1976

Programming Phase of Water Response Ecosystem Model: Perennial Plant, Nitrogen and Decomposition Submodels

P. W. Lommen

D. C. Wilkin

Recommended Citation

Lommen, P.W., Wilkin, D.C. 1976. Programming Phase of Water Response Ecosystem Model: Perennial Plant, Nitrogen and Decomposition Submodels. U.S. International Biological Program, Desert Biome, Utah State University, Logan, Utah. Reports of 1975 Progress, Volume 1: Central Office, Modeling, RM 76-39.

1975/76 PROGRESS REPORT

PROGRAMMING PHASE OF WATER RESPONSE ECOSYSTEM MODEL:
II. ABIOTIC SUBMODELS

P. W. Lommen* and K. A. Marshall
Utah State University

(*now at HDR Ecosciences, Santa Barbara, California)

US/IBP DESERT BIOME
RESEARCH MEMORANDUM 76-37

in

REPORTS OF 1975 PROGRESS
Volume 1: Central Office, Modeling
Modeling Section, pp. 11-91

1975/76 Proposal No. 2.1.3

Printed 1976

Citation format: Author(s). 1976. Title.
US/IBP Desert Biome Res. Memo. 76-37.
Utah State Univ., Logan. 81 pp.

Ecology Center, Utah State University, Logan, Utah 84322

This report describes a portion of the Desert Biome Water Response Ecosystem Model. Five Research Memoranda comprise the full description: Introduction and support programs (RM 76-36); Abiotic submodels (RM 76-37); Animal submodel (RM 76-38); Perennial plant, nitrogen and decomposition submodels (RM 76-39); and Annual plant submodel (RM 76-40). The objectives of the Water Response Model, information on the arrangement of material distributed among the five Research Memoranda and descriptions of program MAIN and support programs F1, F3 and FTAVE are contained in Research Memorandum 76-36, **Programming phase of water response ecosystem model: I. Introduction and support programs.** The relationships between various sections of the model, their interactions and location in the report series are summarized in Table 1 of RM 76-36.

INTRODUCTION

This research memorandum describes submodels PSWG, HEAT and WATER (and their support programs) of the Desert Biome Water Response Model. These submodels provide environmental (or driving) variables to the plant, animal and soil-microbe submodels. PSWG (for Pseudo Stochastic Weather Generator) provides meteorological variables (air temperature, precipitation, plus several others). HEAT calculates soil temperature profile, given soil heat flow characteristics and air temperature. WATER determines soil water potential profile given initial conditions, soil moisture flow characteristics and several water inputs and outputs (precipitation, evaporation, transpiration, runon-runoff). Transpiration, through its effect on soil water potential, is the only biological effect on any of the ten or so environmental variables determined in these submodels.

A. PSWG
(including support programs EVAP, RINT, IPROB, RNOR)

K. A. Marshall and P. W. Lommen

GENERAL DESCRIPTION OF PSWG SUBMODEL

PSWG is a submodel which generates driving variables for the Water Response Model. (The name PSWG is generated from Pseudo Stochastic Weather Generator).

These variables are maximum, minimum and mean time-step air temperature at 2 m; precipitation and its intensity for the time-step; potential evaporation for the time-step; and mean values for the time-step of relative humidity, wind speed, fraction of possible sunlight and photoperiod.

For any run, one of two versions is used depending on data availability. Version I is used for debugging, tuning and validation runs. For debugging runs, all driving variables are stochastically generated. For tuning and validation runs where driving variables must correspond to the weather of a given year or years, daily temperature and precipitation records for the site are used. The remaining driving variables are chosen stochastically because daily records for the site are either not available at all or are not complete enough. Variables are stochastically chosen with the aid of random numbers and parameters generated from six years' data of the variable in question obtained from the United States Weather Bureau at the nearest complete weather station to our site. For Curlew Valley simulation, data from Pocatello, Idaho, were used. For Rock Valley simulations, Las Vegas, Nevada, data were used. Refer to Table A-1 for definitions of variables for both versions of PSWG. Definitions will also be found in the complete program listings at the end of this section.

Version II of PSWG is used for five-year (or longer) simulations. It generates nothing stochastically. Instead, it reads temperature, precipitation, wind speed, fraction of possible sunlight and relative humidity data from a weather data file. (For five-year Curlew Valley runs, data were used from the U.S. Weather Bureau station at Pocatello.)

Table A-1. Variable dictionary for PSWG, Versions I and II

AMT(6)	AMT(I) IS AMOUNT OF PRECIPITATION IN MM IN EVENT CLASS I.
DECL	ANGLE CALCULATED IN PHOTOPERIOD CALCULATION.
EVAP	SUBROUTINE WHICH CALCULATES POTENTIAL EVAPORATION.
FACTOR	PRECIPITATION FACTOR BY WHICH EACH EVENT IS MULTIPLIED.
HUM(10,13)	INTEGRATED PROBABILITY DISTRIBUTION OF RELATIVE HUMIDITY CLASS BY PERIOD OF YEAR.
IPROB	FUNCTION WHICH DETERMINES DEPENDENT VARIABLE GIVEN INTEGRATED PROBABILITY DISTRIBUTION, BY RANDOMLY CHOOSING INDEPENDENT VARIABLE IN RANGE 0-1.
ISEED	SEED FROM WHICH RNOR GENERATES NEXT RANDOM NUMBER.
IYEST	EQUALS 1 IF THERE WAS NO PRECIPITATION YESTERDAY. IF EQUALS 2 THEN THERE WAS PRECIPITATION YESTERDAY AND PROBABILITY OF PRECIPITATION TODAY IS INCREASED.
J	CLASS INDEX, USED IN PRECIP, WIND, SUN, HUM GENERATION.
JDAY	JULIAN DATE AT BEGINNING OF TIMESTEP.
LAT	LATITUDE OF SITE.
MDAY	CURRENT JULIAN DAY (INCREMENTED IN PSWG FROM JDAY TO JDAY+PMDT).
MTIME	INDEX OF 4 WEEK PERIOD OF YEAR IN WHICH MDAY FALLS.
NNN	COUNTER TO DETERMINE WHEN TO START PRINTING ON NEXT PAGE.
PMDT	TIMESTEP LENGTH, DAYS.
PREC(6,13)	INTEGRATED PROBABILITY DISTRIBUTION OF PRECIPITATION CLASS BY PERIOD OF YEAR.
PRECIP	PRECIPITATION FOR CURRENT DAY, MM.
PMJDAT	JULIAN DATE AT BEGINNING OF TIMESTEP. VALUE DETERMINED IN MAIN PROGRAM.
PWFAC	FACTOR FOR ADJUSTING AVERAGE WIND SPEED IF SITE IS KNOWN TO HAVE DIFFERENT AVERAGE WIND SPEED.
RAIN(2,13)	ARRAY HOLDING VALUES OF (1. - PROBABILITY OF PRECIP. TODAY) GIVEN PERIOD OF YEAR AND WHETHER OR NOT WE HAD PRECIP. YESTERDAY.

Table A-1, continued

RCHECK(20)	VARIABLE USED TO READ AND WRITE COMMENTS IN WITH INITIAL DATA.
READ	IF =.TRUE. THEN READ TEMPERATURE PRECIPITATION DATA FROM SEPARATE DATA FILE.
RINT	SUBROUTINE WHICH CALCULATES PRECIPITATION INTENSITY.
RNOR	RANDOM NORMAL NUMBER GENERATOR (68% OF VALUES LIE BETWEEN -1 AND +1).
SHUM	RUNNING SUM OF UP TO PMDT DAYS' RELATIVE HUMIDITY.
SMAX	RUNNING SUM OF UP TO PMDT DAYS' MAXIMUM TEMPERATURES. DEGREES CELCIUS.
SMIN	RUNNING SUM OF UP TO PMDT DAYS' MINIMUM TEMPERATURES. DEGREES CELCIUS.
SPREC	RUNNING SUM OF UP TO PMDT DAYS' PRECIPITATIONEVENTS,MM.
SSUN	RUNNING SUM OF UP TO PMDT DAYS' PER CENT POSSIBLE SUNLIGHT
SUN(10,13)	INTEGRATED PROBABILITY DISTRIBUTION OF PER CENT POSSIBLE SUNLIGHT CLASS BY PERIOD OF YEAR.
SWIND	RUNNING SUM OF UP TO PMDT DAYS' AVERAGE WIND SPEEDS. KM/HR.
TMAX	MAXIMUM TEMPERATURE FOR CURRENT DAY, DEGREES CELCIUS.
TMIN	MINIMUM TEMPERATURE FOR CURRENT DAY, DEGREES CELCIUS.
TRAIN(20)	ARRAY WHICH HOLDS DAILY PRECIP VALUES FOR TIMESTEP (USED BY RAIN INTENSITY ROUTINE).
WIND(7,13)	INTEGRATED PROBABILITY DISTRIBUTION OF WIND SPEED CLASS BY PERIOD OF YEAR,
XMAX(1),XMAX(2)	PARAMETERS FOR SINE WAVE FIT TO SEVERAL YEARS' DAILY MAXIMUM TEMPERATURES, TAKEN AT NEAREST WEATHER BUREAU SITE.
XMAX(3)	STANDARD DEVIATION OF DATA ABOUT SINE WAVE FIT USING XMAX(1) AND XMAX(2).
XMIN(1),XMIN(2)	PARAMETERS FOR LINEAR REGRESSION BETWEEN TMAX AND TMIN.
XMIN(3)	STANDARD DEVIATION OF TMIN.
ZAIRT	AVERAGE DAILY AIR TEMPERATURE FOR TIMESTEP, DEGREES CELCIUS. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZESUM	SUM FROM OCTOBER 1 OF ZEVAP, MM.
ZEVAP	POTENTIAL EVAPORATION, MM/TIMESTEP. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZPHPD	PHOTOPERIOD OF FIRST DAY IN TIMESTEP, HOURS. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZRAIN	TOTAL PRECIPITATION FOR TIMESTEP, MM. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZRH	AVERAGE DAILY RELATIVE HUMIDITY, DECIMAL FRACTION FROM 0-1. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZRINT	PRECIPITATION INTENSITY, MM/HR. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZRSUM	SUM FROM OCTOBER 1 OF ZRAIN, MM.
ZSUN	AVERAGE DAILY PER CENT POSSIBLE SUNLIGHT FOR TIMESTEP, DECIMAL FRACTION FROM 0-1. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZTMAX	AVERAGE DAILY MAXIMUM TEMPERATURE FOR TIMESTEP. DEGREES CELCIUS. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZTMIN	AVERAGE DAILY MINIMUM TEMPERATURE FOR TIMESTEP, DEGREES CELCIUS. DRIVING VARIABLE USED BY OTHER SUBMODELS.
ZWIND	AVERAGE DAILY WIND SPEED FOR TIMESTEP. DRIVING VARIABLE USED BY OTHER SUBMODELS.

PROGRAM DESCRIPTION

PSWG, Version I

Only the important segments of the FORTRAN code are shown and described. Sequence numbers are shown to aid in reference to the full code listing which follows the Program Description. All comment cards, specification statements and bookkeeping sections have also been deleted. Definitions of variable names may be found in Table A-1, which also appears at the beginning of the program listing.

```
    DO 3 I=1,PMDT                                                   PSG1 183
```

Beginning of main loop in program. Go through loop once for each day of time-step in order to get daily weather information for appropriate average or summed value.

```
    MTIME=MDAY/28+1                                                 PSG1 184
    IF (MTIME .GT. 13) MTIME=13                                     PSG1 185
```

Determine MTIME, index of four-week period of year in which MDAY, current Julian day, falls.

```
    IF(.NOT.READ)GOTO100                                            PSG1 189
020 READ(8,25,END=500)TMAX,TMIN,PRECIP                              PSG1 190
025 FORMAT(7X,2F6.1,36X,F6.1)                                       PSG1 191
    GOTO30                                                          PSG1 192
```

If flag is set (i.e., if READ is .TRUE.) then read temperature and precipitation data from separate weather file. Then check if data are reasonable.

```
500 READ=.FALSE.                                                    PSG1 194
    GOTO100                                                         PSG1 197
```

If control reaches here then all data have been read and remainder of temperatures and precipitation amounts will be generated stochastically.

```
    IF(TMAX.LT.TMIN)GOTO40                                          PSG1 202
    IF((TMAX.LT.-30.).OR.(TMAX.GT.45.))GOTO40                       PSG1 203
    IF((TMIN.LT.-30.).OR.(TMIN.GT.45.))GOTO40                       PSG1 204
    IF(((PRECIP.LT.50.).AND.(PRECIP.GE.0.0)).OR.(PRECIP.GT.5000.))  PSG1 205
   *  GOTO60                                                        PSG1 206
```

Check data just read and see if they are reasonable.

```
040 WRITE(6,50)PMJDAT,TMAX,TMIN,PRECIP                              PSG1 210
050 FORMAT(' ',I5,3F12.5)                                           PSG1 211
    STOP                                                            PSG1 212
```

We reach here only if a datum was judged unreasonable. Write offending card and stop execution.

```
060 CONTINUE                                                            PSG1 215
    SMAX=SMAX+TMAX                                                      PSG1 216
    SMIN=SMIN+TMIN                                                      PSG1 217
    SPREC=SPREC+PRECIP                                                  PSG1 218
    TRAIN(I)=PRECIP                                                     PSG1 220
    GOTO2                                                               PSG1 221
```

Keep running sums over PMDT of temperature and precipitation information. Also load appropriate place in TRAIN array for rain intensity calculation. Go on to wind speed section.

```
100 CONTINUE                                                            PSG1 225
    TMAX = XMAX(1) + XMAX(2)*SIN(.017214*MOD((PMJDAT+344.),365.)        PSG1 234
   C  -1.570796)                                                        PSG1 235
001 TMAX = TMAX + XMAX(3)*RNOR(ISEED)                                   PSG1 237
    TMIN=XMIN(1)+XMIN(2)*TMAX                                           PSG1 238
    TMIN=TMIN+XMIN(3)*RNOR(ISEED)                                       PSG1 239
    IF (TMIN .GT. TMAX) GO TO 1                                         PSG1 240
    SMAX=SMAX+TMAX                                                      PSG1 241
    SMIN=SMIN+TMIN                                                      PSG1 242
```

Control reaches here if we wish to stochastically generate the current day's temperatures. First, calculate the mean maximum temperature for the day. Next, generate the day's minimum temperature from the maximum. Next, make sure the minimum is less than or equal to the maximum. Finally, add these temperatures to the sums of temperatures for previous days of the time-step.

```
    IYEST=IPROB(RAIN(IYEST,MTIME),1)                                    PSG1 246
    IF(IYEST .NE. 2) TRAIN(I)=0.0                                       PSG1 247
    IF (IYEST .NE. 2) GO TO 2                                           PSG1 248
    J=IPROB(PREC(1,MTIME),5)                                            PSG1 249
    SPREC=SPREC+AMT(J)                                                  PSG1 250
    TRAIN(I)=AMT(J)                                                     PSG1 251
```

Stochastically generate the day's precipitation. First determine IYEST for current day which depends on previous day's value. (If previous value was 2 rather than 1, then probability of IYEST = 2 today is considerably increased.) If IYEST = 1, then there is no precipitation today and control goes to wind speed section. If IYEST = 2, then there is precipitation today and so the amount is then determined. Calculate J, the size class of the event, an integer from 1 to 6 which depends on time of year. Once size class is determined, the amount corresponding to this class is added to SPREC, and loaded in TRAIN(I).

```
002 J=IPROB(WIND(1,MTIME),7)                                            PSG1 254
    SWIND=SWIND+(J-1)*8.05                                              PSG1 255
```

Generate today's wind speed. First find J, the size class. Today's wind speed is then (J—1)·8.05 km/hr (8.05 km/hr = 5.00 mi/hr). Add this value to the time-step running total of wind speeds.

```
    J=IPROB(SUN(1,MTIME),10)                                            PSG1 258
    SSUN=SSUN+(J-1)*0.1                                                 PSG1 259
    J=IPROB(HUM(1,MTIME),10)                                            PSG1 262
    SHUM=SHUM+(J-1)*0.1                                                 PSG1 263
```

Determine fraction of possible sunlight and relative humidity similarly to wind speed. Possible values taken on will be 0, 0.1, 0.2, . . . , 1.0.

```
003 MDAY=MDAY+1                                                           PSG1 264
```

The main loop of PSWG is ended by incrementing the Julian date.

```
      X=1./PMDT                                                           PSG1 269
      ZTMAX=SMAX*X                                                        PSG1 270
      ZTMIN=SMIN*X                                                        PSG1 271
      ZWIND=SWIND*X                                                       PSG1 272
```

Determine time-step averages for maximum and minimum air temperatures and for wind speed.

```
      ZWIND = ZWIND * PWFAC                                               PSG1 276
```

If site is known to have different average wind speed than measuring station, multiply wind speed just determined by this factor.

```
      ZSUN=SSUN*X                                                         PSG1 279
      ZRH=SHUM*X                                                          PSG1 280
      ZRAIN=SPREC                                                         PSG1 281
      ZAIRT=(ZTMAX+ZTMIN)/2.                                              PSG1 282
```

ZSUN, ZRH and ZAIRT are averages for the time-step. ZRAIN is the total time-step precipitation.

```
      DECL=ARSIN(SIN(6.2831853*23.45/360.)*                               PSG1 285
     C COS(MOD((PMJDAT+193.),365.)*6.2831853/365.))                       PSG1 286
      X=(SIN(LAT*6.2831853/360.)*TAN(DECL)+SIN(.875*6.2831853/360.))/     PSG1 287
     C COS(LAT*6.2831853/360.)                                            PSG1 288
      IF (X .GT. 1.) X=1.                                                 PSG1 289
      IF (X .LT. -1.) X=-1.                                               PSG1 290
      ZPHPD=12. + .1333333*ARSIN(X)*360./6.2831853                        PSG1 291
```

Calculate ZPHPD, photoperiod, in hours, of first day of time-step.

```
      CALL RINT(ZRINT, TRAIN, ZRAIN, PMDT, PMJDAT, ISEED)                 PSG1 295
      CALL EVAP                                                           PSG1 300
```

Determine rain intensity and potential evaporation.

```
      ZRAIN=FACTOR*ZRAIN                                                  PSG1 306
```

Change time-step precipitation if FACTOR is not equal to 1. By changing FACTOR from one run to another, keeping all else the same, we determine the response of the entire model to changes in precipitation, a primary objective of the Water Response Model, as its name implies.

```
      ENTRY ZINIT                                                         PSG1 333
```

Entry point for initialization purposes. Called from MAIN once. Variables peculiar to PSWG are then read and written, with comment cards at the beginning, middle and end.

```
      CALL EVINIT
      CALL RIINIT
```

```
PSG1 418
PSG1 419
```

Call initialization sections of subroutines EVAP and RINT.

PSWG, Version II

Version II is much more simple than Version I.

```
      DO 3 I=1,PMDT
```

```
PSG2 120
```

Begin main loop. Go through loop once for each day of time-step.

```
      READ(8,30,END=500)TMAX,TMIN,PRECIP,WIND,PPS,RH
   30 FORMAT(10X,2F5.0,F5.2,F5.1,2F5.2)
```

```
PSG2 122
PSG2 123
```

Read today's weather data.

```
      TMAX=(TMAX-32.)*0.555555
      TMIN=(TMIN-32.)*0.555555
      PRECIP=PRECIP*25.4
      WIND=WIND*1.60934
```

```
PSG2 133
PSG2 134
PSG2 137
PSG2 140
```

Convert temperature, precipitation and wind speed to metric units.

```
      IF(TMIN.GT.TMAX)GOTO40
      IF((TMAX.GT.42.).OR.(TMIN.LT.-43.))GOTO40
      IF((PRECIP.GT.40.).OR.(PRECIP.LT.0.0))GOTO40
      IF((WIND.GT.60.).OR.(WIND.LT.0.0))GOTO40
      IF((PPS.GT.1.0).OR.(PPS.LT.0.0))GOTO40
      IF((RH.LE.1.00).AND.(RH.GE.0.0))GOTO60
   40 WRITE(6,50)TMAX,TMIN,PRECIP,WIND,PPS,RH,PMJDAT
   50 FORMAT(' ',7F10.3)
      STOP
```

```
PSG2 144
PSG2 145
PSG2 146
PSG2 147
PSG2 148
PSG2 149
PSG2 152
PSG2 153
PSG2 154
```

Check if values just read are reasonable. If not, print offending card and stop. If reasonable, go on to summing section.

```
      SMAX=SMAX+TMAX
      SMIN=SMIN+TMIN
      SPREC=SPREC+PRECIP
      SWIND=SWIND+WIND
      SSUN=SSUN+PPS
      SHUM=SHUM+RH
```

```
PSG2 158
PSG2 159
PSG2 160
PSG2 161
PSG2 162
PSG2 163
```

Keep running sums of values for appropriate averages to be calculated below.

```
      TRAIN(I)=PRECIP
```

```
PSG2 166
```

Load TRAIN(I) for use in rain intensity subroutine.

```
3 CONTINUE
```

PSG2 168

End of main loop.

From here on, Version II is the same as Version I, with one exception: the entry point of Version II is much simpler than that of Version I because all the parameters for stochastically generating variables are not read in.

SUBROUTINE EVAP

EVAP calculates potential evaporation from soil surface in millimeters, during a time-step. The Blaney-Criddle method is used and closely follows the aproach of Griffin et al. 1974.

Refer to Table A-2 for definitions of variables used in this program. This table is also found in the complete program listing at the end of this section.

```
ZEVAP=F1(ZAIRT, ARRAY1, NPTS)
```

EVAP 056

Here, ZEVAP is multiplying factor ($\lesssim 1$) depending on air temperature, which has been empirically determined.

Table A-2. Variable dictionary for EVAP

A, B, C, Z	TEMPORARY VARIABLES USED IN DAYLIGHT CALCULATION.
ARRAY(5,2)	PAIRS OF DATA POINTS FOR INTERPOLATION BY F1 IN CALCULATING BLANEY CRIDDLE FACTOR.
CVVCOV	FRACTION OF SOIL SURFACE COVERED BY ANNUAL AND PERENNIAL VEGETATION.
DALITE(365)	ARRAY INDEXED BY JULIAN DATE . GIVES NUMBER OF MINUTES OF DAYLIGHT ON THAT JULIAN DATE.
DLTFR(365)	ARRAY INDEXED BY JULIAN DATE AND IS FRACTION OF YEAR'S DAYLIGHT WHICH FALLS ON THAT JULIAN DATE.
F1	FUNCTION WHICH LINEARLY INTERPOLATES BETWEEN PAIRS OF DATA POINTS TO DETERMINE DEPENDENT VARIABLE.
LAT	LATITUDE OF SITE, DEGREES.
MM	FLAG USED ONLY IN CASE PHJDAT IS EVER ZERO.
NPTS	NUMBER OF PAIRS OF DATA POINTS ACTUALLY USED IN ARRAY.
PHDT	LENGTH OF TIMESTEP, DAYS.
PHJDAT	JULIAN DATE OF FIRST DAY OF TIMESTEP.
RCHECK(20)	ARRAY USED TO READ AND WRITE COMMENTS IN WITH INITIAL DATA.
TOTMIN	MINUTES OF DAYLIGHT IN YEAR.
ZAIRT	AVERAGE TIMESTEP AIR TEMPERATURE, DEGREES CELCIUS.
ZEVAP	POTENTIAL EVAPORATION FOR TIMESTEP, MM, DRIVING VARIABLE USED BY OTHER SUBMODELS.

```
      IF(PMJDAT ,GT, 0) GO TO 670                                       EVAP 057
      PMJDAT=1                                                          EVAP 058
      MM=1                                                              EVAP 059
  670 CONTINUE                                                          EVAP 060
```

PMJDAT is used as a subscript in the next line. These lines simply set it equal to 1 in case it should ever be 0.

```
      ZEVAP=ZEVAP*(1,8*ZAIRT + 32,)*DLTFR(PMJDAT)*(1,-CVVCOV)*25,4*PMDT EVAP 063
```

This, in essence, is subroutine EVAP. The potential evaporation equals the empirical factor times the temperature in degrees Fahrenheit, times the fraction of the year's daylight occurring today, times the fraction of uncovered soil surface, times 25.4 to change from inches to millimeters, times PMDT to change from one day to the value for the entire time-step.

```
      CVVCOV=0,0                                                        EVAP 064
```

Reset CVVCOV. It is calculated each time-step by one or both vegetation submodels.

```
      IF(MM ,EQ, 0) GO TO 680                                           EVAP 065
      PMJDAT=0                                                          EVAP 066
      MM=0                                                              EVAP 067
  680 CONTINUE                                                          EVAP 068
```

Set PMJDAT back to 0 if it was changed to 1 in lines 57-60 above.

```
      ENTRY EVINIT                                                      EVAP 073
```

Entry point called once from PSWG to read initial data and do daylight calculations which need to be done only once.

```
      DO 650 I=1, 365                                                   EVAP 108
      A=730,-,274*LAT+,00793*(LAT**2)                                   EVAP 109
      B=34,2-,78*LAT+,1*(LAT**2)                                        EVAP 110
      C=I                                                               EVAP 111
      Z=2,*3,1416*((C+285,)/365,)                                       EVAP 112
      DALITE(I)=A+B*SIN(Z)                                              EVAP 114
      TOTMIN=DALITE(I)+TOTMIN                                           EVAP 115
  650 CONTINUE                                                          EVAP 116
```

In this loop calculate the length of daylight in minutes for each day of the year at latitude LAT. Also calculate TOTMIN, the total minutes of daylight in a year.

```
      DO 660 I=1, 365                                                   EVAP 119
      DLTFR(I)=DALITE(I)/TOTMIN                                         EVAP 120
  660 CONTINUE                                                          EVAP 121
```

Calculate DLTFR, the fraction of daylight which occurs each day of the year.

SUBROUTINE RINT

This program calculates ZRINT, precipitation intensity in mm/hr. ZRINT is used by WATER in determining rate of infiltration into soil. Intensity is important mainly if the rate of water arriving at the soil surface is greater than the infiltration rate. Under these conditions, runon or runoff could occur.

Refer to Table A-3 for definitions of variables used in this program. This table is also repeated in the complete program listing at the end of this section.

```
DO 550 I=1,PMDT                                             RINT 070
```

Make one pass through this loop for each day of time-step.

Table A-3. Variable dictionary for RINT

A, B	PARAMETERS IN HIGH INTENSITY CALCULATION.
ARRAY(10)	HOLDS UP TO 5 PAIRS OF DATA POINTS FOR INTERPOLATION BY FUNCTION F1 IN DETERMINING NORMAL INTENSITY.
BA	INTERMEDIATE VARIABLE IN HIGH INTENSITY CALCULATION.
B1	INTERMEDIATE VARIABLE IN HIGH INTENSITY CALCULATION.
F1	FUNCTION WHICH LINEARLY INTERPOLATES BETWEEN NPTS PAIRS OF DATA POINTS.
ISEED	INTEGER FROM WHICH RANDOM NUMBERS ARE GENERATED.
N	INDEX OF TIME OF YEAR. IN MIDDLE OF YEAR N=1. OTHER TIMES OF YEAR N=2.
NPTS	NUMBER OF PAIRS OF DATA POINTS IN NORMAL INTENSITY CALCULATION.
PDAY1,PDAY2	BEGINNING AND ENDING JULIAN DATES OF CENTRAL SEGMENT OF YEAR.
PMDT	LENGTH OF TIMESTEP, DAYS.
PMJDAT	JULIAN DATE OF BEGINNING OF TIMESTEP.
PTH	PRECIP AMOUNT ABOVE WHICH HIGH INTENSITY CAN OCCUR
PA(2)	COMPARED WITH R TO SEE IF HIGH INTENSITY ACTUALLY OCCURRED.
R	TEMPORARY VARIABLE. SET EQUAL TO RANDOM NUMBER
RCHECK(20)	ARRAY USED FOR READING AND WRITING COMMENTS IN WITH INITIALIZATION DATA.
TR	TEMPORARY VARIABLE.
TRAIN(20)	DAILY PRECIPITATION AMOUNTS, MM. ARRAY INDEXED BY DAY OF TIMESTEP
X	TODAY'S RAIN
Y	TODAY'S PRECIPITATION INTENSITY , MM/HR.
YSUM	SUM OVER TIMESTEP OF DAILY PRECIP INTENSITY TIMES PRECIP AMOUNT.
ZRAIN	TIMESTEP PRECIPTATION,MM. DRIVING VARIABLE USED BY WATER SUBMODEL.
ZRINT	RAIN INTENSITY, MM/HR. DRIVING VARIABLE USED BY WATER SUBMODEL.

```
    X=TRAIN(I)                                                RINT 074
```

Get today's precipitation amount, x, from appropriate location in TRAIN, which was loaded in PSWG.

```
    IF(X .GT. 1.E-5) GO TO 100                                RINT 077
    GO TO 550                                                 RINT 079
```

If x $\geqslant 1.0 \times 10^{-5}$, this is considered precipitation so an intensity must be determined. Otherwise, go to end of loop.

```
100 IF(X .LT. PTH) GO TO 500                                  RINT 082
```

If there is precipitation today but the amount is less than threshold value PTH, go to statement 500 and calculate normal intensity for this time of year.

```
    R=RANDOM(ISEED)                                           RINT 085
```

If we reach here, a high intensity is possible. Choose a random number uniformly distributed in interval 0 to 1.

```
    N=2                                                       RINT 091
    IF((PMJDAT .GE. PDAY1) .AND. (PMJDAT .LE. PDAY2)) N=1     RINT 092
```

Determine which of two segments of year we're in. In central segment (summer, roughly) N = 1.

```
    IF(R .GT. PA(N)) GO TO 500                                RINT 095
```

If random number chosen above is greater than a threshold PA, which varies with segment of year, then go to section calculating normal intensity for this time of year.

```
    R=R/PA(N)                                                 RINT 097
```

If we reach here, a high intensity will result for today's precipitation. Generate a new random number.

```
150 TR=BA*(1.-R)                                              RINT 099
    IF(TR .LE. 1.E-10) TR=1.E-10                              RINT 100
    Y=-B1*ALOG(TR)                                            RINT 102
    IF(Y .GT. 25.) Y=25.                                      RINT 105
    GO TO 520                                                 RINT 106
```

Calculate intensity from exponential distribution. TR is not allowed to be smaller than 10^{-10} in order to keep ALOG function manageable. Also, intensity is kept less than 25 mm/hr (a value it would almost never reach anyway). Go on to YSUM calculation.

Lines 99 and 102 require several lines of algebra to derive. From "Local Climatological Data" sheets obtained from the U.S. Weather Bureau for the station of interest (Pocatello, Idaho, for Curlew Valley runs), a histogram was constructed of

(frequency of precipitation events per .01 in/hr interval where intensity $\geqslant 0.1$ in/hr)

vs.

(intensity, in/hr).

Data over a period of several years were used. This histogram was fit with the expression:

$$\text{frequency} = ae^{-bx}, \qquad \text{(A-1)}$$

where x is the intensity, in/hr.

Then, if intensity >0.1 in/hr, probability for intensity to be between x and $x+dx$ is

$$P(x)dx = (ae^{-bx})\,(dx/.01). \qquad \text{(A-2)}$$

If intensity > 0.1 in/hr, the probability for it to be between .1 and ∞ is 1, i.e.:

$$1 = \int_{0.1}^{\infty} (ae^{-bx})\,(dx/0.01) = [a/(.01b)]e^{-.1b}. \qquad \text{(A-3)}$$

If we let S be the probability that 0.1 in/hr $<$ intensity $\leqslant x^*$, then

$$S = \int_{0.1}^{x^*} (ae^{-bx})\,(dx/0.01). \qquad \text{(A-4)}$$

If the integral is carried out,

$$S = 1 - [a/(0.01b)]e^{-bx^*}. \qquad \text{(A-5)}$$

Now, if we let $S = R$, where R is a random number chosen uniformly over the range from 0 to 1, and solve Equation A-5 for x, we get

$$x^* = -(1/b)\ln[(0.01b/a)\,(1 - R)]. \qquad \text{(A-6)}$$

The values of x^* will then have a distribution given by ae^{-bx}, as required.

The transformation of Equation A-5 into FORTRAN code is straightforward. Parameters a and b become A and B. The factor 0.01b/a becomes 0.01 * B/A = BA; 1/b becomes 1./B => 25.4/B = B1, so that intensities come out in mm/hr, not in/hr. Thus, (0.01b/a) (1—R) becomes BA*(1.—R) = TR; x* becomes Y; and Y = —B1*ALOG (TR).

```
500 Y=F1(PMJDAT, ARRAY, NPTS)
```

```
RINT 110
```

This is the normal intensity calculation. This line is missed only if intensity is high (i.e., this line is seldom missed). Function F1 interpolates between values found in ARRAY, dependent on time of year.

```
520 YSUM=X*Y + YSUM                                            RINT 113
```

Continue running sum over time-step of daily intensity times amount.

```
    IF(ZRAIN .LE. 1.E-6) GO TO 570                             RINT 118
    ZRINT = YSUM/ZRAIN                                         RINT 119
    GO TO 580                                                  RINT 120
570 ZRAIN=0.0                                                  RINT 121
    ZRINT=0.0                                                  RINT 122
580 IF((ZRINT*24.*PMDT) .LT. ZRAIN)  ZRINT=ZRAIN/(24.*PMDT)    RINT 123
```

Determine average rain intensity, making sure of two things first: 1) that we don't divide by zero, and 2) that the precipitation amount we'd get if precipitation fell at rate ZRINT for the entire time-step is at least as large as the total time-step precipitation already determined in PSWG.

```
    ENTRY RIINIT                                               RINT 129
```

Entry point for reading and writing initial data and comments, and for calculations which need to be done only once.

```
    BA=0.01*B/A                                                RINT 176
    B1=25.4/B                                                  RINT 177
```

Calculate BA and B1 which are used in high intensity calculation (lines 99 and 102).

