KANSAS GEOLOGICAL SURVEY
OPEN-FILE REPORT 98-59

A COMPUTER MODEL FOR WATER MANAGEMENT IN THE RATTLESNAKE CREEK BASIN, KANSAS: DEVELOPMENT AND DOCUMENT OF SWATMOD

by

Samuel P. Perkins
Marios Sophocleous

Kansas Geological Survey
1930 Constant Avenue
University of Kansas
Lawrence, KS 66047-3726

Rattlesnake Creek Stream-Aquifer Project

A computer model for water management in the Rattlesnake Creek Basin, Kansas: development and documentation of SWATMOD

Sam P. Perkins[a] and Marios A. Sophocleous[b]

Open-File Report No. 98–59, December 1998
Kansas Geological Survey, The University of Kansas
1930 Constant Ave., Lawrence KS 66047

[a]Research Assistant, perkins@kgs.ukans.edu; [b]Senior Scientist, marios@kgs.ukans.edu

A computer model for water management in the Rattlesnake Creek Basin, Kansas: development and documentation of SWATMOD

Samuel P. Perkins[a] and Marios Sophocleous[b]
[a]Research Assistant, perkins@kgs.ukans.edu; [b]Senior Scientist, marios@kgs.ukans.edu

Open-File Report No. 98–59, December 1998
Kansas Geological Survey, The University of Kansas
1930 Constant Ave., Lawrence KS 66047

Contents

Introduction

The purpose of this report is to give an overview of a watershed modeling program, and provide sufficient documentation to serve as a user's manual. The program, referred to here as SWATMOD, incorporates both Swat (Arnold et al., 1994), a watershed modeling program, and Modflow (McDonald et al., 1988), a stream-aquifer modeling program. The first version of this report was written as Appendix I to the Combined KSU/KGS Final Report (Sophocleous et al., 1997a) to document the model code, referred to here as SWATMOD, with specific reference to a model of the Rattlesnake Creek watershed. This model was developed for the Division of Water Resources (DWR), Kansas Department of Agriculture, as a joint effort of Kansas Geological Survey (KGS) and Kansas State University (KSU). This revised edition, published as KGS Open-File Report 98-59, is intended to be a self-contained document describing SWATMOD and its application to the Rattlesnake Creek Watershed, so that it may serve as a guide or example for similar applications.

The combined SWAT-MODFLOW computer code was applied to models of two other watersheds during its development. Initial tests were based on a model of a small watershed with an area of 1.23 km^2 at Riesel, TX that is gaged and operated by USDA-ARS and identified as Y7. A SWAT-based model of Y7 was used by ARS for SWAT program verification; results from this model are summarized in the SWAT manual (Arnold et al., 1994). Results from our tests of Y7 are described in progress reports to KWO for a study of the Lower Republican River Basin (Sophocleous et al., 1995a and b). The combined SWAT-MODFLOW computer code in a version referred to here as SWATMOD2 was also used to simulate the Lower Republican River Basin in a study for the Kansas Water Office (KWO). The computer code for SWATMOD2 and its application for this study are documented elsewhere (Sophocleous and Perkins, 1997b; Perkins and Sophocleous, 1997 and 1999).

New to this edition is a summary of how surface water diversions were included in the model of Rattlesnake Creek Basin using SWATMOD2. Surface water diversions were modeled using the SURFACE package, one of several packages we have developed or modified to expand MODFLOW's capabilities. Documentation of these packages will be forthcoming in 1999. The most recent application of these packages has been to develop a comprehensive watershed model code for application to Wet Walnut Creek (Sophocleous and Perkins, 1998). In this model code, daily simulations of watershed hydrology are performed by the program POTYLDR (Koelliker, 1994) instead of SWAT.

This report is supplemented by appendices as follows. Appendix A was written as an extension to Chapter 3 of Swat's manual (v. 2), added as Section 3.4, to describe the format for input data files read by Swat, including additional input required for coordinating Swat with Modflow. Appendix B provides operational details of running the combined programs of Swat and Modflow, compiling and linking the programs, and coordinating file device numbers. Appendix C describes input changes made to MODFLOW for its application to the combined watershed-stream-aquifer models. Appendix D contains the source code for Hydbal, Modswb, and Swbavg, the key programs developed for combining Swat and Modflow. Appendix E describes a subroutine written at KGS to calculate potential evaporation based on the Penman method, which was added to Swat and can be called as an alternative to Swat's original options. Appendix F describes two programs written at KGS for use as Modflow postprocessors.

Coordination of Swat and Modflow: program organization

Swat and Modflow are coordinated either by independent execution with data transfer by file, or by linked execution with data transfer by access to common memory. Their interaction is handled by two sets of subroutines developed for this purpose, which are Hydbal, a "package", or set of subroutines called by Swat; and Modswb, a package called by Modflow. A third program, Swbavg, provides a way to combine results from Swat to represent spatial heterogeneities existing within the watershed by calculating areally weighted averages of "hydrologic response units" (HRUs), i.e. the various areal components of the watershed with respect to soil type, land use, and management practices. Spatial heterogeneity with respect to soil type, land use, and management practice are considered.

Swat begins by reading the input files associated with a particular case, and opening those with daily values, i.e., weather data; see the manual on input instructions in

Appendix A for details. If Swat and Modflow are to be run together, then prior to entering the annual simulation loop, Swat makes an initial call to Modflow in order to allocate memory common to both programs.

Swat calculates hydrologic fluxes for each subbasin of the watershed on a daily basis and accumulates them over the aquifer time step, designated to be a day, month, year, or an average of all years of simulation. At the end of each aquifer time step (a day, month, or year), Swat passes a hydrologic summary to subroutine Hydbal, which calculates a mass balance on soil, stream, and aquifer control volumes, and writes a hydrologic summary to a "balance" file to be used as input to either Swbavg or Modswb.

Modflow, coded as subroutine Modflo and called either by Swat or by a small mainline for stand-alone execution, includes the Modswb package, which provides an interface between Swat's lumped parameter model of a watershed and Modflow's distributed parameter model of an aquifer. The Modswb package consists of four subroutines whose names and functions conform to the conventions followed in Modflow. Swb1al allocates memory in Modflow for the watershed-aquifer interface. Swb1rp maps the geographical extent of a watershed's subbasins onto a gridded aquifer domain and the subbasin outflows onto a stream network. Swb1fm converts results obtained from Swat for each subbasin into conditions for Modflow's aquifer solution, including recharge, irrigation, evaporation, and tributary flow to streams. Swb1bd summarizes baseflow and evaporation from the water table for each subbasin. Bothe Swb1fm and Swb1bd make necessary conversions between Swat's hydrologic depths (mm), i.e. flow rates divided by subbasin area, and Modflow's flow rates [L^3/T].

If Swat and Modflow are to be run separately, Modflow calls subroutine Swb1fm in the Modswb package for each time step to read a hydrologic summary from the file written by Hydbal into array SHED; and to set up conditions on which Modflow's solution is based, including recharge, pumping, and lateral inflow to streams.

In a linked execution, Swat calls Modflow through entry points Modflo, to initialize a simulation; Modper, to initialize each stress period, and Modstp, to solve the groundwater equations for a time step. For each time step, Swat calls Modflow according to a procedure that is illustrated below for monthly time steps. Swt2mod is first called to pass the hydrologic summary from array SSUB in Swat to array SHED in Modflow. Modflow is then called via entry point Modstp to solve the aquifer equations, and Mod2swt retrieves baseflow and water table evaporation as hydrologic depths, based on Modflow's solution. Hydbal then writes the summary of hydrologic fluxes to a file, incorporating results from Modflow. The use of Modflow's entry points and Hydbal's subroutines are outlined in the following excerpt of code from file month.h, included in Swat's mainline (SwatMain):

```
write (balttl,'(i4,i3,1x,a)') iyr,mo1, !monthly hydrologic balance:'
dtswat = nd                            !monthly SWAT time step(days)
if (iopswt.gt.0) then                  !does SWAT call MODFLOW?
      delt = dtswat/cnvtim             !convert to MODFLOW time step

                    pass monthly fluxes from SWAT to MODSWB's array SHED:
      call SWT2MOD (lutot,x(LCSHED),ssub)

                    call MODFLOW to execute one solution step, then return:
      call MODSTP (iopswt,mxcurs,icursr,jkkopt, cnvtim,delt,kper,nstp,kstp)

                    retrieve results for MODFLOW solution from array SHED:
      call MOD2SWT (lutot,x(LCSHED),ssub,smm,flu,tir)
end if
              summarize combined results on balance file:
call HYDBAL (iobal,balttl,lu,x(LCSHED),ssub)
```

The remainder of this section provides details on specifying options for the linkage between Swat and Modflow; the data that are passed between the two by file and memory reference; and its implementation.

Input data for the combined Swat-Modflow program (SwatMod)

Most of the input data for the combined Swat-Modflow program are described by the respective manuals for Swat (Arnold et al., 1994) and Modflow (McDonald et al., 1988; Prudic, 1989). The manual for Swat describes input for the program but not the format in which it is read, under the assumption that input files are written by a preprocessor available with Swat that runs under GRASS, a public domain, raster-based graphical interface system (GIS). However, Arc-Info is the GIS used here in Kansas by the parties interested in the combined Swat-Modflow program (KGS, KU, KSU, DWR, and KWO). Because a preprocessor for Swat that runs under Arc-Info was not available, a preliminary version of one has been developed (L. Bian, personal communication), although it does not take into account revisions to input data format for Swat (v. 2).

Revisions to Swat's original input data format include its input control file (~.cod), where options regarding variations on original Swat procedures and coordination of Swat and Modflow are specified; and input files for soils and weather data, whose formats were revised to conform more closely to version 1 of Swat (1993).

For the combined Swat-Modflow program, an extension to the Swat manual has been written to describe Swat's input format, including revisions. This extension, reproduced here as Appendix A (and inserted after Section 3.3 of the Swat manual as Section 3.4) allows input files to be constructed without the aid of a GIS-based preprocessor. For the purpose of data entry, some preprocessing assistance can be provided by spreadsheet templates for exporting text files that conform to Swat input file formats described by Appendix A.

Changes to Modflow input format have been made that allow the program to operate as expected, given standard input as described by Modflow's manual. The Stream package contains some nonstandard input options, although only standard input is used for the Rattlesnake Creek watershed model. The Well package was modified to allow pumping wells to be distinguished from model wells used to represent nonzero flow boundary conditions.

The Modswb package reads an "SWB" input file (see "Input instructions for Modswb") to associate (a) subbasin outflow locations with reaches of a stream network represented by Modflow's Stream package; and (b) the domain of each subbasin with grid cells of the aquifer through an integer array, IBSHED, similar to the IBOUND array read from the input file for Modflow's Basic package.

If Swat and Modflow are executed separately, Modswb also reads a hydrologic summary of results from Swat for each aquifer time step from a balance input file. Alternatively, Swat can call Modflow as a subroutine, and Swat's results are passed via subroutine Hydbal to Modflow by reference to array Shed in Modswb; see overviews for linked Swat-Modflow execution control, Hydbal, and Modswb.

Structure of balance file for Swat-Modflow data transfer

A hydrologic summary is written by a call from Swat to subroutine Hydbal for each aquifer time step. The data contained in this summary are either read by Modswb in a subsequent run of Modflow (**iopswt** = 0) or, in the case of a linked Swat-MODFLOW simulation (**iopswt** > 0), copied directly between Swat and Modflow arrays. An abbreviated version of the example file is the following.

```
365. (days)   1977 annual hydrologic balance:                                        1
 id So Va Aq St To Sb Bas Sh Component   Mean StdDev       1       2       3         2
       2569.600 km**2 area  fractions:                0.22015 0.04909 0.20435         3
 ponds (areal fractions):                0.01994      0.03430 0.00310 0.00640         4
 active gridded fraction:                0.12599      0.07325 0.20532 0.23676         5
  1  1  0  0  0  1  1   1 10   PREC MM  993.4  107.5 1035.10  895.30 1113.80
  2  1  0 -1  0  0 22   0 11    IRR MM  136.1    6.1  137.52  133.12  127.01
  3 -1  0  0  0 -1 12   7 12     ET MM  868.9   24.9  885.77  834.88  859.75
  4 -1  0  0  1  0  4   3 13   SURQ MM  100.8   39.6  124.66   63.53  152.81
  5  0  0  1  0  1 13  38 14  TLOSS MM    3.8    1.7    3.29    5.23    6.80
  6 -1  0  0  c  0  5   4 15   LATQ MM    0.3    0.1    0.38    0.15    0.29
  7 -1  1  0  0  0 11   5 16   PERC MM  192.9   53.8  192.70  166.60  265.51
  8  0 -1  1  0  0  9 107 17   GWRE MM  192.9   53.8  192.70  166.60  265.51
  9  0  0 -1  0 -1  7 105 18    REV MM    6.1    6.3    9.42   22.36    8.70
 10  0  0 -1  1  0  6 104 19   GW Q MM    5.1    9.6    6.79   27.36   12.86
 11  0  0  1  0  1 16  20 20   PSEP MM    2.5    2.2    4.67    0.87    1.18
Balances:
  1                         soil        -33.5    2.5  -30.90  -36.74  -37.56
  2                         vadose        0.0    0.0    0.00    0.00    0.00
  3                         aquifer      49.3   54.0   42.27  -11.01  123.73
  4                         stream      104.3   43.3  127.54   90.84  164.99
  5                         combined    124.7   94.0  147.87   44.16  253.33
```

Contents of file records 1-5 in this example, numbered at the right, are described as follows; numbers in parentheses refer to vectors in the SHED array (below).

Record contents:

1 length of time step (days), calendar year, and the phrase "hydrologic balance", used by Modswb to identify the beginning of the summary;
2 headings that apply to columns for the hydrologic components that follow line 5.
3 total basin area and the areal fraction for each subbasin (1);
4 noncontributing area fraction for ponds: basinwide average, and value for each subbasin (8);
5 fraction of area corresponding to active nodes of aquifer for basin and each subbasin (3); in Swat's stand-alone mode (**iopswt** = 0), these appear as zero, since they are defined in Modflow.

The columns in the table of hydrologic components following line 5 are identified by the headings in line 2. In the middle of the table are the names of the hydrologic components to be passed between Swat and Modflow. To the right of the component names lie columns for mean and standard deviation values for the components taken over all (nine) subbasins; to the right of these lie the component values for the first three of nine subbasins in this example. To the left of the component names, column **Sh** identifies vectors 10-20 of the SHED array, defined below. Columns **Sb** and **Bas** identify the vectors for Swat's subbasin and basin arrays, respectively, that contain these components. At the left are vector columns corresponding to soil (**So**), vadose (**Va**), aquifer (**Aq**), stream (**St**), and total (**To**) control volumes that indicate how the components add into a mass balance for each (but subject to definitions of recharge and tributary flow reflected by option **jkkopt** for Modswb, below). For each of these control volumes, mass conservation is expressed by $ds = \Sigma q_i$, i.e., the time rate of change of mass per unit area within the control volume equals the net flux into the control volume by the components q_i. The balances are evaluated according to the following equations by subroutine Hydbal:

soil: *ds*(1) = prec(10) + irr(11) – et(12) – surq(13) – latq(15)– perc(16);
vadose: *ds*(2) = perc(16) – rech(17);
aquifer: *ds*(3) = –irr(11) + tloss(14) + perc(16) – rev(18) – gw_q(19) + psep(20);
stream: *ds*(4) = *c*·[surq(13) + latq(15)] + gw_q(19);
total: *ds*(5) = prec(10) – et(12) - rev(18),
where *c* = 1 - pond fraction (8). Abbreviations and indices for the terms in these equations correspond to the components shown in the balance file (above) and array SHED definitions (below).

Discussion of mass balance terms

The above equations indicate that of the total runoff (13) and subsurface flow (15) leaving the soil control volume, only their contributing fraction are shown entering the stream control volume; a balance on ponds might be useful in showing the fate of their noncontributing fraction.

A distinction is drawn between definitions for recharge in Swat and Modflow. Swat's definition of recharge (17) is restricted to soil water that has dropped through both the soil profile and the vadose zone, and is represented by Swat as delayed percolation (16) from the soil profile. This definition is narrower than that used in Modswb to calculate recharge for Modflow, which includes transmission losses, percolation and pond seepage, subject to added Swat option **jkkopt**, described below. Modflow's version of the recharge flux is distinguished from Swat's definition as SHED vector 25.

Mass balance terms affected by the Swat-Modflow linkage are handled through subroutines in the Modswb package as follows.

1. Swat and Modflow are coordinated to represent annual water use for irrigation through input options **ioplim** for Swat (input file ~.cod) and irropt for Modflow (input file ~.swb). If Swat runs without calling Modflow but option ioplim>0, Swat reads a maximum annual irrigation depth from the ~.cod input file. Then for each aquifer time step, Swat calculates an irrigation depth (11) subject not only to its own scheduling (~.mgt) and plant stress-based (~.mco) options, but also, for ioplim>0, to specified annual use limits. In the case that Swat calls Modflow (iopswt>0) and ioplim>0, Swat obtains annual irrigation limits from Modflow as follows. Swat calls Modflow's entry point Modper at the beginning of each stress period (year), and Modswb subroutine Swb1rp calculates irrigation depth (9) for each subbasin, given by total reported use according to the Well package input file, divided by subbasin area. Swat then retrieves these irrigation depths (subroutine Passflx) to limit annual irrigation as described above.

2. For each aquifer time step, Swb1fm sets up the conditions for Modflow's solution that are based on the hydrologic components calculated by Swat. Irrigation depth (11),

subject to limits described above, are treated as follows. If option irropt=0 (specified by Modswb's input file), irrigation pumping is given as read from the Well input file, and Swat's irrigation depth is ignored. Otherwise (irropt>0), the pumping rate of each well within a subbasin is scaled by the ratio of depths given by SHED vectors (11) and (9), so that total groundwater pumping in each subbasin corresponds to that assigned by Swat. The irrigation depth calculated by Swat is part of Swat's hydrologic summary that is obtained by Modflow as described above.

3. Components of Swat's hydrologic summary are combined to represent tributary inflow (22) and groundwater recharge (25) for Modflow's stream-aquifer model as described below, using definitions that are somewhat specific to the Rattlesnake Creek watershed model, and which are invoked by setting option **jkkopt** > 0.

4. Swb1bd calculates the hydrologic components revap(18) and baseflow (19) that depend on Modflow's aquifer head solution. Revap, in Swat's terminology, is based on evapotranspiration from the water table calculated by the Evt package, and baseflow is based on the negative of streambed leakage calculated by the Stream package. For a combined Swat-Modflow run, these components as calculated by Modflow appear in the balance file; otherwise, Swat's version of these components are included, which are based on a lumped model of groundwater with relatively specific assumptions and application (Arnold, 1994).

Definition of array SHED for Swat-Modflow data transfer

Modflow array SHED stores data related to subbasins; vectors 6 and 27–30 are not currently used. Dimensions for the Rattlesnake Creek aquifer model are in feet [L] and days [T], so flow rates are expressed in ft^3/day, and velocities (e.g. hydraulic conductivity and evaporation rates) in ft/day. Swat's hydrologic summary is expressed as a depth (mm) accumulated over the number of days in the aquifer time step, dtswat. The hydrologic summary in vectors 10-20 of array SHED is obtained either directly from arrays in Swat (through Hydbal subroutine Swt2mod) in the case of a linked Swat-Modflow run (**iopswt** > 0), or from a balance file written by Swat and read by Modswb subroutine Swb1fm.

