END IF
      DO 100 I=1,II
      IF(FMTIN.EQ.'(FREE)') THEN
         READ(LOCAT,*) (A(J,I),J=1,JJ)
      ELSE
         READ(LOCAT,FMTIN) (A(J,I),J=1,JJ)
      END IF
  100 CONTINUE
      GO TO 300
C
C4C-----LOCAT<0; READ UNFORMATTED ARRAY VALUES.
  200 LOCAT=-LOCAT
      IF(K.GT.0) THEN
         WRITE(IOUT,201) ANAME,K,LOCAT
  201    FORMAT(1X,///11X,A,' FOR LAYER',I4,/
     1    1X,'READING BINARY ON UNIT',I4)
      ELSE IF(K.EQ.0) THEN
         WRITE(IOUT,202) ANAME,LOCAT
  202    FORMAT(1X,///1X,A,/
     1    1X,'READING BINARY ON UNIT',I4)
      ELSE
         WRITE(IOUT,203) ANAME,LOCAT
  203    FORMAT(1X,///1X,A,' FOR CROSS SECTION',/
     1    1X,'READING BINARY ON UNIT',I4)
      END IF
      READ(LOCAT) KSTP,KPER,PERTIM,TOTIM,TEXT,NCOL,NROW,ILAY
      READ(LOCAT) A
C
C5------IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.
  300 IF(ICLOSE.NE.0) CLOSE(UNIT=LOCAT)
      ZERO=0.
      IF(CNSTNT.EQ.ZERO) GO TO 320
      DO 310 I=1,II
      DO 310 J=1,JJ
      A(J,I)=A(J,I)*CNSTNT
  310 CONTINUE
C
C6------IF PRINT CODE (IPRN) >0 OR =0 THEN PRINT ARRAY VALUES.
  320 IF(IPRN.GE.0) CALL ULAPRW(A,ANAME,0,0,JJ,II,0,IPRN,IOUT)
C
C7------RETURN
      RETURN
C
C8------CONTROL RECORD ERROR.
  500 IF(K.GT.0) THEN
         WRITE(IOUT,501) ANAME,K
  501    FORMAT(1X,/1X,'ERROR READING ARRAY CONTROL RECORD FOR ',A,
     1     ' FOR LAYER',I4,':')
      ELSE
         WRITE(IOUT,502) ANAME
  502    FORMAT(1X,/1X,'ERROR READING ARRAY CONTROL RECORD FOR ',A,':')
      END IF
      WRITE(IOUT,'(1X,A)') CNTRL
      STOP
      END
```

List of Variables for Module U2DREL

Variable	Range	Definition
A	Argument	DIMENSION (JJ,II), Array to be defined.
ANAME	Argument	CHARACTER*24, Label that identifies array A.
CNSTNT	Module	Constant used in defining array A. If the array is a constant, CNSTNT is that constant. If each element is read, the values are multipled by CNSTNT.
CNTRL	Module	CHARACTER*80, The control record that has been read as text so that it can be parsed for words by Module URWORD.
FMTIN	Module	CHARACTER*20, Format that will be used to read array A.
FNAME	Module	CHARACTER*80, Name of file when U2DREL opens the file from which array A will be read.
I	Module	Index for rows.
ICLOSE	Module	Flag used to indicate if the file from which array A is being read is an OPEN / CLOSE file.
ICOL	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.
IFREE	Module	Flag that is not 0 when free-format (alphabetic) control record is being used.
II	Argument	Number of rows in array A.
ILAY	Module	Identifier read from first record of an unformatted file: layer number.
IN	Argument	The unit number from which the control record is read.
IOUT	Global	Unit number for writing to the listing file.
IPRN	Module	Code that indicates which format to use to print array A.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
J	Module	Index for columns.
JJ	Argument	Number of columns in array A.
K	Argument	The layer to which array A is associated. If K is 0, there is no layer; if K < 0, array A is a cross section.
KPER	Module	Identifier read from first record of an unformatted file: stress period.
KSTP	Module	Identifier read from first record of an unformatted file: time step.
LOCAT	Module	Code that indicates how the elements in array A are defined: > 0, formatted values are read using LOCAT as the unit. = 0, all elements are set equal to CNSTNT. < 0, unformatted values are read using -LOCAT as the unit.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NCOL	Module	Identifier read from first record of an unformatted file: number of columns.
NROW	Module	Identifier read from first record of an unformatted file: number of rows.
NUNOPN	Module	Unit used when U2DREL opens the file from which array A is read.
PERTIM	Module	Identifier read from first record of an unformatted file: time within the stress period.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.
TEXT	Module	Identifier read from first record of an unformatted file: name of array.

TOTIM	Module	Identifier read from first record of an unformatted file: simulation time.
ZERO	Module	The constant 0.

Module U2DINT

Narrative for Module U2DINT

Module U2DINT defines values for a two-dimensional real array. It first reads an "array control record"; then based upon the contents of the control record, it defines the values for all elements in the array. The elements are either all set to the same value, or they are individually read from a file. If array elements are read from a file, the file may be formatted or unformatted. Variable LOCAT is a numeric code that controls all of these options. LOCAT is either directly read from the control record, or it is defined by an alphabetic word in the control record. If one of the allowed words is not found as the first field of the record, then a numeric code is assumed. U2DINT performs its functions as follows:

1. Read the array control record as character data (CHARACTER variable CNTRL).

2. Use URWORD to look for an alphabetic word -- either "CONSTANT', "INTERNAL", or EXTERNAL". If found, set IFREE flag to 1 and set LOCAT to the appropriate value:

"CONSTANT" -- all elements of array are the same value, which is CNSTNT; set LOCAT = 0

"INTERNAL" -- elements of array will be read on same unit as control record: set LOCAT = IN

"EXTERNAL" -- elements of array are read on the unit specified in the following field in the control record: call URWORD to obtain this value and store it in LOCAT.

"OPEN / CLOSE" -- elements of array are read from the file whose name is specified in the following field. Call URWORD to obtain this file name, store its name in FNAME, write its name, set LOCAT equal to NUNOPN (99), and set flag ICLOSE to 1 to indicate that this file should be opened prior to reading and closed after reading.

A. Alphabetic word was not found; set IFREE flag to 0 and read the control record using a fixed format.

3. If free-format control record is being used (IREE not 0), parse the remaining fields. First get the value for CNSTNT, which is always required. If LOCAT is not 0, a format and print code are required. After getting the format, check for an OPEN / CLOSE file; if so, open the file as either formatted or unformatted. If the fomat is 'BINARY', then set LOCAT to a negative value to indicated an unformatted file.

4. Test LOCAT to determine how to define elements in array.

A. LOCAT is 0. Set all elements of array equal to ICONST. Print the value, and RETURN.

B. LOCAT > 0. Read formatted records on unit LOCAT. Prior to reading the data, print a heading for the array. There are 3 forms for the header: array name with a layer specifier, array name for a cross section, and array name without any other designation. If format in FMTIN is "(FREE)", then read the data using free format; otherwise, use FMTIN. After reading the array go to step 5.

