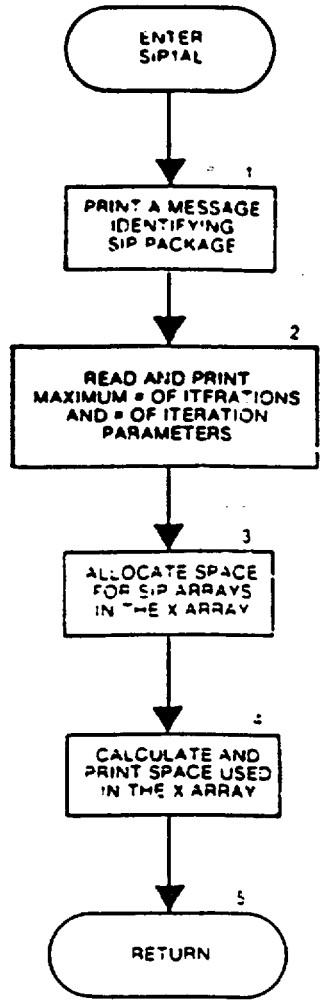
Flow Chart for Module SIPIAL

X array is the pool of memory space from which space is allocated for arrays used by various packages.



```
      SUBROUTINE SIP1AL(ISUM,LENX,LCEL,LCFL,LCGL,LCV,LCHOCG,LCLRCH,
     1        LCW,MXITER,NPARM,NCOL,NROW,NLAY,IN,IOUT)
C
C-----VERSION 1001 08DEC1983 SIP1AL
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(1H0,'SIP1 -- STRONGLY IMPLICIT PROCEDURE SOLUTION PACKAGE'
     1,', VERSION 1, 12/08/83',' INPUT READ FROM UNIT',I3)
C
C2------READ AND PRINT MXITER AND NPARM
      READ(IN,2) MXITER,NPARM
    2 FORMAT(2I10)
      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
      LCHOCG=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,I6,' ELEMENTS IN X ARRAY ARE USED BY SIP')
      ISUM1=ISUM-1
      WRITE(IOUT,5) ISUM1,LENX
    5 FORMAT(1X,I6,' ELEMENTS OF X ARRAY USED OUT OF',I7)
      IF(.ISUM1.GT.LENX) WRITE(IOUT,6)
    6 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C5------RETURN
      RETURN
      END
```

List of Variables for Module SIPIAL

Variable	Range	Definition
IN	Package	Primary unit number from which input for this package will be read.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
ISIZ	Module	Number of cells in the grid.
ISOLD	Package	Before this module allocates space, ISOLD is set equal to ISUM. After allocation, ISOLD is subtracted from ISUM to get ISP, the amount of space in the X array allocated by this module.
ISP	Module	Number of words in the X array allocated by this module.
ISUM	Global	Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.
ISUM1	Module	ISUM-1.
LCEL	Package	Location in the X array of the first element of array EL.
LCFL	Package	Location in the X array of the first element of array FL.
LCGL	Package	Location in the X array of the first element of array GL.
LCHDCG	Package	Location in the X array of the first element of array HDCG.
LCLRCH	Package	Location in the X array of the first element of array LRCH.
LCV	Package	Location in the X array of the first element of array V.
LCW	Package	Location in the X array of the first element of array W.
LENX	Global	Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.
MXITER	Package	Maximum number of iterations.
NCOL	Global	Number of columns in the grid.
NLAY	Global	Number of layers in the grid.
NPARM	Package	Number of iteration parameters.
NRC	Module	Number of cells in a layer.
NROW	Global	Number of rows in the grid.

Narrative for Module SIP1RP

Module SIP1RP reads data for the SIP package: the acceleration parameter (ACCL), the closure criterion (HCLOSE), the iteration-parameter seed (WSEED), a flag indicating whether WSEED is to be calculated or specified by the user (IPCALC), and the interval for printing head change (IPRSIP). If IPCALC is zero, iteration parameters are calculated using WSEED as the seed. Module SIP1RP performs its functions in the following order:

1. Read the data. If ACCL is zero, substitute a default of 1.0. If IPRSIP is less than or equal to zero, substitute an interval of 999 time steps. The defaults are provided as a convenience to the user.

2. Print the data read in step 1.

3. Check IPCALC which is a flag that indicates the source of the iteration-parameter seed (WSEED).

   (a) If IPCALC is not zero, submodule SSIP1I will calculate a seed and the resulting iteration parameters at the start of the first iteration. Print a message telling of this option.

   (b) If IPCALC is zero, use WSEED to calculate iteration parameters. The i-th iteration parameter (I_i) is given by the expression

$$I_i = 1 - (WSEED)^{\frac{i-1}{NPARM-1}}.$$

Print the parameters.

4. RETURN.

Flow Chart for Mocule SIP1RP

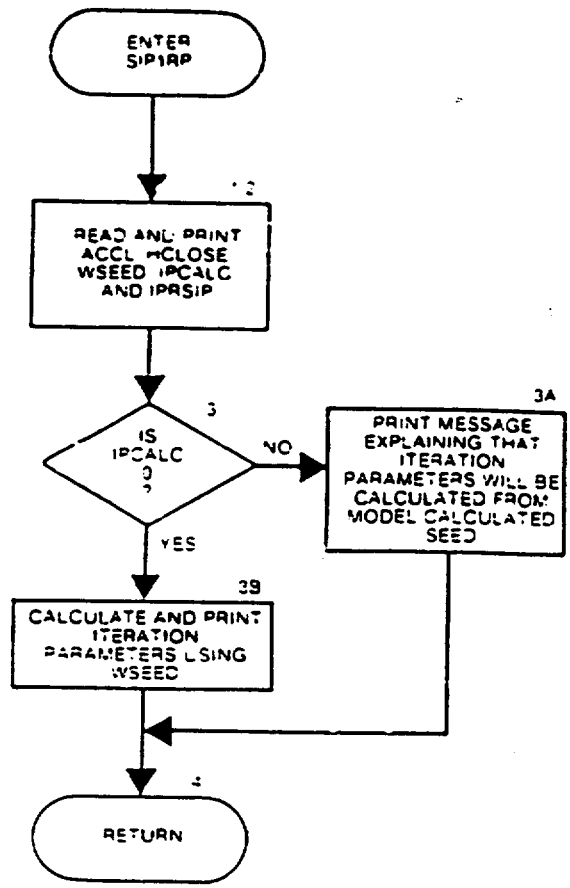
ACCL is a multiplier of calculated head change which is used to control the convergence rate.

HCLOSE is the head change closure criterion. When head change in all model cells is less than or equal to HCLOSE, iteration stops.

WSEED is the seed, specified by the user, on which the calculation of iteration parameters is based if IPCALC is zero.

IPRSIP is the time step interval for printing the maximum head change for each iteration of a time step. Head change is printed every IPRSIP time step. Head change is printed at the end of a stress period regardless of the interval.

IPCALC is a flag. If it is set equal to one, iteration parameters will be calculated from a seed calculated within the program. If it is clear (equal to zero), iteration parameters will be calculated from a seed provided by the user.



```
      SUBROUTINE SIP1RP(NPARM,MXITER,ACCL,HCLOSE,W,IN,IPCALC,IPRSIP,
     1                  IOUT)
C
C-----VERSION 0925 16DEC1982 SIP1RP
C
C     ******************************************************************
C     READ DATA FOR SIP
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION W(NPARM)
C     ------------------------------------------------------------------
C
C1------READ ACCL,HCLOSE,WSEED,IPCALC,IPRSIP
      READ(IN,1) ACCL,HCLOSE,IPCALC,WSEED,IPRSIP
    1 FORMAT(2F10.0,I10,F10.0,I10)
      IF(ACCL.EQ.0.) ACCL=1.
C
C2------PRINT DATA VALUES JUST READ
      WRITE(IOUT,100)
  100 FORMAT(1H0,///57X,'SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE'
     1/57X,43('-'))
      WRITE(IOUT,115) MXITER
  115 FORMAT(1H0,47X,'MAXIMUM ITERATIONS ALLOWED FOR CLOSURE =',I9)
      WRITE(IOUT,120) ACCL
  120 FORMAT(1H ,63X,'ACCELERATION PARAMETER =',G15.5)
      WRITE(IOUT,125) HCLOSE
  125 FORMAT(1H ,52X,'HEAD CHANGE CRITERION FOR CLOSURE =',E15.5)
      IF(IPRSIP.LE.0)IPRSIP=999
      WRITE(IOUT,130) IPRSIP
  130 FORMAT(1H ,52X,'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(1H0,52X,'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 P1=-1.
      P2=NPARM-1
      DO 160 I=1,NPARM
      P1=P1+1.
  160 W(I)=1.-WSEED**(P1/P2)
      WRITE(IOUT,161) NPARM,WSEED,(W(J),J=1,NPARM)
  161 FORMAT(1H0,/,15,' ITERATION PARAMETERS CALCULATED FROM',
     1      ' SPECIFIED WSEED =',F11.8,' :'//(10X,6E15.7))
C
C4------RETURN
 1000 RETURN
      END
```

List of Variables for Module SIP1RP

Variable	Range	Definition
ACCL	Package	Acceleration parameter.
HCLOSE	Package	Closure criterion for the iterative procedure.
I	Module	Index for iteration parameters.
IN	Package	Primary unit number from which input for this package will be read.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IPCALC	Package	Flag. ≠ 0, seed for iteration will be calculated in the program. = 0, seed will be specified by the user.
IPRSIP	Package	Frequency (in time steps) with which the maximum head changes for each iteration will be printed.
J	Module	Index for iteration parameters.
MXITER	Package	Maximum number of iterations.
NPARM	Package	Number of iteration parameters.
P1	Module	I - 1.
P2	Module	NPARM - 1.
W	Package	DIMENSION (NPARM), Iteration parameters.
WSEED	Module	Seed for calculating iteration parameters.

Narrative for Module SIP1AP

Module SIP1AP performs one iteration of the Strongly Implicit Procedure (SIP) algorithm for solving the flow equation. To save computational time, all arrays are declared one dimensional. The one-dimensional indexes are calculated from the layer, row, and column indexes normally used to access the arrays in three dimensions. Computational time is saved because knowledge of the geometry is used to make the calculations efficient and because the calculations are not repeated for identical indexes as would be done by internal FORTRAN addressing routines if three-dimensional subscripts were used.

This module is complex, partly because the SIP solution process requires that the same calculations be performed with two methods of ordering the equations. This is implemented by a generalized algorithm that uses the same computer statements to handle both ordering schemes. Checks are made to detect which ordering scheme is used, and array indexes are calculated accordingly.

Double precision is used for most calculations in this module in order to allow accurate answers to be calculated for a wide range of problems. Mixed precision arithmetic has been avoided by setting double-precision variables equal to single-precision values and then using the double-precision variables to make a completely double-precision expression.

Also, the reverse has been done. In the explanations below, each use of an assignment statement to change precision is not explained because of the large amount of text required. It is left to the reader to discern when this is occurring. When changing this module, care should be used

to maintain expressions that have unmixed precision. Mixed precision expressions can cause erroneous results with some compilers.

Module SIP1AP performs its functions in the following order:

1. If the user has specified (IPCALC ≠ 0) that iteration parameters should be calculated by the program, CALL submodule SSIP1I to calculate both the seed and the parameters.

2. Assign values to fields that are constant during an iteration.

3. Initialize the variables that track maximum head change during an iteration.

4. Clear SIP work arrays.

5. Determine the ordering of equations and set the ordering flag (IDIR) accordingly. This flag alternates between 1 and -1 each iteration. Calculate indexes IDNRC and IDNCOL which are used when calculating locations of neighboring cells.

6. Calculate the matrix $\bar{\bar{U}}$ and intermediate vector $\bar{V}$ using forward substitution. The elements in matrix $\bar{\bar{L}}$ are used as they are calculated; therefore, they are not saved. In the explanation of SIP concepts, the diagonals in the matrix $\bar{\bar{U}}$ were designated "e," "f," and "g." The corresponding field names in the program are EL (e lower case), FL (f lower case), and GL (g lower case). Similarly, the diagonals in the $\bar{\bar{L}}$ array which are "a," "b," "c," and "d" in the explanation are "AL," "BL," "CL," and "DL" in the program. The codes for the diagonals in matrix $\bar{\bar{A}}$ in the explanation are the same in the program. The codes for diagonals in $\overline{\overline{A+B}}$ in the explanation are followed by a "P" in the program. Hence, Z' in the

explanation is ZP (Z prime) in the program. The intermediate vector $\bar{V}$ in the explanation is the array "V" in the program.

(a) Set current cell indexes, II, JJ, KK. For normal ordering, the equation order is the same as the order of the loop indexes I,J,K. For reverse ordering, loop indexes I and K are inverted to produce the proper sequence of cells.

(b) Calculate the one-dimensional subscript of the current cell. If this cell is constant head or no flow, skip calculations for this cell and go on to the next.

(c) Calculate the one-dimensional subscripts for the six neighboring cells.

(d) Calculate the one-dimensional subscripts for conductance to each of the six neighboring cells. Since conductances between cells are assigned to array elements at specific cells (for example, CR(I,J,K) stores conductance between cells I,J,K and I,J+1,K), the four or five conductance subscripts are not simply the cell locations of the six neighboring cells as calculated in step 6(c). Also, the subscripts depend on equation ordering.

(e) Calculate or assign variables that are required for forward substitution and involve neighboring cells. Whenever a neighboring cell is outside of the grid, the variables are set to zero.

(1) Neighboring cell is one row back.

(2) Neighboring cell is one row ahead.

(3) Neighboring cell is one column back.

(4) Neighboring cell is one column ahead.

(5) Neighboring cell is one layer back.

(6) Neighboring cell is one layer ahead.

(f) Calculate the components of the upper and lower matrices $\bar{\bar{U}}$ and $\bar{\bar{L}}$, which are the factors of matrix $\overline{\overline{A+B}}$.

(g) Calculate the residual $\overline{RES}$. The calculation of HNW times HCOF is done in single precision so that the calculation will have precision comparable to similar calculations made in the formulation modules, all of which use single precision.

(h) Calculate the intermediate vector $\overline{V}$, which is stored in array V. This step completes the forward-substitution process for one cell.

7. Step through the cells solving for head change using back substitution.

(a) Set current cell indexes II, JJ, KK. The ordering is the reverse of that used for forward substitution (step 6(a)).

(b) Calculate the one-dimensional subscript of the current cell. If this cell is constant head or no flow, skip calculations for this cell and go to the next.

(c) Calculate the one-dimensional subscripts for the three neighboring cells behind (relative to the direction of the back-substitution ordering) the current cell.

(d) Back substitute, solving for head change. Store head change in array V in place of the intermediate values of vector $\overline{V}$. This doubling up of storage is used to save the cost of additional computer storage.

(e) Save the value of head change whose absolute value is largest during this iteration. Also, save the cell location where this head change occurred and the absolute value of the head change.

(f) Add the head change this iteration to head from the previous iteration to get a new estimate of head.

8. Store the head change whose absolute value is greatest this iteration and its cell location in arrays HDCG and LRCH. These may be printed in step 10 at the end of the time step. Set the convergence flag to one if the convergence criterion is met.

9. If the iteration is complete, print the number of iterations for the step; otherwise, RETURN.

10. Print the maximum head change and cell location each iteration if the SIP printout interval (IPRSIP) is reached. Printout occurs at the end of a stress period regardless of the interval.

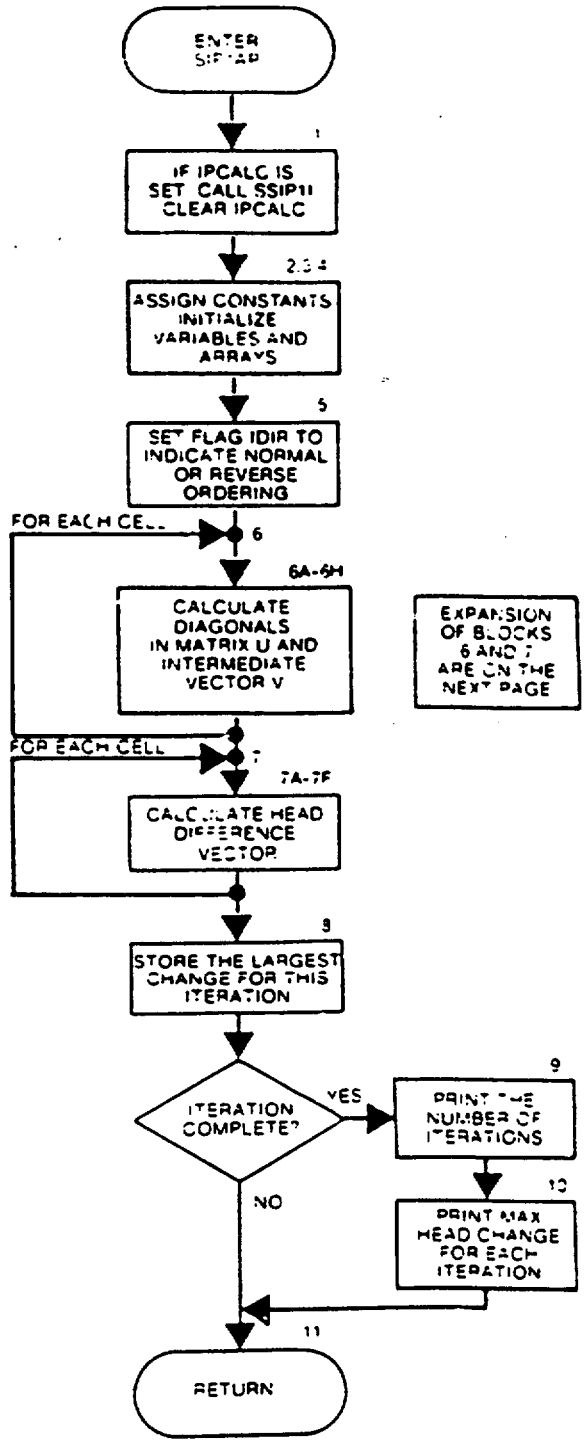
11. RETURN.

Flow Chart for Module SIP1AP

IPCALC is a flag. If it is set equal to one, the program calculates a seed from which iteration parameters are calculated. It may be set by the user at the beginning of the simulation. It is cleared during the first iterations. SSIP11 will never be called more than once. If IPCALC is not set equal to zero, the user specifies the seed for the iteration parameters.

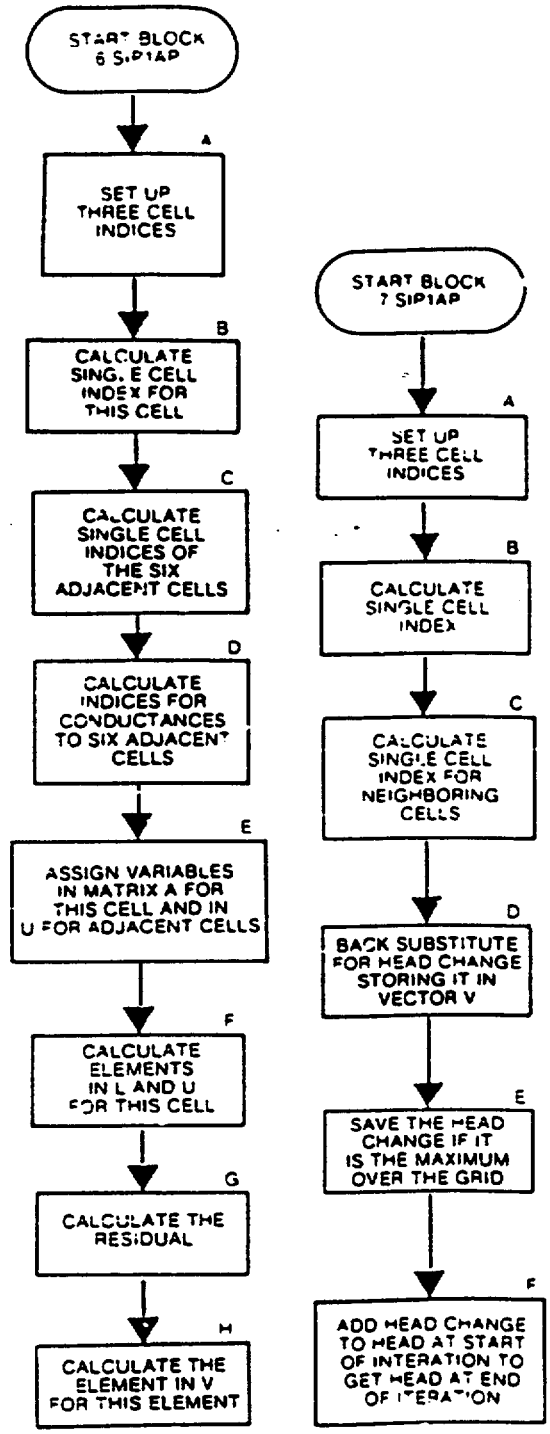
IDIR indicates whether the ordering of equations is normal (1) or reverse (-1).

SSIP11 is a submodule which calculates iteration parameters.



Flow Chart for Module SIP1AP (Continued)

Single Cell Index: In this module, a single index is used to identify each cell. This is in opposition to the three indices (I,J,K) used in most other modules.



```
      SUBROUTINE SIP1AP(HNEW,IBOUND,CR,CC,CV,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 1454 02DEC1982 SIP1AP
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,HCFHNW
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 SSIP11(CR,CC,CV,IBOUND,NPARM,W,NCOL,NROW,NLAY,IOUT)
      IPCALC=0
C
C2------ASSIGN VALUES TO FIELDS THAT ARE CONSTANT DURING AN ITERATION
      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=0.
C
C4------CLEAR SIP WORK ARRAYS.
      DO 100 I=1,NODES
      EL(I)=0.
      FL(I)=0.
      GL(I)=0.
  100 V(I)=0.
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(NRH)
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*EL-CL
      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)
      HNW=HNEW(N)
      HCFHNW=HNW*HCOF(N)
      RES=RRHS-ZHNEW-BHNE-DHNEW-E*HNEW(N)-HCFHNW-FHNEW-HHNEW-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
C7A-----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(1H0)
      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 SSIP1P(HDCG,LRCH,KITER,KSTP,KPER,MXITER,IOUT)
C
C11-----RETURN
600   RETURN
C
      END
```

List of Variables for Module SIP1AP

Variable	Range	Definition
AC	Module	Double-precision acceleration parameter (ACCL).
ACCL	Package	Acceleration parameter.
AL	Module	Diagonal from the lower factor. (AL stands for A-lower case.)
AP	Module	Diagonal element in the modified coefficient matrix. (AP stands for A-prime.)
B	Module	Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the last row.
BHNEW	Module	Head in the adjacent cell which is in the last row.
BIG	Module	Largest head change for an iteration.
BIGG	Module	Largest absolute value of head change for an iteration.
BL	Module	Diagonal from the lower factor. (BL stands for B-lower case.)
CC	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).
CL	Module	Diagonal from the lower factor. (CL stands for C-lower case.)
CP	Module	Diagonal element in the modified coefficient matrix. (CP stands for C-prime.)
CR	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K)
CV	Global	DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).
D	Module	Diagonal label in the coefficient matrix. Conductance from the adjacent node which is in the last column.
DHNEW	Module	Head in the adjacent cell which is in the last column.
DITPAR	Module	Double-precision iteration parameter.
DL	Module	Diagonal from the lower factor. (DL stands for D-lower case.)
DONE	Module	Double-precision field containing a one.
DZERO	Module	Double-precision field containing a zero.
E	Module	Main diagonal in the coefficient matrix.
EL	Module	DIMENSION (NODES), Diagonal from the upper factor. (EL stands for E-lower case.)
ELNCL	Module	EL (E-lower case) from the cell in the last column.
ELNLL	Module	EL (E-lower case) from the cell in the last layer.
ELNRL	Module	EL (E-lower case) from the cell in the last row.
ELXI	Module	Intermediate result.
F	Module	Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the next column.
FHNEW	Module	Head in the adjacent cell which is in the next column.
FL	Module	DIMENSION (NODES), Diagonal from the upper factor. (FL stands for F-lower case.)
FLNCL	Module	FL (F-lower case) from the cell in the last column.
FLNLL	Module	FL (F-lower case) from the cell in the last layer.
FLNRL	Module	FL (F-lower case) from the cell in the last row.

