APPENDIX

MODIFICATIONS TO MODBRANCH INPUT FILES AND SOURCE CODE

Only a small alteration of the MODBRANCH input file format is necessary for use of the version of the model modified to use the reach-transmissivity leakage relationship. The additional input data required consist of a third line of initial condition data for each channel segment; this new line follows the existing two lines. The input format for this third line is:

Data	—	LSTRM (1, MAXS)	LSTRM (2, MAXS)	LSTRM (3, MAXS)	LLK (MAXS)	RSTRM (1, MAXS)	RSTRM (2, MAXS)	RSTRM (3, MAXS)	RLK (MAXS)
Format	—	I3	I3	I3	F10.4	I3	I3	I3	F10.4

These variables have the following definitions, in which left and right sides of the channel are defined when facing in the direction of flow:

LSTRM (1, MAXS)	—	Layer number of model cell from which left side ground-water reference head is obtained for this channel segement
LSTRM (2, MAXS)	—	Row number of model cell from which left side ground-water reference head is obtained for this channel segment
LSTRM (3, MAXS)	—	Column number of model cell from which left side ground-water reference head is obtained for this channel segment
LLK (MAXS)	—	Reach-transmissivity leakage coefficient for left side of channel segement
RSTRM (1, MAXS)	—	Layer number of model cell from which right side ground-water reference head is obtained for this channel segment
RSTRM (2, MAXS)	—	Row number of model cell from which right side ground-water reference head is obtained for this channel segment
RSTRM (3, MAXS)	—	Column number of model cell from which right side ground-water reference head is obtained for this channel segment
RLK (MAXS)	—	Reach-transmissivity leakage coefficient for right side of channel segment

The incorporation of the reach-transmissivity relationship into MODBRANCH required modification of several of the program's subroutines; all of these subroutines were part of the BRANCH′ (Swain and Wexler, 1996) source code. The modified versions of the subroutines are presented on the subsequent pages. Alterations from the original version appear in bold type. Lines of code that have been replaced appear in strikethrough.

The subroutine BRICXS reads the initial conditions and vertical flow leakage coefficients of the surface-water channel. An additional read statement was added to allow input of the reach-transmissivity leakage coefficients and specification of the model cells from which the reference ground-water heads are obtained.

```
SUBROUTINE BRICXS

C     THIS SUBROUTINE READS IN THE INITIAL CONDITION AND GEOMETRY
RECORDS
c**********************************************************************
*
c Modified to include reach transmissivity relationship parameters
c**********************************************************************
*

      INCLUDE 'dimens.cmn'
      INCLUDE 'comcon.cmn'
      INCLUDE 'lunums.cmn'
      INCLUDE 'dtcomp.cmn'
      INCLUDE 'xsinit.cmn'
      INCLUDE 'xsichr.cmn'
      INCLUDE 'geom.cmn'
      INCLUDE 'ceta.cmn'
      INCLUDE 'branch.cmn'
      INCLUDE 'branam.cmn'
      INCLUDE 'dtypes.cmn'
      INCLUDE 'units.cmn'
      INCLUDE 'logics.cmn'
      INCLUDE 'limits.cmn'
      INCLUDE 'volcom.cmn'
      CHARACTER*10 zijchr, qijchr, tenblk
      CHARACTER msg*90, cdmy*4, fvflg*2
      INTEGER i, ns, ij, j, k, nd, nxs
      REAL qij, cdatum, stg1, stg2
      EXTERNAL CHKSTA, SETAB1, ERRSCR, ERRPAR, ERRPRT, ERRLIN, GETCHR
      REAL ABS
      INTRINSIC ABS
      DATA tenblk/'          '/
C
      Error = .FALSE.
      Xskt(1) = MAXS
      Tabeta = MAXS.EQ.MXTFXS
      Nfvxs = 0
      DO 50 i = 1, Nbch
        j = 0
        READ (Luinit,9015,END=100,ERR=110) Ijf(i), Ijt(i), Nsec(i),
     &       Brname(i), Ijfen(i), Ijten(i), Prtbch(i), Prtsum(i),
     &       Ppltbh(i), Pltbch(i), Prtxsg(i)
        IF (Ijf(i).LT.1 .OR. Ijf(i).GT.Njnc) THEN
          WRITE (msg,9070) 'F', Ijf(i), 'SOURCE', i, Njnc
          GOTO 140
        ENDIF
```

```
        IF (Ijt(i).LT.1 .OR. Ijt(i).GT.Njnc) THEN
          WRITE (msg,9070) 'T', Ijt(i), 'OUTLET', i, Njnc
          GOTO 140
        ENDIF
        WRITE (*,9005) i, Ijf(i), Ijt(i), Brname(i)
        IF (i.GT.1) Xskt(i) = Xskt(i-1) - Nsec(i-1)
        ns = Nsec(i)
        ij = MAXS - Xskt(i)
        DO 40 j = 1, ns
          ij = ij + 1
          IF (ij.GT.MAXS) THEN
            nxs = MAXS - Xskt(i) + ns
            CALL ERRPAR(1,nxs,2,MAXS,' MAXS ',
     &                 'NUMBER OF CROSS SECTIONS INPUT EXCEEDS
MAXIMUM')
          ENDIF
          IF (j.EQ.ns) THEN
            Dx(ij) = 0.0
            READ (Luinit,9030,END=90,ERR=120) Z(ij), Q(ij), Rho(ij),
     &            zijchr, qijchr
          ELSE
            READ (Luinit,9020,END=90,ERR=120) Z(ij), Q(ij), Rho(ij),
     &            Dx(ij), T(ij), (Rn(k,ij),k=1,3), zijchr, qijchr
            IF (ABS(Dx(ij)).LT.SMLVL) Dx(ij) = 0.00001
            Rn4(ij) = Rn(1,ij)
            IF (ABS(T(ij)).LT.SMLVL) T(ij) = H2otmp
          ENDIF
C
C---ASSIGN SMALL INITIAL-CONDITION STAGE AND DISCHARGE VALUES IN
C---PLACE OF ZERO VALUES FOR SUBSEQUENT ASSIGNMENT OF UNSPECIFIED
C---INITIAL CONDITIONS
          IF (ABS(Z(ij)).LT.SMLVL .AND. zijchr.NE.tenblk) Z(ij) =
0.0001
          IF (ABS(Q(ij)).LT.SMLVL .AND. qijchr.NE.tenblk) Q(ij) =
0.01
          IF (ABS(Q(ij)).GT.SMLVL) THEN
            qij = ABS(Q(ij)*0.005)
            IF (Qtol.GT.qij) Qtol = qij
          ENDIF
          READ (Luinit,9025,END=90,ERR=120) Orient(ij), Betvel(ij),
     &          Qlat(ij), Ulat(ij)
          IF (Imodfw.NE.0) THEN
            READ (Luinit,9060,END=90,ERR=120) Istrm(3,ij),
Istrm(2,ij),
     &            Istrm(1,ij), Clk(ij), Zbot(ij)
            IF (Istrm(3,ij).LT.0) Lstruc = .TRUE.
```

```
c***Read reach transmissivity parameters
            READ (Luinit,9061,END=90,ERR=120)Lstrm(3,ij),Lstrm(2,ij),
     &            Lstrm(1,ij),Llk(ij),Rstrm(3,ij),Rstrm(2,ij),
     &            Rstrm(1,ij),Rlk(ij)

          ENDIF
          IF (Rho(ij).LE.0.0) Rho(ij) = H2oden
          IF (Betvel(ij).LT.1.0 .AND. Istrm(3,ij).GE.0) Betvel(ij)
     &        = Glbeta
C
C---READ GEOMETRY RECORDS
          READ (Lugeom,9035,END=70,ERR=80) Ipt(ij), Xstatn(ij),
     &          Gdatum(ij), fvflg, (Itypeo(k,ij),k=1,4)
          CALL CHKSTA(Xstatn(ij))
          IF (fvflg.EQ.'FV' .OR. fvflg.EQ.'fv') THEN
            Nfvxs = Nfvxs + 1
            Ijvol(Nfvxs) = ij
            Fvol(Nfvxs) = 0.0
            Istapr(Nfvxs) = Xstatn(ij)
          ENDIF
C
C---CHECK STATION NUMBER AND DATA TYPES SPECIFIED FOR TDDB OUTPUT
          IF (Ottddb) THEN
            IF (Xstatn(ij)(15:16).EQ.Iblk) THEN
              IF (Itypeo(1,ij).EQ.Iblk .AND. Itypeo(2,ij).EQ.Iblk
.AND.