C. LOCAT < 0. Prior to reading unformatted data, print a heading for the array. There are 3 forms for the header: array name with a layer specifier, array name for a cross section, and array name without any other designation. Negate LOCAT to obtain the unit number for reading the unformatted records. First read a header record, and then read a record containing all elements in the array.

5. Close the file if ICLOSE is not 0. If ICONST is not 0, multiply all elements in array by ICONST.

6. If the print code, IPRN, is less than 0, RETURN.

7. Array will be printed. If print code is out of range, set code to 6. Call UCOLNO to print column numbers at the top of the page.

8. Loop through each row and print using the selected format.

9. RETURN.

10. Error reading control record. Write a message, and STOP.

U2DINT

```
      SUBROUTINE U2DINT(IA,ANAME,II,JJ,K,IN,IOUT)
C
C
C-----VERSION 0801 01NOV1995 U2DINT
C     ****************************************************************
C     ROUTINE TO INPUT 2-D INTEGER DATA MATRICES
C       IA IS ARRAY TO INPUT
C       ANAME IS 24 CHARACTER DESCRIPTION OF IA
C       II IS NO. OF ROWS
C       JJ IS NO. OF COLS
C       K IS LAYER NO. (USED WITH NAME TO TITLE PRINTOUT --
C              IF K=0, NO LAYER IS PRINTED
C              IF K<0, CROSS SECTION IS PRINTED)
C       IN IS INPUT UNIT
C       IOUT IS OUTPUT UNIT
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      CHARACTER*24 ANAME
      DIMENSION IA(JJ,II)
      CHARACTER*20 FMTIN
      CHARACTER*80 CNTRL
      CHARACTER*80 FNAME
      DATA NUNOPN/99/
C     ----------------------------------------------------------------
C
C1------READ ARRAY CONTROL RECORD AS CHARACTER DATA.
      READ(IN,'(A)') CNTRL
C
C2------LOOK FOR ALPHABETIC WORD THAT INDICATES THAT THE RECORD IS FREE
C2------FORMAT.  SET A FLAG SPECIFYING IF FREE FORMAT OR FIXED FORMAT.
      ICLOSE=0
      IFREE=1
      ICOL=1
      CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,1,N,R,IOUT,IN)
      IF (CNTRL(ISTART:ISTOP).EQ.'CONSTANT') THEN
         LOCAT=0
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'INTERNAL') THEN
         LOCAT=IN
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'EXTERNAL') THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,2,LOCAT,R,IOUT,IN)
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'OPEN/CLOSE') THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,0,N,R,IOUT,IN)
         FNAME=CNTRL(ISTART:ISTOP)
         LOCAT=NUNOPN
         WRITE(IOUT,15) LOCAT,FNAME
   15    FORMAT(1X,/1X,'OPENING FILE ON UNIT',I4,':',/1X,A)
         ICLOSE=1
      ELSE
C
C2A-----DID NOT FIND A RECOGNIZED WORD, SO NOT USING FREE FORMAT.
C2A-----READ THE CONTROL RECORD THE ORIGINAL WAY.
         IFREE=0
         READ(CNTRL,1,ERR=600) LOCAT,ICONST,FMTIN,IPRN
    1    FORMAT(I10,I10,A20,I10)
      END IF
C
C3------FOR FREE FORMAT CONTROL RECORD, READ REMAINING FIELDS.
      IF(IFREE.NE.0) THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,2,ICONST,R,IOUT,IN)
         IF(LOCAT.NE.0) THEN
            CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,1,N,R,IOUT,IN)
            FMTIN=CNTRL(ISTART:ISTOP)
            IF(ICLOSE.NE.0) THEN
               IF(FMTIN.EQ.'(BINARY)') THEN
                  OPEN(UNIT=LOCAT,FILE=FNAME,FORM='UNFORMATTED')
               ELSE
                  OPEN(UNIT=LOCAT,FILE=FNAME)
               END IF
            END IF
            IF(LOCAT.GT.0 .AND. FMTIN.EQ.'BINARY') LOCAT=-LOCAT
            CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,2,IPRN,R,IOUT,IN)
         END IF
      END IF
C
```

```
C4------TEST LOCAT TO SEE HOW TO DEFINE ARRAY VALUES.
      IF(LOCAT) 200,50,90
C
C4A-----LOCAT=0; SET ALL ARRAY VALUES EQUAL TO ICONST. RETURN.
   50 DO 80 I=1,II
      DO 80 J=1,JJ
   80 IA(J,I)=ICONST
      IF(K.GT.0) WRITE(IOUT,82) ANAME,ICONST,K
   82 FORMAT(1X,/1X,A,' =',I15,' FOR LAYER',I4)
      IF(K.LE.0) WRITE(IOUT,83) ANAME,ICONST
   83 FORMAT(1X,/1X,A,' =',I15)
      RETURN
C
C4B-----LOCAT>0; READ FORMATTED RECORDS USING FORMAT FMTIN.
   90 IF(K.GT.0) THEN
         WRITE(IOUT,94) ANAME,K,LOCAT,FMTIN
   94    FORMAT(1X,///11X,A,' FOR LAYER',I4,/
     1       1X,'READING ON UNIT',I4,' WITH FORMAT: ',A)
      ELSE IF(K.EQ.0) THEN
         WRITE(IOUT,95) ANAME,LOCAT,FMTIN
   95    FORMAT(1X,///11X,A,/
     1       1X,'READING ON UNIT',I4,' WITH FORMAT: ',A)
      ELSE
         WRITE(IOUT,96) ANAME,LOCAT,FMTIN
   96    FORMAT(1X,///11X,A,' FOR CROSS SECTION',/
     1       1X,'READING ON UNIT',I4,' WITH FORMAT: ',A)
      END IF
      DO 100 I=1,II
      IF(FMTIN.EQ.'(FREE)') THEN
         READ(LOCAT,*) (IA(J,I),J=1,JJ)
      ELSE
         READ(LOCAT,FMTIN) (IA(J,I),J=1,JJ)
      END IF
  100 CONTINUE
      GO TO 300
C
C4C-----LOCAT<0; READ UNFORMATTED RECORD CONTAINING ARRAY VALUES.
  200 LOCAT=-LOCAT
      IF(K.GT.0) THEN
         WRITE(IOUT,201) ANAME,K,LOCAT
  201    FORMAT(1X,///11X,A,' FOR LAYER',I4,/
     1    1X,'READING BINARY ON UNIT',I4)
      ELSE IF(K.EQ.0) THEN
         WRITE(IOUT,202) ANAME,LOCAT
  202    FORMAT(1X,///11X,A,/
     1    1X,'READING BINARY ON UNIT',I4)
      ELSE
         WRITE(IOUT,203) ANAME,LOCAT
  203    FORMAT(1X,///11X,A,' FOR CROSS SECTION',/
     1    1X,'READING BINARY ON UNIT',I4)
      END IF
      READ(LOCAT)
      READ(LOCAT) IA
C
C5------IF ICONST NOT ZERO THEN MULTIPLY ARRAY VALUES BY ICONST.
  300 IF(ICLOSE.NE.0) CLOSE(UNIT=LOCAT)
      IF(ICONST.EQ.0) GO TO 320
      DO 310 I=1,II
      DO 310 J=1,JJ
      IA(J,I)=IA(J,I)*ICONST
  310 CONTINUE
C
C6------IF PRINT CODE (IPRN) <0 THEN RETURN.