(1) **Areal frc** areal fraction of subbasin; read in SW1RP.
(2) area[L^2] area of subbasin [L^2]: product of bsarea and (1); SWB1RP.
(3) **GrdArea[L^2]** gridded subbasin area; see also (7); SWB1RP.
(4) **Irr_use[L]** irrigation [L^3/T]: total irrigation pumping accumulated for each subbasin from WELL module input at beginning of each stress period; SWB1RP.
(5) **IrrSwatFrc** not used; previously, ratio of irrigation fluxes (11)/(9); see irropt.
(7) **AreGrd/Act** gridded active subbasin area as fraction of actual subbasin area, (3)/(2); SWB1RP; passed back to SWAT to be written in the balance file (~.bal) by HYDBAL in a combined SWAT-MODFLOW run (iopswt>0).

(8) **Pond fract** noncontributing areal fraction of subbasin, which contributes instead to ponds; read from ~.balance file; SWB1FM.
(9) **Irr_use** irrigation flux based on total pumping in subbasin for the stress period, (4); SWB1RP.
(10) Prec mm precipitation
(11)Irr mm irrigation [L] according to SWAT; used to scale pumping if irropt>0.
(12)ET mm evapotranspiration from ground surface [L]; see also (21)
(13)SURQ mm surface runoff [L] to tributaries leaving subbasin
(14)TLoss mm transmission loss [L]
(15)LATQ mm lateral flow [L] to tributaries leaving subbasin
(16)PERC mm percolation from the root zone
(17)GWRE mm groundwater recharge. According to SWAT, this consists only of percolation from the root zone, with a possible delay time set by SWAT input. For zero time delay (i.e., no storage in the vadose zone), recharge equals percolation from the root zone. For MODFLOW definition of recharge, see (25).
(18)GW ET mm evaporation from water table ("revap" in Swat); the value from SWAT is superseded by results from MODFLOW's EVT package, summarized in SWB1BD.
(19)Baseflow mm baseflow; the value from SWAT is superseded by the negative of streambed leakage, calculated by MODFLOW's STREAM package and summarized for watershed subbasins in SWB1BD.
(20)PndSeep mm pond seepage into aquifer
(21) **POT ET mm** potential et; used to define maximum evaporation rate from water table if EVT is invoked; see options iopet and ievopt.
(22) **Tribflow mm** tributary inflow to streams from subbasins given by the contributing sum of surface and subsurface runoff. Defining $c = 1$ - pond fraction(8),

Qtrib(22) $=c$*[SURQ(13) + LATQ(15)]

unless **jkkopt** > 0; SWB1FM.
(23) **DTW [L]** avg depth to water for shallow nodes if EVT is invoked; SWB1BD.
(24) **Pump rate mm** irrigation depth (mm) calculated in SWB1FM as a check: if irropt>0, it should equal assigned irrigation (11) from SWAT; otherwise, it should equal irrigation use (9), calculated in SWB1RP from MODFLOW's ~.WEL input.
(25) **Recharge mm** groundwater recharge flux is defined for MODFLOW by the sum

RchMod(25) =xmloss(14) +perc(16) +pond seepage(20)

unless **jkkopt** > 0; SWB1FM.
(26) **RchInp mm** recharge depth (mm) corresponding to initial array RECH as read from Modflow input file if ioprch > 0; flux is for initial time step duration, **delt0**, and with respect to area of active nodes in subbasin; SWB1FM.

Summary of Swat-Modflow options used for the Rattlesnake model

Options added to Swat

Several options have been added to Swat, both to coordinate Swat with Modflow and to revise Swat's watershed model. These options are read from Swat's options input file (Appendix A). Those used for the Rattlesnake Creek watershed model are defined below, keyed to their order in the input control file description. They include Swat-Modflow coordination options **iopmod**, **iopswt**, **ioplim**, and **jkkopt**; and Swat model revision options **iopwfl**, **iopet**, and **iopwea**:

iopmod (2): option for Swat to summarize hydrologic results on an annual(1), monthly(2) or daily(3) basis for each subbasin and call Hydbal to write results to a "balance" file named below (nambal, rec. 4) that can be read by Modflow on a subsequent run; see **iopswt**.

iopswt (3): This is a Swat option (read from Swat's ~.cod input file) that defines how Swat and Modflow are to be coordinated. A zero value signifies separate execution for both Swat and Modflow, in which case Swat calls Hydbal to write the balance file without input from Modflow; on a subsequent run of Modflow, the balance file is read by Swb1fm in Modswb. Otherwise (iopswt > 0), Swat and Modflow are coordinated in the following manner.

iopwfl (7): option (y=1,n=0) to read daily precipitation data for all stations from the same file, and to do similarly with temperature data. Weather stations are identified in the ~.cio file by an integer, *i*, from 1 to n corresponding to the order of data columns from left to right. Data format for temperature and precipitation data are defined in records 5-6, below.

ioplim (10): option (y>0,n=0) to limit daily irrigation pumping by wsfdmx (mm/day) and annual irrigation pumping by groundwater appropriation pmpmax (mm/yr), reduced by pumping efficiency pmpeff; wsfdmx, pmpmax and pmpeff are defined in records 8-9, below, subject to the following conditions. If Swat does not call Modflow (specified by **iopswt=0** and **ioplim**>0), annual irrigation limits are read from record 9 (below). If Swat calls Modflow (specified by **iopswt**>0 and ioplim>0), Swat calls subr Passflx to obtain annual maximum pumping flux rates (mm/yr) for each subbasin, based on Modflow's ~.WEL input file.

iopet (15): option (y=1,n=0) to use the reference crop evaporation calculated by subr Penman instead of the value calculated by Swat's subroutine EVAP8, the default option. Subroutine Penman was written and linked with Swat by SPP, and calculates reference crop evaporation according to the method recommended in the Handbook of Hydrology (Maidment, ed., 1993). This method includes the effect of long-wave emission in calculating net radiation, in contrast to Swat's subroutine; see also the following related option, **iopwea**.

iopwea (16): option (y=1,n=0 to read, rather than synthesize, daily values for insolation (ly/d), relative humidity (pct) and wind speed (m/s at 10 m above ground surface) in that order from an input file with name namrad (rec. 4) in format fmtrad (rec.7). The default option (**iopwea=0**) is to generate them as in the standard Swat program.

jkkopt (19): an option passed to Swb1fm, where it determines how terms from Swat's hydrologic summary are combined to define groundwater recharge (25) and tributary flow (22) for Modflow. The default Swat option for these combinations corresponds to **jkkopt** = 0; the exception, **jkkopt** > 0, is a special case formulated for the Rattlesnake Creek watershed model by Prof. Jim Koelliker, Kansas State University. Defining c = 1 - pond fraction(8), these definitions are as follows.

tributary inflow (22)	$= c\cdot$[surq(13) + latq(15)],	default
	$= -(1/c)\cdot$xmloss(14) $+c\cdot$latq(15),	jkkopt > 0
Recharge (25)	= xmloss(14)+perc(16)+pond seepage(20),	(default)
	= xmloss(14)+$c\cdot$latq(15)+perc(16)+pond seepage(20),	jkkopt > 0

Options for the MODSWB package

Options **irropt**, **ioprch**, **ievopt**, and **jkkopt** affect the conditions of the groundwater solution in Modflow, which are set up in Swb1fm. The first three options are read by SWB1AL from the SWB input file. Option **jkkopt** is passed from Swat to Modflow either by way of the balance file written by Hydbal or by subroutine argument list, depending on option **iopswt**, described below. Option **iopmod** specifies the length of an aquifer time step as a day, month, or year. Options **jkkopt**, **iopmod**, and **iopswt** are three of several options read by Swat and summarized below.

irropt: option (y=1,n=0) to scale the pumping rates of irrigation wells read from MODFLOW's ~.WEL input for a stress period so that subbasin j's total pumping flow rate varies according to the flux specified by SWAT for each time step; otherwise (irropt=0), use the rate specified in Modflow's well input file ~.wel (SWB1FM; shed vectors 4,9,11,24; well vectors 4,5).

ioprch: option (y=1,n=0) to maintain the recharge distribution within each subbasin according to the recharge input file array RECH, but scale the recharge within each subbasin for each time step according to the sum of fluxes to the aquifer given by shed vector 25. Note: option jkkopt determines which components are included in this sum. (SWB1FM: shed vectors 25 and 26)

ievopt: option (y=1,n=0), applicable if Modflow's EVT package is invoked: if **ievopt** > 0, evaporation from the water table is controlled by potential evaporation at ground surface as calculated by Swat. However, it is coded below using Swat's actual evaporation, shed (12); potential evaporation will have to be entered into SHED(21), which is not currently being done. The next version of Swat-Modflow, i.e. SwatMod96, is a little further along than this version representing evapotranspiration from the water table. Otherwise (**ievopt** < or = 0), evaporation at ground surface is specified by standard Modflow input to array evtr in the Evt module.

Flow of execution for Swat and Modflow

Linked Swat-Modflow execution

Options **iopswt** and **iopmod**, read from Swat's input control file (~.cod), affect execution control as follows. For each year of a simulation, Swat simulates watershed hydrology on a daily basis, and summarizes its results at the end of each aquifer time step (a day, month, or year, depending on iopmod). If *iopswt*>0, Swat calls Hydbal package subroutines Passflx, Swt2mod, and Mod2swt to transfer data directly between Swat and Modflow arrays; and Swat calls Modflow through entry points Modflo, Modper, Modstp, and Modend as shown in the program outline below. Option **iopswt** is passed to Modflow through these entry points to return execution control from Modflow back to Swat just before the next entry point is encountered, as shown below for the modified Modflow program structure. If **iopswt=0**, Swat bypasses calls to Modflow, and calls subroutine Hydbal at the end of each time step to write a hydrologic summary that can be read in a subsequent, separate run of Modflow.

Linked Swat-Modflow execution:

```
begin
  Initialize simulation;
  if (iopswt > 0) call Modflo to initialize the aquifer model;
  for each year:
    if (iopswt > 0) then at the beginning of each year's simulation:
      call Modper to begin an aquifer stress period;
      call Passflx to retrieve annual groundwater use for each subbasin to provide annual
      limits on Swat's automatic use of irrigation.
    end if
    Simulate watershed hydrology on a daily basis;
    for each aquifer time step (a day, month, or year, based on iopmod), do the following:
      if (iopswt > 0) then coordinate with Modflow as follows:
        call Swt2Mod: copy the hydrologic summary into array SHED for Modswb;
        call Modstp: solve hydraulic head distribution for the time step;
        call Mod2Swt: retrieve Modflow results (baseflow and, optionally, evaporation
          from water table);
      end if
    call Hydbal to calculate a mass balance and write the hydrologic summary to the
      balance file, including components calculated by Modflow if **iopswt** > 0);
  end of year;
  if (**iopswt** > 0) call Modend to close files used in Modflow;
end.
```

<u>Modified Modflow program structure</u>

The modified structure of Modflow is summarized by the outline shown below, which corresponds to the flowchart in Fig. 13 of Modflow's manual (McDonald et al., 1988). Calls to subroutines in the Modswb package, added to this structure to coordinate Modflow and a watershed model, are identified by bold type. The Modswb package is invoked by setting **iunit**(6) > 0 in Modflow's basic package input; see Modflow manual, p. 4.9. Modflow may be run either as a stand-alone program or from a daily watershed simulation program (Swat), using the option **iopswt**, described above. To allow Modflow to be called from Swat via entry points Modflo, Modper, and Modstp, Modflow's looping structures on stress periods and time steps have been converted from **do** loops to a more primitive but functionally equivalent form using conditional jumps with counters, so as to circumvent Fortran compilers' normally justified prohibition of jumps into a **do** loop.

Flow of execution for modified Modflow program structure:

```
begin
  subroutine **Modflo** (main entry point)
    call Bas1df to define grid size (rows, columns, layers), stress periods, options;
    allocate array space (AL routines);
    call **Swb1al** (after Str1al) to allocate arrays for watershed-aquifer linkage;
    read and prepare data that remain constant throughout simulation (RP routines);
    **if (iopswt > 0) return**
  for each stress period:
    **entry Modper:**
      define stress period length and divide into time steps;
      read and prepare data that remain in effect for the stress period (RP routines);
      call **Swb1rp** (after Ghb1rp) for watershed-aquifer linkage and pumping;
      **if (iopswt > 0) return**
    for each time step:
      **entry Modstp:**
        call **Swb1fm** (before Bas1ad) to set up conditions based on Swat's hydrologic
          summary;
        call Bas1ad to advance time and initial hydraulic heads; calculate time step;
        for each iteration on approximate solution:
        formulate finite difference equations: calculate coefficients (FM routines);
        approximate solution;
        end iteration on approximate solution;
        calculate budget terms for mass balance (BD routines);
        call Swb1bd (after Str1bd) to summarize watershed terms;
        call Bas1ot to optionally save and print results;
        if (iopswt > 0) return
    end time step;
  end stress period;
  entry Modend
    close files;
end.
```

Stand-alone Modflow execution

In stand-alone mode, calling program ModMain, shown below, calls subroutine Modflo, passing **iopswt**=0 so that the complete structure of Modflow shown above is executed before returning to ModMain.

```
  program modmain
  data iopswt/0/      ! set option to run modflow as stand-alone program.
  data cnvtim /1./    ! time unit conversion factor (for calling MODFLOW from SWAT)
    call MODFLO (iopswt, mxcurs, icursr, jkkopt, cnvtim, delt, kper, nstp, kstp)
  stop
end
```

Definition of argument list passed to subroutine Modflo

In the case of stand-alone execution (**iopswt**=0, above), only arguments **iopswt** and **cnvtim** need be defined by the mainline, as shown; the remaining arguments are defined within the call to subroutine Modflo. For combined Swat-Modflow execution (**iopswt**>0), the argument lists for Modflo, Modper, and Modstp are the same, and are defined as follows.

mxcurs: max. number of pointers (more specifically, indexes that point) to beginning locations of arrays within the x array; specifies dimension for array **icursr**.

icursr(mxcurs): array of indices to arrays within Modflow's "x" array, and calculated in Modflow's memory allocation routines. This allows Swat (or other calling programs) to reference any of the arrays allocated space within the x-array. Note: **intcrs** (10) is an internal version of array **icursr** that is equivalenced to the x-array indices for convenient copying to **icursr**.

jkkopt: option for defining recharge and tributary flow (see "Definition of options used in the MODSWB package"); **jkkopt** is either read from balance file if **iopswt**=0 or passed from SWAT if **iopswt**=1.

cnvtim: conversion from Swat time units (days) to Modflow time units [Tmd]; given by **cnvtim = tsec(itmuni)**/86400 [days/Tmd]; **tsec** and **itmuni** are defined below. **cnvtim** is calculated in MODSWB subroutine SWB1RP and, if iopswt=1, is passed back to Swat via MODFLO's argument list on Swat's initial call to MODFLO.

itmuni: time unit option (0-5); specified in MODFLOW's Basic Package input file.

tsec: time unit length (sec) corresponding to itmuni as follows:

itmuni	tsec(*itmuni*)	time unit
0:	1	undefined
1:	1	second
2:	60	minute
3:	3600	hour
4:	86400	day
5:	31536000	year (365 days)

delt: time step [T]; not used until after the MODSTP entry point, but the argument list of the first entry point should contain all arguments of ensuing entry points. For the stand-alone option (*iopswt*=0), time step and multiplier are used as specified in MODFLOW's basic package input file, i.e. standard input is default.

kper: index to current stress period;

nstp: number of time steps for current stress period, kper;

kstp: index to current time step.

Hydbal: summarizing Swat's hydrologic results for Modflow

The Hydbal package includes the subroutines PASSFLX, SWT2MOD, and Mod2swt, which are called by SWAT only for a linked SWAT-MODFLOW run, and Hydbal, which is called whether or not Swat calls Modflow. The Hydbal subroutines are described in the context of their use by Swat.

Passflx: get annual irrigation limits from Modflow for simulated irrigation in Swat

For each year, prior to beginning daily simulation, Swat first calls Modflow's entry point Modper to define a stress period corresponding to a calendar year, and summarize the year's estimated irrigation use as a depth, based on input to Modflow's Well package. Irrigation depth, d, is calculated for each subbasin, j, by $d = cQ_j/\Delta t/A_j$, where Q_j = total pumping flow rate, A_j = subbasin area, and c = unit conversion factor for length and time. Swat then calls Passflx to retrieve this estimated irrigation use, which can be interpreted as an annual constraint on irrigation by a modified version of Swat's automatic irrigation procedure. If Swat and Modflow are executed independently, these annual irrigation depths can be read from Swat's ~.cod input file.

Swt2mod: copy Swat results to Modswb's SHED array

After simulating an aquifer time step (day, month, or year, based on option iopmod, ~.cod input file) for a combined Swat-Modflow run, Swat calls Swt2mod just prior to calling Modflow's entry point Modstp to pass a hydrologic summary calculated by Swat, including surface runoff, lateral flow, percolation from root zone, irrigation and surface evaporation, to Modflow's SHED array. If Swat and Modflow are run separately, Modflow obtains the hydrologic summary by reading the balance file written by Hydbal.

Mod2swt: retrieve results from Modswb's SHED array for Swat

After calling Modstp, Swat calls Mod2swt to retrieve a hydrologic summary of baseflow and groundwater evapotranspiration calculated by Modflow from the Shed array, to replace the values for these terms calculated by Swat in the balance file written by Hydbal. If Swat does not call Modflow, the balance file includes the values calculated by Swat for these terms.

Hydbal: summarize Swat or combined Swat results

After simulating the days of an aquifer time step (and calling Modflow if **iopswt** > 0), Swat calls Hydbal to calculate mass balances for the control volumes of interest and (2) write the hydrologic summary and mass balance to the balance file. The mass balances for soil, vadose zone, aquifer, stream, and total control volumes provide a check on mass conservation in the combined Swat-Modflow linkage. Specifics regarding the contents and format of the balance file are described for Modswb; see "Input requirements for Modswb."

Program Swbavg: representing heterogeneity within subbasins

Program Swbavg was written to represent spatial heterogeneities existing within subbasins of a watershed by calculating areally weighted averages of "hydrologic response units" (HRUs), i.e., the various soil types, land uses, and management practices existing within each subbasin, in order to produce composite results from Swat to be used as input for Modflow. For each subbasin and time step of the simulation, the areal fractions corresponding to its HRUs, i.e., components of the watershed's spatial heterogeneity, are calculated. These areal fractions represent weight functions that are distinct for each subbasin, and are used to calculate a weighted average over the HRUs for each hydrologic component (SHED vectors 10-20 in the balance file). For each HRU, or component of the watershed's heterogeneity, a separate case of the watershed model is simulated by Swat, for which all subbasins of the watershed are assumed to be homogeneous in the characteristics of that HRU.

Program Swbavg is illustrated by file swbavg.inp for the Rattlesnake Creek watershed. In this example, each HRU corresponds to one of 15 combinations of five soil types and three land use-management schemes used to represent The five soil types are represented by Harney, Pratt, Tivoli, Naron, and Carwile. Land use and management practices are represented by three schemes: wheat, sorghum, and fallow rotations with no irrigation ("wsf"); irrigated corn ("irc"); and range and pasture without irrigation ("r/p"). The initial table of weight functions used to average the HRUs are shown in the table below: the 29 rows correspond to subbasins, and the columns from left to right correspond to 15 HRUs associated with the order in which names of the balance files are read from the file.

Whereas areal fractions of soil types within each subbasin are naturally fixed with respect to time, their land use areal fractions vary from year to year. To represent this land use change, the initial table of weight functions is updated every five years.