     &            Itypeo(3,ij).EQ.Iblk .AND. Itypeo(4,ij).EQ.Iblk)
     &            GOTO 20
              Error = .TRUE.
              msg = 'INVALID STATION NUMBER ('//Xstatn(ij)
     &              //') SPECIFIED'
              CALL ERRSCR(0,i,j,msg)
            ENDIF
            DO 10 k = 1, 4
              Dtype = Itypeo(k,ij)
              IF (Dtype.NE.Iblk .AND. Dtype.NE.Qtype .AND.
     &            Dtype.NE.Ztype .AND. Dtype.NE.Atype .AND.
     &            Dtype.NE.Btype) THEN
                Error = .TRUE.
                msg = 'INVALID DATA TYPE ('//Dtype//') SPECIFIED'
                CALL ERRSCR(0,i,j,msg)
              ENDIF
   10       CONTINUE
          ENDIF
```

```
C
C---INITIALIZE FIRST FORWARD VALUES
   20     Zp(ij) = Z(ij)
          Qp(ij) = Q(ij)
          nd = Ipt(ij)
          cdatum = Gdatum(ij)
          IF (nd.LT.2 .OR. nd.GT.MXPT) THEN
            WRITE (msg,9050) i, j
            CALL ERRPAR(1,nd,2,MXPT,' MXPT ',msg)
          ENDIF
          IF (Tabeta) THEN
            READ (Lugeom,9040,END=70,ERR=80)
     &           (Za(k,ij),Aa(k,ij),Bb(k,ij),Bs(k,ij),Wp(k,ij),
     &           Eta(k,ij),Qa(k,ij),Ta(k,ij),k=1,nd)
          ELSE
            READ (Lugeom,9045,END=70,ERR=80)
     &           (Za(k,ij),Aa(k,ij),Bb(k,ij),Bs(k,ij),Wp(k,ij),k=
1,nd)
          ENDIF
          stg1 = Za(1,ij)
          Storar(1,ij) = 0.0
          DO 25 k = 2, nd
            stg2 = Za(k,ij)
            Storar(k,ij) = Storar(k-1,ij) +
     &                     0.5*(Bs(k,ij)+Bs(k-1,ij))*(stg2-stg1)
            stg1 = stg2
   25     CONTINUE
          IF (Wp(nd,ij).GT.SMLVL) THEN
            Rrflg(ij) = .TRUE.
            IF (Wp(nd,ij).LT.0.1) THEN
              WRITE (Printr,9065) i, j
              WRITE (*,9065) i, j
              CALL GETCHR
     &        (' Enter a carriage return to continue or "Q" to
quit',cdmy)
            ENDIF
          ELSE
            Rrflg(ij) = .FALSE.
          ENDIF
          IF (j.GT.1 .AND. (Rrflg(ij).AND..NOT.Rrflg(ij-1).OR.
     &         .NOT.Rrflg(ij).AND.Rrflg(ij-1))) THEN
            msg =
     &'CANNOT MIX USE OF HYDRAULIC RADIUS AND HYDRAULIC DEPTH WITHIN A
B
     &RANCH.'
            GOTO 130
          ENDIF
```

```
          Zbot(ij) = Zbot(ij) + cdatum
          Za(1,ij) = Za(1,ij) + cdatum
          DO 30 k = 2, nd
            Za(k,ij) = Za(k,ij) + cdatum
            IF (Za(k-1,ij).GE.Za(k,ij)) THEN
              WRITE (msg,9055) Za(k-1,ij), Za(k,ij)
              GOTO 130
            ENDIF
   30     CONTINUE
          IF (Typeta.NE.1 .AND. Tabeta .AND. ABS(Eta(1,ij)).GT.SMLVL)
     &         THEN
            IF (ABS(Eta(1,ij)).GT.1.0) WRITE (Printr,9010) i, j
            CALL SETAB1(ij)
          ENDIF
          IF (Za(1,ij).LT.Ztmin) Ztmin = Za(1,ij)
          IF (Za(nd,ij).GT.Ztmax) Ztmax = Za(nd,ij)
C
C---CHECK CONSTANT ETA SPECIFIED AND ASSIGN GLOBAL DEFAULT IF GIVEN
          IF (j.NE.ns) THEN
            IF (.NOT.(Typeta.NE.1 .OR. ABS(Rn4(ij)).GT.SMLVL .OR.
     &           (Tabeta .AND. ABS(Eta(1,ij)).GT.SMLVL))) THEN
              IF (ABS(Gleta).LT.SMLVL) CALL ERRSCR(1,i,j,
     & 'CONSTANT ETA INDICATED, BUT NO GLOBAL OR LOCAL VALUE
SPECIFIED')
              Rn4(ij) = Gleta
            ENDIF
          ENDIF
   40   CONTINUE
   50 CONTINUE
      IF (Error) STOP
      DO 60 k = 1, ij
        Sumeta(k) = 0.0
        Sumczq(k) = 0.0
        Sczqsq(k) = 0.0
        Szqeta(k) = 0.0
        Indxs(k) = 2
   60 CONTINUE
      Dtype = Ztype
      Zdatum = (Ztmax+Ztmin)*0.5
      IF (ABS(Qtol-BIGPOS).LT.0.00005) Qtol = 1.0
      IF (Qqtol.LT.SMLVL) Qqtol = Qtol
      WRITE (*,'(/,A)') ' Initial-condition and geometry read end!'
      RETURN
   70 msg = 'END-OF-FILE READING GEOMETRY RECORDS'
      GOTO 130
   80 msg = 'IMPROPER FORMAT READING GEOMETRY RECORDS'
      GOTO 130
```

```
   90 msg = 'END-OF-FILE READING INITIAL CONDITIONS'
      GOTO 130
  100 msg = 'END-OF-FILE READING BRANCH IDENTIFICATION RECORD'
      GOTO 130
  110 msg = 'IMPROPER FORMAT READING BRANCH IDENTIFICATION RECORD'
      GOTO 130
  120 msg = 'IMPROPER FORMAT READING INITIAL CONDITION RECORDS'
  130 CALL ERRSCR(1,i,j,msg)
  140 CALL ERRPRT(msg)
      WRITE (msg,'(A,I3)') 'MUST BE GREATER THAN ZERO AND LESS THAN',
     &                     Njnc+1
      CALL ERRLIN(msg)
      STOP
C
 9005 FORMAT (' BRANCH',I3,' FROM JUNCTION',I3,' TO',I3,' : ',A40)
 9010 FORMAT (/,
     &' *WARNING* POSSIBLE ERROR INVOLVING USE OF TABULAR ETA, ETA
VALUE
     & > 1.0 FOUND.',/,11X,'FOR BRANCH',I3,' SECTION',I3,
     &' ETA VALUES SHOULD BE IN COLUMNS 51-60.',/,11X,
     &'NOTE, THE TABULAR ETA FIELD WAS REASSIGNED FOR MODEL
VERSIONS',/,
     &11X,'AFTER 09/26/95.')
 9015 FORMAT (3I2,A40,27X,7I1)
 9020 FORMAT (2F10.3,F10.4,2F10.2,3E10.4,T1,2A10)
 9025 FORMAT (2F10.3,2F10.4)
 9030 FORMAT (2F10.3,F10.4,T1,2A10)
 9035 FORMAT (I2,1X,A16,44X,F7.3,5A2)
 9040 FORMAT (5F10.3,E10.4,2F10.3)
 9045 FORMAT (5F10.3)
 9050 FORMAT ('NUMBER OF DATA POINTS FOR CROSS SECTION',I2,', BRANCH',
     &        I3,' EXCEEDS MAXIMUM')
 9055 FORMAT ('DUPLICATE OR OUT-OF-RANGE STAGES IN GEOMETRY TABLE,',
     &        F8.3,'>=',F8.3)
 9060 FORMAT (3I3,2F10.4)
 9061 format (3i3,f10.4,3i3,f10.4)
 9065 FORMAT (/,
     &' *WARNING* POSSIBLE ERROR ON GEOMETRY RECORDS, WETTED PERIMETER
<
     & 0.1 FOUND.',/,11X,'FOR BRANCH',I3,' SECTION',I3,
     &' WETTED PERIMETER IS READ FROM COLUMNS',/,11X,
     &'41-50 AND ETA VALUES FROM COLUMNS 51-60.  NOTE, INPUT OF
WETTED',
     &/,11X,
     &'PERIMETER WAS ADDED AND THE TABULAR ETA FIELD MOVED FOR MODEL'
     &,/,11X,'VERSIONS AFTER 09/26/95.',/)
```

```
 9070 FORMAT ('INVALID JUNCTION NUMBER (IJ',A1,' =',I3,
     &        ') SPECIFIED FOR ',A6,' OF POSITIVE FLOW FOR
BRANCH',I3,/,
     &        ' MUST BE GREATER THAN ZERO AND LESS THAN',I3)
      END
```

The subroutines LEAKVL, AQFRHD, and AQRHEAD were modified to include additional variables required for the matrix BRANCH' uses to solve the finite-difference equations.