  320 IF(IPRN.LT.0) RETURN
C
C7------PRINT COLUMN NUMBERS AT TOP OF PAGE.
      IF(IPRN.GT.9 .OR. IPRN.EQ.0) IPRN=6
      GO TO(401,402,403,404,405,406,407,408,409), IPRN
401   CALL UCOLNO(1,JJ,4,60,2,IOUT)
      GO TO 500
402   CALL UCOLNO(1,JJ,4,40,3,IOUT)
      GO TO 500
403   CALL UCOLNO(1,JJ,4,30,4,IOUT)
      GO TO 500
404   CALL UCOLNO(1,JJ,4,25,5,IOUT)
      GO TO 500
405   CALL UCOLNO(1,JJ,4,20,6,IOUT)
      GO TO 500
406   CALL UCOLNO(1,JJ,4,10,12,IOUT)
      GO TO 500
```

```
407   CALL UCOLNO(1,JJ,4,25,3,IOUT)
      GO TO 500
408   CALL UCOLNO(1,JJ,4,15,5,IOUT)
      GO TO 500
409   CALL UCOLNO(1,JJ,4,10,7,IOUT)
C
C8------PRINT EACH ROW IN THE ARRAY.
500   DO 510 I=1,II
      GO TO(501,502,503,504,505,506,507,508,509), IPRN
C
C----------------FORMAT 60I1
  501 WRITE(IOUT,551) I,(IA(J,I),J=1,JJ)
  551 FORMAT(1X,I3,1X,60(1X,I1):/(5X,60(1X,I1)))
      GO TO 510
C
C----------------FORMAT 40I2
  502 WRITE(IOUT,552) I,(IA(J,I),J=1,JJ)
  552 FORMAT(1X,I3,1X,40(1X,I2):/(5X,40(1X,I2)))
      GO TO 510
C
C----------------FORMAT 30I3
  503 WRITE(IOUT,553) I,(IA(J,I),J=1,JJ)
  553 FORMAT(1X,I3,1X,30(1X,I3):/(5X,30(1X,I3)))
      GO TO 510
C
C----------------FORMAT 25I4
  504 WRITE(IOUT,554) I,(IA(J,I),J=1,JJ)
  554 FORMAT(1X,I3,1X,25(1X,I4):/(5X,25(1X,I4)))
      GO TO 510
C
C----------------FORMAT 20I5
  505 WRITE(IOUT,555) I,(IA(J,I),J=1,JJ)
  555 FORMAT(1X,I3,1X,20(1X,I5):/(5X,20(1X,I5)))
      GO TO 510
C
C----------------FORMAT 10I11
  506 WRITE(IOUT,556) I,(IA(J,I),J=1,JJ)
  556 FORMAT(1X,I3,1X,10(1X,I11):/(5X,10(1X,I11)))
      GO TO 510
C
C----------------FORMAT 25I2
  507 WRITE(IOUT,557) I,(IA(J,I),J=1,JJ)
  557 FORMAT(1X,I3,1X,25(1X,I2):/(5X,25(1X,I2)))
      GO TO 510
C
C----------------FORMAT 15I4
  508 WRITE(IOUT,558) I,(IA(J,I),J=1,JJ)
  558 FORMAT(1X,I3,1X,15(1X,I4):/(5X,10(1X,I4)))
      GO TO 510
C
C----------------FORMAT 10I6
  509 WRITE(IOUT,559) I,(IA(J,I),J=1,JJ)
  559 FORMAT(1X,I3,1X,10(1X,I6):/(5X,10(1X,I6)))
C
  510 CONTINUE
C
C9------RETURN
      RETURN
C
C10-----CONTROL RECORD ERROR.
  600 IF(K.GT.0) THEN
         WRITE(IOUT,601) ANAME,K
  601    FORMAT(1X,/1X,'ERROR READING ARRAY CONTROL RECORD FOR ',A,
     1     ' FOR LAYER',I4,':')
      ELSE
         WRITE(IOUT,602) ANAME
  602    FORMAT(1X,/1X,'ERROR READING ARRAY CONTROL RECORD FOR ',A,':')
      END IF
      WRITE(IOUT,'(1X,A)') CNTRL
      STOP
      END
```

List of Variables for Module U2DINT

Variable	Range	Definition
ANAME	Argument	CHARACTER*24, Label that identifies array IA.
CNTRL	Module	CHARACTER*80, The control record that has been read as text so that it can be parsed for words by Module URWORD.
FMTIN	Module	CHARACTER*20, Format that will be used to read array IA.
FNAME	Module	CHARACTER*80, Name of file when U2DINT opens the file from which array IA will be read.
I	Module	Index for rows.
IA	Argument	DIMENSION (JJ,II), Array to be defined.
ICLOSE	Module	Flag used to indicate if the file from which array IA has been read must be closed.
ICOL	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.
ICONST	Module	Constant used in defining array IA. If the array is a constant, ICONST is that constant. If each element is read, the values are multipled by ICONST.
IFREE	Module	Flag that is not 0 when free-format (alphabetic) control record is being used.
II	Argument	Number of rows in array IA.
IN	Argument	The unit number from which the control record is read.
IOUT	Global	Unit number for writing to the listing file.
IPRN	Module	Code that indicates which format to use to print array IA.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
J	Module	Index for columns.
JJ	Argument	Number of columns in array IA.
K	Argument	The layer to which array IA is associated. If K is 0, there is no layer; if K < 0, array IA is a cross section.
LOCAT	Module	Code that indicates how the elements in array IA are defined: > 0, formatted values are read using LOCAT as the unit. = 0, all elements are set equal to ICONST. < 0, unformatted values are read using -LOCAT as the unit.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NUNOPN	Module	Unit used when U2DINT opens the file from which array IA is read.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.

Module U1DREL

Narrative for Module U1DREL

Module U1DREL defines values for a one-dimensional real array. It first reads an "array control record"; then based upon the contents of the control record, it defines the values for all elements in the array. The elements are either all set to the same value, or they are individually read from a file. If array elements are read from a file, the file may be formatted or unformatted. Variable LOCAT is a numeric code that controls all of these options. LOCAT is either directly read from the control record, or it is defined by an alphabetic word in the control record. If one of the allowed words is not found as the first field of the record, then a numeric code is assumed. U1DREL performs its functions as follows:

1. Read the array control record as character data (CHARACTER variable CNTRL).

2. Use URWORD to look for an alphabetic word -- either "CONSTANT', "INTERNAL", or EXTERNAL". If found, set IFREE flag to 1 and set LOCAT to the appropriate value:

   "CONSTANT" -- all elements of array are the same value, which is CNSTNT; set LOCAT = 0

   "INTERNAL" -- elements of array will be read on same unit as control record: set LOCAT = IN

   "EXTERNAL" -- elements of array are read on the unit specified in the following field in the control record: call URWORD to obtain this value and store it in LOCAT.

   "OPEN / CLOSE" -- elements of array are read from the file whose name is specified in the following field. Call URWORD to obtain this file name, store its name in FNAME, write its name, open it on unit NUNOPN (99), and set flag ICLOSE to 1 to indicate that this file should be closed after reading.