Input data requirements for Swbavg are described below, and are summarized as follows. Line 1 shows the name of the hydrologic balance file that is to be written by Swbavg containing the averaged results, and which can be read in a subsequent run of Modflow. In addition to this file, Swbavg writes a summary file containing an areally weighted average over all subbasins for each hydrologic component. Line 2 of the file lists the years in which the set of weight functions is to be updated, primarily reflecting the change in land use over time. Following line 2 are the names of fifteen hydrologic balance files previously written by Swat that represent the HRUs to be averaged. This list is followed by an initial set of weight functions for each of 29 subbasins, used to calculate averages for each hydrologic component in each subbasin. This initial set of weights is updated in the years shown in line 2 by tables of weight functions (not shown) following the first set of weights.

For each hydrologic flux (SHED vectors 10-20) in each time step and subbasin, the corresponding weight function is used to calculate an average over all 15 HRUs read from the balance files; the results are written to file spp60-91.bal, which is written in the same format as the component balance files, so that it may be read by Modflow to represent a weighted average of the spatial heterogeneities in the Rattlesnake Creek watershed. The code in Swbavg that reads the component balance files and writes the average balance file was adapted from the code in subroutine Swb1fm of the Modswb package in Modflow used to read balance files.

The following input instructions are illustrated below by file swbavg.inp; corresponding input records are identified as lines 1-6.

Input data requirements for program Swbavg with example file swbavg.inp

1. Number of subbasins, HRUs, years, weight function revisions, output file name:

```
read (in,*) nwshed,ncond,mxper,nrevis,iopsub,outbal                     (1
```

Definitions for line 1:

nwshed: no. subbasins

ncond: no. component cases (HRUs) represented by balance files written by Swat to be averaged (the number no. of combinations of soil types and land use-management schemes;

nper: no. years of study

nrevis: no. times the initial table of weight functions is updated;

iopsub: option to either (yes=1) specify a separate weight function for each subbasin; or (no=0) specify one weight function to be applied to all subbasins.

outbal: name of output balance file containing average of the HRUs; enclose name with apostrophes for free format input.

2. Years of revisions:

```
if (nrevis.gt.0) read (in,*) (irevyr(i),i=1,nrevis)                    (2
```

irevyr(i) = beginning year for each revision of weight functions, i from 1 to nrevis'//

3. Balance file names for component cases (HRUs):
For each component case k=1,ncond:

```
    read (in,*) condnm(k), inpbal(k)                                   (3
```

condnm = identifying name (10 characters) for each component case (HRU);

inpbal = name of balance file (up to 30 characters) written by Swat for each component case (HRU); enclose with apostrophes as shown.

4-6. Tables of weight functions: In the first year (not specified) and in subsequent years (given by line 2), read weight functions. For each table of weights, read two lines of headings: one to identify which set of weights is being read, and one to associate columns of weights, from left to right, with the order in which the balance file names are read.

Following the header lines is a table of weight functions given as percentages; i.e., each row should add to 100 pct If iopsub=0, only one weight function is read, which is applied to all subbasins.

```
read (in,'(a)') wthdg                                                  (4
read (in,'(a)') wthdg                                                  (5
```

For each subbasin j=1,nwshed:

```
    if (j=1 or iopsub>0) read (in,*) (condwt(j,ic),ic=1,ncond)         (6
```

Example: program Swbavg input file swbavg.inp

Instructions for this file are given above.

```
29,15,32,6,1,'spp60-91.bal',    nwshed,ncond,nper,nrevis,iopsub,outfil                (1
1965  1970  1975  1980  1985  1990                                                    (2
'Harney-wsf','CW-H.bal'                                                               (3
'Harney-irc','IRC-H.bal'
'Harney-r/p','RP-H.bal'
'Pratt -wsf','CW-P.bal'
'Pratt -irc','IRC-P.bal'
'Pratt -r/p','RP-P.bal'
'Tivoli-wsf','CW-T.bal'
'Tivoli-irc','IRC-T.bal'
'Tivoli-r/p','RP-T.bal'
'Naron -wsf','CW-N.bal'
'Naron -irc','IRC-N.bal'
'Naron -r/p','RP-N.bal'
'Carwil-wsf','CW-C.bal'
'Carwil-irc','IRC-C.bal'
'Carwil-r/p','RP-C.bal'
normalized weights from beginning year (next record is column header):               (4
  1   2   3    4   5   6    7   8   9   10  11  12   13  14  15                       (5
 79   1  20    0   0   0    0   0   0    0   0   0    0   0   0                       (6
 76   4  20    0   0   0    0   0   0    0   0   0    0   0   0
 27   0   8   19   3  14   17   0  12    0   0   0    0   0   0
 38   0  12   12   0  13    0   0   0   11   1  13    0   0   0
  0   0   0   25   1  24    0   0   0   25   1  24    0   0   0
  0   0   0   14   1  25    8   0  12   14   1  25    0   0   0
 30   0  10   20   1   9    0   0   0   20   1   9    0   0   0
  0   0   0   21   1  18    0   0   0   38   1  21    0   0   0
  0   0   0   20   0  10    0   0   0   32   1  16   14   0   7
  0   0   0   25   1  14    0   0   0   37   2  21    0   0   0
  0   0   0   27   1  17    0   0   0   32   2  21    0   0   0
  0   0   0   24   1  25   15   0  15    0   0   0   12   0   8
  0   0   0   15   0   5    0   0   0   32   3  16   21   0   8
  0   0   0   19   1  10    0   0   0   25   1  14   20   0  10
  0   0   0   12   0   8    0   0   0   23   1  16   24   0  16
  0   0   0   14   0  16    0   0   0   21   1  28    9   0  11
  0   0   0    0   0   0   12   0   8   34   0  16   24   0   6
  0   0   0    0   0   0    8   0  12   19   1  30   12   0  18
  0   0   0    0   0   0    0   0   0   79   0  21    0   0   0
  0   0   0    0   0   0   12   0   8   32   0  18   20   0  10
  0   0   0    0   0   0    0   0   0   32   4  64    0   0   0
  0   0   0   10   0  10    0   0   0   28   2  30   10   0  10
  0   0   0   16   1  33    8   0  17    0   0   0    9   0  16
  0   0   0   12   0  18    0   0   0   20   0  30    8   0  12
  0   0   0   16   0  14    0   0   0   20   2  18   16   0  14
  0   0   0   26   1  22    0   0   0    0   0   0   28   0  23
  0   0   0    5   0  30    4   0  26    0   0   0    5   0  30
  0   0   0    5   0  45    5   0  45    0   0   0    0   0   0
  0   0   0    5   0  45    5   0  45    0   0   0    0   0   0
```

Modswb: a Swat-Modflow connection

Modswb associates the subbasins defined in Swat with aquifer grid domains, and watershed outflows with the stream network defined in Modflow; updates water rights appropriations for each subbasin at the beginning of each stress period; maps hydrologic fluxes calculated by Swat onto the aquifer grid, and summarizes for each subbasin the fluxes relevant to Swat's hydrologic balance. Options affecting Modswb execution are summarized below; for details of implementation, see source code in Appendix D.

The Modswb package consists of subroutines Swb1al, Swb1rp, Swb1fm, and Swb1bd, and is used to connect Modflow's stream-aquifer model to a watershed hydrology model, which is provided by Swat (Arnold et al., 1990) for the Rattlesnake Creek watershed. "Modswb" is an acronym for Soil Water Balance, referring to the hydrologic summary of Swat's results written by subroutine Hydbal. Modflow's basic package input file invokes the Modswb package by setting the Basic input variable iunit(6) > 0.

Modswb subroutines **Swb1al** and **Swb1rp** obtain input data regarding array space requirements, options, and connections between the watershed, stream network and aquifer grid from the Swb input file; see, e.g., file repub.swb. If Modflow is run as a stand-alone program, the last record of the Swb file provides the name of the balance file written by Hydbal, e.g. repub.bal; the Swb and balance input files are described below. **SWB1FM** reads a hydrologic summary for one time step.

<u>Swb1al: memory allocation for Modswb</u>

At the beginning of the simulation, Modflow calls **Swb1al** to read the first record of the Swb input file to define options and allocate memory in Modflow's common x array, conforming to the conventions used in the memory allocation subroutines for Modflow's standard packages. Key memory requirements are provided by the Shed array, defined above.

<u>Swb1rp: subbasin-aquifer grid connections and annual irrigation water use</u>

At the beginning of each stress period (year), Modflow calls **Swb1rp**. In the first stress period, Swb1rp reads the Swb file to define the following:

(a) associate the outflow from each subbasin of a watershed with tributaries that flow into a stream network specified by Modflow's Stream package;

(b) map the geographical extent of the watershed's subbasins onto the gridded aquifer domain specified by Modflow's Basic and Block-Centered Flow packages;

(c) If the Evt package is invoked (see also option **ievopt**, below), calculate an initial evaporation rate from the water table for shallow nodes;

(d) Swb1rp also reads a length conversion factor, cnvlen, and calculates a time conversion factor; both are used in Swb1fm to convert Swat depths to Modflow flow rates, and in Swb1bd to do the reverse.

For each stress period (representing a year), **Swb1rp** summarizes pumping specified by Modflow's Well input for each subbasin, which is used to assign pumping for each time step in Swb1fm. On a combined Swat-Modflow run, pumping as a depth for each subbasin, Shed vector (9), is calculated just prior to entering the daily loop in Swat, and is retrieved for Swat by subr Passflx, to be used as an annual limit on irrigation if Swat's option **ioplim** > 0.

<u>Swb1fm: specify tributary flow, recharge, potential evaporation, and irrigation</u>

For each aquifer time step (day, month, or year),Modflow (or Modstp from Swat) calls **Swb1fm** to transform hydrologic results from Swat into the conditions for Modflow's solution. For stand-alone execution, Swb1fm obtains Swat's results by reading the balance file into vectors 10-20 of array Shed; otherwise, Swat's results are copied to array Shed in subroutine Swt2mod, part of the Hydbal package. Swb1fm calculates the aquifer time step for Modflow, given by **delt = dtswat/cnvtim**, where **dtswat** = the number of days in the time step (passed from Swat), and **cnvtim** = Swat-Modflow time units conversion factor (passed from Swb1rp). The following conditions for Modflow are defined in Swb1fm:

Tributary flow (subbasin outflow)

Subbasin outflow, SHED(22), as a depth for each subbasin is defined as a sum of surface and subsurface flow according to option **jkkopt**. It is assigned as lateral inflow to the stream network as STRM (18) in the STREAM package.

Recharge

Groundwater recharge, SHED(25), as a depth for each subbasin is a combination of hydrologic components from Swat subject to Swat option **jkkopt**. Its distribution in Modflow's recharge package array RECH within each subbasin can be specified according to Modflow option **ioprch**.

Potential evaporation from water table

Maximum water table evaporation rate, array EVTR in the Evt package, is based on potential evaporation, SHED(21), if Modflow option **ievopt** > 0; otherwise, array EVTR is specified by Modflow input to the Evt module.

Irrigation

Irrigation depth assigned by Swat is assigned on the basis of either schedules (in ~.mgt files) or a threshhold on plant water stress (set in ~.mco files); water stress is defined as the fraction of potential plant water evaporation that is available from the soil. In addition, Swat's irrigation assignment is subject to a daily limit and, if option **ioplim** > 0, an annual limit. Option **ioplim** and the daily and annual limits are specified in the added Swat -Modflow coordination section of data in Swat's ~.cod input file; except that if Swat calls Modflow, the annual limit on irrigation depth is retrieved by subr Passflx after Swat calls Swb1rp at the beginning of the year as described above.

Groundwater pumping rates for Modflow depend on Modswb option **irropt** as follows. If **irropt** = 0, the pumping rates are specified by Modflow's Well package input file, and are independent of Swat's results. If **irropt** = 1, the pumping rates given by Modflow's Well input file are scaled according to the irrigation depths in SHED(11) assigned by Swat.

Swb1bd: summarize evaporation from water table and baseflow

After the aquifer equations have been solved in Modflow, **Swb1bd** is called to summarize results relevant to the watershed model's hydrologic summary as follows.

Evaporation from water table

Evaporation from the water table is summarized in SHED(18) as a depth for each subbasin, is based on array EVAP calculated in the Evt package, and replaces Swat's calculation of "revap". Average depth to water, SHED(23), is also calculated for each subbasin. If Modflow's evaporation (Evt) package is invoked, the maximum evaporation rate is set by array EVTR according to option ievopt (see Swb1fm above); otherwise, (18) and (23) are not calculated, and Swat's version of (18) is retained.

Baseflow

Baseflow from aquifer to stream, SHED(19) as a depth for each subbasin, is based on the negative of streambed leakage from stream to aquifer, given by STRM(11) in the Stream package, and added over all stream reaches within the subbasin,.

Input requirements for the Modswb package (SWB and balance files)

The following SWB input section numbers 1-9 refer to entries in the example input file, repub.swb. The SWB input file is read by subroutines Swb1al, Swb1rp, and Swb1fm of the Modswb package. For stand-alone execution of Modflow (**iopswt** = 0), Swb1fm then reads the balance file repub.bal, which is listed after repub.swb, below.

Swb1al (at the beginning of the simulation):
1. Number of subbasins; options for irrigation, evaporation and recharge; other inputs.

```
read (in,*) nwshed,noutfl,nsoils,mxslay,irropt,ievopt,ioprch
```

Definitions:
nwshed = no. subbasins in watershed;
noutfl = total no. streams leaving subbasins; now require that noutfl = nwshed.
nsoils = no. soils; each subbasin is associated with 1 soil type.
mxslay = max. no. soil layers
Options irropt, ievopt, and ioprch are defined above.

Swb1rp (called for each stress period):
In only the first stress period, read watershed-aquifer correspondence data from the ~.SWB file that remain constant.

2. list of tributaries and the subbasins from which they flow: Read list heading and then list of **nwshed** records numbered 1 through nwshed, i.e. one for each subbasin. Since the model is now restricted to one outflow per subbasin, this list is unnecessary but is read to avoid changing the input file.

```
read (in,'(1x,a)') record     !table heading
do i=1,nwshed
    read (in,'(a)') record    !number these records 1 to nwshed.
end do
```

3. length conversion between Swat and Modflow: Swat calculates hydrologic fluxes (flow per unit area) in units of depth (mm); use cnvlen to convert from mm to Modflow's length unit L, corresponding to flow rates [L^3/T]; e.g. cnvlen = 304.8 mm/L for Modflow length unit L=1 ft and corresponding flow rates (cfs). Similarly, subroutine

Modswb converts Swat time units (days) into Modflow time units by cnvtim = tsec(itmuni)/86400 (days/Modflow time unit), where itmuni is Modflow's time unit input option specified in the input file for Modflow's Basic package.

```
read (in,*) cnvlen       ! (e.g. cnvlen=1000 mm/m, or 304.8 mm/ft)
```

4. Basin area and subbasin areal fractions: these should be the same values read by Swat, except that SWAT reads the basin area in km^2, and bsarea should be in Modflow's units of [L^2].

```
read (in,*) bsarea, (shed(1,i),i=1,nwshed)
```

5. soil id for each subbasin (read but not used):

```
read (in,*) (ished(4,i),i=1,nwshed)
```

6. for each soil, number of layers and depth to bottom of each layer (mm) (read but not used):

```
do i=1,nsoils
    read (in,*) nlsoil, (zsoil(j,i),j=1,nlsoil)
end do
```

7. location of subbasin outflow on the network defined by the Stream package:
First, read two lines of headings for location of subbasin outflow. Then, for each subbasin, j = *1* to **nwshed**, read subbasin and tributary index (both equal to index *j*), and the segment and reach of the stream network as defined by the input file for the Stream package.

```
do j=1,2
    read (in,'(1x,a)') record                    !2 table heading records
end do
do j=1,nwshed
    read (in,*) j,j,idxseg,idxrch  ! repeated subbasin index, j; stream
segment and reach.
end do
```

8. correspondence between subbasins and aquifer nodes: This is a 2-D integer array, **ibshed**, that is read according to Modflow's convention used for the **ibound** array in the Modflow's basic package input file (see Modflow manual). Each nonzero entry in the array is an integer, *j*, in the range [1, **nwshed**] to associate the grid cell with subbasin *j*.

```
read (in,'(2i10,a20,i10)') locat, iconst, fmtbnd, iprn
do ir=1,nrow
    read (in,fmtbnd) (ibshed(ic,ir),ic=1,ncol)
end do
```

Swb1fm (called for each time step):

9. For only the first time step of the first stress period (**kkstp** = **kkper** = 1), Swb1fm reads the hydrologic balance file name. Swb1fm then closes the SWB input file and opens the balance file (e.g. repub.bal, below) and reads its first two records.

Balance file input: For a stand-alone run of modswb95 (**iopswt** = 0), Modswb reads the balance input file according to the instructions below, just as it was written by Swat's subroutine Hydbal; otherwise (**iopswt** >0), this information is passed from Swat to Modswb's SHED array by subroutine Swt2mod, part of the Hydbal package.

For only the first time step of the first stress period:

```
    if (kkstp.eq.1 .and. kkper.eq.1) then
        read (in,'(a)') nambal  !hydrologic flux file (~.bal)
             SWB1FM then closes the ~.swb file, opens the ~.bal file,
             and continue reading as follows:
        read (in,*) nyrs, iyr, lutot, jkkopt       !~.bal file record 1
            read subbasin flux input format:
        read (in,'(a)') fmtswb                     !~.bal file record 2
    end if
```

For each time step of each stress period, read file records until one is found containing the phrase '**hydrologic balance**'. From that record, extract the SWAT time step, **dtswat** (days); divide **dtswat** by **cnvtim** to convert **dtswat** into a Modflow time step, **delt**.

```
    read (in,'(t53,10i8)') (idx,j=1,nwshed)         column heading

        subbasin areas given as fractions of basin area:
    read (in,220) avgval, (subval(j),j=1,nwshed)
220 format (f15.3,t53,10f8.5/,(t53,10f8.5))

        noncontributing areas given as fractions of subbasin area:
    read (in,225) fpd,(shed(8,j),j=1,nwshed)
 225 format (t39,f8.5,6x,10f8.5/,(t53,10f8.5))

        active gridded area as fractions of subbasin (calc. in MODSWB):
    read (in,225) avgval,(subval(j),j=1,nwshed)
```

Read hydrologic fluxes calculated by Swat for all subbasins:

```
    do i=10,20
        read (in,fmtswb) idx, coltxt, swtflx(i), flxdev(i),
            (shed(i,j),j=1,nwshed)
    end do

[end of reading balance file written by SWAT for iopswt=0]
```