Function IPROB (A,N)

IPROB (A,N) determines an index from 1 to N+1 given an integrated probability distribution A and a uniformly chosen random number X, $0 \leqslant X \leqslant 1$. A is an array with N values such that $A(1) < A(2) < A(3) < \ldots < A(N) \leqslant 1$. If $A(1) > X$, then IPROB = 1. If $A(2) > X > A(1)$, then IPROB = 2, etc. If $X > A(N)$, then IPROB = N+1. Array A can be set up in many ways. The index determined can be for precipitation size class, wind speed class, relative humidity (RH) class, etc. For example, we have RH classes set up so that if IPROB = 1, RH = 0, if IPROB = 2, RH = 0.1, . . . , if IPROB = 11, RH = 1.0.

```
    X=RANDOM(ISEED)                                            IPROB 04
```

Choose random number between 0 and 1.

```
    DO 1 I=1,N                                                 IPROB 05
    IF (A(I) .GT. X) GO TO 2                                   IPROB 06
001 CONTINUE                                                   IPROB 07
```

This DO loop takes each value of A, starting with the smallest, and checks it against the value of X, until a value is found greater than X. The loop is then exited.

```
002 IPROB=I                                                    IPROB 08
```

Set value of function equal to current value of I. Usually this means $A(I-1) < X < A(I)$. If DO loop was exited normally, i.e., $X >$ all values of A, then $I = N + 1$.

COMPLETE PROGRAM LISTING

Subroutine PSWG, Version I

```
      SUBROUTINE PSWG
C
C  THE PURPOSE OF THIS SUBROUTINE IS TO PROVIDE DRIVING VARIABLES FOR
C  USE BY SUBMODELS OF THE WATER RESPONSE MODEL.
C  THIS VERSION OF PSWG USED IN TUNING AND VALIDATION RUNS.
C
C  THIS VERSION OF PSWG CAN EITHER PSEUDO STOCHASTICALLY GENERATE ALL
C  WEATHER VARIABLES OR IT CAN READ TEMP AND PRECIP DATA AND GENERATE
C  THE REST.
C
C
C  WRITTEN BY KIM MARSHALL
C                DESERT BIOME
C                UTAH STATE UNIVERSITY  UMC 52
C                LOGAN, UTAH 84322
C   AUGUST 1976
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C      VARIABLE DICTIONARY                                             C
C                                                                      C
C                                                                      C
C  AMT(6)       AMT(I) IS AMOUNT OF PRECIPITATION IN MM IN EVENT       C
C               CLASS I.                                               C
C                                                                      C
C  DECL         ANGLE CALCULATED IN PHOTOPERIOD CALCULATION.           C
C                                                                      C
C  EVAP         SUBROUTINE WHICH CALCULATES POTENTIAL EVAPORATION.     C
C                                                                      C
C  FACTOR      PRECIPITATION FACTOR BY WHICH EACH EVENT IS             C
C               MULTIPLIED.                                            C
C                                                                      C
C  HUM(10,13)   INTEGRATED PROBABILITY DISTRIBUTION OF RELATIVE        C
C               HUMIDITY CLASS BY PERIOD OF YEAR.                      C
C                                                                      C
C  IPROB        FUNCTION WHICH DETERMINES DEPENDENT VARIABLE GIVEN     C
C               INTEGRATED PROBABILITY DISTRIBUTION, BY RANDOMLY       C
C               CHOOSING INDEPENDENT VARIABLE IN RANGE 0-1.            C
C                                                                      C
C  ISEED        SEED FROM WHICH RNOR GENERATES NEXT RANDOM NUMBER.     C
C                                                                      C
C  IYEST        EQUALS 1 IF THERE WAS NO PRECIPITATION YESTERDAY.   IF C
C               EQUALS 2 THEN THERE WAS PRECIPITATION YESTERDAY AND    C
C               PROBABILITY OF PRECIPITATION TODAY IS INCREASED.       C
C                                                                      C
C  J            CLASS INDEX, USED IN PRECIP, WIND, SUN, HUM GENERATION.C
C                                                                      C
C  JDAY         JULIAN DATE AT BEGINNING OF TIMESTEP.                  C
C                                                                      C
C  LAT          LATITUDE OF SITE.                                      C
C                                                                      C
C  MDAY         CURRENT JULIAN DAY (INCREMENTED IN PSWG FROM           C
C               JDAY TO JDAY+PMDT).                                    C
C                                                                      C
C  MTIME        INDEX OF 4 WEEK PERIOD OF YEAR IN WHICH MDAY FALLS.    C
C                                                                      C
C  NNN          COUNTER TO DETERMINE WHEN TO START PRINTING ON         C
C               NEXT PAGE.                                             C
C                                                                      C
C  PMDT         TIMESTEP LENGTH, DAYS.                                 C
C                                                                      C
C  PREC(6,13)   INTEGRATED PROBABILITY DISTRIBUTION OF PRECIPITATION   C
C               CLASS BY PERIOD OF YEAR.                               C
C                                                                      C
C  PRECIP       PRECIPITATION FOR CURRENT DAY, MM.                     C
C                                                                      C
C  PMJDAT       JULIAN DATE AT BEGINNING OF TIMESTEP.  VALUE           C
C               DETERMINED IN MAIN PROGRAM.                            C
C                                                                      C
C  PWFAC        FACTOR FOR ADJUSTING AVERAGE WIND SPEED IF SITE IS     C
C               KNOWN TO HAVE DIFFERENT AVERAGE WIND SPEED.            C
C                                                                      C
C  RAIN(2,13)   ARRAY HOLDING VALUES OF (1. - PROBABILITY OF PRECIP.   C
C               TODAY) GIVEN PERIOD OF YEAR AND WHETHER OR NOT WE HAD  C
C               PRECIP. YESTERDAY.                                     C
C                                                                      C
C  RCHECK(20)   VARIABLE USED TO READ AND WRITE COMMENTS IN WITH       C
C               INITIAL DATA.                                          C
C                                                                      C
C  READ         IF =.TRUE. THEN READ TEMPERATURE PRECIPITATION DATA    C
C               FROM SEPARATE DATA FILE.                               C
C                                                                      C
C  RINT         SUBROUTINE WHICH CALCULATES PRECIPITATION INTENSITY.   C
C                                                                      C
C  RNOR         RANDOM NORMAL NUMBER GENERATOR (68% OF VALUES LIE      C
```

```
C              BETWEEN -1 AND +1).                                    C PSG1 087
C                                                                     C PSG1 088
C   SHUM       RUNNING SUM OF UP TO PMDT DAYS' RELATIVE HUMIDITY.     C PSG1 089
C                                                                     C PSG1 090
C   SMAX       RUNNING SUM OF UP TO PMDT DAYS' MAXIMUM TEMPERATURES,  C PSG1 091
C              DEGREES CELCIUS.                                       C PSG1 092
C                                                                     C PSG1 093
C   SMIN       RUNNING SUM OF UP TO PMDT DAYS' MINIMUM TEMPERATURES,  C PSG1 094
C              DEGREES CELCIUS.                                       C PSG1 095
C                                                                     C PSG1 096
C   SPREC      RUNNING SUM OF UP TO PMDT DAYS' PRECIPITATIONEVENTS,MM.C PSG1 097
C                                                                     C PSG1 098
C   SSUN       RUNNING SUM OF UP TO PMDT DAYS' PER CENT POSSIBLE      C PSG1 099
C              SUNLIGHT                                               C PSG1 100
C                                                                     C PSG1 101
C   SUN(10,13) INTEGRATED PROBABILITY DISTRIBUTION OF PER CENT        C PSG1 102
C              POSSIBLE SUNLIGHT CLASS BY PERIOD OF YEAR.             C PSG1 103
C                                                                     C PSG1 104
C   SWIND      RUNNING SUM OF UP TO PMDT DAYS' AVERAGE WIND SPEEDS,   C PSG1 105
C              KM/HR.                                                 C PSG1 106
C                                                                     C PSG1 107
C   TMAX       MAXIMUM TEMPERATURE FOR CURRENT DAY, DEGREES CELCIUS.  C PSG1 108
C                                                                     C PSG1 109
C   TMIN       MINIMUM TEMPERATURE FOR CURRENT DAY, DEGREES CELCIUS.  C PSG1 110
C                                                                     C PSG1 111
C   TRAIN(20)  ARRAY WHICH HOLDS DAILY PRECIP VALUES FOR TIMESTEP     C PSG1 112
C              (USED BY RAIN INTENSITY ROUTINE).                      C PSG1 113
C                                                                     C PSG1 114
C   WIND(7,13) INTEGRATED PROBABILITY DISTRIBUTION OF WIND SPEED CLASSC PSG1 115
C              BY PERIOD OF YEAR,                                     C PSG1 116
C                                                                     C PSG1 117
C   XMAX(1),XMAX(2)  PARAMETERS FOR SINE WAVE FIT TO SEVERAL YEARS'   C PSG1 118
C             DAILY MAXIMUM TEMPERATURES, TAKEN AT NEAREST WEATHER    C PSG1 119
C             BUREAU SITE.                                            C PSG1 120
C                                                                     C PSG1 121
C   XMAX(3)    STANDARD DEVIATION OF DATA ABOUT SINE WAVE FIT USING   C PSG1 122
C              XMAX(1) AND XMAX(2).                                   C PSG1 123
C                                                                     C PSG1 124
C   XMIN(1),XMIN(2)  PARAMETERS FOR LINEAR REGRESSION BETWEEN         C PSG1 125
C             TMAX AND TMIN.                                          C PSG1 126
C                                                                     C PSG1 127
C   XMIN(3)   STANDARD DEVIATION OF TMIN.                             C PSG1 128
C                                                                     C PSG1 129
C   ZAIRT      AVERAGE DAILY AIR TEMPERATURE FOR TIMESTEP, DEGREES    C PSG1 130
C              CELCIUS.  DRIVING VARIABLE USED BY OTHER SUBMODELS.    C PSG1 131
C                                                                     C PSG1 132
C   ZESUM     SUM FROM OCTOBER 1 OF ZEVAP, MM.                        C PSG1 133
C                                                                     C PSG1 134
C   ZEVAP      POTENTIAL EVAPORATION, MM/TIMESTEP.                    C PSG1 135
C              DRIVING VARIABLE USED BY OTHER SUBMODELS.              C PSG1 136
C                                                                     C PSG1 137
C   ZPHPD       PHOTOPERIOD OF FIRST DAY IN TIMESTEP, HOURS.  DRIVING C PSG1 138
C              VARIABLE USED BY OTHER SUBMODELS.                      C PSG1 139
C                                                                     C PSG1 140
C   ZRAIN      TOTAL PRECIPITATION FOR TIMESTEP, MM.  DRIVING VARIABLEC PSG1 141
C              USED BY OTHER SUBMODELS.                               C PSG1 142
C                                                                     C PSG1 143
C   ZRH        AVERAGE DAILY RELATIVE HUMIDITY, DECIMAL FRACTION FROM C PSG1 144
C              0-1.  DRIVING VARIABLE USED BY OTHER SUBMODELS.        C PSG1 145
C                                                                     C PSG1 146
C   ZRINT      PRECIPITATION INTENSITY, MM/HR. DRIVING VARIABLE USED  C PSG1 147
C              BY OTHER SUBMODELS.                                    C PSG1 148
C                                                                     C PSG1 149
C   ZRSUM      SUM FROM OCTOBER 1 OF ZRAIN, MM.                       C PSG1 150
C                                                                     C PSG1 151
C   ZSUN       AVERAGE DAILY PER CENT POSSIBLE SUNLIGHT FOR TIMESTEP, C PSG1 152
C              DECIMAL FRACTION FROM 0-1.  DRIVING VARIABLE USED      C PSG1 153
C              BY OTHER SUBMODELS.                                    C PSG1 154
C                                                                     C PSG1 155
C   ZTMAX      AVERAGE DAILY MAXIMUM TEMPERATURE FOR TIMESTEP.        C PSG1 156
C              DEGREES CELCIUS. DRIVING VARIABLE USED BY OTHER        C PSG1 157
C              SUBMODELS.                                             C PSG1 158
C                                                                     C PSG1 159
C   ZTMIN      AVERAGE DAILY MINIMUM TEMPERATURE FOR TIMESTEP,        C PSG1 160
C              DEGREES CELCIUS.  DRIVING VARIABLE USED BY OTHER       C PSG1 161
C              SUBMODELS.                                             C PSG1 162
C                                                                     C PSG1 163
C   ZWIND      AVERAGE DAILY WIND SPEED FOR TIMESTEP.  DRIVING        C PSG1 164
C              VARIABLE USED BY OTHER SUBMODELS.                      C PSG1 165
C                                                                     C PSG1 166
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC PSG1 167
C                                                                       PSG1 168
C                                                                       PSG1 169
C                                                                       PSG1 170
      SMAX=0.                                                           PSG1 171
      SMIN=0.                                                           PSG1 172
      SPREC=0.                                                          PSG1 173
      SSUN=0.                                                           PSG1 174
      SWIND=0.                                                          PSG1 175
      SHUM=0.                                                           PSG1 176
```

```
      JDAY=PMJDAT
      MDAY=JDAY
C
C
C
C  MAIN LOOP
      DO 3 I=1,PMOY
      MTIME=MDAY/28+1
      IF (MTIME .GT. 13) MTIME=13
C
C
C  READ IN TEMP AND PRECIP DATA IF FLAG SET
      IF(.NOT.READ)GOTO100
  020 READ(8,25,END=500)TMAX,TMIN,PRECIP
  025 FORMAT(7X,2F6.1,36X,F6.1)
      GOTO030
C
  500 READ=.FALSE.
C  IF WE REACH HERE THEN ADDITIONAL TEMPS AND PRECIP WILL BE
C  SUPPLIED BY THE STOCHASTIC WEATHER GENERATOR.
      GOTO100
C
  030 CONTINUE
C
C  NOW CHECK DATA FOR BEING REASONABLE
      IF(TMAX.LT.TMIN)GOTO040
      IF((TMAX.LT.-30.).OR.(TMAX.GT.45.))GOTO040
      IF((TMIN.LT.-30.).OR.(TMIN.GT.45.))GOTO040
      IF(((PRECIP.LT.50.).AND.(PRECIP.GE.0.0)).OR.(PRECIP.GT.5000.))
     *  GOTO060
C  SOMETIMES ZERO PRECIP IN DSCODES IS LISTED AS A RIDICULOUSLY LARGE NO
C
C  SOMETHING WRONG - WRITE OFFENDING CARD AND STOP
  040 WRITE(6,50)PMJDAT,TMAX,TMIN,PRECIP
  050 FORMAT(' ',I5,3F12.5)
      STOP
C
C
  060 CONTINUE
      SMAX=SMAX+TMAX
      SMIN=SMIN+TMIN
      SPREC=SPREC+PRECIP
C  LOAD PRECIP INFORMATION FOR ZRINT CALCULATION
      TRAIN(I)=PRECIP
      GOTO2
C
C
C
  100 CONTINUE
C  TEMPERATURE SECTION
C  TMAX IS A QUADRATIC FIT TO THE YEAR'S TEMPERATURES
C  RNOR IS 'RANDOM NORMAL' NUMBER GENERATOR ( .68 OF VALUES BETWEEN
C  -1 AND +1)
C  XMAX(3) IS THE STANDARD DEVIATION ABOUT THIS FIT.
C  THE FIRST TMAX BELOW IS A SINE WAVE FIT TO THE YEAR'S MEAN MAX TEMP
C  .017214 = 2 * PI / 365.
C  1.570796 = PI / 2.
      TMAX = XMAX(1) + XMAX(2)*SIN(.017214*MOD((PMJDAT+344.),365.)
     C  -1.570796)
C  THE FOLLOWING TMAX IS THE DAY'S MAXIMUM TEMPERATURE.
  001 TMAX = TMAX + XMAX(3)*RNOR(ISEED)
      TMIN=XMIN(1)+XMIN(2)*TMAX
      TMIN=TMIN+XMIN(3)*RNOR(ISEED)
      IF (TMIN .GT. TMAX) GO TO 1
      SMAX=SMAX+TMAX
      SMIN=SMIN+TMIN
C
C  PRECIPITATION SECTION
C  IYEST HAS SOMETHING TO DO WITH WHETHER OR NOT WE HAD RAIN YESTERDAY.
      IYEST=IPROB(RAIN(IYEST,MTIME),1)
      IF(IYEST .NE. 2) TRAIN(I)=0.0
      IF (IYEST .NE. 2) GO TO 2
      J=IPROB(PREC(1,MTIME),5)
      SPREC=SPREC+AMT(J)
      TRAIN(I)=AMT(J)
C
C  WIND SECTION
  002 J=IPROB(WIND(1,MTIME),7)
      SWIND=SWIND+(J-1)*8.05
C
C  PERCENT POSSIBLE SUNLIGHT SECTION
      J=IPROB(SUN(1,MTIME),10)
      SSUN=SSUN+(J-1)*0.1
C
C  RELATIVE HUMIDITY SECTION
      J=IPROB(HUM(1,MTIME),10)
      SHUM=SHUM+(J-1)*0.1
  003 MDAY=MDAY+1
C  END MAIN LOOP
C
```

```
C
C
      X=1./PMDT
      ZTMAX=SMAX*X
      ZTMIN=SMIN*X
      ZWIND=SWIND*X
C
C  HERE'S A FACTOR FOR ADJUSTING AVERAGE WIND SPEED IF SITE IS KNOWN TO
C  BE DIFFERENT FROM MEASURING STATION
      ZWIND = ZWIND * PWFAC
C  ZWIND IS IN  KM/HR
C
      ZSUN=SSUN*X
      ZRH=SHUM*X
      ZRAIN=SPREC
      ZAIRT=(ZTMAX+ZTMIN)/2.
C
C  THIS IS WILKIN'S PHOTOPERIOD CALCULATION
      DECL=ARSIN(SIN(6.2831853*23.45/360.)*
     C COS(MOD((PMJDAT+193.),365.)*6.2831853/365.))
      X=(SIN(LAT*6.2831853/360.)*TAN(DECL)+SIN(.875*6.2831853/360.))/
     C COS(LAT*6.2831853/360.)
      IF (X .GT. 1.) X=1.
      IF (X .LT. -1.) X=-1.
      ZPHPD=12. + .1333333*ARSIN(X)*360./6.2831853
C
C
C  DETERMINE RAIN INTENSITY
      CALL RINT(ZRINT, TRAIN, ZRAIN, PMDT, PMJDAT, ISEED)
C  ZRINT IS RETURNED IN MM/HR
C
C
C  DETERMINE ZEVAP
      CALL EVAP
C
C  ZEVAP IS IN MM/TIMESTEP
C
C
C  THIS IS THE WEATHER MODIFICATION
      ZRAIN=FACTOR*ZRAIN
C
C
C  SUM SOME VARIABLES FROM OCT 1 - - WATER YEAR
      IF(PMJDAT.LT.270)GOTO900
      IF(ABS(PMJDAT-270) .GE. PMDT) GO TO 900
      ZRSUM=0.0
      ZESUM=0.0
  900 ZRSUM=ZRSUM+ZRAIN
      ZESUM=ZESUM+ZEVAP
C
C
      IF(NNN .LT. 58) GO TO 910
      WRITE(4,930)
      NNN=0
  910 WRITE(4,940)PMJDAT,ZTMAX,ZTMIN,ZRH,ZPHPD,ZRAIN,ZRINT,ZRSUM,ZSUN
     *,ZEVAP,ZESUM,ZWIND
      NNN=NNN+1
  930 FORMAT('1', 'PMJDAT    ZTMAX     ZTMIN     ZRH      ZPHPD      ZRAI
     *N     ZRINT       ZRSUM         ZSUN       ZEVAP       ZESUM      ZWIN
     *D', /)
  940 FORMAT(' ', 4X, I3, 2(4X,F6.2),5X, F4.2, 3(5X, F5.2),
     * 2(5X, F7.2), 5X, F5.2, 5X, F7.2, 5X, F5.2)
C
      RETURN
C
C
      ENTRY ZINIT
C
      COMMON TIME,TSTART,TEND,DT,DTPR,DTPL,
     1CAAR(6,8),CADEST(6,8),CADETH,CAUWST,CDDL(12,4),CHD(10),CHDX(10),
     2CHDXX(10),CNFIX,CNSDF,CNSDS,CNSOMF,CNSOMS,CNSUP,CVDETH(12,4),
     3CVLTFR(11,4),CVPHEN(6,8),CVPHS(6),CVRDST(6,10),CVTSPR(6,8),CVVCOV,
     4CWINF,CWPSI(10),PMDT,PMDTPL,PMDTPR,PMFGPS,PMJDAT,PMN,PMNCOH,PMNSP,
     5PWK(10),XAA(4),XAAVWT(4,8),XAAWT(4),XAFTS(4),XAFWT(4),XANUMB(4,8),
     6XASA(4),XASAWT(4),XAYNG(4),XAYWT(4),XHSOLT(10),XNMN,XNSOMF,XNSOMS,
     7XVFG(6,5),XVLITR(12,4),XVPLNT(6,8),XVTOTL,XWTHTA(10),XWSTND,ZAIRT,
     8ZEVAP,ZESUM,ZPHPD,ZRAIN,ZRSUM,ZRH,ZRINT,ZRISUM,ZSUN,ZTMAX,ZTMIN,
     9ZWIND
C
      LOGICAL READ
C
      REAL LAT
C
      INTEGER PMJDAT
C
      COMMON /SUNNY/ IYEST,ISEED
C
      DIMENSION XMAX(3),XMIN(3),RAIN(2,13),PREC(6,13),AMT(6),
     *   WIND(7,13), SUN(10,13), HUM(10,13)
      DIMENSION TRAIN(20)
      DIMENSION RCHECK(20)
```

```
C                                                                       PSG1 358
      WRITE(4,930)                                                      PSG1 359
      NNN=0                                                             PSG1 360
C                                                                       PSG1 361
      READ(5,710) RCHECK                                                PSG1 362
      WRITE(6,720) RCHECK                                               PSG1 363
      READ(5,710) RCHECK                                                PSG1 364
      WRITE(6,720) RCHECK                                               PSG1 365
C                                                                       PSG1 366
      READ(5,/) ISEED                                                   PSG1 367
      WRITE(6,/) ISEED                                                  PSG1 368
C                                                                       PSG1 369
      READ(5,/) LAT                                                     PSG1 370
      WRITE(6,/) LAT                                                    PSG1 371
C                                                                       PSG1 372
      READ(5,/) XMAX,XMIN                                               PSG1 373
      WRITE(6,/) XMAX                                                   PSG1 374
      WRITE(6,/) XMIN                                                   PSG1 375
C                                                                       PSG1 376
      READ(5,/) RAIN                                                    PSG1 377
      WRITE(6,/) RAIN                                                   PSG1 378
C                                                                       PSG1 379
      READ(5,/) PREC                                                    PSG1 380
      WRITE(6,/) PREC                                                   PSG1 381
C                                                                       PSG1 382
      READ(5,/) AMT                                                     PSG1 383
      WRITE(6,/) AMT                                                    PSG1 384
C                                                                       PSG1 385
      READ(5,/) WIND                                                    PSG1 386
      WRITE(6,/) WIND                                                   PSG1 387
C                                                                       PSG1 388
      READ(5,710) RCHECK                                                PSG1 389
      WRITE(6,720) RCHECK                                               PSG1 390
C                                                                       PSG1 391
      READ(5,/) SUN                                                     PSG1 392
      WRITE(6,/) SUN                                                    PSG1 393
C                                                                       PSG1 394
      READ(5,/) HUM                                                     PSG1 395
      WRITE(6,/) HUM                                                    PSG1 396
C                                                                       PSG1 397
      READ(5,/) PWFAC                                                   PSG1 398
      WRITE(6,/) PWFAC                                                  PSG1 399
C                                                                       PSG1 400
      READ(5,/)READ                                                     PSG1 401
      WRITE(6,/) READ                                                   PSG1 402
C                                                                       PSG1 403
      READ(5,/)FACTOR                                                   PSG1 404
      WRITE(6,/) FACTOR                                                 PSG1 405
C                                                                       PSG1 406
C                                                                       PSG1 407
      READ(5,710) RCHECK                                                PSG1 408
      WRITE(6,720) RCHECK                                               PSG1 409
C                                                                       PSG1 410
  710 FORMAT(20A4)                                                      PSG1 411
  720 FORMAT(' ', 20A4)                                                 PSG1 412
C                                                                       PSG1 413
C                                                                       PSG1 414
      IYEST=1                                                           PSG1 415
C                                                                       PSG1 416
C  CALL INITIALIZATION SECTION OF SUBROUTINES EVAP AND RINT             PSG1 417
      CALL EVINIT                                                       PSG1 418
      CALL RIINIT                                                       PSG1 419
C                                                                       PSG1 420
C                                                                       PSG1 421
      RETURN                                                            PSG1 422
      END                                                               PSG1 423
```

Subroutine PSWG, Version II

```
      SUBROUTINE PSWG                                                   PSG2 001
C                                                                       PSG2 002
C                                                                       PSG2 003
C  THIS VERSION OF PSWG USED IN 5-YEAR RUNS.                            PSG2 004
C  WEATHER VARIABLES ARE READ FROM WEATHER DATA FILE.                   PSG2 005
C                                                                       PSG2 006
C                                                                       PSG2 007
C                                                                       PSG2 008
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC PSG2 009
C                                                                     C PSG2 010
C      VARIABLE DICTIONARY                                            C PSG2 011
C                                                                     C PSG2 012
C                                                                     C PSG2 013
C  DECL        ANGLE CALCULATED IN PHOTOPERIOD CALCULATION.           C PSG2 014
C                                                                     C PSG2 015
```

```
C  EVAP         SUBROUTINE WHICH CALCULATES POTENTIAL EVAPORATION.      C PSG2 016
C                                                                       C PSG2 017
C  FACTOR      PRECIPITATION FACTOR BY WHICH EACH EVENT IS              C PSG2 018
C               MULTIPLIED.                                             C PSG2 019
C                                                                       C PSG2 020
C  ISEED        SEED FROM WHICH RNOR GENERATES NEXT RANDOM NUMBER.      C PSG2 021
C                                                                       C PSG2 022
C  JDAY         JULIAN DATE AT BEGINNING OF TIMESTEP.                   C PSG2 023
C                                                                       C PSG2 024
C  LAT          LATITUDE OF SITE.                                       C PSG2 025
C                                                                       C PSG2 026
C  NNN          COUNTER TO DETERMINE WHEN TO START PRINTING ON          C PSG2 027
C               NEXT PAGE.                                              C PSG2 028
C                                                                       C PSG2 029
C  PMDT         TIMESTEP LENGTH, DAYS.                                  C PSG2 030
C                                                                       C PSG2 031
C  PRECIP       PRECIPITATION FOR CURRENT DAY, MM.                      C PSG2 032
C                                                                       C PSG2 033
C  PMJDAY       JULIAN DATE AT BEGINNING OF TIMESTEP.  VALUE            C PSG2 034
C               DETERMINED IN MAIN PROGRAM.                             C PSG2 035
C                                                                       C PSG2 036
C  PWFAC        FACTOR FOR ADJUSTING AVERAGE WIND SPEED IF SITE IS      C PSG2 037
C               KNOWN TO HAVE DIFFERENT AVERAGE WIND SPEED.             C PSG2 038
C                                                                       C PSG2 039
C  RCHECK(20)   VARIABLE USED TO READ AND WRITE COMMENTS IN WITH        C PSG2 040
C               INITIAL DATA.                                           C PSG2 041
C                                                                       C PSG2 042
C  RINT         SUBROUTINE WHICH CALCULATES PRECIPITATION INTENSITY.    C PSG2 043
C                                                                       C PSG2 044
C  SHUM         RUNNING SUM OF UP TO PMDT DAYS' RELATIVE HUMIDITY.      C PSG2 045
C                                                                       C PSG2 046
C  SMAX         RUNNING SUM OF UP TO PMDT DAYS' MAXIMUM TEMPERATURES,   C PSG2 047
C               DEGREES CELCIUS.                                        C PSG2 048
C                                                                       C PSG2 049
C  SMIN         RUNNING SUM OF UP TO PMDT DAYS' MINIMUM TEMPERATURES,   C PSG2 050
C               DEGREES CELCIUS.                                        C PSG2 051
C                                                                       C PSG2 052
C  SPREC        RUNNING SUM OF UP TO PMDT DAYS' PRECIPITATIONEVENTS,MM.C PSG2 053
C                                                                       C PSG2 054
C  SSUN         RUNNING SUM OF UP TO PMDT DAYS' PER CENT POSSIBLE       C PSG2 055
C               SUNLIGHT                                                C PSG2 056
C                                                                       C PSG2 057
C  SWIND        RUNNING SUM OF UP TO PMDT DAYS' AVERAGE WIND SPEEDS.    C PSG2 058
C               KM/HR.                                                  C PSG2 059
C                                                                       C PSG2 060
C  TMAX         MAXIMUM TEMPERATURE FOR CURRENT DAY, DEGREES CELCIUS.   C PSG2 061
C                                                                       C PSG2 062
C  TMIN         MINIMUM TEMPERATURE FOR CURRENT DAY, DEGREES CELCIUS.   C PSG2 063
C                                                                       C PSG2 064
C  TRAIN(20)    ARRAY WHICH HOLDS DAILY PRECIP VALUES FOR TIMESTEP      C PSG2 065
C               (USED BY RAIN INTENSITY ROUTINE).                       C PSG2 066
C                                                                       C PSG2 067
C  ZAIRT        AVERAGE DAILY AIR TEMPERATURE FOR TIMESTEP, DEGREES     C PSG2 068
C               CELCIUS.  DRIVING VARIABLE USED BY OTHER SUBMODELS.     C PSG2 069
C                                                                       C PSG2 070
C  ZESUM       SUM FROM OCTOBER 1 OF ZEVAP, MM.                         C PSG2 071
C                                                                       C PSG2 072
C  ZEVAP        POTENTIAL EVAPORATION, MM/TIMESTEP.                     C PSG2 073
C               DRIVING VARIABLE USED BY OTHER SUBMODELS.               C PSG2 074
C                                                                       C PSG2 075
C  ZPHPD         PHOTOPERIOD OF FIRST DAY IN TIMESTEP, HOURS.  DRIVING C PSG2 076
C               VARIABLE USED BY OTHER SUBMODELS.                       C PSG2 077
C                                                                       C PSG2 078
C  ZRAIN        TOTAL PRECIPITATION FOR TIMESTEP, MM.  DRIVING VARIABLEC PSG2 079
C               USED BY OTHER SUBMODELS.                                C PSG2 080
C                                                                       C PSG2 081
C  ZRH          AVERAGE DAILY RELATIVE HUMIDITY, DECIMAL FRACTION FROM C PSG2 082
C               0-1.  DRIVING VARIABLE USED BY OTHER SUBMODELS.         C PSG2 083
C                                                                       C PSG2 084
C  ZRINT        PRECIPITATION INTENSITY, MM/HR. DRIVING VARIABLE USED   C PSG2 085
C               BY OTHER SUBMODELS.                                     C PSG2 086
C                                                                       C PSG2 087
C  ZRSUM        SUM FROM OCTOBER 1 OF ZRAIN, MM.                        C PSG2 088
C                                                                       C PSG2 089
C  ZSUN         AVERAGE DAILY PER CENT POSSIBLE SUNLIGHT FOR TIMESTEP, C PSG2 090
C               DECIMAL FRACTION FROM 0-1.   DRIVING VARIABLE USED      C PSG2 091
C               BY OTHER SUBMODELS.                                     C PSG2 092
C                                                                       C PSG2 093
C  ZTMAX        AVERAGE DAILY MAXIMUM TEMPERATURE FOR TIMESTEP.         C PSG2 094
C               DEGREES CELCIUS. DRIVING VARIABLE USED BY OTHER         C PSG2 095
C               SUBMODELS.                                              C PSG2 096
C                                                                       C PSG2 097
C  ZTMIN        AVERAGE DAILY MINIMUM TEMPERATURE FOR TIMESTEP,         C PSG2 098
C               DEGREES CELCIUS.   DRIVING VARIABLE USED BY OTHER       C PSG2 099
C               SUBMODELS.                                              C PSG2 100
C                                                                       C PSG2 101
C  ZWIND        AVERAGE DAILY WIND SPEED FOR TIMESTEP.   DRIVING        C PSG2 102
C               VARIABLE USED BY OTHER SUBMODELS.                       C PSG2 103
C                                                                       C PSG2 104
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC PSG2 105
```

```
C
C
      SMAX=0.
      SMIN=0.
      SPREC=0.
      SWIND=0.
      SSUN=0.
      SHUM=0.
      JDAY=PMJDAT
      MDAY=JDAY
C
C
C
C  MAIN LOOP
      DO 3 I=1,PMDT
C  READ TODAY'S DATA
      READ(8,30,END=500)TMAX,TMIN,PRECIP,WIND,PPS,RH
   30 FORMAT(10X,2F5.0,F5.2,F5.1,2F5.2)
C
      GOTO501
  500 CONTINUE
      STOP
  501 CONTINUE
C
C  CONVERT TO METRIC UNITS
C
C  TEMPS FROM F TO C
      TMAX=(TMAX-32.)*0.555555
      TMIN=(TMIN-32.)*0.555555
C
C  PRECIP FROM INCHES TO MM
      PRECIP=PRECIP*25.4
C
C  ABERAGE WIND SPEED FROM MILES / HR  TO KM/HR
      WIND=WIND*1.60934
C
C
C  CHECK IF VALUES ARE SENSIBLE
      IF(TMIN.GT.TMAX)GOTO40
      IF((TMAX.GT.42.).OR.(TMIN.LT.-43.))GOTO40
      IF((PRECIP.GT.40.).OR.(PRECIP.LT.0.0))GOTO40
      IF((WIND.GT.60.).OR.(WIND.LT.0.0))GOTO40
      IF((PPS.GT.1.0).OR.(PPS.LT.0.0))GOTO40
      IF((RH.LE.1.00).AND.(RH.GE.0.0))GOTO60
C
C  IF WE REACH HERE SOMETHING IS WRONG.   WRITE OFFENDING CARD AND STOP.
   40 WRITE(6,50)TMAX,TMIN,PRECIP,WIND,PPS,RH,PMJDAT
   50 FORMAT(' ',7F10.3)
      STOP
C
C
   60 CONTINUE
      SMAX=SMAX+TMAX
      SMIN=SMIN+TMIN
      SPREC=SPREC+PRECIP
      SWIND=SWIND+WIND
      SSUN=SSUN+PPS
      SHUM=SHUM+RH
C
C  LOAD TRAIN FOR USE IN SUBROUTINE RINT
      TRAIN(I)=PRECIP
C
    3 CONTINUE
C  END MAIN LOOP
C
C
C
C
      X=1./PMDT
C
      ZTMAX=SMAX*X
      ZTMIN=SMIN*X
      ZWIND=SWIND*X
C  HERE'S A FACTOR FOR ADJUSTING AVERAGE WIND SPEED IF SITE IS KNOWN TO
C  BE DIFFERENT FROM MEASURING STATION
      ZWIND = ZWIND * PWFAC
      ZSUN=SSUN*X
      ZRH=SHUM*X
      ZRAIN=SPREC
      ZAIRT=(ZTMAX+ZTMIN)/2.
C
C  THIS IS WILKIN'S PHOTOPERIOD CALCULATION
      DECL=ARSIN(SIN(6.2831853*23.45/360.)*
     C COS(MOD((PMJDAT+193.),365.)*6.2831853/365.))
      X=(SIN(LAT*6.2831853/360.)*TAN(DECL)+SIN(.875*6.2831853/360.))/
     C COS(LAT*6.2831853/360.)
      IF (X .GT. 1.) X=1.
      IF (X .LT. -1.) X=-1.
      ZPHPD=12. + .1333333*ARSIN(X)*360./6.2831853
C
```

```
C  DETERMINE ZRINT
      CALL RINT(ZRINT, TRAIN, ZRAIN, PMDT, PMJDAT, ISEED)
C
C  DETERMINE ZEVAP
      CALL EVAP
C
C
C
C
C  HERE'S THE RAINFALL MODIFICATION
      ZRAIN=FACTOR*SPREC
C
C
C
C  SUMMING SECTION
C  DO FOLLOWING SUMS FROM OCTOBER 1 - - WATER YEAR
      IF(PMJDAT.LT.270)GOTO900
      IF(ABS(PMJDAT-270) .GE. PMDT) GO TO 900
      ZRSUM=0.0
      ZESUM=0.0
  900 ZRSUM=ZRSUM+ZRAIN
      ZESUM=ZESUM+ZEVAP
      IF(NNN .LT. 58) GO TO 910
      WRITE(4,930)
      NNN=0
  910 WRITE(4,940)PMJDAT,ZTMAX,ZTMIN,ZRH,ZPHPD,ZRAIN,ZRINT,ZRSUM,ZSUN
     *,ZEVAP,ZESUM,ZWIND
      NNN=NNN+1
  930 FORMAT('1', 'PMJDAT    ZTMAX     ZTMIN     ZRH      ZPHPD     ZRAI
     *N     ZRINT        ZRSUM         ZSUN       ZEVAP      ZESUM      ZWIN
     *D', /)
  940 FORMAT(' ', 4X, I3, 2(4X,F6.2),5X, F4.2, 3(5X, F5.2),
     * 2(5X, F7.2), 5X, F5.2, 5X, F7.2, 5X, F5.2)
C
C
      RETURN
C
      ENTRY ZINIT
C
C
      COMMON TIME,TSTART,TEND,DT,DTPR,DTPL,
     1CAAR(6,8),CADEST(6,8),CADETH,CAUWST,CDDL(12,4),CHD(10),CHDX(10),
     2CHDXX(10),CNFIX,CNSDF,CNSDS,CNSOMF,CNSOMS,CNSUP,CVDETH(12,4),
     3CVLTFR(11,4),CVPHEN(6,8),CVPHS(6),CVRDST(6,10),CVTSPR(6,8),CVVCOV,
     4CWINF,CWPST(10),PMDT,PMDTPL,PMDTPR,PMFGPS,PMJDAT,PMN,PMNCOH,PMNSP,
     5PWK(10),XAA(4),XAAVWT(4,8),XAAWT(4),XAFTS(4),XAFWT(4),XANUMB(4,8),
     6XASA(4),XASAWT(4),XAYNG(4),XAYWT(4),XHSOLT(10),XNMN,XNSOMF,XNSOMS,
     7XVFG(6,5),XVLITR(12,4),XVPLNT(6,8),XVTOTL,XWTHTA(10),XWSTND,ZATRT,
     8ZEVAP,ZESUM,ZPHPD,ZRAIN,ZRSUM,ZRH,ZRINT,ZRISUM,ZSUN,ZTMAX,ZTMIN,
     9ZWIND
C
      REAL LAT
C
      COMMON /SUNNY/ IYEST,ISEED
C
      DIMENSION TRAIN(20)
      DIMENSION RCHECK(20)
C
C
      WRITE(4,930)
      NNN=0
C
C
      READ(5,710) RCHECK
      WRITE(6,720) RCHECK
C
      READ(5,/) ISEED
      WRITE(6,/) ISEED
C
      READ(5,/) LAT
      WRITE(6,/) LAT
C
      READ(5,/) PWFAC
      WRITE(6,/) PWFAC
C
      READ(5,/)FACTOR
      WRITE(6,/) FACTOR
C
      READ(5,710) RCHECK
      WRITE(6,720) RCHECK
C
  710 FORMAT(20A4)
  720 FORMAT(' ', 20A4)
C
C  CALL INITTIALIZATION SECTIONS OF EVAP AND RINT
      CALL EVINIT
      CALL RIINIT
C
C
      RETURN
      END
```