   A. Alphabetic word was not found; set IFREE flag to 0 and read the control record using a fixed format.

3. If free-format control record is being used (IREE not 0), parse the remaining fields. First get the value for CNSTNT, which is always required. If LOCAT is not 0, a format and print code are required.

4. Test LOCAT to determine how to define elements in array.

   A. LOCAT is 0 or less than 0. Set all elements of array equal to CNSTNT. Print the value, and RETURN.

   B. LOCAT > 0. Read formatted records on unit LOCAT. Prior to reading the data, print a heading for the array. If format in FMTIN is "(FREE)", then read the data using free format; otherwise, use FMTIN. Close the file if ICLOSE is not 0.

5. If CNSTNT is not 0, multiply all elements in array by CNSTNT.

6. If the print code, IPRN, is greater than or equal to 0, print the array.

7. RETURN.

8. Error reading control record. Write a message, and STOP.

U1DREL

```
      SUBROUTINE U1DREL(A,ANAME,JJ,IN,IOUT)
C
C
C-----VERSION 1740 18APRIL1993 U1DREL
C     ****************************************************************
C     ROUTINE TO INPUT 1-D REAL DATA MATRICES
C       A IS ARRAY TO INPUT
C       ANAME IS 24 CHARACTER DESCRIPTION OF A
C       JJ IS NO. OF ELEMENTS
C       IN IS INPUT UNIT
C       IOUT IS OUTPUT UNIT
C     ****************************************************************
C
C        SPECIFICATIONS:
C     ----------------------------------------------------------------
      CHARACTER*24 ANAME
      DIMENSION A(JJ)
      CHARACTER*20 FMTIN
      CHARACTER*80 CNTRL
      CHARACTER*80 FNAME
      DATA NUNOPN/99/
C     ----------------------------------------------------------------
C
C1------READ ARRAY CONTROL RECORD AS CHARACTER DATA.
      READ(IN,'(A)') CNTRL
C
C2------LOOK FOR ALPHABETIC WORD THAT INDICATES THAT THE RECORD IS FREE
C2------FORMAT.  SET A FLAG SPECIFYING IF FREE FORMAT OR FIXED FORMAT.
      ICLOSE=0
      IFREE=1
      ICOL=1
      CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,1,N,R,IOUT,IN)
      IF (CNTRL(ISTART:ISTOP).EQ.'CONSTANT') THEN
         LOCAT=0
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'INTERNAL') THEN
         LOCAT=IN
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'EXTERNAL') THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,2,LOCAT,R,IOUT,IN)
      ELSE IF(CNTRL(ISTART:ISTOP).EQ.'OPEN/CLOSE') THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,0,N,R,IOUT,IN)
         FNAME=CNTRL(ISTART:ISTOP)
         LOCAT=NUNOPN
         WRITE(IOUT,15) LOCAT,FNAME
   15    FORMAT(1X,/1X,'OPENING FILE ON UNIT',I4,':',/1X,A)
         OPEN(UNIT=LOCAT,FILE=FNAME)
         ICLOSE=1
      ELSE
C
C2A-----DID NOT FIND A RECOGNIZED WORD, SO NOT USING FREE FORMAT.
C2A-----READ THE CONTROL RECORD THE ORIGINAL WAY.
         IFREE=0
         READ(CNTRL,1,ERR=500) LOCAT,CNSTNT,FMTIN,IPRN
    1    FORMAT(I10,F10.0,A20,I10)
      END IF
C
C3------FOR FREE FORMAT CONTROL RECORD, READ REMAINING FIELDS.
      IF(IFREE.NE.0) THEN
         CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,3,N,CNSTNT,IOUT,IN)
         IF(LOCAT.GT.0) THEN
            CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,1,N,R,IOUT,IN)
            FMTIN=CNTRL(ISTART:ISTOP)
            CALL URWORD(CNTRL,ICOL,ISTART,ISTOP,2,IPRN,R,IOUT,IN)
         END IF
      END IF
C
C4------TEST LOCAT TO SEE HOW TO DEFINE ARRAY VALUES.
      IF(LOCAT.GT.0) GO TO 90
C
C4A-----LOCAT <0 OR =0; SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN.
      DO 80 J=1,JJ
   80 A(J)=CNSTNT
      WRITE(IOUT,3) ANAME,CNSTNT
    3 FORMAT(1X,/1X,A,' =',G15.7)
      RETURN
C
C4B-----LOCAT>0; READ FORMATTED RECORDS USING FORMAT FMTIN.
   90 WRITE(IOUT,5) ANAME,LOCAT,FMTIN
    5 FORMAT(1X,///11X,A,/
     1       1X,'READING ON UNIT',I4,' WITH FORMAT: ',A20)
      IF(FMTIN.EQ.'(FREE)') THEN
         READ(LOCAT,*) (A(J),J=1,JJ)
```

```
      ELSE
         READ(LOCAT,FMTIN) (A(J),J=1,JJ)
      END IF
      IF(ICLOSE.NE.0) CLOSE(UNIT=LOCAT)
C
C5------IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.
      ZERO=0.
      IF(CNSTNT.EQ.ZERO) GO TO 120
      DO 100 J=1,JJ
  100 A(J)=A(J)*CNSTNT
C
C6------IF PRINT CODE (IPRN) =0 OR >0 THEN PRINT ARRAY VALUES.
120   IF(IPRN.EQ.0) THEN
         WRITE(IOUT,1001) (A(J),J=1,JJ)
1001     FORMAT((1X,1PG12.5,9(1X,G12.5)))
      ELSE IF(IPRN.GT.0) THEN
         WRITE(IOUT,1002) (A(J),J=1,JJ)
1002     FORMAT((1X,1PG12.5,4(1X,G12.5)))
      END IF
C
C7------RETURN
      RETURN
C
C8------CONTROL RECORD ERROR.
500   WRITE(IOUT,502) ANAME
502   FORMAT(1X,/1X,'ERROR READING ARRAY CONTROL RECORD FOR ',A,':')
      WRITE(IOUT,'(1X,A)') CNTRL
      STOP
      END
```

List of Variables for Module U1DREL

Variable	Range	Definition
A	Argument	DIMENSION (JJ,II), Array to be defined.
ANAME	Argument	CHARACTER*24, Label that identifies array A.
CNSTNT	Module	Constant used in defining array A. If the array is a constant, CNSTNT is that constant. If each element is read, the values are multipled by CNSTNT.
CNTRL	Module	CHARACTER*80, The control record that has been read as text so that it can be parsed for words by Module URWORD.
FMTIN	Module	CHARACTER*20, Format that will be used to read array A.
FNAME	Module	CHARACTER*80, Name of file when U1DREL opens the file from which array A will be read.
ICLOSE	Module	Flag used to indicate if the file from which array A has been read must be closed.
ICOL	Module	Index pointing to the location within a character string at which Module URWORD begins looking for a word.
IFREE	Module	Flag that is not 0 when free-format (alphabetic) control record is being used.
IN	Argument	The unit number from which the control record is read.
IOUT	Global	Unit number for writing to the listing file.