Example: input file repub.swb for Modswb

```
    9  9  3 10  1  0  1  nwshed noutfl nsoils nlmax irropt ievopt irchop                  1
 watershed: no. tribs and identifiers (5x,i5,10i5)                                        2
  1 n    1    1
  2 s    1    2
  3 s    1    3
  4 n    1    4
  5 n    1    5
  6 n    1    6
  7 n    1    7
  8 s    1    8
  9 n    1    9
 304.8, 365, cnvlen (mm/ft), daystp (days/yr)                                             3
 27.6589e9, .22015,.04909,.20435,.08729,.05282,.08247,.10945,.11788,.07649, bsarea,frc    4
1,2,2,3,2,2,2,2,2, (ished(4,i),i=1,nwshed): soil id for each subbasin                     5
 4, 10.,457.,1219.,1524.,      Carr (z(j,i),j=1,ns) depth (mm) (10f8.3)                   6
 4, 10.,178.,914.,1524.,       Crete
 4, 10.,203.,457.,1524.,       Kipson
watershed outflow with corresponding stream, aquifer node and watershed                   7
 trib shed  seg  rch tributary (4i5,1x,a):       r  c    side
    1    1    1   25 Salt Cr                     4 16     n
    2    2    1   20 Oak Cr                      6 14     s
    3    3    1   31 Elm Cr                      6 21     s
    4    4    1   37 Elk Cr                      5 24     n
    5    5    1   46 Scribner Cr                 7 30     n
    6    6    1   46 Parsons Cr                  7 30     n
    7    7    1   51 Peats Cr                   11 32     n
    8    8    1   68 Five Cr                    21 36     s
    9    9    1   71 Huntress Cr (Spring & Dry) 21 37     n
        66             1 (39i2)                  2      iwshed.mod                        8
 0 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 4 4 4 4 4 4 5 5 5 6 6 6 6 6 6 7 7 7 7 7 7
 0 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 4 4 4 4 4 4 5 5 5 6 6 6 6 6 6 7 7 7 7 7 7
 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 4 4 4 4 5 5 5 5 6 6 6 6 7 7 7 7 7 7 7
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 4 4 4 4 5 5 5 5 6 6 6 6 7 7 7 7 7 7 7
 1 1 1 1 1 2 1 1 1 2 2 2 2 1 1 3 3 3 3 3 4 4 4 4 5 5 5 5 6 6 6 7 7 7 7 7 7 7 7
 0 0 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 6 6 7 7 7 7 7 7 7 7
 0 0 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 5 5 5 5 6 6 7 7 7 7 7 9 9 0
 0 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 5 5 5 3 7 7 7 7 7 7 9 9 9
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 7 7 7 7 7 9 9 9 9
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 7 7 7 7 9 9 9 9 9
 0 0 0 0 0 0 0 0 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 7 7 7 9 9 9 9 9
 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 7 7 9 9 9 9 9
 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 3 3 3 3 3 3 3 3 3 8 8 3 3 3 3 3 7 7 9 9 9 9 9
 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 3 3 3 3 3 8 8 8 8 8 8 3 3 3 3 7 9 9 9 9 9 9
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 3 3 8 8 8 8 8 8 8 8 8 3 3 3 9 9 9 9 9 9 9
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 3 8 8 8 8 8 8 8 8 8 3 3 9 9 9 9 9 9 9 9
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 8 8 8 8 8 8 8 3 3 9 9 9 9 9 9 9
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 8 8 8 8 8 8 8 3 3 9 9 9 9 9 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 8 8 8 8 8 8 8 8 3 3 3 9 9 9 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 8 8 8 8 8 8 8 8 8 8 3 9 9 9 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 8 8 8 8 8 8 8 8 8 8 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 8 8 8 8 8 8 8 0 0 0 0
repub.bal                                                                                 9
```

Example: input file repub.bal (first time step, 1 year) for Modswb

```
   18 1977     9     0 file carr-ire.bal                                                   1
(25x,i3,a,2f7.1,10f8.2/,(t53,10f8.2))                                                      2
 balttl = 1977 initialize subr HYDBAL
 basttl = annual
           365. (days)   1977 annual hydrologic balance:
 id So Va Aq St To Sb Bas Sh component   mean  std d       1       2       3       4       5       6       7       8       9
       2569.600 km**2 area  fractions:                 0.22015 0.04909 0.20435 0.08729 0.05282 0.08247 0.10945 0.11788 0.07649
 ponds (areal fractions):              0.01994         0.03430 0.00310 0.00640 0.02680 0.00210 0.00000 0.01310 0.04139 0.02827
 active gridded fraction:              0.12599         0.07325 0.20532 0.23676 0.09238 0.20991 0.01222 0.08288 0.02565 0.25037
  1  1  0  0  0  1  1   1 10    PREC MM  993.4  107.5 1035.10  895.30 1113.80  855.70  855.70  855.70  855.70 1079.60 1079.60
  2  1  0 -1  0  0 22   0 11     IRR MM  136.1    6.1  137.52  133.12  127.01  133.11  133.76  133.46  145.19  142.80  142.85
  3 -1  0  0  0 -1 12   7 12      ET MM  868.9   24.9  885.77  834.88  859.75  842.52  851.06  839.14  850.67  910.92  903.16
  4 -1  0  0  1  0  4   3 13    SURQ MM  100.8   39.6  124.66   63.53  152.81   55.67   48.57   50.74   51.56  111.52  113.18
  5  0  0  1  0  1 13  38 14   TLOSS MM    3.8    1.7    3.29    5.23    6.80    3.43    2.06    1.42    3.25    2.64    3.43
  6 -1  0  0  c  0  5   4 15    LATQ MM    0.3    0.1    0.38    0.15    0.29    0.43    0.37    0.25    0.19    0.44    0.27
  7 -1  1  0  0  0 11   5 16    PERC MM  192.9   53.8  192.70  166.60  265.51  122.57  121.23  130.07  132.05  232.32  239.72
  8  0 -1  1  0  0  9 107 17    GWRE MM  192.9   53.8  192.70  166.60  265.51  122.57  121.23  130.07  132.05  232.32  239.72
  9  0  0 -1  0 -1  7 105 18     REV MM    6.1    6.3    9.42   22.36    8.70    0.00   17.72    0.00    2.11    0.00    0.00
 10  0  0 -1  1  0  6 104 19    GW Q MM    5.1    9.6    6.79   27.36   12.86    6.04  -23.96    6.34   -0.32    1.01   -2.52
 11  0  0  1  0  1 16  20 20    PSEP MM    2.5    2.2    4.67    0.87    1.18    1.92    0.36    0.00    0.00    6.07    4.08
 12  0  0  0  0  0 25 108 21   ETPOT MM 1367.5   34.7 1371.68 1306.96 1322.86 1347.39 1393.00 1373.94 1375.39 1422.19 1416.17
 Balances:
  1                          soil       -33.5    2.5  -30.90  -36.74  -37.56  -32.37  -31.76  -31.03  -33.58  -32.79  -33.88
  2                          vadose       0.0    0.0    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
  3                          aquifer     49.3   54.0   42.27  -11.01  123.73  -13.16   -4.22   -8.32  -11.68   91.14  102.82
  4                          stream     104.3   43.3  127.54   90.84  164.99   60.63   24.86   57.33   50.75  108.34  107.72
  5                          combined   124.7   94.0  147.87   44.16  253.33   18.52  -10.65   17.98    6.17  177.39  183.95
 water yield:
  1                10   6       WYLD MM   95.3   37.8  118.21   58.27  145.63   51.25   46.77   49.57   48.33  104.91  107.03
 Combined Swat fluxes for Modflow: jkkopt=0
 GWRE = Qtrans_loss + Qperc_rz + Qpond_seep;
 subbasin outflow to streams = c*(Qlat + Qsurf)
 GW recharge:                            0.00          200.66  172.70  273.49  127.91  123.65  131.48  135.30  241.03  247.22
 Tributary flow:                         0.00          125.04   63.68  153.11   56.09   48.93   50.99   51.75  111.96  113.45
 1977 avg temperature, C:               12.8    0.5    12.5    12.5    12.5    12.5    13.1    13.1    13.1    13.7    13.7
    1977appropriation (mm):                            115.00  115.00  115.00  115.00  115.00  115.00  115.00  115.00  115.00
    1977GW pumped (mm):                                137.52  133.12  127.01  133.11  133.76  133.46  145.19  142.80  142.85
    1977GW applications:                43.3               31      41      40      51      53      51      50      49      47
```

Surface water diversions

In the original conceptualization of the Rattlesnake Creek Basin computer model to which SWAT and MODFLOW were to be applied, diversions from surface water, i.e., from streamflow, were omitted, since they were considered to have only a minimal impact on streamflow, and because the computer model lacked the capability to represent them. The computer code that combined SWAT and MODFLOW for a model of the Rattlesnake Creek Basin is referred to here as SWATMOD. However, the computer model of the Rattlesnake Creek Basin was run under the control of a program referred to as a Decision Support System (DSS), which provided a graphical user interface that was used to specify simulation conditions, translate these specifications into input files for SWAT and MODFLOW, and extract simulation results from MODFLOW for plotting, including contour plots of changes in hydraulic head distributions.

As part of a concurrent project to model the Lower Republican River Basin, a computer code also based on SWAT and MODFLOW was developed, referred to here as SWATMOD2. As part of this code, the SURFACE package was written to be called by MODFLOW. The SURFACE package provided an operational model of irrigation that could be supplied by both ground and surface water rights, represented by the WELL and SURFACE packages, respectively. The area of the basin under irrigation according to DWR water rights records and monthly irrigation demand specified as a depth by SWAT's simulations for each subbasin were used by the MODSWB package to determine the total irrigation demand as a flow rate for the basin. This total monthly demand was supplied by distributing it over both ground and surface water rights appropriated for irrigation. This irrigation model and its application to the Lower Republican Basin are described in Volume 1 of the study report (Sophocleous et al., 1997b) and in Perkins and Sophocleous (1999); the SURFACE package is documented in Volume 2 of the study (Perkins and Sophocleous, 1997).

In a more recent application to Walnut Creek Basin, a computer code for a comprehensive watershed model was developed based on MODFLOW and POTYLDR (Koelliker, 1994), which was used to replace SWAT's functions for daily simulations of watershed hydrology. Monthly irrigation demand is specified by POTYLDR and distributed over both ground and surface water diversions using a version of MODSWB. For this computer code, MODFLOW's WELL package was modified to represent both ground and surface water diversions, a change that eliminated the need for the separate SURFACE package, simplifying both the computer code and input data preparation. The development and application of the computer model to Walnut Creek is described in Sophocleous and Perkins (1998). Documentation of the computer code for this model, including the interface to POTYLDR and the modified WELL, STREAM, and MODSWB packages, will be forthcoming in 1999.

Surface water diversions in Rattlesnake Creek Basin

Questions arose during the Rattlesnake Creek Basin study regarding the effect of seasonal diversions from Rattlesnake Creek to fill ponds, which occurred primarily during March and from August through October. In order to consider this effect, but without incorporating a model of surface water diversions into the code for SWATMOD and the DSS, the code developed for the Republican River Basin model was used for the limited purpose of modeling surface water diversions to fill ponds. This was done in the following steps. First, a version of the base case developed for the Rattlesnake Creek model was run under SWATMOD2, the Republican River Basin code, requiring some minor input data changes due to format inconsistencies with SWATMOD. Once the base case simulation was shown to be the same under both SWATMOD and SWATMOD2, a variation on the base case was set up to include surface water diversions. The input data describing these diversions are included in the example below. For more complete documentation of SWATMOD2, see Perkins and Sophocleous (1997).

The SURFACE package for SWATMOD2 represents surface water diversions similarly to groundwater diversions. They are represented as lateral outflow from stream reaches with which they are associated. This is the same mechanism used in MODSWB to represent runoff from the watershed as tributary flow, which is associated with a particular stream reach as lateral inflow for each subbasin. The net lateral inflow due to the sum of tributary inflows and surface water diversions is incorporated into streamflow by a modified version of the stream routing procedure in the STREAM package.

Surface water diversions are specified for each stress period, corresponding to a calendar year. The pumping rate for the diversion is constant over the year unless the diversion is specified to be for irrigation or to fill ponds. For the Rattlesnake Creek basin model, only the pond-filling diversions are represented.

Diversions to fill ponds

This special type of surface water diversion is indicated by "p" in the field for water use type in the input data file (see definitions of input data). The annual water use for each of these diversions is assumed to be allocated in the four months of March and August-October. The average flow rate during each of these months is given by the product $Q \cdot r$, where Q = annual pumping rate specified for the diversion, and $r = 365/(31+31+30+31)$, the ratio of days in a year to days in this four-month period. In the remaining eight months, $r = 0$.

Example: added data for base case (1955-1994) to include surface water rights

MODFLOW as modified for use with SWAT to simulate the Lower Republican River Basin, was also used to simulate the Rattlesnake Creek Basin for the limited purpose of examining the effect of surface water diversions. The use of the SURFACE package included in SWATMOD2 for the Republican River basin to simulate the Rattlesnake Creek Basin is documented below. SWATMOD2 is executed as follows:

```
modswb96 <srfc96.rsp >srfc96.jnl
```

For this run, file **srfc96.rsp**, listed below, contains the redirected keyboard responses to MODFLOW's queries for case name and names of files associated with packages invoked by input for the Basic package; modified or added packages are shown in **bold** type. Redirected terminal output is written to file **srfc96.jnl**, providing diagnostics for the base case simulation. MODFLOW's standard output is written to file **srfc96.prn** (the extension ".prn" is appended to the case name "srfc96")..

File srfc2.rsp:

```
srfc296                                case name (~.log, ~.prn, ~.rsp)
srfc                                   .bas unit   1 Monthly Basic package
x33e_3sb                               .bcf unit  61 Block-centered flow
rttlsnk2                               .wel unit  62 Well: groundwater use
626                                    .riv unit  63
x0                                     .evt unit  65 Evapotranspiration
626cal                                 .swb unit  66 Soil water balance
neww5695                               .ghb unit  70 generalized boundary conditions
```

```
02west                                       .rch unit  64 Recharge
trans                                        .sip unit  68 strongly implicit solver
tran12                                       .oc  unit  69 Output control
2zonea3                                      .str unit  67 monthly Strmflow, Ks=0.54 ft/day
rat-srf2                                     .wel unit  71 Surf: surface water use
```

<u>Basic package input data changes to invoke added packages</u>

Line 4 of input requirements for the Basic package was modified to allow added packages to be invoked. Except for Line 4, input requirements for the Basic package are the same as described in the MODFLOW manual (McDonald and Harbaugh, 1988). The following additional packages are to be invoked to simulate the Rattlesnake base case with surface water diversions using SWATMOD2:

(a) The Soil Water Balance (SWB) package connects hydrologic flow paths from a watershed, calculated on a daily basis by SWAT (Arnold et al., 1990), to groundwater and streamflow represented by MODFLOW (McDonald et al., 1988).

(b) The SURFACE package represents diversions from streamflow for irrigation. This package is analogous to the WELL package in its input format and operation.

The modified format for line 4 follows.

```
4. Data: iunit(24)
    Format: 24I3
    (bcf wel drn riv evt swb ghb rch sip pcg sor  oc str srf)
   index        abbrev  package
                bas     basic
      1         bcf     Block-centered flow
      2         wel     Well
      3         drn     Drain
      4         riv     River
      5         evt     Evapotranspiration
      6         swb     Soil water balance
      7         ghb     General head boundary
      8         rch     Recharge
      9         sip     Strongly implicit solver
     10         pcg     Preconditioned conjugate gradient
     11         sor     Successive overrelaxation
     12         oc      Output control
     13         str     Stream
     14         srf     Surface water diversions
```

In the following partial listing of base case input file **srfc.bas**, line 4 identifies the packages invoked according to the above definition. This input file also defines array IBOUND, the boundary conditions and the active domain of the solution; array STRT, the starting heads for the solution (only the first row of the array is listed); and the duration (s) and number of time steps for each stress period.

Basic package input file example

```
Pre-dev transient '60-'60 DWR Rattlesnake Creek 1/12/96 w/ Swat: 6060swat.bas          (1
                                                                                       (2
        1        47        190        40         4                                     (3
 61 62  0 63 65 66 70 64 68  0   0 69 67 71                                            (4
```

At the beginning of each stress period (year), the following data are read for each surface water diversion.

layer,row,column;
annual estimated use for year, Q (cfs);
Annual appropriation, Qwr (cfs): multiplied by annual water use fraction **frcuse** to estimate water use for scenarios;
Distance to stream **dsstrm**, corresponding stream reach index **idxrch**, and water right application number **iappno** are used in water scenarios as described below.
DWR use code (**wruse** = 3 indicates irrigation);
Index **iwr** associates well with lists of water rights (vol. 1, App. 3B2) and wells included in November 1994 water level survey (vol. 1, App. 3B3).
grid coordinates to precision of .01 as fraction of cell width (**rwr**, **cwr**);

Both annual estimated use Q and appropriation Q_{wr} are specified for each diversion. With option **itmp** = -2, annual use can be estimated from appropriations for a previous stress period by $Q = \text{frcuse} \cdot Q_{wr}$, where **frcuse** = the water use fraction for the current stress period. This option is used to estimate annual use for each surface water diversion for management scenarios the same way it is applied with the modified WELL package by multiplying its appropriation for 1994 times the water use fractions estimated for future years.

As with the modified WELL package, surface water diversion flow rates are scaled and selected on the basis of application number **iappno** and distance to stream **dsstrm** according to management scenarios 1-11 (Sophocleous et al., 1997, vol. 2), which are specified by input options **iadcod** and **welmpy** in the SWB package input file.

Input instructions for the SURFACE package

```
For each simulation (SRF1AL):
1.    Data:    MXSURF,ISRFCB
      Format:  (2I10)

For each stress period (SRF1RP):
2.    Data:    ITMP,iyrsrf,frcuse,accwr
      Format:  (2I10,2f10.0)
C
For each surface water right (SRF1RP): items in bold are required.
DO II=1,NSURFS
3.    Data:    layer,row,column,Q,Qwr,dsstrm,istrch,iwryr,wruse,rwr,cwr,wright
```

```
Format:  (3i10,3f10.0,2i5,1x,a1,2f8.0,1x,a)
```

Definition of input data for the SURFACE package

```
Line 1 (for each simulation).
MXSURF maximum number of surface water diversions used at any time.
ISRFCB flag: print(<0) or save(>0) cell-by-cell flow rates.

Line 2 (for each stress period).
ITMP   flag:
     > 0: NSURFS = ITMP, surface water diversions active in this stress period.
     < 0: use data from the previous stress period, with the following exception.
     =-2: The assigned pumping rate for each diversion during a given time step is based
          on the diversion's appropriation Qwr (Line 3) specified in the previous stress
          period.  As in the Well package, this option is used to scale the pumping rate in
          each time step to supply irrigation demand.
iyrsrf calendar year.
frcuse total estimated water use as a fraction of appropriations; this is based on
       analysis of water use reports available for years 1980-1993, and on precipitation
       models derived from these reports for the remaining years of the 1977-1994 study.
accwr  total surface water rights.

Line 3: input format (columns) for each surface water right:
c------------------------------------------------------------------------------
c         (a) water use input file:
c  1-10: i10    ilay   grid layer
c 11-20: i10    irow   grid row
c 21-30: i10    icol   grid column
c 31-40: f10.0  Q      annual avg estimated water use [L^3/T], e.g. cfs
c 61-65  i5     istrch index to stream reach (order in which stream reach is read)
c 66-70  i5     iwryr  1st yr of appropriation, corresp. to applic. no
c 72     a1     wruse  water use type (3=irrig use, *=b.c. code, "p" = "ponds")
c                      (see discussion of surface water diversion model)
c 91-110 a20    water right identifier
c------------------------------------------------------------------------------
```

SURFACE package input example: file rat-srfc.wel

File rat-srf2.wel, describing surface water rights for 1955-1994, is listed as follows.

```
    3         0
    0      1955
    0      1956
    0      1957
    0      1958
    0      1959
    0      1960
    0      1961
    0      1962
    0      1963
    0      1964
    0      1965
    0      1966
    0      1967
    0      1968
    0      1969
    0      1970
    0      1971
    0      1972
    0      1973
    0      1974
    0      1975
    0      1976
    0      1977
    0      1978
    0      1979
    0      1980
    3      1981
    1        24       136   -1315.1                      591 1977 p                 A02927301
    1        25       152   -2630.1                      690 1941 p                 V  000900
    1        25       151     -43.8                      689 1941 p                 V  001100
    3      1982
    1        24       136   -3156.2                      591 1977 p                 A02927301
    1        25       152  -25319.5                      690 1941 p                 V  000900
    1        25       151    -109.6                      689 1941 p                 V  001100
    3      1983
    1        24       136   -3787.4                      591 1977 p                 A02927301
    1        25       152   -3156.2                      690 1941 p                 V  000900
    1        25       151    -109.6                      689 1941 p                 V  001100
    2      1984
    1        25       152  -29245.8                      690 1941 p                 V  000900
    1        25       151    -109.6                      689 1941 p                 V  001100
    2      1985
    1        25       152   -3682.2                      690 1941 p                 V  000900
    1        25       151    -109.6                      689 1941 p                 V  001100
    2      1986
    1        25       152  -11046.6                      690 1941 p                 V  000900
    1        25       151    -109.6                      689 1941 p                 V  001100
    1      1987
    1        25       152  -15780.8                      690 1941 p                 V  000900
    2      1988
    1        25       152   -7890.4                      690 1941 p                 V  000900
    1        25       151     -87.7                      689 1941 p                 V  001100
    1      1989
    1        25       152  -13698.6                      690 1941 p                 V  000900
    1      1990
    1        25       152     -24.1                      690 1941 p                 V  000900
    1      1991
    1        25       152   -6575.3                      690 1941 p                 V  000900
    1      1992
    1        25       152   -7890.4                      690 1941 p                 V  000900
    1      1993
    1        25       152   -8219.2                      690 1941 p                 V  000900
    1      1994
    1        25       152   -3682.2                      690 1941 p                 V  000900
layer       row    column         Q        Qwr   dsstrm irch wryr Use    rwr     cwr approp id
```

References

Arnold, J.G., J.R. Williams, R. Srinivasan, K.W. King, and R.H. Griggs, 1994. SWAT, Soil and Water Assessment Tool. USDA, Agricultural Research Service, Grassland, Soil and Water Research Laboratory, Temple, TX.