Subroutine EVAP

```
      SUBROUTINE EVAP
C
C
C  EVAP CALCULATES POTENTIAL EVAPORATION FROM SOIL SURFACE DURING
C  TIMESTEP, IN MM.
C  THIS VERSION OF EVAP USES BLANEY CRIDDLE METHOD.
C  IT CLOSELY FOLLOWS CALCULATION DONE BY GRIFFIN AND HANKS IN THEIR
C  D.B. RESEARCH MEMORANDUM.
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                    VARIABLE DICTIONARY FOR EVAP                      C
C                                                                      C
C                                                                      C
C  A, B, C, Z TEMPORARY VARIABLES USED IN DAYLIGHT CALCULATION.        C
C                                                                      C
C  ARRAY(5,2) PAIRS OF DATA POINTS FOR INTERPOLATION BY F1 IN          C
C             CALCULATING BLANEY CRIDDLE FACTOR.                       C
C                                                                      C
C  CVVCOV     FRACTION OF SOIL SURFACE COVERED BY ANNUAL AND           C
C             PERENNIAL VEGETATION.                                    C
C                                                                      C
C  DALITE(365)  ARRAY INDEXED BY JULIAN DATE .  GIVES NUMBER OF        C
C             MINUTES OF DAYLIGHT ON THAT JULIAN DATE.                 C
C                                                                      C
C  DLTFR(365) ARRAY INDEXED BY JULIAN DATE AND IS FRACTION OF YEAR'S  C
C             DAYLIGHT WHICH FALLS ON THAT JULIAN DATE.                C
C                                                                      C
C  F1         FUNCTION WHICH LINEARLY INTERPOLATES BETWEEN PAIRS OF    C
C             DATA POINTS TO DETERMINE DEPENDENT VARIABLE.             C
C                                                                      C
C  LAT        LATITUDE OF SITE, DEGREES.                               C
C                                                                      C
C  MM         FLAG USED ONLY IN CASE PMJDAT IS EVER ZERO.              C
C                                                                      C
C  NPTS       NUMBER OF PAIRS OF DATA POINTS ACTUALLY USED IN ARRAY.   C
C                                                                      C
C  PMDT       LENGTH OF TIMESTEP, DAYS.                                C
C                                                                      C
C  PMJDAT     JULIAN DATE OF FIRST DAY OF TIMESTEP.                    C
C                                                                      C
C  RCHECK(20) ARRAY USED TO READ AND WRITE COMMENTS IN WITH            C
C             INITIAL DATA.                                            C
C                                                                      C
C  TOTMIN     MINUTES OF DAYLIGHT IN YEAR.                             C
C                                                                      C
C  ZAIRT      AVERAGE TIMESTEP AIR TEMPERATURE, DEGREES CELCIUS.       C
C                                                                      C
C  ZEVAP      POTENTIAL EVAPORATION FOR TIMESTEP, MM,  DRIVING         C
C             VARIABLE USED BY OTHER SUBMODELS.                        C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
      ZEVAP=F1(ZAIRT, ARRAY1, NPTS)
      IF(PMJDAT .GT. 0) GO TO 670
      PMJDAT=1
      MM=1
  670 CONTINUE
C  IF ZAIRT IN NEXT EQUATION LOOKS LIKE IT'S BEING CHANGED TO DEGREES
C  F,  THAT'S BECAUSE IT IS.
      ZEVAP=ZEVAP*(1.8*ZAIRT + 32.)*DLTFR(PMJDAT)*(1.-CVVCOV)*25.4*PMDT
      CVVCOV=0.0
      IF(MM .EQ. 0) GO TO 680
      PMJDAT=0
      MM=0
  680 CONTINUE
C
C
      RETURN
C
      ENTRY EVINIT
C
      COMMON TIME, TSTART, TEND, DT, DTPR, DTPL,
     1CAAR(6,8),CADEST(6,8),CADETH,CAUWST,CODL(12,4),CHD(10),CHDX(10),
     2CHDXX(10),CNFIX,CNSDF,CNSDS,CNSOMF,CNSOMS,CNSUP,CVDETH(12,4),
     3CVLTFR(11,4),CVPHEN(6,8),CVPHS(6),CVRDST(6,10),CVTSPR(6,8),CVVCOV,
     4CWINF,CWPSI(10),PMDT,PMDTPL,PMDTPR,PMFGPS,PMJDAT,PMN,PMNCOM,PMNSP,
     5PWK(10),XAA(4),XAAVWT(4,8),XAAWT(4),XAFTS(4),XAFWT(4),XANUMB(4,8),
     6XASA(4),XASAWT(4),XAYNG(4),XAYWT(4),XHSOLT(10),XNMN,XNSOMF,XNSOMS,
     7XVFG(6,5),XVLITR(12,4),XVPLNT(6,8),XVTOTL,XWTHTA(10),XWSTND,ZAIRT,
     8ZEVAP,ZESUM,ZPHPD,ZRAIN,ZRSUM,ZRH,ZRINT,ZRISUM,ZSUN,ZTMAX,ZTMIN,
     9ZWIND
C
      DIMENSION ARRAY1(5,2), DALITE(365), DLTFR(365)
      DIMENSION RCHECK(20)
C
```

```
      INTEGER PMJDAT
C
      REAL LAT
C
C
      READ(5,710) RCHECK
      WRITE(6,720) RCHECK
C
      READ(5,/) LAT
      WRITE(6,/) LAT
      READ(5,/) ARRAY1
      WRITE(6,/) ARRAY1
      READ(5,/)NPTS
      WRITE(6,/) NPTS
C
  710 FORMAT(20A4)
  720 FORMAT(' ', 20A4)
C
C  FROM GRIFFIN, HANKS BIOME REPORT
      DO 650 I=1, 365
      A=730.-.274*LAT+.00793*(LAT**2)
      B=34.2-.78*LAT+.1*(LAT**2)
      C=I
      Z=2.*3.1416*((C+285.)/365.)
C  LENGTH OF DAY IN MINUTES
      DALITE(I)=A+B*SIN(Z)
      TOTMIN=DALITE(I)+TOTMIN
  650 CONTINUE
C
C   FRACTION OF DAYLIGHT FOR EACH DAY
      DO 660 I=1, 365
      DLTFR(I)=DALITE(I)/TOTMIN
  660 CONTINUE
C
      WRITE(6,681) DALITE
      WRITE(6,681) TOTMIN
  681 FORMAT(' ', 10F12.5)
C
      READ(5,710) RCHECK
      WRITE(6,720) RCHECK
C
      RETURN
      END
```

SUBROUTINE RINT

```
      SUBROUTINE RINT(ZRINT, TRAIN, ZRAIN, PMDT, PMJDAT, ISEED)
C
C  RAIN INTENSITY SUBROUTINE   4-76  PAUL LOMMEN
C  RINT CALCULATES PRECIPITATION INTENSITY IN MM/HR.   USED BY WATER.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                         VARIABLE DICTIONARY FOR RINT                 C
C                                                                      C
C  A, B        PARAMETERS IN HIGH INTENSITY CALCULATION.               C
C                                                                      C
C  ARRAY(10)   HOLDS UP TO 5 PAIRS OF DATA POINTS  FOR INTERPOLATION   C
C              BY FUNCTION F1 IN DETERMINING NORMAL INTENSITY.         C
C                                                                      C
C  BA          INTERMEDIATE VARIABLE IN HIGH INTENSITY CALCULATION.    C
C                                                                      C
C  B1          INTERMEDIATE VARIABLE IN HIGH INTENSITY CALCULATION.    C
C                                                                      C
C  F1          FUNCTION WHICH LINEARLY INTERPOLATES BETWEEN NPTS PAIRS C
C              OF DATA POINTS.                                         C
C                                                                      C
C  ISEED       INTEGER FROM WHICH RANDOM NUMBERS ARE GENERATED.        C
C                                                                      C
C  N           INDEX OF TIME OF YEAR.  IN MIDDLE OF YEAR N=1.          C
C              OTHER TIMES OF YEAR N=2.                                C
C                                                                      C
C  NPTS        NUMBER OF PAIRS OF DATA POINTS IN NORMAL  INTENSITY     C
C              CALCULATION.                                            C
C                                                                      C
C  PDAY1,PDAY2  BEGINNING AND ENDING JULIAN DATES OF CENTRAL           C
C              SEGMENT OF YEAR.                                        C
C                                                                      C
C  PMDT        LENGTH OF TIMESTEP, DAYS.                               C
C                                                                      C
C  PMJDAT      JULIAN DATE OF BEGINNING OF TIMESTEP.                   C
C                                                                      C
C  PTH         PRECIP AMOUNT ABOVE WHICH HIGH INTENSITY CAN OCCUR      C
```

```
C                                                                      C
C  PA(2)       COMPARED WITH R TO SEE IF HIGH INTENSITY ACTUALLY       C
C              OCCURRED.                                               C
C                                                                      C
C  R           TEMPORARY VARIABLE.   SET EQUAL TO RANDOM NUMBER        C
C                                                                      C
C  RCHECK(20)  ARRAY USED FOR READING AND WRITING COMMENTS IN WITH     C
C              INITIALIZATION DATA.                                    C
C                                                                      C
C  TR          TEMPORARY VARIABLE.                                     C
C                                                                      C
C  TRAIN(20)   DAILY PRECIPITATION AMOUNTS, MM.  ARRAY INDEXED BY      C
C              DAY OF TIMESTEP                                         C
C                                                                      C
C  X           TODAY'S RAIN                                            C
C                                                                      C
C  Y           TODAY'S PRECIPITATION INTENSITY , MM/HR.                C
C                                                                      C
C  YSUM        SUM OVER TIMESTEP OF DAILY PRECIP INTENSITY TIMES       C
C              PRECIP AMOUNT.                                          C
C                                                                      C
C  ZRAIN       TIMESTEP PRECIPTATION,MM.  DRIVING VARIABLE USED BY     C
C              WATER SUBMODEL.                                         C
C                                                                      C
C  ZRINT       RAIN INTENSITY, MM/HR.  DRIVING VARIABLE USED BY        C
C              WATER SUBMODEL.                                         C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      YSUM=0.0
C
C
      DO 550 I=1,PMDT
C
C  X IS TODAY'S PRECIP
C  Y IS TODAY'S PRECIP INTENSITY
      X=TRAIN(I)
C
C  DO WE HAVE ANY RAIN TODAY?
      IF(X .GT. 1.E-5) GO TO 100
C
      GO TO 550
C
C  IF X LARGE ENOUGH ALLOW FOR HIGH INTENSITIES
  100 IF(X .LT. PTH) GO TO 500
C
C  WILL NEED A RANDOM NUMBER TO DETERMINE IF WE HAVE HIGH INTENSITY
      R=RANDOM(ISEED)
C
C  DIVIDE YEAR INTO TWO SEGMENTS, ALLOWING DIFFERENT HIGH INTENSITY
C  PROBABILITIES IN EACH
C
C  IN CENTRAL SEGMENT N=1
      N=2
      IF((PMJDAT .GE. PDAY1) .AND. (PMJDAT .LE. PDAY2)) N=1
C  N NOW TELLS US WHICH SEGMENT OF YEAR WE'RE IN
C
      IF(R .GT. PA(N)) GO TO 500
C  GENERATE NEW (SEMI) RANDOM NUMBER
      R=R/PA(N)
C
  150 TR=BA*(1.-R)
      IF(TR .LE. 1.E-10) TR=1.E-10
C
      Y=-B1*ALOG(TR)
C
C  LIMIT Y TO AN INCH AN HOUR
      IF(Y .GT. 25.) Y=25.
      GO TO 520
C
C
C  NORMAL INTENSITY CALCULATION
  500 Y=F1(PMJDAT, ARRAY, NPTS)
C
C
  520 YSUM=X*Y + YSUM
C
  550 CONTINUE
C
C
      IF(ZRAIN .LE. 1.E-6) GO TO 570
      ZRINT = YSUM/ZRAIN
      GO TO 580
  570 ZRAIN=0.0
      ZRINT=0.0
  580 IF((ZRINT*24.*PMDT) .LT. ZRAIN)  ZRINT=ZRAIN/(24.*PMDT)
C
C
      RETURN
C
```

```
C
      ENTRY RIINIT
C
      DIMENSION PA(2), ARRAY(10), RCHECK(20)
      DIMENSION TRAIN(20)
C
      INTEGER PHOT
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      WRITE(6,630)
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      READ(5,/) PTH
      WRITE(6,620) PTH
      WRITE(6,630)
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      READ(5,/) PDAY1, PDAY2
      WRITE(6,620) PDAY1, PDAY2
      WRITE(6,630)
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      READ(5,/) ARRAY
      WRITE(6,620) ARRAY
      WRITE(6,630)
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      READ(5,/) NPTS
      WRITE(6,620) NPTS
      WRITE(6,630)
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      READ(5,/) PA
      WRITE(6,620) PA
      WRITE(6,630)
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      READ(5,/) A, B
      WRITE(6,620) A, B
      WRITE(6,630)
C
      BA=0.01*B/A
      B1=25.4/B
      WRITE(6,*/) BA, B1
C
      READ(5,600) RCHECK
      WRITE(6,610) RCHECK
      WRITE(6,630)
      WRITE(6,630)
      WRITE(6,630)
C
C
  600 FORMAT(20A4)
  610 FORMAT(' ', 20A4)
  620 FORMAT(10F12.5)
  630 FORMAT(' ')
C
C
      RETURN
      END
```

Function IPROB

```
      FUNCTION IPROB(A,N)
      DIMENSION A(1)
      COMMON /SUNNY/ IYEST,ISEED
      X=RANDOM(ISEED)
      DO 1 I=1,N
      IF (A(I) .GT. X) GO TO 2
  001 CONTINUE
  002 IPROB=I
      RETURN
      END
```

Function RNOR

```
      FUNCTION RNOR(IR)
C  THIS FUNCTION COURTESY DR, REX HURST, DEPT, COMPUTER SCIENCE AND
C  APPLIED STATISTICS, UTAH STATE UNIVERSITY, LOGAN, UTAH 84322
 $SET OWN
      DATA I/0/
 $RESET OWN
      IF (I ,GT, 0) GO TO 30
  010 X=2,*RANDOM(IR)-1,
      Y=2,*RANDOM(IR)-1,
      S=X*X+Y*Y
      IF (S ,GE, 1,) GO TO 10
      S=SQRT(-2,*ALOG(S)/S)
      RNOR=X*S
 $SET OWN
      GO2=Y*S
 $RESET OWN
      I=1
      GO TO 40
  030 RNOR=GO2
      I=0
  040 RETURN
      END
```

LITERATURE CITED

Griffin, R. A., R. J. Hanks, and S. Childs. 1974. Model for estimating water, salt, and temperature distribution in the soil profile. US/IBP Desert Biome Res. Memo. 74-61. Utah State Univ., Logan. 12 pp.

B. HEAT

P. W. Lommen

GENERAL DESCRIPTION OF THE HEAT SUBMODEL

This submodel determines average temperature during a time-step for each soil layer. Soil temperature information is needed by VEG in determining root respiration rates, by DCMP in determining decomposition rates and by WATER in checking if the soil is frozen. Temperature determinations more frequent than once each time-step (generally 4 days) would be more accurate but, given the overall model objectives, not likely to be more useful.

The approach used is very similar to that of Hanks et al. (1971). The soil surface temperature is taken to be the mean daily air temperature at 2 m for the time-step (determined in the weather submodel). The temperature at the bottom of the profile (60-cm depth was used for the Curlew Valley simulation) is set equal to the mean air temperature for the previous 30 days (adapted from the approach of Beckman et al. 1973). Once these boundary conditions are established the temperature profile is determined by the solution of Equation B-1:

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial z} \left(\sigma \frac{\partial T}{\partial z}\right) , \qquad \text{(B-1)}$$

where

T = the soil temperature at a given point at a given instant of time, °C;

t = time, day;

z = depth in soil, cm;

σ = soil thermal diffusivity (the ratio of thermal conductivity, cal·cm^{-1}·day^{-1}·°C^{-1}, to specific heat, cal·cm^{-3}·°C^{-1}), cm^{2}/day.

This differential equation is converted to a difference equation for computation. A set of nodes is introduced in the soil profile (at Curlew Valley the depths of these eight nodes are 0, 3, 10, 20, 30, 40, 50 and 60 cm) and the Crank-Nicholson iteration scheme (Richtmyer 1957) used to generate N—2 equations with N—2 unknowns. N is the number of soil nodes and the N—2 unknowns are the temperatures at all except top and bottom nodes. A tri-diagonal matrix method of solution (Richtmyer 1957) is then applied to the N—2 equations.

Discussion

Thermal diffusivity is assumed to be independent of soil water content. Because of the boundary conditions and the relatively long time-step used (4 days for Curlew Valley), soil temperatures turn out to be fairly insensitive to the values used for thermal diffusivity. Values for thermal conductivity and specific heat were taken from Hanks et al. (1971).

Curlew Valley Implementation

Thermal conductivity is equal to 86.4 cal·cm^{-1}·day^{-1}·°C^{-1} throughout profile (Griffin et al. 1974). Specific heat, taken from the same source, is 0.3 cal·cm^{-3}·°C^{-1} throughout profile. The set of eight soil nodes at depths of 0, 3, 10, 20, 30, 40, 50 and 60 cm was chosen with a number of factors in mind: 1) almost all roots occur in the first 60 cm; 2) variations occur very slowly at 60 cm; 3) caliche layer depth is about 60 cm (the water submodel uses these same nodes; 4) 10-cm increments are convenient to work with.

PROGRAM DESCRIPTION

Only the important segments of the code in the program listing are shown and described. Sequence numbers are shown to aid in reference to full code listing which follows the program description. All comment cards, specification statements and bookkeeping sections have been left out. Almost all initialization has been deleted also. Definitions of variable names may be found in Table B-1, which also appears at the beginning of the program listing.

Table B-1. Variable dictionary for HEAT

CHD(10)	DEPTH TO SOIL NODE, CM.
CHDX(10)	THICKNESS OF LAYER AROUND NODE, COUNTING FROM 2ND NODE, CM.
CHDXX(10)	DISTANCE BETWEEN ADJACENT NODES, CHDXX(I) = CHD(I+1) - CHD(I), CM.
DBGFLG	IF=.TRUE. WRITE DEBUGGING INFORMATION THIS TIME STEP
DEBUG1	NUMBER OF TIME STEPS BETWEEN DEBUGGING OUTPUTS
DT	=PMDT, LENGTH OF TIME STEP, DAY.
FTAVE(PMDT,N)	FUNCTION TO AVERAGE PREVIOUS N DAYS' AIR TEMPERATURES BY AVERAGING APPROPRIATE NUMBER OF PREVIOUS TIME STEP AVERAGE TEMPERATURES.
PHCV(10)	SPECIFIC HEAT OF SOIL LAYER, (REMEMBER, 1ST LAYER IS CENTERED ON 2ND NODE.) CAL CM-3 C-1
PHK(10)	THERMAL CONDUCTIVITY OF SOIL BETWEEN NODES, (1ST VALUE IS FOR REGION BETWEEN NODES 1 AND 2), CAL CM-1 DAY-1 C-1.
PMDT	LENGTH OF TIME STEP, DAY.
PMJDAT	JULIAN DATE OF BEGINNING OF TIMESTEP.
PMN	NUMBER OF SOIL NODES.
THA(10)	THA(I) IS THE NEGATIVE OF THE COEFFICIENT OF THU(I+1) IN THE ITH EQUATION IN THE SET OF EQUATIONS TO SOLVE FOR THE THU ARRAY.
THB(10)	THB(I) IS THE COEFFICIENT OF THU(I) IN THE ITH EQUATION IN THE SET OF EQUATIONS TO SOLVE FOR THE THU ARRAY.
THC(10)	THC(I) IS THE NEGATIVE OF THE COEFFICIENT OF THU(I-1) IN THE ITH EQUATION IN THE SET OF EQUATIONS TO SOLVE FOR THE THU ARRAY.
THCCC(20)	AN ARRAY USED TO READ AND WRITE COMMENTS.
THD(10)	THD(I) IS THE CONSTANT, RIGHT HAND SIDE OF THE ITH EQUATION IN STHE SET OF EQUATIONS TO SOLVE FOR THE THU ARRAY.
THM	AN INTEGER EQUAL TO THE LARGEST WHOLE NUMBER OF TIMESTEPS IN 30 DAYS.
THMR	A REAL VARIABLE EQUAL TO THM.
THTA(10)	TEMPERATURE OF SOIL NODES AT BEGINNING OF TIMESTEP, DEGREES C.
THTB(10)	TEMPERATURE OF SOIL NODES AT END OF TIMESTEP, DEGREES C.
THTHA(10)	TEMPORARY ARRAY USED IN CALCULATING MATRIX ELEMENTS IN CALL TO TRI-DIAGONAL MATRIX SUBROUTINE.
THTHB(10)	ANOTHER TEMPORARILY USED ARRAY FOR CALCULATING MATRIX ELEMENTS IN CALL TO TRI-DIAGONAL MATRIX SUBROUTINE.
THU(10)	THIS ARRAY CONTAINS THE TEMPERATURES OF THE INNER NODES (2 THROUGH PMN-1) AT THE END OF PMDT. THESE TEMPERATURES ARE THE SOLUTIONS OBTAINED BY THE TRI-DIAGONAL MATRIX SUBROUTINE.
TIME	TIME AT START OF TIMESTEP. EQUALS TSTART PLUS (NUMBER OF TIMESTEPS) * PMDT
TSTART	JULIAN DATE OF BEGINNING OF SIMULATION
XHSOLT(10)	AVERAGE TEMPERATURE OF SOIL NODE DURING PMDT, DEGREES C.
ZAIRT	AVERAGE AIR TEMPERATURE FOR PRESENT TIMESTEP. DETERMINED BY PSWG, DEGREES C.
ZHAIRT(31)	HOLDS THM PRESENT AND PAST TIME STEP AVERAGE AIR TEMPERATURES. ZHAIRT(1) HOLDS PRESENT TIME STEP TEMP. ZHAIRT(2) HOLDS PREVIOUS TIME STEP TEMP, ETC.

```
17 THTB(1) = ZAIRT                                               HEAT 195
```

Assume surface temperature equals average air temperature for time-step.

```
   NTHM=THM+1                                                    HEAT 200
18 ZHAIRT(NTHM)=ZHAIRT(NTHM-1)                                   HEAT 201
   NTHM=NTHM-1                                                   HEAT 202
   IF(NTHM .GE. 2) GO TO 18                                      HEAT 203
   ZHAIRT(1)=ZAIRT                                               HEAT 204
```

Variable NTHM is essentially a DO loop index which decreases by one each time (Burroughs FORTRAN doesn't accept negative DO loop increments). This loop shifts temperature in ZHAIRT, dropping the oldest time-step temperature from ZHAIRT (THM + 1) and loading present temperature in ZHAIRT (1).

```
   THTB(PMN)=FTAVE(PMDT,30)                                      HEAT 208
```

Set the temperature of the bottom node equal to the average temperature of the previous 30 days. Accomplish this using function FTAVE.

```
28 DO 30 I= 1, PMN-2                                             HEAT 223
30 THD(I) = THTA(I) * THTHA(I) + THTA(I+1) *                     HEAT 224
 *   (THTHB(I) - THTHA(I) - THTHA(I+1) ) + THTA(I+2) * THTHA(I+1) HEAT 225
   THD(1) = THD(1) + THTB(1) * THTHA(1)                          HEAT 228
   THD(PMN-2) = THD(PMN-2) + THTB(PMN) * THTHA(PMN-1)            HEAT 229
```

Calculate the right-hand sides of the PMN—2 difference equations. For details see "Derivation of Difference Equations."

```
   CALL TDM(THA, THB, THC, THD, THU, PMN-2)                      HEAT 239
```

TDM is the subroutine which solves PMN—2 equations for PMN—2 unknowns (temperatures at nodes 2 through PMN—1). The equations are of the form:

$$-C_iU_{i-1} + B_iU_i - A_iU_{i+1} = D\,.$$

The *U*'s (or THU's) are the temperatures determined by TDM and returned to HEAT. For more information see the detailed program description of subroutine TDM.

```
   DO 32 I=1,PMN-2                                               HEAT 246
32 THTB(I+1) = THU(I)                                            HEAT 247
```

Load THU array, just determined by TDM, into the appropriate places in the THTB array.

```
   DO 34 I=1,PMN                                                 HEAT 250
34 XHSOLT(I) = (THTA(I) + THTB(I)) / 2.                          HEAT 251
```

Make XHSOLT the average temperature at the soil nodes during PMDT.

```
      ENTRY HINIT                                                       HEAT 274
```

Heat Initialization Section

Initial data are read in and written here. Also, calculations which need to be done only once each run are done here.

```
      READ(5,80)THCCC                                                   HEAT 281
      WRITE(6,81)                                                       HEAT 282
      WRITE(6,82)THCCC                                                  HEAT 283
```

First, a comment card is read (THCC is used only to read and write comments). This card is intended to be a general comment about HEAT, e.g., INITIALIZATION DATA FOR HEAT. Then come four spaces and this comment is written.

```
      READ(5,80)THCCC                                                   HEAT 285
      WRITE(6,82)THCCC                                                  HEAT 286
      READ(5,/)CHD                                                      HEAT 287
      WRITE(6,84)CHD                                                    HEAT 288
```

Another comment card is read and written, saying something about CHD (the depths of the nodes). The last characters on the card are the dimensions of the variable. Then values of the variable CHD, dimensioned 10, are read in and written.

An analogous procedure is then followed for reading in CHDX, PHCV, PHK, XHSOLT and ZHAIRT.

```
      READ(5,80)THCCC                                                   HEAT 317
      WRITE(6,82)THCCC                                                  HEAT 318
```

Finally, a comment is read and written, which says END READING INITIAL VALUES FOR HEAT.

```
      DO 110 I=1,PMN-1                                                  HEAT 328
  110 CHDXX(I)=CHD(I+1)-CHD(I)                                          HEAT 329
```

This array is simply the distances between adjacent soil nodes.

```
      THM=30. / PMDT                                                    HEAT 338
```

THM is an integer equal to the largest whole number of time-steps in 30 days.

```
      DO 140 I=1,PMN-1                                                  HEAT 342
      THTHA(I)=PHK(I) / CHDXX(I)                                        HEAT 343
  140 THTHB(I)=2.*PHCV(I)*CHDX(I) / PMDT                                HEAT 344
      DO 160 I=1,PMN-2                                                  HEAT 347
      THA(I) = THTHA(I+1)                                               HEAT 348
      THB(I) = THTHB(I) + THTHA(I) + THTHA(I+1)                         HEAT 349
  160 THC(I)=THTHA(I)                                                   HEAT 350
      THA(PMN-2)=0.                                                     HEAT 351
      THC(1) = 0.                                                       HEAT 352
```

These calculations load arrays THA, THB and THC for use when the tridiagonal matrix subroutine (TDM) is called. Since there is no water dependence on specific heat or thermal conductivity, these arrays do not change during the run and can be calculated just once. For details see "Derivation of Difference Equations."