IPRN	Module	Code that indicates which format to use to print array A.
ISTART	Module	Index pointing to the start of a word found by Module URWORD.
ISTOP	Module	Index pointing to the end of a word found by Module URWORD.
J	Module	Index for elements in array A.
JJ	Argument	Number of elements in array A.
LOCAT	Module	Code that indicates how the elements in array A are defined: > 0, formatted values are read using LOCAT as the unit. = 0 or < 0, all elements are set equal to CNSTNT.
N	Module	Argument place holder for calls to URWORD in which the argument is unused.
NUNOPN	Module	Unit used when U1DREL opens the file from which array A is read.
R	Module	Argument place holder for calls to URWORD in which the argument is unused.
ZERO	Module	The constant 0.

Module URWORD

Narrative for Module URWORD

Module URWORD extracts a word from a character string. Words are separated by a comma or one or more spaces. Words that have spaces or commas in them can be enclosed in single quotes. In order to allow successive calls to URWORD to get a series of words, URWORD returns a pointer to the character in the string from which scanning should continue. The word itself is not returned as an argument, but rather pointers to the first and last characters in the word are returned. Also, the word can be converted to an integer or real number as an option. There is no argument that indicates an error; if there is an error, the returned word will be a single blank character. The caller is always guaranteed to get a valid string returned. URWORD performs its functions as follows:

1. Start by setting the returned word to be a single blank character. This is done by setting the last character of the string to a blank, so a user should include one extra character in the string for this purpose. If the starting location within the string is outside of the string, go to step 6.

2. Loop through the string from the starting pointer to the end looking for the first character that is not a blank or a comma, which indicates the start of a word. If the start is found, go to step 3. If the end of the string is reached without finding a starting character, then set the location pointer to the blank character at the end of the string and go to step 6.

3. A starting character has been found; now look for an ending character:

   A. If the starting character is a quote, then loop through characters after the quote looking for a second quote. If found, go to step 4.

   B. If the starting character is not a quote, then loop through characters after the starting character for a space or comma. If found, go to step 4.

   C. The end of the string has been reached without finding the end of the word. Set the pointer to the ending character equal to the location of the blank character at the end of the string, and continue to step 4.

4. A word has been found, and J points to the ending character. Set the starting location for scanning another word (ICOL) to J+1, and the set J to point to the end of the word. If the end of word pointer is less than the start of word pointer, which can happen if there are two quotes together, then go to step 6. Otherwise, set the return values for start of word and end of word pointers: ISTART and ISTOP.

5. If NCODE is 1, convert the word to upper case and RETURN.

6. Convert the word to a number if NCODE is 2 or 3. Right justify the word in the 20-character variable RW. If more than 20 characters, then go to step 7. Convert RW to an integer if NCODE is 2 or to a real if NCODE is 3. RETURN.

7. Number conversion error. Set STRING equal to the kind of number being converted -- real or integer. Then act according to output unit (IOUT):

A. If output unit is negative, set last character of string equal to 'E' and RETURN.

B. If output unit is positive, write a message to that output unit. If the input unit is positive, the message will include the input unit. If the input is not positive, the message will refer to keyboard input.

C. If the output unit is 0, write a message to the default output.

D. STOP after writing message.

URWORD

```
      SUBROUTINE URWORD(LINE,ICOL,ISTART,ISTOP,NCODE,N,R,IOUT,IN)
C
C
C-----VERSION 1003 05AUG1992 URWORD
C     ******************************************************************
C     ROUTINE TO EXTRACT A WORD FROM A LINE OF TEXT, AND OPTIONALLY
C     CONVERT THE WORD TO A NUMBER.
C        ISTART AND ISTOP WILL BE RETURNED WITH THE STARTING AND
C          ENDING CHARACTER POSITIONS OF THE WORD.
C        THE LAST CHARACTER IN THE LINE IS SET TO BLANK SO THAT IF ANY
C          PROBLEMS OCCUR WITH FINDING A WORD, ISTART AND ISTOP WILL
C          POINT TO THIS BLANK CHARACTER.  THUS, A WORD WILL ALWAYS BE
C          RETURNED UNLESS THERE IS A NUMERIC CONVERSION ERROR.  BE SURE
C          THAT THE LAST CHARACTER IN LINE IS NOT AN IMPORTANT CHARACTER
C          BECAUSE IT WILL ALWAYS BE SET TO BLANK.
C        A WORD STARTS WITH THE FIRST CHARACTER THAT IS NOT A SPACE OR
C          COMMA, AND ENDS WHEN A SUBSEQUENT CHARACTER THAT IS A SPACE
C          OR COMMA.  NOTE THAT THESE PARSING RULES DO NOT TREAT TWO
C          COMMAS SEPARATED BY ONE OR MORE SPACES AS A NULL WORD.
C        FOR A WORD THAT BEGINS WITH "'", THE WORD STARTS WITH THE
C          CHARACTER AFTER THE QUOTE AND ENDS WITH THE CHARACTER
C          PRECEDING A SUBSEQUENT QUOTE.  THUS, A QUOTED WORD CAN
C          INCLUDE SPACES AND COMMAS.  THE QUOTED WORD CANNOT CONTAIN
C          A QUOTE CHARACTER.
C        IF NCODE IS 1, THE WORD IS CONVERTED TO UPPER CASE.
C        IF NCODE IS 2, THE WORD IS CONVERTED TO AN INTEGER.
C        IF NCODE IS 3, THE WORD IS CONVERTED TO A REAL NUMBER.
C        NUMBER CONVERSION ERROR IS WRITTEN TO UNIT IOUT IF IOUT IS
C          POSITIVE; ERROR IS WRITTEN TO DEFAULT OUTPUT IF IOUT IS 0;
C          NO ERROR MESSAGE IS WRITTEN IF IOUT IS NEGATIVE.
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*(*) LINE
      CHARACTER*20 RW,STRING
C     ------------------------------------------------------------------
C
C1------Set last char in LINE to blank and set ISTART and ISTOP to point
C1------to this blank as a default situation when no word is found.  If
C1------starting location in LINE is out of bounds, do not look for a
C1------word.
      LINLEN=LEN(LINE)
      LINE(LINLEN:LINLEN)=' '
      ISTART=LINLEN
      ISTOP=LINLEN
      LINLEN=LINLEN-1
      IF(ICOL.LT.1 .OR. ICOL.GT.LINLEN) GO TO 100
C
C2------Find start of word, which is indicated by first character that
C2------is not a blank and not a comma.
      DO 10 I=ICOL,LINLEN
      IF(LINE(I:I).NE.' ' .AND. LINE(I:I).NE.',') GO TO 20
10    CONTINUE
      ICOL=LINLEN+1
      GO TO 100
C
C3------Found start of word.  Look for end.
C3A-----When word is quoted, only a quote can terminate it.
20    IF(LINE(I:I).EQ.'''') THEN
         I=I+1
         IF(I.LE.LINLEN) THEN
            DO 25 J=I,LINLEN
            IF(LINE(J:J).EQ.'''') GO TO 40
25          CONTINUE
         END IF
C
C3B-----When word is not quoted, space or comma will terminate.