Birdie, T., M. Sophocleous, R. Buddemeier, S. Perkins, N. Stadnyk and J. Ma, June 1996. Rattlesnake Creek Sub-basin modeling: status report on model calibration and verification. 3rd progress report to Kansas Dept of Water Resources from Kansas Geological Survey.

Koelliker, J.K., 1994. User's manual for Potential Yield model Revised (POTYLDR). Unpublished document, Civil Engineering Dept., Kansas State University, Manhattan, KS, 41 pp.

Koussis, A.D., M.A. Sophocleous, L. Bian, S. Zou, 1994. Lower Republican River Basin: stream-aquifer study. First year report to Kansas Water Office, 142 pp.

Maidment, D.R., ed., 1993. Handbook of Hydrology, McGraw-Hill.

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

Perkins, S.P., and M.A. Sophocleous, 1997. Lower Republican Stream Aquifer Project, Vol. 2. Development and documentation of a combined watershed/stream-aquifer program. Open-File Report No. 97-9, Kansas Geological Survey, Lawrence, KS.

Perkins, S.P., and M.A. Sophocleous, 1999. Development of a comprehensive watershed model applied to study stream yield under drought conditions. Accepted for publication in Ground Water.

Prudic, D. E., 1989. Documentation of a computer program to simulate stream-aquifer relations, U.S. Geological Survey Open-File Report 88–729, Carson City, Nevada, 113 pp.

Sophocleous, M.A., 1994. Numerical model selection considerations and data collection progress report for the Rattlesnake Creek Subbasin, Kansas. Kansas Geological Survey, Open-File Report 94–31, 41 pp.

Sophocleous, M.A., T. Birdie, R.W. Buddemeier, S.P. Perkins, and J. Ma, 1994. Computer model selection and data assimilation for the Rattlesnake Creek Sub-basin water resources management program. Final Report (FY 1994) to the Division of Water Resources (DWR), Kansas State Board of Agriculture, from Kansas Geological Survey (KGS), Open-File Report 94–62, 81 pp.

Sophocleous, M.A., S.P. Perkins, S. Moustakas, and R. Kaushal, 1995a. Republican River stream-aquifer study: model development. Fiscal year progress report to Kansas Water Office from Kansas Geological Survey, Open-File Report 95–27 (May).

Sophocleous, M.A., S.P. Perkins, S. Moustakas, and R. Kaushal, 1995b. Republican River stream-aquifer study: model development and application. Sept 1995 year-end progress report to Kansas Water Office from Kansas Geological Survey, Open-File Report 95–53.

Sophocleous, M.A., J.K. Koelliker, R.S. Govindaraju, T. Birdie, S.R. Ramireddygari, and S.P. Perkins, 1997a. A computer model for water management in the Rattlesnake

Creek basin, Kansas. Project completion report to Division of Water Resources, Kansas Department of Agriculture, January, 204 pp.

Sophocleous, M.A., S.P. Perkins, N.G. Stadnyk, and R.S. Kaushal, 1997b. Lower Republican Stream Aquifer Project, Vol. 1. Combined watershed/stream-aquifer model conceptualization and application. Open-File Report No. 97-8, Kansas Geological Survey, Lawrence, KS.

Sophocleous, M.A., and S.P. Perkins, 1998. Evaluation of Wet Walnut water supply availability: development and application of an integrated watershed model; a year-end progress report. Open-File Report No. 98-60, Kansas Geological Survey, Lawrence, KS.

Appendix A: SWAT input data format (manual supplement)

This was written to be a supplement to SWAT's manual (v. 2), added to Chapter 3 as Section 3.4, describing the format for input data files read by SWAT. Input files are described roughly in order of their appearance in Sections 3.1 and 3.2 of the SWAT manual, which summarizes the contents of all input and output files associated with program SWAT. This also describes changes and additions to input file format and to SWAT program procedures as described below.

3.4. Preparing input data for Swat without an operational GIS front end

This format description is especially useful for creating and modifying input data files for SWAT if a preprocessor is not available to handle the formats. A preprocessor for SWAT is available that is based on the GRASS system, a public domain graphical information system (GIS). However, Arc-Info is used at the University of Kansas, Kansas Geological Survey, and the Kansas state agencies that have an interest in SWAT and its connection to MODFLOW. A preliminary version of a preprocessor for SWAT based on Arc-Info has been developed (L. Bian, personal communication). However, the scope of that project has thus far been limited to the original SWAT (v. 2) program, and has not considered changes made to SWAT input for some of the reasons mentioned below.

3.4.1. Summary of changes to Swat94.2

a) Input data file changes

Input file formats are for the most part in original Swat (v. 2) format, with some exceptions. The codes data file (~.cod) includes options relevant to our revision of Swat and its linkage with Modflow. The general basin file (~.bsn) was changed to correct a problem in v. 2's initialization of soil moisture profiles. Formats for the soils data files (~.sol) and weather data files (rf and tmp) were changed back to those used in v. 1 of Swat. This change was made partly in order to allow data files made for v. 1 to be used in both versions, thereby making comparison of results easier; but also in part to remedy apparent bugs in v. 2 as described below, or to make file handling easier for the user, e.g. an option to read weather data either from one file for all stations or from individual files for each station.

b) Addition of log files to track data input:

1. Swat.Log, unit iolog=81, to which a log is written of input file openings and readings during program execution; opened in Main and written in Main, Open, Open2, Open3, Read, Readinpt.
2. Weather.Log, unit iomeas=82, to which a record is written showing rainfall for each station for rainy days only; opened in Main and written in Clicon.

c) Compile-time bugs under Lahey Fortran (likely allowed by Microsoft Fortran)

1. **blockd**: comment out data statements for variables with the prefix "thr" not in common.
2. comment out **$debug** in **main**, **readinpt**, **subbasin**, **swata-e**.

3. **main**: fix 12301 format (ln 266) by adding a comma to separate fields.
4. **swata-e**: subr **apply** (ln 73) and **clgen4** (ln 141): fix illegal transfers from outside a **do** range into the range.
5. **swatp-r**: comment out data statements for the following arrays in subr **resis**: storec, releasec, storek, releasek, and initialize them in block data (file **blockd.for**)

d) Execution time bugs:

1. General basin (**bsn**) data bug: input variable **ffcb** is read but is not passed from subr **read** to **readinpt**, where it is used, and it is not used properly in **readinpt**; a description of the bug and correction follows.

General basin input variable **ffcb**, called ffc in Swat manual, is supposed to allow specifying a nonzero initial fraction of field capacity; if input as zero, then an initial fraction is calculated on the basis of the rainfall statistics file (wgn). However, in the original source code for Swat (v. 94.2), ffcb was not passed from Read to Readinpt, and Readinpt did not handle ffcb properly anyway.

The fraction of field capacity calculation was revised to do what was apparently intended, which is to base it on avg annual rainfall (r(8)=sum of smy, above), to be overridden by fraction ffcb read from file ~.bsn, subr Read. ffc in '93 version was passed from Read() to Readinpt() via arg lists. For '94.2 version, ffc expanded to subbasin resolution and moved to common; ffc was changed to ffcb in Read(), but ffcb was not passed to Readinpt().--spp jun 95

2. **weather** (rf and tmp) data bug: daily rainfall and temperatures from multiple stations were not read and assigned properly to subbasins; a description of this bug and correction follows.

2. weather (rf and tmp) data bug: daily rainfall and temperatures from multiple stations were not read and assigned properly to subbasins in the '94.2 version. The weather data reading code in Clicon was changed (by ARS) from a form in the '93 version that worked to one that does not appear to work for multiple weather stations in the '94.2 version, so the code from Swat's '93 version was used instead to read one daily rainfall value per record and two daily temperature values (max and min). The (5x,2f10.1) format used here (5000 format) is the same as the '93 version's 10300 format. --spp jun 11 95, KGS

3. Soils data (sol)

The '93 version of silt percentage SIL as read from the soils input files (~.sol) appears to have included clay, but in the '94 version the SIL just includes silt. For example, the Swat Y7 test case shows apparently unchanged data from the '93 version, with SIL=92; results for this case show a negative sand percentage, which is calculated by sand = 100 - (silt + clay).

Soils input data are read according to the same format used in the '93 program version, and not the form provided by the Runsoil program, which extracts soils data from database files; see soil hydraulics (~.sol) data file description.

4. Certain conditions result in the Swat program to reference zero-valued array indices, notably in subr crpmd and one or two others. These were encountered during execution as a result of routinely using Lahey Fortran's bounds checking option for compiling all source files, and verified by successful execution without the bounds

checking option for the routines where the error occurred. This problem was remedied by checking for zero-valued indices in these subroutines, rather than allowing the condition to occur by executing without bounds checking.

e) Swat program user manual bugs

Certain equations were found to be incorrect in the manual, including equations (103) and (108). The coding of these equations in the program is consistent with their expression in SWWRB by Arnold et al., 1990. This indicates that the remainder of the manual's model description should be compared with the book's description for consistency.

3.4.2. Specifying a Swat case with files (~.cio) and options (~.cod)

The file named "file.cio" specifies most file names associated with a particular simulation case to be run, and is read on device number 2 by Swat routines Open, Main and Open2. Certain other file names associated with Swat-Modflow coordination and Swat revisions are specified on the ~.cod file. In both cases, file names of up to 13 characters in length are read from these input files. Regarding weather files, option iopwfl on the ~.cod file allows all precipitation data stations to be read from the same file, and similarly for temperature data stations. See also added option iopwea (~.cod file) to read daily values for solar radiation, relative humidity, and wind speed.

<u>cio(2): Swat input control "file.cio" (Open, Main, Open2) ex. corn90dc.cio</u>

part 1 in subr open (on source file swatf-o.for):

```
      open (70,file='kevin.out',status='unknown')
      open (2,file='file.cio')
      3 lines of text for a case description:
read (2,'(20a4)') (title(j),j=1,60)                                    (1
      basin file names:
read (2,'(6a)') stdout,sbsout,rchout,rsvout,lwqout,pestout             (2
read (2,'(6a)') event,cropdb,tilldat,pestidat,codedat,basndat          (3
read (2,'(6a)') lwqdat,routin,statin,bigsub,watqal,wqout               (4
open files:
(3,sbsout) (8,rchout);(1,rsvout); (4,lwqout); (5,pestout); (6,stdout);
(7,event); (10,codedat); (11,basndat) (77,bigsub);(14,lwqdat)
(9,routin);
(15,cropdb=crop.dat: crop data base);
(16,tilldat=till.dat: tillage data base);
(17,pestidat=pest.dat: pesticide data base)
```

part 2 in main (main.for)

```
      no. rainfall and temperature gaging stations (see notes on options
      iopwfl and iopwea, above):
read (2,'(2i4,1x,a)') nrgage, ntgage                                   (5
      18 fields for precipitation file names: (3 records):
read (2,'(6a13)') (rfile(j),j = 1,18)                                  (6
      18 fields for temperature file names: (3 records):
read (2, '(6a13)') (tfile(j),j = 1,18)                                 (7
      18 fields for reservoir file names: (3 records):
read (2, '(6a13)') (resvo(j),j = 1,6)                                  (8
```

part 3 in subr open2 (swatf-o.for)

```
For each subbasin I=1 to Lu:
    read (2,'(i4,i2,i4,i2,i3,5a)') isb, id2, id6, id8, icnty,          (9
     1         subdat, routdat, ponddat,chemdat, soildat
    read (2, '(2x,4a,2i4)') mgtdat, mcodat, gwdat, wgendat,            (10
     1    irgage(i), itgage(i)
open (10,subdat); (11,routdat); (12,ponddat); (13,chemdat);
(14,soildat); (17,mgtdat); (18,mcodat); (19,gwdat); (20,wgendat)
end do
```

Example corn90dc.cio (rename as file.cio for execution)

```
Rattlesnake Creek Watershed: 29 subbasins  corn90dc.cio: decoupled ET(ioprev=0)         (1
corn-fallow rotation: cornfall.mgt; stress factor=0.90, ioplim=1,iopwea=1
oct 26 95 ponds, 5% noncontributing area; k=1 mm/h 1960-1993: 34 yrs
 corn90dc.std   case23.sbs      rat.rch      rat.rsv      rat.lqo      rat.pso         (2
      rat.eve     crop.dat     till.dat      pest.dat corn90dc.cod    snake.bsn        (3
      rat.lwq      rat.fig      rat.sta      rat.bsb                                   (4
   8   6  ,nrgage, ntgage                                                              (5
   rs6093.pcp                                                                          (6


      all.tmp                                                                          (7


                                                                                       (8
   1 0    0 0  0    rat01.sub     rat01.rte     p5011.pnd chemical.chm  pratt.sol      (9
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        1   2                       (10
   2 0    0 0  0    rat02.sub     rat02.rte     p5012.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        1   2
   3 0    0 0  0    rat03.sub     rat03.rte     p5013.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        1   2
   4 0    0 0  0    rat04.sub     rat04.rte     p5014.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        3   2
   5 0    0 0  0    rat05.sub     rat05.rte     p5015.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        3   2
   6 0    0 0  0    rat06.sub     rat06.rte     p5016.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        3   2
   7 0    0 0  0    rat07.sub     rat07.rte     p5017.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        3   2
   8 0    0 0  0    rat08.sub     rat08.rte     p5018.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   2
   9 0    0 0  0    rat09.sub     rat09.rte     p5019.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   2
  10 0    0 0  0    rat10.sub     rat10.rte    p50110.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        5   2
  11 0    0 0  0    rat11.sub     rat11.rte    p50111.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        5   4
  12 0    0 0  0    rat12.sub     rat12.rte    p50112.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   6
  13 0    0 0  0    rat13.sub     rat13.rte    p50113.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   4
  14 0    0 0  0    rat14.sub     rat14.rte    p50114.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   4
  15 0    0 0  0    rat15.sub     rat15.rte    p50115.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        5   4
  16 0    0 0  0    rat16.sub     rat16.rte    p50116.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   6
  17 0    0 0  0    rat17.sub     rat17.rte    p50117.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   4
  18 0    0 0  0    rat18.sub     rat18.rte    p50118.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   6
  19 0    0 0  0    rat19.sub     rat19.rte    p50119.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        8   5
  20 0    0 0  0    rat20.sub     rat20.rte    p50120.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        6   5
  21 0    0 0  0    rat21.sub     rat21.rte    p50121.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        4   3
  22 0    0 0  0    rat22.sub     rat22.rte    p50122.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        4   3
  23 0    0 0  0    rat23.sub     rat23.rte    p50123.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        4   3
  24 0    0 0  0    rat24.sub     rat24.rte    p50124.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        4   3
  25 0    0 0  0    rat25.sub     rat25.rte    p50125.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        6   5
  26 0    0 0  0    rat26.sub     rat26.rte    p50126.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        4   3
  27 0    0 0  0    rat27.sub     rat27.rte    p50127.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        4   3
  28 0    0 0  0    rat28.sub     rat28.rte    p50128.pnd chemical.chm  pratt.sol
      3-wsf.mgt     rat90.mco       rat.gw rat.wgn        2   1
  29 0    0 0  0    rat29.sub     rat29.rte      p501.pnd chemical.chm  pratt.sol
```

```
3-wsf.mgt    rat90.mco      rat.gw rat.wgn          4   3
```

cod(10): Options input file (Main); ex. file corn90dc.cod

Contents for the options input control file were modified to specify options both for Swat-Modflow coordination and for Swat revisions (e.g. calculation of evaporation); see example file corn90dc.cod, below, and "Options added to Swat and used for the Rattlesnake Creek watershed model". The following is read by code included in Swat's mainline as file swatmod2.h.

Example: Swat options input file corn90dc.cod

In this example, annual irrigation depths (mm) corresponding to years 1960-1994 are specified using option ioplim=3. These irrigation depths represent basinwide averages for Rattlesnake Creek based on total estimated irrigation water use divided by total estimated irrigated area. Annual irrigation depth limits for each subbasin can also be specified using option ioplim=1.

```
Rattlesnake Creek watershed - Control file corn90dc.cod: irrig w/ pump limits         (1
  21960  29   2   3   3   0   1   0   0   0   0   0   0   0   0   0   0  29   0   0   (2
   0   1   1   0   0   1   1   0   0   3   0   119771977   1   1   0                  (3
 corn90dc.bal theissen.inp sand6092.rad                                               (4
(5x,12f5.0)                          temperature data format (file all.tmp)           (5
(4x,8f6.0)                           rainfall data format (file all.pcp)              (6
(14x,f5.0,12x,2f6.0)                 rad(ly/d), relhum(%), wind(m/s) (sand6092.rad)   (7
25.4,1.0, 38.00,-99.00,     wsfdmx, swminf, pmpeff, phideg, rlamdg                    (8
 369.1 358.7 332.8   391   397 329.2 380.6 384.3 369.2   331                          (9
 485.8 385.3   332 358.2 417.7 423.9 466.1 396.2 473.4 397.8
 476.1   363   378 379.1 469.3 352.1 348.3 343.4 452.1 359.5
 436   444.9 265.5   265 421.4
```

```
read (10,(a)) title                                                (1
```

Original options described by Swat manual, 3.2, pp. 5-8.

```
read(10,(20i4)) nbyr,iyr,lu,ipd,nsim,msim,ign,iwst,isst,ires,      (2
       igraf,irain,itemp,iresq,idaf,idal,iprn,iprp,iopt,ipet
```

Note regarding input variables iwst, isst and iopt:

If iwst or isst > 0, then iopt must be specifed as the reach for which observed and calculated yields are to be compared via subr vald25 for both streamflow and sediment yields. A valid reach index must be specified; reaches are referenced on the routing configuration file (~.rte). For more clues regarding comparison of observed and calculated yields, see the chapter in Swat's manual on routing.

Swat-Modflow coordination options

```
read (10,'(20i4)') ilog,iopmod,iopswt,ibgndv,ienddv,iopsol,        (3
    1              iopwfl,ithpcp,ithtmp,ioplim,iopqtl,iopcmp,
    1              iopyrf,iopyrl,iopet,iopwea,ioprev,itrace,jkkopt
```

1 **ilog**: an original Swat option; see Swat manual, 3.2, p. 8.

2. **iopmod**: option for Swat to summarize hydrologic results on an annual(1), monthly(2) or daily(3) basis for each subbasin and call Hydbal to write results to a "balance" file

named below (nambal, rec. 4) that can be read by Modflow on a subsequent run; see iopswt.