List of Variables for Module SIP1AP (Continued)

Variable	Range	Definition
FLXI	Module	Intermediate result.
GL	Module	DIMENSION (NODES), Diagonal from the upper factor. (GL stands for G-lower case.)
GLNCL	Module	GL (G-lower case) from the cell in the last column.
GLNLL	Module	GL (G-lower case) from the cell in the last layer.
GLNRL	Module	GL (G-lower case) from the cell in the last row.
GLX1	Module	Intermediate result.
GP	Module	Diagonal element in the modified coefficient matrix. (GP stands for G-prime.)
H	Module	Diagonal label in the coefficient matrix. Conductance from the adjacent node which is in the next row.
HCFHNW	Module	Product of head and HCOF for a cell.
HCLOSE	Package	Closure criterion for the iterative procedure.
HCOF	Global	DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K) in the finite-difference equation.
HDCG	Package	DIMENSION (MXITER), Maximum head change for each iteration.
HHCOF	Module	Double-precision HCOF.
HHNEW	Module	Head in the adjacent cell which is in the next row.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
HNW	Module	Temporary field for HNEW(N).
I	Module	Index for nodes and rows.
IB	Module	Row number of the cell having the largest head change.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell. < 0, constant-head cell = 0, inactive cell > 0, variable-head cell
ICNVG	Global	Flag is set equal to one when the iteration procedure has converged.
IDIR	Module	Indicator for direction of solution algorithm. +1 - forward -1 - reverse
IDNCOL	Module	Intermediate result used to calculate indices.
IDNRC	Module	Intermediate result used to calculate indices.
II	Module	Row number.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IPCALC	Package	Flag. = 0, iteration parameter seed (WSEED) is entered by the user. = 1, seed is calculated in the program.
IPRSIP	Package	Frequency (in time steps) with which the maximum head changes for each iteration will be printed.
J	Module	Index for columns.
JB	Module	Column number of the cell having the largest head change.
JJ	Module	Column index.
K	Module	Index for layers.
KB	Module	Layer of the cell having the largest head change.
KITER	Global	Iteration counter. Reset at the start of each time step.

List of Variables for Module SIP1AP (Continued)

Variable	Range	Definition
KK	Module	Layer index.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. Reset at the start of each stress period.
LRCH	Package	DIMENSION (MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.
MXITER	Package	Maximum number of iterations.
N	Module	Cell index.
NC	Module	Index for the adjacent cell in the last column.
NCD	Module	One-dimensional subscript of conductance to the adjacent cell which is in the last column.
NCF	Module	One-dimensional subscript of conductance to the adjacent cell which is in the next column.
NCL	Module	One-dimensional subscript of the cell index of the adjacent cell which is in the last column.
NCN	Module	One-dimensional subscript of the cell index of the adjacent cell which is in the next column.
NCOL	Global	Number of columns in the grid.
NL	Module	Index for the adjacent cell in the last layer.
NLAY	Global	Number of layers in the grid.
NLL	Module	One-dimensional subscript of the cell index of the adjacent cell which is in the last layer.
NLN	Module	One-dimensional subscript of the cell index of the adjacent cell which is in the next layer.
NLS	Module	One-dimensional subscript of conductance to the adjacent cell which is in the next layer.
NLZ	Module	One-dimensional subscript of conductance to the adjacent cell which is in the last layer.
NODES	Global	Number of cells (nodes) in the finite-difference grid.
NPARM	Package	Number of iteration parameters.
NR	Module	Index for the adjacent cell in the last row.
NRB	Module	One-dimensional subscript of conductance to the adjacent cell which is in the last row.
NRC	Module	Number of cells in the layer.
NRH	Module	One-dimensional subscript of conductance to the adjacent cell which is in the next row.
NRL	Module	One-dimensional subscript of the cell index of the adjacent cell which is in the last row.
NRN	Module	One-dimensional subscript of the cell index of the adjacent cell which is in the next row.
NROW	Global	Number of rows in the grid.
NSTP	Global	Number of time steps in the current stress period.
NTH	Module	Index for iteration parameters.
RES	Module	Residual.
RHS	Global	DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.

List of Variables for Module SIP1AP (Continued)

Variable	Range	Definition
RP	Module	Diagonal element in the modified coefficient matrix. (RP stands for R-prime.)
RRHS	Module	Double-precision right hand side of the equation.
S	Module	Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the next layer.
SHNEW	Module	Head in the adjacent cell which is in the next layer.
TCHK	Module	Absolute value of head change for a single cell.
TP	Module	Diagonal element in the modified coefficient matrix. (TP stands for T-prime.)
UP	Module	Diagonal element in the modified coefficient matrix. (UP stands for U-prime.)
V	Package	DIMENSION (NODES), Intermediate result.
VN	Module	Temporary double-precision V(N).
VNCL	Module	Element in the intermediate vector for the cell in the last column.
VNLL	Module	Element in the intermediate vector for the cell in the last layer.
VNLLI	Module	Artifact of an earlier version of the program.
VNRL	Module	Element in the intermediate vector for the cell in the last row.
W	Package	DIMENSION (NPARM), Iteration parameters.
XI	Module	Double-precision V(N).
Z	Module	Diagonal label in the coefficient matrix--conductance from the adjacent node which is in the last layer.
ZHNEW	Module	Head in the adjacent cell which is in the last layer.

Narrative for Module SSIP1P

Submodule SSIP1P prints the largest value of head change (HDCG) out of all cells for each iteration of a time step. Also printed is the cell location (LRCH) where the change occurs. The submodule is so short that no numbered comments are used and no flow chart is provided.

```
      SUBROUTINE SSIP1P(HDCG,LRCH,KITER,KSTP,KPER,MXITER,IOUT)
C
C
C-----VERSION 1504 08DEC1982 SSIP1P
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(1H0,'MAXIMUM HEAD CHANGE FOR EACH ITERATION:'/
     1    1H0,5(' HEAD CHANGE LAYER,ROW,COL')/1X,132('-'))
      WRITE (IOUT,10) (HDCG(J),(LRCH(I,J),I=1,3),J=1,KITER)
   10 FORMAT((1X,5(G12.4,' (',I3,',',I3,',',I3,')'))))
      WRITE(IOUT,11)
   11 FORMAT(1H0)
C
      RETURN
C
      END
```

List of Variables for Module SSIP1P

Variable	Range	Definition
HDCG	Package	DIMENSION (MXITER), Maximum head change for each iteration.
I	Module	Index for cell location.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
J	Module	Index for iterations.
KITER	Global	Iteration counter. Reset at the start of each time step.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. Reset at the start of each stress period.
LRCH	Package	DIMENSION (MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.
MXITER	Package	Maximum number of iterations.

Narrative for Module SSIP1I

Submodule SSIP1I calculates an iteration-parameter seed using model-conductance values and grid dimensions. Although a single seed is required, the method of calculation requires that three-directional seeds be calculated for each active cell. Then a cell seed, the minimum of the three, is selected. Finally, all the cell seeds are averaged to give the grid seed. This grid seed is then used to calculate the iteration parameters. The minimum cell seed is also printed.

Submodule SSIP1I performs its functions in the following order:

1. Calculate constants and initialize variables. In order to calculate the average cell seed, accumulators AVGSUM (sum of the cell seeds) and NODES (sum of the active cells for which a seed is calculated) are required. These are initialized to zero. WMINMN is used to store the smallest cell seed. Since this value must always be less than one, it is initialized to 1.0. The three coefficients, CCOL, CROW, and CLAY are set equal to $\pi^2/2(NCOL)^2$, $\pi^2/2(NROW)^2$, and $\pi^2/2(NLAY)^2$, respectively.

2. Loop through all cells, calculating a cell seed for each active cell.

(a) Find the conductances from the cell to each of the six adjacent cells. Conductance across the grid boundary is set equal to zero.

(b) Find the maximum and minimum of the two conductances in the row direction (DFMX, DFMN), in the column direction (BHMX, BHMN), and in the vertical direction (ZSMX, ZSMN). If the minimum is zero (which indicates that a neighbor is no flow), set the minimum equal to the maximum.

(c) Calculate three-directional seeds (WCOL, WROW, WLAY) using the relations

WCOL = CCOL/(1. + (BHMX + ZSMX)/DFMN);

WROW = CROW/(1. + (DFMX + ZSMX)/BHMN); and

WLAY = CLAY/(1. + (DFMX + BHMX)/ZSMN).

If the minimum conductance is zero (that is, both the minimum and the maximum are zero), set the seed equal to 1.0. This value will be ignored when the cell seed (the minimum-directional seed) is selected in step 2(d) because any valid seed will be less than 1.0.

(d) Select the minimum of the three-directional seeds as the cell seed. If it is the smallest cell seed used so far, store it in WMIMMN. Accumulate the sum of the cell seeds and the total number of active cells so that the average of all cell seeds can be calculated in step 3.

3. Calculate the grid seed (the average cell seed) and print it along with the minimum seed.

4. Calculate and print iteration parameters using the grid seed with the relation

$$I_i = 1 - (\mathrm{SEED})^{\frac{i-1}{\mathrm{NPARM} - 1}}$$

where

I_i is the i-th iteration parameter, and

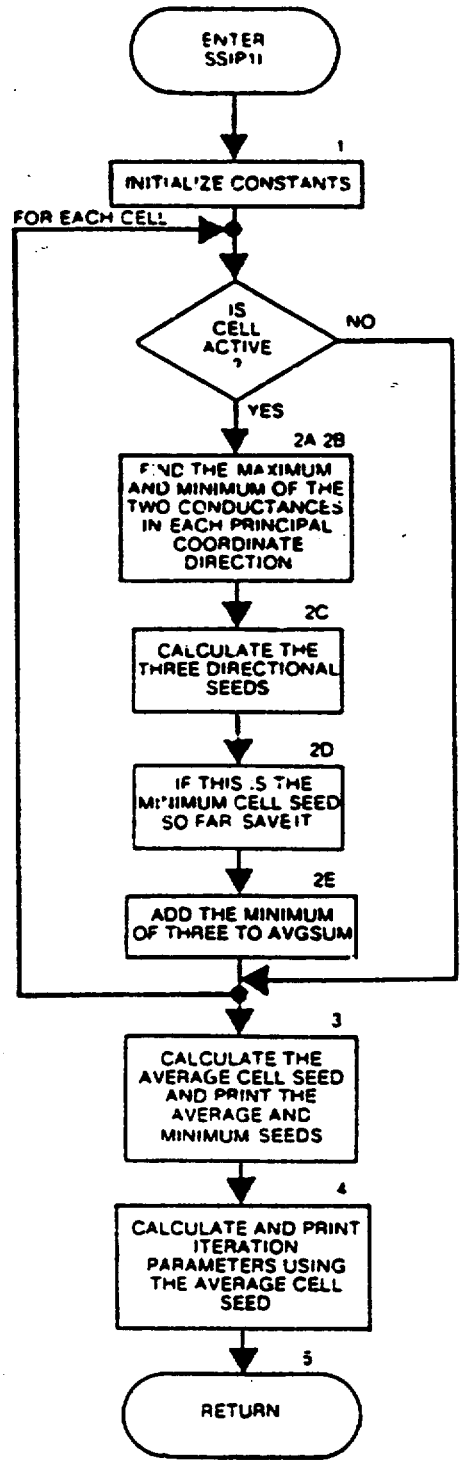
NPARM is the number of iteration parameters.

5. RETURN.

Flow Chart for Module SSIP11

Seed: the "grid seed" is the single parameter used to calculate the iteration parameters. To calculate the grid seed, several intermediate variables, called "cell seeds," are used. For each cell, three "directional seeds" are calculated. The minimum directional seed for a cell is the "cell seed." The "grid seed" is the average of the cell seeds.

AVGSUM is an accumulator to which each cell seed is added. It is then divided by the number of cells to obtain the average cell seed which is used as the grid seed.



```
      SUBROUTINE SSIP11(CR,CC,CV,IBOUND,NPARM,W,NCOL,NROW,NLAY,
     1          IOUT)
C
C-----VERSION 1743 26APR1983 SSIP11
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
      PIEPIE=9.869604
      R=NROW
      C=NCOL
      L=NLAY
      CCOL=PIEPIE/(2.*C*C)
      CROW=PIEPIE/(2.*R*R)
      CLAY=PIEPIE/(2.*L*L)
      WMINMN=1.
      AVGSUM=0.
      NODES=0
C
C2------LOOP THROUGH ALL CELLS, CALCULATING A SEED FOR EACH CELL
C2------THAT IS ACTIVE
      DO 100 K=1,NLAY
      DO 100 I=1,NROW
      DO 100 J=1,NCOL
      IF(IBOUND(J,I,K).LE.0) GO TO 100
C
C2A-----CONDUCTANCE FROM THIS CELL
C2A-----TO EACH OF THE 6 ADJACENT CELLS
      D=0.
      IF(J.NE.1) D=CR(J-1,I,K)
      F=0.
      IF(J.NE.NCOL) F=CR(J,I,K)
      B=0.
      IF(I.NE.1) B=CC(J,I-1,K)
      H=0.
      IF(I.NE.NROW) H=CC(J,I,K)
      Z=0.
      IF(K.NE.1) Z=CV(J,I,K-1)
      S=0.
      IF(K.NE.NLAY) S=CV(J,I,K)
C
C2B-----FIND THE MAXIMUM AND MINIMUM OF THE 2 CONDUCTANCE COEFFICIENTS
C2B-----IN EACH PRINCIPAL COORDINATE DIRECTION
      DFMX=AMAX1(D,F)
```

```
      BHMX=AMAX1(B,H)
      ZSMX=AMAX1(Z,S)
      DFMN=AMIN1(D,F)
      BHMN=AMIN1(B,H)
      ZSMN=AMIN1(Z,S)
      IF(DFMN.EQ.0.) DFMN=DFMX
      IF(BHMN.EQ.0.) BHMN=BHMX
      IF(ZSMN.EQ.0.) ZSMN=ZSMX
C
C2C-----CALCULATE A SEED IN EACH PRINCIPAL COORDINATE DIRECTION
      WCOL=1.
      IF(DFMN.NE.0.) WCOL=CCOL/(1.+(BHMX+ZSMX)/DFMN)
      WROW=1.
      IF(BHMN.NE.0.) WROW=CROW/(1.+(DFMX+ZSMX)/BHMN)
      WLAY=1.
      IF(ZSMN.NE.0.) WLAY=CLAY/(1.+(DFMX+BHMX)/ZSMN)
C
C2D-----SELECT THE CELL SEED, WHICH IS THE MINIMUM SEED OF THE 3.
C2D-----SELECT THE MINIMUM SEED OVER THE WHOLE GRID.
      WMIN=AMIN1(WCOL,WROW,WLAY)
      WMINMN=AMIN1(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(1H0,'AVERAGE SEED =',F11.8/1X,'MINIMUM SEED =',F11.8)
C
C4------CALCULATE AND PRINT ITERATION PARAMETERS FROM THE AVERAGE SEED
      P1=-1.
      P2=NPARM-1
      DO 50 I=1,NPARM
      P1=P1+1.
   50 W(I)=1.-AVGMIN**(P1/P2)
      WRITE(IOUT,150) NPARM,(W(J),J=1,NPARM)
  150 FORMAT(1H0,/,I5,' ITERATION PARAMETERS CALCULATED FROM',
     1       ' AVERAGE SEED:'//(10X,6E15.7))
C
C5------RETURN
      RETURN
      END
```

List of Variables for Module SSIP11

Variable	Range	Definition
AVGMIN	Module	Mean WMIN.
AVGSUM	Module	Sum of all of WMIN's.
B	Module	Conductance between this node and the one to the rear.
BHMN	Module	Minimum of B and H (if the minimum is 0, it is the maximum).
BHMX	Module	Maximum of B and H.
C	Module	Number of columns.
CC	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J+1,I,K).
CCOL	Module	Intermediate factor.
CLAY	Module	Intermediate factor.
CR	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K).
CROW	Module	Intermediate factor.
CV	Global	DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).
D	Module	Conductance between this node and the one to the left.
DFMN	Module	Minimum of D and F (if the minimum is 0, it is the maximum).
DFMX	Module	Maximum of D and F.
DWMIN	Module	Double precision WMIN.
F	Module	Conductance between this node and the one to the right.
H	Module	Conductance between this node and the one to the front.
I	Module	Index for rows.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell. < 0, constant-head cell = 0, inactive cell > 0, variable-head cell

List of Variables for Module SSIP11 (Continued)

Variable	Range	Definition
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
J	Module	Index for columns.
K	Module	Index for layers.
L	Module	Number of layers.
NCOL	Global	Number of columns in the grid.
NLAY	Global	Number of layers in the grid.
NODES	Module	Number of variable-head (IBOUND > 0) cells in the grid.
NPARM	Package	Number of iteration parameters.
NROW	Global	Number of rows in the grid.
PIEPIE	Module	PI squared.
P1	Module	Index for the number of parameters.
P2	Module	NPARM-1.
R	Module	Number of rows.
S	Module	Conductance between this node and the one below.
TMP	Module	Temporary field for nodes.
W	Package	DIMENSION (NPARM), Iteration parameters.
WCOL	Module	Seed in the column direction for a cell.
WLAY	Module	Seed in the layer direction for a cell.
WMIN	Module	Minimum of (WCOL, WLAY, WROW).
WMINMN	Module	Minimum WMIN.
WROW	Module	Seed in the row direction for a cell.
Z	Module	Conductance between this node and the one above.
ZSMN	Module	Minimum of Z and S (if minimum is 0, it is the maximum).
ZSMX	Module	Maximum of Z and S.

CHAPTER 13

SLICE-SUCCESSIVE OVERRELAXATION PACKAGE

Conceptualization and Implementation

Slice-successive overrelaxation (SSOR) is a technique for iteratively solving a system of linear equations. The SSOR package, in particular, solves the finite-difference equations associated with the cells in the finite-difference grid. Those equations with the iteration counter "ℓ" included have the form

$$
\begin{aligned}
&CV_{i,j,k-1/2}h^{m,\ell+1}_{i,j,k-1} + CC_{i-1/2,j,k}h^{m,\ell+1}_{i-1,j,k} + CR_{i,j-1/2,k}h^{m,\ell+1}_{i,j-1,k} \\
&+ (-CV_{i,j,k-1/2} - CC_{i-1/2,j,k} - CR_{i,j-1/2,k} - CR_{i,j+1/2,k} \\
&- CC_{i+1/2,j,k} - CV_{i,j,k+1/2} - HCOF_{i,j,k})h^{m,\ell+1}_{i,j,k} + CR_{i,j+1/2,k}h^{m,\ell+1}_{i,j+1,k} \\
&+ CC_{i+1/2,j,k}h^{m,\ell+1}_{i+1,j,k} + CV_{i,j,k+1/2}h^{m,\ell+1}_{i,j,k+1} = RHS_{i,j,k} \qquad (122)
\end{aligned}
$$

where

m is the time step counter; and

ℓ is the iteration counter.

Equation 122 can be written in residual form as

$$
\begin{aligned}
&CV_{i,j,k-1/2,}(h^{m,\ell+1}_{i,j,k-1} - h^{m,\ell}_{i,j,k-1}) + CC_{i-1/2,j,k}(h^{m,\ell+1}_{i-1,j,k} - h^{m,\ell}_{i-1,j,k}) \\
&+ CR_{i,j-1/2,k}(h^{m,\ell+1}_{i,j-1,k} - h^{m,\ell}_{i,j-1,k}) + (-CV_{i,j,k-1/2} - CC_{i-1/2,j,k} \\
&- CR_{i,j-1/2,k} - CR_{i,j+1/2,k} - CC_{i+1/2,j,k} - CV_{i,j,k+1/2} \\
&+ HCOF_{i,j,k})(h^{m,\ell+1}_{i,j,k} - h^{m,\ell}_{i,j,k}) + CR_{i,j+1/2,k}(h^{m,\ell+1}_{i,j+1,k} - h^{m,\ell}_{i,j+1,k}) \\
&+ CC_{i+1/2,j,k}(h^{m,\ell+1}_{i+1,j,k} - h^{m,\ell}_{i+1,j,k}) + CV_{i,j,k+1/2}(h^{m,\ell+1}_{i,j,k+1} - h^{m,\ell}_{i,j,k+1}) =
\end{aligned}
$$

$$RHS_{i,j,k} - CV_{i,j,k-1/2}h^{m,\ell}_{i,j,k-1} - CC_{i-1/2,j,k}h^{m,\ell}_{i-1,j,k}$$

$$- CR_{i,j-1/2,k}h^{m,\ell}_{i,j-1,k} - (- CV_{i,j,k-1/2} - CC_{i-1/2,j,k} - CR_{i,j-1/2,k}$$

$$- CR_{i,j+1/2,k} - CC_{i+1/2,j,k} - CV_{i,j,k+1/2} + HCOF_{i,j,k})h^{m,\ell}_{i,j,k}$$

$$- CR_{i,j+1/2,k}h^{m,\ell}_{i,j+1,k} - CC_{i+1/2,j,k}h^{m,\ell}_{i+1,j,k} - CV_{i,j,k+1/2}h^{m,\ell}_{i,j,k+1} \quad (123)$$

The left hand side of equation 123 consists of terms involving the seven head changes e.g., $(h^{m,\ell+1}_{i,j,k}-h^{m,\ell}_{i,j,k})$ for iteration $\ell+1$. At the beginning of iteration $\ell+1$, those head changes are unknown. The right hand side of equation 123 consists of the term RHS and terms involving the head at the end of iteration ℓ, all of which are known at the beginning of iteration $\ell+1$. The difficulty encountered in solving the system of equations having the form of equation 123 is that there are too many equations--there is an equation for each cell in the grid. SSOR reduces the number of equations by simultaneously solving only those equations representing cells in a single slice (fig. 54). Note that the slice is identified by its row number; therefore, in this discussion, the terms row and slice will be used interchangeably.