      ELSE
         DO 30 J=I,LINLEN
         IF(LINE(J:J).EQ.' ' .OR. LINE(J:J).EQ.',') GO TO 40
30       CONTINUE
      END IF
C
C3C-----End of line without finding end of word; set end of word to
C3C-----end of line.
      J=LINLEN+1
C
C4------Found end of word; set J to point to last character in WORD and
```

```
C-------set ICOL to point to location for scanning for another word.
40    ICOL=J+1
      J=J-1
      IF(J.LT.I) GO TO 100
      ISTART=I
      ISTOP=J
C
C5------Convert word to upper case and RETURN if NCODE is 1.
      IF(NCODE.EQ.1) THEN
         IDIFF=ICHAR('a')-ICHAR('A')
         DO 50 K=ISTART,ISTOP
            IF(LINE(K:K).GE.'a' .AND. LINE(K:K).LE.'z')
     1           LINE(K:K)=CHAR(ICHAR(LINE(K:K))-IDIFF)
50       CONTINUE
         RETURN
      END IF
C
C6------Convert word to a number if requested.
100   IF(NCODE.EQ.2 .OR. NCODE.EQ.3) THEN
         RW=' '
         L=20-ISTOP+ISTART
         IF(L.LT.1) GO TO 200
         RW(L:20)=LINE(ISTART:ISTOP)
         IF(NCODE.EQ.2) READ(RW,'(I20)',ERR=200) N
         IF(NCODE.EQ.3) READ(RW,'(F20.0)',ERR=200) R
      END IF
      RETURN
C
C7------Number conversion error.
200   IF(NCODE.EQ.3) THEN
         STRING= 'A REAL NUMBER'
         L=13
      ELSE
         STRING= 'AN INTEGER'
         L=10
      END IF
C
C7A-----If output unit is negative, set last character of string to 'E'.
      IF(IOUT.LT.0) THEN
         N=0
         R=0.
         LINE(LINLEN+1:LINLEN+1)='E'
         RETURN
C
C7B-----If output unit is positive; write a message to output unit.
      ELSE IF(IOUT.GT.0) THEN
         IF(IN.GT.0) THEN
            WRITE(IOUT,201) IN,LINE(ISTART:ISTOP),STRING(1:L),LINE
         ELSE
            WRITE(IOUT,202) LINE(ISTART:ISTOP),STRING(1:L),LINE
         END IF
201      FORMAT(1X,/1X,'FILE UNIT',I4,' : ERROR CONVERTING "',A,
     1       '" TO ',A,' IN LINE:',/1X,A)
202      FORMAT(1X,/1X,'KEYBOARD INPUT : ERROR CONVERTING "',A,
     1       '" TO ',A,' IN LINE:',/1X,A)
C
C7C-----If output unit is 0; write a message to default output.
      ELSE
         IF(IN.GT.0) THEN
            WRITE(*,201) IN,LINE(ISTART:ISTOP),STRING(1:L),LINE
         ELSE
            WRITE(*,202) LINE(ISTART:ISTOP),STRING(1:L),LINE
         END IF
      END IF
C
C7D-----STOP after writing message.
      STOP
      END
```

List of Variables for Module URWORD

Variable	Range	Definition
I	Module	Index that is used to find the start of a word.
ICOL	Argument	Location within LINE to start looking for a word.
IDIFF	Module	Difference between a lowercase character and an uppercase character.
IN	Argument	Unit from which LINE was read.
IOUT	Global	Unit number for writing to the listing file.
ISTART	Argument	Returned location for the start of a word.
ISTOP	Argument	Returned location for the end of a word.
J	Module	Index that is used to find the end of a word.
K	Module	Index for stepping through the characters in a word.
L	Module	Index in RW, and the number of characters that have been defined in STRING.
LINE	Argument	CHARACTER*(*), Line of text that is being parsed for a word.
LINLEN	Module	At the beginning of URWORD, LINLEN is the length of LINE. LINLEN is then decremented by 1.
N	Argument	When NCODE is 2, N is the converted integer.
NCODE	Argument	Code for what to do when a word is found: Not 1, 2, or 3 - leave word exactly as found. 1 - convert word to uppercase. 2 - convert word to an integer. 3 - convert word to a real number.
R	Argument	When NCODE is 3, R is the converted real number.
RW	Module	CHARACTER*20, Right justified word that will be converted to a number.
STRING	Module	CHARACTER*20, When a number conversion error occurs, STRING contains a description of the kind of number that was being converted, which is printed in an error message.

Module UBDSV1

Narrative for Module UBDSV1

Module UBDSV1 is one of several routines for writing cell-by-cell budget data to disk. This particular routine writes one value for each model cell much as UBUDSV does. The difference is that UBDSV1 writes an additional record containing time step length (DELT), stress period time (PERTIM), and total simulation time (TOTIM); these time parameters may be needed by other programs along with the budget data. Also, note that UBDSV1 receives IBOUND as a calling argument even though IBOUND is unused by UBDSV1. The purpose for doing this is to facilitate the substitution of a replacement module that might make use of IBOUND. UBDSV1 performs its functions as follows:

1. Write two unformatted records identifying the budget data that will follow. The first record is identical to the first record written by UBUDSV, except that the layer number is negated. This can be used by a program that reads these data in order to detect how to read the remaining data. The second record contains a code that defines whether UBDSV1, UBDSV2 combined with UBDSVA, or UBDSV3 is being used to save the budget data. A value of 1 indicates that UBDSV1 is used. The 2nd record also contains DELT, PERTIM, and TOTIM.

2. Write a record containing one budget value for each cell in the model grid.

3. RETURN.

UBDSV1

```
      SUBROUTINE UBDSV1(KSTP,KPER,TEXT,IBDCHN,BUFF,NCOL,NROW,NLAY,IOUT,
     1          DELT,PERTIM,TOTIM,IBOUND)
C-----VERSION 1002 18DEC1992 UBDSV1
C     ***************************************************************
C     RECORD CELL-BY-CELL FLOW TERMS FOR ONE COMPONENT OF FLOW AS A 3-D
C     ARRAY WITH EXTRA RECORD TO INDICATE DELT, PERTIM, AND TOTIM.
C     ***************************************************************
C
C       SPECIFICATIONS:
C     -----------------------------------------------------------------
      CHARACTER*16 TEXT
      DIMENSION BUFF(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY)
C     -----------------------------------------------------------------
C
C1------WRITE TWO UNFORMATTED RECORDS IDENTIFYING DATA.
      IF(IOUT.GT.0) WRITE(IOUT,1) TEXT,IBDCHN,KSTP,KPER
    1 FORMAT(1X,'UBDSV1 SAVING "',A16,'" ON UNIT',I4,
     1     ' AT TIME STEP',I3,', STRESS PERIOD',I3)
      WRITE(IBDCHN) KSTP,KPER,TEXT,NCOL,NROW,-NLAY
      WRITE(IBDCHN) 1,DELT,PERTIM,TOTIM
C
C2------WRITE AN UNFORMATTED RECORD CONTAINING VALUES FOR
C2------EACH CELL IN THE GRID.