3. **iopswt**: option to either (0) run Swat and Modflow independently or (1) call Modflow from Swat at the end of each aquifer time step. If **iopswt=1**, Swat calls Modflow at the end of each time step specified by **iopmod** (above).
4. **ibgndv**: beginning day of year to print daily results for rain days.
5. **ienddv**: ending day of year for printing daily results for rain days beginning to **ibgndv**.

6 **iopsol**: option to show initial and final states of soil on hydrologic balance file after avg annual results have been calculated; Swat calls SoilStat after last call to Hydbal.

7 **iopwfl**: option (y=1,n=0) to read daily precipitation data for all stations from the same file, and to do similarly with temperature data. Weather stations are identified in the ~.cio file by an integer i from 1 to n corresponding to the order of data columns from left to right. Data format for temperature and precipitation data are defined in records 5-6, below.

8 **ithpcp**: option (y=1,n=0) to use Theissen weights to calculate spatial averages of daily precipitation reported at weather stations. Theissen polygon weights are read from file (**namths**) specified on record 4.

9 **ithtmp**: option (y=1,n=0) to use Theissen weights to calculate spatial averages of daily temperature at weather stations. Theissen polygon weights are read from file **namths** specified on record 4.

10 **ioplim**: option (y>0,n=0) to limit daily irrigation pumping by **wsfdmx** (mm) and annual irrigation pumping by groundwater appropriation **pmpmax** (mm), reduced by pumping efficiency **pmpeff**; these are defined in records 8-9, below, subject to the following conditions. If Swat and Modflow are run independently (specified by setting **iopswt=0** and **ioplim>0**), annual irrigation limits are read from record 9 (below). Otherwise, i.e., if Swat calls Modflow (specified by **iopswt>0** and **ioplim>0**), Swat calls subroutine Passflx at the beginning of each year (stress period) to obtain annual maximum pumping flux rates (mm) for each subbasin, based on Modflow's ~.WEL input file and summarized by Modswb subroutine Swb1rp.

Note: based on option **irropt** specified by input to the Modswb package, subroutine Swb1fm either assigns pumping rates specified by the ~.WEL input file (**irropt=0**), or scales these pumping rates to correspond to the irrigation depths assigned by Swat for each subbasin (**irropt>0**); see "Definition of options for the Modswb package."

11 **iopqtl**: option (y=1,n=0) to include transmission loss with infiltration as input to soil profile (passed to subroutine purk18 with transmission loss).

12 **iopcmp**: option (y=1,n=0) to write daily results of evaporation calculations to file <casename>.ET. If **iopcmp=1**, daily results are written for the entire period of simulation; if **iopcmp=0**, daily results are restricted to the range of years given by **iopyrF** and **iopyrL** (below); set both to zero to prevent writing daily results. Output includes independent calculation of reference ET by subroutine Penman (see option **iopet**, below).

13 **iopyrF**: first year of writing daily evaporation results (see **iopcmp**, above);

14 **iopyrL**: last year of writing daily evaporation results (see **iopcmp**, above);

15 **iopet**: option (y=1,n=0) to use the reference crop evaporation calculated by subr Penman instead of the value calculated by Swat's subroutine EVAP8, the default option. Subroutine Penman (Appendix F) was written and linked with Swat by SPP, and calculates reference crop evaporation according to the method recommended by Shuttleworth (1993). This method includes the effect of long-wave emission in calculating net radiation, in contrast to Swat's subroutine; see also the following related option, **iopwea**.

16 **iopwea**: option (y=1,n=0 to read, rather than synthesize, daily values for insolation (ly/d), relative humidity (pct) and wind speed (m/s at 10 m above ground surface) in that order from an input file with name namrad (rec. 4) in format fmtrad (rec.7). The default option (iopwea=0) is to generate them as in the standard Swat program.

17 **ioprev**: option (y=1,n=0) to reduce evaporative demand on soil moisture by the depth evtgw evaporated from the water table according to Modflow.

18 **itrace**: trace option (y=1,n=0) to print the Julian day at the top of the daily loop.

19 **jkkopt**: option (y=1,n=0) to evaluate groundwater recharge and watershed contribution to streamflow according to a scheme conceived by Prof. James K. Koelliker at KSU for the Rattlesnake basin study, hence the option's name (see "Definition of options for the Modswb package").

File names: balance, weights, radiation

```
read (10,'(3a13)') nambal, namths, namrad                          (4
```

1. nambal: output file name for hydrologic balance to be passed to Modflow
2. namths: input file name for Thiessen weight functions for averaging rainfall or temperature based on options ithpcp and ithtmp, above. Format for this input file is described below.
3. namrad: input file name for measured daily radiation (ly/d), relative humidity (pct) and wind speed (m/s) 2 m above land surface (ALS).

Weather input data formats, up to 30 chars each (may be 'FREE'):

```
read (10,'(a)') fmttmp   ! temperature data format                 (5
read (10,'(a)') fmtpcp   ! rainfall data format                    (6
read (10,'(a)') fmtrad   ! radiation, rel. humidity & wind speed   (7
```

Data in free format, defined below:

```
read (10,*) wsfdmx, swminf,pmpeff, phideg,rlamdg                   (8
```

wsfdmx: Daily irrigation pumping limit (mm).

swminf: Available water capacity threshhold, below which irrigation is applied during the growing season, but only if irrigation option irr(j) = 3 as read from the ~.mco file (subr readinpt). If irr(j) = 1, then the plant stress factor controls irrigation according to Swat's original version, but subject to added daily and annual constraints.

pmpeff: pumping efficiency fraction: applied irrigation/water pumped; the loss is assumed to be evaporative.

phideg =latitude (deg); rlamdg = longitude (deg)

Rec. 9: read max. annual irrigation depth (mm/yr) subject to ioplim (above) as follows.

If ioplim=1, annual pumping limits are read from the ~.cod input file for each subbasin from within the annual loop in the mainline for every year as follows:

```
read (10,*) iyr, (pmpmax(j),j=1,lu)
```

if ioplim=2, read irrigation limits for each subbasin as above but only in the first year; these limits then apply to every year.

If ioplim=3, read irrigation limit, constant over all subbasins, for each year.

```
read (10,*) (pmpmax(k),k=1,nyrs)                                           (9
```

3.4.3. Data base input files (~.dat, read by subr Readinpt)

dat(15): Crop data base file crop.dat

Example (excerpt from crop.dat):

```
 1  SOYB    25.0     .30    25.0    10.0     5.0     .90  15.010  50.950     1.0
      1.0     3.0   .0100     .85    35.0      .8    2.00  660.31    .200   .0650
    .0091    .220     .60    .330  370.00    .130   .0524   .0265   .0258   .0074
    .0037   .0035   1.266    .633    .729      1.   5.010   15.95    5.00     .50
     4.75    1.00    0.40    0.20   52.20    1.00
 2  CORN    40.0     .50    25.0     8.0     5.0     .80  15.050  50.950  1.0000
     1.00     3.0   .0070     .85    20.0     2.0    2.00  660.44    .200   .0175
    .0025     .01     .60   2.510  100.00    .150   .0440   .0164   .0128   .0062
    .0023   .0018    .433    .433    .213      4.   5.010   15.95    8.00     .50
     4.75    2.00    0.40    0.20   47.60   1.000
```

```
c****icnum = crop number
c     cpnm = crop name
c     be = biomass-energy ratio
c     hi = harvest index
c     to = optimal temperature for plant growth degrees c
c     tb = base temperature for plant growth degrees c
c     blai = max lai for subbasin j
c     dlai = fraction of growing season when leaf area declines
c     dlp1 = 1st point on optimal leaf development curve
c     dlp2 = 2nd point on optimal leaf area development curve
c     gsi = maximum stomatal conductance
c     chtmx = maximum canopy height (m)
c     rdmx = maximum root depth(m)
c     pt2 = co2 concentraion in future atmosphere/resulting WA value
c     cvm = minimum value of C factor for water erosion
c     wsyf = water stress yield factor
c     wvap = parm relating vapor press deficit to WA
c     vpth = threshold VPD (SPA) (F=1.)
c     vpd2 = VPD  value (KPA) / F2  1
c     ird = vegetation for crop (1) annual (2) perennial
```

```
* Alternative reading using a text buffer incrop for debugging:
      do 1112 ic = 1, mcrdb
        write (iolog,'(/,1x,a,i3)') 'ic=',ic
        do j=1,5
            read (15,'(a)') incrop(j)
            if (j.eq.1) print '(1x,a)',incrop(j)(1:16)
        end do
        read (incrop(1),'(i2,2x,a4,8f8.3)')
     1      icnum(ic), cpnm(ic), be(ic), hi(ic), to(ic),
     *      tb(ic), blai(ic), dlai(ic), dlp1(ic), dlp2(ic)
        read (incrop(2),'(16x,f8.3,16x,4f8.3)')
     *      gsi(ic), chtmx(ic), rdmx(ic), pt2(ic), cvm(ic)
```

```
        read (incrop(3),'(8x,f8.3)') wsyf(ic)
        read (incrop(4),'(64x,2f8.3)') wavp(ic), vpth(ic)
        read (incrop(5),'(f8.3,64x,i8)') vpd2(ic), ird(ic)
* spp  The above replaces this previous, more obscured version:
*spp    read (15,1111) icnum(ic), cpnm(ic), be(ic), hi(ic), to(ic),
*spp *      tb(ic), blai(ic), dlai(ic), dlp1(ic), dlp2(ic),
*spp *      gsi(ic), chtmx(ic), rdmx(ic), pt2(ic), cvm(ic),
*spp *      wsyf(ic),
*spp *      wavp(ic), vpth(ic),
*spp *      vpd2(ic), ird(ic)
 1111   format (i2,2x,a4,8f8.3,/,16x,f8.3,16x,4f8.3,8x,/,8x,f8
     *      .3,64x,/,64x,2f8.3,/,f8.3,64x,i8)
```

Calculations after reading crop.dat:

```
        cvm(ic) = alog(cvm(ic))
        if (be(ic).gt.0.) then
        temp = vpd2(ic) - int(vpd2(ic))
        vpd2(ic) = (1.-vpd2(ic)) / (temp-vpth(ic))
        vpd2(ic) = -1. / vpd2(ic)
        wac21(ic) = be(ic) * .01
        co21 = 330.
        co22 = pt2(ic)
        wac22(ic) = pt2(ic) - int(pt2(ic))
        call ascrv(wac21(ic),wac22(ic),co21,co22)
        pt1max = .01 * int(dlp1(ic))
        pt2max = .01 * int(dlp2(ic))
        dlp1(ic) = dlp1(ic) - int(dlp1(ic))
        dlp2(ic) = dlp2(ic) - int(dlp2(ic))
        call ascrv(dlp1(ic),dlp2(ic),pt1max,pt2max)
```

<u>dat(16): Tillage data base file till.dat</u>

Example (excerpt from till.dat):

```
  1    LISTRPLT   19.77     .15   10.00   40.00   75.00    1.00      .00      .00
  2    ROW PLT    20.00     .05    5.00   60.00   10.00     .86      .00      .00
  3    PLANT DR   17.54     .25   10.00   40.00   25.00     .17      .00      .00
  4    TRSPLANT   19.77     .15   10.00  500.00   75.00    1.00      .00      .00
  5                 .00     .00     .00     .00     .00     .00      .00      .00
  6    INJECT-P     .00     .00     .00    75.0     .00     .00      .00      .00
  7                 .00     .00     .00     .00     .00     .00      .00      .00
  8                 .00     .00     .00     .00     .00     .00      .00      .00
  9    IRSTRSCH     .00     .00     .00     .00     .00     .00      .00      .00
 10    SPREADER    7.66     .00     .00     .00     .00     .00      .00      .00
 11    SPRAYER     6.80     .00     .00     .00     .00     .00      .00      .00
 12    ANHYD AP    4.94     .15   13.00   75.00   25.00     .30      .00      .00
 13    --------     .00     .00     .00     .00     .00     .00      .00      .00
 14    --------     .00     .00     .00     .00     .00     .00      .00      .00
 15    LISTER      9.14     .80   25.00  100.00  150.00    1.00      .00      .00
 16    DISK BED    9.74     .70     .00  100.00   75.00    1.00      .00      .00
 17    ROWBUILD     .00     .50   15.00  350.00  300.00    1.78      .00      .00
 18    CULTPACK     .00     .10    5.00   40.00   25.00    1.78      .00      .00
 19    ROW CULT   13.52     .30   15.00   25.00  100.00    1.00      .00      .00
 20    FLD CULT   12.36     .30    6.00   50.00   25.00     .25      .00      .00
```

```
c***itnum = tillage number
c     till = tillage name
c     effmix = mixing efficiency of operation
      do 1113 it = 1, 83
        read (16,1114) itnum(it), till(it), effmix(it)
```

```
 1114   format (i4,4x,a8,8x,f8.3)
 1113 continue
```

dat(17): Pesticide data base file pest.dat

Example (excerpt from pest.dat):

```
  1 Aldrin            20000.0     .05     2.0     28.0    0.75      0.1
  2 Balan             10700.0    1.00    24.0     24.0    0.75     50.0
  3 Banvel                8.0     .65     9.0      8.0    0.75   4500.0
  4 Basagran             35.0     .60     2.0     10.0    0.75500000.0
  5 Benlate             200.0     .25     6.0     10.0    0.75      0.1
  6 Benzex            55000.0     .05     3.0    600.0    0.75      0.1
  7 Bidrin               20.0     .70    20.0      7.0    0.75  10000.0
  8 Bladex              168.0     .60     2.0     14.0    0.75    165.0
  9 Bolstar             550.0     .55     0.5     14.0    0.75     45.0
 10 Bravo              4000.0     .50    10.0     18.0    0.75      0.6
 11 Carbofos           1800.0     .90     3.0     25.0    0.75    145.0
 12 Chlordane        100000.0     .05     2.5    100.0    0.75      0.1
 13 Cotoran             100.0    1.00    12.0     12.0    0.75     90.0
 14 Counter            1000.0     .60     2.5      5.0    0.75     15.0
 15 Cygon                 9.0     .95     3.0      7.0    0.75  25000.0
 16 2,4-D                74.0     .45     9.0     10.0    0.75    900.0
 17 Dasanit           10000.0    1.00    24.0     24.0    0.75      .01
 18 DDT              240000.0     .05     4.0    120.0    0.75      0.1
 19 DEF                5000.0     .25     7.0     10.0    0.75      1.0
 20 Dieldrin          50000.0     .05     5.0   1400.0    0.75      0.1
```

```
c*****pnum = pesticide number
c     pname = pesticide name
c     skoc = soil partition coefficient
c     wof = wash off fraction
c     hl = half-life on foliage(days)
c     skk = half-life on ground (days)
c     efa = application efficiency
c     wsol = water solubility(ppm)
      do 1115 ip = 1, mpdb
        read (17,1116) ipnum(ip), pname(ip), skoc(ip), wof(ip), hl(ip),
     *      skk(ip), efa(ip), wsol(ip)
 1116   format (i3,a16,f11.1,f8.2,2f8.1,f8.2,f8.1)
        hl(ip) = exp(-.693/hl(ip))
        skk(ip) = exp(-.693/skk(ip))
        pab(ip) = 0.
 1115 continue
```

3.4.4. Daily weather input data (subr Clicon)

Input file names for daily rainfall and temperature measurements are given on the input control file (File.cio), which also associates measurement stations with subbasins. Input format for temperature, precipitation, radiation, relative humidity and wind speed are specified by records 5-7 on the options input control file (~.cod); an example is shown below. The options input control file also includes a choice, **iopwea**, to read precipitation data from separate files for each station or all stations from one file; and similarly for temperature data.

Weather data are either read (subr Clicon) or generated (subroutines Clicon, CLGEN4, Alpha9). Case control input File.cio (above) contains the number of rain (nrgage) and temperature (ntgage) stations, and the names of daily precipitation and temperature data files, which are read from devices rf(9+*k*) for *k*=1 to **nrgage** and tmp(27+*k*) for *k*=1 to **ntgage** in subr Clicon. Option **iopwfl** (see options input file ~.cod) allows all rainfall data to be read from one file, and all temperature data on another. If data are missing,daily values are generated by subroutines Clicon, Clgen4, and Alpha9 (which estimates fraction of total rainfall in 1/2 hour). See also notes on input bug fixes below.

In SWAT's standard version, daily radiation, relative humidity, wind speed and [CO2] are only simulated (subr CLGEN4). With added option **iopwea** (~.cod file), daily values for the first three of these can be read from a data file.

Daily weather values for subbasin k:		read?	generate?	options (*)
subp(k)	= rainfall (mm)	yes	yes	1
tmx(k)	maximum temperature (deg C)	yes	yes	1
tmn(k)	minimum temperature (deg C)	yes	yes	1
ra(k)	radiation (langleys)	yes(*)	yes	2
rhd(k)	relative humidity	yes(*)	yes	2
	wind speed	yes(*)	yes	2
co2(k)	carbon dioxide conc?		yes	

(*) Note on options **iopwfl**(1) and **iopwea**(2):

The options input file corn90dc.cod specifies (with **iopwfl**>0) that precipitation and temperature data are to be consolidated into one file each for pcp and tmp; and (with **iopwea**>0) that solar radiation, relative humidity, and wind speed are to be read from file sand6092.rad. Input format for weather data files are given on the options input file as follows (which may also be given as 'FREE'):

```
(5x,12f5.0)          temperature data format (file all.tmp)         (5
(4x,8f6.0)           rainfall data format (file rs6093.pcp)         (6
(14x,f5.0,12x,2f6.0) rad(ly/d), relhum(%), wind(m/s) (sand6092.rad) (7
```

Input control codes **nsim** and **msim** (read in ~.cod) control rain and temperature input as follows (for daily rainfall input) and msim (for daily max and min temperature input):

1 or 2: read or simulate , respectively, for entire basin

3 or 4: read or simulate, respectively, for each sub-basin

If a daily value < -97 is read, then it is replaced by a simulated value (subr CLGEN4); see also note on weather (rainfall and temperature) input data file bug (p. 38).

tmp(27+k): Read daily temperature min and max (ex. file all.tmp)

```
      if (iopwfl.eq.0) then
        do k=1,ntgage
          if (itfree.gt.0) then
            read (27+k,*) txmeas(k),tnmeas(k)
          else
            read (27+k,fmttmp) txmeas(k),tnmeas(k)
          end if
        end do
      else
        if (itfree.gt.0) then
            read (27+1,*) (txmeas(kfld),tnmeas(kfld),
     1          kfld=1,ntgage)
        else
            read (27+1,fmttmp) (txmeas(kfld),tnmeas(kfld),
     1          kfld=1,ntgage)
        end if
      end if
```