```
SUBROUTINE LEAKVL(NCOL,NROW,NLAY,HNEW,HOLD,IBOUND,KKITER)
      INTEGER NCOL, NLAY, NROW
      DOUBLE PRECISION HNEW(NCOL,NROW,NLAY)
      REAL HOLD(NCOL,NROW,NLAY)
      INTEGER IBOUND(NCOL,NROW,NLAY)
      INTEGER itria1, itria2, itria3, KKITER
      INCLUDE 'dimens.cmn'
      INCLUDE 'branch.cmn'
      INCLUDE 'comcon.cmn'
      INCLUDE 'xsinit.cmn'
      INCLUDE 'dtcomp.cmn'
      INCLUDE 'modbrn.cmn'
      REAL onechi
      INTEGER i, j, nsm1, ij, ijfi, ijp1, ijti, mod1, mod2, mod3
      EXTERNAL AQFRHD, AQRHEAD, AQRDRY
      INTEGER MOD
      INTRINSIC MOD
C
      onechi = (1.0-Chi)*0.5
C
C14A--THE NEW AND OLD AQUIFER HEADS ARE USED TO INTERPOLATE THE
C     HEAD AT THE PRESENT BRANCH TIMESTEP (HAQ) FOR CALCULATING
C     THE TOTAL VOLUME OF LEAKAGE.
C
      DO 20 i = 1, Nbch
        nsm1 = Nsec(i) - 1
        ij = MAXS - Xskt(i)
        ijfi = MAXS + Ijf(i)
        ijp1 = ij + 1
        Qlsum(ijfi) = Qlsum(ijfi) - Qp(ijp1)/Ntsaq
        DO 10 j = 1, nsm1
          ij = ij + 1
          ijp1 = ij + 1
          IF (Istrm(3,ij).LT.0) GOTO 10
          CALL AQFRHD(ij,NCOL,NROW,NLAY,HNEW,HOLD,IBOUND)
          CALL AQRDRY(ij)
```

```
c****Old Qlsum calculation*****************************************
~~      Qlsum(ij) = Qlsum(ij)~~
~~     &                  + (Chi*(Clkpp1*Bp(ijp1)*(Zp(ijp1)Haqpp1)~~
~~     &                 +Clkp*Bp(ij)*(Zp(ij) Haqp))~~
~~     &                 +onechi*(Clkp1*B(ijp1)*(Z(ijp1) Haqp1)~~
~~     &                 +Clkij*B(ij)*(Z(ij) Haq)))*Dx(ij)/(2.*Ntsaq)~~

c*****New Qlsum calculation****************************************
           Qlsum(ij) = Qlsum(ij)
     &                 + (Chi*(Rlkpp1*(Zp(ijp1)-Raqpp1)
     &                 +Rlkp*(Zp(ij)-Raqp))
     &                 +onechi*(Rlkp1*(Z(ijp1)-Raqp1)
     &                 +Rlkij*(Z(ij)-Raq)))*Dx(ij)/(2.*Ntsaq)
     &                 + (Chi*(Llkpp1*(Zp(ijp1)-Laqpp1)
     &                 +Llkp*(Zp(ij)-Laqp))
     &                 +onechi*(Llkp1*(Z(ijp1)-Laqp1)
     &                 +Llkij*(Z(ij)-Laq)))*Dx(ij)/(2.*Ntsaq)

           IF (KKITER.LT.51) THEN
             CALL AQRHEAD(ij,NCOL,NROW,NLAY,HNEW,HOLD)
C
C     THE DRYNESS CONDITIONS ARE CHECKED TO SET VALUES OF ITRIAL
             itria1 = 0
             IF (Zp(ij).LT.Zbot(ij)-Zdatum) itria1 = itria1 + 10
             IF (Zp(ijp1).LT.Zbot(ijp1)-Zdatum) itria1 = itria1 + 20
             IF (Haqp.LT.Zbot(ij)) itria1 = itria1 + 1
Crsr should above be haqp1, since haqp=haqpp1
Crsr as used in next statement???
             IF (Haqpp1.LT.Zbot(ijp1)) itria1 = itria1 + 2
             itria2 = Itrial(Icount+1,ij)
             itria3 = Itrial(Icount,ij)
             mod1 = MOD(itria1,10)
             mod2 = MOD(itria2,10)
             mod3 = MOD(itria3,10)
             IF (itria3.GE.30) THEN
               IF (itria3.LT.80) THEN
                 IF (itria1.GE.30) itria1 = mod1 + itria3 - mod3
               ELSEIF (itria1.GE.30) THEN
                 itria1 = mod1 + itria3 - mod3 + 10
               ELSE
                 itria1 = mod1 + itria3 - mod3 - 10
               ENDIF
             ENDIF
             IF (KKITER.LE.5 .OR. (itria1-mod1)/10.EQ.mod2 .OR.
     &           itria1-mod1.LT.itria3-mod3) Itrial(Icount+1,ij) =
itria1
```

```
            Itrial(Icount+1,ij) = Itrial(Icount+1,ij)
     &                            - MOD(Itrial(Icount+1,ij),10) + mod1
          ENDIF
   10   CONTINUE
        ijti = MAXS + Ijt(i)
        Qlsum(ijti) = Qlsum(ijti) + Qp(ijp1)/Ntsaq
   20 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE AQFRHD(IJ,NCOL,NROW,NLAY,HNEW,HOLD,IBOUND)
C
C---THE NEW AND OLD AQUIFER HEADS ARE USED TO INTERPOLATE THE HEAD
C---AT THE PRESENT BRANCH TIMESTEP (HAQ).
C
      INTEGER IJ, NCOL, NROW, NLAY
      DOUBLE PRECISION HNEW(NCOL,NROW,NLAY)
      REAL HOLD(NCOL,NROW,NLAY)
      INTEGER IBOUND(NCOL,NROW,NLAY)
      INCLUDE 'dimens.cmn'
      INCLUDE 'xsinit.cmn'
      INCLUDE 'modbrn.cmn'
      INCLUDE 'comcon.cmn'
      EXTERNAL AQRHEAD
      REAL AMAX1
      INTRINSIC AMAX1
C
      Clkij = Clk(IJ)
      IF (IBOUND(Istrm(1,IJ),Istrm(2,IJ),Istrm(3,IJ)).LT.1) Clkij = 0.0
      Clkij = Clkij
      Clkp1 = Clkij
      Clkp = Clkij
      Clkpp1 = Clkij
C
****Assign values to reach transmissivity leakage coefficients
      Rlkij = Rlk(IJ)
      IF (IBOUND(Rstrm(1,IJ),Rstrm(2,IJ),Rstrm(3,IJ)).LT.1) Rlkij = 0.0
      Rlkij = Rlkij
      Rlkp1 = Rlkij
      Rlkp = Rlkij
      Rlkpp1 = Rlkij
C
      Llkij = Llk(IJ)
      IF (IBOUND(Lstrm(1,IJ),Lstrm(2,IJ),Lstrm(3,IJ)).LT.1) Llkij = 0.0
      Llkij = Llkij
      Llkp1 = Llkij
      Llkp = Llkij
      Llkpp1 = Llkij
```

```
C
      CALL AQRHEAD(IJ,NCOL,NROW,NLAY,HNEW,HOLD)
C
C---IF THE AQUIFER IS BELOW THE STREAMBED, HEAD IN STREAM IS USED
C---TO CALCULATE LEAKAGE.