      WRITE(IBDCHN) BUFF
C
C3------RETURN
      RETURN
      END
```

List of Variables for Module UBDSV1

Variable	Range	Definition
BUFF	Argument	DIMENSION (NCOL,NROW,NLAY), Budget data to be written.
DELT	Global	Length of the current time step.
IBDCHN	Argument	Unit number to which output is written.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell IBOUND is unused in this module, but it is being made available in case there is a need to enhance this module.
IOUT	Global	Unit number for writing to the listing file.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
TEXT	Argument	CHARACTER*16, Label that identifies the budget term.
TOTIM	Global	Elapsed time in the simulation.

Module UBDSV2

Narrative for Module UBDSV2

Module UBDSV2 is one of several routines for writing cell-by-cell budget data to disk. This particular routine is part of a pair of routines that writes a header record for a type of stress (for example wells, rivers, or drains) followed by one record for each individual stress of that type. For example, in the Well Package, the user specifies a list of wells. UBDSV2 is called once to write header information for wells, and UBDSVA is called once for each well to write the flow rate.

Like UBDSV1 and UBDSV3, UBDSV2 writes a record containing time step length (DELT), stress period time (PERTIM), and total simulation time (TOTIM); these time parameters may be needed by other programs along with the budget data. Also, note that UBDSV2 receives IBOUND as a calling argument even though IBOUND is unused by UBDSV2. The purpose for doing this is to facilitate the substitution of a replacement module that might make use of IBOUND. UBDSV2 performs its functions as follows:

1. Write three unformatted records identifying the budget data that will follow. The first record is identical to the first record written by UBUDSV, except that the layer number is negated. This can be used by a program that reads these data in order to detect how to read the remaining data. The second record contains a method code that defines whether UBDSV1, UBDSV2 combined with UBDSVA, or UBDSV3 is being used to save the budget data. A method code of 2 indicates that UBDSV2 with UBDSVA is used. The 2nd record also contains DELT, PERTIM, and TOTIM. The third record contains the number of records that will be written by UBDSVA -- 1 for each individual stress.

2. RETURN.

UBDSV2

```
      SUBROUTINE UBDSV2(KSTP,KPER,TEXT,IBDCHN,NCOL,NROW,NLAY,
     1          NLIST,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C-----VERSION 0805 18DEC1992 UBDSV2
C     ******************************************************************
C     WRITE HEADER RECORDS FOR CELL-BY-CELL FLOW TERMS FOR ONE COMPONENT
C     OF FLOW USING A LIST STRUCTURE.  EACH ITEM IN THE LIST IS WRITTEN
C     BY MODULE UBDSVA
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 TEXT
      DIMENSION IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------WRITE THREE UNFORMATTED RECORDS IDENTIFYING DATA.
      IF(IOUT.GT.0) WRITE(IOUT,1) TEXT,IBDCHN,KSTP,KPER
    1 FORMAT(1X,'UBDSV2 SAVING "',A16,'" ON UNIT',I4,
     1     ' AT TIME STEP',I3,', STRESS PERIOD',I3)
      WRITE(IBDCHN) KSTP,KPER,TEXT,NCOL,NROW,-NLAY
      WRITE(IBDCHN) 2,DELT,PERTIM,TOTIM
      WRITE(IBDCHN) NLIST
C
C2------RETURN
      RETURN
      END
```

List of Variables for Module UBDSV2

Variable	Range	Definition
DELT	Global	Length of the current time step.
IBDCHN	Argument	Unit number to which output is written.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell IBOUND is unused in this module, but it is being made available in case there is a need to enhance this module.
IOUT	Global	Unit number for writing to the listing file.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NLIST	Argument	The number of budget values that will be written with 1 value per record.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
TEXT	Argument	CHARACTER*16, Label that identifies the budget term.
TOTIM	Global	Elapsed time in the simulation.

Module UBDSVA

Narrative for Module UBDSVA

Module UBDSVA is one of several routines for writing cell-by-cell budget data to disk. This particular routine is part of a pair of routines that writes a header record for a type of stress (for example wells, rivers, or drains) followed by one record for each individual stress of that type. For example, in the Well Package, the user specifies a list of wells. UBDSV2 is called once to write header information for wells, and UBDSVA is called once for each well to write the flow rate. UBDSVA performs its functions as follows:

1. Calculate the cell number for the stress. The cell number is the sequential (1-dimensional) number corresponding to the layer, row, and column that contains the stress. By using a 1-dimensional value, only a single value is required to indicate the location rather than three values.

2. Write a record containing the cell number and the flow rate.

3. RETURN.

UBDSVA

```
      SUBROUTINE UBDSVA(IBDCHN,NCOL,NROW,J,I,K,Q,IBOUND,NLAY)
C-----VERSION 0809 18DEC1992 UBDSVA
C     ******************************************************************
C     WRITE ONE VALUE OF CELL-BY-CELL FLOW USING A LIST STRUCTURE.
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------CALCULATE CELL NUMBER
      ICRL= (K-1)*NROW*NCOL + (I-1)*NCOL + J
C
C2------WRITE CELL NUMBER AND FLOW RATE
      WRITE(IBDCHN) ICRL,Q
C
C3------RETURN
      RETURN
      END
```

List of Variables for Module UBDSVA

Variable	Range	Definition
I	Argument	Row of cell for which budget data will be written.
IBDCHN	Argument	Unit number to which output is written.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell IBOUND is unused in this module, but it is being made available in case there is a need to enhance this module.
ICRL	Module	One-dimensional index for cell (J,I,K).
J	Argument	Column of cell for which budget data will be written.
K	Argument	Layer of cell for which budget data will be written.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NROW	Global	The number of rows in the grid.
Q	Argument	The budget value to be written for cell (J,I,K).

Module UBDSV3

Narrative for Module UBDSV3

Module UBDSV3 is one of several routines for writing cell-by-cell budget data to disk. This particular routine writes one value for each of NCOL*NROW cells. That is, it writes only one layer's worth of data as might be needed for either the Recharge or Evapotranspiration Packages. If any of the budget values correspond to a layer other than layer 1, IBDSV3 will write an integer array of layer numbers that correspond to each of the budget values. Like UBDSV1 and UBDSV2, UBDSV3 writes a record containing time step length (DELT), stress period time (PERTIM), and total simulation time (TOTIM); these time parameters may be needed by other programs along with the budget data. Also, note that UBDSV3 receives IBOUND as a calling argument even though IBOUND is unused by UBDSV3. The purpose for doing this is to facilitate the substitution of a replacement module that might make use of IBOUND. UBDSV3 performs its functions as follows:

1. Write two unformatted records identifying the budget data that will follow. The first record is identical to the first record written by UBUDSV, except that the layer number is negated. This can be used by a program that reads these data in order to detect how to read the remaining data. The second record contains a method code that defines whether UBDSV1, UBDSV2 combined with UBDSVA, or UBDSV3 is being used to save the budget data. A method code of 3 or 4 indicates that UBDSV3 is used. Three indicates the budget values are all for layer 1, and no array of layer numbers will be written. Four indicates that an array of layer numbers corresponding to each budget value will be written. The 2nd record also contains DELT, PERTIM, and TOTIM.