Consider the equation for cells in row 1--that is, those equations for which i is equal to one. Since i-1 will then be equal to zero, the heads $h^{m,\ell+1}_{i-1,j,k}$ and $h^{m,\ell}_{i-1,j,k}$ will be equal to zero. Therefore, the term $CC_{i-1/2,j,k}(h^{m,\ell+1}_{i-1,j,k}-h^{m,\ell}_{i-1,j,k})$ is equal to zero. Furthermore, the head $h^{m,\ell+1}_{i+1,j,k}$ which is the head in a cell in row 2, can be approximated--only while heads in row 1 are being calculated---by $h^{m,\ell}_{i+1,j,k}$ (the head calculated at the previous iteration). Then the term $CC_{i+1/2,j,k}(h^{m,\ell+1}_{i+1,j,k}-h^{m,\ell}_{i+1,j,k})$ is equal to zero. Thus equation 123 for cells in row 1 can be approximated by the equation

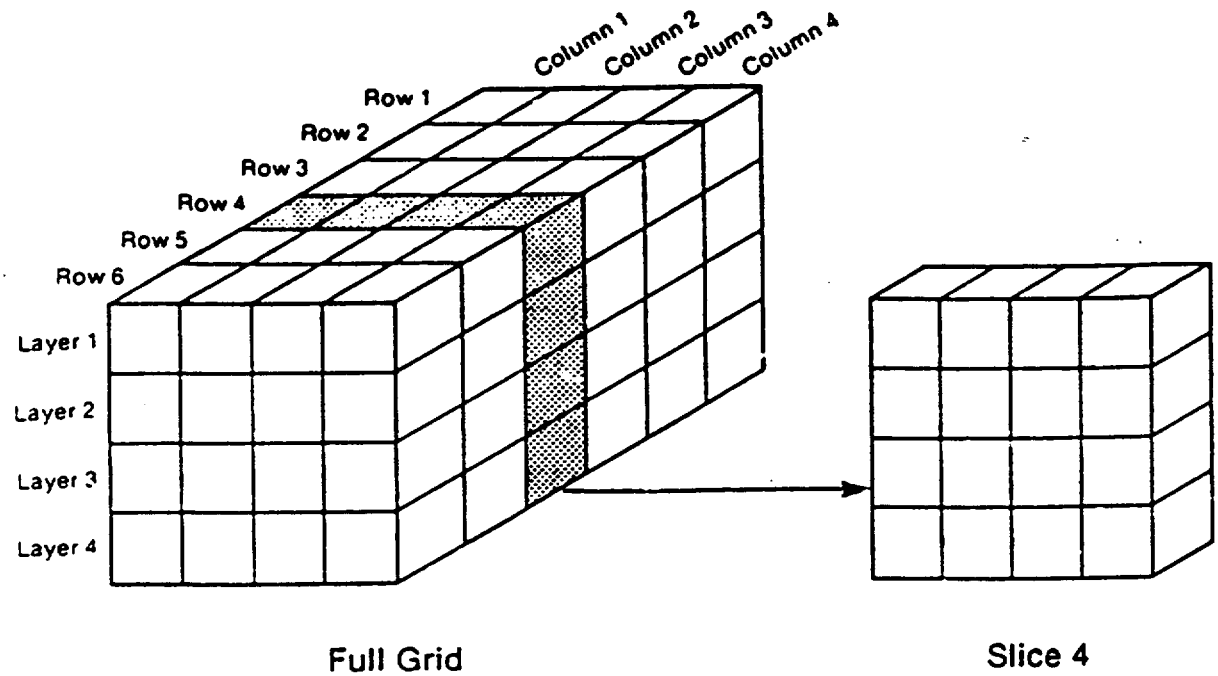


Figure 54.—SSOR reduces the number of equations that must be solved simultaneously by considering a single vertical slice at a time.

$$CV_{i,j,k-1/2,k}(\tilde{h}^{m,\ell+1}_{i,j,k-1}-h^{m,\ell}_{i,j,k-1}) + CR_{i,j-1/2,k}(\tilde{h}^{m,\ell+1}_{i,j-1,k}-h^{m,\ell}_{i,j-1,k})$$

$$+ (- CV_{i,j,k-1/2} - CC_{i-1/2,j,k} - CR_{i,j-1/2,k} - CR_{i,j+1/2,k}$$

$$- CC_{i+1/2,j,k} - CV_{i,j,k+1/2} + HCOF_{i,j,k})(\tilde{h}^{m,\ell+1}_{i,j,k}-h^{m,\ell}_{i,j,k})$$

$$+ CR_{i,j+1/2,k}(\tilde{h}^{m,\ell+1}_{i,j+1,k}-h^{m,\ell}_{i,j+1,k}) + CV_{i,j,k+1/2}(\tilde{h}^{m,\ell+1}_{i,j,k+1}-h^{m,\ell+1}_{i,j,k+1}) =$$

$$RHS_{i,j,k} - CV_{i,j,k-1/2}h^{m,\ell}_{i,j,k-1} - CC_{i-1/2,j,k}h^{m,\ell+1}_{i-1,j,k}$$

$$- CR_{i,j-1/2,k}h^{m,\ell}_{i,j-1,k} - (- CV_{i,j,k-1/2} - CC_{i-1/2,j,k} - CR_{i,j-1/2,k}$$

$$- CR_{i,j+1/2,k} - CC_{i+1/2,j,k} - CV_{i,j,k+1/2} + HCOF_{i,j,k})h^{m,\ell}_{i,j,k}$$

$$- CR_{i,j+1/2,k}h^{m,\ell}_{i,j+1,k} - CC_{i+1/2,j,k}h^{m,\ell}_{i+1,j,k} - CV_{i,j,k+1/2}h^{m,\ell}_{i,j,k+1} \quad (124)$$

where $\tilde{h}^{m,\ell+1}_{i,j,k}-h^{m,\ell}_{i,j,k}$ is approximately equal to $h^{m,\ell+1}_{i,j,k}-h^{m,\ell}_{i,j,k}$. The term $\tilde{h}^{m,\ell+1}_{i,j,k}-h^{m,\ell}_{i,j,k}$ is a first estimate of head change for cell i,j,k at iteration $\ell+1$. The equations of the form of equation 123 for cells in row 1 can be replaced by equations of the form of equation 124. Those equations constitute a system of n simultaneous linear equations (where n is the number of cells in row 1) in n unknowns. In most ground-water problems, n is relatively small; therefore, the n equation can be solved directly using Gaussian elimination to get values of $\tilde{h}^{m,\ell+1}_{i,j,k}-h^{m,\ell}_{i,j,k}$ in row 1. The value $h^{m,\ell+1}_{i,j,k}$ can then be calculated by the equation

$$h^{m,\ell+1}_{i,j,k} = h^{m,\ell}_{i,j,k} + \omega(\tilde{h}^{m,\ell+1}_{i,j,k} - h^{m,\ell}_{i,j,k}) \quad (125)$$

where ω is an acceleration parameter--generally between 1 and 2. Each of the rest of the slices can be handled in a similar manner. Terms in equations for slice P which involve heads in slice P+1 use heads from

the preceding iteration. Thus the equations for slice P approximate equations of the form of equation 123. Solution of those equations gives approximate head changes for the current iteration. The approximate head changes are multiplied by the relaxation factor to get the final head changes for the iteration.

Implementation of slice-successive overrelaxation involves processing the rows one at a time, first reformulating the equations for each row from the form of equation 123 to the form of equation 124, and then solving the resulting equations to get the approximate head changes for each cell in the row. Finally, the head changes must be multiplied by the relaxation factor and added to the head at the previous iteration to get the new head value.

The equations for a row can be written in matrix form

$$\bar{\bar{A}}\bar{X} = \bar{B} \tag{126}$$

where

$\bar{\bar{A}}$ is the coefficient matrix;

$\bar{X}$ is the vector of head changes from iteration ℓ to iteration $\ell+1$; and

$\bar{B}$ is the vector of residuals.

The matrix $\bar{\bar{A}}$ is symmetric and banded (fig. 55) with a maximum half-bandwidth equal to the number of layers. Therefore, just the lower triangular portion of the coefficient matrix needs to be stored (fig. 55). It is stored in a rectangular array. $\bar{\bar{A}}$, dimensioned by the maximum half-bandwidth plus one and the number of cells in a slice.

a_{11}	a_{12}		a_{14}					
a_{12}	a_{22}	a_{23}		a_{25}				
	a_{23}	a_{33}			a_{36}			
a_{14}			a_{44}	a_{45}		a_{47}		
	a_{25}		a_{45}	a_{55}	a_{56}			
		a_{36}		a_{56}	a_{66}		a_{68}	
			a_{47}			a_{77}		
					a_{68}		a_{88}	a_{99}
							a_{89}	a_{99}

Full Matrix

a_{11}	a_{22}	a_{33}	a_{44}	a_{55}	a_{66}	a_{77}	a_{88}	a_{99}
a_{12}	a_{23}		a_{45}	a_{56}			a_{89}	
					a_{68}			
a_{14}	a_{25}	a_{36}	a_{47}					

Compressed Matrix

Figure 55.—Organization of elements in a matrix for a slice in conventional notation and in compressed notation.

At the end of each iteration, the maximum head change for the iteration is compared to the closure criterion. If the maximum head change is smaller than the closure criterion, a flag (ICNVG) is set equal to one. The flag is returned to the main program. If it is set, the main program terminates the iteration loop.

Slice-Successive Overrelaxation Package Input

Input to the Slice-Successive Overrelaxation (SOR) Package is read from the unit specified in IUNIT(11).

FOR EACH SIMULATION

SOR1AL

1. Data: MXITER
   Format: I10

SOR1RP

2. Data: ACCL HCLOSE IPRSOR
   Format: F10.0 F10.0 I10

Explanation of Fields Used in
Input Instructions

MXITER--is the maximum number of iterations allowed in a time step.

ACCL--is the acceleration parameter, usually between 1.0 and 2.0.

HCLOSE--is the head change criterion for convergence. When the maximum absolute value of head change from all nodes during an iteration is less than or equal to HCLOSE, iteration stops.

IPRSOR--is the printout interval for SOR. IF IPRSOR is equal to zero, it is changed to 999. The maximum head change (positive or negative) is printed for each iteration of a time step whenever the time step is an even multiple of IPRSOR. This printout also occurs at the end of each stress period regardless of the value of IPRSOR.

Module Documentation for the Slice-Successive Overrelaxation Package

The Slice-Successive Overrelaxation Package (SOR1) consists of three primary modules and one submodule. They are:

Primary Modules

SOR1AL	Allocates space for arrays.
SOR1RP	Reads control information needed by the SOR1 Package.
SOR1AP	Performs one iteration of slice-successive overrelaxation.

Submodule

SSOR1B	Solves a system of linear equations.

Narrative for Module SOR1AL

Module SOR1AL allocates space in the X array for SOR arrays. The SOR arrays are A, RES, IEQPNT, HDCG, and LRCH. "A" holds the main diagonal and the lower diagonals of the symmetric coefficient matrix for a single slice. RES holds the residual vector (the right hand sides) for a single slice. IEQPNT holds a sequential identification number for each cell in a slice. HDCG holds the maximum head change for each iteration. LRCH holds the location of the cell (row, column, and layer) which had the maximum head change for each iteration.

Module SOR1AL performs its functions in the following order:

1. Print a message identifying the SOR Package.

2. Read and print the maximum number of iterations.

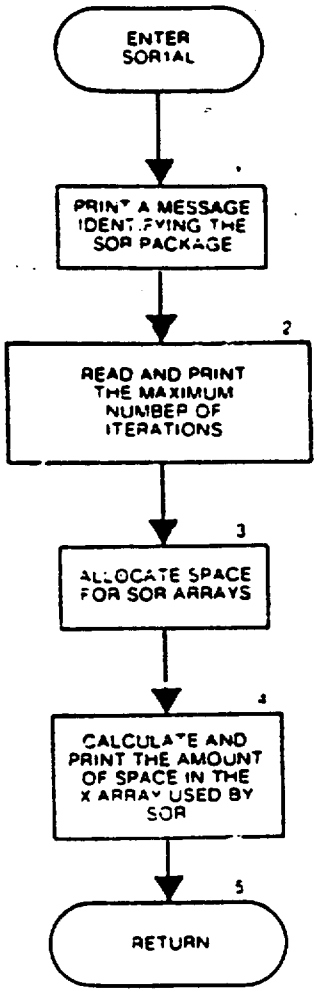
3. Allocate the required space in the X array. The X-array location pointer (ISUM) is saved in the variable ISOLD prior to allocation so that the space required for SOR can be calculated in step 4. To allocate space for an array, the array-location variable is set equal to ISUM. Then ISUM is incremented by the required number of elements.

4. Calculate and print the space used in the X array. The space used by SOR is ISUM - ISOLD.

5. RETURN

Flow Chart for Module SOR1AL

X array is the pool of memory space from which space is allocated for arrays used by various packages.



```
      SUBROUTINE SOR1AL(ISUM,LENX,LCA,LCRES,LCHDCG,LCLRCH,LCIEQP,
     1        MXITER,NCOL,NROW,NLAY,NSLICE,MBW,IN,IOUT)
C
C-----VERSION 1003 03DEC1983 SOR1AL
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(1H0,'SOR1 -- SLICE-SUCCESSIVE OVERRELAXATION PACKAGE'
     1,', VERSION 1, 12/08/83',' INPUT READ FROM UNIT',I3)
C
C2------READ AND PRINT MXITER (MAXIMUM # OF ITERATIONS)
      READ(IN,2) MXITER
    2 FORMAT(I10)
      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,I6,' ELEMENTS IN X ARRAY ARE USED BY SOR')
      ISUM1=ISUM-1
      WRITE(IOUT,5) ISUM1,LENX
    5 FORMAT(1X,I6,' ELEMENTS OF X ARRAY USED OUT OF',I7)
      IF(ISUM1.GT.LENX) WRITE(IOUT,6)
    6 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C5------RETURN
      RETURN
      END
```

List of Variables for Module SORIAL

Variable	Range	Definition
IN	Package	Primary unit number from which input for this package will be read.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
ISOLD	Package	Before this module allocates space, ISOLD is set equal to ISUM. After allocation, ISOLD is subtracted from ISUM to get ISP, the amount of space in the X array allocated by this module.
ISP	Module	Number of words in the X array allocated by this module.
ISUM	Global	Index number of the lowest element in the X array which has not yet been allocated. When space is allocated for an array, the size of the array is added to ISUM.
ISUM1	Module	Index number of the last element of the X array allocated by this module.
LCA	Package	Location in the X array of the first element of array A.
LCHDCG	Package	Location in the X array of the first element of array HDCG.
LCIEOP	Package	Location in the X array of the first element of array IEOPNT.
LCLRCH	Package	Location in the X array of the first element of array LRCH.
LCRES	Package	Location in the X array of the first element of array RES.
LENX	Global	Length of the X array in words. This should always be equal to the dimension of X specified in the MAIN program.
MBW	Package	Maximum bandwidth of the coefficient matrix +1.
MXITER	Package	Maximum number of iterations.
NCOL	Global	Number of columns in the grid.
NLAY	Global	Number of layers in the grid.
NROW	Global	Number of rows in the grid.
NSLICE	Package	Number of cells in a slice.

Narrative for Module SOR1RP

Module SOR1RP reads data for the SOR package: the acceleration parameter (ACCL), also called the relaxation factor; the closure criterion (HCLOSE); and the time-step interval (IPRSOR) for printing head change. This module does not have a flow chart. Module SOR1RP performs its functions in the following order:

1. Read the acceleration parameter (ACCL), the closure criterion (HCLOSE), and the interval for printing head change (IPRSOR). If ACCL is zero, substitute a default value of 1.0. If IPRSOR is less than one, set it equal to 999.

2. Print the maximum number of iterations (MXITER), the acceleration parameter (ACCL), the closure criterion (HCLOSE), and the head-change interval (IPRSOR).

3. RETURN.

```
      SUBROUTINE SOR1RP(MXITER,ACCL,HCLOSE,IN,IPRSOR,IOUT)
C
C
C-----VERSION 1005 16MAR1983 SOR1RP
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).
      READ(IN,1) ACCL,HCLOSE,IPRSOR
    1 FORMAT(2F10.0,I10)
      IF(ACCL.EQ.0.) ACCL=1.
      IF(IPRSOR.LT.1) IPRSOR=999
C
C2------PRINT ACCL, HCLOSE, IPRSOR
      WRITE(IOUT,100)
  100 FORMAT(1H0,///57X,'SOLUTION BY SLICE-SUCCESSIVE OVERRELAXATION'
     1/57X,43('-'))
      WRITE(IOUT,115) MXITER
  115 FORMAT(1H0,47X,'MAXIMUM ITERATIONS ALLOWED FOR CLOSURE =',I9)
      WRITE(IOUT,120) ACCL
  120 FORMAT(1H ,63X,'ACCELERATION PARAMETER =',G15.5)
      WRITE(IOUT,125) HCLOSE
  125 FORMAT(1H ,52X,'HEAD CHANGE CRITERION FOR CLOSURE =',E15.5)
      WRITE(IOUT,130) IPRSOR
  130 FORMAT(1H ,52X,'SOR HEAD CHANGE PRINTOUT INTERVAL =',I9)
C
C3------RETURN
      RETURN
      END
```

List of Variables for Module SOR1RP

Variable	Range	Definition
ACCL	Package	Acceleration parameter.
HCLOSE	Package	Closure criterion for the iterative procedure.
IN	Package	Primary unit number from which input for this package will be read.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IPRSOR	Package	Frequency (in time steps) with which the maximum head changes for each iteration will be printed.
MXITER	Package	Maximum number of iterations.

Narrative for Module SOR1AP

Module SOR1AP performs one iteration of the Slice-Successive Over-relaxation (SSOR) algorithm for solving the system of finite-difference equations. The conductances CC, CR, and CV and the composite terms HCOF and RHS (see equation 27) which are calculated by the formulation procedure are combined, row by row (slice by slice), to form the coefficient matrix $\bar{\bar{A}}$ and the vector $\overline{RES}$ on the right hand side of the matrix equation for a single slice. Since the coefficient matrix is symmetric and banded, only main diagonals and NLAY subdiagonals are saved As heads are calculated, they are stored in the array HNEW. The matrix $\bar{\bar{A}}$ and the vector $\overline{RES}$ are passed to a submodule SSOR1B which solves the matrix equation for a vector of approximate head changes which is then multiplied by the relaxation factor to get the final head changes for the iteration. The final head changes are added to the heads from the preceding iteration to get the heads for the current iteration. The final head changes for the iteration are compared to the closure criterion to see if the iterative procedure has closed.

Module SOR1AP performs its functions in the following order:

1. Calculate the number of elements in the compressed coefficient matrix $\bar{\bar{A}}$.

2. Process the slices (rows) one at a time (DO STEPS 3-7).

3. Clear the A array.

4. Assign integers sequentially to the active cells in the slice (remember that finite-difference equations are formulated only for active cells).

5. Calculate the elements in the compressed coefficient matrix $\bar{\bar{A}}$ and the residual vector $\overline{RES}$. Process the cells in the slice one cell at a time.

If the cell is inactive, move on to the next cell. The elements in the main diagonal of the coefficient matrix (the multipliers of $h_{i,j,k}$) will consist of HCOF plus conductances to the six adjacent cells. They will be formed in an accumulator called EE. The contents of EE multiplied by the head from the previous iteration (HNEW) are subtracted from an accumulator (R) to form the residual.

(a) Determine the equation number (NEQ) of the cell If NEQ is zero, the cell is inactive. Move on to the next cell.

(b) Set the accumulators EE and R equal to HCOF and RHS, respectively. Note: HNEW contains head from the last iteration.

(c) If there is a node to the left, subtract the conductance from EE and subtract the conductance times HNEW from R.

(d) If there is a node to the right, subtract the conductance from EE, and subtract the conductance times HNEW from R; and, if the cell to the right is active, move the conductance into the compressed coefficient matrix $\bar{\bar{A}}$. Remember that the coefficient matrix is symmetric so the conductance to the left in step 5(c) did not have to be stored.

(e) If there is a node to the rear, subtract the conductance from EE and subtract the conductance times HNEW from R.

(f) If there is a node to the front, subtract the conductance from EE and subtract the conductance times HNEW from R. Remember that the

form of the SSOR equations does not have terms containing head in adjacent rows on the left hand side.

(g) If there is a node above, subtract the conductance from EE and subtract the conductance times HNEW from R.

(h) If there is a node below, subtract the conductance from EE and subtract the conductance times HNEW from R; and, if the cell below is active, move the conductance into A.

(i) Move EE into the first row of A. The first row in A corresponds to the main diagonal in the "full" coefficient matrix. Subtract EE times HNEW from R and store it in the residual vector.

6. If there are no equations for this slice, go on to the next slice. If there is only one equation, solve it directly and leave the result in the residual vector $\overline{RES}$. If there are two or more equations, call submodule SSOR1B to solve the system of equations for the slice leaving the results (first estimate of head change for this iteration) in the vector $\overline{RES}$.

7. For each cell in the slice, calculate the head for the current iteration.

(a) Multiply the first estimate of head change for this iteration by the relaxation factor to get the final estimate of head change for this iteration.

(b) Add the final head change for this iteration to the head from the last iteration to get the head for this iteration.

(c) If the head change for this cell is greater than that for any other cell, store the head change and the location of the cell.

8. Save the largest head change from this iteration so that it can be printed at the end of the time step.

9. Compare the biggest head change (BIGG) to the closure criterion (HCLOSE). If HCLOSE is greater than BIGG, set the convergence flag (ICNVG) equal to one.

10. If you have not converged and you have not exceeded the maximum number of iterations, RETURN.

11. Print the number of iterations.

12. If convergence failed, or this is the last time step, or this is the time step interval specified by the user, print the maximum head change for each iteration in this time step.

13. RETURN.

Flow Chart for Module SOPIAP

A is a compressed coefficient matrix for a slice. It contains the main diagonal of the full matrix and the NLAY diagonals below it. (NLAY is the number of layers.)