COMPLETE PROGRAM LISTING

```
 $SET SEPARATE
 $SET OWN
C
C
      SUBROUTINE HEAT
C
C  SOIL HEAT PROFILE SUBMODEL OF DESERT BIOME WATER RESPONSE MODEL
C   MARCH 1976.      WRITTEN BY:   PAUL W. LOMMEN
C                                  ECOLOGY CENTER, UMC 52
C                                  UTAH STATE UNIVERSITY
C                                  LOGAN, UTAH , 84322
C
C
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
C
C              VARIABLE DICTIONARY FOR HEAT
C
C
C  CHD(10)           DEPTH TO SOIL NODE, CM.
C
C  CHDX(10)          THICKNESS OF LAYER AROUND NODE, COUNTING FROM 2ND
C                    NODE, CM.
C
C  CHDXX(10)         DISTANCE BETWEEN ADJACENT NODES, CHDXX(I) = CHD(I+1)
C                    - CHD(I), CM.
C
C  DBGFLG            IF=.TRUE. WRITE DEBUGGING INFORMATION THIS TIME STEP
C
C  DEBUG1            NUMBER OF TIME STEPS BETWEEN DEBUGGING OUTPUTS
C
C  DT                =PMDT, LENGTH OF TIME STEP, DAY.
C
C  FTAVE(PMDT,N)     FUNCTION TO AVERAGE PREVIOUS N DAYS' AIR TEMPERATURES
C                    BY AVERAGING APPROPRIATE NUMBER OF PREVIOUS TIME
C                    STEP AVERAGE TEMPERATURES.
C
C  PHCV(10)          SPECIFIC HEAT OF SOIL LAYER, (REMEMBER, 1ST LAYER
C                    IS CENTERED ON 2ND NODE.) CAL CM-3 C-1
C
C  PHK(10)           THERMAL CONDUCTIVITY OF SOIL BETWEEN NODES,
C                    (1ST VALUE IS FOR REGION BETWEEN NODES 1 AND 2),
C                    CAL CM-1 DAY-1 C-1.
C
C  PMDT              LENGTH OF TIME STEP, DAY.
C
C  PMJDAT            JULIAN DATE OF BEGINNING OF TIMESTEP.
C
C  PMN               NUMBER OF SOIL NODES.
C
C  THA(10)           THA(I) IS THE NEGATIVE OF THE COEFFICIENT OF
C                    THU(I+1) IN THE ITH EQUATION IN THE SET OF
C                    EQUATIONS TO SOLVE FOR THE THU ARRAY.
C
C  THB(10)           THB(I) IS THE COEFFICIENT OF THU(I) IN THE ITH
C                    EQUATION IN THE SET OF EQUATIONS TO SOLVE FOR THE
C                    THU ARRAY.
C
C  THC(10)           THC(I) IS THE NEGATIVE OF THE COEFFICIENT OF
C                    THU(I-1) IN THE ITH EQUATION IN THE SET OF
C                    EQUATIONS TO SOLVE FOR THE THU ARRAY.
C
C  THCCC(20)         AN ARRAY USED TO READ AND WRITE COMMENTS.
C
C  THD(10)           THD(I) IS THE CONSTANT, RIGHT HAND SIDE OF THE ITH
C                    EQUATION IN STHE SET OF EQUATIONS TO SOLVE FOR THE
C                    THU ARRAY.
C
C  THM               AN INTEGER EQUAL TO THE LARGEST WHOLE NUMBER OF
C                    TIMESTEPS IN 30 DAYS.
C
C  THMR              A REAL VARIABLE EQUAL TO THM.
C
C  THTA(10)          TEMPERATURE OF SOIL NODES AT BEGINNING OF TIMESTEP,
C                    DEGREES C.
C
C  THTB(10)          TEMPERATURE OF SOIL NODES AT END OF TIMESTEP,
C                    DEGREES C.
C
C  THTHA(10)         TEMPORARY ARRAY USED IN CALCULATING MATRIX
C                    ELEMENTS IN CALL TO TRI-DIAGONAL MATRIX SUBROUTINE.
C
C  THTHB(10)         ANOTHER TEMPORARILY USED ARRAY FOR CALCULATING
C                    MATRIX ELEMENTS IN CALL TO TRI-DIAGONAL MATRIX
C                    SUBROUTINE.
C
C  THU(10)           THIS ARRAY CONTAINS THE TEMPERATURES OF THE INNER
C                    NODES (2 THROUGH PMN-1) AT THE END OF PMDT. THESE
C                    TEMPERATURES ARE THE SOLUTIONS OBTAINED BY THE
C                    TRI-DIAGONAL MATRIX SUBROUTINE.
```

```
C
C   TIME            TIME AT START OF TIMESTEP.   EQUALS TSTART PLUS
C                   (NUMBER OF TIMESTEPS) * PMDT
C
C   TSTART          JULIAN DATE OF BEGINNING OF SIMULATION
C
C   XHSOLT(10)      AVERAGE TEMPERATURE OF SOIL NODE DURING PMDT,
C                   DEGREES C.
C
C   ZAIRT           AVERAGE AIR TEMPERATURE FOR PRESENT TIMESTEP.
C                   DETERMINED BY PSWG, DEGREES C.
C
C   ZHAIRT(31)      HOLDS THM PRESENT AND PAST TIME STEP AVERAGE AIR
C                   TEMPERATURES.   ZHAIRT(1) HOLDS PRESENT TIME STEP
C                   TEMP.   ZHAIRT(2) HOLDS PREVIOUS TIME STEP TEMP, ETC.
C
C
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
C
C
C
C   BRIEF DESCRIPTION AND MAIN ASSUMPTIONS
C
C   ONE, FOR SOIL SURFACE TEMPERATURE I'LL USE THE MEAN DAILY TEMPERATURE
C   OR HAVE A MEASURED OR SIMULATED TEMPERATURE READ IN.  IF DESIRED AN
C   ENERGY BALANCE APPROACH MIGHT BE BETTER, ALTHOUGH  R. J. HANKS
C   THOUGHT THAT WOULD BE PRETTY HAIRY AND HENCE PROBABLY NOT WORTHWHILE
C   FOR US (REF.  W.A. BECKMAN, J.W. MITCHELL, AND W.P. PORTER, 1973,
C   THERMAL MODEL FOR PREDICTION OF A DESERT IGUANA'S DAILY AND SEASONAL
C   BEHAVIOR, TRANS. ASME, SERIES C 95, 257-262.)
C
C   TWO, FOR THERMAL DIFFUSIVITY I'LL ASSUME IT'S INDEPENDENT OF WATER
C   CONTENT.  BOTH HEAT CONDUCTIVITY AND HEAT CAPACITY HAVE A DEPENDENCE
C   ON WATER CONTENT BUT DIFFUSIVITY, THEIR RATIO HAS ONLY A WEAK WATER
C   DEPENDENCE (R. J. HANKS , PERSONAL COMMUNICATIONS 5-24-74).
C   IF DESIRED WE CAN INCLUDE WATER AS IN  R.J. HANKS, D.D. AUSTIN,
C   AND W.T. ONDRECHEN, 1971, SOIL TEMPERATURE ESTIMATION BY A NUMERICAL
C   METHOD.   SOIL SCI. SOC. AM. PROC. 35, 665.  THEY GIVE A
C   SIMPLE RELATIONSHIP BETWEEN HEAT CAPACITY AND WATER CONTENT.
C   FOR HEAT CONDUCTIVITY THEY USE METHOD OF DE VRIES, D.A., 1963,
C   THERMAL PROPERTIES OF SOILS. CHAPTER 7 IN   PHYSICS OF PLANT
C   ENVIRONMENT, ED. BY VAN WIJK. JOHN WILEY AND SONS, INC., NEW YORK.
C
C   THREE, FOR THE BOTTOM BOUNDARY CONDITION, I'LL SAY FOR NOW THAT THE
C   SOIL TEMPERATURE AT 60 CM  EQUALS THE MEAN MONTHLY AIR TEMPERATURE AT
C   2M (MITCHELL ET AL.,  OP. CIT.,  AND DE VRIES, OP. CIT.).
C
C   THE PROGRAM IS TAKEN PRETTY MUCH FROM    R.J. HANKS, D.D. AUSTIN,
C   AND W.T. ONDRECHEN, 1971, SOIL TEMPERATURE ESTIMATION BY A NUMERICAL
C   METHOD.   SOIL SCI. SOC. AM. PROC. 35, 665.
C   OCCASIONAL INSPIRATION IS TAKEN FROM   HANKS, R.J., AND S.A.
C   BOWERS. 1962. NUMERICAL SOLUTION OF THE MOISTURE FLOW EQUATION FOR
C   INTILTRATION IN LAYERED SOILS.  SOIL SCI. SOC. AMER. PROC. 26,530-534
C   THE FINAL PRODUCT IS CONCEPTUALLY ALMOST THE SAME AS THE TEMPERATURE
C   SECTION OF  R.A. GRIFFIN, R.J. HANKS, AND S. CHILDS, 1973, MODEL
C   FOR ESTIMATING WATER, SALT, AND TEMPERATURE DISTRIBUTION IN THE SOIL
C   PROFILE, US/IBP DESERT BIOME PROGRAM PUBLICATION  USU, LOGAN, UTAH.
C
C   THIS ROUTINE MUST BE CALLED AFTER THE WATER SUBROUTINE IF A THETA
C   DEPENDENCE ON DIFFUSIVITY IS TO BE INCLUDED.
C   THIS DEPENDENCE IS NOT NOW INCLUDED
C
C   FOR NOW THE TEMPERATURE OF THE WATER MOVING BETWEEN LAYERS
C   IS NOT CONSIDERED IN THE TEMPERATURE OF THE LAYER.  NO VAPORIZATION
C   OR CONDENSATION EFFECTS ARE INCLUDED EITHER.
C
C
C
      COMMON TIME,TSTART,TEND,DT,DTPR,DTPL,
     1CAAR(6,8),CADEST(6,8),CADETH,CAUWST,CHD(10),CHDX(10),CHDXX(10),
     2CNFIX,CNSDF,CNSDS,CNSOMF,CNSOMS,CNSUP,CVDETH(12,4),CVLTFR(11,4),
     3CVPHEN(6,8),CVPHS(6),CVRDST(6,10),CVTSPR(6,8),CWINF,CWPSI(10),
     4PMDT,PMDTPL,PMDTPR,PMFGPS,PMJDAT,PMN,PMNCOM,PMNSP,
     5XAA(4),XAAVWT(4,8),XAAWT(4),XAFTS(4),XAFWT(4),XANUMB(4,8),
     6XASA(4),XASAWT(4),XAYMG(4),XAYWT(4),XHSOLT(10),XMMN,XNSOMF,XNSOMS,
     7XPABS,XPLBP,XPP04,XVFG(6,5),XVLITR(12,4),XVPLNT(6,8),XVTOTL,
     8XWTHTA(10),XWSTND,ZAIRT,ZEVAP,ZESUM,ZPHPD,ZRAIN,ZRSUM,ZRH,ZRINT,
     9ZRISUM,ZSUN,ZTMAX,ZTMIN,ZWIND,COOL(12,4)
C
      COMMON/DB/DEBUG1,DEBUG2,DEBUG3
C
C   FT IS IN COMMON WITH FTAVE
      COMMON/FT/ZHATRT(31)
C
      INTEGER PMN, THM
C
      LOGICAL DBGFLG
C
      DIMENSION PHCV(10), PHK(10), THTA(10), THTB(10), THA(10),
```

```
     * THB(10), THC(10), THD(10), THU(10), THTHA(10), THTHB(10)
     * ,THCCC(20)
C
C
CIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
C  INITIALIZE TEMPERATURES
C
C  SET TEMPERATURES AT BEGINNING OF THIS TIMESTEP EQUAL TO TEMPS
C  AT END OF LAST TIMESTEP.
   14 DO 16 I = 1, PMN
   16 THTA(I) = THTB(I)
C
C  ASSUME SURFACE TEMPERATURE EQUALS AVERAGE AIR
C  TEMPERATURE FOR TIMESTEP.
   17 THTB(1) = ZAIRT
C
C  ZHAIRT IS THE ARRAY WHICH HOLDS THE LAST 30 DAYS OF AIR TEMPS
C  SHIFT TEMPS IN ZHAIRT, LOADING PRESENT AIR TEMP IN ZHAIRT(1) AND
C  DROPPING OLDEST TEMP NOW IN ZHAIRT(THM+1)
      NTHM=THM+1
   18 ZHAIRT(NTHM)=ZHAIRT(NTHM-1)
      NTHM=NTHM-1
      IF(NTHM .GE. 2) GO TO 18
      ZHAIRT(1)=ZAIRT
C
C  TEMPERATURE AT 60 CM IS AVERAGE AIR TEMPERATURE FOR 30 PREVIOUS DAYS.
C  NOW USE AVERAGING FUNCTION TO GET TEMP AT 60 CM
      THTB(PMN)=FTAVE(PMDT,30)
C
C  END INITIALIZATION SECTION
CIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
C
C
C  IF A WATER CONTENT DEPENDENCE ON SPECIFIC HEAT AND THERMAL
C  CONDUCTIVITY IS TO BE INCLUDED SOMEDAY IT CAN GO IN HERE
C  CV WILL HAVE TO BE AN AVERAGE OVER DT
C  FOR K WILL NEED VALUES AT BEGINNING AND END OF DT
C
C
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
C  HERE ARE THE MAIN EQUATIONS
C
   28 DO 30 I= 1, PMN-2
   30 THD(I) = THTA(I) * THTHA(I) + THTA(I+1) *
     *  (THTHB(I) - THTHA(I) - THTHA(I+1) ) + THTA(I+2) * THTHA(I+1)
C
C  NOW THE FIRST AND LAST EQUATIONS NEED TOUCHING UP.
      THD(1) = THD(1) + THTB(1) * THTHA(1)
      THD(PMN-2) = THD(PMN-2) + THTB(PMN) * THTHA(PMN-1)
C
C  END EQUATIONS SECTION
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
C
C
C
CTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTD
C
C  NOW CALL TDM TO GET THOSE PMN-2 TEMPERATURES
      CALL TDM(THA, THB, THC, THD, THU, PMN-2)
C
CTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTDMTD
C
C
C
C  NOW LOAD THTB AND XHSOLT
      DO 32 I=1,PMN-2
   32 THTB(I+1) = THU(I)
C
C  XHSOLT IS THE AVERAGE TEMPERATURE OF THE NODE DURING PMDT
      DO 34 I=1,PMN
   34 XHSOLT(I) = (THTA(I) + THTB(I)) / 2.
C
C
C  DEBUGGING
      DBGFLG=.FALSE.
      IF((DEBUGI .GT. 1.E-5) .AND. ((MOD(((TIME-TSTART)/DT), DEBUGI))
     *    .LT. 1.E-5))  DBGFLG=.TRUE.
      IF(.NOT. DBGFLG)  GO TO 710
  610 WRITE(1,680)PMJDAT, PMDT,XHSOLT
  680 FORMAT(' H      PMJDAT=', I5, '      PMDT=', F5.1, '      XHSOLT='
     * , 10F6.2)
  710 CONTINUE
C
C
      RETURN
C
C
C
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
```

```
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
C
      ENTRY HINIT
C
C  IN THIS SECTION ARE THE READING AND WRITING OF INPUT TO THIS
C  SUBROUTINE AND CALCULATIONS WHICH ARE TO BE DONE ONLY ONCE.
C
C  READ AND WRITE INPUT DATA
C  VARIABLE THCCC IS FOR READING AND WRITING COMMENTS
      READ(5,80)THCCC
      WRITE(6,81)
      WRITE(6,82)THCCC
C
      READ(5,80)THCCC
      WRITE(6,82)THCCC
      READ(5,/)CHD
      WRITE(6,84)CHD
C
      READ(5,80)THCCC
      WRITE(6,82)THCCC
      READ(5,/)CHDX
      WRITE(6,84)CHDX
C
      READ(5,80)THCCC
      WRITE(6,82)THCCC
      READ(5,/)PHCV
      WRITE(6,84)PHCV
C
      READ(5,80)THCCC
      WRITE(6,82)THCCC
      READ(5,/)PHK
      WRITE(6,84)PHK
C
      READ(5,80)THCCC
      WRITE(6,82)THCCC
      READ(5,/)XHSOLT
      WRITE(6,84)XHSOLT
C
C  READ IN ZHAIRT  FOR PREVIOUS 30 DAYS OF AIR TEMPERATURES
C  THESE TEMPS ARE TIME STEP AVERAGES NOT NECESSARILY DAILY TEMPS
      READ(5,80)THCCC
      WRITE(6,82)THCCC
      READ(5,/)ZHAIRT
      WRITE(6,84)ZHAIRT
C
      READ(5,80)THCCC
      WRITE(6,82)THCCC
      WRITE(6,81)
C
   80 FORMAT(20A4)
   81 FORMAT(////)
   82 FORMAT('0',20A4)
   84 FORMAT(' ', 10E12.4,/)
C
C
C  NOW DO THE DELTA-X CALCULATIONS
      DO 110 I=1,PMN-1
  110 CHDXX(I)=CHD(I+1)-CHD(I)
C  END DELTA-X CALCULATIONS
C
C
      DO 120 I=1,PMN
      THTA(I)=XHSOLT(I)
  120 THTB(I)=XHSOLT(I)
C
C  THM IS THE CLOSEST INTEGER TO THE NO. OF TIME STEPS IN 30 DAYS
      THM=30. / PMDT
C
C  SINCE NO THETA DEPENDENCE ON CV OR K INCLUDED, THESE
C  CALCULATIONS HAVE TO BE MADE ONLY ONCE
      DO 140 I=1,PMN-1
      THTHA(I)=PHK(I) / CHDXX(I)
  140 THTHB(I)=2.*PHCV(I)*CHDX(I) / PMDT
C
C  PHCV(PMN-1) AND CHDX(PMN-1) ARE NOT USED SO CAN JUST LOAD ZEROES IN
      DO 160 I=1,PMN-2
      THA(I) = THTHA(I+1)
      THB(I) = THTHB(I) + THTHA(I) + THTHA(I+1)
  160 THC(I)=THTHA(I)
      THA(PMN-2)=0.
      THC(1) = 0.
C  END OF EQUATIONS WHICH CAN BE CALCULATED ONLY ONCE IF NO THETA
C  DEPENDENCE IS INCLUDED IN CV OR K
C
C
C
      RETURN
      END
```

DERIVATION OF DIFFERENCE EQUATIONS

As mentioned in the general description of HEAT, the temperature profile is determined by the solution of Equation B-1:

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial z} (\sigma \frac{\partial T}{\partial z}) . \qquad \text{(B-1)}$$

where

T = the soil temperature at a given point at a given instant of time, °C;
t = time, day;
z = depth in soil, cm;
σ = soil thermal diffusivity (the ratio of thermal conductivity, $\text{cal}\cdot\text{cm}^{-1}\cdot\text{day}^{-1}\cdot{}^{\circ}\text{C}^{-1}$, to specific heat, $\text{cal}\cdot\text{cm}^{-3}\cdot{}^{\circ}\text{C}^{-1}$), cm^2/day.

The difference equations are derived as follows:

for a thin layer of soil during a time interval Δt,

$$\text{(heat in)} - \text{(heat out)} = \text{(heat stored)} . \qquad \text{(B-2)}$$

In particular, for layer i during Δt,

$$\text{(heat in)}_i = -\ (K \frac{\partial T}{\partial z}) \text{ at top of layer } i , \qquad \text{(B-3)}$$

where

counting of layers starts at the soil surface and goes down; heat flowing down is positive:

$(\text{heat in})_i$ in Equation B-3 has units of $\text{cal}\cdot\text{cm}^{-2}\cdot\text{day}^{-1}$;

K = soil thermal conductivity, $\text{cal}\cdot\text{cm}^{-1}\cdot\text{day}^{-1}\cdot{}^{\circ}\text{C}^{-1}$;
T = temperature of layer i, °C;
z = depth in soil, cm.

Flesh out Equation B-3,

$$\text{(heat in)}_i = (- K_{i-1/2}) \left\{ \left[\frac{T_i - T_{i-1}}{\Delta z_{i-1/2}} + \frac{T_i^* - T_{i-1}^*}{\Delta z_{i-1/2}} \right] (\tfrac{1}{2}) \right\} , \qquad \text{(B-4)}$$

where

T_i = temperature of layer i at beginning of Δt, assumed located at the center of a layer, called a node. The first node is at the soil surface at 0-cm depth. The second is at 3-cm depth (at Curlew Valley) and is at the center of layer 1, which extends from 1- to 5-cm depth. Node 3 at Curlew is at 10-cm depth and is at the center of layer 2, which extends from 5- to 15-cm depth, etc. The bottom layer is centered on the next to the bottom node. There are PMN nodes and, hence, PMN—2 layers. These distinctions between nodes, layers, regions between nodes, etc., appear confusing but must be made in using the difference equations;
T_{i-1} = temperature of layer i—1 at beginning of Δt;
T_i^* = temperature of layer 1 at end of Δt;
T_{i-1}^* = temperature of layer i—1 at end of Δt;
$K_{i-1/2}$ = conductivity in region between centers of layers i—1 and i (i.e., between nodes i and $i + 1$);
$\Delta z_{i-1/2}$ = distance from center of layer i—1 to center of layer i (i.e., from node i to $i + 1$).

The quantity in braces is the temperature gradient determined by the Crank-Nicholson scheme (Richtmyer 1957) and is the average of the temperature gradients at the beginning and end of Δt.

By similar reasoning:

$$\text{(heat out)}_i = - (K \frac{\partial T}{\partial z}) \text{ at bottom of layer } i,$$

$$= - (K_{i+1/2}) \left\{ \left[\frac{T_{i+1} - T_i}{\Delta z_{i+1/2}} + \frac{T_{i+1}^* - T_i^*}{\Delta z_{i+1/2}} \right] (\tfrac{1}{2}) \right\} . \qquad \text{(B-5)}$$

The heat stored in $\text{cal}\cdot\text{cm}^{-2}\cdot\text{day}^{-1}$

$$= \frac{(Cv_i)(\Delta x_i)}{\Delta t} (T_i^* - T_i) , \qquad \text{(B-6)}$$

where

Cv_i = specific heat of layer i, $\text{cal}\cdot\text{cm}^{-3}\cdot{}^{\circ}\text{C}^{-1}$;
Δx_i = thickness of layer i, cm;
Δt = length of time-step, day.

By substituting Equations B-4, B-5 and B-6 in Equation B-2 we get:

$$\frac{- K_{i-1/2}}{2} \left[\frac{T_i - T_{i-1} + T_i^* - T_{i-1}^*}{\Delta z_{i-1/2}} \right]$$

$$+ \frac{K_{i+1/2}}{2} \left[\frac{T_{i+1} - T_i + T_{i+1}^* - T_i^*}{\Delta z_{i+1/2}} \right]$$

$$= \frac{Cv_i \ \Delta x_i}{\Delta t} (T_i^* - T_i) . \qquad \text{(B-7)}$$

Multiply through by two and collect T^* terms on the left. After two lines of algebra we get:

$$-C_i\,T^*_{i-1}+B_i\,T^*_i-A_i\,T^*_{i+1}=D_i\ , \qquad \text{(B-8)}$$

where

$$C_i=\frac{K_{i-1/2}}{\Delta z_{i-1/2}}\ , \qquad \text{(B-9)}$$

$$B_i=\frac{K_{i-1/2}}{\Delta t_{i-1/2}}+\frac{K_{i+1/2}}{\Delta z_{i+1/2}}+\frac{2Cv_i\ \Delta x_i}{\Delta t}\ , \qquad \text{(B-10)}$$

$$A_i=\frac{K_{i+1/2}}{\Delta z_{i+1/2}}\ , \qquad \text{(B-11)}$$

$$D_i=T_{i-1}\left(\frac{K_{i-1/2}}{\Delta z_{i-1/2}}\right)+T_i\left[\frac{2Cv_i\ \Delta x_i}{\Delta t}-\frac{K_{i-1/2}}{\Delta z_{i-1/2}}-\frac{K_{i+1/2}}{\Delta z_{i+1/2}}\right]+T_{i+1}\left(\frac{K_{i+1/2}}{\Delta z_{i+1/2}}\right). \qquad \text{(B-12)}$$

For top and bottom layers we must make small changes in Equations B-8 to 12 because boundary condition must be included. The temperature at the surface at the end of the time-step is set equal to the average air temperature during Δt:

$$T^*_s=\text{ZAIRT}\ . \qquad \text{(B-13)}$$

The temperature at the bottom node $T_B{}^*$ (there are PMN nodes) at the end of Δt is set equal to the average air temperature during the previous 30 days.

For the top layer then, $(\text{heat in})_1$ contains only one unknown temperature, not two as shown in Equation B-4.

That is

$$(\text{heat in})_1=-(K_{1/2})\left[\frac{T_1-T_s}{\Delta z_{1/2}}-\frac{T^*_1-T^*_s}{\Delta z_{1/2}}\right](\tfrac{1}{2}). \qquad \text{(B-14)}$$

In Equation B-14, only $T_1{}^*$, the temperature of layer 1 (i.e., node 2) is unknown. Equations B-5 and 6 for the top layer with $i=1$ are unchanged. Substituting Equations B-14, 5 and 6 in Equation B-2 now yields by the same procedure as before:

$$B_1\,T^*_1-A_1\,T^*_2=D_1 \qquad \text{(B-15)}$$

where

$$B_1=\frac{K_{1/2}}{\Delta z_{1/2}}+\frac{K_{3/2}}{\Delta z_{3/2}}+\frac{2Cv_1\ \Delta x_1}{\Delta t}\ , \qquad \text{(B-16)}$$

$$A_1=\frac{K_{3/2}}{\Delta z_{3/2}}\ , \qquad \text{(B-17)}$$

$$D_1=T_s\,\frac{K_{1/2}}{\Delta z_{1/2}}+T_1\left[\frac{2Cv_1\ \Delta x_1}{\Delta t}-\frac{K_{1/2}}{\Delta z_{1/2}}-\frac{K_{3/2}}{\Delta z_{3/2}}\right]+T_2\,\frac{K_{3/2}}{\Delta z_{3/2}}+T^*_s\,\frac{K_{1/2}}{\Delta z_{1/2}}\ . \qquad \text{(B-18)}$$

For the bottom layer, $(\text{heat out})_{\text{bottom}}$ contains only one unknown temperature. After several lines of algebra we then get (say the bottom layer is layer number M):

$$-C_m\,T^*_{m-1}+B_m\,T^*_m=D_m\ , \qquad \text{(B-19)}$$

where

$$C_m=\frac{K_{m-1/2}}{\Delta z_{m-1/2}}\ , \qquad \text{(B-20)}$$

$$B_m=\frac{K_{m-1/2}}{\Delta z_{m-1/2}}+\frac{K_{m+1/2}}{\Delta z_{m+1/2}}+\frac{2C_{vm}\ \Delta x_m}{\Delta t}\ , \qquad \text{(B-21)}$$

$$D_m=T_{m-1}\left(\frac{K_{m-1/2}}{\Delta z_{m-1/2}}\right)+T_m\left[\frac{2C_{vm}\ \Delta x_m}{\Delta t}-\frac{K_{m-1/2}}{\Delta z_{m-1/2}}-\frac{K_{m+1/2}}{\Delta z_{m+1/2}}\right]+T_B\left(\frac{K_{m+1/2}}{\Delta z_{m+1/2}}\right)+T^*_B\left(\frac{K_{m+1/2}}{\Delta z_{m+1/2}}\right)\ . \qquad \text{(B-22)}$$

Now summarize these equations for A, B, C and D.

$$A_i=\frac{K_{i+1/2}}{\Delta z_{i+1/2}} \quad \text{for} \quad i=1 \text{ to PMN}-3,$$

$$=0 \quad \text{for} \quad i=\text{PMN}-2 \text{ (bottom layer)}, \qquad \text{(B-23)}$$

$$B_i=\frac{K_{i-1/2}}{\Delta z_{i-1/2}}+\frac{K_{i+1/2}}{\Delta z_{i+1/2}}+\frac{2C_{vi}\ \Delta x_i}{\Delta t}$$

$$\text{for} \quad i=1 \text{ to PMN}-2, \qquad \text{(B-24)}$$

$$C_i=0 \quad \text{for} \quad i=1,$$

$$=\frac{K_{i-1/2}}{\Delta z_{i-1/2}} \quad \text{for} \quad i=2 \text{ to PMN}-2, \qquad \text{(B-25)}$$

$$D_i = T_s \frac{K_{1/2}}{\Delta z_{1/2}} + T_1 \left[\frac{2C_{v1}\ \Delta x_1}{\Delta t} - \frac{K_{1/2}}{\Delta z_{1/2}} - \frac{K_{3/2}}{\Delta z_{3/2}} \right]$$

$$+ T_2 \frac{K_{3/2}}{\Delta z_{3/2}} + T_s^* \frac{K_{1/2}}{\Delta z_{1/2}}$$

$$\text{for} \quad i = 1,$$

$$= T_{i-1} \frac{K_{i-1/2}}{\Delta z_{i-1/2}} + T_i \left[\frac{2C_{vi}\ \Delta x_i}{\Delta t} - \frac{K_{i-1/2}}{\Delta z_{i-1/2}} \right.$$

$$\left. - \frac{K_{i+1/2}}{\Delta z_{i+1/2}} \right] + T_{i+1} \frac{K_{i+1/2}}{\Delta z_{i+1/2}}$$

$$\text{for} \quad i = 2 \text{ to PMN} - 3,$$

$$= T_{i-1} \frac{K_{i-1/2}}{\Delta z_{i-1/2}} + T_i \left[\frac{2C_{vi}\ \Delta x_i}{\Delta t} - \frac{K_{i-1/2}}{\Delta z_{i-1/2}} \right.$$

$$\left. - \frac{K_{i+1/2}}{\Delta z_{i+1/2}} \right] + T_B \frac{K_{i+1/2}}{\Delta z_{i+1/2}} + T_B^* \frac{K_{i+1/2}}{\Delta z_{i+1/2}}$$

$$\text{for} \quad i = \text{PMN} - 2. \tag{B-26}$$

It can easily be seen in Equations B-23, 24 and 25 that there are no temperatures involved. Also, the conductivity, *K*, and specific heat, C_v, are not dependent on water content or temperature. (A water content dependence can be included if it is felt such detail is warranted. See Hanks et al. 1971). Thus, the *A*'s, *B*'s and *C*'s need to be calculated only once. The *D*'s must be calculated each time-step.

Transform Equations B-23, 24, 25 and 26 into computer code:

a. Thermal conductivity	$K_{1/2}$ becomes PHK(1) $K_{3/2}$ becomes PHK(2) . . . $K_{PMN-2+1/2}$ becomes PHK(PMN—1)
b. Specific heat	Cv_i becomes PHCV(I) for I = 1 to PMN—2
c. Distance from node to node	$\Delta z_{1/2}$ becomes CHDXX(1) . . . $\Delta z_{PMN-2+1/2}$ becomes CHDXX(PMN—1)
d. Thickness of layer	Δx_i becomes CHDX(I) for I = 1 to PMN—2
e. Temperatures at beginning of time-step	T_i becomes THTA(I+1) for I = 1 to PMN—2 T_s becomes THTA(1) T_B becomes THTA(PMN)
f. Temperatures at end of time-step	T_s^* becomes THTB(1) T_B^* becomes THTB(PMN)
g. Time-step length	Δt becomes PMDT
h. Tridiagonal matrix elements	A_i, B_i, C_i, D_i become THA(I), THB(I), THC(I), THD(I) for I = 1, PMN—2.

Now, if we define temporary variables THTHA(I) = PHK(I)/CHDXX(I) and THTHB(I) = 2. * PHCV(I) * CHDX(I)/PMDT, then Equation B-23 becomes

```
THA(I) = THTHA(I+1)      for I = 1 to PMN — 3,

       = 0               for I = PMN — 2,
```

Equation B-24 becomes

```
THC(I) = 0               for I = 1,

       = THTHA(I)        for I = 2 to PMN — 2,
```

Equation B-25 becomes

```
THB(I) = THTHA(I)+THTHA(I+1)+THTHB(I)

                         for I = 1 to PMN — 2,
```

Equation B-26 becomes

```
THD(I) = THTA(1)*THTHA(1)+THTA(2)*(THTHB(1)—THTHA(1)

         —THTHA(2))+THTA(3)*THTHA(2)+THTB(1)*THTHA(1)

                         for I = 1,

       = THTA(I)*THTHA(I)+THTA(I+1)*(THTHB(I)—THTHA(I)

         —THTHA(I+1))+THTA(I+2)*THTHA(I+1)

                         for I = 2 to PMN — 3,

       = THTA(PMN—2)*THTHA(PMN—2)+THTA(PMN—1)

         *(THTHB(PMN—2)—THTHA(PMN—2)—THTHA(PMN—1))

         +THTA(PMN)*THTHA(PMN—1)+THTB(PMN)*THTHA(PMN—1)

                         for I = PMN — 2.
```

As mentioned earlier, THA, THB and THC have no water content or temperature dependence and hence are calculated only once, in the initialization section. THD is dependent on temperature and hence is calculated each time-step.