2. Write either one or two records depending on NOPT.

    A. If NOPT is 1, then all budget values are for layer 1. Write a record containing the budget values for layer 1

    B. If NOPT is not 1, then budget values are for different layers. Write one record conatining the layer numbers for the budget values and a second record containing the budget values -- one value for NCOL*NROW cells.

3. RETURN.

UBDSV3

```
      SUBROUTINE UBDSV3(KSTP,KPER,TEXT,IBDCHN,BUFF,IBUFF,NOPT,
     1              NCOL,NROW,NLAY,IOUT,DELT,PERTIM,TOTIM,IBOUND)
C
C
C-----VERSION 1609 18DEC1992 UBDSV3
C     ******************************************************************
C     RECORD CELL-BY-CELL FLOW TERMS FOR ONE COMPONENT OF FLOW AS A 2-D
C     ARRAY OF FLOW VALUES AND OPTIONALLY A 2-D ARRAY OF LAYER NUMBERS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      CHARACTER*16 TEXT
      DIMENSION BUFF(NCOL,NROW,NLAY),IBUFF(NCOL,NROW),
     1          IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------WRITE TWO UNFORMATTED RECORDS IDENTIFYING DATA.
      IF(IOUT.GT.0) WRITE(IOUT,1) TEXT,IBDCHN,KSTP,KPER
    1 FORMAT(1X,'UBDSV3 SAVING "',A16,'" ON UNIT',I4,
     1     ' AT TIME STEP',I3,', STRESS PERIOD',I3)
      WRITE(IBDCHN) KSTP,KPER,TEXT,NCOL,NROW,-NLAY
      IMETH=3
      IF(NOPT.EQ.1) IMETH=4
      WRITE(IBDCHN) IMETH,DELT,PERTIM,TOTIM
C
C2------WRITE DATA AS ONE OR TWO UNFORMATTED RECORDS CONTAINING ONE
C2------VALUE PER LAYER.
      IF(NOPT.EQ.1) THEN
C2A-----WRITE ONE RECORD WHEN NOPT IS 1.  THE VALUES ARE FLOW VALUES
C2A-----FOR LAYER 1.
         WRITE(IBDCHN) ((BUFF(J,I,1),J=1,NCOL),I=1,NROW)
      ELSE
C2B-----WRITE TWO RECORDS WHEN NOPT IS NOT 1.  FIRST RECORD CONTAINS
C2B-----LAYER NUMBERS;  SECOND RECORD CONTAINS FLOW VALUES.
         WRITE(IBDCHN) ((IBUFF(J,I),J=1,NCOL),I=1,NROW)
         WRITE(IBDCHN) ((BUFF(J,I,IBUFF(J,I)),J=1,NCOL),I=1,NROW)
      END IF
C
C3------RETURN
      RETURN
      END
```

List of Variables for Module UBDSV3

Variable	Range	Definition
BUFF	Argument	DIMENSION (NCOL,NROW,NLAY), Budget data to be written.
DELT	Global	Length of the current time step.
I	Module	Index for rows.
IBDCHN	Argument	Unit number to which output is written.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell IBOUND is unused in this module, but it is being made available in case there is a need to enhance this module.
IBUFF	Argument	DIMENSION (NCOL,NROW), Array of layer numbers corresponding to the budget values in BUFF.
IMETH	Module	Code that is written in the budget file to indicate how budget values are written: 3 - BUFF and IBUFF are written. 4 - Only BUFF for layer 1 is written.
IOUT	Global	Unit number for writing to the listing file.
J	Module	Index for columns.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
NCOL	Global	The number of columns in the grid.
NLAY	Global	The number of layers in the grid.
NOPT	Argument	Code for how budget values are to be written: ≠ 1, BUFF and IBUFF are written. = 1, Only BUFF for layer 1 is written.
NROW	Global	The number of rows in the grid.
PERTIM	Global	Elapsed time during the current stress period.
TEXT	Argument	CHARACTER*16, Label that identifies the budget term.
TOTIM	Global	Elapsed time in the simulation.

Module ULASV2

Narrative for Module ULASV2

Module ULASV2 writes a 1-layer array using formatted output. ULASV2 performs its functions as follows:

1. Write a label identifying the array if LBLSAV is not 0. The identification includes the time step (KSTP), stress period (KPER), stress period time (PERTIM), total simulation time (TOTIM), array name (TEXT), number of columns (NCOL), number of rows (NROW), the layer number (ILAY), and the format (FMTOUT).

2. Write the data using the specified format.

3. RETURN.

ULASV2

```
      SUBROUTINE ULASV2(BUFF,TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                  NROW,ILAY,ICHN,FMTOUT,LBLSAV,IBOUND)
C
C-----VERSION 0929 27NOV1992 ULASV2
C     ***************************************************************
C     SAVE 1 LAYER ARRAY ON DISK USING FORMATTED OUTPUT
C     ***************************************************************
C
C        SPECIFICATIONS:
C     -------------------------------------------------------------
      CHARACTER*16 TEXT
      DIMENSION BUFF(NCOL,NROW),IBOUND(NCOL,NROW)
      CHARACTER*20 FMTOUT
C     -------------------------------------------------------------
C
C1------WRITE A LABEL IF LBLSAV IS NOT 0.
      IF(LBLSAV.NE.0) WRITE(ICHN,5) KSTP,KPER,PERTIM,TOTIM,TEXT,NCOL,
     1                  NROW,ILAY,FMTOUT
5     FORMAT(1X,2I5,1P,2E15.6,1X,A,3I6,1X,A)
C
C2------WRITE THE ARRAY USING THE SPECIFIED FORMAT.
      DO 10 IR=1,NROW
      WRITE(ICHN,FMTOUT) (BUFF(IC,IR),IC=1,NCOL)
10    CONTINUE
C
C3------RETURN
      RETURN
      END
```

List of Variables for Module ULASV2

Variable	Range	Definition
BUFF	Argument	DIMENSION (NCOL,NROW), Data to be written.
FMTOUT	Argument	CHARACTER*20, Format for writing BUFF.
IBOUND	Global	DIMENSION (NCOL,NROW), Status of cells in the grid: < 0, constant-head cell = 0, no-flow (inactive) cell > 0, variable-head cell IBOUND is unused in this module, but it is made available in case there is a need to enhance this module.
IC	Module	Index for columns.
ICHN	Argument	Unit number to which output is written.
ILAY	Argument	Layer number corresponding to BUFF.
IR	Module	Index for rows.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. KSTP is reset to 1 at the start of each stress period.
LBLSAV	Argument	Label flag: = 0, do not write a label identifying the data. ≠ 0, write a label identifying the data.
NCOL	Argument	The number of columns in BUFF.
NROW	Argument	The number of rows in BUFF.
PERTIM	Global	Elapsed time during the current stress period.
TEXT	Argument	CHARACTER*16, Label that identifies the data in BUFF.
TOTIM	Global	Elapsed time in the simulation.

REFERENCES

Harbaugh, A.W. and McDonald, M.G., 1996, User's documentation for MODFLOW-96, an update to the U.S. Geological Survey modular finite-difference ground-water flow model: U.S. Geological Survey Open-File Report 96-485, 56 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, chap. A1, 586 p.