      Haq = AMAX1(Haq,Zbot(IJ))
      Haqp = AMAX1(Haqp,Zbot(IJ))
      Haqp1 = AMAX1(Haqp1,Zbot(IJ+1))
      Haqpp1 = AMAX1(Haqpp1,Zbot(IJ+1))
      Haq = Haq - Zdatum
      Haqp = Haqp - Zdatum
      Haqp1 = Haqp1 - Zdatum
      Haqpp1 = Haqpp1 - Zdatum
C
c***Assing left and right cell heads in R-T relationship
      raq = AMAX1(raq,Zbot(IJ))
      raqp = AMAX1(raqp,Zbot(IJ))
      raqp1 = AMAX1(raqp1,Zbot(IJ+1))
      raqpp1 = AMAX1(raqpp1,Zbot(IJ+1))
      raq = raq - Zdatum
      raqp = raqp - Zdatum
      raqp1 = raqp1 - Zdatum
      raqpp1 = raqpp1 - Zdatum
C
      laq = AMAX1(laq,Zbot(IJ))
      laqp = AMAX1(laqp,Zbot(IJ))
      laqp1 = AMAX1(laqp1,Zbot(IJ+1))
      laqpp1 = AMAX1(laqpp1,Zbot(IJ+1))
      laq = laq - Zdatum
      laqp = laqp - Zdatum
      laqp1 = laqp1 - Zdatum
      laqpp1 = laqpp1 - Zdatum
C
      RETURN
      END
C
C
      SUBROUTINE AQRHEAD(IJ,NCOL,NROW,NLAY,HNEW,HOLD)
C
C---THE NEW AND OLD AQUIFER HEADS ARE USED TO INTERPOLATE THE HEAD
C---AT THE PRESENT BRANCH TIMESTEP (HAQ).
C
      INTEGER NCOL, NLAY, NROW, IJ
      DOUBLE PRECISION HNEW(NCOL,NROW,NLAY)
      REAL HOLD(NCOL,NROW,NLAY), hnew1, hold1
      real hnewr, holdr, hnewl, holdl
```

```
      INCLUDE 'dimens.cmn'
      INCLUDE 'xsinit.cmn'
      INCLUDE 'modbrn.cmn'
      REAL FLOAT
      INTRINSIC FLOAT
C
      hnew1 = HNEW(Istrm(1,IJ),Istrm(2,IJ),Istrm(3,IJ))
      hold1 = HOLD(Istrm(1,IJ),Istrm(2,IJ),Istrm(3,IJ))
      Haq = hold1 + FLOAT(Icount-1)*(hnew1-hold1)/FLOAT(Ntsaq)
      Haqp = hold1 + FLOAT(Icount)*(hnew1-hold1)/FLOAT(Ntsaq)
      Haqp1 = Haq
      Haqpp1 = Haqp
C
c***Assign aquifer heads for R-T relationship
      hnewr = HNEW(Rstrm(1,IJ),Rstrm(2,IJ),Rstrm(3,IJ))
      holdr = HOLD(Rstrm(1,IJ),Rstrm(2,IJ),Rstrm(3,IJ))
      Raq = holdr + FLOAT(Icount-1)*(hnewr-holdr)/FLOAT(Ntsaq)
      Raqp = holdr + FLOAT(Icount)*(hnewr-holdr)/FLOAT(Ntsaq)
      Raqp1 = Raq
      Raqpp1 = Raqp
C
      hnewl = HNEW(Lstrm(1,IJ),Lstrm(2,IJ),Lstrm(3,IJ))
      holdl = HOLD(Lstrm(1,IJ),Lstrm(2,IJ),Lstrm(3,IJ))
      Laq = holdl + FLOAT(Icount-1)*(hnewl-holdl)/FLOAT(Ntsaq)
      Laqp = holdl + FLOAT(Icount)*(hnewl-holdl)/FLOAT(Ntsaq)
      Laqp1 = Laq
      Laqpp1 = Laqp
C
      RETURN
      RETURN
      END
```

The subroutine SOLVER was modified to employ the finite difference equations developed for the reach-transmissivity relationship instead of the vertical flow relationship.

```
SUBROUTINE SOLVER(MMM,NSTP,ISS,IBRPRN,KKITER,KKPER,NCOL,NROW,NLAY,
     &                  HNEW,HOLD,IBOUND)
C
C     + + + DUMMY ARGUMENTS + + +
      INTEGER MMM, NSTP, ISS, IBRPRN, KKITER, KKPER, NCOL, NROW, NLAY
      DOUBLE PRECISION HNEW(NCOL,NROW,NLAY)
      REAL HOLD(NCOL,NROW,NLAY)
      INTEGER IBOUND(NCOL,NROW,NLAY)
C
C     + + + DUMMY ARGUMENT DEFINITIONS + + +
C     MMM    - Number of first time step to solve
C     NSTP   - Number of time steps to solve
```

```
C
C     + + + Arguments used in computle MODFLOW/BRANCH simulations + + +
C     ISS    - Steady-state flag
C     IBRPRN - Print flag
C     KKITER - MODFLOW iteration count
C     KKPER  - MODFLOW stress period number
C     NCOL, NROW, NLAY - Number of columns, rows, and layers in
MODFLOW grid
C     HNEW, HOLD - Current and previous heads for the MODFLOW grid
C     IBOUND - Boundary array for MODFLOW grid
C
      INCLUDE 'dimens.cmn'
      INCLUDE 'dtcomp.cmn'
      INCLUDE 'matrix.cmn'
      INCLUDE 'branch.cmn'
      INCLUDE 'xsinit.cmn'
      INCLUDE 'cploop.cmn'
      INCLUDE 'modbrn.cmn'
      INCLUDE 'geom.cmn'
      INCLUDE 'ceta.cmn'
      INCLUDE 'comcon.cmn'
      INCLUDE 'untmes.cmn'
      INCLUDE 'units.cmn'
      INCLUDE 'boundy.cmn'
      INCLUDE 'mdatas.cmn'
      INCLUDE 'lunums.cmn'
      INCLUDE 'logics.cmn'
      INCLUDE 'bctime.cmn'
      INCLUDE 'datime.cmn'
      INCLUDE 'wind.cmn'
      INCLUDE 'nodal.cmn'
      INCLUDE 'limits.cmn'
      INCLUDE 'partim.cmn'
C
      DOUBLE PRECISION u(MAXS2), uu(MAXS4), bu(MXBH2), buu(MXBH4)
      DOUBLE PRECISION c1, c2, c3, c4, uuijp1, uuijp2, uuijp3, uuijp4
C
C---CONSTANTS AND COEFFICIENTS
      REAL betcor, thetdx, wratio, rnij, dazpij, dxij, dxiji
C
C---DEPENDENT VARIABLES
      REAL zij, qij, zijp1, qijp1, zp1mz, qp1mq, zp1pz, qp1pq
      REAL zpij, qpij, zpijp1, qpijp1, zpp1mz, qpp1mq, zpp1pz, qpp1pq
      REAL ztemp, qtemp, bigz, bigq, ztol
C
C---QUANTITIES AVERAGED IN SPACE AT CHI IN TIME STEP
      REAL zavg, qavg, ravg, bavg, btavg, aavg, aavgsq, aavgcu, aavgi
```

```
C
C---MATRIX COEFFICIENTS
      REAL lambda, sigma, mu, zeta, omega, gamma, delta, epslon, det,
     &     alpha, tmp
C
C---VARIABLES FOR SCREEN DISPLAY OF EPSILON TERMS
C     REAL ep1, ep2, ep3, ep4, ep5, ep6, ep7, ep8, ep9
C
C---LOCAL VARIABLES
      REAL totc, totm
      INTEGER ijp1, nsm1, jp1, ij2, ij4, ij4p1, ij4p2, ij4p3, ij4p4,
     &       ij2p1, ij2p2, i2, i4, i4p1, i4p2, i4p3, i4p4, i2p1, i2p2,
     &        mm, nnn, ibigz, jbigz, ibigq, jbigq, i, j, k, ij
      LOGICAL convrg
C
C---FUNCTIONS AND ROUTINES
      REAL SETA, SETAB
      INTEGER IAR
      EXTERNAL GETWND, DTPREP, ARBALL, AQFRHD, SETAB, DTOUT, DTMAP,
     &         TRACK, PRTPLT, DAILY, OPLOT, INTBC, GETBVD, GETNDF,
     &         EXTBC, GEMXP, OPTETA, LEAKVL, SETA, ERRSTP, DTFILE,
     &         WRTERM, AQRDRY
      INTEGER IFIX, MOD
      REAL ABS, SNGL
      DOUBLE PRECISION DBLE
      INTRINSIC IFIX, ABS, MOD, DBLE, SNGL
C
C     STATEMENT FUNCTION FOR LOCATING ELEMENTS IN COEFFICIENT MATRIX
      IAR(i,j,Ii) = i + Ii*(j-1)
C
C---BEGIN COMPUTATION LOOP
      Icount = 0
      convrg = .TRUE.