LITERATURE CITED

BECKMAN, W. A., J. W. MITCHELL, and W. P. PORTER. 1973. Thermal model for prediction of a desert iguana's daily and seasonal behavior. Trans. ASME, Series C 95: 257-262.

GRIFFIN, R. A., R. J. HANKS, and S. CHILDS. 1974. Model for estimating water, salt and temperature distribution in the soil profile. US/IBP Desert Biome Res. Memo. 74-61. Utah State Univ., Logan. 12 pp.

HANKS, R. J., D. D. AUSTIN, and W. T. ONDRECHEN. 1971. Soil temperature estimation by a numerical method. Soil Sci. Soc. Amer. Proc. 35:665-667.

RICHTMYER, R. D. 1957. Difference methods for initial value problems. Interscience Publ., New York.

C. WATER
(including WBAL and WTIME)
P. W. Lommen

GENERAL DESCRIPTION OF THE WATER SUBMODEL

In the Water Response Model, soil water potential is needed by the plant submodels in order to predict growth. This is provided by submodel WATER. Soil water potential also helps determine rates in nitrogen and decomposition submodels. Inputs required each time-step are precipitation, evaporation and transpiration. Soil parameters needed once are: 1) relation between soil water potential and soil water content (theta); 2) relation between hydraulic conductivity and theta; 3) wet and dry limits to soil water potential; 4) conductivity of caliche layer; 5) soil layer thicknesses and locations; 6) average runon:runoff ratio per day; 7) largest permissible value of change in theta during water time-step (which is generally much smaller than overall model time-step); 8) initial values of theta and depth of water standing on surface.

Various papers (Hanks and Bowers 1962; Hanks et al. 1969; Nimah and Hanks 1973; Griffin et al. 1974) have provided ideas which have contributed to this effort.

A set of nodes is introduced in the soil profile (the same nodes as for submodel HEAT) as reference points for soil layers. Figure C-1 is a schematic diagram of submodel WATER. The soil water potential in soil layer i is the solution of Equation C-1:

$$C_i \left[\frac{\partial H}{\partial t} \right] = \left[\frac{\partial}{\partial z} \left(K \frac{\partial H}{\partial z} \right) \right] + A_i . \qquad \text{(C-1)}$$

A_i is the removal of water by roots in layer i (transpiration) and is proportional to water available in layer i and relative root amount in layer i.

The remainder of the equation is a standard diffusion equation: H is the soil water potential in layer i, bars; K is the soil hydraulic conductivity at layer i, $cm^2 \cdot bar^{-1} \cdot day^{-1}$; C_i is the specific water capacity of layer i, bar^{-1}; Z is soil depth, centimeters; t is time, days.

The upper boundary condition on Equation C-1 is determined by the surface flux of water which depends on evaporation, precipitation and precipitation intensity, and standing water. The bottom boundary condition is determined by the caliche layer conductivity and water content, both of which are read in with initial parameters.

A set of difference equations is then generated. It is solved by a tridiagonal matrix method of Richtmyer (1957). For more details, see "Derivation of Difference Equations" below, and Detailed Program Description of subroutine TDM in section D of this research memorandum.

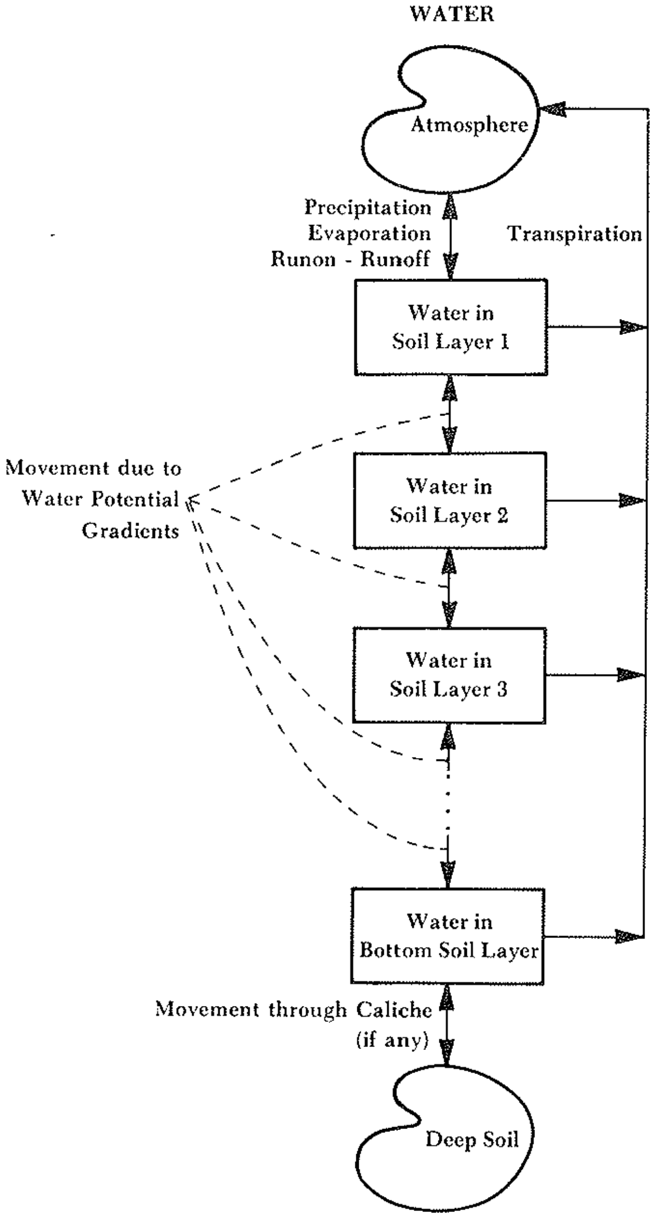

Figure C-1. Schematic diagram of submodel WATER.

PROGRAM DESCRIPTION

Only the important segments of the FORTRAN code are shown and described. Sequence numbers are shown to aid in reference to the full code listing that follows the program description. All comment cards, specification statements and bookkeeping sections have been deleted also. Definitions of variable names may be found in Table C-1, which also appears at the beginning of the program listing.

Before proceeding to the code, several additional comments should be made. First, volumetric water content of the soil, a dimensionless quantity (cm of water per cm of soil), is a very important and much-used variable throughout the program and will be called *theta* in the description. Similarly, soil water potential, measured in bars pressure, will be abbreviated SWP

Second, water subdivides PMDT, the overall model time-step (typically 4 days), as needed. If water is moving rapidly into the soil profile during a precipitation event, e.g., the WATER time-step, TWDT will be cut down to 30 min (the minimum allowed). WATER tries to find the largest TWDT consistent with the requirement that theta does not change more than PWTLIM (typically .015) during TWDT. We want TWDT large enough to be efficient but small enough to be accurate. There is a fair amount of code concerned with determining the best value of TWDT.

Third, in making difference equations from the main differential equation, it has been necessary to define nodes, layers and regions between nodes for the soil profile. This can be confusing. Figure C-2 shows how this was done for Curlew Valley. There are PMN nodes or reference points, PMN—1 regions between nodes and PMN—2 layers. Also shown in Figure C-2 are the variables which are indexed by layers, nodes and regions.

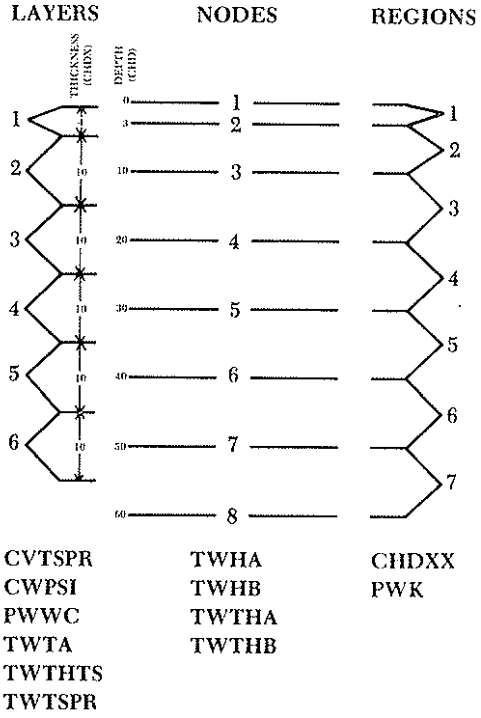

Figure C-2. Relations between layers, nodes and regions between nodes for the soil profile as used in Curlew Valley runs. The first layer is centered on the second node, etc. Also shown are layer width (CHDX) and node depth (CHD), both in centimeters. Finally, arrays indexed by layer are listed under column LAYERS, etc.

Table C-1. Variable dictionary for WATER

LIST OF VARIABLES FOR WATER

CHD(10)	DEPTHS, CM, OF NODES BELOW SURFACE (A NODE IS A POINT WHERE A WATER POTENTIAL IS CALCULATED) STARTING AT SURFACE
CHDX(10)	THICKNESS OF REGION CENTERED ON NODE WHERE COUNTING STARTS AT FIRST NODE BELOW SURFACE AND GOES DOWN
CHDXX(10)	CHDXX(I)=CHD(I+1) - CHD(I)
CVTSPR(10)	MM OF WATER TAKEN FROM EACH LAYER DURING PMDT
CWINF	NET TOP SURFACE FLUX FOR PWDT, CM, DOES NOT INCLUDE TRANSPIRATION.
CWPSI(10)	SOIL WATER POTENTIAL, BARS, VALUE AT END OF PMDT
DBGFLG	IF=.TRUE. WRITE DEBUGGING INFORMATION THIS TIME STEP
DEBUG1	NUMBER OF TIME STEPS BETWEEN DEBUGGING OUTPUTS
FLGFLG	IF=.TRUE. WRITE DEBUGGING INFORMATION THIS TWDT
PMDT	TIME STEP DETERMINED IN MAIN, DAYS
PMN	NUMBER OF NODES, SAME AS NUMBER OF NODES IN HEAT, COUNTING STARTS AT SURFACE
PWDELW	INCREMENT IN VWC, TYPICALLY 0.01-0.02
PWH(60)	TABLE OF HYDRAULIC PRESSURE HEAD VERSUS THETA, BARS, USING SAME THETA SCALE AND SPACING AS PWKIN
PWHDRY	ALLOWABLE LOW PRESSURE LIMIT FOR ANY LAYER
PWHWET	ALLOWABLE HIGH PRESSURE LIMIT FOR ANY LAYER
PWK(10)	HYDRAULIC CONDUCTIVITY, CM2 BAR-1 DAY-1, PWK(I) IS CONDUCTIVITY AVERAGED OVER TWDT, IN REGION BETWEEN NODES I AND I+1, WHERE COUNTING STARTS AT SURFACE.
PWKCAL	CONDUCTIVITY OF CALICHE LAYER
PWKIN(60)	TABLE OF CONDUCTIVITY VERSUS THETA, CM2 BAR-1 DAY-1, STARTING WITH VALUE AT THETA = 0.0, READ IN IN WINIT.
PWKSUM(60)	ARRAY USED TO CALCULATE AVERAGE K, (CM-BARS)/DAY
PWLLIM	LOWER LIMIT OF THETA, DIMENSIONLESS CALCULATED IN WINIT AND = SUM OF CONDUCTIVITY X DELTA H
PWM	NUMBER OF ENTRIES IN CONDUCTIVITY AND PRESURE TABLES
PWRNRF	RUNON TO RUNOFF RATIO PER DAY
PWT(60)	ARRAY CONTAINING UNIFORMLY DISTRIBUTED VALUES OF THETA
PWTLIM	LARGEST CHANGE IN THETA ALLOWED FOR NODES 2 TO PMN-1 DURING ANY TIME STEP TWDT. TYPICAL VALUE .01 TO .02
PWWC(10)	WATER CAPACITY, BAR-1, OF REGIONS SURROUNDING NODES, COUNTING STARTS AT FIRST NODE BELOW SURFACE
R	A VERY TEMPORARY VARIABLE - USED IN REFINED TWDT ESTIMATE, TRANSPIRATION SECTION, REDUCTION OF TWDT IF DELTA THETA TOO LARGE, AND AS RATIO OF WATER IN TO WATER STORED AS CALCULATED IN WBAL.
S	A VERY TEMPORARY VARIABLE - USED IN REFINED TWDT ESTIMATE, TRANSPIRATION SECTION, CONDUCTIVITY SECTION, AND REDUCTION OF TWDT IF DELTA THETA TOO LARGE.
T	A VERY TEMPORARY VARIABLE - USED IN REFINED TWDT ESTIMATE.
TDM	A SUBROUTINE USED BY WATER AND HEAT TO SOLVE A SET OF SIMULTANEOUS DIFFERENCE EQUATIONS, THE LEFT HAND SIDES OF WHICH FORM A TRI-DIAGONAL MATRIX.
TIME	CURRENT JULIAN DATE - BOOKKEEPING DONE IN MAIN PROGRAM.
TSTART	JULIAN DATE OF START OF RUN - INITIALIZED IN MAIN PROGRAM
TWAA	FRACTION OF PWDELW INTERVAL, USED IN INTERPOLATING
TWA(10)	PARAMETER A FOR TRI-DIAGONAL MATRIX ROUTINE (TDM ROUTINE)
TWAAA(10)	ARRAY CONTAINING VALUES OF TWTHA(I) FOR I=2,PMN-1

Table C-1, continued

TWB(10)	PARAMETER B FOR TDM ROUTINE
TWBE(10)	CONTAINS ORDERED VALUES OF TWAAA.
TW	A VERY TEMPORARY VARIABLE - USED IN SURFACE FLUX CALCULATION, TRANSPIRATION SECTION, AND CONDUCTIVITY SECTION.
TWC(10)	PARAMETER C FOR TDM ROUTINE
TWCCC	USED TO READ AND WRITE COMMENTS IN INPUT DATA
TWD(10)	PARAMETER D FOR TDM ROUTINE
TWDT	ACTUAL TIME STEP WATER IS USING
TWDTHP(10)	CHANGE IN THETA DURING LAST TWDT
TWDTP	LENGTH OF PREVIOUS WATER TIME STEP
TWDTQ	HOLDS CURRENT VALUE OF TWDT AS IT IS BEING REDUCED BECAUSE DELTA THETA IS TOO LARGE.
TWEVAP	EVAPORATIVE DEMAND, MM, WATER TO EVAPORATE FROM SURFACE DURING REMAINDER OF PWDT
TWFRZN	EQUALS .TRUE. IF SOIL PROFILE FROZEN, EQUALS .FALSE. IF NOT FROZEN.
TWI	TEMPORARY VARIABLE USED TO SET UP PWT ARRAY.
TWHA(10)	PRESSURE HEAD AT BEGINNING OF TWDT, BARS
TWHB(10)	PRESSURE HEAD AT END OF TWDT, BARS
TWHHA	INTERPOLATED VALUES OF PWH
TWIN	EQUALS .TRUE. IF SURFACE PRESSURE IS WITHIN LIMITS
TWJA	INTEGER USED IN PICKING VALUES OF K, H OFF TABLES
TWJB	INTEGER USED IN PICKING VALUES OF K, H OFF TABLES
TWJH	INTEGER COUNTER USED TO LIMIT NUMBER OF TWDT HALVINGS
TWKSA	INTERPOLATED VALUE OF PWKSUM
TWONE	EQUALS .TRUE. IF FIRST DELTA THETA APPROXIMATION HAS ALREADY BEEN DONE DURING CURRENT TWDT.
TWR	RAIN, MM, WATER TO INFILTRATE, LEAVE STANDING OR RUNOFF DURING REMAINDER OF PWDT.
TWRAIN	EQUALS .TRUE. IF RAIN OCCURS DURING TWDT
TWRP	EQUALS .TRUE. IF TWRAIN WAS TRUE LAST TWDT.
TWSF	SURFACE FLUX OF WATER OR VAPOR,CM
TWSFB	FLUX OF WATER MOVING INTO PROFILE DURING TWDT THROUGH CALICHE, CM.
TWSFBT	FLUX OF WATER MOVING INTO PROFILE DURING PWDT THROUGH CALICHE, CM.
TWSFC	CALCULATED SURFACE FLUX USING TWHB(1) AND PWK(1).
TWSFD	EQUALS TWSF-TWSFC
TWSTND	STANDING WATER, MM, AFTER RUNON-RUNOFF ASSUMPTIONS HAVE BEEN CONSIDERED.
TWSTRD	CHANGE DURING TIME STEP IN WATER STORED IN LAYERS OF PROFILE, CM. (SHOULD EQUAL TWWIN)
TWTA(10)	WATER AVAILABLE IN LAYERS, DIMENSIONLESS
TWTHAA(10)	VOLUMETRIC WATER CONTENT AT BEGINNING OF PWDT
TWTHA(10)	VOLUMETRIC WATER CONTENT (VWC) AT BEGINNING OF TWDT
TWTHB(10)	VWC AT END OF TWDT
TWTHC	THETA AVERAGED OVER TWDT FOR WHATEVER LAYER WE'RE CALCULATING AT THE MOMENT

Table C-1, continued

TWTK	PROPORTIONALITY CONSTANT WHICH IS CALCULATED TO MAKE TRANSPIRATION DEMAND FROM A LAYER PROPORTIONAL TO WATER AVAILABLE AND RELATIVE ROOT AMOUNTS IN A LAYER.
TWTHTS(10)	RUNNING SUM OVER TWDT'S OF TRANSPIRATION BY LAYER, CM
TWTSPR(10)	TRANSPIRATION LOSS OF WATER FROM REGION SURROUNDING NODE, CM, (TOP AND BOTTOM NODES EXCLUDED) DURING TWDT
TWTTOT	TOTAL TIME THAT HAS ELAPSED IN PWDT. DOES NOT INCLUDE CURRENT TWDT, DAYS
TWTW(6)	TRANSPIRATIONAL DEMAND BY FUNCTIONAL GROUP, CM
TWU(10)	WATER POTENTIALS IN BARS OF SOIL LAYERS AS RETURNED FROM SUBROUTINE TDM.
TWWA(10)	USED IN SETTING UP CALL TO TDM
TWWB(10)	USED IN SETTING UP CALL TO TDM
TWWIN	FLUX OF WATER MOVING IN TO PROFILE THIS TIME STEP, EQUALS FLUX IN AT TOP + FLUX IN FROM BOTTOM - TRANSPIRATION, CM. (SHOULD EQUAL TWSTRD)
U	A VERY TEMPORARY VARIABLE-USED IN REFINED TWDT ESTIMATE
WBAL	SUBROUTINE WHICH CHECKS OVERALL ACCURACY OF CALCULATIONS DONE IN WATER - DETERMINES WATER INTO OR OUT OF PROFILE AND COMPARES IT TO CHANGE IN WATER STORED IN PROFILE.
WTIME	FUNCTION WHICH RETURNS ELAPSED CPU TIME IN SECONDS FROM START OF RUN.
XHSOLT(10)	SOIL TEMPERATURE PROFILE, C. INDEXED BY NODES, SO XHSOLT(2), E.G., IS TEMPERATURE OF FIRST LAYER.
XWSTND	STATE VARIABLE CALCULATED BY WATER EQUAL TO AMOUNT OF WATER STANDING ON SOIL SURFACE, MM.
XWTHTA(10)	THETA OF LAYER AT END OF PWDT
Y	A VERY TEMPORARY VARIABLE - USED ONLY TO HOLD VALUE OF FUNCTION WTIME.
Z	A DUMMY USED AS AN ARGUMENT IN WTIME
ZEVAP	MM WATER POTENTIALLY EVAPORATED FROM SOIL SURFACE IN PWDT VALUE OBTAINED FROM SUBROUTINE PSWG
ZRAIN	RAINFALL, MM IN PWDT
ZRINT	RAINFALL INTENSITY, MM/HR

Subroutine WATER

```
      IF(XHSOLT(2) .LE. -1) TWFRZN=.TRUE.
```

WATR 290

If temperature of the top soil layer is less than or equal to —1 C, the entire soil profile is considered frozen.

```
      00 50 I=1,6
      DO50J=1,PHN-2
   50 TWTW(I)=TWTW(I)+1.E-5*CVTSPR(I,J)
```

WATR 319
WATR 320
WATR 322

Determine TWTW(I), transpiration demand in centimeters for plant functional group I, for I = 1 to 6.

```
      IF(.NOT. TWFRZN) GO TO 70
      TWRAIN=.FALSE.
      TWEVAP=0.0
      TWRP=.FALSE.
      GOTO130
```

WATR 334
WATR 335
WATR 336
WATR 337
WATR 338

If soil is frozen, set evaporation equal to zero, set flags indicating there is no precipitation this TWDT, nor was there any last TWDT, and go to section refining TWDT estimate.

```
   70 IF((TWR+TWSTND).LE.4.)GOTO80
      TWRAIN = .TRUE.
      TWDT=0.02083
      IF(TWRP)GOTO90
      GO TO 120
   80 TWRAIN=.FALSE.
   90 TWDT=TWDTP
```

WATR 339
WATR 340
WATR 341
WATR 342
WATR 343
WATR 344
WATR 345

If the amount of water remaining on the surface to be infiltrated is less than or equal to 4 mm, set TWRAIN = FALSE (not raining this TWDT), set TWDT = TWDTP (previous value) and go on to section refining TWDT estimate. If water amount is greater than 4 mm (no rain this TWDT), set TWDT equal to .02083 (= 1/48 day = 30 min = minimum value) and check if rain during last TWDT. If yes, set TWDT = TWDTP and go on to section refining TWDT estimate. If no, keep TWDT at minimum value and skip section refining TWDT estimate.

```
      R=0.3*PWTLIM
      S=0.9*PWTLIM
      T=0.0
      DO110I=2,PHN-1
      U=ABS(TWDTHP(I))
      IF(U.LT.R)GOTO110
      IF(U.GT.S)GOTO120
      IF(U.GT.T)T=U
  110 CONTINUE
      IF(T.LT.R)T=R
      IF(T.LE.0.0)GOTO120
      TWDT=0.9*TWDT*(PWTLIM/T)
```

WATR 352
WATR 353
WATR 354
WATR 355
WATR 356
WATR 357
WATR 358
WATR 359
WATR 360
WATR 361
WATR 362
WATR 363

The only circumstances under which we do not reach this point are if TWRAIN = .TRUE. and TWRP = .FALSE. (The most common example of this would be that the surface was dry last TWDT but rain has occurred since.) The point of this section is to increase TWDT as much as possible within the constraint that theta does not change more than PWTLIM during TWDT. R, S and T are defined as shown. Each pass through DO loop (one pass per soil layer), U is set equal to magnitude of change in theta during previous TWDT. If U is ever found to be greater than 0.9 * PWTLIM, remainder of this section is skipped (TWDT retains present value). At the completion of the loop plus the next statement, T is the larger of: 1) 0.3 * PWTLIM and 2) the largest value of U (maximum value 0.9 * PWTLIM). After a precautionary check for zero value of T, TWDT is increased by the factor PWTLIM/T. The factor 0.9 is a precautionary measure so TWDT isn't increased too much because if TWDT is set too large here, a later check may reduce it.

```
120 IF(TWDT .GT. (PMDT-TWTTOT)) TWDT=PMDT-TWTTOT          WATR 365
```

Check the sum of TWDT's to be sure it isn't greater than PMDT.

```
130 CONTINUE                                              WATR 375
```

Control returns to this point if TWDT is reduced in the line below.

```
    IF(TWFRZN) GO TO 150                                  WATR 380
```

If soil is frozen, skip potential surface flux calculation.

```
    TW=AMIN1(TWR,(ZRINT*TWDT*24.))                        WATR 391
```

The contribution to the surface flux from precipitation, TW, is the minimum of the following: 1) ZRINT * TWDT * 24 (which is the amount of precipitation in TWDT at rate ZRINT); and 2) TWR, the remaining precipitation to infiltrate into the soil. At the end of each TWDT, ZRINT * TWDT * 24 is subtracted from TWR (line 801). Thus, line 391 is necessary to ensure that the sum of ZRINT * TWDT * 24 does not become larger than ZRAIN, total precipitation for the time-step.

```
    TWSF=TW+TWSTND-TWEVAP*TWDT/PMDT                       WATR 392
140 TWSF=TWSF*0.1                                         WATR 395
```

Potential surface flux, in centimeters, equals TW from line 391 above, plus standing water, minus that portion of evaporation for PMDT which occurs this TWDT.

```
150 TWSF=0.0                                              WATR 397
```

We reach here only if soil is frozen (line 380). If frozen, then set surface flux equal to zero and go on.

```
    R=TWDT/PMDT                                           WATR 413
    DO170I=1,PMN-2                                        WATR 414
170 TWTSPR(I)=0.0                                         WATR 415
    DO 210 J=1,6                                          WATR 418
    TWTK=0.0                                              WATR 419
    DO180I=1,PMN-2                                        WATR 420
    TWTA(I)=TWTHA(I+1)-PWLLIM                             WATR 422
180 TWTK=TWTK+TWTA(I)*CVRDST(J,I)                         WATR 423
    IF(TWTK .LE. 1.E-8) GO TO 190                         WATR 426
    TWTK=R*TWTW(J)/TWTK                                   WATR 427
190 CONTINUE                                              WATR 428
    DO200I=1,PMN-2                                        WATR 429
200 TWTSPR(I)=TWTSPR(I)+TWTK*TWTA(I)*CVRDST(J,I)          WATR 430
210 CONTINUE                                              WATR 431
```

Determine TWTSPR(I), centimeters, transpiration demand by layer. It is proportional to water available in the layer TWTA(I), and the fraction of roots in the layer CVRDST(J,I). Transpiration demand is assumed uniformly distributed over PMDT.

Thus, each TWDT, the demand is R = TWDT/PMDT times the total demand. Line 418 begins the main DO loop over plant functional groups (FG's). TWTK first becomes the sum over layers of water available in the layer times fraction of roots of FG J in the layer. Water available is dimensionless and equal to the value of theta at the beginning of TWDT, TWTHA(I+1), minus the lower limit of theta, PWLLIM. The indexing looks strange because TWTHA is indexed over nodes, and TWTA over layers (see Fig. C-2). Line 426 will be true if there is no FG represented by current value of J. (For Curlew Valley simulation, e.g., perennial FG's fill places 1, 2, 3 in XVPLNT, CVRDST, etc., and annual FG's fill places 5, 6, leaving place 4 vacant.) In line 427, TWTK becomes the proportionality factor needed. Lines 429, 430 are a DO loop over soil layers, where, in each pass through the loop, the transpirational demand due to roots of FG J in layer I is added to the demand from layer I. The algebra behind these lines of code is not obvious so more explanation is necessary. If one puts the equation in lines 423 and 427 into line 430 and writes it in normal algebraic form we get:

$$\Delta[\mathrm{TWTSPR(I)}]_J = \frac{\mathrm{TWDT}}{\mathrm{PMDT}}\,[\mathrm{TWTW(J)}]\,\frac{[\mathrm{TWTA(I)}]\,[\mathrm{CVRDST(J,I)}]}{\sum_I[\mathrm{TWTA(I)}]\,[\mathrm{CVRDST(J,I)}]}\;,$$

where $\Delta[\mathrm{TWTSPR(I)}]_J$ means the addition to TWTSPR(I) from FG J. If we sum both sides over I (layers), we find that the total transpirational demand from FG J for this TWDT is (TWDT/PMDT)[TWTW(J)], which is just what we want since TWTW(J) is the demand from FG J over PMDT.

```
      TW=0.0                                                WATR 434
      DO230I=1,PMN-2                                        WATR 435
      S=TWTSPR(I)/CHDX(I)                                   WATR 436
      IF(S.LE.(TWTA(I)*0.5))GOTO220                         WATR 438
      TWTSPR(I)=TWTA(I)*0.5*CHDX(I)                         WATR 439
      S=TWTSPR(I)                                           WATR 440
  220 CONTINUE                                              WATR 441
      IF(TW.LT.S)TW=S                                       WATR 442
  230 CONTINUE                                              WATR 443
```

S = centimeters of transpiration from a layer ÷ thickness of layer = change in theta for layer from transpiration (line 436). Check all layers and set TW = largest value of S. If S ever represents more than half the water available from a layer, reduce transpiration and S to a value equal to half the water available. After this loop, TWTSPR is actual transpiration. A precaution should be noted here. It is possible for the plant submodel(s) to demand more transpiration than WATER is removing from the soil. For one or two TWDT's, this is not serious, but if it occurs over several consecutive PMDT's, the plants will be growing as if there is more soil water available than actually exists. This situation should, and can, be avoided by keying transpiration to soil water in such a manner that, as water is removed from the soil, transpiration becomes zero before theta reaches PWLLIM.

```
      IF((TW.LE.PWTLIM).OR.(TWJM.GE.2).OR.(TWDT.LE.0.02083))GOTO240     WATR 445
      TWJM=TWJM+1                                                       WATR 446
      TWDT=TWDT*0.9*PWTLIM/TW                                           WATR 447
      IF(TWDT.LE.0.02083)TWDT=0.02083                                   WATR 448
      GO TO 130                                                         WATR 449
  240 CONTINUE                                                          WATR 450
```

TWDT will not be reduced if: 1) TW, calculated just above, is less than or equal to the largest allowable change in theta; 2) if it has already been reduced twice before; or 3) if it is already at the minimum value of 30 min. If it is too large, we increment the counter, reduce TWDT and ensure that its value is not below minimum.

Overview of Conductivity and Specific Water Capacity Calculation

In order to accurately simulate water content and potential, it is important to determine hydraulic conductivity, PHK, as accurately as possible. Average PHK for a region between nodes (see Fig. C-2) over TWDT can be calculated if theta is known at the beginning and end of TWDT. But to determine theta at the end of TWDT, we must use average PWK during TWDT. Thus, an iteration scheme would appear to be in order. With a large subroutine such as this, however, it appears likely that it would be easy to set something up which would eat up considerable computer time. To avoid excessive computer-time use, no iterations are performed and, whenever possible, the change in theta for the current TWDT is estimated from the change in theta during the previous TWDT. "Whenever possible" turns out to be over 90% of the time.

```
      IF(TWONE) GO TO 280                                             WATR 466
      TWONE=.TRUE.                                                    WATR 467
```

Use the flag TWONE to ensure going through this section only once each TWDT.

```
      IF((TWRP .AND. TWRAIN) .OR. ((.NOT. TWRP) .AND. (.NOT. TWRAIN)))  WATR 468
     1 GO TO 250                                                        WATR 469
      GO TO 280                                                         WATR 470
  250 CONTINUE                                                          WATR 471
```

If there was rain last TWDT and it is raining now, or if no rain occurred last TWDT and there is no rain now, we can estimate the change in theta this TWDT from change last TWDT. Otherwise, go to the section calculating conductivity and specific water capacity.

```
      DO270I=2,PWN                                                    WATR 472
```

DO loop through nodes (all except top one).

```
      S=TWDT/TWDTP                                                    WATR 474
      IF(S.GT.1.)S=SQRT(S)                                            WATR 475
      TW=TWTHB(I)+S*TWDTHP(I)                                         WATR 476
      IF(TW .GE. PWT(PWM)) GO TO 260                                  WATR 477
      IF(TW .LT. PWLLIM) TW=PWLLIM                                    WATR 478
```

Tentative change in theta is equal to change last TWDT times S. S is the ratio of present and past time-steps if ratio $\leqslant 1$. If ratio > 1, S equals the square root of the ratio as a precaution against overestimating change in theta. Finally, perform checks to ensure that theta is within limits.

```
TWTHB(I)=TW                                                  WATR 479
```

Set theta for current node equal to tentative value.

```
IF(I.GT.2)GOTO270                                            WATR 480
TWJA=TWTHB(I) / PWDELW + 1.                                  WATR 483
TWAA=(TWTHB(I)-PWT(TWJA))/PWDELW                             WATR 485
TWHB(I)=PWH(TWJA)+(PWH(TWJA+1)-PWH(TWJA))*TWAA               WATR 486
GO TO 270                                                    WATR 487
```

If current node is below second node, skip these lines. Otherwise, calculate TWHB, SWP at end of TWDT, for first layer (second node) by interpolating between appropriate values in PWH table. It is used below in the calculation for actual surface flux (line 574).

```
DO 320I=2,PMN-1                                              WATR 494
```

DO loop over PMN—2 inner nodes. In this loop, calculate specific water capacity of each of the PMN—2 layers and hydraulic conductivity of the PMN—3 regions between the inner nodes.

```
TWTHC =(TWTHA(I)+TWTHB(I)) / 2.                              WATR 495
IF(TWTHC .GT. PWT(PWM)) TWTHC=PWT(PWM)                       WATR 496
IF(TWTHC .LT. PWLLIM) TWTHC=PWLLIM                           WATR 497
```

Calculate average water content during TWDT and ensure that values lie between limits.

```
TWJA=TWTHC/PWDELW + 1.                                       WATR 499
IF(TWJA .GE. PWM) TWJA=PWM-1                                 WATR 500
TWAA=(TWTHC-PWT(TWJA))/PWDELW                                WATR 502
TWKSA(I)=PWKSUM(TWJA)+(PWKSUM(TWJA+1)-PWKSUM(TWJA))*TWAA     WATR 504
TWHHA(I)=PWH(TWJA)+(PWH(TWJA+1)-PWH(TWJA))*TWAA              WATR 505
```

Using value of average water content just found, interpolate between appropriate values in PWKSUM and PWH tables, and call these values TWKSA and TWHHA.

```
PWWC(I-1) = PWDELW/(PWH(TWJA+1)-PWH(TWJA))                   WATR 506
```

The specific water capacity is simply the inverse of the slope of the SWP vs. theta relationship.

```
IF(I.LE.2)GOTO310                                            WATR 509
```

Need TWKSA and TWHHA for two adjacent nodes to calculate PWK, hydraulic conductivity, for region between.

```
IF(TWJB .EQ. TWJA) GO TO 300                                 WATR 510
```

If average water content of previous node and current node are close (between same pairs of values in PWKSUM and PWH table), calculate PWK differently. This avoids any chance of a divide-by-zero (or nearly zero) situation.

```
PWK(I-1)=(TWKSA(I)-TWKSA(I-1))/(TWHHA(I)-TWHHA(I-1))                WATR 511
```

Calculate hydraulic conductivity. Hanks and Bowers (1962) call it a "round about" method in that diffusivity is calculated from measured conductivity values, and then average conductivity is determined from diffusivity. If one goes through their algebra, PWK is seen to be average conductivity over a region of soil, weighted with respect to SWP. PWKSUM, calculated in line 1013, is

$$\text{PWKSUM}(M) = \sum_{i=1}^{M} K_i(\Delta\text{SWP})_i ,$$

where

i = index which goes through theta values in the conductivity and SWP tables ($i = 1 \Rightarrow$ theta $=$ PWTLIM; $i = 2 \Rightarrow$ theta $= 2\cdot$PWTLIM, $\ldots$; $i = m \Rightarrow$ theta $= M\cdot$PWTLIM);

K_i = measured conductivity (read in as PWKIN) at theta $= i\cdot$PWTLIM;

$(\Delta\text{SWP})_i$ = change in soil water potential (from PWH array) from theta $= i\cdot$PWTLIM to theta $= (i + 1)\cdot$PWTLIM.