Example: temperature input file all.tmp (excerpt)
Format: (5x,12f5.1)
Contents: year; (tmn(j),tmx(j),j=1,6)
Begin:

```
1960   3.3 -1.1  8.9  -.6  8.9   .6  7.8  -.6  8.9 -1.1 10.0  1.7
1960  -1.1 -7.2  1.1 -8.3  1.7 -7.2   .6 -8.3 -1.1 -7.8  2.8 -7.2
1960  -2.2-10.0  2.2-10.6  -.6 -9.4   .6-10.0   .6 -9.4   .6-10.0
1960    .0-11.7   .6-12.2   .0-10.6   .6-12.2  -.6-12.8  1.1-12.2
1960  -2.8 -8.9  -.6 -8.9 -2.2 -8.9 -1.7 -9.4  2.8 -8.9 -1.7 -8.3
1960   9.4 -7.8  8.9 -9.4  8.9 -8.3  8.9 -9.4  8.3 -8.3  9.4 -8.9
1960  12.2 -5.0 13.3 -6.7 11.7 -5.0 13.9 -5.0 11.7 -3.9 15.0 -6.1
1960  13.3 -3.9 14.4 -4.4 12.2 -3.9 14.4 -5.6 12.8 -5.0 11.7 -6.1
1960   8.9 -3.9 16.1 -3.9 11.7 -3.3 14.4 -5.6 11.7 -3.9 15.0 -3.9
1960  10.0 -3.9 13.3 -3.9 11.7 -3.3 11.7 -4.4 10.6 -3.3 12.2 -3.3
1960  13.3 -1.7 13.9 -2.2 13.3 -1.7 13.9 -1.7 12.2 -2.2 13.3 -1.1
1960  14.4  7.2 15.6  7.8 15.6  7.8 14.4  7.2 13.9  6.1 18.3  9.4
1960   7.2  1.1  7.8  -.6  7.8  2.8  7.2   .0  6.7   .0 10.0  2.8
1960   3.9 -5.6  5.0 -5.0  5.6 -5.0  5.6 -3.9  5.6 -5.0  8.9 -2.8
1960  -3.9 -7.8  1.1 -6.1 -2.2 -6.7  -.6 -5.6 -1.7 -8.3   .6 -5.0
1960  -2.2-10.0 -2.2 -7.8 -1.7 -9.4 -2.8 -8.9 -2.2 -9.4 -1.1 -7.8
1960  -3.3 -9.4 -2.2-11.1 -1.7 -8.3 -2.8 -9.4 -2.2-11.1 -1.1 -7.2
1960  -7.8-12.2 -8.9-13.3 -7.2-12.2 -7.2-13.3 -5.6-12.8 -6.7-12.2
1960  -5.6-15.6 -4.4-17.8 -5.0-15.6 -2.8-16.1 -5.6-16.1 -3.9-16.1
1960  -3.3-16.1  -.6-17.2 -2.8-15.6 -1.1-17.2 -2.8-15.0  1.1-15.0
1960  -4.4-15.0  -.6-13.3 -3.3-13.3 -2.2-15.0 -2.2-13.3 -1.1-12.2
```

rf(9+k): Read daily precipitation (ex. file rs6093.pcp)

```
      if (iopwfl.eq.0) then
        do k=1,nrgage
          if (irfree.gt.0) then
            read (9+k,*) rmeas(k)
          else
            read (9+k,fmtpcp) rmeas(k)
          end if
      end do
    else
      if (irfree.gt.0) then
          read (9+1,*) (rmeas(kfld),kfld=1,nrgage)  ! -spp
      else
          read (9+1,fmtpcp) (rmeas(kfld),kfld=1,nrgage)  ! -spp
      end if
    end if
```

Example: precipitation input data rs6093.pcp (excerpt)
Format: (i4,8f6.1)
(Format is specified by fmtpcp on input control file corn90dc.cod)
Contents: year; daily precipitation at eight stations:
Buckl Grbnd Grns Hudson Kinsley Larned Pratt Trusd
Begin:

```
1960   2.5   1.3   2.3   1.5   1.3   1.0   1.0    .8
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0   1.8   1.3    .0    .0    .0    .5    .5
1960    .0    .0    .0    .0   1.0   1.0    .0    .0
1960  18.8  26.7  26.2  20.3  22.9  20.1  24.1  23.4
1960   4.6   7.6    .0   5.1   4.1   2.5   2.5   2.3
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960  12.4  11.4  10.2   7.4  12.2   8.4   8.4   8.1
1960    .0   1.3    .0   2.8   2.3    .8   1.3    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
1960    .0    .0    .0    .0    .0    .0    .0    .0
```

rad: Read radiation, rel. humidity and wind speed (ex. file sand6092.rad)

These data are read subject to added option **iopwea** (see options file ~.cod); previously, Swat only generated these data as described in the Swat manual.

Whether read or generated, wind speed measurements are assumed by Swat to be taken 10 m above land surface (ALS). The measurement at 10 m is used in Swat as $u_{10}(j)$ for subbasin j to evaluate evaporation in subr evap8 according to the Ritchie model as described in the Swat manual. From these, wind speed at 2 m ALS is estimated to be given by $u_2=f*u_{10}$ for a reduction factor f=0.749 according to a logarithmic wind speed profile described in Appendix E.

Read radiation (ly/d), relative humidity (pct) and wind speed (m/s) measured 10m above land surface (ALS):

```
      if (iopwea.gt.0) then
          if (isfree.gt.0) then
              read (iorad,*) radmea,relhum,wndspd
          else
              read (iorad,fmtrad) radmea,relhum,wndspd
          end if
          if (relhum.gt.-97. .and. relhum.le.100.) relhum = relhum/100.
          do k=1,Lu
              if (radmea.gt.-97. .and. radmea.lt.9999.) ra(k) = radmea
              if (relhum.gt.-97. .and. relhum.lt.9999.) rhd(k) = relhum
              if (wndspd.gt.-97. .and. wndspd.lt.9999.) u10(k) = wndspd
          end do
      end if
      do j=1,Lu
          u2(j) = 0.749*u10(j)
      end do
```

Example: radiation, rel. humidity and wind speed, sand6092.rad
Begin: (excerpt):

```
 1  1 1960   1 203.    6.   -7.   97. 10.43    .0
 1  2 1960   2 239.   -2.   -8.   75.  8.24    .0
 1  3 1960   3 239.    0.   -9.   75.  5.00    .0
 1  4 1960   4 166.    1.  -12.   90.  4.35    .0
 1  5 1960   5 162.   -1.   -9.   87.  4.60    .0
 1  6 1960   6 248.    7.   -9.   84.  5.02    .0
 1  7 1960   7 233.   12.   -6.   67.  5.15    .0
 1  8 1960   8 243.   13.   -1.   66.  5.32    .0
 1  9 1960   9 248.   14.   -3.   85.  4.52    .0
 1 10 1960  10 200.   10.   -2.   87.  4.73    .0
 1 11 1960  11  73.   13.   -3.  100.  7.38    .0
 1 12 1960  12 194.   12.    3.   96.  6.24   1.0
 1 13 1960  13  70.    5.   -1.  100.  4.71   7.4
 1 14 1960  14 147.    3.   -6.  100. 11.58  10.9
 1 15 1960  15 255.   -1.   -9.   84.  7.19    .0
 1 16 1960  16  40.   -3.   -9.   95.  4.37   5.8
 1 17 1960  17 150.   -4.  -11.   92.  8.56   4.3
 1 18 1960  18 187.   -9.  -14.   88. 11.58    .0
 1 19 1960  19 266.   -7.  -18.   85.  6.08    .0
 1 20 1960  20 276.   -2.  -16.   77.  5.13    .0
 1 21 1960  21 258.   -6.  -13.   63.  5.95    .0
```

3.4.5. Basin input files

bsn(11): general basin input (subr Read); ex. snake.bsn

See note on input variable **ffcb** data bug fix (3.4.1, "Changes to Swat").

Example: general basin input file snake.bsn

```
Rattlesnake basin                file snake.bsn (approx.)                    (1
 5302.40     1.00    1.000     0.00      0.33    0    0    0                 (2
```

Instructions for general basin input

```
character*80 title
      real mumax, kl, kn, kp, lambda0, lambda1, lambda2
```

```
C     DA = BASIN AREA (KM**2)
C     P2=RAINFALL CORRECTION(RATIO OF AVE AN RAINFALL TO AVE AN FOR GAGE
C     TP5 = TP-40 10 YR FREQ .5H RAINFALL(MM)
C     TP6 = TP-40 10 YR FREQ 6H  RAINFALL(MM)
C     TP24= NO YRS RECORD MAX .5H RAIN
C     YLT = LATITUDE
C     BRT = BASIN LAG TIME(D)
C     FFC = FRACTION OF FIELD CAP--INITIAL WATER STORAGE (0. ALLOWS
C     SWRRB TO ESTIMATE FFC)
      read (11,'(a)') title                                          (1
      read (11,5000) da, p2(1), bff, brt, ffcb,(npno(k),k=1,10)      (2
5000 format (f10.3,f6.3,3f8.3,10i4)
      write (kw,5600) bff, brt ! baseflow factor, basin lag time (days)
      write (kw,6000) ign      ! generator cycles
      write (kw,5800) iwst, isst ! water stats, sediment stats
      write (kw,6100) k1, k2, k3, k4, k5, k6, k7, k8, k9  ! random seeds
      af = 1000. * da
      da9 = 100. * da
      if (brt.lt..01) brt = 1.
      brt = 1. - exp(-1./brt)
      uob = 1. - exp(-ub)
C     IF MULT RAINGAGES ARE TO BE GENERATED, READ CENTROID COORDINATES
C     OF SUB-BASINS (KM).  (reads are commented out)
CREAD
c     read (11,6500) (xij(i),i = 1,lu)
c     read (11,6500) (yij(i),i = 1,lu)
*
c spp   if (ffcb.eq.0) ffc(i) = xx / (xx+exp(9.043-.002135*xx))
* The above line is replaced by the code below, which assigns ffcb to
ffc(i) for positive-valued ffcb as presumably intended
        if (ffcb.gt.0.) then
            ffc(i) = ffcb
        else
            xx = r(8)
            ffc(i) = xx / (xx+exp(9.043-.002135*xx))
        end if
```

lwq(8): Lake water quality input (subr Readlk)

Input instructions:
```
Input for lake toxic balance and lake phosphorus balance:
C     CPEST1 = INITIAL PESTICIDE CONCENTRATION (MG/M3)
C     VREAC = REACTION COEFFICIENT              (1/DAY)
C     VVOL = VOLATILIZATION COEFFICIENT         (M/DAY)
C     VPART = PARTITION COEFFICIENT             (M**3/G)
C     VSETL = SETTLING VELOCITY                 (M/DAY)
C     VRSUP = RESUSPENSION VELOCITY             (M/DAY)
C     VMIX = MIXING VELOCITY (DIFFUSION)        (M/DAY)
      read (8,5300) cpest1, vreac, vvol, vpart, vsetl, vrsup, vmix
5300 format (10f8.3)
C     CPEST2 = INITIAL PESTICIDE CONCENTRATION IN BOTTOM
C     SEDIMENTS (MM/M3)
C     VREAC2 = REACTION COEFFICIENTS      (1/DAY)
C     VBURY  = BURIAL VELOCITY            (M/DAY)
C     DACT   = DEPTH OF ACTIVE SEDIMENT LAYER (M)
      read (8,5300) cpest2, vreac2, vbury, dact
C     VSETLP = PHOSPHORUS SETTLING RATE (M/DAY)
C     PHOSL  = INITIAL TOTAL PHOSPHORUS CONCENTRATION IN LAKE (mg/L)
      read (8,5300) vsetlp, phosl
C     ISUBL = SUBBASINS WEATHER TO USE IN LAKE BALANCE
      read (8,5700) isubl
5700 format (20i4)
C     UMW = AVERAGE MONTHLY WIND SPEED (M/S)
C     EFFLQ(12) = AVERAGE DAILY EFFLUENT FLOW BY MONTH (M3/DAY)
C     EFFLT(12) = AVERAGE TEMP OF EFFLUENT BY MONTH (DEG C)
C     FLOWT(12) = AVERAGE TEMP OF NATURAL INFLOW BY MONTH (DEG C)
C     TD(12) = AVE MONTHLY DEWPOINT POINT TEMP (DEG C)
C     TLAKE = INTIAL LAKE TEMP (DEG C)
      read (8,6500) (umw(i),i = 1,12)
      read (8,6500) (efflq(i),i = 1,12)
      read (8,6500) (efflt(i),i = 1,12)
      read (8,6500) (flowt(i),i = 1,12)
      read (8,6500) (td(i),i = 1,12)
      read (8,5300) tlake
6500 format (12f6.2)
```

fig(9): Watershed configuration input (Main); ex. file rat.fig

Example input file rat.fig:

```
subbasin        1     1     1     1
subbasin        1     2     2     2
subbasin        1     3     3     3
subbasin        1     4     4     4
dsubbasin        1     5     5     5
subbasin        1     6     6     6
subbasin        1     7     7     7
subbasin        1     8     8     8
subbasin        1     9     9     9
subbasin        1    10    10    10
subbasin        1    11    11    11
subbasin        1    12    12    12
subbasin        1    13    13    13
subbasin        1    14    14    14
subbasin        1    15    15    15
subbasin        1    16    16    16
subbasin        1    17    17    17
subbasin        1    18    18    18
subbasin        1    19    19    19
subbasin        1    20    20    20
subbasin        1    21    21    21
subbasin        1    22    22    22
subbasin        1    23    23    23
subbasin        1    24    24    24
subbasin        1    25    25    25
subbasin        1    26    26    27
subbasin        1    27    27    27
subbasin        1    28    28    28
subbasin        1    29    29    29
route           2    30     2     1
add             5    31     2    30
route           2    32     3    31
add             5    33     3    32
route           2    34     5    33
add             5    35     5    34
add             5    36     4    35
route           2    37     6    36
add             5    38     6    37
add             5    39     7    38
route           2    40     8    39
add             5    41     8    40
route           2    42     9    41
add             5    43     9    42
add             5    44    11    10
route           2    45    14    44
add             5    46    14    45
route           2    47    13    46
add             5    48    13    47
add             5    49    43    48
route           2    50    12    49
add             5    51    12    50
route           2    52    16    51
add             5    53    16    52
route           2    54    18    53
add             5    55    18    54
add             5    56    19    55
route           2    57    21    56
add             5    58    21    57
route           2    59    20    15
add             5    60    20    59
add             5    61    17    60
route           2    62    22    61
add             5    63    22    62
add             5    64    58    63
route           2    65    25    64
add             5    66    25    65
route           2    67    24    66
add             5    68    24    67
```

```
route          2    69    23    68
add            5    70    23    69
add            5    71    26    70
route          2    72    27    71
add            5    73    27    72
route          2    74    28    73
add            5    75    28    74
route          2    76    29    75
add            5    77    29    76
finish         0
```

Instructions for watershed configuration input

```
c     CALCULATE THE NUMBER OF SUBBASINS, REACHES, AND RESERVOIRS
      nrch = 0
      nres = 0
      lu = 0
      lubtot = 0
      do idum = 1, mhyd
        read (9,'(a10,5i6,f6.3,i9)') a,
     *       icodes(idum), ihouts(idum), inum1s(idum),
     *       inum2s(idum), inum3s(idum), rnum1s(idum), inum4s(idum)
        if (a(1:1).eq.'*') go to 90
        if (icodes(idum).eq.0) go to 100
      end do
100   continue
c
c       icode = 1  subbasin command
c       icode = 2  route command
c       icode = 3  route reservoir command
c       icode = 7  read monthly flows (avg daily cms) from monthin(18)
        if (icodes(idum).eq.7) then
          read (9,(10x,6a)) monthin
          open (18,file=monthin)
          read (18,(a)) title
          read (18,5202) noyrs
5202      format (i6)
         'for each year, read 2 records of monthly flows (6e12.4)'
          do iy = 1, noyrs
c***format changed for pc swat from  1 rec/yr (12e12.4) to 2 recs/yr:
            read(18,5204)(flomon(inum1s(idum),iy,mo),mo=1,6)
            read(18,5204)(flomon(inum1s(idum),iy,mo),mo=7,12)
5204        format(6e12.4)
          end do
        end if
c
c       icode = 8  read in EPIC daily epd file
        if (icodes(idum).eq.8) then
          dart(ihouts(idum)) = rnum1s(idum)
          read (9,5201) dayin
5201      format (10x,6a)
          open (45+inum1s(idum),file=dayin)
          do ii = 1, 6
              read (45+inum1s(idum),'(a)') title
          end do
        end if
c
c       icode = 9  output to eve file for input to another SWAT run
        if (icodes(idum).eq.9) then
          do ii = 1, 6
```

```
               write (7,9900) title
          end do
        end if
c
c       icode = 10  read daily values with water in cms and rest in tons
        if (icodes(idum).eq.10) then
          read (9,5201) dayin
          call caps(dayin)
          open (55+inum1s(idum),file=dayin)
          do ii = 1, 6
               read (55+inum1s(idum),9900) title
          end do
        end if
```

sta(9): Measured streamflow and sediment yield at gaged station (Main)

```
      open (9,file=statin)
      if (iwst.ne.0.or.isst.ne.0) then
        kk = nbyr * 12
        sump = 0.
        summ = 0.
        do 810 i = 1, kk
          read (9,8400) wob(i), sob(i)
8400      format (10e12.4)
          wob(i) = wob(i) * 86400 / 35.3146
          sob(i) = sob(i) / 35.3146
c         convert m^3/month to m^3/s
c         wob(i) = wob(i) / 2592000.
c         wpd(i) = wpd(i) / 2592000.
c         write(7,777) wob(i), wpd(i)
c         777     format(2f16.6)
          summ = summ + wob(i)
          sump = sump + wpd(i)
  810   continue
COMPUTE STATS ON MEASURED AND PREDICTED MONTHLY WATER YIELDS
          if (summ.gt.0.) call vald25(wob,wpd,kk)
COMPUTE STATS ON MEASURED AND PREDICTED MONTHLY SEDIMENT YIELDS
          if (sump.gt.0.) call vald25(sob,spd,kk)
        end if
      end if
c     close(9)
```

3.4.6. Subbasin input data files (read by subr. Readinpt)

Example: files for the first subbasin are specified in file.cio as follows:

```
1 0   0 0  0    rat01.sub    rat01.rte    p5011.pnd chemical.chm  pratt.sol
     3-wsf.mgt    rat90.mco      rat.gw rat.wgn         1   2
```

These files are used to illustrate the input data descriptions given below.