      M = MMM
   10 IF (IBRPRN.GT.0 .AND. Khr.EQ.24 .AND. .NOT.Oplots .AND.
ISS.EQ.0)
     &    WRITE (*,9005) Kyr, Kmo, Kda, Khr, Kmn
      Lastn = N
      IF (Lastn.GT.Nit) Lastn = Nit
      Kt = Kt + 1
      Icount = Icount + 1
C
C---ASSIGN WIND SPEED AND DIRECTION FROM TIME-DEPENDENT WIND INPUT
      IF (Inwind.NE.0) THEN
        k = IFIX((Nwd-1)*Dt/Wdtt+1.0001)
        IF (Morewd .AND. k.GT.Nwpart) THEN
          CALL GETWND(Prtmsg)
```

```
          k = 1
        ENDIF
        wratio = ((Nwd-1)*Dt-(k-1)*Wdtt)/Wdtt
        Wspeed = (Windsp(k)+wratio*(Windsp(k+1)-Windsp(k)))*Wsfact
        Wdirec = Winddr(k) + wratio*(Winddr(k+1)-Winddr(k))
        Cw = Cwfact*Wspeed*Wspeed
CC      cw = WSDRAG*AIRDEN/(H2ODEN*G)*WSPEED*WSPEED
      ENDIF
C
C---CALL ROUTINE TO FILE FLOW RESULTS FOR POST-PROCESS PLOTTING
      IF (Luout.NE.0) CALL DTFILE
C
C---PREPARE FOR NEXT TIME STEP
      CALL DTPREP(Kt)
C
C---WRITE NETCDF FILE
      IF (Otfile.EQ.4) CALL CDFOUT
C
C---BEGIN ITERATIVE IMPROVEMENT LOOP
   20 DO 120 N = 1, Nit
C
C---COMPUTE NEW GEOMETRY
        CALL ARBALL
C
C---IS THIS THE FIRST ITERATION (N=1) OF THIS TIME STEP (M) ?
        IF (N.EQ.1 .AND. .NOT.Modeta) THEN
          IF (IBRPRN.GT.0) THEN
C
C---CALL DTOUT TO PRINT FINAL RESULTS FOR LAST TIME STEP WHEN N EQUALS
C---ONE-- LASTN EQUALS NUMBER OF SOLUTIONS REQUIRED IN LAST TIME STEP
C---(Dtprt IS .TRUE. WHEN IPROPT .EQ. 0, 1, 5, 6, 7, OR 8)
C---(PRINTS TIME-STEP RESULTS WHEN IPROPT = 0, 5, OR 7; ONLY PRINTS
C---DAILY OR FINAL SUMMARY AS NEEDED WHEN IPROPT = 1, 6, OR 8)
            IF (Dtprt .AND. (Lprtdt .OR. MOD(M,Idtrat).EQ.1)) THEN
              CALL DTOUT(0,Lastn,Q,Z,A,B,Bt)
              IF (Lstruc) THEN
                DO 40 i = 1, Nbch
                  ij = MAXS - Xskt(i)
                  nsm1 = Nsec(i) - 1
                  DO 30 j = 1, nsm1
                    ij = ij + 1
                    IF (Istrm(3,ij).LT.0) Rn4(ij) = Rn(3,ij)
   30             CONTINUE
   40           CONTINUE
              ENDIF
            ENDIF
```

```
C
C---PRINT NONCONVERGENCE MESSAGE IF NO CONVERGENCE AND IWMOPT 0 OR 2
            IF (.NOT.convrg .AND. Iwmopt.EQ.0 .OR. Iwmopt.EQ.2) THEN
              convrg = .TRUE.
              WRITE (Printr,9010) Khr, Kmn, Kyr, Kmo, Kda, ibigz,
jbigz,
     &                            bigz, ibigq, jbigq, bigq
c             write(*,*) ztol, qtol, zztol, qqtol
            ENDIF
C
C---CALL OUTPUT ROUTINES FOR VARIOUS OPTIONS
            IF (Otfile.EQ.3) CALL DTMAP
            IF (Ipropt.EQ.9) CALL TRACK
            IF (Ptplt) THEN
              IF (Kt.GE.MAXCZQ .OR. Khr.EQ.24) THEN
                Ketime = Ietime + (M-1)*Idtm
                CALL PRTPLT
                Kt = 0
              ENDIF
            ELSEIF (Kt.GE.Idtpdy) THEN
              IF (Daysum) CALL DAILY
              Kt = 0
              IF (Oplots) THEN
                Ketime = Ietime + (M-1)*Idtm
                CALL OPLOT
              ENDIF
            ENDIF
          ENDIF
C
C---IF END OF TIME PERIOD THEN RETURN
          IF (M.EQ.NSTP) GOTO 140
C
C---INCREMENT DATE AND TIME FOR NEXT TIME STEP
C         Kmn = Kmn + Idtm
          Ksc = Ksc + Idts
          Kmn = Ksc/60
          IF (Kmn.GE.60) THEN
            Khr = Khr + Kmn/60
            Kmn = MOD(Kmn,60)
            Ksc = MOD(Ksc,3600)
          ENDIF
          IF (.NOT.(Khr.LT.24 .OR. (Khr.EQ.24 .AND. Kmn.EQ.0))) THEN
            Khr = Khr - 24
            Kda = Kda + 1
          ENDIF
          IF (Kda.GT.Dypmo(Kmo)) THEN
```

```
            Kda = Kda - Dypmo(Kmo)
            Kmo = Kmo + 1
          ENDIF
          IF (Kmo.GT.12) THEN
            Kmo = 1
            Kyr = Kyr + 1
            IF (Kyr.GT.99) Kyr = 0
            Dypmo(2) = 28 + (4-MOD(Kyr,4))/4
          ENDIF
C
C---CALL DTOUT TO PRINT INITIAL CONDITIONS FOR START OF NEXT TIME STEP
C---(Noprit IS .TRUE. WHEN IPROPT .NE. 1, 6, AND 8)
C---(ONLY CALLED WHEN IPROPT EQUALS 1, 6, OR 8)
C
          IF (.NOT.Noprit) THEN
            Lastn = N - 1
            IF (Lprtdt .OR. MOD(M+1,Idtrat).EQ.1)
     &          CALL DTOUT(1,Lastn,Qp,Zp,Ap,Bp,Btp)
          ENDIF
C
        ENDIF
C
C---COMPUTE EQUATION COEFFICIENTS AND CONSTRUCT COEFFICIENT MATRICES
        DO 80 i = 1, Nbch
          ij = MAXS - Xskt(i)
          nsm1 = Nsec(i) - 1
C
          DO 70 j = 1, nsm1
            jp1 = j + 1
            ij = ij + 1
            ijp1 = ij + 1
            dxij = Dx(ij)
            dxiji = 1.0/dxij
            thetdx = Theta*dxiji
            qij = Q(ij)
            zij = Z(ij)
            qijp1 = Q(ijp1)
            zijp1 = Z(ijp1)
            zpij = Zp(ij)
            qpij = Qp(ij)
            zpijp1 = Zp(ijp1)
            qpijp1 = Qp(ijp1)
            zp1mz = zijp1 - zij
            zp1pz = zijp1 + zij
            qp1mq = qijp1 - qij
            qp1pq = qijp1 + qij
```

```
            zpp1mz = zpijp1 - zpij
            zpp1pz = zpijp1 + zpij
            qpp1mq = qpijp1 - qpij
            qpp1pq = qpijp1 + qpij
            dazpij = Apzpij(ijp1) - Ap(ij)
            bavg = Chihf*(Bp(ij)+Bp(ijp1)) + Chi1hf*(B(ij)+B(ijp1))
            btavg = Chihf*(Btp(ij)+Btp(ijp1)) +
Chi1hf*(Bt(ij)+Bt(ijp1))
            aavg = Chihf*(Ap(ij)+Ap(ijp1)) + Chi1hf*(A(ij)+A(ijp1))
            aavgi = 1.0/aavg
            aavgsq = aavgi*aavgi
            ravg = Chihf*(Rp(ij)+Rp(ijp1)) + Chi1hf*(R(ij)+R(ijp1))
            qavg = Chihf*qpp1pq + Chi1hf*qp1pq
            zavg = Chihf*zpp1pz + Chi1hf*zp1pz + Zdatum
            betcor = (Betvel(ij)+Betvel(ijp1))*0.5
            aavgcu = aavgsq*aavgi*betcor*qavg*qavg
C
C---STRUCTURES (PUMPS, GATES, CULVERTS) ARE INCLUDED IN THE MATRIX OF
C---EQUATIONS
            IF (Istrm(3,ij).LT.0) THEN
              CALL
STRUC(ij,zpij,zpijp1,qpij,qpijp1,zij,zijp1,qij,qijp1,
     &                  Zdatum,G,IBRPRN,KKPER,NCOL,NROW,NLAY,HOLD,Z,
     &
Rn(3,ij),epslon,zeta,omega,gamma,alpha,delta,
     &                  det)
              GOTO 60
            ENDIF
C
C---SET FRICTION COEFFICIENT FROM INPUT CONSTANT, RELATION, OR TABLE
            IF (Tabeta .AND. Eta(1,ij).GT.SMLVL) THEN
              Rn4(ij) = SETAB(ij,zavg,qavg,aavg,ravg)
            ELSE
              IF (Typeta.NE.1) Rn4(ij) = SETA(ij,zavg,qavg,aavg,ravg)
            ENDIF
            rnij = Rn4(ij)
C
C---THE MATRIX COEFFICIENTS LAMBDA, SIGMA, MU, EPSLON, ZETA, OMEGA,
C---GAMMA, AND DELTA ARE COMPUTED FOR BRANCH AND ALPHA FOR MODFLOW.