If theta at the top of the region corresponds to $M \cdot$ PWTLIM and theta at the bottom corresponds to $N \cdot$ PWTLIM, then rewrite line 511:

$$\text{average conductivity} = \frac{\sum_{i=1}^{M} K_i\,(\Delta\text{SWP})_i - \sum_{i=1}^{N} K_i\,(\Delta\text{SWP})_i}{\text{SWP}_M - \text{SWP}_N} .$$

Say $M > N$, then

$$\text{average conductivity} = \frac{\sum_{i=N}^{M} K_i\,(\Delta\text{SWP})_i}{\sum_{i=N}^{M} (\Delta\text{SWP})_i} .$$

Thus, average conductivity for a region is an average over measured conductivity, weighted with respect to SWP, as asserted above.

To illustrate, do a numerical example with theta at the node at the top of the region = 0.25 and = 0.15 at the bottom. Use input data for Curlew Valley run.

theta	0.15	0.25
TWKSA	0.4047	0.9157
TWHHA	-8.090	-.8800
PWKIN	0.0198	0.4630

$$\text{weighted average conductivity} = \frac{\text{TWKSA}_{.15} - \text{TWKSA}_{.25}}{\text{TWHHA}_{.15} - \text{TWHHA}_{.25}} ;$$

$$= \frac{0.4047 - .9157}{-8.090 - (-.8800)} = 0.0709 .$$

$$\text{unweighted average conductivity} = \frac{0.01980 + 0.4630}{2} = 0.2414 .$$

The weighted average value, which is the one used in the program, is considerably smaller than the unweighted average. This is consistent with the findings of Hanks and Bowers (1962) that the dry soil (lower conductivity) controls the flow of water more than the wetter soil.

```
 300 PWK(I-1)=(PWKSUM(TWJA+1)-PWKSUM(TWJA)) / (PWH(TWJA+1)-PWH(TWJA))  WATR 513
```

If theta values at nodes outlining the current region are between the same pairs of values in PWKSUM and PWH tables, use this formula which is a special case of the one in line 511 and avoids the possibility of dividing by zero or a very small number.

```
 310 TWJB=TWJA                                    WATR 514
```

We need to know previous value of TWJA on the next pass through the loop.

```
 320 CONTINUE                                     WATR 517
```

End of loop.

Overview of Top Boundary Condition Section

Potential top surface flux was calculated in line 392. Here, we check if we can meet potential by setting soil surface conditions to wettest conditions (if potential flux > 0, or into soil) or to driest (if potential flux < 0). If potential flux can be met, actual flux is set equal to potential flux and program goes on. Since no submodel, including WATER, uses soil surface conditions, the surface node is always either at its wettest or driest condition, and more precise values are not determined. If potential flux cannot be met, actual flux is calculated, and difference between actual and potential is determined. If this difference is positive, it ultimately becomes standing water (line 794). If negative, it eventually is lost.

```
      IF(TWSF.LT.0.0)GOTO330                      WATR 543
      TWHB(1)=PWHWET                              WATR 544
      TWTHB(1)=PWT(PWM)                           WATR 545
      GOTO340                                     WATR 546
 330  CONTINUE                                    WATR 548
      TWHB(1)=PWHDRY                              WATR 551
      TWTHB(1)=PWLLIM                             WATR 552
```

Check sign of surface flux and set surface conditions wettest if positive, driest if negative.

```
 340 CONTINUE                                     WATR 554
C  CALCULATE PWK(1)                               WATR 555
      TWTHC=TWTHB(1)                              WATR 556
      IF(TWTHC .GT. PWT(PWM)) TWTHC=PWT(PWM)      WATR 557
```

```
      IF(TWTHC .LT. PWLLIM) TWTHC=PWLLIM                                WATR 558
C   PWT(TWJA) AND PWT(TWJA+1) BRACKET TWTHC                             WATR 559
      TWJA=TWTHC/PWDELW + 1.                                            WATR 560
      IF(TWJA .GE. PWM) TWJA=PWM-1                                      WATR 561
C   CHECK IF WATER CONTENTS OF FIRST TWO NODES ARE VERY CLOSE           WATR 562
      IF((TWKSA(2) .GE. PWKSUM(TWJA)) .AND. (TWKSA(2) .LE. PWKSUM(TWJA+1WATR 563
     1 ))) GO TO 350                                                    WATR 564
C   TWAA IS FOR INTERPOLATING                                           WATR 565
      TWAA=(TWTHC-PWT(TWJA))/PWDELW                                     WATR 566
      TWKSA(1)=PWKSUM(TWJA)+(PWKSUM(TWJA+1)-PWKSUM(TWJA))*TWAA          WATR 567
      TWHHA(1)=TWHB(1)                                                  WATR 568
      PWK(1)=(TWKSA(2)-TWKSA(1))/(TWHHA(2)-TWHHA(1))                    WATR 569
      GO  TO 360                                                        WATR 570
  350      PWK(1) = (PWKSUM(TWJA+1)-PWKSUM(TWJA)) / (PWH(TWJA+1)        WATR 571
     1 -PWH(TWJA))                                                      WATR 572
  360 CONTINUE                                                          WATR 573
```

Determine PWK(1) just as PWK(I) for I = 2 to PMN—2 were determined in lines 494-517. The only difference is that water content is not averaged in line 556 as it was in line 495 (averaging surface node led to nasty oscillations).

```
      TWSFC=PWK(1)*TWDT*(TWHB(1)-TWHB(2)+9.833E-4*CHDXX(1))/CHDXX(1)    WATR 574
```

Calculated surface flux equals conductivity times time-step times SWP gradient [the term 9.833—4*CHDXX(1) is contribution due to gravity].

```
      IF(TWSF.LT.0.0)GOTO370                                            WATR 575
      IF(TWSFC.GE.TWSF)GOTO390                                          WATR 576
      GOTO380                                                           WATR 577
  370 IF(TWSFC.LE.TWSF)GOTO390                                          WATR 578
```

Check if potential surface flux condition can be met.

```
  380 CONTINUE                                                          WATR 580
      TWIN=.FALSE.                                                      WATR 582
      TWSFD=TWSF-TWSFC                                                  WATR 583
      TWSF=TWSFC                                                        WATR 584
      GOTO400                                                           WATR 585
```

If we reach here, they can't be met. Set flag announcing this, determine difference between potential and calculated fluxes, set surface flux equal to calculated value and go on.

```
  390 CONTINUE                                                          WATR 587
      TWIN=.TRUE.                                                       WATR 591
      TWSFD=0.0                                                         WATR 592
  400 CONTINUE                                                          WATR 594
```

If we reach here, potential surface flux condition can be met. Set flag announcing this, set difference between potential and actual equal to zero and go on.

```
      IF(.NOT. TWFRZN) GO TO 420                                        WATR 600
      DO410I=1,PMN-2                                                    WATR 601
  410 PWK(I)=PWK(I)*1.E-6                                               WATR 602
  420 CONTINUE                                                          WATR 603
```

If soil is frozen, reduce hydraulic conductivity by a factor of a million and stop soil water movement.

```
      CO 430 I=1,PMN-1                                                  WATR 641
      TWWA(I) = PWK(I) / CHDXX(I)                                       WATR 642
      IF(I .EQ. PMN-1) GO TO 430                                        WATR 643
      TWWB(I) = (2.*PWWC(I)*CHDX(I)) / TWDT                             WATR 644
  430 CONTINUE                                                          WATR 645
      DO 440 I=2, PMN-3                                                 WATR 647
      TWA(I) = TWWA(I+1)                                                WATR 648
      TWC(I) = TWWA(I)                                                  WATR 649
      TWB(I) = TWWB(I)+TWA(I)+TWC(I)                                    WATR 650
  440 TWD(I) = TWHA(I+1)*(TWWB(I)-TWA(I)-TWC(I)) + TWHA(I)*TWC(I)       WATR 651
     1 + TWHA(I+2)*TWA(I) + 2.*(PWK(I) - PWK(I+1))  *9.833E-4           WATR 652
     2 -2.*TWTSPR(I)/TWDT                                               WATR 653
      TWA(1) = TWWA(2)                                                  WATR 655
      TWC(1) = 0.                                                       WATR 656
      TWB(1) = TWWB(1) + TWA(1)                                         WATR 657
      TWD(1) = TWHA(2)*(TWWB(1)-TWA(1)) + TWHA(3)*THA(1)                WATR 658
     1  + TWSF*2./TWDT   -2.*PWK(2)*9.833E-4                            WATR 659
     2 -2.*TWTSPR(1)/TWDT                                               WATR 660
      TWA(PMN-2) = 0.                                                   WATR 661
      TWC(PMN-2) = TWWA(PMN-2)                                          WATR 662
      TWB(PMN-2) = TWWB(PMN-2) + TWC(PMN-2) +TWWA(PMN-1)                WATR 663
      TWD(PMN-2) = TWHA(PMN-1)*(TWWB(PMN-2)-TWC(PMN-2)-TWWA(PMN-1))     WATR 664
     1 + TWHA(PMN-2)*TWC(PMN-2) + 2.*TWHB(PMN)*TWWA(PMN-1)              WATR 665
     2 +2.*(PWK(PMN-2)-PWK(PMN-1)) *9.833E-4                            WATR 666
     3 -2.*TWTSPR(PMN-2)/TWDT                                           WATR 667
```

Generate matrix elements of tridiagonal matrix. These elements are coefficients of SWP in the system of difference equations (one for each layer) we need to solve to determine SWP of each layer. For more information see "Derivation of Difference Equations." These matrix elements can be determined only after we've determined: 1) hydraulic conductivity; 2) specific water capacity; 3) transpiration; 4) top surface flux.

```
      CALL TDM(TWA, TWB, TWC, TWD, TWU, PMN-2)                          WATR 677
```

TDM is the subroutine which solves the tridiagonal matrix, generated above, for PMN—2 values of SWP. These values are returned to WATER in array TWU. For more details, see the Detailed Program Description of TDM.

```
      DO450I=2,PMN-1                                                    WATR 690
      TWTHB(I)=TWTHA(I)+PWWC(I-1)*(TWU(I-1)-TWHA(I))                    WATR 691
      IF(TWTHB(I) .GT. PWT(PWM)) TWTHB(I)=PWT(PWM)                      WATR 692
      IF(TWTHB(I) .LT. PWLLIM) TWTHB(I)=PWLLIM                          WATR 693
  450 CONTINUE                                                          WATR 694
```

Calculate TWTHB, theta of nodes at end of TWDT, from values of TWU just returned from subroutine TDM. Equation used is

$$(\text{theta}) = (\text{specific water capacity}) \times [\Delta(\text{SWP})].$$

Ensure that values obtained are within range.

```
      CO460I=2,PMN-1                                                    WATR 697
      TWJA=TWTHB(I)/PWDELW+1.                                           WATR 698
      IF(TWJA .GE. PWM) TWJA=PWM-1                                      WATR 699
      TWAA=(TWTHB(I)-PWT(TWJA))/PWDELW                                  WATR 700
  460 TWHB(I)=PWH(TWJA)+(PWH(TWJA+1) - PWH(TWJA))*TWAA                  WATR 701
```

Calculate TWHB, SWP of nodes at end of TWDT, from values of theta just calculated (same method as that starting in lines 495 and 556). This is roundabout in that we could have set TWHB(I) = TWU(I—1) for I = 2, PMN—1. By using the specific water capacity, we find theta much faster than by hunting backwards through array PWH, but we run the risk of the two values not exactly coinciding because of the various averages used. By determining SWP from theta rather than directly from TWU, then, small errors might occur, but theta and SWP will be consistent with each other.

```
      DO470I=2,PMN-1                                      WATR 705
      S=ABS(TWTHB(I)-TWTHA(I))                            WATR 706
      IF(S.GT.PWTLIM)GOTO480                              WATR 707
  470 CONTINUE                                            WATR 708
      GO TO 510                                           WATR 709
```

Check if any of the delta thetas just calculated are larger than the limit PWTLIM. If yes, see if TWDT can be reduced. If no, go on to check of water balance.

```
  480 IF(TWJM.GE.2)GOTO510                                WATR 710
      TWDTO=TWDT                                          WATR 711
      TWDT=TWDT*0.9*PWTLIM/S                              WATR 712
      TWJM=TWJM+1                                         WATR 713
      IF(TWDT.LT.0.02083)TWDT=0.02083                     WATR 715
      R=TWDT/TWDTO                                        WATR 716
  490 DO500I=2,PMN-1                                      WATR 717
      TWHB(I)=TWHA(I)+(TWHB(I)-TWHA(I))*R                 WATR 718
  500 TWTHB(I)=TWTHA(I)+(TWTHB(I)-TWTHA(I))*R             WATR 719
      GO TO 130                                           WATR 720
  510 CONTINUE                                            WATR 721
```

Reduce TWDT if: 1) it hasn't been reduced two or more times already (limit the number of iterations to cut down on computer time); and 2) if it isn't already at its lower limit of 30 min (0.02083 days). In line 712, reduce TWDT; the factor 0.9 is to be on the safe side. Then increment the interation counter and make sure TWDT is not less than 30 min. Finally, reduce the estimates of SWP and theta just calculated above by making the change proportional to the ratio of new TWDT to old TWDT, and go back to statement 130 (line 375) which is the start of the loop to calculate SWP once TWDT is set.

```
      TWSFB=-PWKCAL*TWDT*((TWHB(PMN-1) + TWHA(PMN-1)) * .5   WATR 726
     1 +9.833E-4*CHDXX(PMN-1) - TWHB(PMN)) / CHDXX(PMN-1)    WATR 727
```

Calculate water leaving bottom of soil profile (remember that convention is: in is +). This is basically the same equation as line 574.

```
      CALL WBAL( TWTSPR, TWSF, TWSFB, TWTHA, TWTHB, CHDX, PMN,   WATR 730
     1   TWWIN, TWSTRD, R)                                       WATR 731
```

Call subroutine WBAL to check if the sum of the amounts of water into or out of the profile (rain, evaporation, transpiration, water through caliche layer) is equal to the change in the water stored in the profile. For more detail see "Subroutine WBAL," below.

```
      IF(TWDT.LT.0.05)GOTO520                             WATR 733
      IF((R .GT. 99.) .OR. (ABS(TWWIN-TWSTRD) .LT. 0.006)  WATR 736
     1 .OR.(ABS(R-1.).LT.0.01))GOTO520                    WATR 737
      TWDT=0.5*TWDT                                       WATR 738
```

```
      R=0.5                                       WATR 739
      GOTO490                                     WATR 740
  520 CONTINUE                                    WATR 741
```

If the amount of water into or out of the profile is equal to the change in water stored, TWDT remains unchanged. If they are not equal, TWDT may be halved. TWDT will not be halved if its value is currently small (less than .05). Also, it will not be halved if one of the following conditions is met: 1) if R, the ratio of water in to water stored, is greater than 99 (to avoid a divide by zero, WBAL sets R = 99.9999 if TWSTRD is less than 0.001); 2) if TWWIN is sufficiently close to TWSTRD (within 0.006); 3) if R is sufficiently close to 1.0 (within 0.01).

```
      CWINF=CWINF+TWSF                            WATR 778
      TWSFBT = TWSFBT + TWSFB                     WATR 782
```

Accumulate top and bottom surface fluxes. Neither flux includes transpiration.

```
      IF(.NOT. TWFRZN) GO TO 570                  WATR 785
      TWSTND=TWSTND+TWR                           WATR 786
      TWR=0.0                                     WATR 787
      GO TO 620                                   WATR 788
  570 CONTINUE                                    WATR 789
```

If soil is frozen, add precipitation to standing water and skip the normal standing water calculation.

```
      IF(TWIN) GO TO 580                          WATR 792
      TWSTND=TWSFD*10.*PWRNRF**TWDT               WATR 794
      IF (TWSTND .LT. 0.0) TWSTND=0.0             WATR 795
      GO TO 590                                   WATR 796
  580 TWSTND=0.0                                  WATR 797
  590 CONTINUE                                    WATR 798
```

If potential surface flux conditions are met there will be no standing water. If unmet, then in line 794 calculate new level of standing water using parameter PWRNRF, which provides a crude estimate of runon or runoff and will have a value near 1. (A value of 1.25 for Curlew Valley, e.g., made water continually build up on a surface once the spring melting occurred.) TWSFD is the potential surface flux minus the actual surface flux; the factor 10 changes units to millimeters; and PWRNRF to the TWDT power is a compound interest type factor.

```
  600 TWR=TWR-ZRINT*TWDT*24.                      WATR 801
  610 IF(TWR .LT. 0.0) TWR=0.0                    WATR 802
```

Determine the amount of precipitation remaining to infiltrate this PMDT. Make sure it doesn't get negative.

```
      XWSTND=TWSTND                               WATR 805
```

Set state variable XWSTND equal to TWSTND.

```
     DO640I=1,PWN-2                                              WATR 826
     XWTHTA(I)=TWTHB(I+1)                                        WATR 827
 640 CWPSI(I)=TWHB(I+1)                                          WATR 828
```

Set state variable XWTHTA, theta of layers, equal to theta of appropriate node at end of PMDT (remember layer 1 is centered on node 2). Set communication variable CWPSI, SWP of layers, equal to SWP of appropriate node at the end of PMDT.

```
     ENTRY WINIT                                                 WATR 858
```

This entry point is called once, from MAIN. It is used only for initialization purposes: 1) to read in WATER variables; 2) to initialize miscellaneous flags and variables; 3) to perform calculations which need to be done only once.

```
     READ(5,670)TWCCC                                            WATR 870
     WRITE(6,680)                                                WATR 871
     WRITE(6,690)TWCCC                                           WATR 872
```

Read and write a comment which may say, e.g., INITIALIZATION DATA FOR WATER.

```
     READ(5,670)TWCCC                                            WATR 875
     WRITE(6,690)TWCCC                                           WATR 876
     READ(5,/)PWDELW                                             WATR 877
     WRITE(6,710)PWDELW                                          WATR 878
```

Read and write a comment which describes variable PWDELW, which is then read and written. In similar manner, variables PWH, PWKCAL, PWKIN, PWHDRY, PWHWET, PWM, PWTLIM, PWLLIM, PWRNRF, TWSTND and TWTHB are read and written.

```
     DO 750 I=1, PWM                                             WATR 999
     TWI = I-1                                                   WATR1000
 750 PWT(I)=TWI*PWDELW                                           WATR1001
```

Set up array PWT, containing PWM values of theta, each PWDELW apart. The conductivity in PWKIN(I) and the SWP in PWH(I) correspond to a theta value of PWT(I).

```
     DO 760 I=1,PWN                                              WATR1004
     TWJA=TWTHB(I)/PWDELW + 1.                                   WATR1005
     IF(TWJA .GE. PWM) TWJA=PWM-1                                WATR1006
     TWAA=(TWTHB(I)-PWT(TWJA)) / PWDELW                          WATR1007
 760 TWHB(I)=PWH(TWJA)+(PWH(TWJA+1)-PWH(TWJA))*TWAA              WATR1008
```

Calculate initial SWP values from theta values just read in TWTHB. Approach used is same as in line 483.

```
     PWKSUM(1)=(PWKIN(1)+PWKIN(2))*(PWH(2)-PWH(1))*0.5           WATR1011
     DO 770 I=2,PWM-1                                            WATR1012
 770 PWKSUM(I)=(PWKIN(I)+PWKIN(I+1))*(PWH(I+1)-PWH(I))*0.5 +PWKSUM(I-1)WATR1013
     PWKSUM(PWM)=2.*PWKSUM(PWM-1)-PWKSUM(PWM-2)                  WATR1017
```

Calculate PWKSUM, sum of conductivity times delta SWP. This is used in line 504 in average conductivity calculations. Each element in PWKSUM equals the previous element plus an increment. Take PWKSUM(I), e.g. It equals PWKSUM(I—1) plus the increment, which is the average conductivity for theta between values of PWT(I) and PWT(I+1), times the change in SWP for theta between values of PWT(I+1) and PWT(I). In line 1017, the last increment is taken equal to the second to last increment.

```
      WRITE(6,780)                                                      WATR1020
  780 FORMAT('   PWKSUM,   SUM OF CONDUCTIVITY TIMES DELTA PRESSURE')   WATR1021
      WRITE(6,720) PWKSUM                                               WATR1022
```

The array PWKSUM, which was just calculated, is written out.

```
      READ(5,670)TWCCC                                                  WATR1025
      WRITE(6,690)TWCCC                                                 WATR1026
```

Read and write a comment which says, e.g., END INITIALIZATION FOR WATER.

SUBROUTINE WBAL

WBAL is a small, simple subroutine which acts as an overall check on the accuracy of the calculations performed in WATER. For the time period in question, either TWDT or PMDT, it calculates WIN, amount of water into or out of the soil profile by transpiration, and top and bottom surface flux. It also calculates WSTRD, change in water stored during time period, from the changes in theta of each layer and from layer thickness. WBAL then determines the ratio of these two quantities. The calling program, WATER, in this case, then determines if the ratio is close enough to 1.

```
      DO100I=1,PMN-2                                                    WBAL  16
      A=A+TSPR(I)                                                       WBAL  17
  100 WSTRD=WSTRD+(THTAF(I+1)-THTAI(I+1))*DX(I)                         WBAL  18
```

This is a DO loop through layers. Variable A accumulates transpiration, WSTRD accumulates changes in water held in layers.

```
      WIN = SF - A + SFB                                                WBAL  19
```

Calculate WIN, change in water in profile from surface fluxes (top and bottom) and transpiration (convention is that water in is positive).

```
      IF(ABS(WSTRD).LT.1.E-3)RATIO=99.9999                              WBAL  20
```

Set RATIO equal to a strange large number if WSTRD is small (avoid divide by zero or near zero).

```
      IF(ABS(WSTRD).GE.1.E-3)RATIO=WIN/WSTRD                            WBAL  21
```

Actually calculate RATIO if WSTRD is large enough.

FUNCTION WTIME

The purpose of this function is to determine total elapsed CPU time since the beginning of the run. Because we use the variable name TIME for other purposes in the model, the Burroughs Intrinsic Function TIME must be put in a subprogram of some kind in order for it to be used. The Z in the argument of WTIME is a completely dummy variable -- a FORTRAN function needs an argument whether it is used or not.

```
WTIME=0.016667*TIME(2)                                WTM 08
```

TIME(2) is CPU time in 60ths of a second. The factor 0.016667 = 1/60 changes the right-hand side to seconds. CPU time is useful for debugging purposes.