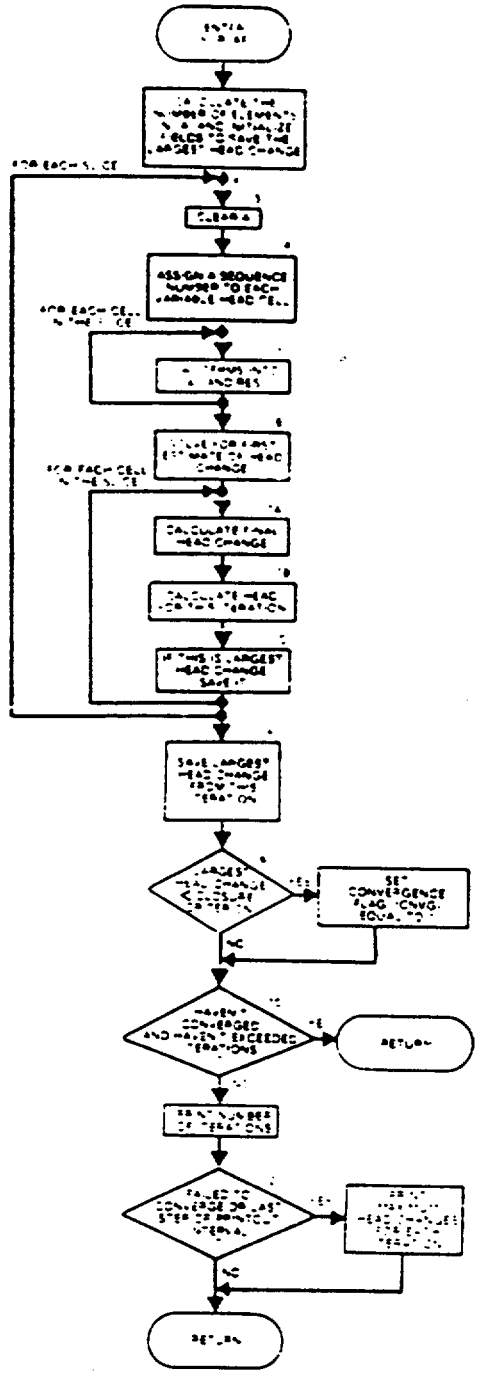
Sequence Number is a number used to identify the internal (variable-head) cells in a slice and also the equations for each internal cell.

RES is a vector containing the residuals for a slice. It consists of RHS (from the basic finite-difference equation) plus all of those terms which are moved to the right hand side to get the equations ready for solution in residual form.

First estimates of head change: these are the head changes calculated by simultaneously solving the equations for a slice. They will be multiplied by the relaxation factor to get final estimates of head change.

Final estimates of head change: these are the head changes calculated by multiplying first estimates by the relaxation factor. They are added to the heads from the previous iteration to get head for the current iteration.

ICNVG is the convergence flag. It is set in the approximator and returned to the MAIN Program so that the iteration loop can be terminated.



```
      SUBROUTINE SOR1AP(HNEW,IBOUND,CR,CC,CV,HCOF,RHS,A,RES,IEQPNT,
     1      HDCG,LRCH,KITER,HCLOSE,ACCL,ICNVG,KSTP,KPER,
     2      IP-SOR,MXITER,NSTP,NCOL,NROW,NLAY,NSLICE,MBW,IOUT)
C-----VERSION 0936 09MAY1983 SOR1AP
C     ******************************************************************
C     SOLUTION BY SLICE-SUCCESSIVE OVERRELAXATION -- 1 ITERATION
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DOUBLE PRECISION HNEW,DIFF,DP,EE,R,HCFHNW,HHCOF
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
      BIG=0.
      ABSBIG=0.
      IB=0
      JB=0
      KB=0
C
C2------PROCESS EACH SLICE
      DO 500 I=1,NROW
C
C3------CLEAR A
      DO 110 J=1,NSLICE
      DO 110 K=1,MBW
  110 A(K,J)=0.
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=0.
      R=RHS(J,I,K)
C
C5C-----IF NODE TO LEFT SUBTRACT TERMS FROM EE AND R
      IF(J.EQ.1) GO TO 120
```

```
      DP=CR(J-1,I,K)
      R=R-DP*HNEW(J-1,I,K)
      EE=EE-DP
C
C5D-----IF NODE TO RIGHT SUBTRACT TERMS FROM EE & R, MOVE COND TO A
  120 IF(J.EQ.NCOL) GO TO 125
      SP=CR(J,I,K)
      DP=SP
      R=R-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)
      A(1,NEQ)=EE+HHCOF
      HNW=HNEW(J,I,K)
      HCFHNW=HNW*HCOF(J,I,K)
      RES(NEQ)=R-EE*HNEW(J,I,K)-HCFHNW
  300 CONTINUE
C
C6------IF NO EQUATIONS GO TO NEXT SLICE, IF ONE EQUATION SOLVE
C6------DIRECTLY, IF 2 EQUATIONS CALL SSOR1B 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 SSOR1B(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 EXCEEDED ITERATIONS THEN RETURN
      IF(ICNVG.EQ.0 .AND. KITER.NE.MXITER) RETURN
      IF(KSTP.EQ.1) WRITE(IOUT,600)
  600 FORMAT(1H0)
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(1H0,'MAXIMUM HEAD CHANGE FOR EACH ITERATION:'/
     1    1H0,4('     HEAD CHANGE  LAYER,ROW,COL')/1X,120('-'))
      WRITE (IOUT,10) (HDCG(J),(LRCH(I,J),I=1,3),J=1,KITER)
   10 FORMAT((1X,4(4X,G12.4,' (',I3,',',I3,',',I3,')')))
      WRITE(IOUT,11)
   11 FORMAT(1H0)
C
C13-----RETURN
  700 RETURN
C
      END
```

List of Variables for Module SORIAP

Variable	Range	Definition
A	Package	DIMENSION (MBW,NSLICE), Compressed coefficient matrix for a slice.
ABSDIG	Module	Largest ABSDH for this iteration.
ABSDH	Module	Absolute value of head change in a cell for the current iteration.
ACCL	Package	Acceleration parameter.
CC	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the column direction. CC(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K).
CR	Global	DIMENSION (NCOL,NROW,NLAY), Conductance in the row direction. CR(J,I,K) contains conductance between nodes (J,I,K) and (J,I+1,K).
CV	Global	DIMENSION (NCOL,NROW,NLAY-1), Conductance in the vertical direction. CV(J,I,K) contains conductance between nodes (J,I,K) and (J,I,K+1).
DH	Module	Change in head in a cell during one iteration.
DIFF	Module	Double-precision change in head (DH).
DP	Module	Double-precision temporary field.
EE	Module	Main diagonal term in the finite-difference equation.
HCLOSE	Package	Closure criterion for the iterative procedure.
HCOF	Global	DIMENSION (NCOL,NROW,NLAY), Coefficient of head in the cell (J,I,K) in the finite-difference equation.
HDCG	Package	DIMENSION (MXITER), Maximum head change for each iteration.
HNEW	Global	DIMENSION (NCOL,NROW,NLAY), Most recent estimate of head in each cell. HNEW changes at each iteration.
I	Module	Index for rows.
IB	Module	Row number of the cell containing the largest head change.
IBOUND	Global	DIMENSION (NCOL,NROW,NLAY), Status of each cell. < 0, constant-head cell = 0, inactive cell > 0, variable-head cell
ICNVG	Global	Flag is set equal to one when the iteration procedure has converged.
IEQPNT	Global	DIMENSION (NLAY,NCOL), Sequence numbers for variable-head cells in a slice.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IPRSOR	Package	Frequency (in time steps) with which the maximum head changes for each iteration will be printed.
J	Module	Index for columns.
JB	Module	Column number of the cell containing the largest head change.
K	Module	Index for layers.
KB	Module	Layer number of the cell containing the largest head change.
KITER	Global	Iteration counter. Reset at the start of each time step.
KPER	Global	Stress period counter.

List of Variables for Module SOR1AP (Continued)

Variable	Range	Definition
KSTP	Global	Time step counter. Reset at the start of each stress period.
LRCH	Package	DIMENSION (MXITER), Layer, row, and column of the cell containing the maximum head change (HDCG) for each iteration.
MBW	Package	Maximum bandwidth of the coefficient matrix +1.
MXITER	Package	Maximum number of iterations.
NA	Package	Number of elements in the compressed coefficient matrix (A).
NCOL	Global	Number of columns in the grid.
NEQ	Module	Index for equations (variable-head cells) in a slice.
NEQT	Package	Number of equations (variable-head cells) in a slice.
NLAY	Global	Number of layers in the grid.
NROW	Global	Number of rows in the grid.
NSLICE	Package	Number of cells in a slice.
NSTP	Global	Number of time steps in the current stress period.
NXT	Module	Sequence number of the cell to the right.
R	Module	Right hand side of the finite-difference equation as modified (terms for the adjacent rows moved to the right) for solution by the slice-successive overrelaxation.
RES	Package	DIMENSION (NSLICE), Residual.
RHS	Global	DIMENSION (NCOL,NROW,NLAY), Right hand side of the finite-difference equation. RHS is an accumulation of terms from several different packages.
SP	Module	Single-precision temporary field.

Narrative for Module SSOR1B

Module SSOR1B uses Gaussian elimination to solve a matrix equation of the form $\bar{\bar{A}}\bar{X} = \bar{B}$. The matrix $\bar{\bar{A}}$ is symmetric and banded with rank "N." It is stored in the compressed format shown in figure 55 and uses a single subscript. The vector $\bar{X}$ as it is calculated is stored in the space reserved for vector $\bar{B}$.

The indices used in the module flow chart are those for standard matrix organization. The indices actually used in the program are based on the compressed format and a single index. Module SSOR1B performs its functions in the following order:

1. Work through the first N-1 rows using each one, in sequence, as the pivot row (row I).

2. Calculate the inverse of the main diagonal element---$a_{I,I}$. (The index (ID) points to $a_{I,I}$ which is the first element of column I in the compressed matrix $\bar{\bar{A}}$.)

3. Modify each of the rows after row I so that the terms corresponding to the pivot term are eliminated. Since the coefficient matrix is banded, there are only MBW-1 equations (where MBW is the maximum half-bandwidth plus one) where the term to be eliminated is not already equal to zero. The rows are indexed by "L." (The equation corresponding to row "L" is referred to as equation "L.")

4. Calculate the coefficient C which when multiplied by the pivot equation and subtracted from equation L will eliminate a term in equation L.

(The index LB points to a coefficient in the pivot equation which, because of symmetry, is equal to the coefficient to be eliminated.)

5. Calculate the new coefficients in equation L for each of the terms to the right of the coefficient that is being eliminated. Because the matrix is banded, there are only MBW-1 nonzero terms to the right of the pivot. Therefore, at most, MBW-1 coefficients have to be calculated.

6. Subtract C times a coefficient in the pivot equation from the corresponding coefficient in equation L.

7. Subtract C times the right side of the pivot equation from the right side of equation L. (The index LB points to the coefficient in equation L which must be calculated.)

8. Solve equation N for X(N) putting the result in B(N).

9. Work backward from equation N-1 solving each equation (equation L) for X(L).

10. Set the accumulator "SUM" equal to zero.

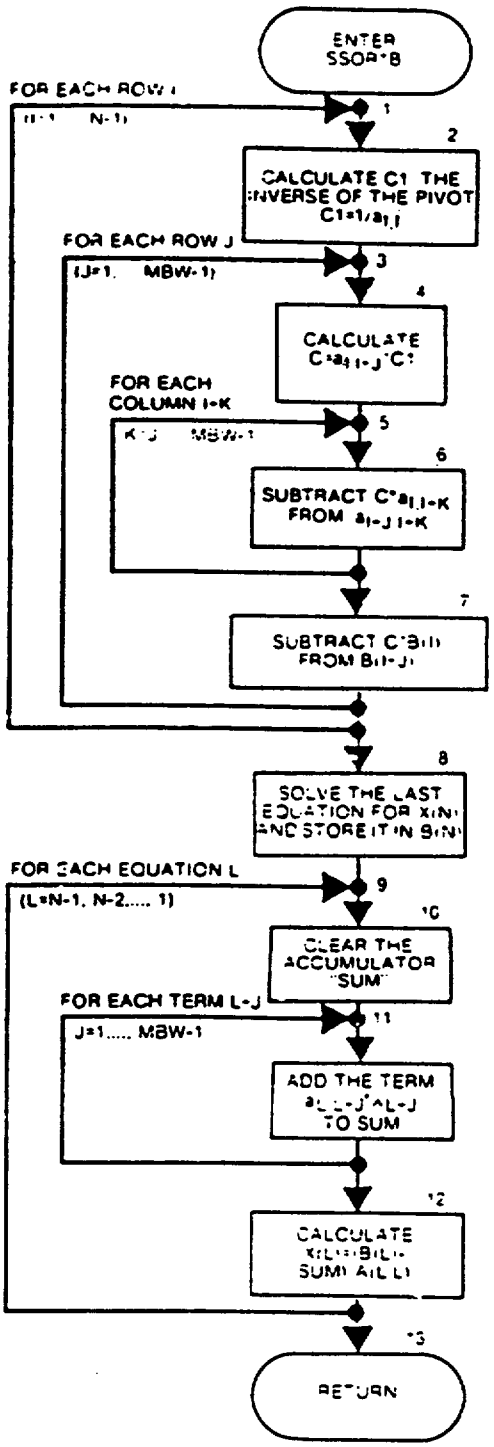
11. Multiply the coefficient to the right of the main diagonal (in equation L) by the corresponding value of X and add it to the sum.

12. Calculate the value of X(L) and store it in B(L).

13. RETURN.

Flow Chart for Module SSOR1B

MBW is the maximum half bandwidth of the coefficient matrix plus one.



```
      SUBROUTINE SSOR1B(A,B,N,NA,MBW)
C
C
C-----VERSION 1359 31MAR1983 SSOR1B
C     ******************************************************************
C     SOLVE A SYMMETRIC SET OF EQUATIONS
C        A IS COEFFICIENT MATRIX IN COMPRESSED FORM
C        B IS RIGHT HAND SIDE AND IS REPLACED BY SOLUTION
C        N IS NUMBER OF EQUATIONS TO BE SOLVED
C        MBW IS BANDWIDTH OF A
C        NA IS ONE-DIMENSION SIZE OF A
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION A(NA),B(N)
C     ------------------------------------------------------------------
C
      NM1=N-1
      MBW1=MBW-1
      ID=1-MBW
C
C1------SEQUENTIALLY USE EACH OF THE FIRST N-1 ROWS AS
C1------THE PIVOT ROW.
      DO 20 I=1,NM1
C
C2------CALCULATE THE INVERSE OF THE PIVOT.
      ID=ID+MBW
      C1=1./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=0.0
      L=N-I
      MBW1M=MIN0(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
```

List of Variables for Module SSOR1B

Variable	Range	Definition
A	Package	DIMENSION (MBW,NSLICE), Compressed coefficient matrix for a slice.
B	Package	DIMENSION (N), Right-hand-side vector.
C	Module	Factor needed to eliminate a term in the target row.
C1	Module	Inverse of pivot.
I	Module	Index for rows in the SSOR matrix (not the grid).
IB	Module	Index for elements to the right of the pivot.
ID	Module	Index of pivots.
J	Module	Index for columrs.
K	Module	Index for columns.
L	Module	Index for equations.
LB	Module	Index for elements in the target row to the right of the main diagonal.
LD	Module	Index of the main diagonal elements in the target rows.
MBW	Package	Maximum bandwidth of the coefficient matrix +1.
MBW1	Module	Maximum bandwidth of the coefficient matrix.
MBW1M	Module	Maximum possible number of the nonzero elements to the right of the main diagonal.
N	Package	Number of equations to be solved.
NA	Package	One-dimension size of compressed matrix "A."
NM1	Module	N-1.
SUM	Module	In back substitution--in equation L, sum of terms to the right of the main diagonal term (L,L).

CHAPTER 14

UTILITY MODULES

Utility modules are those submodules which perform general tasks common to several different packages. The name of a utility module is a "U" followed by a five-character mnemonic. There are eight utility modules:

UBUDSV	Writes an unformatted record consisting of an array with one real number for each cell in the grid.
ULASAV	Writes an unformatted record consisting of an array with one real number for each cell in a layer.
ULAPRS and ULAPRW	Prints one two-dimensional array which contains one real number for each cell in a layer. ULAPRS prints, in strip form, the first N columns (where N is the number of values that can fit on one print line) of each row and then the next N columns, etc., until all columns of each row are printed (fig. 56). ULAPRW prints, in wrap form, all of row 1, all of row 2, and all of row 3, etc. The format for printing arrays is shown in table 2.
UCOLNO	Prints column numbers at the top of each page of data printed by ULAPRS and ULAPRW.
U2DREL	Reads a two-dimensional array of real numbers.
U2DINT	Reads a two-dimensional array of integers.
U1DREL	Reads a one-dimensional array of real numbers.

	1	2	3	4	5	6	7	8	9	10
	11	12	13	14	15	16	17			
1	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79			
2	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79			
3	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79			
4	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79			
5	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79			
6	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79			
7	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79			

WRAP FORM

	1	2	3	4	5	6	7	8	9	10
1	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
2	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
3	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
4	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
5	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
6	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
7	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79

	11	12	13	14	15	16	17
1	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
2	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
3	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
4	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
5	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
6	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79
7	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79	1325.79

STRIP FORM

Figure 56.—Illustration of wrap and strip forms of printed output for a layer containing 7 rows and 17 columns.

Table 2.--Print-format codes for utility modules ULAPRS and ULAPRW.

IPRN	FORMAT
1	11G10.3
2	9G13.6
3	15F7.1
4	15F7.2
5	15F7.3
6	15F7.4
7	20F5.0
8	20F5.1
9	20F5.2
10	20F5.3
11	20F5.4
12	10G11.4

Utility Input

The utility module only reads from channels specified by calling programs. The real two-dimensional array reader (U2DREL), the integer two-dimensional array reader (U2DINT), and the real one-dimensional array reader (U1DREL) read one array-control record and, optionally, a data array in a format specified on the array-control record.

FOR REAL ARRAY READER (U2DREL or U1DREL)

Data:	LOCAT	CNSTNT	FMTIN	IPRN
Format:	I10	F10.0	5A4	I10

FOR INTEGER ARRAY READER (U2DINT)

Data:	LOCAT	ICONST	FMTIN	IPRN
Format:	I10	I10	5A4	I10

Explanation of Fields Used in Input Instructions

LOCAT--indicates the location of the data which will be put in the array.

If LOCAT < 0, the sign is reversed to give the unit number from which an unformatted record will be read.

If LOCAT = 0, every element in the array will be set equal to the value CNSTNT/ICONST.

If LOCAT > 0, it is the unit number from which data values will be read in the format specified in the third field of the array-control record (FMTIN).

CNSTNT/ICONST--is a constant. Its use depends on the value of LOCAT.

If LOCAT = 0, every element in the array is set equal to CNSTNT/ICONST.

If LOCAT ≠ 0, and if CNSTNT/ICONST ≠ 0, every element in the array is multiplied by CNSTNT/ICONST.

FMTIN--is the format of records containing the array values. It is used only if the first field in the array-control record (LOCAT) contains a positive number. The format must be enclosed in parentheses; for example, (15F5.0) for real data and (15I5) for integer data.

IPRN--is a flag indicating that the array being read should be printed and a code for indicating the format that should be used. It is used only if LOCAT is not equal to zero. The format codes are different for each

of the three modules. IPRN is set to zero when the specified value exceeds those defined in the chart below. If IPRN is less than zero, the array will not be printed.

IPRN	U2DREL	U2DINT	U1DREL
0	10G11.4	10I11	10G12.5
1	11G10.3	6011	
2	9G13.6	4012	
3	15F7.1	3013	
4	15F7.2	2514	
5	15F7.3	2015	
6	15F7.4		
7	20F5.0		
8	20F5.1		
9	20F5.2		
10	20F5.3		
11	20F5.4		
12	10G11.4		

Narrative for Module UBUDSV

Utility module UBUDSV writes an unformatted record consisting of an array dimensioned (NCOL, NROW, NLAY). The record containing the array is preceded by an unformatted record containing identifying information. The identifying information consists of:

KSTP	current time step	integer	1 word
KPER	current stress period	integer	1 word
TEXT	label	character string	4 words
NCOL	number of columns	integer	1 word
NROW	number of rows	integer	1 word
NLAY	number of layers	integer	1 word

Documentation of this module consists only of comments in the program and a list of variables.

```
      SUBROUTINE UBUDSV(KSTP,KPER,TEXT,IBDCHN,BUFF,NCOL,NROW,NLAY,IOUT)
C
C
C-----VERSION 1305 28DEC1983 UBUDSV
C     ***************************************************************
C     RECORD CELL-BY-CELL FLOW TERMS FOR ONE COMPONENT OF FLOW.
C     ***************************************************************
C
C        SPECIFICATIONS:
C     ---------------------------------------------------------------
      DIMENSION TEXT(4),BUFF(NCOL,NROW,NLAY)
C     ---------------------------------------------------------------
C
C1------WRITE AN UNFORMATTED RECORD CONTAINING IDENTIFYING
C1------INFORMATION.
      WRITE(IOUT,1) TEXT,IBDCHN,KSTP,KPER
    1 FORMAT(1X,'"',4A4,'" BUDGET VALUES WILL BE SAVED ON UNIT',I3,
     1      ' AT END OF TIME STEP',I3,', STRESS PERIOD',I3)
C
      WRITE(IBDCHN) KSTP,KPER,TEXT,NCOL,NROW,NLAY
C
C2------WRITE AN UNFORMATTED RECORD CONTAINING VALUES FOR
C2------EACH CELL IN THE GRID. THE ARRAY IS DIMENSIONED
C2------(NCOL,NROW,NLAY)
      WRITE(IBDCHN) BUFF
C
C3------RETURN
      RETURN
      END
```

List of Variables for Module UBUDSV

Variation	Range	Definition
BUFF	Global	DIMENSION (NCOL,NROW,NLAY), Buffer used to accumulate information before printing or recording it.
IBDCHN	Module	Unit number on which the array will be recorded.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. Reset at the start of each stress period.
NCOL	Global	Number of columns in the grid.
NLAY	Global	Number of layers in the grid.
NROW	Global	Number of rows in the grid.
TEXT	Module	Label to be printed or recorded with the array data.

Narrative for Module ULASAV

Utility module ULASAV writes an unformatted record consisting of an array dimensioned (NCOL, NROW). The record containing the array is preceded by an unformatted record containing identifying information. The identifying information consists of:

KSTP	current time step	integer	1 word
KPER	current stress period	integer	1 word
PERTIM	elapsed time in the current stress period	real	1 word
TOTIM	elapsed time in the simulation	real	1 word
TEXT	label	character string	4 words
NCOL	number of columns	integer	1 word
NROW	number of rows	integer	1 word
ILAY	layer number	integer	1 word

```
      SUBROUTINE ULASAV(BUF,TEXT,KSTP,KPER,PERTIM,TOTIM,NCOL,
     1                  NROW,ILAY,ICHN)
C
C-----VERSION 1445 20APR1983 ULASAV
C     ***************************************************************
C     SAVE 1 LAYER ARRAY ON DISK
C     ***************************************************************
C
C        SPECIFICATIONS:
C     ---------------------------------------------------------------
      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
```

List of Variables for Module ULASAV

Variable	Range	Definition
BUF	Module	Buffer containing data to be printed or recorded.
IC	Module	Index for columns.
ICHN	Module	Unit number on which the array is to be recorded.
ILAY	Module	Layer number.
IR	Module	Index for rows.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. Reset at the start of each stress period.
NCOL	Global	Number of columns in the grid.
NROW	Global	Number of rows in the grid.
PERTIM	Package	Elapsed time during the current stress period.
TEXT	Module	Label to be printed or recorded with the array data.
TOTIM	Package	Elapsed time in the simulation.