            lambda = dxij*Dt2gth*aavgi
            sigma = ABS(qavg)*rnij*rnij*dxij*Chi2th*aavgsq/ravg**
     &              1.3333333
C
C---THE DRYNESS CONDITION IS CHECKED FOR FRICTION AND LEAKAGE
            IF (Modflow) THEN
              IF (ISS.NE.0) lambda = 0.0
```

```
C Old version of solution procedure
c   USE NEW AND OLD AQUIFER HEADS TO DETERMINE HEAD AT THIS TIME STEP
            CALL AQFRHD(ij,NCOL,NROW,NLAY,HNEW,HOLD,IBOUND)
            IF (Itrial(Icount+1,ij).GT.9) sigma = sigma*Dcfm64
            DO 50 k = 3, 8
              IF (Itrial(Icount+1,ij).GE.k*10) sigma = sigma*2.0
 50         CONTINUE
            CALL AQRDRY(ij)
            alpha = dxij*Chi*Clkp*Bp(ij)*Theta2i
            gamma = Chi*dxij*Clkpp1*Bp(ijp1)*Theta2i
            tmp =  Dtheta*qp1mq +
     &           dxij*(Chi*(Clkp*Haqp*Bp(ij)+Clkpp1*Haqpp1*Bp(ijp1))
     &            +Chi1hf*(Clkij*Haq*B(ij)+Clkp1*Haqp1*B(ijp1)))
     &            *Theta2i
            IF (ISS.EQ.0) THEN
              alpha = alpha + dxij*btavg*Twodtn*Thetai
              gamma = gamma + dxij*btavg*Dt2tha
              delta = tmp +
     &                (bavg*dxij*Dt2tha-Chi1hf*dxij*Clkp1*B(ijp1)
     &                *Theta2i)*Z(ijp1)
     &                + (bavg*dxij*Dt2tha-Chi1hf*dxij*Clkij*B(ij)
     &                *Theta2i)*Z(ij)
            ELSE
              delta = tmp -Chi1hf*dxij*Theta2i*(Clkp1*B(ijp1)*Z(ijp1)
     &                +Clkij*B(ij)*Z(ij))
            ENDIF
          ELSE
            gamma = dxij*btavg*Dt2tha
            delta = gamma*zp1pz  Dtheta*qp1mq + Thetai*dxij*Qlat(ij)
          ENDIF
          mu = betcor*qavg*Ginv2*aavgsq
          epslon = (lambda-Dchi*sigma)*qp1pq  Dtheta*(mu*qp1mq+zp1mz)
     &             + aavgcu*Gtheta*(bavg*zpp1mz+dazpij)
     &             + Cw*bavg*Wangle(ij)*dxij*Thetai*aavgi
     &             Denthi*ravg*0.5*(Rho(ijp1)-Rho(ij))
     &             + dxij*Gtheta*aavgi*Qlat(ij)*Ulat(ij)
          zeta = lambda + sigma + mu
          omega = lambda + sigma - mu
          det = 1.0/(1.0-zeta*gamma)
C
c*************************************************************
c   Begin modified code.
c*************************************************************
C---USE NEW AND OLD AQUIFER HEADS TO DETERMINE HEAD AT THIS TIME STEP
            CALL AQFRHD(ij,NCOL,NROW,NLAY,HNEW,HOLD,IBOUND)
            IF (Itrial(Icount+1,ij).GT.9) sigma = sigma*Dcfm64
```

```
              DO 50 k = 3, 8
                IF (Itrial(Icount+1,ij).GE.k*10) sigma = sigma*2.0
   50         CONTINUE
              CALL AQRDRY(ij)
              alpha = dxij*Chi*(Rlkp+Llkp)*Theta2i
              gamma = Chi*dxij*(Rlkpp1+Llkpp1)*Theta2i
              tmp = -Dtheta*qp1mq +
     &           dxij*(Chi*((Rlkp*Raqp+Llkp*Laqp)
     &           +(Rlkpp1*Raqpp1+Llkpp1*Laqpp1))
     &           +Chi1hf*((Rlkij*Raq+Llkij*Laq)
     &           +(Rlkp1*Raqp1+Llkp1*Laqp1)))
     &           *Theta2i
              IF (ISS.EQ.0) THEN
                alpha = alpha + dxij*btavg*Twodtn*Thetai
                gamma = gamma + dxij*btavg*Dt2tha
                delta = tmp +
     &                  (bavg*dxij*Dt2tha-Chi1hf*dxij*(Rlkp1+Llkp1)
     &                  *Theta2i)*Z(ijp1)
     &                  + (bavg*dxij*Dt2tha-Chi1hf*dxij*(Rlkij+Llkij)
     &                  *Theta2i)*Z(ij)
              ELSE
               delta = tmp - Chi1hf*dxij*Theta2i*((Rlkp1+Llkp)*Z(ijp1)
     &                  +(Rlkij+Llkij)*Z(ij))
              ENDIF
            ELSE
              gamma = dxij*btavg*Dt2tha
              delta = gamma*zp1pz - Dtheta*qp1mq +Thetai*dxij*Qlat(ij)
            ENDIF
            mu = betcor*qavg*Ginv2*aavgsq
            epslon = (lambda-Dchi*sigma)*qp1pq-Dtheta*(mu*qp1mq+zp1mz)
     &               + aavgcu*Gtheta*(bavg*zpp1mz+dazpij)
     &               + Cw*bavg*Wangle(ij)*dxij*Thetai*aavgi -
     &               Denthi*ravg*0.5*(Rho(ijp1)-Rho(ij))
     &               + dxij*Gtheta*aavgi*Qlat(ij)*Ulat(ij)
            zeta = lambda + sigma + mu
            omega = lambda + sigma - mu
            det = 1.0/(1.0-zeta*gamma)
C
C---PRINT GOVERNING EQUATION TERMS AT PRINTOUT FREQUENCY (IDTRAT)
C---PRINT GOVERNING EQUATION TERMS AT PRINTOUT FREQUENCY (IDTRAT)
C---EVERY TIME STEP AT CONVERGENCE (IPROPT=5) OR FOR EVERY SOLUTION
C---(IPROPT=6)
   60       IF (Ipropt.EQ.5 .OR. Ipropt.EQ.6) THEN
              IF (.NOT.(MOD(M,Idtrat).NE.0 .OR. (Ipropt.EQ.5 .AND.
N.NE.