COMPLETE PROGRAM LISTING

SUBROUTINE WATER

```
      SUBROUTINE WATER
C
C  WRITTEN BY PAUL W. LOMMEN
C             DESERT BIOME - ECOLOGY CENTER UMC 52
C             UTAH STATE UNIVERSITY
C             LOGAN, UTAH 84322
C
C   AUGUST 1976
C
C  THIS SUBMODEL OF THE WATER RESPONSE MODEL CALCULATES SOIL WATER
C  POTENTIAL IN UP TO 8 SOIL LAYERS.   INPUTS NEEDED ARE PRECIPITATION,
C  POTENTIAL EVAPORATION, TRANSPIRATION, SOIL HYDRAULIC CONDUCTIVITY -
C  SOIL WATER CONTENT RELATIONSHIP, AND SOIL WATER POTENTIAL - SOIL
C  WATER CONTENT RELATIONSHIP.
C
C
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
C             LIST OF VARIABLES FOR WATER
C
C  CHD(10)    DEPTHS, CM, OF NODES BELOW SURFACE (A NODE IS A POINT
C             WHERE A WATER POTENTIAL IS CALCULATED) STARTING AT SURFACE
C
C  CHOX(10)   THICKNESS OF REGION CENTERED ON NODE WHERE COUNTING
C             STARTS AT FIRST NODE BELOW SURFACE AND GOES DOWN
C
C  CHDXX(10)  CHDXX(I)=CHD(I+1) - CHD(I)
C
C  CVTSPR(10) MM OF WATER TAKEN FROM EACH LAYER DURING PWDT
C
C  CWINF      NET TOP SURFACE FLUX FOR PWDT, CM, DOES
C             NOT INCLUDE TRANSPIRATION.
C
C  CWPSI(10)  SOIL WATER POTENTIAL, BARS, VALUE AT END OF PWDT
C
C  DBGFLG     IF=.TRUE. WRITE DEBUGGING INFORMATION THIS TIME STEP
C
C  DEBUG1          NUMBER OF TIME STEPS BETWEEN DEBUGGING OUTPUTS
C
C  FLGFLG     IF=.TRUE. WRITE DEBUGGING INFORMATION THIS TWDT
C
C  PWDT       TIME STEP DETERMINED IN MAIN, DAYS
C
C  PWN        NUMBER OF NODES, SAME AS NUMBER OF NODES IN HEAT,
C             COUNTING STARTS AT SURFACE
C
C  PWDELW     INCREMENT IN VWC, TYPICALLY 0.01-0.02
C
C  PWH(60)    TABLE OF HYDRAULIC PRESSURE HEAD VERSUS THETA, BARS,
C             USING SAME THETA SCALE AND SPACING AS PWKIN
C
C  PWHDRY     ALLOWABLE LOW PRESSURE LIMIT FOR ANY LAYER
C
C  PWHWET     ALLOWABLE HIGH PRESSURE LIMIT FOR ANY LAYER
C
C  PWK(10)    HYDRAULIC CONDUCTIVITY, CM2 BAR-1 DAY-1, PWK(I) IS
C             CONDUCTIVITY AVERAGED OVER TWDT, IN REGION BETWEEN NODES
C             I AND I+1, WHERE COUNTING STARTS AT SURFACE.
C
C  PWKCAL     CONDUCTIVITY OF CALICHE LAYER
C
C  PWKIN(60)  TABLE OF CONDUCTIVITY VERSUS THETA, CM2 BAR-1 DAY-1,
C             STARTING WITH VALUE AT THETA = 0.0,  READ IN IN WINIT.
C
C  PWKSUM(60) ARRAY USED TO CALCULATE AVERAGE K, (CM-BARS)/DAY
C
C  PWLLIM     LOWER LIMIT OF THETA, DIMENSIONLESS
C             CALCULATED IN WINIT AND = SUM OF CONDUCTIVITY X DELTA H
C
C  PWM        NUMBER OF ENTRIES IN CONDUCTIVITY AND PRESURE TABLES
C
C  PWRNRF     RUNON TO RUNOFF RATIO PER DAY
C
C  PWT(60)    ARRAY CONTAINING UNIFORMLY DISTRIBUTED VALUES OF THETA
C
C  PWTLIM     LARGEST CHANGE IN THETA ALLOWED FOR NODES 2 TO PWN-1
C             DURING ANY TIME STEP TWDT. TYPICAL VALUE .01 TO .02
C
C  PWWC(10)   WATER CAPACITY, BAR-1, OF REGIONS SURROUNDING NODES,
C             COUNTING STARTS AT  FIRST NODE BELOW SURFACE
C
C  R          A VERY TEMPORARY VARIABLE - USED IN REFINED TWDT ESTIMATE,
C             TRANSPIRATION SECTION, REDUCTION OF TWDT IF DELTA THETA
C             TOO LARGE, AND AS RATIO OF WATER IN TO WATER STORED
```

```
C             AS CALCULATED IN WBAL.                                    WATR 085
C                                                                       WATR 086
C  S          A VERY TEMPORARY VARIABLE - USED IN REFINED TWDT ESTIMATE,WATR 087
C             TRANSPIRATION SECTION, CONDUCTIVITY SECTION, AND REDUCTIONWATR 088
C             OF TWDT IF DELTA THETA TOO LARGE.                         WATR 089
C                                                                       WATR 090
C  T          A VERY TEMPORARY VARIABLE - USED IN REFINED TWDT ESTIMATE.WATR 091
C                                                                       WATR 092
C  TDM        A SUBROUTINE USED BY WATER AND HEAT TO SOLVE A SET OF     WATR 093
C             SIMULTANEOUS DIFFERENCE EQUATIONS, THE LEFT HAND SIDES OF WATR 094
C             WHICH FORM A TRI-DIAGONAL MATRIX.                         WATR 095
C                                                                       WATR 096
C  TIME       CURRENT JULIAN DATE - BOOKKEEPING DONE IN MAIN PROGRAM.   WATR 097
C                                                                       WATR 098
C  TSTART     JULIAN DATE OF START OF  RUN - INITIALIZED IN MAIN PROGRAMWATR 099
C                                                                       WATR 100
C  TWAA       FRACTION OF PWDELW INTERVAL, USED IN INTERPOLATING        WATR 101
C                                                                       WATR 102
C  TWA(10)    PARAMETER A FOR TRI-DIAGONAL MATRIX ROUTINE (TDM ROUTINE) WATR 103
C                                                                       WATR 104
C  TWAAA(10)  ARRAY CONTAINING VALUES OF TWTHA(I) FOR I=2,PMN-1         WATR 105
C                                                                       WATR 106
C  TWB(10)    PARAMETER B FOR TDM ROUTINE                               WATR 107
C                                                                       WATR 108
C  TWBE(10)   CONTAINS ORDERED VALUES OF TWAAA.                         WATR 109
C                                                                       WATR 110
C  TW         A VERY TEMPORARY VARIABLE - USED IN SURFACE FLUX          WATR 111
C             CALCULATION, TRANSPIRATION SECTION, AND CONDUCTIVITY      WATR 112
C             SECTION.                                                  WATR 113
C                                                                       WATR 114
C  TWC(10)    PARAMETER C FOR TDM ROUTINE                               WATR 115
C                                                                       WATR 116
C  TWCCC      USED TO READ AND  WRITE COMMENTS IN INPUT DATA            WATR 117
C                                                                       WATR 118
C  TWD(10)    PARAMETER D FOR TDM ROUTINE                               WATR 119
C                                                                       WATR 120
C  TWDT       ACTUAL TIME STEP WATER IS USING                           WATR 121
C                                                                       WATR 122
C  TWDTHP(10) CHANGE IN THETA DURING LAST TWDT                          WATR 123
C                                                                       WATR 124
C  TWDTP      LENGTH OF PREVIOUS WATER TIME STEP                        WATR 125
C                                                                       WATR 126
C  TWDTQ      HOLDS CURRENT VALUE OF TWDT AS IT IS BEING REDUCED        WATR 127
C             BECAUSE DELTA THETA IS TOO LARGE.                         WATR 128
C                                                                       WATR 129
C  TWEVAP     EVAPORATIVE DEMAND, MM, WATER TO EVAPORATE FROM SURFACE   WATR 130
C             DURING REMAINDER OF PMDT                                  WATR 131
C                                                                       WATR 132
C  TWFRZN     EQUALS .TRUE. IF SOIL PROFILE FROZEN, EQUALS              WATR 133
C             .FALSE. IF NOT FROZEN.                                    WATR 134
C                                                                       WATR 135
C  TWI        TEMPORARY VARIABLE USED TO SET UP PWT ARRAY.              WATR 136
C                                                                       WATR 137
C  TWHA(10)   PRESSURE HEAD AT BEGINNING OF TWDT, BARS                  WATR 138
C                                                                       WATR 139
C  TWHB(10)   PRESSURE HEAD AT END OF TWDT, BARS                        WATR 140
C                                                                       WATR 141
C  TWHHA      INTERPOLATED VALUES OF PWH                                WATR 142
C                                                                       WATR 143
C  TWIN       EQUALS .TRUE. IF SURFACE PRESSURE IS WITHIN LIMITS        WATR 144
C                                                                       WATR 145
C  TWJA       INTEGER USED IN PICKING VALUES OF K, H OFF TABLES         WATR 146
C                                                                       WATR 147
C  TWJB       INTEGER USED IN PICKING VALUES OF K, H OFF TABLES         WATR 148
C                                                                       WATR 149
C  TWJM       INTEGER COUNTER USED TO LIMIT NUMBER OF TWDT HALVINGS     WATR 150
C                                                                       WATR 151
C  TWKSA      INTERPOLATED VALUE OF PWKSUM                              WATR 152
C                                                                       WATR 153
C  TWONE      EQUALS .TRUE. IF FIRST DELTA THETA APPROXIMATION HAS      WATR 154
C             ALREADY BEEN DONE DURING CURRENT TWDT.                    WATR 155
C                                                                       WATR 156
C  TWR        RAIN, MM, WATER TO INFILTRATE, LEAVE STANDING OR          WATR 157
C             RUNOFF DURING REMAINDER OF PMDT.                          WATR 158
C                                                                       WATR 159
C  TWRAIN     EQUALS .TRUE. IF RAIN OCCURS DURING TWDT                  WATR 160
C                                                                       WATR 161
C  TWRP       EQUALS .TRUE. IF TWRAIN WAS TRUE LAST TWDT.               WATR 162
C                                                                       WATR 163
C  TWSF       SURFACE FLUX OF WATER OR VAPOR,CM                         WATR 164
C                                                                       WATR 165
C  TWSFB      FLUX OF WATER MOVING INTO PROFILE DURING TWDT THROUGH     WATR 166
C             CALICHE, CM.                                              WATR 167
C                                                                       WATR 168
C  TWSFBT     FLUX OF WATER MOVING INTO PROFILE DURING PMDT THROUGH     WATR 169
C             CALICHE, CM.                                              WATR 170
C                                                                       WATR 171
C  TWSFC      CALCULATED SURFACE FLUX USING TWHB(1) AND PWK(1).         WATR 172
C                                                                       WATR 173
```

```
C  TWSFD       EQUALS TWSF-TWSFC                                         WATR 174
C                                                                        WATR 175
C  TWSIND      STANDING WATER, MM, AFTER RUNON-RUNOFF ASSUMPTIONS HAVE   WATR 176
C              BEEN CONSIDERED.                                          WATR 177
C                                                                        WATR 178
C  TWSTRD      CHANGE DURING TIME STEP IN WATER STORED IN LAYERS OF      WATR 179
C              PROFILE, CM. (SHOULD EQUAL TWWIN)                         WATR 180
C                                                                        WATR 181
C  TWTA(10)    WATER AVAILABLE IN LAYERS, DIMENSIONLESS                  WATR 182
C                                                                        WATR 183
C  TWTHAA(10)  VOLUMETRIC WATER CONTENT AT BEGINNING OF PMDT             WATR 184
C                                                                        WATR 185
C  TWTHA(10)   VOLUMETRIC WATER CONTENT (VWC) AT BEGINNING OF TWDT       WATR 186
C                                                                        WATR 187
C  TWTHB(10)   VWC AT END OF TWDT                                        WATR 188
C                                                                        WATR 189
C  TWTHC       THETA AVERAGED OVER TWDT FOR WHATEVER LAYER WE'RE         WATR 190
C              CALCULATING AT THE MOMENT                                 WATR 191
C                                                                        WATR 192
C  TWTK        PROPORTIONALITY CONSTANT WHICH IS CALCULATED TO MAKE      WATR 193
C              TRANSPIRATION DEMAND FROM A LAYER PROPORTIONAL TO WATER   WATR 194
C              AVAILABLE AND RELATIVE ROOT AMOUNTS IN A LAYER.           WATR 195
C                                                                        WATR 196
C  TWTHTS(10)  RUNNING SUM OVER TWDT'S OF TRANSPIRATION BY LAYER, CM     WATR 197
C                                                                        WATR 198
C  TWTSPR(10)  TRANSPIRATION LOSS OF WATER FROM REGION SURROUNDING NODE, WATR 199
C              CM, (TOP AND BOTTOM NODES EXCLUDED) DURING TWDT           WATR 200
C                                                                        WATR 201
C  TWTTOT      TOTAL TIME THAT HAS ELAPSED IN PMDT. DOES NOT INCLUDE     WATR 202
C              CURRENT TWDT, DAYS                                        WATR 203
C                                                                        WATR 204
C  TWTW(6)     TRANSPIRATIONAL DEMAND BY FUNCTIONAL GROUP, CM            WATR 205
C                                                                        WATR 206
C  TWU(10)     WATER POTENTIALS IN BARS OF SOIL LAYERS AS RETURNED       WATR 207
C              FROM SUBROUTINE TDM.                                      WATR 208
C                                                                        WATR 209
C  TWWA(10)    USED IN SETTING UP CALL TO TDM                            WATR 210
C                                                                        WATR 211
C  TWWB(10)    USED IN SETTING UP CALL TO TDM                            WATR 212
C                                                                        WATR 213
C  TWWIN       FLUX OF WATER MOVING IN TO PROFILE THIS TIME STEP,        WATR 214
C              EQUALS FLUX IN AT TOP + FLUX IN FROM BOTTOM               WATR 215
C              - TRANSPIRATION, CM.  (SHOULD EQUAL TWSTRD)               WATR 216
C                                                                        WATR 217
C  U           A VERY TEMPORARY VARIABLE-USED IN REFINED TWDT ESTIMATE   WATR 218
C                                                                        WATR 219
C  WBAL        SUBROUTINE WHICH CHECKS OVERALL ACCURACY OF CALCULATIONS  WATR 220
C              DONE IN WATER - DETERMINES WATER INTO OR OUT OF PROFILE   WATR 221
C              AND COMPARES IT TO CHANGE IN WATER STORED IN PROFILE.     WATR 222
C                                                                        WATR 223
C  WTIME       FUNCTION WHICH RETURNS ELAPSED CPU TIME IN SECONDS FROM   WATR 224
C              START OF RUN.                                             WATR 225
C                                                                        WATR 226
C  XHSOLT(10)  SOIL TEMPERATURE PROFILE, C.  INDEXED BY NODES, SO        WATR 227
C              XHSOLT(2), E.G., IS TEMPERATURE OF FIRST LAYER.           WATR 228
C                                                                        WATR 229
C  XWSIND      STATE VARIABLE CALCULATED BY WATER EQUAL TO AMOUNT        WATR 230
C              OF WATER STANDING ON SOIL SURFACE, MM.                    WATR 231
C                                                                        WATR 232
C  XWTHTA(10)  THETA OF LAYER AT END OF PMDT                             WATR 233
C                                                                        WATR 234
C  Y           A VERY TEMPORARY VARIABLE - USED ONLY TO HOLD VALUE       WATR 235
C              OF FUNCTION WTIME.                                        WATR 236
C                                                                        WATR 237
C  Z           A DUMMY USED AS AN ARGUMENT IN WTIME                      WATR 238
C                                                                        WATR 239
C  ZEVAP       MM WATER POTENTIALLY EVAPORATED FROM SOIL SURFACE IN PMDT WATR 240
C              VALUE OBTAINED FROM SUBROUTINE PSWG                       WATR 241
C                                                                        WATR 242
C  ZRAIN       RAINFALL, MM IN PMDT                                      WATR 243
C                                                                        WATR 244
C  ZRINT       RAINFALL INTENSITY, MM/HR                                 WATR 245
C                                                                        WATR 246
C                                                                        WATR 247
C              END OF VARIABLES LIST FOR WATER                           WATR 248
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVWATR 249
CVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVWATR 250
C                                                                        WATR 251
C                                                                        WATR 252
C                                                                        WATR 253
      INTEGER PMN,PWM,TWJA,TWJB,TWJM,PMFGPS                              WATR 254
C                                                                        WATR 255
      LOGICAL DBGFLG,FLGFLG,TWFRZN,TWIN,TWONE,TWRAIN,TWRP                WATR 256
C                                                                        WATR 257
      DIMENSION PWWC(10),         TWA(10), TWB(10), TWC(10), TWD(10),    WATR 258
     1 TWTHA(10), TWTHB(10), TWHA(10), TWHB(10), TWWA(10), TWWB(10),     WATR 259
     2 PWKIN(60), PWH(60), PWKSUM(60), PWT(60) , TWU(10),TWCCC(20)       WATR 260
     3 , TWTSPR(10), TWKSA(10), TWHHA(10), TWDTHP(10)                    WATR 261
     4 , TWTHAA(10), TWTHTS(10), TWTA(10), TWTW(6)                       WATR 262
C                                                                        WATR 263
```

```
      COMMON/DB/DEBUG1,DEBUG2,DEBUG3
C
      COMMON TIME,TSTART,TEND,DT,DTPR,DTPL,
     1CAAR(6,8),CADEST(6,8),CADETH,CAUWST,CDDL(12,4),CHD(10),CHDX(10),
     2CHDXX(10),CNFIX,CNSDF,CNSDS,CNSOMF,CNSOMS,CNSUP,CVDETH(12,4),
     3CVLTFR(11,4),CVPHEN(6,8),CVPHS(6),CVRDST(6,10),CVTSPR(6,8),CVVCOV,
     4CWINF,CWPSI(10),PMDT,PMDTPL,PMDTPR,PMFGPS,PMJDAT,PMN,PMNCOH,PMNSP,
     5PWK(10),XAA(4),XAAVWT(4,8),XAAWT(4),XAFTS(4),XAFWT(4),XANUMB(4,8),
     6XASA(4),XASAWT(4),XAYNG(4),XAYWT(4),XHSOLT(10),XNMN,XNSOMF,XNSOMS,
     7XVFG(6,5),XVLITR(12,4),XVPLNT(6,8),XVTOTL,XWTHTA(10),XWSTND,ZAIRT,
     8ZEVAP,ZESUM,ZPHPD,ZRAIN,ZRSUM,ZRH,ZRINT,ZRISUM,ZSUN,ZTMAX,ZTMIN,
     9ZWIND
C
C
CIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
C  INITIAL DECISIONS AND SETTING UP (MADE ONCE EACH CALL TO WATER)
C
C
      TWDT=PMDT
      CWINF=0.0
      TWSFBT=0.0
      TWTTOT=0.
      Y=WTIME(Z)
      TWEVAP=ZEVAP
      TWR=ZRAIN
      TWFRZN=.FALSE.
      IF(XHSOLT(2) .LE. -1) TWFRZN=.TRUE.
      DBGFLG=.FALSE.
      IF(DEBUG1 .LT. 1.E-5) GO TO 10
      IF(MOD(((TIME-TSTART)/PMDT), DEBUG1) .LT. 1.E-5) DBGFLG=.TRUE.
   10 CONTINUE
C  INITIALIZE VOLUMETRIC WATER CONTENT (VWC) AND PRESSURE HEAD
      DO 20 I=1,PMN
      TWTHAA(I)=TWTHB(I)
      TWTHA(I)=TWTHB(I)
   20 TWHA(I)=TWHB(I)
C  TWTHAA HOLDS VWC AT BEGINNING OF PMDT WHICH ISN'T ALWAYS THE SAME
C  AS THE BEGINNING OF TWDT.
C  TWTHA AND TWTHB NOW CONTAIN VWC AT BEGINNING OF TWDT
C  TWHA AND TWHB NOW CONTAIN PRESSURE HEAD AT BEGINNING OF TWDT
C
C
C
C  TRANSPIRATION SECTION
C  TRANSPIRATION IS CVTSPR(FG,L) BY FUNCTIONAL GROUP AND LAYER IN
C  KG HA-1  (1 KG HA-1 IS A LAYER OF WATER 10-5 CM THICK).
C  OLD WAY WAS TOO CRUDE AND THOUGHT IT COULD DECIDE AT BEGINNING OF
C  PMDT HOW MUCH TRANSPIRATION COULD BE FOR EACH LAYER.  NOW DECIDE
C  THIS FOR EACH TWDT
C
      DO30I=1,PMN-2
   30 TWTHTS(I)=0.0
C  SUM WATER DEMAND FOR EACH FUNCTIONAL GROUP.
      DO 40 I=1,6
   40 TWTW(I)=0.0
      DO 50 I=1,6
      DO50J=1,PMN-2
C  TWTW(I) IS WATER DEMAND OF FUNCTIONAL GROUP FOR PMDT, CM
   50 TWTW(I)=TWTW(I)+1.E-5*CVTSPR(I,J)
C
C
C
C  INITIAL STAB AT TWDT
C  IF CONDITIONS ARE ROUGHLY THE SAME AS PAST TIME STEP, KEEP TWDT SAME.
C  IF RAINFALL DURING PMDT IS GREATER THAN 4MM, SAY, MAKE TIME STEP IN
C  WATER  1/48 DAY(1/48=0.02083).
C
C  BEGINNING OF TWDT CALCULATION
   60 CONTINUE
C  CHECK FOR FROZEN SOIL
      IF(.NOT. TWFRZN) GO TO 70
      TWRAIN=.FALSE.
      TWEVAP=0.0
      TWRP=.FALSE.
      GOTO100
   70 IF((TWR+TWSTND).LE.4.)GOTO80
      TWRAIN = .TRUE.
      TWDT=0.02083
      IF(TWRP)GOTO90
      GO TO 120
   80 TWRAIN=.FALSE.
   90 TWDT=TWDTP
C
  100 CONTINUE
C  REFINED TWDT ESTIMATE
C  CHECK IF LARGEST VALUE OF TWDTWP, CHANGE IN THETA LAST TWDT IS
C  BETWEEN 0.3 AND 0.9 TIMES PWTLIM.   IF IT IS ESTIMATE NEXT TWDT FROM
C  IT.
      R=0.3*PWTLIM
      S=0.9*PWTLIM
```

```
      T=0.0
      DO110I=2,PMN-1
      U=ABS(TWDTHP(I))
      IF(U.LT.R)GOTO110
      IF(U.GT.S)GOTO120
      IF(U.GT.T)T=U
  110 CONTINUE
      IF(T.LT.R)T=R
      IF(T.LE.0.0)GOTO120
      TWDT=0.9*TWDT*(PWTLIM/T)
C
  120 IF(TWDT .GT. (PMDT-TWTTOT)) TWDT=PMDT-TWTTOT
C  END INITIAL STAB AT TWDT
C
C
      TWJM = 0
C  TWJM IS A COUNTER WHICH ALLOWS A LIMITED NUMBER OF TWDT
C  HALVINGS
C
C
C
  130 CONTINUE
C  BEGINNING OF LOOP TO CALCULATE PRESSURES ONCE TWDT IS SET
C
C
C  CHECK FOR FROZEN SOIL
      IF(TWFRZN) GO TO 150
C  SURFACE FLUX CONSIDERS RAIN, STANDING WATER AND EVAPORATION (BUT NOT
C  TRANSPIRATION).
C  WATER IN IS PLUS, OUT IS MINUS.  THIS IS POTENTIAL FLUX.
C
C  BEGIN MAIN ITERATION LOOP WITHIN TWDT
C  TWSF NOT TWSFC IS USED FOR SURFACE FLUX IN EQUATIONS SECTION.
C  THE ALTERNATIVE IS TO USE TWSFC AND PASS ON A CORRECTION = TWSF -
C  TWSFC TO NEXT TIME STEP.
C
C  TWR, TWSTND, TWEVAP IN MM.
      TW=AMIN1(TWR,(ZRINT*TWDT*24.))
      TWSF=TW+TWSTND-TWEVAP*TWDT/PMDT
C  BOTH TWSTND AND TWSF ARE IN MM AT THE MOMENT
C  REALLY WANT SURFACE FLUX IN CM FOR TWDT NOT MM
  140 TWSF=TWSF*0.1
      GO TO 160
  150 TWSF=0.0
  160 CONTINUE
C  END INITIALIZATION SECTION
CIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
C
C
C
C
CTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
C  TRANSPIRATION SECTION
C
C  WITHDRAW WATER FROM A LAYER PROPORTIONAL TO WATER AVAILABLE AND
C  FRACTION OF ROOTS IN THAT LAYER.   TWTK IS PROPORTIONALITY CONSTANT
C  FOR FUNCTIONAL GROUP BEING CALCULATED.
C  TWTSPR(I) IS TRANSPIRATION FROM LAYER I DURING TWDT, CM.
C
      R=TWDT/PMDT
      DO170I=1,PMN-2
  170 TWTSPR(I)=0.0
C  OUTER DO LIMIT GOES TO 6 RATHER THAN PMFGPS IN ORDER TO
C  INCLUDE ANNUALS IN PLACES 5, 6 IN ARRAYS CVTSPR AND CVROST
      DO 210 J=1,6
      TWTK=0.0
      DO180I=1,PMN-2
C  TWTA IS WATER AVAILABLE IN LAYER, DIMENSIONLESS
      TWTA(I)=TWTHA(I+1)-PWLLIM
  180 TWTK=TWTK+TWTA(I)*CVROST(J,I)
C  FOLLOWING LINE ADDED SO LACK OF FUNCTIONAL GROUP OR LACK OF ROOTS
C  SOMEWHERE DOESN'T SCREW TRANSPIRATION CALCULATION.
      IF(TWTK .LE. 1.E-8) GO TO 190
      TWTK=R*TWTM(J)/TWTK
  190 CONTINUE
      DO200I=1,PMN-2
  200 TWTSPR(I)=TWTSPR(I)+TWTK*TWTA(I)*CVROST(J,I)
  210 CONTINUE
C
C  FIND LARGEST DELTA THETA THIS WOULD RESULT IN FOR THIS TWDT.
      TW=0.0
      DO230I=1,PMN-2
      S=TWTSPR(I)/CHDX(I)
C  NEVER TAKE ALL THE AVAILABLE WATER.
      IF(S.LE.(TWTA(I)*0.5))GOTO220
      TWTSPR(I)=TWTA(I)*0.5*CHDX(I)
      S=TWTSPR(I)
  220 CONTINUE
      IF(TW.LT.S)TW=S
  230 CONTINUE
```

```
C  IF TW IS TOO LARGE, REDUCE TWDT ACCORDINGLY
      IF((TW.LE.PWTLIM).OR.(TWJM.GE.2).OR.(TWDT.LE.0.02083))GOTO240
      TWJM=TWJM+1
      TWDT=TWDT*0.9*PWTLIM/TW
      IF(TWDT.LE.0.02083)TWDT=0.02083
      GO TO 130
  240 CONTINUE
C
C  END TRANSPIRATION SECTION
CTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
C
C
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  GENERATE VALUES OF CONDUCTIVITY AND SPECIFIC WATER CAPACITY
C  FROM VALUES OF VWC
C
C  IF CONDITIONS ARE ROUGHLY THE SAME THIS TWDT AS LAST,
C  APPROXIMATE CHANGES IN THETA BY ASSUMING FOR NOW THAT CHANGES
C  THIS TWDT EQUAL CHANGES LAST TWDT.   IF CONDITIONS ARE NOT THE SAME
C  ASSUME FOR NOW THAT BEGINNING AND END VALUES ARE THE SAME.
C  GO THROUGH THIS SECTION ONLY ONCE FOR EACH TWDT
      IF(TWONE) GO TO 280
      TWONE=.TRUE.
      IF((TWRP .AND. TWRAIN) .OR. ((.NOT. TWRP) .AND. (.NOT. TWRAIN)))
     1 GO TO 250
      GO TO 280
  250 CONTINUE
      DO270I=2,PMN
C
      S=TWDT/TWDTP
      IF(S.GT.1.)S=SQRT(S)
      TW=TWTHB(I)+S*TWDTHP(I)
      IF(TW .GE. PWT(PWM)) GO TO 260
      IF(TW .LT. PWLLIM) TW=PWLLIM
      TWTHB(I)=TW
      IF(I.GT.2)GOTO270
C  RECALCULATE PRESSURE VALUE OF TOP LAYER
C  PWT(TWJA) AND PWT(TWJA+1) BRACKET TWTHB(I)
      TWJA=TWTHB(I) / PWDELW + 1.
C  TWAA IS FOR INTERPOLATING
      TWAA=(TWTHB(I)-PWT(TWJA))/PWDELW
      TWHB(I)=PWH(TWJA)+(PWH(TWJA+1)-PWH(TWJA))*TWAA
      GO TO 270
  260 TWTHB(I)=PWT(PWM)
      TWHB(I)=PWH(PWM)
  270 CONTINUE
  280 CONTINUE
C
C
      DO 320I=2,PMN-1
      TWTHC =(TWTHA(I)+TWTHB(I)) / 2.
      IF(TWTHC .GT. PWT(PWM)) TWTHC=PWT(PWM)
      IF(TWTHC .LT. PWLLIM) TWTHC=PWLLIM
C  PWT(TWJA) AND PWT(TWJA+1) BRACKET TWTHC
      TWJA=TWTHC/PWDELW + 1.
      IF(TWJA .GE. PWM) TWJA=PWM-1
C  TWAA IS FOR INTERPOLATING
      TWAA=(TWTHC-PWT(TWJA))/PWDELW
C  TWKSA AND TWHHA ARE PART OF AVERAGE CONDUCTIVITY CALCULATION
      TWKSA(I)=PWKSUM(TWJA)+(PWKSUM(TWJA+1)-PWKSUM(TWJA))*TWAA
      TWHHA(I)=PWH(TWJA)+(PWH(TWJA+1)-PWH(TWJA))*TWAA
C  CALCULATE WATER CAPACITY
C  IT IS SIMPLY THE RECIPROCAL OF THE SLOPE OF PRESSURE VS. THETA
      PWHC(I-1) = PWDELW/(PWH(TWJA+1)-PWH(TWJA))
      IF(I.LE.2)GOTO310
      IF(TWJB .EQ. TWJA) GO TO 300
      PWK(I-1)=(TWKSA(I)-TWKSA(I-1))/(TWHHA(I)-TWHHA(I-1))
      GO TO 310
  300 PWK(I-1)=(PWKSUM(TWJA+1)-PWKSUM(TWJA)) / (PWH(TWJA+1)-PWH(TWJA))
  310 TWJB=TWJA
C  TWJB IS USED TO HOLD ON TO THE VALUE OF TWJA FOR ONE MORE PASS
C  THROUGH DO LOOP
  320 CONTINUE
C
C  END CONDUCTIVITY (EXCEPT FOR TOP LAYER) AND WATER CAP. CALCULATIONS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C
CBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
C  BOUNDARY CONDITION SECTION
C
C
C  FOR THE BOTTOM BOUNDARY CONDITION MUST BE ABLE TO SET CONDUCTIVITY IN
C  THE REGION IMMEDIATELY ABOVE THE BOTTOM NODE.  WE MUST ALSO SET
C  PRESSURE HEAD AT BOTTOM NODE.    PRESSURE HEAD WILL BE CALCULATED
C  FOR NODES 2 THROUGH PMN-1
C  PWK(PMN-1) AND TWHB(PMN) WERE READ IN IN WINIT.  THEY REPRESENT
```

```
C  CALICHE LAYER CHARACTERISTICS.
C
C
C  TOP BOUNDARY CONDITION
C  EARLIER IN THIS ROUTINE ACTUAL SURFACE FLUX WAS SET EQUAL TO
C  POTENTIAL EVAPORATION OR RAIN
      IF(TWSF.LT.0.0)GOTO330
C
C  TWSF GREATER THAN OR EQUAL TO ZERO
C  CAN TWSF BE MET WITH MAXIMUM (WETTEST) CONDITIONS?
      TWHB(1)=PWHWET
      TWTHB(1)=PWT(PWM)
      GOTO340
C
  330 CONTINUE
C  TWSF LESS THAN ZERO
C  CAN TWSF BE MET WITH MAXIMUM (DRIEST) CONDITIONS?
      TWHB(1)=PWHDRY
      TWTHB(1)=PWLLIM
C
  340 CONTINUE
C  CALCULATE PWK(1)
      TWTHC=TWTHB(1)
      IF(TWTHC .GT. PWT(PWM)) TWTHC=PWT(PWM)
      IF(TWTHC .LT. PWLLIM) TWTHC=PWLLIM
C  PWT(TWJA) AND PWT(TWJA+1) BRACKET TWTHC
      TWJA=TWTHC/PWDELW + 1.
      IF(TWJA .GE. PWM) TWJA=PWM-1
C  CHECK IF WATER CONTENTS OF FIRST TWO NODES ARE VERY CLOSE
      IF((TWKSA(2) .GE. PWKSUM(TWJA)) .AND. (TWKSA(2) .LE. PWKSUM(TWJA+1
     1 ))) GO TO 350
C  TWAA IS FOR INTERPOLATING
      TWAA=(TWTHC-PWT(TWJA))/PWDELW
      TWKSA(1)=PWKSUM(TWJA)+(PWKSUM(TWJA+1)-PWKSUM(TWJA))*TWAA
      TWHHA(1)=TWHB(1)
      PWK(1)=(TWKSA(2)-TWKSA(1))/(TWHHA(2)-TWHHA(1))
      GO  TO 360
  350       PWK(1) = (PWKSUM(TWJA+1)-PWKSUM(TWJA)) / (PWH(TWJA+1)
     1 -PWH(TWJA))
  360 CONTINUE
      TWSFC=PWK(1)*TWDT*(TWHB(1)-TWHB(2)+9.833E-4*CHDXX(1))/CHDXX(1)
      IF(TWSF.LT.0.0)GOTO370
      IF(TWSFC.GE.TWSF)GOTO390
      GOTO380
  370 IF(TWSFC.LE.TWSF)GOTO390
C
  380 CONTINUE
C  CANNOT MEET TWSF EVEN WITH MAXIMUM SURFACE CONDITIONS
      TWIN=.FALSE.
      TWSFD=TWSF-TWSFC
      TWSF=TWSFC
      GOTO400
C
  390 CONTINUE
C  CAN MEET OR EXCEED DEMAND BUT THERE'S NO POINT FINDING MORE
C  ACCURATE VALUES FOR TWHB(1) OR PWK(1) SINCE NEITHER VEG NOR WATER
C  USE EITHER ONE.
      TWIN=.TRUE.
      TWSFD=0.0
C
  400 CONTINUE
C
C
C  CHECK IF PROFILE FROZEN
C  IF YES, DECREASE CONDUCTIVITY GREATLY AND PREVENT WATER MOVEMENT
C  WITHIN PROFILE.   LEAVE CALICHE LAYER CONDUCTIVITY AS IS.
      IF(.NOT. TWFRZN) GO TO 420
      DO410I=1,PMN-2
  410 PWK(I)=PWK(I)*1.E-6
  420 CONTINUE
C
C
C  END TOP BOUNDARY CONDITION CALCULATION
CBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
C
C
C
C
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
C  NOW, HAUL OUT THE MAIN EQUATIONS, CALCULATE THE COEFFICIENTS AND
C  CALL TDM (THE  TRI-DIAGONAL-MATRIX ROUTINE) TO DETERMINE THE
C  PRESSURE HEAD AT THE PMN-2 NODES.
C
C  I'M AFRAID SOME OF THE NUMBERING IN HERE CAN BE CONFUSING.  THE
C  PRESSURE AT NODES 2 THROUGH PMN-1 ARE DETERMINED BY THE TDM
C  ROUTINE.   AT NODE 1, SURFACE FLUX DETERMINES PRESSURE (TWHB(1))
C  AND AT NODE PMN THE PRESSURE IS A CONSTANT WHICH IS READ IN.
C  THE CONDUCTIVITIES ARE AVERAGES OVER TWDT FOR THE REGIONS BETWEEN THE
C  NODES.   THE FIRST VALUE IS FOR THE REGION BETWEEN NODES 1 AND 2,
C  AND THE LAST VALUE, PWK(PMN-1), IS FOR THE REGION BETWEEN NODES
```

```
C  PMN-1 AND PMN AND IS READ IN AND REPRESENTS THE CONDUCTIVITY OF THE
C  CALICHE LAYER.   THE WATER CAPACITY PWWC(I) IS AN AVERAGE OVER TWDT
C  FOR A REGION CENTERED ON NODE I+1.   VALUES ARE NEEDED FOR NODES
C  2 THROUGH PMN-1.   THE THICKNESS OF A REGION CENTERED ON A NODE I+1
C  IS CHDX(I).   VALUES ARE NEEDED FOR NODES 2 THROUGH PMN-1.
C  THERE IS FURTHER POSSIBILITY FOR CONFUSION BECAUSE EQUATION I IN
C  TDM IS FOR NODE I+1.
C
C  IT HAS BEEN DECIDED (NOT BY ME OBVIOUSLY) THAT THE UNITS FOR
C  PRESSURE IN THIS SUBROUTINE ARE BARS.   THUS, CONDUCTIVITY HAS UNITS
C  OF CM+2 BAR-1 DAY-1.   WATER CAPACITY IS IN BAR-1.  PRESSURE HEADS
C  DUE TO HEIGHT DIFFERENCES MUST BE MULTIPLIED BY 9.833E-4 BAR/CM
C  TO GET BARS.   TO CONVERT TO PRESSURE UNITS IN CM MUST MULTIPLY
C  CONDUCTIVITY BY 9.833E-4 BAR/CM AND GET UNITS OF CM/DAY.   CAN USE
C  PRESSURE DUE TO HEIGHT DIFFERENCES DIRECTLY IN CM.  THE CONVERSION
C  FACTOR IS 9.833E-4 BAR/CM OR 1017 CM/BAR.
C
      DO 430 I=1,PMN-1
      TWWA(I) = PWK(I) / CHDXX(I)
      IF(I .EQ. PMN-1) GO TO 430
      TWWB(I) = (2.*PWWC(I)*CHDX(I)) / TWDT
  430 CONTINUE
C  WITH TWWA AND TWWB GENERATE ABCD'S FOR TDM
      DO 440 I=2, PMN-3
      TWA(I) = TWWA(I+1)
      TWC(I) = TWWA(I)
      TWB(I) = TWWB(I)+TWA(I)+TWC(I)
  440 TWD(I) = TWHA(I+1)*(TWWB(I)-TWA(I)-TWC(I)) + TWHA(I)*TWC(I)
     1 + TWHA(I+2)*TWA(I) + 2.*(PWK(I) - PWK(I+1))  *9.833E-4
     2 -2.*TWTSPR(I)/TWDT
C  CALCULATE FIRST AND LAST VALUES
      TWA(1) = TWWA(2)
      TWC(1) = 0.
      TWB(1) = TWWB(1) + TWA(1)
      TWD(1) = TWHA(2)*(TWWB(1)-TWA(1)) + TWHA(3)*TWA(1)
     1  + TWSF*2./TWDT   -2.*PWK(2)*9.833E-4
     2 -2.*TWTSPR(1)/TWDT
      TWA(PMN-2) = 0.
      TWC(PMN-2) = TWWA(PMN-2)
      TWB(PMN-2) = TWWB(PMN-2) + TWC(PMN-2) +TWWA(PMN-1)
      TWD(PMN-2) = TWHA(PMN-1)*(TWWB(PMN-2)-TWC(PMN-2)-TWWA(PMN-1))
     1 + TWHA(PMN-2)*TWC(PMN-2) + 2.*TWHB(PMN)*TWWA(PMN-1)
     2 +2.*(PWK(PMN-2)-PWK(PMN-1)) *9.833E-4
     3 -2.*TWTSPR(PMN-2)/TWDT
C
C  END EQUATION SECTION
C
C
CEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
C
C
CTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
C  CALL TDM TO CALCULATE PMN-2 PRESSURES FOR NODES 2 THROUGH PMN-1
      CALL TDM(TWA, TWB, TWC, TWD, TWU, PMN-2)
CTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
C
C
C
C
CMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
C  HERE ARE SOME MISCELLANEOUS CHECKS, DECISIONS AND CALCULATIONS
C
C  CALCULATE THETA FOR NODES 2 TO PMN-1 USING PRESSURE VALUES JUST
C  OBTAINED FROM TDM.   USE WATER CAPACITY A LA HANKS.   DELTA THETA=
C  DELTA PRESSURE TIMES WATER CAPACITY.
C
      DO450I=2,PMN-1
      TWTHB(I)=TWTHA(I)+PWWC(I-1)*(TWU(I-1)-TWHA(I))
      IF(TWTHB(I) .GT. PWT(PWM)) TWTHB(I)=PWT(PWM)
      IF(TWTHB(I) .LT. PWLLIM) TWTHB(I)=PWLLIM
  450 CONTINUE
C
C  CALCULATE TWHB FROM TWTHB
      DO460I=2,PMN-1
      TWJA=TWTHB(I)/PWDELW+1.
      IF(TWJA .GE. PWM) TWJA=PWM-1
      TWAA=(TWTHB(I)-PWT(TWJA))/PWDELW
  460 TWHB(I)=PWH(TWJA)+(PWH(TWJA+1) - PWH(TWJA))*TWAA
C
C
C  CHECK IF ANY DELTA THETA'S TOO LARGE.  IF YES, REDUCE TWDT
      DO470I=2,PMN-1
      S=ABS(TWTHB(I)-TWTHA(I))
      IF(S.GT.PWTLIM)GOTO480
  470 CONTINUE
      GO TO 510
  480 IF(TWJM.GE.2)GOTO510
      TWDTQ=TWDT
      TWDT=TWDT*0.9*PWTLIM/S
      TWJM=TWJM+1
C  DON'T LET TWDT GET SMALLER THAN 30 MINUTES
```

```
      IF(TWDT.LT.0.02083)TWDT=0.02083
      R=TWDT/TWDTQ
  490 DO500I=2,PMN-1
      TWHB(I)=TWHA(I)+(TWHB(I)-TWHA(I))*R
  500 TWTHB(I)=TWTHA(I)+(TWTHB(I)-TWTHA(I))*R
      GO TO 130
  510 CONTINUE
C
C
C  CHECK WATER BALANCE
C
      TWSFB=-PWKCAL*TWDT*((TWHB(PMN-1) + TWHA(PMN-1)) * .5
     1 +9.833E-4*CHDXX(PMN-1) - TWHB(PMN)) / CHDXX(PMN-1)
C
C
      CALL WBAL( TWTSPR, TWSF, TWSFB, TWTHA, TWTHB, CHDX, PMN,
     1   TWWIN, TWSTRD, R)
C
      IF(TWDT.LT.0.05)GOTO520
C  WATER IS SOMETIMES NOT CONSERVED
C  WHEN THIS IS THE CASE, HALVE TWDT.
      IF((R .GT. 99.) .OR. (ABS(TWWIN-TWSTRD) .LT. 0.006)
     1 .OR.(ABS(R-1.).LT.0.01))GOTO520
      TWDT=0.5*TWDT
      R=0.5
      GOTO490
  520 CONTINUE
C
CMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
C
C
C
C
CWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
C  THIS IS THE WRAP UP SECTION.  ONCE WE'RE HERE WE EITHER RETURN TO
C  CALLING PROGRAM OR DO ANOTHER TWDT.
C
C
C
C  DEBLGGING
      IF((.NOT. FLGFLG) .OR. (.NOT. DBGFLG))  GO TO 550
      Y=WTIME(Z)
      WRITE(1,530) TWDT,TWR,TWRP,TWRAIN,TWONE,TWIN,TWSTND,TWSF,
     1 TWJM,TWWIN,TWSTRD,R,Y
      WRITE(1,540) TWTHB,TWHB
  530 FORMAT(' W  ',2E9.3,4L1,F5.2,E9.3,I3,'     TWWIN, TWSTRD, RATIO',
     1 3F9.4,' CPU TIME=',F7.3)
  540 FORMAT(' W  TWTHB=', 10F5.3, '      TWHB=', 10F6.2)
  550 CONTINUE
C  END DEBUGGING
C
C  SET SOME VALUES UP FOR NEXT TWDT.
      TWDTP=TWDT
      TWRP=TWRAIN
      TWONE=.FALSE.
      DO 560 I=1,PMN-1
      TWHA(I)=TWHB(I)
      TWDTHP(I)=TWTHB(I)-TWTHA(I)
      TWTHA(I)=TWTHB(I)
  560 CONTINUE
C
C
C  CWINF IS THE NET TOP SURFACE FLUX - DOES NOT INCLUDE TRANSPIRATION
      CWINF=CWINF+TWSF
C
C
C  TWSFBT IS NET BOTTOM SURFACE FLUX FOR PWDT,  OUT IS MINUS
      TWSFBT = TWSFBT + TWSFB
C
C  CHECK FOR FROZEN SOIL
      IF(.NOT. TWFRZN) GO TO 570
      TWSTND=TWSTND+TWR
      TWR=0.0
      GO TO 620
  570 CONTINUE
C
C  ANY STANDING WATER?
      IF(TWIN) GO TO 580
C  TWSFD  = RAIN + STANDING WATER - EVAPORATION - ACTUAL SURFACE FLUX
      TWSTND=TWSFD*10.*PWRNRF**TWDT
      IF (TWSTND .LT. 0.0) TWSTND=0.0
      GO TO 590
  580 TWSTND=0.0
  590 CONTINUE
C
C  DETERMINE AMOUNT OF RAIN REMAINING TO INFILTRATE.
  600 TWR=TWR-ZRINT*TWDT*24.
  610 IF(TWR .LT. 0.0) TWR=0.0
C
  620 CONTINUE
```

```
      XWSTND=TWSTND
C
C  CALCULATE SUM OF TRANSPIRATION, CM
      DO 630 I=1,PMN-2
  630 TWTHTS(I)=TWTHTS(I)+TWTSPR(I)
C
C  RESET TIMING
      TWTTOT=TWTTOT+TWDT
C
      IF(ABS((TWTTOT-PMDT)/PMDT) .GT. 0.002) GO TO 60
C
C  FND WRAP UP SECTION FOR TWDT
CWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
C
C
C
C
CFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
C  THIS IS THE FINAL SECTION
C  DO THE LAST COUPLE OF THINGS NECESSARY BEFORE RETURNING .
C
      DO640I=1,PMN-2
      XWTHTA(I)=TWTHB(I+1)
  640 CWPSI(I)=TWHB(I+1)
C
C
C
C  DEBUGGING
      IF(.NOT. DBGFLG)  GO TO 660
C  CHECK WATER BALANCE FOR THIS PMDT
      CALL WBAL( TWTHTS, CWINF, TWSFBT, TWTHAA, TWTHB, CWDX, PMN,
     1    TWWIN, TWSTRD, R)
      WRITE(1,650) XWTHTA,CWPSI,TWTHTS,CWINF,TWWIN,TWSTRD,R
  650 FORMAT(' WW  XWTHTA=', 10F5.3, '  CWPSI=',10F6.2,/,' WW TWTHTS='
     1 , 10F6.4, '  CWINF=',F8.4,' TWWIN,TWSTRD,RATIO',3F8.4)
  660 CONTINUE
C
C  FND DEBUGGING
C
C
C
C  FND FINAL SECTION
CFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
C
C
C
      RETURN
C
C
CEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
CEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
CEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
C
      ENTRY WINIT
C
CRRPRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
C  THIS ENTRY POINT IS CALLED ONCE FROM MAIN.  VARIABLES PECULIAR TO
C  WATER ARE READ IN HERE AND PRINTED OUT FROM HERE BOTH TO GET THESE
C  DATA LISTED AS PART OF THE PROGRAM AND TO SERVE AS A DATA CHECK.
C  I NEED TO READ: PWDELW,  K-THETA TABLE, H-THETA TABLE, PWKCAL,
C  PWM, TWTHB(10), TWHB(10), PWHDRY, PWHWET,
C  CONSTANT PRESSURE OF CALICHE READ IN AS TWHB(PMN)
C  MUST ALSO READ IN TWSTND, PWTLIM, PWRNRF
C
C  PEAC AND WRITE DATA AND ACCOMPANYING COMMENTS
      READ(5,670)TWCCC
      WRITE(6,680)
      WRITE(6,690)TWCCC
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)PWDELW
      WRITE(6,710)PWDELW
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)PWM
      WRITE(6,720)PWM
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)PWKCAL
      WRITE(6,720)PWKCAL
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
```

```
      READ(5,/)PWKIN
      WRITE(6,720)PWKIN
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)PWHDRY,PWHWET
      WRITE(6,710)PWHDRY,PWHWET
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)PWM
      WRITE(6,730)PWM
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)PWTLIM
      WRITE(6,710)PWTLIM
      WRITE(6,700)
C
      READ(5,670) TWCCC
      WRITE(6,690) TWCCC
      READ(5,/) PWLLIM
      WRITE(6,710) PWLLIM
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)PWRNRF
      WRITE(6,710)PWRNRF
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)TWSTND
      WRITE(6,710)TWSTND
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      READ(5,/)TWTHB
      WRITE(6,710)TWTHB
      WRITE(6,700)
C
  670 FORMAT(20A4)
  680 FORMAT(////)
  690 FORMAT(' ',20A4)
  700 FORMAT(' ')
  710 FORMAT(' ',10F12.5)
  720 FORMAT(' ',10E12.4)
  730 FORMAT(' ', I5)
C  END READING AND WRITING INITIAL DATA FOR WATER
C
CRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
C
C
CIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
C  SOME INITIAL INITIALIZING
C
C  ANOTHER USE OF THIS ENTRY POINT IS TO DO CALCULATIONS WHICH WILL
C  BE DONE ONLY ONCE FOR THE SIMULATION.
C
C  IF TWRP IS TRUE THERE WAS RAIN IN LAST TWDT, THE WATER INTERNAL TIME
C  STEP.    I'M JUST INITIALIZING IT HERE.
      TWRP=.FALSE.
      TWONE=.FALSE.
      TWDT=PWDT
      TWDTP=PWDT
C
C  SET UP DEBUGGING
      Z=0.0
      FLGFLG=.FALSE.
      IF(DEBUG1 .LT. 100.)  GO TO 740
      FLGFLG=.TRUE.
C
      DEBUG1=DEBUG1-100.
  740 CONTINUE
C  LOAD CALICHE LAYER CONDUCTIVITY IN PWK(PWN-1).   THIS IS REGION
C  BETWEEN NODES PWN-1 AND PWN.
      PWK(PWN-1)=PWKCAL
CIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
C
C
CKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKK
C  SECTION TO SET UP CALCULATION FOR AN AVERAGE CONDUCTIVITY
C
C  METHOD IS FROM HANKS AND BOWERS 1962.  HANKS SEES IT AS CALCULATING
C  DIFFUSIVITY FROM CONDUCTIVITY AND THEN GETTING THE AVERAGE
```

```
C  CONDUCTIVITY BACK FROM DIFFUSIVITY.       THIS IS INDEED A "ROUND
C  ABOUT" WAY AS HE PUTS IT.  IF YOU GO THROUGH THE ALGEBRA,
C  THE CONDUCTIVITY USED IS AN AVERAGE WEIGHTED WITH RESPECT TO PRESSURE
C  IN TALKING WITH HANKS ON 6-26-74 HE SAID THIS CONDUCTIVITY
C  CALCULATION IS ONE OF THE MAIN TRICKS OF HIS PROGRAM.  HE TRIED MANY
C  APPROACHES BEFORE HE FOUND THIS ONE WHICH WORKED.  THE PROBLEM SEEMS
C  TO BE THAT CONDUCTIVITY IS EXTREMELY VARIABLE AND AN UNWEIGHTED
C  AVERAGE BASED ON VOLUMETRIC WATER  CONTENT GIVES TOO HIGH A VALUE.
C  IF THE WETTING FRONT IS BETWEEN TWO NODES, E.G., THE CONDUCTIVITY AT
C  THE WET NODE MIGHT BE 1000 TIMES HIGHER THAN AT THE DRY NODE.
C  AN UNWEIGHTED AVERAGE CONDUCTIVITY WOULD BE ABOUT HALF THE WET
C  VALUE BUT A USEFUL AVERAGE CONDUCTIVITY IS CLOSER TO THE DRY VALUE.
C
C  PWT IS AN ARRAY CONTAINING THE VALUES OF THETA
      DO 750 I=1, PWM
      TWI = I-1
  750 PWT(I)=TWI*PWDELW
C
C  CALCULATE INITIAL TWHB VALUES FROM TWTHB VALUES JUST READ IN.
      DO 760 I=1,PMN
      TWJA=TWTHB(I)/PWDELW + 1.
      IF(TWJA .GE. PWM) TWJA=PWM-1
      TWAA=(TWTHB(I)-PWT(TWJA)) / PWDELW
  760 TWHB(I)=PWH(TWJA)+(PWH(TWJA+1)-PWH(TWJA))*TWAA
C
C  CALCULATE SUM OF CONDUCTIVITY TIMES DELTA PRESSURE
      PWKSUM(1)=(PWKIN(1)+PWKIN(2))*(PWH(2)-PWH(1))*0.5
      DO 770 I=2,PWM-1
  770 PWKSUM(I)=(PWKIN(I)+PWKIN(I+1))*(PWH(I+1)-PWH(I))*0.5 +PWKSUM(I-1)
C  THIS GIVES US PWM-1 VALUES FO PWKSUM.  I NEED PWM VALUES.  HOKE UP
C  LAST VALUE BY MAKING DIFFERENCE BETWEEN PWM AND PWM-1 VALUES SAME AS
C  BETWEEN PWM-1 AND PWM-2 VALUES.
      PWKSUM(PWM)=2.*PWKSUM(PWM-1)-PWKSUM(PWM-2)
C
C
      WRITE(6,780)
  780 FORMAT('   PWKSUM,   SUM OF CONDUCTIVITY TIMES DELTA PRESSURE')
      WRITE(6,720) PWKSUM
      WRITE(6,700)
C
      READ(5,670)TWCCC
      WRITE(6,690)TWCCC
      WRITE(6,680)
C
C  END SECTION TO SET UP CONDUCTIVITY CALCULATION
CKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKK
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      RETURN
      END
```