Narrative for Module ULAPRS

Module ULAPRS prints a two-dimensional array in strip form (fig. 56) using one of twelve FORTRAN formats. Module ULAPRS performs its tasks in the following order:

1. Get the format code (IP). If it is less than 1 or greater than 12, set it equal to 12 (the default).

2. Use the format code (IP) to determine the number of values (NCAP) to be printed on one line.

3. Calculate the number of spaces used for each value (NCPF) and the number of strips (NSTRIP). Initialize the fields to store the first column (J1) and the last column (J2) for each strip.

4. Loop through the strips (DO STEPS 5-8).

5. Calculate the first (J1) and last (J2) column for this strip.

6. Print a title on each strip.

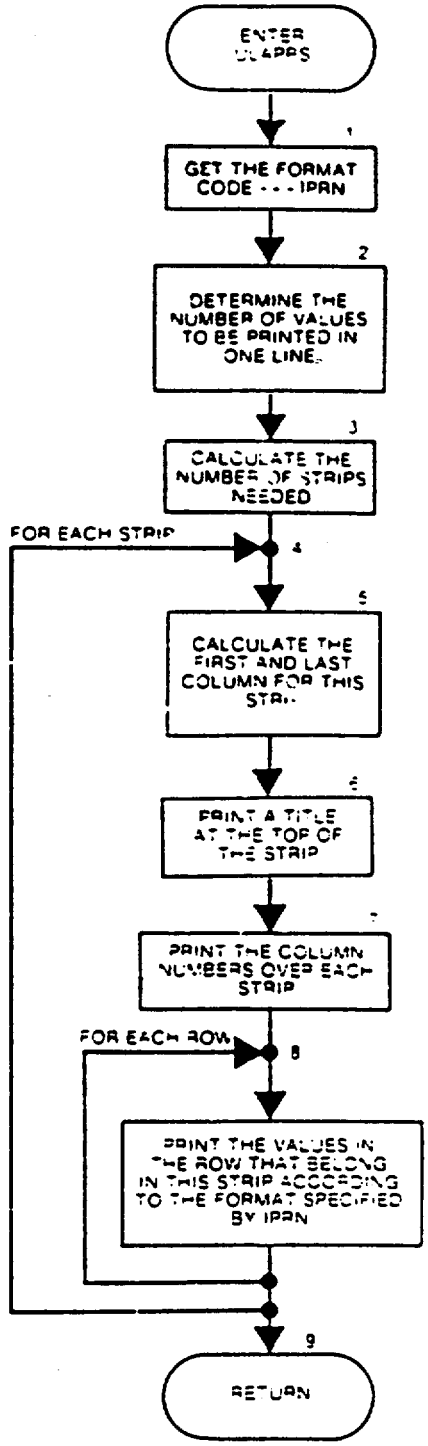
7. Call module UCOLNO to print the column numbers above each strip.

8. Loop through the rows printing columns J1 through J2 using the appropriate format (IP).

9. RETURN.

Flow Chart for Module ULAPRS

IPRN is a code indicating the format to be used in printing array values. If it is not between 1 and 12, it is set equal to 12.



```
      SUBROUTINE ULAPRS(BUF,TEXT,KSTP,KPER,NCOL,NROW,ILAY,IPRN,IOUT)
C
C
C-----VERSION 1448 20APR1983 ULAPRS
C     ******************************************************************
C     PRINT A 1 LAYER ARRAY IN STRIPS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION BUF(NCOL,NROW),TEXT(4)
C     ------------------------------------------------------------------
C
C1------MAKE SURE THE FORMAT CODE (IP OR IPRN) IS BETWEEN 1
C1------AND 12.
      IP=IPRN
      IF(IP.LT.1 .OR. IP.GT.12) IP=12
C
C2------DETERMINE THE NUMBER OF VALUES (NCAP) PRINTED ON ONE LINE.
      IF(IP.EQ.1) NCAP=11
      IF(IP.EQ.2) NCAP=9
      IF(IP.GT.2 .AND. IP.LT.7) NCAP=15
      IF(IP.GT.6 .AND. IP.LT.12) NCAP=20
      IF(IP.EQ.12) NCAP=10
C
C3------CALCULATE THE NUMBER OF STRIPS (NSTRIP).
      NCPF=129/NCAP
      ISP=0
      IF(NCAP.GT.12) ISP=3
      NSTRIP=(NCOL-1)/NCAP + 1
      J1=1-NCAP
      J2=0
C
C4------LOOP THROUGH THE STRIPS.
      DO 2000 N=1,NSTRIP
C
C5------CALCULATE THE FIRST(J1) & THE LAST(J2) COLUMNS FOR THIS STRIP
      J1=J1+NCAP
      J2=J2+NCAP
      IF(J2.GT.NCOL) J2=NCOL
C
C6-------PRINT TITLE ON EACH STRIP
      WRITE(IOUT,1) TEXT,ILAY,KSTP,KPER
    1 FORMAT(1H1,10X,4A4,' IN LAYER',I3,' AT END OF TIME STEP',I3,
     1      ' IN STRESS PERIOD',I3/11X,71('-'))
C
C7------PRINT COLUMN NUMBERS ABOVE THE STRIP
      CALL UCOLNO(J1,J2,ISP,NCAP,NCPF,IOUT)
C
C8------LOOP THROUGH THE ROWS PRINTING COLS J1 THRU J2 WITH FORMAT IP
      DO 1000 I=1,NROW
      GO TO(10,20,30,40,50,60,70,80,90,100,110,120), IP
C
C------------FORMAT 10G10.3
   10 WRITE(IOUT,11) I,(BUF(J,I),J=J1,J2)
   11 FORMAT(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,I3,2X,1PG11.4,9(1X,G11.4))
C
 1000 CONTINUE
 2000 CONTINUE
C
C9------RETURN
      RETURN
      END
```

List of Variables for Module ULAPRS

Variable	Range	Definition
BUF	Module	Buffer containing data to be printed or recorded.
I	Module	Index for rows.
ILAY	Module	Layer number.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IP	Module	Format code.
IPRN	Module	Code for the format to be used when printing arrays.
ISP	Module	Number of spaces.
J	Module	Index for columns.
J1	Module	First column in a strip.
J2	Module	Last column in a strip.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. Reset at the start of each stress period.
N	Module	Index for strips.
NCAP	Module	Number of columns on a line.
NCOL	Global	Number of columns in the grid.
NCPF	Module	Number of columns per field.
NROW	Global	Number of rows in the grid.
NSTRIP	Module	Number of strips.
TEXT	Module	Label to be printed or recorded with the array data.

Narrative for Module ULAPRW

Module ULAPRW prints a two-dimensional array in wrap form (fig. 56) using one of twelve FORTRAN formats. Module ULAPRW performs its tasks in the following order:

1. Print a header.

2. Set the format code (IP). If it is less than 1 or greater than 12, set it equal to 12 (the default).

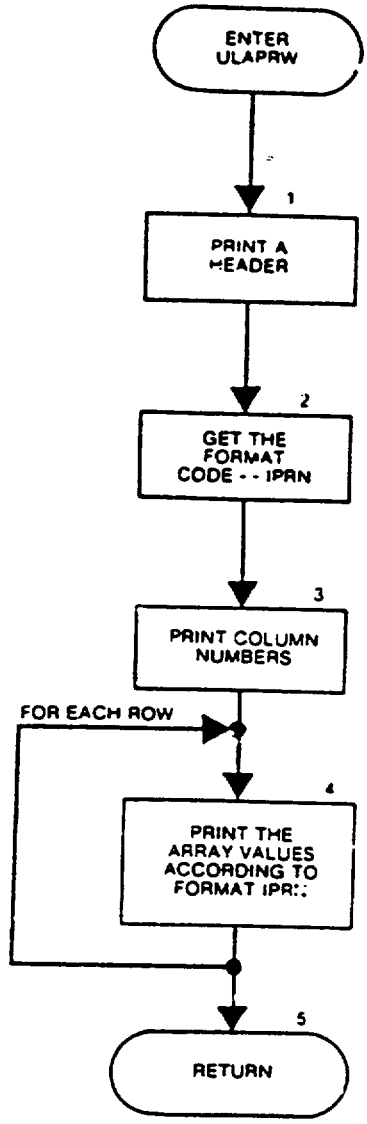
3. Call the module UCOLNO to print column numbers.

4. Loop through the rows printing each one in its entirety using the appropriate format code.

5. RETURN.

Flow Chart for Module ULAPRW

IPRN is a code indicating the format to be used in printing array values. If it is not between 1 and 12, it is set equal to 12.



```
      SUBROUTINE ULAPRW(BUF,TEXT,KSTP,KPER,NCOL,NROW,ILAY,IPRN,IOUT)
C
C
C-----VERSION 1245 04MAY1983 ULAPRW
C     ******************************************************************
C     PRINT 1 LAYER ARRAY
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION BUF(NCOL,NROW),TEXT(4)
C     ------------------------------------------------------------------
C
C1------PRINT A HEADER
      IF(ILAY.LE.0) GO TO 5
      WRITE(IOUT,1) TEXT,ILAY,KSTP,KPER
    1 FORMAT(1H1,10X,4A4,' IN LAYER',I3,' AT END OF TIME STEP',I3,
     1      ' IN STRESS PERIOD',I3/11X,71('-'))
C
C2------MAKE SURE THE FORMAT CODE (IP OR IPRN) IS
C2------BETWEEN 1 AND 12.
    5 IP=IPRN
      IF(IP.LT.1 .OR. IP.GT.12) 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)
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), IP
C
C------------ FORMAT 11G10.3
   10 WRITE(IOUT,11) I,(BUF(J,I),J=1,NCOL)
   11 FORMAT(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,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(1H0,I3,2X,1PG11.4,9(1X,G11.4)/(5X,10(1X,G11.4)))
C
 1000 CONTINUE
C
C5------RETURN
      RETURN
      END
```

List of Variables for Module ULAPRW

Variable	Range	Definition
BUF	Module	Buffer containing data to be printed or recorded.
I	Module	Index for rows.
ILAY	Module	Layer number.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IP	Module	Format code.
IPRN	Module	Code for the format to be used when printing arrays.
J	Module	Index for columns.
KPER	Global	Stress period counter.
KSTP	Global	Time step counter. Reset at the start of each stress period.
NCOL	Global	Number of columns in the grid.
NROW	Global	Number of rows in the grid.
TEXT	Module	Label to be printed or recorded with the array data.

Narrative for Module UCOLNO

Module UCOLNO prints column numbers at the top of a page when arrays of numbers are printed. It performs its functions in the following order:

1. Calculate the number of columns to be printed (NLBL), the width of a line (NTOT), and the number of lines needed to print all of the column numbers (NWRAP). Initialize the fields J1 and J2 which contain the first and last column number on each print line.

2. Build and print each line (DO STEPS 3-6).

3. Clear the line buffer (BF) in which the line is built.

4. Determine the first (J1) and last (J2) column number for the current line.

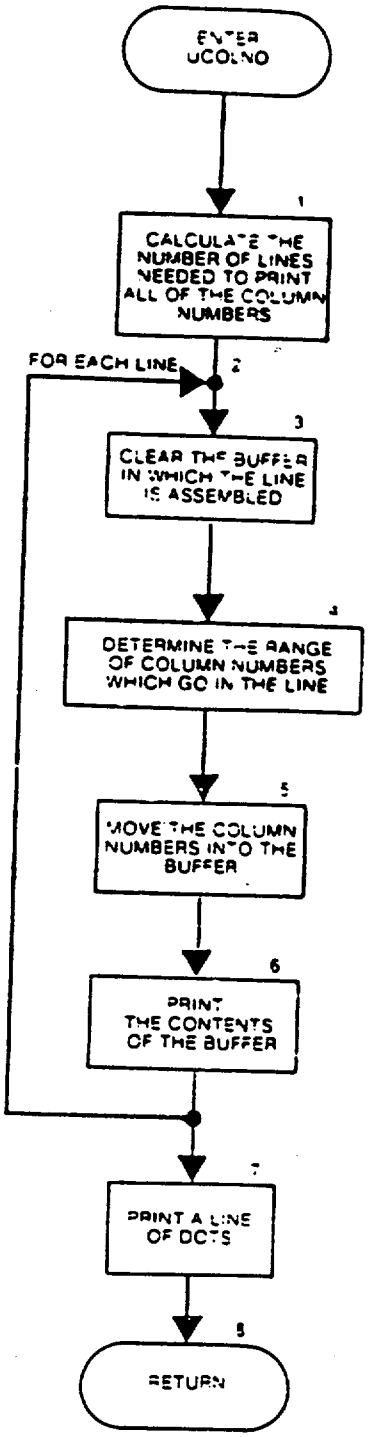
5. Put the column numbers in the line buffer. They are selected from the array DG. The indices I1, I2, and I3 point to the units digit, tens digit, and hundreds digit, respectively.

6. Print the line.

7. Print a line of dots.

8. RETURN.

Flow Chart for Module UCOLNO



```
      SUBROUTINE UCOLNO(NLBL1,NLBL2,NSPACE,NCPL,NDIG,IOUT)
C
C
C-----VERSION 1446 20APR1983 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     ------------------------------------------------------------------
      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        4H0    ,4H1    ,4H2    ,4H3    ,4H4    ,4H5    ,4H6    ,
     2        4H7    ,4H8    ,4H9    /
      DATA DOT,SPACE/4H.    ,4H    /
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+5
      IF(NTOT.GT.130) NTOT=130
      WRITE(IOUT,51) (DOT,I=1,NTOT)
   51 FORMAT(1X,130A1)
C
C8------RETURN
      RETURN
      END
```

List of Variables for Module UCOLNO

Variable	Range	Definition
BF	Module	DIMENSION (130), Buffer in which a line is assembled.
DG	Module	DIMENSION (10), Digits 0 through 9.
DOT	Module	Field containing a period.
I	Module	Index for BF.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
I1	Module	Index for DG (units digit).
I2	Module	Index for DG (tens digit).
I3	Module	Index for DG (hundreds digit).
J	Module	Index for column numbers.
J1	Module	First column number on the current line.
J2	Module	Last column number on the current line.
N	Module	Number of column numbers.
NBF	Module	Index for BF.
NCPL	Module	Number of column numbers per line.
NDIG	Module	Number of characters in each column number field.
NLBL	Module	Number of column numbers to be printed.
NLBL1	Module	Start column number.
NLBL2	Module	Stop column number.
NSPACE	Module	Number of blank spaces at start of line.
NTOT	Module	Total number of characters on a line.
NWRAP	Module	Number of lines needed in wrap format.
SPACE	Module	Field containing blanks.

Narrative for Module U2DREL

Module U2DREL reads values for a two-dimensional real array. First it reads an "array-control record." Then, based on the contents of the array-control record, it may read array values. The array-control record contains four fields: location (LOCAT), constant (CNSTNT), format (FMTIN), and printout indicator (IPRN). The LOCAT field determines where array values will come from. If LOCAT is positive, it is the unit number from which array values will be read in the format specified in FMTIN. If LOCAT is negative, the sign is reversed to give the unit number from which an unformatted record containing the array values will be read. (Before the array record is read, a record will be read and ignored. Thus output from the module ULASAV can be read.) If LOCAT is zero, all of the array values will be set equal to CNSTNT. When LOCAT is not zero and CNSTNT is not zero, the array values will be multiplied by the value of CNSTNT. The field IPRN contains a code number for a FORTRAN format to be used when printing the array.

Module U2DREL performs its tasks in the following order:

1. Read the array-control record (LOCAT, CNSTNT, FMTIN, and IPRN).

2. Use LOCAT to determine where the array values are coming from. GO TO STEPS 3, 4, OR 5.

3. If LOCAT equals zero, set all array values equal to CNSTNT, print a message to that effect, and RETURN.

4. If LOCAT is greater than zero, read array values according to the format in FMTIN. GO TO STEP 6.

5. If LOCAT is less than zero, read an unformatted dummy record and then read an unformatted record containing the array values. GO TO STEP 6.

6. If CNSTNT is not equal to zero, multiply array values by CNSTNT.

7. If IPRN is greater than or equal to zero, call utility module ULAPRW to print the array values using IPRN as the format code.

8. RETURN.

Flow Chart for Module U2DREL

Array Control Record controls the input of array values. It contains four fields: LOCAT, CNSTNT, FMTIN, and IPRN.

LOCAT is a code showing where array values will come from.

If LOCAT < 0, array values will be read from an unformatted record from a unit number equal to -LOCAT.

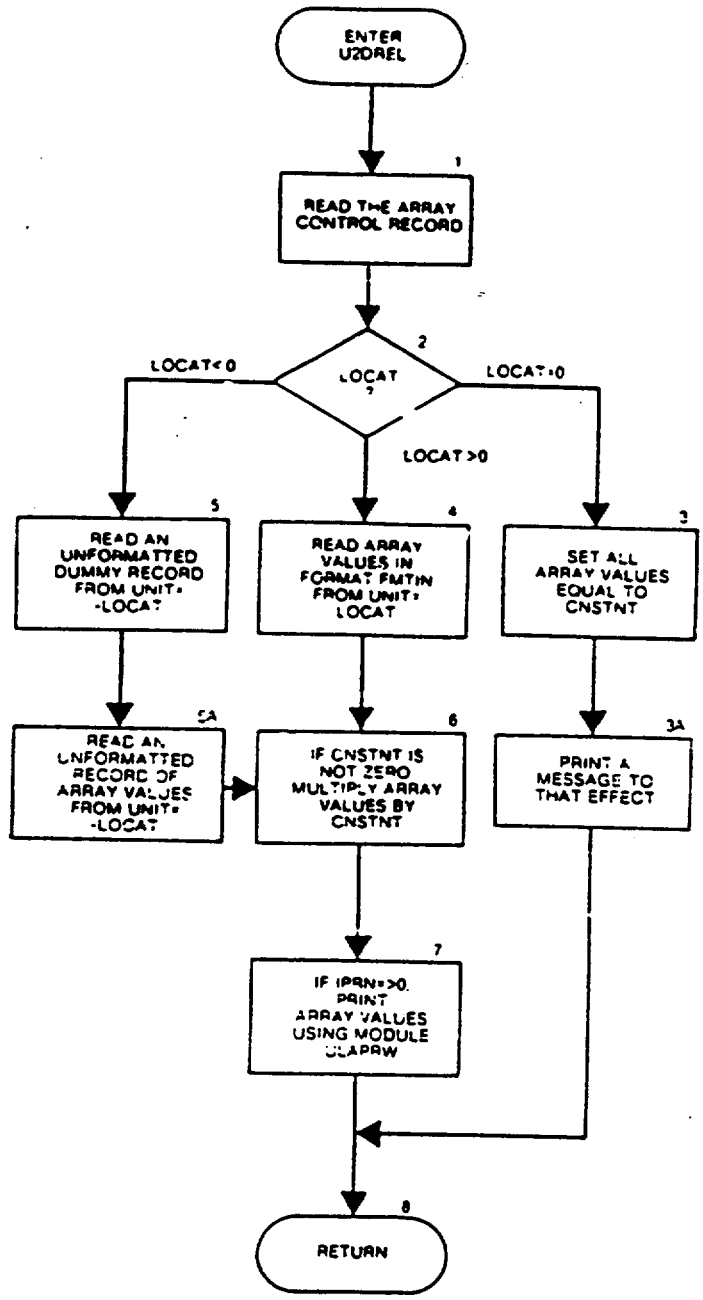
If LOCAT = 0, array values will be set equal to CNSTNT.

If LOCAT > 0, array values will be read from the unit number equal to LOCAT in the format specified in FMTIN.

CNSTNT is the constant to which all array values are set if LOCAT is equal to zero, and it is the constant by which all array values are multiplied if LOCAT is not equal to zero.

FMTIN is the format in which array values are read if LOCAT is greater than zero.

IPRN is a code showing the format to be used if array values are to be printed.



```
      SUBROUTINE U2DREL(A,ANAME,II,JJ,K,IN,IOUT)
C
C
C-----VERSION 1430 2MAR1983 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 UNLESS K IS 0)
C       IN IS INPUT UNIT
C       IOUT IS OUTPUT UNIT
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION A(JJ,II),ANAME(6),FMTIN(5)
C     ------------------------------------------------------------------
C
C1------READ ARRAY CONTROL RECORD.
      READ (IN,1) LOCAT,CNSTNT,FMTIN,IPRN
    1 FORMAT(I10,F10.0,5A4,I10)
C
C2------USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
      IF(LOCAT) 200,50,90
C
C3------IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN
   50 DO 80 I=1,II
      DO 80 J=1,JJ
   80 A(J,I)=CNSTNT
      IF(K.GT.0) WRITE(IOUT,2) ANAME,CNSTNT,K
    2 FORMAT(1H0,52X,6A4,' =',G15.7,' FOR LAYER',I3)
      IF(K.LE.0) WRITE(IOUT,3) ANAME,CNSTNT
    3 FORMAT(1H0,52X,6A4,' =',G15.7)
      RETURN
C
C4------IF LOCAT>0 THEN READ FORMATTED RECORDS USING FORMAT FMTIN.
   90 IF(K.GT.0) WRITE(IOUT,4) ANAME,K,LOCAT,FMTIN
    4 FORMAT(1H0,///30X,6A4,' FOR LAYER',I3,' WILL BE READ ON UNIT',
     1       I3,' USING FORMAT: ',5A4/30X,96('-'))
      IF(K.LE.0) WRITE(IOUT,5) ANAME,LOCAT,FMTIN
    5 FORMAT(1H0,///30X,6A4,' WILL BE READ ON UNIT',
     1       I3,' USING FORMAT: ',5A4/30X,83('-'))
      DO 100 I=1,II
      READ (LOCAT,FMTIN) (A(J,I),J=1,JJ)
  100 CONTINUE
      GO TO 300
C
C5------LOCAT<0 THEN READ UNFORMATTED RECORD CONTAINING ARRAY VALUES
  200 LOCAT=-LOCAT
      IF(K.GT.0) WRITE(IOUT,201) ANAME,K,LOCAT
  201 FORMAT(1H0,///30X,6A4,', LAYER',I3,
     1     ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,73('-'))
      IF(K.LE.0) WRITE(IOUT,202) ANAME,LOCAT
  202 FORMAT(1H0,///30X,
     1     ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,60('-'))
C
C5A-----READ AN UNFORMATTED DUMMY RECORD FIRST.
      READ(LOCAT)
      READ(LOCAT) A
C
C6------IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.
  300 IF(CNSTNT.EQ.0.) GO TO 320
      DO 310 I=1,II
      DO 310 J=1,JJ
      A(J,I)=A(J,I)*CNSTNT
  310 CONTINUE
C
C7------IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
  320 IF(IPRN.LT.0) RETURN
      CALL ULAPRW(A,ANAME,0,0,JJ,II,0,IPRN,IOUT)
      RETURN
C
C8------RETURN
      END
```

List of Variables for Module U2DREL

Variable	Range	Definition
A	Module	DIMENSION (JJ,II), Input array.
ANAME	Module	Label for printout of the input array.
CNSTNT	Module	Constant to which all array values are set if LOCAT is equal to zero or by which all array values are multiplied if LOCAT is not equal to zero.
FMTIN	Module	DIMENSION (5), Format under which array values will be read.
I	Module	Index for rows.
II	Module	Number of rows.
IN	Module	Unit number from which the array control record will be read.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IPRN	Module	Code for format to be used when printing the arrays.
J	Module	Index for columns.
JJ	Module	Number of columns.
K	Module	Layer number.
LOCAT	Module	Location of values to fill in the array. < 0, read an unformatted record containing the array values. = 0, set all the array values equal to constant (CNSTNT). > 0, read the formatted records containing the array values.