     &           1))) THEN
                IF (N.EQ.1 .AND. i.EQ.1 .AND. j.EQ.1)
```

```
     &             WRITE (Printr,9015)
                 totc = Dzdt(ij) + Dqdxc(ij) - Qlat(ij)
                 totm = Dqdt(ij) + Dqdxm(ij) + Dadx(ij) + Dzdx(ij)
     &                + Fric(ij) + Wind(ij) + Dpdx(ij) + Qlatm(ij)
                 WRITE (Printr,9020) Dzdt(ij), Dqdxc(ij), Qlat(ij),
totc,
     &                              Dqdt(ij), Dqadx(ij), Dzdx(ij),
     &                              Fric(ij), Wind(ij), Dpdx(ij),
     &                              Qlatm(ij), totm
C---CALL ROUTINE TO FILE EQUATION TERMS FOR POST-PROCESS PLOTTING
                 IF (Luterm.NE.0) CALL WRTERM(i,j,ij)
               ENDIF
               Dzdt(ij) = btavg*Twodtn*(zpp1pz-zp1pz)
               Dqdxc(ij) = thetdx*(qpp1mq+Dtheta*qp1mq)
               Dqdt(ij) = thetdx*lambda*(qpp1pq-qp1pq)
               Dqdxm(ij) = thetdx*mu*(qpp1mq+Dtheta*qp1mq)
               Dadx(ij) = -Ginv*dxiji*aavgcu*(bavg*zpp1mz+dazpij)
               Dqadx(ij) = Dqdxm(ij) + Dadx(ij)
               Dzdx(ij) = thetdx*(zpp1mz+Dtheta*zp1mz)
               Fric(ij) = thetdx*sigma*(qpp1pq+Dchi*qp1pq)
               Wind(ij) = Cw*bavg*Wangle(ij)*aavgi
               Dpdx(ij) = H2odi*(ravg*0.5)*(Rho(ijp1)-Rho(ij))*dxiji
               Qlatm(ij) = Qlat(ij)*Ulat(ij)*Ginv*aavgi
C
C---PRINT VARIABLES AND COEFFICIENTS AT PRINTOUT FREQUENCY (IDTRAT)
C---EVERY TIME STEP AT CONVERGENCE (IPROPT=7) OR FOR EVERY SOLUTION
C---(IPROPT=8)
C
             ELSEIF (Ipropt.EQ.7 .OR. Ipropt.EQ.8) THEN
               IF (.NOT.(MOD(M,Idtrat).NE.0 .OR. (Ipropt.EQ.7 .AND.
N.NE.
     &           1))) THEN
                 WRITE (Printr,9025) N, i, j, ij, jp1, ijp1
                 WRITE (Printr,9030) zij, qij, zijp1, qijp1, zpij, qpij,
     &                               zpijp1, qpijp1
                 WRITE (Printr,9030) Apzpij(ijp1), A(ij), B(ij), R(ij),
     &                               A(ijp1), B(ijp1), R(ijp1)
                  WRITE (Printr,9030) Ap(ij), Bp(ij), Rp(ij),
Ap(ijp1),
     &                               Bp(ijp1), Rp(ijp1)
                 WRITE (Printr,9030) dxij, bavg, aavg, ravg, qavg,
     &                               betcor, rnij, Wangle(ij)
                 WRITE (Printr,9030) lambda, sigma, mu, epslon, zeta,
     &                               omega, gamma, delta
               ENDIF
             ENDIF
```

```
C
C---SEGMENT MATRIX COMPUTATION
            ij2 = (ij-1)*2
            ij2p1 = ij2 + 1
            ij2p2 = ij2 + 2
            ij4 = ij2*2
            ij4p1 = ij4 + 1
            ij4p2 = ij4 + 2
            ij4p3 = ij4 + 3
            ij4p4 = ij4 + 4
            i2 = (i-1)*2
            i4 = i2*2
            i4p1 = i4 + 1
            i4p2 = i4 + 2
            i4p3 = i4 + 3
            i4p4 = i4 + 4
            i2p1 = i2 + 1
            i2p2 = i2 + 2
            IF (Modflow) THEN
              uu(ij4p1) = DBLE((1.0+zeta*alpha)*det)
              uu(ij4p3) = DBLE((-alpha-gamma)*det)
            ELSE
              uu(ij4p1) = DBLE((1.0+zeta*gamma)*det)
              uu(ij4p3) = DBLE((-2.0*gamma)*det)
            ENDIF
            uu(ij4p2) = DBLE((-omega-zeta)*det)
            uu(ij4p4) = DBLE((1.0+omega*gamma)*det)
            u(ij2p1) = DBLE((epslon-zeta*delta)*det)
            u(ij2p2) = DBLE((delta-epslon*gamma)*det)
C
C---BRANCH MATRIX COMPUTATION
            IF (j.GT.1) THEN
              c1 = buu(i4p1)
              c2 = buu(i4p2)
              c3 = buu(i4p3)
              c4 = buu(i4p4)
              uuijp1 = uu(ij4p1)
              uuijp2 = uu(ij4p2)
              uuijp3 = uu(ij4p3)
              uuijp4 = uu(ij4p4)
              buu(i4p1) = uuijp1*c1 + uuijp2*c3
              buu(i4p2) = uuijp1*c2 + uuijp2*c4
              buu(i4p3) = uuijp3*c1 + uuijp4*c3
              buu(i4p4) = uuijp3*c2 + uuijp4*c4
              c1 = bu(i2p1)
              c2 = bu(i2p2)
              bu(i2p1) = uuijp1*c1 + uuijp2*c2 + u(ij2p1)
              bu(i2p2) = uuijp3*c1 + uuijp4*c2 + u(ij2p2)
```

```
            ELSE
              buu(i4p1) = uu(ij4p1)
              buu(i4p2) = uu(ij4p2)
              buu(i4p3) = uu(ij4p3)
              buu(i4p4) = uu(ij4p4)
              bu(i2p1) = u(ij2p1)
              bu(i2p2) = u(ij2p2)
            ENDIF
   70     CONTINUE
   80   CONTINUE
C
C---SET UP NETWORK MATRIX AND VECTOR
        Nn = 1
        mm = 1
        DO 90 i = 1, Nbch
C
C---INSERT BRANCH MATRICES
          nnn = IAR(Nn,mm,Ii)
          i2 = (i-1)*2
          i4 = i2*2
          Am(nnn) = SNGL(buu(i4+1))
          Am(nnn+Ii) = SNGL(buu(i4+2))
          Am(nnn+Ii2) = -1.0
          nnn = IAR(Nn+1,mm,Ii)
          Am(nnn) = SNGL(buu(i4+3))
          Am(nnn+Ii) = SNGL(buu(i4+4))
          Am(nnn+Ii3) = -1.0
C
C---CONSTRUCT RIGHT SIDE VECTOR
          Bmx(Nn) = SNGL(-bu(i2+1))
          Bmx(Nn+1) = SNGL(-bu(i2+2))
          Nn = Nn + 2
          mm = mm + 4
   90   CONTINUE
C
C---INSERT BOUNDARY CONDITIONS FOR INTERNAL JUNCTIONS
        CALL INTBC
C
C---RETRIEVE ADDITIONAL BOUNDARY-VALUE DATA FROM DATA BASE STORAGE
        IF (N.EQ.1 .AND. .NOT.Modeta) THEN
          IF (Morebd) THEN
            k = IFIX(Nd*Dt/Dtt(1)+1.0001)
            IF (k.GT.Ndpart) THEN
              CALL GETBVD
              Nd = IFIX(Dtmax/Dt+0.0001)
            ENDIF
          ENDIF
```

```
C
C---RETRIEVE ADDITIONAL NODAL FLOW DATA FROM DATA BASE STORAGE
          IF (Morenf) THEN
            k = IFIX(Nnd*Dt/Dttn(1)+1.0001)
            IF (k.GT.Nnpart) THEN
              CALL GETNDF(Dt,Prtmsg)
              Nnd = IFIX(Dtmxnf/Dt+0.0001)
            ENDIF
          ENDIF
        ENDIF
C
C---INSERT BOUNDARY CONDITIONS FOR EXTERNAL JUNCTIONS
        CALL EXTBC
C
C---CHECK FOR SQUARE MATRIX
        IF (Ii.NE.Nn-1) CALL ERRSTP(
     &'MATRIX NOT SQUARE: REVIEW SCHEMATIZATION AND EXTERNAL B.C.