SUBROUTINE WBAL

```
      SUBROUTINE WBAL( TSPR, SF, SFB, THTAI, THTAF, DX, PMN, WIN,
     1    WSTRD, RATIO)
C
C  THIS FUNCTION CHECKS WATER BALANCE OF A TIME PERIOD
C  TSPR IS TRANSPIRATION, CM
C  SF IS TOP SURFACE FLUX (IN IS +)
C  SFB IS BOTTOM SURFACE FLUX (IN IS +)
C  THTAI IS THETA INITIAL , CM/CM
C  THTAF IS THETA FINAL, CM/CM
C  DX IS THICKNESSES OF LAYERS, CM
C  PMN IS NUMBER OF SOIL NODES
C
      DIMENSION TSPR(PMN),THTAI(PMN),THTAF(PMN),DX(PMN)
      A=0.0
      WSTRD=0.0
      DO100I=1,PMN-2
      A=A+TSPR(I)
  100 WSTRD=WSTRD+(THTAF(I+1)-THTAI(I+1))*DX(I)
      WIN = SF - A + SFB
      IF(ABS(WSTRD).LT.1.E-3)RATIO=99.9999
      IF(ABS(WSTRD).GE.1.E-3)RATIO=WIN/WSTRD
      RETURN
      END
```

FUNCTION WTIME

```
      FUNCTION WTIME(Z)                                                 WTM 0100
C             MARCH 1976            PAUL LOMMEN                         WTM 0200
C  THIS FUNCTION SIMPLY DETERMINES CPUTIME  IN SECONDS.   TIME(2) IS AN WTM 0300
C  INTRINSIC FUNCTION.                                                  WTM 0400
C  I HAD TO DO IT THIS WAY BECAUSE  TIME  IS A VARIABLE NAME WE'VE      WTM 0500
C  ALREADY USED AND IS IN THE COMMON BLOCK.                             WTM 0600
C  TIME(2) IS IN 60THS OF A SECOND.  0.016667=1/60                      WTM 0700
      WTIME=0.016667*TIME(2)                                            WTM 0800
      RETURN                                                            WTM 0900
      END                                                               WTM 1000
```

DERIVATION OF DIFFERENCE EQUATIONS

Soil nodes are introduced into the soil profile as reference points for the set of difference equations WATER will generate and then solve. This derivation will proceed very much as the derivation of difference equations for HEAT, since both Equations B-1 and C-1 are diffusion equations.

For a thin layer of soil we rather arbitrarily divide water movement into four components, for convenience:

$$\text{(water in through top)} - \text{(water out through bottom)} - \text{(transpiration)} = \text{(water stored)}. \qquad \text{(C-2)}$$

For layer i:

$$\text{(water in through top)}_i = -K \frac{\partial H}{\partial z}, \qquad \text{(C-3)}$$

where

the right-hand side is evaluated at the top of layer i;

water moving down is a positive flow;

counting of layers starts at soil surface and goes down;

each side of Equation C-3 has units of cm·day^{-1};

K = soil hydraulic conductivity, cm^2·bar^{-1}·day^{-1};

H = soil hydraulic potential head, bars (includes matric potential plus gravitational potential);

z = depth in soil, cm.

Now fill out Equation C-3 as a difference equation:

$$\text{(water in through top)}_i$$

$$= (-K_{i-\frac{1}{2}})\left\{\left[\frac{h_i - h_{i-1}}{\Delta z_{i-\frac{1}{2}}} + \frac{h_i^* - h_{i-1}^*}{\Delta z_{i-\frac{1}{2}}}\right]\left(\frac{1}{2}\right) - \frac{(G)\,(\Delta z_{i-\frac{1}{2}})}{\Delta z_{i-\frac{1}{2}}}\right\};$$

$$= (-K_{i-\frac{1}{2}})\left[\frac{h_i - h_{i-1} + h_i^* - h_{i-1}^* - 2G(\Delta z_{i-\frac{1}{2}})}{2\,\Delta z_{i-\frac{1}{2}}}\right], \qquad \text{(C-4)}$$

where

h_i = soil matric water potential (hereafter SWP) of layer i at beginning of time-step of length Δt days, located at the node in the center of layer i;
h_{i-1} = SWP of layer i—1 at beginning of Δt;
h_i^* = SWP of layer i at end of Δt;
h_{i-1}^* = SWP of layer i—1 at end of Δt;
$K_{i-\frac{1}{2}}$ = hydraulic conductivity of soil in region from center of layer i—1 to center of layer i (i.e., from node i to node i+1, see Fig. C-2);
G = constant to convert gravitational contribution to total soil hydraulic potential heat from cm water potential to bars, = 9.833 x 10^{-4} bar·cm^{-1};
$\Delta z_{i-\frac{1}{2}}$ = distance from node i to node i+1.

Similarly:

$$\text{(water out through bottom)}_i$$

$$= (-K_{i+\frac{1}{2}})\left[\frac{h_{i+1}^* - h_i + h_{i+1} - h_i^* - 2G(\Delta z_{i-\frac{1}{2}})}{2\Delta z_{i+\frac{1}{2}}}\right]. \qquad \text{(C-5)}$$

Transpiration in cm·day^{-1} is

$$\text{(transpiration)}_i = \frac{T_i}{\Delta t}, \qquad \text{(C-6)}$$

where

T_i = transpiration removed from layer i during Δt, centimeters.

Water stored in cm·day^{-1} is

$$\text{(water stored)}_i = \frac{(\Delta\theta_i)\,(\Delta x_i)}{\Delta t};$$

$$= \frac{(h_i - h_i)\,(S_i)\,(\Delta x_i)}{\Delta t}, \qquad \text{(C-7)}$$

where

$\Delta\theta_i$ = change in theta of layer i during Δt, dimensionless (θ_i = cm water in layer i ÷ cm soil in layer i);
Δx_i = thickness of layer i, cm;
S_i = specific water capacity of layer i during Δt, bar^{-1}, $S_i = \Delta\theta_i / \Delta h_i$.

In the same manner as in the HEAT submodel, combine Equations C-4, C-5, C-6 and C-7 with Equation C-2. After several lines of algebra we get

$$-C_i h_{i-1}^* + B_i h_i^* - A_i h_{i+1}^* = D_i, \qquad \text{(C-8)}$$

where

$$C_i = \frac{K_{i-\frac{1}{2}}}{\Delta z_{i-\frac{1}{2}}}; \qquad \text{(C-9)}$$

$$A_i = \frac{K_{i+\frac{1}{2}}}{\Delta z_{i+\frac{1}{2}}}; \qquad \text{(C-10)}$$

$$B_i = A_i + C_i + \frac{2S_i \; \Delta x_i}{\Delta t} \; ; \tag{C-11}$$

$$D_i = h_i \left(\frac{2S_i \; \Delta x_i}{\Delta t} - A_i - C_i\right) + h_{i-1}C_i + h_{i+1}A_i + 2G\left(K_{i-\frac{1}{2}} - K_{i+\frac{1}{2}}\right) - 2\,\frac{T_i}{\Delta t} \; . \tag{C-12}$$

For top and bottom layers, adjustments must be made in Equations C-9 through C-12. The water in through the top of the top layer is the surface flux, as calculated in lines 537 through 594 of WATER, divided by Δ t. Then, A_1, B_1, C_1 and D_1 are determined by going through the same sequence of steps which resulted in Equations C-9 through C-12. For the bottom layer the difference which must be accommodated is that the SWP below the layer is fixed.

Now summarize the equations for A, B, C, D for all layers. It is first convenient to define:

M = number of layers;

$$Y_i = K_{i-\frac{1}{2}} / \Delta z_{i-\frac{1}{2}} \; ; \tag{C-13}$$

$$R_i = 2\,S_i \; \Delta z_i / \Delta t. \tag{C-14}$$

The result is a set of M equations of the form shown in Equation C-8, which form a tridiagonal matrix, where

$$\begin{aligned} A_i &= Y_{i+1} \quad \text{for } i = 1, \ldots, M-1; \\ &= 0 \quad \text{for } i = M; \end{aligned} \tag{C-15}$$

$$\begin{aligned} C_i &= 0 \quad \text{for } i = 1; \\ &= Y_i \quad \text{for } i = 2, \ldots, M; \end{aligned} \tag{C-16}$$

$$\begin{aligned} B_i &= Y_2 + R_1 \quad \text{for } i = 1; \\ &= Y_i + Y_{i+1} + R_i \quad \text{for } i = 2, \ldots, M; \end{aligned} \tag{C-17}$$

$$\begin{aligned} D_i &= h_1(R_1 - Y_2) + h_2Y_2 + \frac{2F}{\Delta t} - 2GK_{\frac{3}{2}} - \frac{2T_i}{\Delta t} \quad \text{for } i = 1; \\ &= h_i(R_i - Y_i - Y_{i+1}) + h_{i-1}\,Y_i + h_{i+1}\,Y_{i+1} + 2G\left(K_{i-\frac{1}{2}} - K_{i+\frac{1}{2}}\right) - \\ &\quad \frac{2T_i}{\Delta t} \quad \text{for } i = 2, \ldots, M-1; \\ &= h_M(R_M - Y_M - Y_{M+1}) + h_{M-1}\,Y_M + h^{**}_{M+1}\,(2\,Y_{M+1}) \\ &\quad + 2\,G\left(K_{M-\frac{1}{2}} - K_{M+\frac{1}{2}}\right) - \frac{2T_M}{\Delta t} \quad \text{for } i = M, \end{aligned} \tag{C-18}$$

where

F = surface flux into top layer during Δ t, cm;
h^{*}_{M+1} = fixed potential of region below bottom layer, bars.

Finally, translate Equations C-15 through C-18 into computer code with the aid of Table C-2.

Table C-2. Transformation of algebraic to FORTRAN variables

Algebraic Variable	FORTRAN Variable (or Constant)
i	I
Δt	TWDT
F	TWSF
G	9.833E-4
M	PMN-2
$K_{i-\frac{1}{2}}$	PWK(I)
h_{i+1}	TWHA(I+2)
h^{*}_{i+1}	TWHB(I+2)
h^{**}_{M+1}	TWHB(PMN)
$\Delta z_{i-\frac{1}{2}}$	CHDXX(I)
Δx_i	CHDX(I)
S_i	PWWC(I)
T_i	TWTSPR(I)
Y_i	TWWA(I) = PWK(I)/CHDXX(I)
R_i	TWWB(I) = 2.*PWWC(I)*CHDX(I)/TWDT
A_i	TWA(I)
B_i	TWB(I)
C_i	TWC(I)
D_i	TWD(I)

Equation C-15 becomes

```
TWA(I) = TWWA(I+1)  for I = 1, . . . , PMN -3;
       = 0          for I = PMN-2.
```

(C-19)

Equation C-16 becomes

```
TWC(I) = 0  for I = 1;
       = TWWA(I) for I = 2, . . . , PMN-2.
```

(C-20)

Equation C-17 becomes

```
TWB(I) = TWWA(2) + TWWB(1) for I = 1;
       = TWWA(I) + TWWA(I+1) + TWWB(I)
         for I = 2, . . . , PMN-2.
```

(C-21)

And finally, Equation C-18 becomes

```
TWD(I) = TWHA(2)*(TWWB(1) - TWWA(2))
         +TWHA(3)*TWWA(2) + 2.*TWSF/TWDT
         -2.*9.833E-4*PWK(2) - 2.*TWTSPR(1)/TWDT
         for I = 1;
```

```
= TWHA (I+1)*(TWWB(I) - TWWA(I) - TWWA(I+1))
  +TWHA(I)*TWWA(I) + TWHA(I+2)*TWWA(I+1)
  +2.*9.833E-4*(PWK(I) - PWK(I+1))
  -2.*TWTSPR(I)/TWDT
  for I = 2, . . . , PMN-3;

= TWHA(PMN-1)*(TWWB(PMN-2) - TWWA(PMN-2) - TWWA(PMN-1))
  +TWHA(PMN-2)* TWWA(PMN-2) + 2.*TWHB(PMN)* TWWA(PMN-1)
  +2.*9.833E-4*(PWK(PMN-2) - PWK(PMN-1) -2.TWTSPR(PMN-2)/TWDT
  for I = PMN-2.                                      (C-22)
```

Equations C-19 through C-22 are the FORTRAN versions of the difference equations used to solve for SWP of the soil layers. They are found in the program in lines 647 through 667 and immediately precede the call to the tridiagonal matrix subroutine which solves them.

LITERATURE CITED

GRIFFIN, R. A., R. J. HANKS, and S. CHILDS. 1974. Model for estimating water, salt and temperature distribution in the soil profile. US/IBP Desert Biome Res. Memo. 74-61. Utah State Univ., Logan. 12 pp.

HANKS, R. J., and S. A. BOWERS. 1962. Numerical solution of the moisture flow equation for infiltration into layered soils. Soil Sci. Soc. Amer. Proc. 26: 530-534.

HANKS, R. J., A. KLUTE, and E. BRESLER. 1969. A numeric method for estimating infiltration, redistribution, drainage, and evaporation of water from soil. Water Resour. Res. 5:1064-1069.

NIMAH, M. N., and R. J. HANKS. 1973. Model for estimating soil water, plant, and atmospheric interrelations. I. Description and sensitivity. Soil Sci. Soc. Amer. Proc. 37:522-527.

RICHTMYER, R. D. 1957. Difference methods for initial value problems. Interscience Publ., New York.

D. SUPPORT PROGRAM TDM

P. W. Lommen

GENERAL DESCRIPTION OF SUBROUTINE TDM

This subroutine solves a set of N linear equations in N unknowns if the coefficients of the unknowns form a tridiagonal matrix. Difference equation approximations to the diffusion equation are typically of the tridiagonal type. The method used here is from Richtmyer (1957, page 103).

Briefly, this routine goes forward through the equations once, eliminating the U(J) for the smallest J at each step. At the N—1 equation, one is left with two equations in two unknowns. Then U(J) is solved for and the routine goes backwards through the equations solving for a value of U at each step.

—A, B, —C are the coefficients of the U's to the right of the main diagonal, on the diagonal and to the left of the diagonal, respectively; the D's are the constant, right-hand sides of the equations. The U's are the solutions. A, B, C, D must be evaluated in the routine which calls TDM.

PROGRAM DESCRIPTION

To illustrate the method of solution, first write out the equations:

$$\begin{aligned} B_1U_1 - A_1U_2 \qquad\qquad &= D_1, \quad \text{(D-1.1)}\\ -C_2U_1 + B_2U_2 - A_2U_3 \qquad &= D_2, \quad \text{(D-1.2)}\\ - C_3U_2 + B_3U_3 - A_3U_4 &= D_3, \quad \text{(D-1.3)}\\ &\;\vdots\\ - C_NU_{N-1} + B_NU_N &= D_N. \quad \text{(D-1.N)} \end{aligned}$$

```
      E(1) = A(1) / B(1)
      F(1) = D(1) / B(1)
```

If Equation D-1.1 is divided by B_1, we get

$$U_1 - (A_1/B_1)U_2 = D_1/B_1, \qquad \text{(D-2)}$$

transposing we get

$$U_1 = (A_1/B_1)U_2 + (D_1/B_1) = E_1U_2 + F_1. \qquad \text{(D-3)}$$

```
      DO 10 I=2, N-1
      DD =   B(I) - C(I)*E(I-1)
      E(I) = A(I) / DD
   10 F(I) = ( D(I) + C(I)*F(I-1) ) / DD
```

To see how these equations came about, substitute Equation D-3 into Equation D-1.2 and solve for U_2:

$$\begin{aligned} -C_2[E_1U_2 + F_1] + B_2U_2 - A_2U_3 &= D_2,\\ (B_2 - C_2E_1)U_2 &= D_2 + A_2U_3 + X_2F_1,\\ \text{or } U_2 &= E_2U_3 + F_2, \qquad \text{(D-4)} \end{aligned}$$

where

$$E_2 = A_2/(B_2 - C_2E_1), \qquad (D\text{-}5)$$

and

$$F_2 = (D_2 + C_2F_1)/(B_2 - C_2E_1). \qquad (D\text{-}6)$$

Continuing in this manner, we get

$$E_i = A_i/DD, \qquad (D\text{-}7)$$

$$F_i = (D_i + C_iF_{i-1})/DD, \qquad (D\text{-}8)$$

where

$$DD = (B_i - C_iE_{i-1}), \qquad (D\text{-}9)$$

and

$$U_i = E_i\,U_{i+1} + F_i. \qquad (D\text{-}10)$$

The quantity DD is not subscripted because it is not needed again.

```
      U(N) = ( C(N)*F(N-1) + D(N) ) / ( B(N) - C(N)*E(N-1) )            TDM 37
```

The last equation generated in the DO loop just discussed is

$$U_{N-1} = E_{N-1}\,U_N + F_{N-1}. \qquad (D\text{-}11)$$

Substitute Equation D-11 into Equation D-1.N:

$$-C_N\,[E_{N-1}\,U_N + F_{N-1}] + B_N\,U_N = D_N,$$
$$U_N = (C_NF_{N-1} + D_N)/(B_N - C_NE_{N-1}), \qquad (D\text{-}12)$$

which is what we have in line 37.

Since all the quantities on the right in Equation D-12 have been evaluated in the calling program or in TDM, U_N is thus determined.

```
      I=N                                                               TDM 38
   20 I=I-1                                                             TDM 39
      U(I) = U(I+1) * E(I) + F(I)                                       TDM 40
      IF ( I .GT. 1 ) GO TO 20                                          TDM 41
```

Once we know U_N from Equation D-12, then from Equation D-11 we get U_{N-1}. Proceeding backwards using Equation D-10 for I = N—2, N—3, . . . , 2, 1, we determine the remainder of the U's.

COMPLETE PROGRAM LISTING

```
      SUBROUTINE TDM(A, B, C, D, U, N)
C
C
C   MARCH     1976    PAUL LOMMEN
C
C   THIS SUBROUTINE SOLVES A SET OF N LINEAR EQUATIONS IN N UNKNOWNS
C   IF THE COEFFICIENTS OF THE UNKNOWNS FORM A TRI-DIAGONAL MATRIX.
C   DIFFERENCE EQUATION APPROXIMATIONS TO THE DIFFUSION EQUATION ARE
C   TYPICALLY OF THE TRI-DIAGONAL TYPE.  THERE ARE SEVERAL MEANS OF
C   SOLVING SUCH A SET OF EQUATIONS.  THE MOST STRAIGHTFORWARD, AND THE
C   ONE THAT I USE HERE AND HANKS USES IN SEVERAL OF HIS MODELS COMES
C   FROM,     ROBERT D. RICHTMYER, 1957, DIFFERENCE METHODS FOR
C   INITIAL-VALUE PROBLEMS, INTERSCIENCE PUBLISHERS, INC., NEW YORK.
C   SEE PAGE 103.  NOTATION USED HERE IS RICHTMYERS'.
C
C   BRIEFLY, THIS ROUTINE GOES FORWARD THROUGH THE EQUATIONS ONCE,
C   ELIMINATING THE U(J) FOR THE SMALLEST J AT EACH STEP.    AT THE
C   N-1  EQUATION ONE IS LEFT WITH 2 EQUATIONS IN 2 UNKNOWNS.  THEN
C   U(N) IS SOLVED FOR AND THEN THE ROUTINE GOES BACKWARDS THROUGH THE
C   EQUATIONS SOLVING FOR A VALUE OF U AT EACH STEP.
C
C   -A, C -C, ARE THE COEFFICIENTS OF THE U'S TO THE RIGHT OF THE MAIN
C   DIAGONAL, ON THE DIAGONAL AND TO THE LEFT OF THE DIAGONAL
C   RESPECTIVELY.   THE D'S ARE THE CONSTANT, RIGHT HAND SIDES OF THE
C   EQUATIONS.   THE U'S ARE WHAT IS SOLVED FOR.    A, B, C, D MUST
C   BE EVALUATED IN THE ROUTINE WHICH CALLS  TDM.
C
C
C
      DIMENSION A(20), B(20), C(20), D(20), E(20), F(20), U(20)
      E(1) = A(1) / B(1)
      F(1) = D(1) / B(1)
      DO 10 I=2, N-1
      DD =   B(I) - C(I)*E(I-1)
      E(I) = A(I) / DD
   10 F(I) = ( D(I) + C(I)*F(I-1) ) / DD
      U(N) = ( C(N)*F(N-1) + D(N) ) / ( B(N) - C(N)*E(N-1) )
      I=N
   20 I=I-1
      U(I) = U(I+1) * E(I) + F(I)
      IF ( I .GT. 1 ) GO TO 20
      RETURN
      END
```

LITERATURE CITED

Richtmyer, R. D. 1957. Difference methods for initial value problems. Interscience Publ., New York.