Narrative for Module U2DINT

Module U2DINT reads values for a two-dimensional integer array. First it reads an "array-control record." Then, based on the contents of the array-control record, it may read array values. The array-control record contains four fields: location (LOCAT), constant (ICONST), format (FMTIN), and printout indicator (IPRN). The LOCAT field determines where array values will come from. If LOCAT is positive, it is the unit number from which array values will be read in the format specified in FMTIN. If LOCAT is negative, the sign is reversed to give the unit number from which an unformatted record containing the array values will be read. (Before the array record is read, a record will be read and ignored. Thus output from the module ULASAV can be read.) If LOCAT is zero, all of the array values will be set equal to ICONST. When LOCAT is not zero and ICONST is not zero, the array values will be multiplied by the value of ICONST. The field IPRN (table 1) contains a code number for a FORTRAN format to be used when printing the array.

Module U2DINT performs its tasks in the following order:

1. Read the array-control record (LOCAT, ICONST, FMTIN, and IPRN).

2. Use LOCAT to determine where the array is coming from. GO TO STEPS 3, 4, OR 5.

3. If LOCAT equals zero, set all array values equal to CNSTNT, print a message to that effect, and RETURN.

4. If LOCAT is greater than zero, read array values according to the format in FMTIN. GO TO STEP 6.

5. If LOCAT is less than zero, read an unformatted dummy record and then read an unformatted record containing the array values. GO TO STEP 6.

6. If ICONST is not equal to zero, multiply array values by ICONST.

7. If IPRN is greater than or equal to zero, print the array values using IPRN as the format code.

8. Call utility module UCOLNO to print column numbers at the top of the page.

9. Print each row in the array.

10. Select the format for printing.

11. RETURN.

Flow Chart for Module U2DINT

Array Control Record controls the input of array values. It contains four fields: LOCAT, ICONST, FMTIN, and IPRN.

LOCAT is a code showing where array values will come from.

If LOCAT < 0, array values will be read from an unformatted record from a unit number equal to -LOCAT.

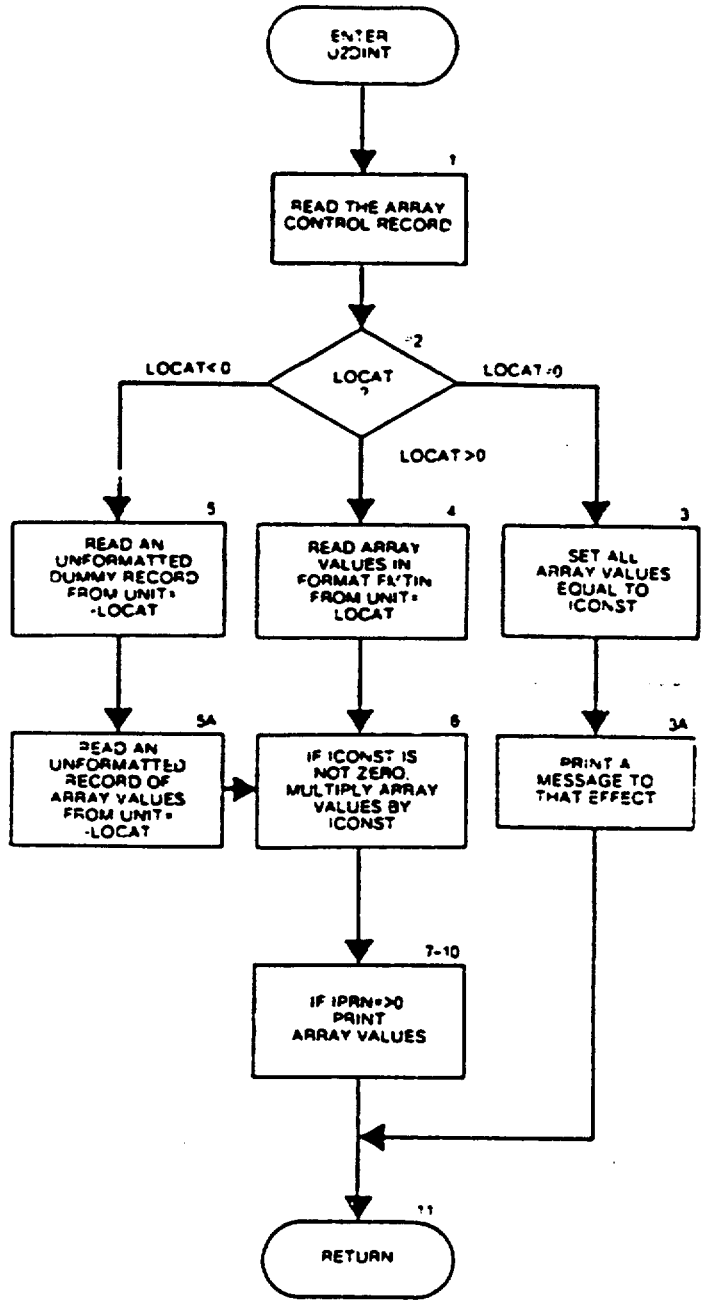
If LOCAT = 0, array values will be set equal to ICONST.

If LOCAT > 0, array values will be read from the unit number equal to LOCAT in the format specified in FMTIN.

ICONST is the constant to which all array values are set if LOCAT is equal to zero, and it is the constant by which all array values are multiplied if LOCAT is not equal to zero.

FMTIN is the format in which array values are read if LOCAT is greater than zero.

IPRN is a code showing the format to be used if array values are to be printed.



```
      SUBROUTINE U2DINT(IA,ANAME,II,JJ,K,IN,IOUT)
C
C
C-----VERSION 1442 20APR1983 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 UNLESS K IS 0)
C       IN IS INPUT UNIT
C       IOUT IS OUTPUT UNIT
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION IA(JJ,II),ANAME(6),FMTIN(5)
C     ------------------------------------------------------------------
C
C1------READ ARRAY CONTROL RECORD.
      READ (IN,1) LOCAT,ICONST,FMTIN,IPRN
    1 FORMAT(I10,I10,5A4,I10)
C
C2------USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
      IF(LOCAT) 200,50,90
C
C3------IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO ICONST. RETURN
   50 DO 90 I=1,II
      DO 80 J=1,JJ
   80 IA(J,I)=ICONST
      IF(K.GT.0) WRITE(IOUT,2) ANAME,ICONST,K
    2 FORMAT(1H0,52X,6A4,' =',I15,' FOR LAYER',I3)
      IF(K.LE.0) WRITE(IOUT,3) ANAME,ICONST
    3 FORMAT(1H0,52X,6A4,' =',I15)
      RETURN
C
C4------IF LOCAT>0 THEN READ FORMATTED RECORDS USING FORMAT FMTIN.
   90 IF(K.GT.0) WRITE(IOUT,4) ANAME,K,LOCAT,FMTIN
    4 FORMAT(1H0,///30X,6A4,' FOR LAYER',I3,' WILL BE READ ON UNIT',
     1       I3,' USING FORMAT: ',5A4/30X,96('-'))
      IF(K.LE.0) WRITE(IOUT,5) ANAME,LOCAT,FMTIN
    5 FORMAT(1H0,///30X,6A4,' WILL BE READ ON UNIT',
     1       I3,' USING FORMAT: ',5A4/30X,53('-'))
      DO 100 I=1,II
      READ (LOCAT,FMTIN) (IA(J,I),J=1,JJ)
  100 CONTINUE
      GO TO 300
C
C5------LOCAT<0 THEN READ UNFORMATTED RECORD CONTAINING ARRAY VALUES
  200 LOCAT=-LOCAT
      IF(K.GT.0) WRITE(IOUT,201) ANAME,K,LOCAT
  201 FORMAT(1H0,///30X,6A4,', LAYER',I3,
     1     ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,73('-'))
      IF(K.LE.0) WRITE(IOUT,202) ANAME,LOCAT
  202 FORMAT(1H0,///30X,6A4,
     1     ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,60('-'))
C
C5A-----READ AN UNFORMATTED DUMMY RECORD FIRST.
      READ(LOCAT)
      READ(LOCAT) IA
C
C6------IF ICONST NOT ZERO THEN MULTIPLY ARRAY VALUES BY ICONST.
  300 IF(ICONST.EQ.0) GO TO 320
      DO 310 I=1,II
      DO 310 J=1,JJ
      IA(J,I)=IA(J,I)*ICONST
  310 CONTINUE
C
C7------IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
  320 IF(IPRN.LT.0) RETURN
      IF(IPRN.GT.5) IPRN=0
      IPRN=IPRN+1
```

```
C
C8------PRINT COLUMN NUMBERS AT TOP OF PAGE.
      IF(IPRN.EQ.1) CALL UCOLNO(1,JJ,0,10,12,IOUT)
      NL=125/IPRN/5*5
      IF(IPRN.GT.1) CALL UCOLNO(1,JJ,4,NL,IPRN,IOUT)
C
C9------PRINT EACH ROW IN THE ARRAY.
      DO 110 I=1,II
C
C10-----SELECT THE FORMAT
      GO TO(101,102,103,104,105,106), IPRN
C
C---------------FORMAT 10I11
  101 WRITE(IOUT,1001) I,(IA(J,I),J=1,JJ)
 1001 FORMAT(1H0,I3,2X,I11,9(1X,I11)/(5X,10(1X,I11)))
      GO TO 110
C
C---------------FORMAT 60I1
  102 WRITE(IOUT,1002) I,(IA(J,I),J=1,JJ)
 1002 FORMAT(1H0,I3,1X,60(1X,I1)/(5X,60(1X,I1)))
      GO TO 110
C
C---------------FORMAT 40I2
  103 WRITE(IOUT,1003) I,(IA(J,I),J=1,JJ)
 1003 FORMAT(1H0,I3,1X,40(1X,I2)/(5X,40(1X,I2)))
      GO TO 110
C
C---------------FORMAT 30I3
  104 WRITE(IOUT,1004) I,(IA(J,I),J=1,JJ)
 1004 FORMAT(1H0,I3,1X,30(1X,I3)/(5X,30(1X,I3)))
      GO TO 110
C
C---------------FORMAT 25I4
  105 WRITE(IOUT,1005) I,(IA(J,I),J=1,JJ)
 1005 FORMAT(1H0,I3,1X,25(1X,I4)/(5X,25(1X,I4)))
      GO TO 110
C
C---------------FORMAT 20I5
  106 WRITE(IOUT,1006) I,(IA(J,I),J=1,JJ)
 1006 FORMAT(1H0,I3,1X,20(1X,I5)/(5X,20(1X,I5)))
  110 CONTINUE
      RETURN
C
C11-----RETURN
      END
```

List of Variables for Module U2DINT

Variable	Range	Definition
ANAME	Module	Label for the printout of input array.
FMTIN	Module	DIMENSION (5), Format under which the array values will be read.
I	Module	Index for rows.
IA	Module	DIMENSION (JJ,II), Input array.
ICONST	Module	Constant to which all array values are set if LOCAT is equal to zero or by which all array values are multiplied if LOCAT is not equal to zero.
II	Module	Number of rows.
IN	Module	Unit number from which the array-control record will be read.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IPRN	Module	Code for format to be used when printing arrays.
J	Module	Index for columns.
JJ	Module	Number of columns.
K	Module	Layer number.
LOCAT	Module	Location of values to fill in the array. < 0, read an unformatted record containing the array values. = 0, set all the array values equal to constant (CNSTNT). > 0, read formatted records containing the array values.
NL	Module	Number of columns per line.

Narrative for Module U1DREL

Module U1DREL reads values for a one-dimensional real array. First it reads an "array-control record." Then, based on the contents of the array-control record, it may read array values. The array-control record contains four fields: location (LOCAT), constant (CNSTNT), format (FMTIN), and printout indicator (IPRN). The LOCAT field determines where array values will come from. If LOCAT is positive, it is the unit number from which array values will be read in the format specified in FMTIN. If LOCAT is zero, all of the array values will be set equal to CNSTNT. If LOCAT is not zero and CNSTNT is not zero, the array values will be multiplied by the value of CNSTNT. The field IPRN (table 2) contains a code number for a FORTRAN format to be used when printing the array.

Module U1DREL performs its tasks in the following order:

1. Read the array-control record (LOCAT, CNSTNT, FMTIN, and IPRN).

2. Use LOCAT to determine where the array is coming from (DO STEPS 3 OR 4).

3. If LOCAT equals zero, set all array values equal to CNSTNT and print a message to that effect. RETURN.

4. If LOCAT is greater than zero, read array values according to the format in FMTIN.

5. If CNSTNT is not equal to zero, multiply the array values by CNSTNT.

6. If IPRN is greater than or equal to zero, print the array values.

7. RETURN.

Flow Chart for Module U1DREL

Array Control Record controls the input of array values. It contains four fields: LOCAT, CNSTNT, FMTIN, and IPRN.

LOCAT is a code showing where array values will come from.

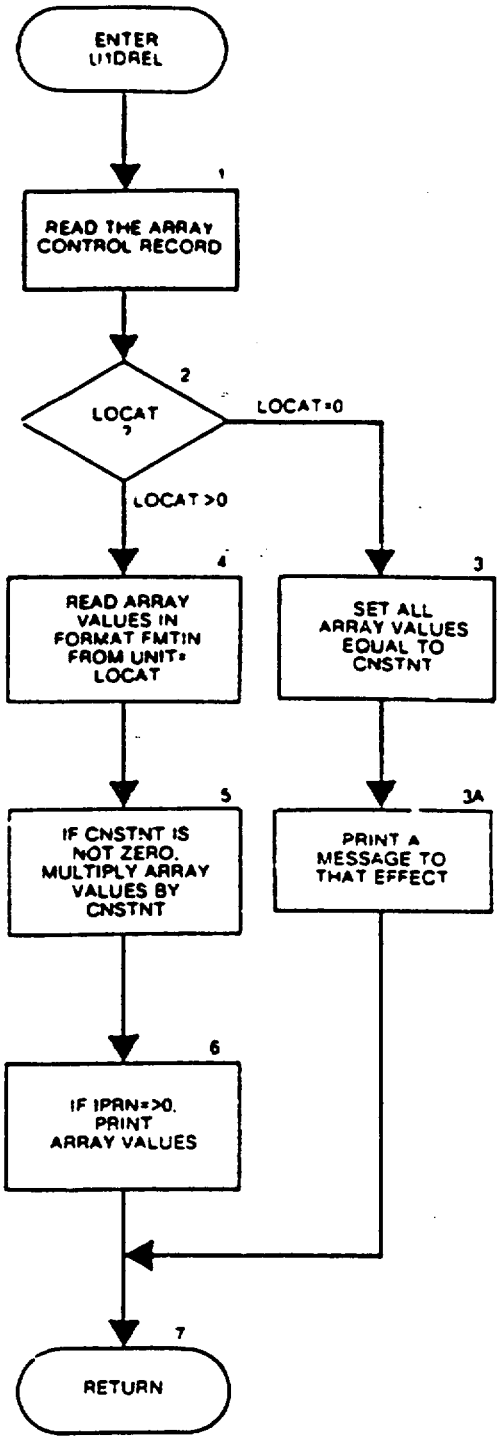
If LOCAT = 0, array values will be set equal to CNSTNT.

If LOCAT > 0, array values will be read from the unit number equal to LOCAT in the format specified in FMTIN.

CNSTNT is the constant to which all array values are set if LOCAT is equal to zero, and it is the constant by which all array values are multiplied if LOCAT is not equal to zero.

FMTIN is the format in which array values are read if LOCAT is greater than zero.

IPRN is a code showing the format to be used if array values are to be printed.



```
      SUBROUTINE U1DREL(A,ANAME,JJ,IN,IOUT)
C
C
C-----VERSION 1436 20MAY1983 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     ------------------------------------------------------------------
      DIMENSION A(JJ),ANAME(6),FMTIN(5)
C     ------------------------------ -----------------------------------
C
C1------READ ARRAY CONTROL RECORD.
      READ (IN,1) LOCAT,CNSTNT,FMTIN,IPRN
    1 FORMAT(I10,F10.0,5A4,I10)
C
C2------USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
      IF(LOCAT.GT.0) GO TO 90
C
C3------IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN
      DO 80 J=1,JJ
   80 A(J)=CNSTNT
      WRITE(IOUT,3) ANAME,CNSTNT
    3 FORMAT(1H0,52X,6A4,' =',G15.7)
      RETURN
C
C4------IF LOCAT>0 THEN READ FORMATTED RECORDS USING FORMAT FMTIN.
   90 WRITE(IOUT,5) ANAME,LOCAT,FMTIN
    5 FORMAT(1H0,///30X,6A4,' WILL BE READ ON UNIT',I3,
     1          ' USING FORMAT: ',5A4/30X,79('-')/)
      READ (LOCAT,FMTIN) (A(J),J=1,JJ)
C
C5------IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.
      IF(CNSTNT.EQ.0.) GO TO 120
      DO 100 J=1,JJ
  100 A(J)=A(J)*CNSTNT
C
C6------IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
 120  IF(IPRN.LT.0) RETURN
      WRITE(IOUT,1001) (A(J),J=1,JJ)
 1001 FORMAT((1X,1PG12.5,9(1X,G12.5)))
      RETURN
C
C7------CONTINUE
      END
```

List of Variables for Module U1DREL

Variable	Range	Definition
A	Module	DIMENSION (JJ), Input array.
ANAME	Module	Label for printout of the input array.
CNSTNT	Module	Constant to which all array values are set if LOCAT is equal to zero or by which all array values are multiplied if LOCAT is not equal to zero.
FMTIN	Module	DIMENSION (5), Format under which the array values will be read.
IN	Module	Unit number from which the array control record will be read.
IOUT	Global	Primary unit number for all printed output. IOUT = 6.
IPRN	Module	Code for the format to be used when printing the arrays.
J	Module	Array index.
JJ	Module	Number of elements in the array.
LOCAT	Module	Location of values to fill in the array. < 0, read an unformatted record containing the array values. = 0, set all the array values equal to constant (CNSTNT). > 0, read the formatted records containing the array values.

REFERENCES

Trescott, Peter C., 1975, Documentation of finite-difference model for simulation of three-dimensional ground-water flow: U.S. Geological Survey Open-File Report 75-438, 32 p.

Trescott, P. C., Pinder, G. F., and Larson, S. P., 1976, Finite-difference model for aquifer simulation in two dimensions with results of numerical experiments: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 7, Chapter C1, 116 p.

Weinstein, H. C., Stone, H. L., and Kwan, T. V., 1969, Iterative procedure for solution of systems of parabolic and elliptic equations in three dimensions: Indus. Engineering Chemistry Fundamentals, v. 8, no. 2, p. 281-287.

APPENDIX A

PROGRAM PORTABILITY

Introduction

One of the major design requirements for the model program was that it should run with a minimum of modification on most computers that are physically capable of running a program of this type. A program that can be easily interchanged among computers is called portable. The program documented in this report is portable. It has run successfully without modification on computers manufactured by many different companies including IBM, Control Data, Digital Equipment Corporation, Cray, Prime, Amdahl, and Univac. The compilers which were used were extensions of either FORTRAN 66 or FORTRAN 77. The following discussion explains, in more detail, the concept of portability and the significance of double precision arithmetic.

The Impact of the Programing Language on Portability

The programing language is the most important factor that determines program portability. There are a variety of programing languages available, and, for each language, there are numerous versions which have resulted from the desire of vendors to improve the power of the language and to take advantage of the hardware features of their particular computers. The most commonly available language suitable to use for the model program is FORTRAN. There are two versions defined by the American National Standards Institute (ANSI) on which most commercial versions are based, ANSI X3.9-1966 and ANSI X3.9-1978 (ANSI, 1966 and ANSI, 1978). These versions are commonly referred to as FORTRAN 66 and FORTRAN 77, respectively. FORTRAN 66 was selected for the model. The only exception to the FORTRAN 66 standard is that character strings in FORMAT and DATA statements are in single quotes rather than using Hollerith constants. Most FORTRAN 66 compilers include this capability. Although FORTRAN 66 is the less powerful of the two, it is far more widely supported in its entirety than is FORTRAN 77. The additional features of FORTRAN 77 would have made programing more convenient, but these features were not necessary to make a computationally efficient program. It is recognized, though, that some users may want to use the program with FORTRAN 77. Only a few changes are required to convert the program to FORTRAN 77 because FORTRAN 66 is nearly a subset of FORTRAN 77. Information about how to convert is provided in a following section (see Conversion to FORTRAN 77). Most commercial versions of FORTRAN 66 include some extended features not defined as part of FORTRAN 66, but such features were not used because they are not widely available. Since the program adheres to the FORTRAN 66 standard as understood by the authors, the program should work on any computer supporting this language provided that the computer has adequate computational power.

The Impact of Computational Precision on Portability

Variation of precision among computers causes some problems with program portability. Computational precision refers to the accuracy with which numbers are calculated and stored in the computer. To prevent the

imposition of constraints on the computers that implement FORTRAN 66, the computational precision was not defined as part of the standard. The accuracy of model results is highly dependent on computational precision, so precision must be considered when moving the model program among computers. Also, numeric precision has an effect on alphanumeric data since alphanumeric characters are stored in numeric variables and arrays in FORTRAN 66.

The model program was developed on computers using 32 binary bits to represent single-precision real numbers. This gives about seven digits of precision and includes the range from 10**-39 to 10**38. Double-precision real numbers are represented by 64 binary bits and have the same range as single-precision numbers, but about 16 decimal digits of precision. The head array, HNEW, and some variables in the solvers are stored as double precision, and, accordingly, some calculations are double precision. This was necessary for accuracy under some conditions. If using a computer with greater than 32-bit precision, the use of double-precision data and calculations may be unnecessary. The program should still work without modification, but computation time and memory would be saved if double precision were eliminated. One test of the program, on a computer using 64 bits to represent a real number, realized a savings of 75 percent of execution time when single-precision arithmetic was used exclusively. If it is determined that single precision is adequate, make the following changes to the program:

1. Delete all DOUBLE-PRECISION specification statements.

2. Change the Basic Package allocate module to allocate only NLAY*NROW*NCOL storage locations for HNEW rather than 2*NLAY*NROW*NCOL locations.