SPECIF
     &ICATIONS')
C
C---SOLVE EQUATION MATRICES
        CALL GEMXP
        Ktmats = Ktmats + 1
C
C---COMPUTE INTERMEDIATE VALUES AT SEGMENT ENDS
        Nn = 1
        bigq = 0.0
        bigz = 0.0
        DO 110 i = 1, Nbch
          ij = MAXS - Xskt(i)
          ijp1 = ij + 1
          ztemp = Zp(ijp1)
          qtemp = Qp(ijp1)
          Zp(ijp1) = Bmx(Nn)
          Qp(ijp1) = Bmx(Nn+1)
          ztol = ABS(ztemp-Zp(ijp1))
          Qtol = ABS(qtemp-Qp(ijp1))
c         write(*,*) ztol,bigz, qtol,bigq
          IF (ztol.GT.bigz) THEN
            bigz = ztol
            ibigz = i
            jbigz = 1
          ENDIF
          IF (Qtol.GT.bigq) THEN
            bigq = Qtol
```

```
            ibigq = i
            jbigq = 1
          ENDIF
          Nn = Nn + 4
          nsm1 = Nsec(i) - 1
          DO 100 j = 1, nsm1
            ij = ij + 1
            ijp1 = ij + 1
            ij2 = (ij-1)*2
            ij4 = (ij-1)*4
            ztemp = Zp(ijp1)
            qtemp = Qp(ijp1)
            Zp(ijp1) =
SNGL(uu(ij4+1)*DBLE(Zp(ij))+uu(ij4+2)*DBLE(Qp(ij)
     &                 )+u(ij2+1))
            Qp(ijp1) =
SNGL(uu(ij4+3)*DBLE(Zp(ij))+uu(ij4+4)*DBLE(Qp(ij)
     &                 )+u(ij2+2))
            IF (Zp(ijp1).GT.Za(Ipt(ijp1),ijp1)) Zp(ijp1)
     &          = Za(Ipt(ijp1),ijp1)
            IF (Zp(ijp1).LT.Za(1,ijp1)) Zp(ijp1) = Za(1,ijp1)
            ztol = ABS(ztemp-Zp(ijp1))
            Qtol = ABS(qtemp-Qp(ijp1))
            IF (ztol.GT.bigz) THEN
              bigz = ztol
              ibigz = i
              jbigz = j + 1
            ENDIF
            IF (Qtol.GT.bigq) THEN
              bigq = Qtol
              ibigq = i
              jbigq = j + 1
            ENDIF
  100     CONTINUE
  110   CONTINUE
C
C---CHECK FOR CONVERGENCE
        IF (bigz.LE.Zztol .AND. bigq.LE.Qqtol) GOTO 130
C
C---CALL DTOUT TO PRINT RESULTS FOR LAST SOLUTION IF NOT CONVERGED
C---(Noprit IS .TRUE. WHEN IPROPT .NE. 1, 6, AND 8)
        IF (.NOT.Noprit .AND. N.NE.Nit .AND.
     &      (Lprtdt .OR. MOD(M+1,Idtrat).EQ.1))
     &      CALL DTOUT(1,N,Qp,Zp,Ap,Bp,Btp)
  120 CONTINUE
```

```
C
C---END ITERATIVE IMPROVEMENT LOOP
      IF (.NOT.Noconv) THEN
        IF (Iunit.EQ.En) THEN
          bigz = bigz*0.3048
          bigq = bigq*0.02832
        ELSE
          bigz = bigz*3.281
          bigq = bigq*35.31
        ENDIF
      ENDIF
      convrg = .FALSE.
  130 IF (ABS(Tolerr).GT.SMLVL) THEN
        CALL OPTETA
        IF (Modeta) GOTO 20
      ENDIF
      Nd = Nd + 1
      Nwd = Nwd + 1
      Nnd = Nnd + 1
      IF (Modflow) CALL LEAKVL(NCOL,NROW,NLAY,HNEW,HOLD,IBOUND,KKITER)
C
C---END COMPUTATION LOOP
      M = M + 1
      IF (M.LE.NSTP) GOTO 10
  140 RETURN
C
C---INPUT/OUTPUT FORMAT STATEMENTS
 9005 FORMAT (' SIMULATION TIME =',I3.2,2('/',I2.2),I3.2,':',I2.2)
 9010 FORMAT (
     &  ' *WARNING* MAXIMUM ITERATIONS COMPLETED WITHOUT CONVERGENCE
AT'
     &  ,I3.2,':',I2.2,' ON',I3.2,2('/',I2.2),5X,'Z-
ZP(',I2,',',I2,')=',
     &  F12.5,' Q-QP(',I2,',',I2,')=',F12.5)
 9015 FORMAT ('   dZ/dt      dQ/dx        q      MASS TOTAL   dQ/dt
',
     & 'd(Q*Q/A)/dx   dZ/dx       Sf         Sw        dp/dx       qu
'
     & ,'    MOM. TOTAL')
 9020 FORMAT (12E11.5E1)
 9025 FORMAT (1X,8I16)
 9030 FORMAT (1X,8E16.8)
      END
```

Finally, two of the BRANCH′ common files containing variable type declarations, MOD-BRN.CMN and XSINIT.CMN, had to be modified to include the additional variables needed for the previous subroutines.

```
C--Begin modbrn.cmn
c**** Modified to include reach transmissivity parameters
C    MAXS = MAX NO. OF INPUT CROSS SECTIONS
C    MXJN = MAX NO. OF JUNCTIONS IN NETWORK
C    100  = MAX NO. OF BRANCH TIME STEPS IN ONE MODFLOW TIME STEP
C
      REAL QLSUM(MAXS+MXJN), HAQ, HAQP, HAQP1, HAQPP1, CLKIJ, CLKP,
     &     raq, raqp, raqp1, raqpp1, rlkij, rlkp, rlkp1, rlkpp1,
     &     laq, laqp, laqp1, laqpp1, llkij, llkp, llkp1, llkpp1,
     &     CLKP1, CLKPP1, rlsum(maxs+mxjn), llsum(maxs+mxjn)
      INTEGER ICOUNT, NTSAQ, ITRIAL(100,MAXS)
      COMMON /MODBRN/ QLSUM, HAQ, HAQP, HAQP1, HAQPP1, CLKIJ, CLKP,
     &                CLKP1, CLKPP1, rlsum, llsum,
     &     raq, raqp, raqp1, raqpp1, rlkij, rlkp, rlkp1, rlkpp1,
     &     laq, laqp, laqp1, laqpp1, llkij, llkp, llkp1, llkpp1,
     &                ITRIAL, ICOUNT, NTSAQ
      SAVE /MODBRN/
C--End modbrn.cmn


C--Begin xsinit.cmn
C     MAXS = MAX NO. OF INPUT CROSS SECTIONS
C
c**** Modified to include reach transmissivity parameters
      REAL Dx(MAXS), T(MAXS), Rn(3,MAXS), Wangle(MAXS), Gdatum(MAXS),
     &   Orient(MAXS), Betvel(MAXS), Rho(MAXS), Qlat(MAXS),
Ulat(MAXS)
      REAL Dzdt(MAXS), Dqdxc(MAXS), Dqdt(MAXS), Dqdxm(MAXS),Dadx(MAXS),
     &     Dqadx(MAXS), Dzdx(MAXS), Fric(MAXS), Wind(MAXS),
Dpdx(MAXS),
     &     Qlatm(MAXS), Sumeta(MAXS), Sumczq(MAXS), Sczqsq(MAXS),
     &
Szqeta(MAXS),Clk(MAXS),Zbot(MAXS),Rn4(MAXS),Llk(MAXS),Rlk(MAXS)
      INTEGER Istrm(3,MAXS),Lstrm(3,MAXS),Rstrm(3,MAXS)
      COMMON /XSINIT/ Dx, T, Rn, Rn4, Wangle, Gdatum, Orient, Betvel,
     &                Rho, Qlat, Ulat, Sumeta, Sumczq, Sczqsq,
Szqeta,
     &       Rstrm, Dzdt, Dqdxc, Dqdt, Dqdxm, Dadx, Dqadx, Dzdx,
Fric,
     &         Lstrm, Wind, Dpdx, Qlatm, Clk, Zbot, Istrm, Rlk, Llk
      SAVE /XSINIT/
C--End xsinit.cmn
```