There are no other statements that require change. This is because the use of double-precision functions and constants are avoided; all conversions between real and double precision are done by implied type changes in assignment statements, and there are no "D" formats. The "D" formats are avoided by moving HNEW to a single-precision buffer before printing.

If less than 32-bit precision for single-precision real numbers is available, double precision may be necessary for all calculations. To convert all real calculations to double precision, do the following:

1. Change all REAL arrays and variables to double precision by the special command available on some compilers or by declaring all REAL arrays and variables as DOUBLE PRECISION in the MAIN Program and in every subroutine. Any variable or array starting with the letters A-H or O-Z should be declared DOUBLE PRECISION.

2. Change the Basic Package allocate module to allocate only NLAY*NROW*NCOL storage locations for HNEW rather than 2*NLAY*NROW*NCOL.

Some alphanumeric data (text data and format specifications) are stored in real variables and arrays. It is assumed that at least four characters can be stored in each variable or array element. It does not matter if more than four characters can fit because the extra characters are ignored.

Alphanumeric data occupy a trivial amount of memory, so the amount wasted if more than four charactres can fit in a variable or array element is insignificant. If less than four characters can fit into a variable or array element, the program must be changed. It is unlikely, however, that less than four characters will fit into one variable or array element unless the computer precision is low enough that all REAL variables and arrays require conversion to double precision as described above. Such conversion would automatically solve the alphanumeric character problem. If, for some reason, less than four characters will fit into a REAL variable or array element and yet all REAL variables and arrays are not being converted to double precision, just the REAL variables and arrays that store alphanumeric data should be declared DOUBLE PRECISION.

Conversion to FORTRAN 77

FORTRAN 66 is nearly a subset of FORTRAN 77. Accordingly, few changes are required to make the model program work on a FORTRAN 77 compiler. Most FORTRAN 77 compilers will accept the program as it is without change. Before modifying the program, try to compile it using your FORTRAN 77 compiler. Any changes that might have to be made are a result of the character data type which was added to FORTRAN 77. Any variables or arrays holding alphanumeric character data must be declared CHARACTER. This is most easily done by specifying that the same number of alphanumeric characters be stored in each character array element as was stored in each REAL variable or array element, which is four characters. This prevents the need to change the A4 format specifiers used to print alphanumeric characters. Thus, each DIMENSION ARRAY(N) statement, where ARRAY is an array holding 4*N alphanumeric characters, should be changed to CHARACTER*4 ARRAY(N). New statements of the form CHARACTER*4 VAR, where VAR is a variable that holds alphanumeric data, should be added to every subroutine in which alphanumeric data are stored in a variable. Also, Hollerith constants in DATA statements must be changed to character constants. Note that Hollerith constants in FORMAT statements need not be changed. These constants are no longer called Hollerith, but rather H edit descriptors; however, the form and function are the same. Before making any of these changes, check the FORTRAN 77 manual for the particular computer being used to see if these changes are actually necessary. Some FORTRAN 77 compilers have extensions that allow FORTRAN 66 programs to run without modification.

The above mentioned modifications to convert the model program to FORTRAN 77 do not take advantage of the added features of FORTRAN 77. The modifications simply allow the program to work on a FORTRAN 77 compiler. Modifying the program to take advantage of new features probably would not be cost effective. Program execution efficiency would not be significantly improved. The only improvement would be in the area of program structure. For example, the number of program statements might be reduced by using the new structured programing statements of FORTRAN 77, and the sometimes confusing "GO TO" statements could be eliminated. However, because the model program is broken into short modules with simple logic, FORTRAN 77 structural statements would provide only small benefits. Also, changes to take advantage of FORTRAN 77 features would limit program portability until such time that FORTRAN 77 is widely supported.

APPENDIX B

SPACE REQUIREMENTS IN THE X ARRAY

The outline below gives the X-array space requirements for each package. The formulas can be used to calculate the exact size of the X array for a given problem. However, it is generally easier to simply run the model program and let it calculate the size. The total space required is printed as part of the model printout even if the X array is dimensioned too small. As a rough estimate, the X array is approximately 10 to 15 times the number of nodes in the model depending on the options selected. In the outline below, NODES is defined as NCOL*NROW*NLAY, the number of nodes in the model.

I. BAS Package

A. 8*NODES + (NLAY - 1)*NCOL*NROW + NROW + NCOL + 4*NLAY

B. Additionally, add NODES if start head is saved (ISTRT is not 0), and add NODES if BUFF is separate from RHS (IAPART is not 0)

II. BCF Package

A. NLAY

B. If a transient simulation (ISS is 0), add NODES

C. For each layer where LAYCON is one or three, add 2*NCOL*NROW

D. For each layer where LAYCON is two or three, add NCOL*NROW

E. If a transient simulation (ISS is 0), for each layer where LAYCON is two or three, add NCOL*NROW

III. WEL Package -- 4*MXWELL

IV. DRN Package -- 5*MXDRAN

V. RIV Package -- 6*MXRIVR

VI. EVT Package

A. Option 1 -- 3*NCOL*NROW

B. Option 2 -- 4*NCOL*NROW

VII. GHB Package -- 5*MXBND

VIII. RCH Package

A. Options 1 and 3 -- NCOL*NROW

B. Option 2 -- 2*NCOL*NROW

IX. SIP Package -- 4*NODES + 4*MXITER + NPARM

X. SOR Package -- (NLAY + 4)*NCOL*NLAY + 4*MXITER

Generally, it is advisable to have the X-array dimension relatively close to the amount of space needed for a specific problem. On the other hand, it is inconvenient to redimension the X array every time a new option is selected. Several load modules with the size of the X array differing by a factor of two should be adequate.

APPENDIX C

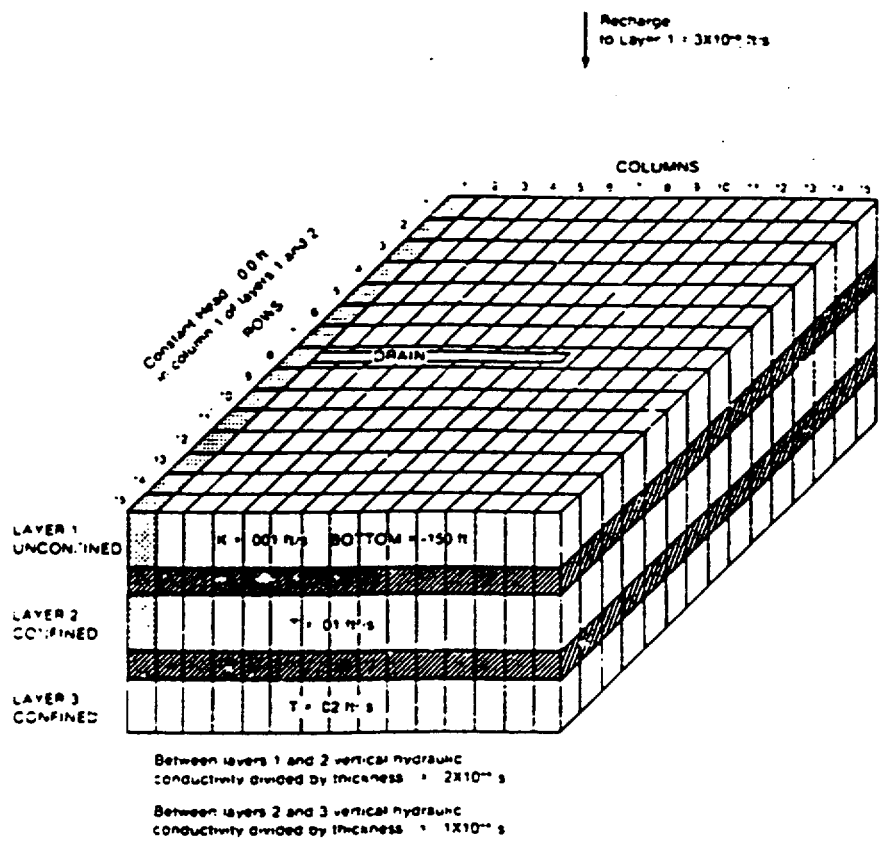
CONTINUATION OF A PREVIOUS RUN

There is often value in breaking long simulations into several short model runs. This allows one to decide, between runs, whether or not to continue the simulation. Although the model program in this report does not have a special option for making continuation runs, it is quite simple to continue a simulation by using the output of one run as input of the next. Simply save the heads from the run that is to be continued on a disk file, and specify that file as starting heads for the next run. The subroutine that reads starting heads (U2DREL) is capable of reading model-generated disk files of saved heads without the need for reformatting. Because volumetric budget terms are always set to zero at the start of a model run, the printed budget on a model run represents only that one run, not the total of all runs in a series of continuation runs. If a total budget for a series of continuation runs is desired, the totals from each run can be added externally. Similarly, the model program keeps track of simulation time only for single model runs, but total simulation time for a series of continuation runs can be calculated externally by adding the simulation times of each run.

APPENDIX D

SAMPLE PROBLEM

This sample problem is intended to illustrate input and output from the program. There are three simulated layers, as shown in the accompanying illustration, which are separated from each other by confining layers. Each layer is a square 75,000 feet on a side and is divided by a grid into 15 rows and 15 columns which form squares 5,000 feet on a side. Flow within the confinfng layers is not simulated, but the effects of the confining layers on flow between the active layers are incorporated in the vertical conductivity (VCONT) terms. Flow into the system is infiltration from precipitation; flow out of the system is to buried drain tubes, discharging wells, and a lake which is represented by a constant-head boundary.



Setting starting heads equal to 0.0, the program was run to get a steady-state solution. The Strongly Implicit Procedure was used to solve the system of difference equations: the error criterion was set at 0.001 feet, the acceleration parameter was set to 1.0, and the maximum number of iterations was set equal to 50. A seed of 0.001 was specified for use in calculating the iteration parameters; 31 iterations were needed to close.

List of Wells

$Q = 5\ ft^3/s$ for each well

Layer	Row	Column
3	5	11
2	4	6
2	6	12
1	9	8
1	9	10
1	9	12
1	9	14
1	11	8
1	11	10
1	11	12
1	11	14
1	13	8
1	13	10
1	13	12
1	13	14

List of Drains

Conductance = $1\ ft^2/s$

Layer	Row	Column	Elevation
1	8	2	0.0
1	8	3	0.0
1	8	4	10.0
1	8	5	20.0
1	8	6	30.0
1	8	7	50.0
1	8	8	70.0
1	8	9	90.0
1	8	10	100.0

```
//      JOB
//S2 EXEC PGM=MODEL1,REGION=700K
//STEPLIB DD DSN=VG4E70M.MODZ100,DISP=SHR
//    DD DSN=SYS1.FORTG.LINKLIBX,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT01F001 DD *
SAMPLE----3 LAYERS, 15 ROWS, 15 COLUMNS, STEADY STATE, CONSTANT HEADS COLUMN 1,
LAYERS 1 AND 2, RECHARGE, WELLS AND DRAINS
         3        15        15         1         1         1
 11 12 13  0  0  0  0 18 19  0  0 00
         0         0     IAPART,ISTRT
         1         1(15I3)                                  3    IBOUND-1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
         1         1(15I3)                                  3    IBOUND-2
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
         0         1                                        IBOUND-3
    999.99
         0        0.                                        HEAD-1
         0        0.                                        HEAD-2
         0        0.                                        HEAD-3
    86400.         1        1.     PERLEN,NSTP,TSMULT
```

```
//FT11F001 DD *
        1          0      ISS,IBCFBD
 1 0 0                                                    TRPY
        0         1.                                      DELR
        0      5000.                                      DELC
        0      5000.                                      HY-1
        0       .001                                      BOT-1
        0      -150.                                      VHY/THICK-1
        0      2.E-8                                      T-2
        0        .01                                      VHY/THICK-2
        0      1.E-8                                      T-3
        0        .02
/*
//FT18F001 DD *
        1          0      NRCHOP,IRCHBD
        1                 INRECH                          RECH-1
        0      3.E-8
/*
//FT19F001 DD *
       50          5        MXITER,NPARM          1  ACCL,ERR,IPCALC,WSEE
       1.       .001          0      .001
//FT13F001 DD *
        9          0      MXDRAI,IDRNBD
        9                 NDRAIN
        1          8          2        0.     1.E00
        1          8          3        0.     1.E00
        1          8          4       10.     1.E00
        1          8          5       20.     1.E00
        1          8          6       30.     1.E00
        1          8          7       50.     1.E00
        1          8          8       70.     1.E00
        1          8          9       90.     1.E00
        1          8         10      100.     1.E00
//FT12F001 DD *
       15          0      MXWELL,IWELBD
       15                 NWELL
        3          5         11       -5.
        2          4          6       -5.
        2          6         12       -5.
        1          9          8       -5.
        1          9         10       -5.
        1          9         12       -5.
        1          9         14       -5.
        1         11          8       -5.
        1         11         10       -5.
        1         11         12       -5.
        1         11         14       -5.
        1         13          8       -5.
        1         13         10       -5.
        1         13         12       -5.
        1         13         14       -5.
```

```
                    U.S. GEOLOGICAL SURVEY MODULAR FINITE-DIFFERENCE GROUND-WATER MODEL
SAMPLE--3 LAYERS,15 ROWS,15 COLUMNS,STEADY STATE,CONSTANT HEADS COLUMN 1,LAYERS 1 AND 2,RECHARGE,WELLS AND DRAINS
   3 LAYERS        15 ROWS        15 COLUMNS
  1 STRESS PERIOD(S) IN SIMULATION
MODEL TIME UNIT IS SECONDS
I/O UNITS:
ELEMENT OF IUNIT:  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
        I/O UNIT: 11 12 13  0  0  0  0 18 19  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
BAS1 -- BASIC MODEL PACKAGE, VERSION 1, 12/08/83 INPUT READ FROM UNIT  1
ARRAYS RHS AND BUFF WILL SHARE MEMORY.
START HEAD WILL NOT BE SAVED -- DRAWDOWN CANNOT BE CALCULATED
  5892 ELEMENTS IN X ARRAY ARE USED BY BAS
  5892 ELEMENTS OF X ARRAY USED OUT OF 100000
BCF1 -- BLOCK-CENTERED FLOW PACKAGE, VERSION 1, 12/08/83 INPUT READ FROM UNIT 11
STEADY-STATE SIMULATION
     LAYER  AQUIFER TYPE
     ------------------
        1         1
        2         0
        3         0
   453 ELEMENTS IN X ARRAY ARE USED BY BCF
  6345 ELEMENTS OF X ARRAY USED OUT OF 100000
WEL1 -- WELL PACKAGE, VERSION 1, 12/08/83 INPUT READ FROM 12
MAXIMUM OF   15 WELLS
    60 ELEMENTS IN X ARRAY ARE USED FOR WELLS
  6405 ELEMENTS OF X ARRAY USED OUT OF 100000
DRN1 -- DRAIN PACKAGE, VERSION 1, 12/08/83 INPUT READ FROM UNIT 13
MAXIMUM OF    9 DRAINS
    45 ELEMENTS IN X ARRAY ARE USED FOR DRAINS
  6450 ELEMENTS OF X ARRAY USED OUT OF 100000
RCH1 -- RECHARGE PACKAGE, VERSION 1, 12/08/83 INPUT READ FROM UNIT 18
OPTION 1 -- RECHARGE TO TOP LAYER
   225 ELEMENTS OF X ARRAY USED FOR RECHARGE
  6675 ELEMENTS OF X ARRAY USED OUT OF 100000
SIP1 -- STRONGLY IMPLICIT PROCEDURE SOLUTION PACKAGE, VERSION 1, 12/08/83 INPUT READ FROM UNIT 15
MAXIMUM OF  50 ITERATIONS ALLOWED FOR CLOSURE
 5 ITERATION PARAMETERS
  2905 ELEMENTS IN X ARRAY ARE USED BY SIP
  9580 ELEMENTS OF X ARRAY USED OUT OF 100000
```

SAMPLE--3 LAYERS,15 ROWS,15 COLUMNS,STEADY STATE,CONSTANT HEADS COLUMN 1,LAYERS 1 AND 2,RECHARGE,WELLS AND DRAINS

BOUNDARY ARRAY FOR LAYER 1 WILL BE READ ON UNIT 1 USING FORMAT: (1513)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
15	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

BOUNDARY ARRAY FOR LAYER 2 WILL BE READ ON UNIT 1 USING FORMAT: (1513)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
15	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

```
                                                    BOUNDARY ARRAY =              1 FOR LAYER  3
AQUIFER HEAD WILL BE SET TO  999.99     AT ALL NO-FLOW NODES (IBOUND=0).
                                                       INITIAL HEAD =   .0          FOR LAYER  1
                                                       INITIAL HEAD =   .0          FOR LAYER  2
                                                       INITIAL HEAD =   .0          FOR LAYER  3
DEFAULT OUTPUT CONTROL -- THE FOLLOWING OUTPUT COMES AT THE END OF EACH STRESS PERIOD:
TOTAL VOLUMETRIC BUDGET
        HEAD

                                          COLUMN TO ROW ANISOTROPY =   1.000000
                                                              DELR =   5000.000
                                                              DELC =   5000.000
                                             HYD. COND. ALONG ROWS =   .9999999E-03 FOR LAYER  1
                                                            BOTTOM =  -150.0000     FOR LAYER  1
                                          VERT HYD COND /THICKNESS =   .2000000E-07 FOR LAYER  1
                                              TRANSMIS. ALONG ROWS =   .1000000E-01 FOR LAYER  2
                                          VERT HYD COND /THICKNESS =   .1000000E-07 FOR LAYER  2
                                              TRANSMIS. ALONG ROWS =   .2000000E-01 FOR LAYER  3


                                              SOLUTION BY THE STRONGLY IMPLICIT PROCEDURE
                                              --------------------------------------------
                                     MAXIMUM ITERATIONS ALLOWED FOR CLOSURE =       50
                                                   ACCELERATION PARAMETER =      1.0000
                                          HEAD CHANGE CRITERION FOR CLOSURE =    0.10000E-02
                                          SIP HEAD CHANGE PRINTOUT INTERVAL =        1

  5 ITERATION PARAMETERS CALCULATED FROM SPECIFIED WSEED = 0.00100000 :

        0.0            0.8221720E+00  0.9683772E+00  0.9943766E+00  0.9990000E+00
                                          STRESS PERIOD NO.   1, LENGTH =   86400.00
                                          --------------------------------------------

                                          NUMBER OF TIME STEPS =     1

                                           MULTIPLIER FOR DELT =     1.000

                                        INITIAL TIME STEP SIZE =   86400.00
```

15 WELLS

LAYER	ROW	COL.	STRESS RATE	WELL NO.
3	5	11	-5.0000	1
2	4	6	-5.0000	2
2	6	12	-5.0000	3
1	9	8	-5.0000	4
1	9	10	-5.0000	5
1	9	12	-5.0000	6
1	9	14	-5.0000	7
1	11	8	-5.0000	8
1	11	10	-5.0000	9
1	11	12	-5.0000	10
1	11	14	-5.0000	11
1	13	8	-5.0000	12
1	13	10	-5.0000	13
1	13	12	-5.0000	14
1	13	14	-5.0000	15

9 DRAINS

LAYER	ROW	COL	ELEVATION	CONDUCTANCE	DRAIN NO.
1	8	2	.0	1.000	1
1	8	3	.0	1.000	2
1	8	4	10.00	1.000	3
1	8	5	20.00	1.000	4
1	8	6	30.00	1.000	5
1	8	7	50.00	1.000	6
1	8	8	70.00	1.000	7
1	8	9	90.00	1.000	8
1	8	10	100.0	1.000	9

RECHARGE = .3000000E-07

31 ITERATIONS FOR TIME STEP 1 IN STRESS PERIOD 1
MAXIMUM HEAD CHANGE FOR EACH ITERATION:

HEAD CHANGE	LAYER,ROW,COL.	HEAD CHANGE	LAYER,ROW,COL	HEAD CHANGE	LAYER,ROW,COL	HEAD CHANGE	LAYER,ROW,COL	HEAD CHANGE	LAYER,ROW,COL	HEAD CHANGE	LAYER,ROW,COL
-22.41	(3, 5, 11)	12.48	(1, 1, 15)	13.39	(3, 1, 14)	46.21	(1, 1, 15)	35.90	(3, 1, 13)		
2.482	(1, 9, 14)	1.430	(3, 10, 13)	6.214	(1, 12, 14)	7.411	(3, 11, 14)	13.66	(1, 15, 15)		
.5503	(3, 8, 7)	.8821	(2, 6, 9)	.4711	(3, 5, 10)	2.019	(1, 11, 14)	2.302	(3, 5, 13)		
.1108	(1, 13, 12)	.7059E-01	(3, 12, 11)	.2819	(1, 14, 14)	.3143	(3, 13, 14)	.3321	(1, 15, 15)		
.7855E-02	(1, 13, 12)	.1586E-01	(2, 11, 11)	.1777E-01	(3, 11, 10)	.7911E-01	(1, 14, 14)	.8596E-01	(3, 7, 14)		
.4169E-02	(1, 13, 14)	.2555E-02	(3, 14, 15)	.9771E-02	(1, 14, 14)	.1082E-01	(3, 13, 14)	.1030E-01	(1, 15, 15)		
.2826E-03	(1, 13, 12)										

HEAD IN LAYER 1 AT END OF TIME STEP 1 IN STRESS PERIOD 1

	1	2	3	4	5	6	7	8	9	10
	11	12	13	14	15					
1	.0	24.94	44.01	59.26	71.82	82.52	91.91	100.0	106.9	112.6
	117.4	121.3	124.3	126.4	127.4					
2	.0	24.45	43.10	57.98	70.17	80.57	90.12	98.40	105.3	111.0
	115.7	119.6	122.7	124.9	126.1					
3	.0	23.45	41.30	55.43	66.78	76.21	86.51	95.20	102.2	107.6
	112.0	116.1	119.6	122.1	123.4					
4	.0	21.92	38.61	51.75	61.79	68.03	81.34	90.75	97.64	102.5
	106.1	110.7	114.9	117.9	119.4					
5	.0	19.73	34.92	47.32	57.69	66.74	77.09	85.76	92.22	96.15
	97.29	103.1	108.8	112.5	114.3					
6	.0	16.51	29.50	40.90	51.30	61.21	71.19	79.85	86.47	90.82
	93.03	94.23	102.1	106.4	108.4					
7	.0	11.55	21.10	31.21	41.40	51.84	63.08	72.68	79.95	84.92
	88.60	91.66	96.43	99.82	101.8					
8	.0	3.483	6.832	16.25	26.30	36.97	52.59	64.31	72.52	77.25
	81.99	85.00	89.27	91.72	94.33					
9	.0	10.54	19.11	28.12	36.92	45.27	52.95	55.38	65.15	66.07
	73.93	73.79	80.84	80.17	86.49					
10	.0	14.62	25.86	35.38	43.49	50.11	54.93	57.55	62.95	65.55
	70.39	72.44	76.72	78.26	81.79					
11	.0	17.11	29.96	40.01	47.78	53.24	55.81	53.33	60.27	59.29
	66.43	65.45	72.22	71.04	77.62					
12	.0	18.68	32.56	43.07	50.81	55.92	58.33	58.47	61.93	63.18
	67.12	68.50	72.29	73.46	76.85					
13	.0	19.67	34.24	45.14	53.01	58.04	59.91	56.75	62.59	60.91
	67.22	65.75	71.90	70.35	76.48					
14	.0	20.27	35.27	46.48	54.61	60.08	63.17	64.52	67.25	68.79
	71.64	73.18	75.84	77.03	79.09					
15	.0	20.56	35.78	47.16	55.48	61.26	65.02	67.52	69.94	72.01
	74.29	76.22	78.22	79.66	80.82					

HEAD IN LAYER 2 AT END OF TIME STEP 1 IN STRESS PERIOD 1

	1 11	2 12	3 13	4 14	5 15	6	7	8	9	10
1	.0 117.2	24.66 121.1	43.73 124.1	59.02 126.2	71.61 127.3	82.32	91.72	99.86	106.7	112.5
2	.0 115.5	24.17 119.4	42.83 122.6	57.74 124.8	69.95 125.9	80.36	89.93	98.22	105.1	110.8
3	.0 111.8	23.17 116.0	41.03 119.5	55.19 121.9	66.53 123.2	75.77	86.29	95.02	102.0	107.4
4	.0 105.4	21.65 110.4	38.34 114.8	51.50 117.7	61.35 119.2	60.17	80.90	90.55	97.45	102.3
5	.0 91.09	19.48 102.1	34.65 108.6	47.07 112.4	57.44 114.2	66.30	76.85	85.57	92.00	95.41
6	.0 92.06	16.27 86.23	29.24 101.7	40.65 106.2	51.07 108.3	60.98	70.98	79.65	86.28	90.54
7	.0 88.35	11.38 91.24	20.95 96.22	31.05 99.65	41.25 101.6	51.70	62.90	72.48	79.76	84.73
8	.0 81.81	4.209 84.86	8.330 89.10	17.58 91.59	27.58 94.17	38.25	52.94	64.19	72.34	77.12
9	.0 73.87	10.38 74.48	18.96 80.77	27.98 80.84	36.79 86.38	45.16	52.86	56.13	65.08	66.79
10	.0 70.24	14.40 72.37	25.61 76.57	35.15 78.20	43.27 81.64	49.91	54.76	57.48	62.79	65.49
11	.0 66.37	16.87 66.18	29.70 72.16	39.78 71.75	47.56 77.51	53.05	55.68	54.09	60.20	60.04
12	.0 66.98	18.43 68.44	32.31 72.15	42.85 73.40	50.60 76.69	55.73	58.16	58.41	61.78	63.12
13	.0 67.16	19.42 66.48	33.98 71.84	44.91 71.06	52.80 76.37	57.85	59.78	57.50	62.53	61.65
14	.0 71.48	20.02 73.06	35.02 75.68	46.26 76.91	54.41 78.93	59.88	62.99	64.39	67.08	68.66
15	.0 74.11	20.30 76.04	35.52 78.04	46.94 79.49	55.28 80.65	61.07	64.84	67.34	69.76	71.84

HEAD IN LAYER 3 AT END OF TIME STEP 1 IN STRESS PERIOD 1

	1	2	3	4	5	6	7	8	9	10
	11	12	13	14	15					
1	1.800	24.34	43.36	58.70	71.33	82.06	91.48	99.63	106.5	112.3
	117.0	120.9	123.9	126.0	127.1					
2	1.764	23.85	42.46	57.42	69.66	80.07	89.68	97.99	104.9	110.6
	115.3	119.2	122.4	124.6	125.7					
3	1.691	22.86	40.67	54.87	66.20	75.28	85.98	94.77	101.7	107.2
	111.5	115.7	119.3	121.7	123.0					
4	1.578	21.35	37.98	51.17	60.85	62.69	80.41	90.28	97.19	101.9
	104.1	110.0	114.5	117.5	119.0					
5	1.415	19.18	34.30	46.75	57.10	65.80	76.54	85.30	91.67	94.17
	77.46	100.7	108.2	112.1	114.0					
6	1.176	15.99	28.91	40.33	50.76	60.67	70.70	79.38	86.01	90.12
	90.60	88.55	101.2	106.0	108.0					
7	.8273	11.21	20.79	30.88	41.09	51.55	62.67	72.22	79.50	84.46
	87.98	90.77	95.94	99.41	101.4					
8	.4331	5.131	10.19	19.27	29.19	39.84	53.40	64.07	72.11	76.95
	81.58	84.68	88.88	91.44	93.95					
9	.7543	10.22	18.82	27.84	36.66	45.06	52.78	57.03	65.02	67.64
	73.81	75.31	80.72	81.64	86.24					
10	1.039	14.13	25.29	34.85	42.99	49.65	54.54	57.44	62.61	65.44
	70.05	72.33	76.39	78.15	81.43					
11	1.224	16.59	29.37	39.47	47.28	52.79	55.53	55.01	60.16	60.94
	66.33	67.06	72.13	72.60	77.38					
12	1.341	18.15	31.97	42.54	50.32	55.47	57.94	58.37	61.60	63.08
	66.80	68.41	71.97	73.36	76.49					
13	1.415	19.14	33.65	44.61	52.53	57.60	59.63	58.39	62.48	62.54
	67.12	67.35	71.80	71.90	76.24					
14	1.460	19.73	34.68	45.96	54.13	59.63	62.76	64.24	66.87	68.52
	71.27	72.91	75.47	76.77	78.71					
15	1.481	20.01	35.18	46.63	55.00	60.81	64.59	67.11	69.52	71.61
	73.87	75.82	77.81	79.27	80.42					

VOLUMETRIC BUDGET FOR ENTIRE MODEL AT END OF TIME STEP 1 IN STRESS PERIOD 1

CUMULATIVE VOLUMES	L**3	RATES FOR THIS TIME STEP	L**3/T
IN:		IN:	
STORAGE =	.0	STORAGE =	.0
CONSTANT HEAD =	.0	CONSTANT HEAD =	.0
WELLS =	.0	WELLS =	.0
DRAINS =	.0	DRAINS =	.0
RECHARGE =	.13608E+08	RECHARGE =	157.50
TOTAL IN =	.13608E+08	TOTAL IN =	157.50
OUT:		OUT:	
STORAGE =	.0	STORAGE =	.0
CONSTANT HEAD =	.83265E+07	CONSTANT HEAD =	50.075
WELLS =	.64800E+07	WELLS =	75.000
DRAINS =	.28011E+07	DRAINS =	32.420
RECHARGE =	.0	RECHARGE =	.0
TOTAL OUT =	.13608E+08	TOTAL OUT =	157.49
IN - OUT =	184.00	IN - OUT =	.21210E-02
PERCENT DISCREPANCY =	0.00	PERCENT DISCREPANCY =	0.00

TIME SUMMARY AT END OF TIME STEP 1 IN STRESS PERIOD 1

	SECONDS	MINUTES	HOURS	DAYS	YEARS
TIME STEP LENGTH	86400.0	1440.00	24.0000	1.00000	.273785E-02
STRESS PERIOD TIME	86400.0	1440.00	24.0000	1.00000	.273785E-02
TOTAL SIMULATION TIME	86400.0	1440.00	24.0000	1.00000	.273785E-02

APPENDIX E

ABBREVIATED INPUT INSTRUCTIONS

These input instructions are intended as a quick reference for the experienced user. Most explanations that are contained in the complete input instructions given in package documentation have been omitted. The format of input fields is given only for those records that contain fields that are not 10 characters wide. Each input item, for which format is not given, is identified as either a record or an array. For records, the fields contained in the record are named. For arrays, only the array name is given. Input fields which contain codes or flags are described. All other field and array descriptions have been dropped.

Array Input

The real two-dimensional array reader (U2DREL), the integer two-dimensional array reader (U2DINT), and the real one-dimensional array reader (U1DREL) read one array-control record and, optionally, a data array in a format specified on the array-control record.

FOR REAL ARRAY READER (U2DREL or U1DREL)

Data:	LOCAT	CNSTNT	FMTIN	IPRN
Format:	I10	F10.0	5A4	I10

FOR INTEGER ARRAY READER (U2DINT)

Data:	LOCAT	ICONST	FMTIN	IPRN
Format:	I10	I10	5A4	I10

IPRN--is a flag indicating that the array being read should be printed and a code for indicating the format that should be used. It is used only if LOCAT is not equal to zero. The format codes are different for each of the three modules. IPRN is set to zero when the specified value exceeds those defined in the chart below. If IPRN is less than zero, the array will not be printed.

IPRN	U2DREL	U2DINT	U1DREL
0	10G11.4	10I11	10G12.5
1	11G10.3	60I1	
2	9G13.6	40I2	
3	15F7.1	30I3	
4	15F7.2	25I4	
5	15F7.3	20I5	
6	15F7.4		
7	20F5.0		
8	20F5.1		
9	20F5.2		
10	20F5.3		
11	20F5.4		
12	10G11.4		

LOCAT--indicates the location of the data which will be put in the array.
If LOCAT < 0, unit number for unformatted records.
If LOCAT = 0, all elements are set equal to CNSTNT or ICONST.
If LOCAT > 0, unit number for formatted records.

Basic Package Input

Input for the Basic (BAS) Package except for output control is read from unit 1 as specified in the main program. If necessary, the unit number for BAS input can be changed to meet the requirements of a particular computer. Input for the output control option is read from the unit number specified in IUNIT(12).

```
FOR EACH SIMULATION
            1.  Record: HEADNG(32)
            2.  Record: HEADNG (continued)
            3.  Record: NLAY    NROW    NCOL    NPER    ITMUNI
            4.  Data:   IUNIT(24)
                Format: 24I3
                   (BCF WEL DRN RIV EVT XXX GHB RCH SIP XXX SOR OC)
                     1   2   3   4   5   6   7   8   9   10  11 12
            5.  Record: IAPART  ISTRT
            6.  Array:  IBOUND(NCOL,NROW)
                        (One array for each layer in the grid)
            7.  Record: HNOFLO
            8.  Array:  Shead(NCOL,NROW)
                        (One array for each layer in the grid)
FOR EACH STRESS PERIOD
            9.  Data:   PERLEN   NSTP  TSMULT
```

ITMUNI--is the time unit of model data.

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

Consistent length and time units must be used for all model data. The user may choose one length unit and one time unit to be used to specify all input data.

IUNIT--is a 24-element table of input units for use by all major options.

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

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

If IAPART ≠ 0, the arrays BUFF and RHS occupy different space.

ISTRT--indicates whether starting heads are to be saved.

If ISTRT = 0, starting heads are not saved.

If ISTRT ≠ 0, starting heads are saved.

IBOUND--is the boundary array.

If IBOUND(I,J,K) < 0, cell I,J,K has a constant head.

If IBOUND(I,J,K) = 0, cell I,J,K is inactive.

If IBOUND(I,J,K) > 0, cell I,J,K is active.

HNOFLO--is the value of head to be assigned to all inactive cells.

Shead--is head at the start of the simulation.

PERLEN--is the length of a stress period.

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

TSMULT--is the multiplier for the length of successive time steps.

Output Control Input

Input to Output Control is read from the unit specified in IUNIT(12). All printer output goes to unit 6 as specified in the main program. If necessary, the unit number for printer output can be changed to meet the requirements of a particular computer.

FOR EACH SIMULATION

1. Record: IHEDFM IDDNFM IHEDUN IDDNUN

FOR EACH TIME STEP

2. Record: INCODE IHDDFL IBUDFL ICBCFL
3. Record: Hdpr Ddpr Hdsv Ddsv

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

IHEDFM--is a code for the format in which heads will be printed.
IDDNFM--is a code for the format in which drawdowns will be printed.

	0 - (10G11.4)	7 - (20F5.0)
	1 - (11G10.3)	8 - (20F5.1)
positive--wrap	2 - (9G13.6)	9 - (20F5.2)
	3 - (15F7.1)	10 - (20F5.3)
negative--strip	4 - (15F7.2)	11 - (20F5.4)
	5 - (15F7.3)	12 - (10G11.4)
	6 - (15F7.4)	

IHEDUN--is the unit number on which heads will be saved.
IDDNUN--is the unit number on which drawdowns will be saved.
INCODE--is the head/drawdown ouput code.
If INCODE < 0, layer-by-layer specifications from the last time steps are used. Input item 3 is not read.
If INCODE = 0, all layers are treated the same way. Input item 3 will consist of one record. IOFLG array will be read.
If INCODE > 0, input item 3 will consist of one record for each layer.
IHDDFL--is a head and drawdown output flag.
If IHDDFL = 0, neither heads nor drawdowns will be printed or saved.
If IHDDFL ≠ 0, heads and drawdowns will be printed or saved.
IBUDFL--is a budget print flag.
If IBUDFL = 0, overall volumetric budget will not be printed.
If IBUDFL ≠ 0, overall volumetric budget will be printed.
ICBCFL--is a cell-by-cell flow-term flag.
If ICBCFL = 0, cell-by-cell flow terms are not saved or printed.
If ICBCFL ≠ 0, cell-by-cell flow terms are printed or recorded on disk depending on flags set in the component of flow packages, i.e., IWELCB, IRCHCB, etc.
Hdpr--is the output flag for head printout.
If Hdpr = 0, head is not printed for the corresponding layer.
If Hdpr ≠ 0, head is printed for the corresponding layer.
Ddpr--is the output flag for drawdown printout.
If Ddpr = 0, drawdown is not printed for the corresponding layer.
If Ddpr ≠ 0, drawdown is printed for the corresponding layer.
Hdsv--is the output flag for head save.
If Hdsv = 0, head is not saved for the corresponding layer.
If Hdsv ≠ 0, head is saved for the corresponding layer.
Ddsv--is the output flag for drawdown save.
If Ddsv = 0, drawdown is not saved for the corresponding layer.
If Ddsv ≠ 0, drawdown is saved for the corresponding layer.

Block-Centered Flow Package Input

Input for the BCF Package is read from the unit specified in IUNIT(1).

FOR EACH SIMULATION

1. Record: ISS IBCFCB
2. Data: LAYCON(NLAY) (maximum of 80 layers)
   Format: 40I2
   (If there are 40 or fewer layers, use one record.)
3. Array: TRPY(NLAY)
4. Array: DELR(NCOL)
5. Array: DELC(NROW)

All of the arrays (items 6-12) for layer 1 are read first; then all of the arrays for layer 2, etc.

IF THE SIMULATION IS TRANSIENT

6. Array: sf1(NCOL,NROW)

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

7. Array: Tran(NCOL,NROW)

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

8. Array: HY(NCOL,NROW)
9. Array: BOT(NCOL,NROW)

IF THIS IS NOT THE BOTTOM LAYER

10. Array: Vcont(NCOL,NROW)

IF THE SIMULATION IS TRANSIENT AND THE LAYER TYPE CODE (LAYCON) is TWO OR THREE

11. Array: sf2(NCOL,NROW)

IF THE LAYER TYPE CODE IS TWO OR THREE

12. Array: TOP(NCOL,NROW)

ISS--is the steady-state flag.
If ISS ≠ 0, the simulation is steady state.
If ISS = 0, the simulation is transient.

IBCFCB--is a flag and a unit number.
If IBCFCB > 0, cell-by-cell flow terms will be recorded if ICBCFL (see Output Control) is set.
If IBCFCB = 0, cell-by-cell flow terms will not be printed or recorded.
If IBCFCB < 0, print flow for constant-head cells if ICBCFL is set.

LAYCON--is the layer type table: 0 - confined, 1 - unconfined, 2 - confined/unconfined (T constant), and 3 - confined/unconfined.

TRPY--is an anisotropy factor for each layer: T or K along a column to T or K along a row.

DELR--is the cell width along rows.

DELC--is the cell width along columns.

sf1--is the primary storage factor.

Tran--is the transmissivity along rows.

HY--is the hydraulic conductivity along rows.

BOT--is the elevation of the aquifer bottom.

Vcont--is the vertical hydraulic conductivity divided by the thickness from a layer to the layer beneath it.

sf2--is the secondary storage factor.

TOP--is the elevation of the aquifer top.

River Package Input

Input to the River (RIV) Package is read from the unit specified in IUNIT(4).

FOR EACH SIMULATION
1. Record: MXRIVR IRIVCB

FOR EACH STRESS PERIOD
2. Record: ITMP
3. Record: Layer Row Column Stage Cond Rbot
(Input item 3 normally consists of one record for each river reach. If ITMP is negative or zero, item 3 is not read.)

IRIVCB--is a flag and a unit number.
If IRIVCB > 0, cell-by-cell flow terms will be recorded.
If IRIVCB = 0, cell-by-cell flow terms will not be printed or recorded.
If IRIVCB < 0, river leakage will be printed if ICBCFL is set.

ITMP--is a flag and a counter.
If ITMP < 0, river data from the last stress period will be reused.
If ITMP $\geq$ 0, ITMP will be the number of reaches active during the current stress period.

Recharge Package Input

Input to the Recharge (RCH) Package is read from the unit specified in IUNIT(8).

FOR EACH SIMULATION
1. Record: NRCHOP IRCHCB

FOR EACH STRESS PERIOD
2. Record: INRECH INIRCH
3. Array: RECH(NCOL,NROW)

IF THE RECHARGE OPTION IS EQUAL TO 2
4. Array: IRCH(NCOL,NROW)

NRCHOP--is the recharge option code.
1 - Recharge is only to the top grid layer.
2 - Vertical distribution of recharge is specified in array IRCH.
3 - Recharge is applied to the highest active cell in each vertical column.

IRCHCB--is a flag and a unit number.
If IRCHCB > 0, unit number for cell-by-cell flow terms.
If IRCHCB $\leq$ 0, cell-by-cell flow terms will not be printed or recorded.

INRECH--is the RECH read flag.
If INRECH < 0, recharge rates from the preceding stress period are used.
If INRECH $\geq$ 0, an array of recharge rates, (RECH) is read.

INIRCH--is similar to INRECH.

Well Package Input

Input for the Well (WEL) Package is read from the unit specified in IUNIT(2).

```
FOR EACH SIMULATION
            1. Record: MXWELL  IWELCB
FOR EACH STRESS PERIOD
            2. Record: ITMP
            3. Record: Layer      Row      Column      0
               (Input item 3 normally consists of one record for each
               well.  If ITMP is negative or zero, item 3 is not read.)
```

MXWELL--is the maximum number of wells used at any time.
IWELCB--is a flag and a unit number.
If IWELCB > 0, unit number for cell-by-cell flow terms.
If IWELCB = 0, cell-by-cell flow terms will not be printed or recorded.
If IWELCB < 0, well recharge will be printed whenever ICBCFL is set.
ITMP--is a flag and a counter.
If ITMP < 0, well data from the last stress period will be reused.
If ITMP ≥ 0, ITMP will be the number of wells active during the current stress period.

Drain Package Input

Input to the Drain (DRN) Package is read from the unit specified in IUNIT(3).

```
FOR EACH SIMULATION
            1.  Record:  MXDRN      IDRNCB
FOR EACH STRESS PERIOD
            2.  Record:  ITMP
            3.  Record:  Layer     Row     Col     Elevation     Cond
            (Input item 3 normally consists of one record for each drain.
            If ITMP is negative or zero, item 3 will not be read.)
```

MXDRN--is the maximum number of drain cells active at one time.
IDRNCB--is a flag and a unit number.
If IDRNCB > 0, unit number for cell-by-cell flow terms.
If IDRNCB = 0, cell-by-cell flow terms will not be printed or recorded.
If IDRNCB < 0, drain leakage for each cell will be printed whenever ICBCFL is set.
ITMP--is a flag and a counter.
If ITMP < 0, drain data from the last stress period will be reused.
If ITMP ≥ 0, ITMP will be the number of drains active during the current stress period.

Evapotranspiration Package Input

Input to the Evapotranspiration (EVT) Package is read from the unit specified in IUNIT (5).

FOR EACH SIMULATION
1. Record: NEVTOP IEVTCB

FOR EACH STRESS PERIOD
2. Record: INSURF INEVTR INEXDP INIEVT
3. Array: SURF
4. Array: EVTR
5. Array: EXDP

IF THE ET OPTION IS EQUAL TO TWO
6. Array: IEVT

NEVTOP--is the evapotranspiration (ET) option code.
1 - ET is calculated only for cells in the top grid layer.
2 - The cell for each vertical column is specified by the user in array IEVT.

IEVTCB--is a flag and a unit number.
If IEVTCB > 0, unit number for cell-by-cell flow terms.
If IEVTCB $\leq$ 0, cell-by-cell flow terms will not be printed or recorded.

INSURF--is the ET surface (SURF) read flag.
If INSURF $\geq$ 0, an array containing the ET surface elevation will be read.
If INSURF < 0, the ET surface from the preceding stress period will be reused.

INEVTR--is similar to INSURF.

INEXDP--is similar to INSURF.

INIEVT--is similar to INSURF.

General-Head Boundary Package Input

Input for the General-Head Boundary (GHB) Package is read from the unit specified in IUNIT(7).

```
FOR EACH SIMULATION
    1. Record:  MXBND      IGHBCB
FOR EACH STRESS PERIOD
    2. Record:  ITMP
                                                  Boundary
    3. Record:  Layer       Row       Column        Head        Cond
       (Input item 3 normally consists of one record for each GHB.
       If ITMP is negative or zero, item 3 is not read.)
```

MXBND--is the maximum number of general-head boundary cells at one time.

IGHBCB--is a flag and a unit number.

If IGHBCB > 0, unit number for cell-by-cell flow terms.

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

If IGHBCB < 0, boundary leakage for each cell will be printed whenever ICBCFL is set.

ITMP--is a flag and a counter.

If ITMP < 0, GHB data from the preceding stress period will be reused.

If ITMP $\geq$ 0, ITMP is the number of general-head boundaries during the current stress period.

Strongly Implicit Procedure Package Input

Input to the Strongly Implicit Procedure (SIP) Package is read from the unit specified in IUNIT(9).

```
FOR EACH SIMULATION
    1. Record:  MXITER     NPARM
    2. Record:  ACCL       HCLOSE      IPCALC      WSEED      IPRSIP
```

IPCALC--is a flag indicating where the iteration parameter seed will come from.

0 - the seed will be entered by the user.

1 - the seed will be calculated at the start of the simulation from problem parameters.

IPRSIP--is the printout interval for SIP.

Slice-Successive Overrelaxation Package Input

Input to the Slice-Successive Overrelaxation (SOR) Package is read from the unit specified in IUNIT(11).

```
FOR EACH SIMULATION
            1. Record: MXITER
            2. Record: ACCL     HCLOSE    IPRSOR
```

IPRSOR--is the printout interval for SOR.