<u>sub(10): General input data (ex. rat01.sub)</u>

Example: file rat01.sub

```
Rattlesnake Creek - Subbasin Data - #1                                          (1
.0426246   65.000              .000  42.297    .001    5.000  75.000    .060        (2
    .200     .000     .000     .000     .500  50.000    .030    .000                (3




                                            .000                                (13
```

```
C******READ BASIN DATA
C        FLU = FRACTION OF BASIN IN EACH SUBAREA
C        CN2 = II COND. SCS CURVE NUMBER
C        SALB = SOIL ALBEDO
C        SNO = INITIAL WATER CONTENT OF SNOW(MM)
C        CHL = MAIN CHANNEL LENGTH (KM)
C        CHS = MAIN CHANNEL SLOPE (M/M)
C        CHW = AVERAGE WIDTH OF MAIN CHANNEL (M)
C        CHK = EFF HYD CONDUCTIVITY OF MAIN CHANNEL
C        CHN = CHANNEL N VALUE
C        OVN = OVERLAND FLOW N VALUE
C        ELEV = MEAN ELEVATION ABOVE SEA LEVEL (M)
read (10,(a)) title
c****format change for swat pc
c         read (10,10301) flu(i), cn2(i), co2(i), sno(i), chl(1,i),
c     *        chs(i), chw(1,i), chk(1,i), chn(i), ovn(i), elev(i)
c         read (10,10302) ovn(i), elev(i)
c10301    format(f9.7,10f8.3)
   new version:
read (10,10301) flu(i), cn2(i), co2(i), sno(i), chl(1,i),
     *       chs(i), chw(1,i), chk(1,i), chn(i)
10301    format(f9.7,8f8.3)
if (co2(i).le.0.) co2(i) = 330.
         if (flu(i).le.0.) flu(i) = .000001
         if (chs(i).le.0.) chs(i) = .0001
         cn2(i) = cn2(i) + incrcn
         dx(i) = da * flu(i)
C RT = RETURN FLOW TRAVEL TIME (D).
C ENTERING ZERO ALLOWS SWRRB TO CALCULATE RT.
C CSS = SEDIMENT CONC IN RETURN FLOW
C ECP = USLE EROSION CONTROL PRACTICE FACTOR P.
C SL = AVERAGE SLOPE LENGTH FOR SUBBASIN I (M)
C STP = AVERAGE SLOPE STEEPNESS FOR SUBBASIN I (M/M)
```

```
C RSDIN = INTIAL RESIDUE COVER (KG/HA) - 0 ALLOWS SWRRB TO ESTIMATE
        read (10,10300) ovn(i),elev(i),rt(i), css(i), ecp(i), sl(i),
    1         stp(i), rsdin(i)
10300    format(10f8.3)
        if (stp(i).le.0.) stp(i) = .0002
        slsoil(i) = sl(i)
c****format change for swat pc
c        read (10,10300) (rfinc(i,mo),mo=1,12)
c        read (10,10300) (tmpinc(i,mo),mo=1,12)
c        read (10,10300) (radinc(i,mo),mo=1,12)
c        read (10,10300) (huminc(i,mo),mo=1,12)
c        read (10,10300) (co2inc(i,mo),mo=1,12)
         read (10,10300) (rfinc(i,mo),mo=1,6)
         read (10,10300) (rfinc(i,mo),mo=7,12)
         read (10,10300) (radinc(i,mo),mo=1,6)
         read (10,10300) (radinc(i,mo),mo=7,12)
         read (10,10300) (huminc(i,mo),mo=1,6)
         read (10,10300) (huminc(i,mo),mo=7,12)
         read (10,10300) (co2inc(i,mo),mo=1,6)
         read (10,10300) (co2inc(i,mo),mo=7,12)
```

rte(11): Routing input data (ex. rat01.rte)

Example: file rat01.rte
```
Subbasin Routing Data for #1 - Rattlesnake Creek Water shed                                    (1
   5.000   1.000    .001  22.297    .060  75.000    .500    .500                               (2
```

```
C        CHW = CHANNEL WIDTH(M)
C        CHD = CHANNEL DEPTH(M)
C        CHSS = CHANNEL SLOPE(M/M)
C        CHL2 = CHANNEL LENGTH(KM)
C        CHNN = CHANNEL N
C        CHK2 = EFF HYD CONDUCTIVITY OF ALLUVIUM (MM/H)
C        CHXK = CHANNEL USLE K FACTOR
C        CHC = CHANNEL USLE C FACTOR
         read (11,'(a)') title
         read (11,10300) chw(2,i), chd(i), chss(i), chl(2,i), chnn(i),
     *       chk(2,i), chxk(i), chc(i)
10300     format(10f8.3)
         if (chss(i).le.0.) chss(i) = .0001
c        if (chl(2,i).le.0.) chl(2,i) = .0010
```

pnd(12): Pond/reservoir data (ex. p5011.pnd)

Example: file p5011.pnd
```
p5011.pnd                                                       15:25 11:11 27Aug92  (1
    .150 109.000 100.000 110.000 200.000    .001    .000    .001    .001   1.000  (2
   1  12  10                                                                      (3
    .000    .000    .000    .000    .000    .000    .000    .000    .000    .000  (4
```

```
C        FP = FRACTION OF SUBBASIN THAT FLOWS INTO PONDS
C        SAX = TOTAL SURFACE AREA OF ALL PONDS IN SUBBASIN TO
C        PRINCIPLE SPILLWAY(HA)
C        VMX = RUNOFF VOLUME FROM POND CATCHMENT AREA REQUIRED TO
C        FILL EMPTY PONDS AT PRINCIPLE SPILLWAY(MM).
C        SAE = TOTAL SURFACE AREA OF ALL PONDS IN SUBBASIN AT
C        EMERGENCY SPILLWAY(HA)
C        VMX = RUNOFF VOLUME FROM POND CATCHMENT AREA REQUIRED TO
C        FILL EMPTY PONDS AT EMERGENCY SPILLWAY(MM).
C        V = INITIAL POND VOLUMES(MM)
C        SEPP = SEEPAGE THROUGH DAM (CONTRIBUTES TO FLOW) (M**3/DAY)
C        CS = INITIAL SEDIMENT CONCENTRATION IN PONDS (PPM)
C        CFP = NORMAL SEDIMENT CONCENTRATION IN PONDS (PPM)
C        HC = HYDRAULIC CONDUCTIVITY OF POND BOTTOMS (MM/H)
         read (12,'(a)') title
         read (12,10303) fp(i), sax(i), vmx(i), sae(i), vme(i), v(i),
     *       sepp(i), cs(i), cfp(i), hc(i)
         read (12,10304) iflod1(i), iflod2(i),
     *       ndtarg(i)
10303    format (10f8.3)
10304    format (3i4)
C******READ WETLAND DATA
C FW = FRACTION OF SUBBASIN THAT FLOWS INTO WETLANDS
C SAXW = TOTAL SURFACE AREA OF ALL WETLANDS IN SUBBASIN TO
C          PRINCIPAL SPILLWAY(HA)
```

```
C VMXW = RUNOFF VOLUME FROM WETLANDS CATCHMENT AREA REQUIRED TO
C        FILL EMPTY PONDS AT PRINCIPLE SPILLWAY(MM).
C SAEW = TOTAL SURFACE AREA OF ALL WETLANDS IN SUBBASIN AT
C        EMERGENCY SPILLWAY(HA)
C VMXW = RUNOFF VOLUME FROM WETLANDS POND CATCHMENT AREA REQUIRED TO
C        FILL EMPTY WETLANDS AT EMERGENCY SPILLWAY(MM).
C VW = INITIAL WETLANDS VOLUMES(MM)
C SEPW = SEEPAGE THROUGH DAM (CONTRIBUTES TO FLOW) (M**3/DAY)
C CSW = INITIAL SEDIMENT CONCENTRATION IN WETLANDS (PPM)
C CFW = NORMAL SEDIMENT CONCENTRATION IN WETLANDS (PPM)
C HCW = HYDRAULIC CONDUCTIVITY OF WETLANDS BOTTOMS (MM/H)
        read (12,10300) fw(i), saxw(i), vmxw(i), saew(i), vmew(i),
     *      vw(i), sepw(i), csw(i), cfw(i), hcw(i)
10300    format(10f8.3)
```

chm(13): Top ten pesticides input data (ex. chemical.chm)

Note: the basin file bsn(11) specifies up to ten pesticides in array npno.

Example: file chemical.chm

```
dummy chemical data file                                            (1
    .000    .000    .000                                            (2
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000
    .000    .000    .000                                            (3
```

Input instructions for ~.chm:

```
read (13,'(a)') title                                               (1
        do 20 j = 1, 10
C******READ PESTICIDE DATA
C         FFP = INITIAL PESTICIDE ON FOLIAGE (KG/HA)
C         GP = INITIAL PESTICIDE ON GROUND (KG/HA)
C         ERP = ENRICHMENT RATIOS FOR PESTICIDES
For each pesticide j=1,10:
          read (13,10300) ffp(j,i), gp(j,i,1), erp(j,i)             (2
C******READ NUTRIENT AND PESTICIDE DATA
C        WN = ORGANIC N CONCENTRATION IN UPPER LAYER(G/T)
C        WPO = PHOSPHORUS CONCENTRATION IN UPPER LAYER(G/T)
C        AP = CONCENTRATION OF LABILE (SOLUABLE) P IN UPPER LAYER (G/T)
        read (13,10300) wn(1,i), wpo(1,i), ap(1,i)                  (3
```

sol(14): Soil input data (ex. pratt.sol; houston.s93)

Example: pratt.sol
```
PRATT                                                                          (1
  4     .17   25.00PRATT                                                       (2
  10.00  304.80 1016.00 1524.00      .00      .00      .00  depth mm to layer bottom(3
   1.47    1.47    1.50    1.52      .00      .00      .00      bulk density  T/m^3 (4
    .11     .11     .10     .10      .00      .00      .00      avail water cap m/m (5
 330.20  330.20  330.20  330.20      .00      .00      .00          Sat. Cond. mm/h (6
   5.00    5.00    7.00    4.00      .00      .00      .00          Clay cont. %    (7
    .47     .47                                                     organic C       (8
  10.00   10.00    5.00    5.00                                     Init. [NO3]     (9
    .00     .00     .00     .00      .00                            part. size dist(10
1524.00                                                             max root depth (11
```

The soils data files extracted from the soils data base via program Runsoils is not in the format read below, so Runsoils output must be converted to the format shown below, at least for this PC version of Swat. Examples of these two forms are shown below. --spp

```
C       READ SOIL HYDRAULIC PROPERTIES FOR EA SUB-BASIN.  SOIL IN EACH
SUB-BASIN IS DIVIDED VERTICALLY INTO A MAXIMUM OF 10 LAYERS.  EACH
HYDRAULIC PROPERTY REQUIRES 1 RECORD.

*   input format:
5200    format(i4,2f8.3,a)
5300    format(10f8.3)
read (14,'(a)') title                                                (1
*   no. soil layers, USLE erosion K-factor, silt content (%), soil name:
read (14,5200) ns(i), ek(i),sil(i),snam(i)                           (2
        nn = ns(i)
read (14,5300) (z(j,i),j = 1,nn)    !depth to bottom of layers (mm)   (3
read (14,5300) (por(j,i),j = 1,nn)  !bulk density (T/m^3)             (4
read (14,5300) (awc(j,i),j = 1,nn)  !available water capacity (m/m)   (5
read (14,5300) (sc(j,i),j = 1,nn)   !saturated conductivity (mm/h)    (6
read (14,5300) (cla(j,i),j = 1,nn)  ! clay content (pct)              (7
read (14,5300) (cbn(j,i),j = 1,nn)  ! organic carbon content (pct)    (8
read (14,5300) (wno3(j,i),j = 1,nn) !initial [NO3] (g/T)              (9
read (14,5300) (psz(j,i),j=1,5)     !particle size distribution      (10
read (14,5300) zmx(i)               !maximum rooting depth (mm)      (11

            Calculation of soil properties:
        san(i)=0.  !sand content (pct)
        rock(i)=0.  !rock fragments (pct)
        salb(i)=.15  !moist soil albedo
        do 877 j = 1,nn
          if (por(j,i).le.1.e-6) por(j,i) = 1.3  !default bult density
 877    continue
        rock(i) = exp(-.053*rock(i))
        if (nn.eq.1) then
          z(2,i) = z(1,i)
          z(1,i) = 10.
          por(2,i) = por(1,i)
          awc(2,i) = awc(1,i)
          sc(2,i) = sc(1,i)
          cbn(2,i) = cbn(1,i)
          cla(2,i) = cla(1,i)
          wno3(2,i) = wno3(1,i)
          ns(i) = 2
          nn = 2
        endif
c5300 format (27x,10f12.2)
        if (cbn(3,i).le.0) then
C****CALCULATE CBN FOR LOWER LAYERS IF ONLY HAVE UPPER LAYER
          xx = z(2,i)
          do 40 l = 3, nn
            dg = (z(l,i)-xx)
            if (cbn(l,i).eq.0.) cbn(l,i) = cbn(l-1,i) * exp(-.001*dg)
            xx = z(l,i)
   40     continue
        end if
        cv(i) = rsd(i)
        sil(i) = sil(i) / 100.
        cla(1,i) = cla(1,i) / 100.
        san(i) = 1.0 - cla(1,i) - sil(i)
        psz(1,i) = (1.-cla(1,i)) ** 2.49 * san(i)
        psz(2,i) = .13 * sil(i)
        psz(3,i) = .20 * cla(1,i)
        if (cla(1,i).le..5) then
          if (cla(1,i).lt..25) go to 140
          psz(4,i) = .28 * (cla(1,i)-.25) + .5
          go to 150
        end if
        psz(4,i) = .57
        go to 150
  140   psz(4,i) = 2. * cla(1,i)
  150   psz(5,i) = 1. - psz(1,i) - psz(2,i) - psz(3,i) - psz(4,i)
        do 160 j = 1, nn
          if (j.eq.1) cla(1,i) = 100. * cla(1,i)
          wp(j,i) = 0.4 * cla(j,i) * por(j,i) / 100.
          up(j,i) = wp(j,i) + awc(j,i)
          por(j,i) = 1. - por(j,i) / 2.65
          if (up(j,i).ge.por(j,i)) then
            up(j,i) = por(j,i) - .05
            wp(j,i) = up(j,i) - awc(j,i)
            if (wp(j,i).le.0.) then
              up(j,i) = por(j,i) * .75
              wp(j,i) = por(j,i) * .25
            end if
          end if
  160   continue
```

Example 2: soil data file Houston.s93

```
Soils Data
   3   0.320  92.000HOUSTON BLACK    D
  10.000 609.6002032.000
   1.400   1.400   1.500
   0.170   0.170   0.170
   1.524   1.524   1.524
  50.000  50.000  50.000
   1.744   1.744
  10.000  10.000   5.000
```

```
    0.00    0.00    0.00    0.00    0.00
2032.000
```

The following (below the column numbers) shows the form of the Houston Black soil data extracted from the soils data base via Runsoils (file houston.s94):

```
         1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
 HOUSTON BLACK
 2HOUSTON BLACK
 Maximum rooting depth                      .00
 Texture                                    .00
 Depth (mm)                              609.60     2032.00         .00         .00
 Bulk Density (t/m**3)                     1.40        1.50         .00         .00
 Available Water Cap (m/m)                  .17         .17         .00         .00
 Sat. Cond. (mm/h)                         1.52        1.52         .00         .00
 Organic Carbon Content(%)                 1.74
 Clay Content (%)                         50.00       50.00         .00         .00
 Silt Content(%)**                          .00
 Sand Content(%)**                          .00
 Rock Fragments(%)**                        .00
 Moist Soil Albedo**                        .00
 Dry Soil Albedo**                          .00
 USLE Erosion K-Factor                      .32
 Salinity                                   .00
 Initial NO3 Conc (g/t)                   10.00        5.00
```

<u>mgt(17): Management input data, i.e. schedules (ex. 3-wsf.mgt)</u>

Example: file 3-wsf.mgt

```
Rattlesnake Creek Watershed - Crop Data for Wheat-Sorghum-Fallow: 3-wsf.mgt
1  3       0      0
   6  25           5 1650.00   4    1.00  660.00     .00   0     0     0

   5   1           1 1650.00   3            0.00   30.00   0     0     0
  10  10           5 1650.00   3            0.00   30.00   0     0     0

   9  20           1 1650.00   4            0.00   30.00   0     0     0


File format:
title                  (a)
igro,nrot,nmgt,nptot   (i1,i3,2i4)
for iro=1,nrot:        (2i4,2(f8.3,i4),3f8.3,i4,2f6.3)
--mo ida  husc itill      hu ncrp   airg    anit    phos inp  pamt cn2up
```

Note: For winter crops, do the following, as shown above.
  set igro=1 to indicate that plants are growing from the first of the year; specify the harvest operation (5) first; follow this with the planting operation (1).

```
C IF PERENNIAL VEGETATION, ENTER START AND END OF GROWING SEASON
C       IN PLACE OF PLANTING AND HARVEST
c       nrot =number of years of rotation in simulation
c       mo = month of planting
c       ida = day of planting
c       husc = heat unit scheduling
c       itill =operation/tillage code number
```

```
c       hu = heat units to maturity
c       ncrp = crop identification number
c       airg = amount of irrigation applied(mm)
c       anit = amount of nitrogen applied (kg/ha)
c       phos = amount of phosphorus applied (kg/ha)
c       inpest = pesticide number
c       pamt = pesticide amount applied(kg/ha)
c       cn2up = SCS runoff curve number update
      read (17,'(a)') title
      read (17,5802) igro(i), nrot(i), nmgt(i), nptot(i)
5802  format (i1,i3,12i4)
      if (igro(i).eq.1) then
          alai(i) = 1.
          dm(i) = 3.
          g(i) = .2
          npl = 1
      end if
      do iro = 1, nrot(i)
          read (17,5800) mo, ida, husc, itill, hu, ncrp, airg, anit,
     *        phos, inpest, pamt, cn2up
5800     format (2i4,f8.3,i4,f8.3,i4,3f8.3,i4,2f6.3)
      end do
ITILL   operation
1     plant
2     irrigate
3     fertilize
4     apply pesticide
5     harvest and kill
6     till
7     harvest only
8     kill only
9     grace
0     initialize cumulative operations for all classes (1-9) to zero.
```

<u>mco(18): Mgmt codes, i.e. stress factor threshholds:(ex. rat90.mco)</u>

Specify stress factor threshholds for rrigating and fertilizing "automatically" during crop growth seasons.

Example: file90.mco

```
Rattlesnake Creek Watershed - MCO file for Subbasin 1                          (1
   1     0.9     0.15       0       0.       0                                  (2
   .500           1.000  21   2   1                                             (3
        0        0        0        0        0        0  wures(mo,i)  1-6      (4
        0        0        0        0        0        0  wures(mo,i) 7-12      (5
        0        0        0        0        0        0  wurch(mo,i)  1-6      (6
        0        0        0        0        0        0  wurch(mo,i) 7-12      (7
        0        0        0        0        0        0  wushal(mo,i)  1-6     (8
        0        0        0        0        0        0  wushal(mo,i) 7-12     (9
        0        0        0        0        0        0  wudeep(mo,i)  1-6    (10
        0        0        0        0        0        0  wudeep(mo,i) 7-12    (11
```

title record
 irr   wsf   efi irrsub   wurtn irtnsb   (i4,2f8.3,i8,f8.3,i8)
 ansf   fnmx   anmx   idmn ipman lafert forgn forgp (3f8.3,3i4,2f8.3)
< then 8 records as above, (6f10.1) >

```
        read (18,'(a)') title
```

```
c    irr = irrigation = 1 automatic by water stress
c       = -1 input date and amount
c       = 0 no irrigation
c    wsf = if irr=1, wsf = water stress factor
c    efi = irrigation runoff ratio
c    irrsub = subbasin to remove the water from
c    wurtn = fraction of water use that returns to stream
c    irtnsb = subbasin that the flow returns to
c    ansf = nitrogen stress factor to trigger fertilization
c    fnmx = soil N level to fertilize to (kg/ha)
c    anmx = max amount of N that can be applied in one year (kg/ha)
c    idmn = minimum interval between fertilizations (days)
c    ipman = P fertilizer management for auto fertilization
c    ipman = 3, high management-restores P in upper 2 layers to 30 ppm
c    ipman = 2 medium management-restores P in upper 2 layers to 20 ppm
c    ipman = 1 low management-restores P in upper 2 layers to 10 ppm
c    lafert = soil layer that fertilizer is applied to
c    forgn = fraction of organic N in fertilizer
c    forgp = fraction of organic P in fertilizer
c    wures, wurch, wushal, wudeep are average monthly water use
c    from the reservoir, reach, shallow and deep aquifer storages (ha-m)
*   Formats:
7200  format (i4,2f8.3,i8,f8.3,i8)
7201  format (3f8.3,3i4,2f8.3)
7203  format (6f10.1)  ! ****format change for swat pc
*
read (18,7200) irr(i), wsf(i), efi(i), irrsub(i), wurtn(i), irtnsb(i)
read (18,7201) ansf(i), fnmx(i), anmx(i), idmn(i), ipman(i),
     *       lafert(i), forgn(i), forgp(i)
read (18,7203) (wures(mo,i),mo = 1,6)
read (18,7203) (wures(mo,i),mo = 7,12)
read (18,7203) (wurch(mo,i),mo = 1,6)
read (18,7203) (wurch(mo,i),mo = 7,12)
read (18,7203) (wushal(mo,i),mo = 1,6)
read (18,7203) (wushal(mo,i),mo = 7,12)
read (18,7203) (wudeep(mo,i),mo = 1,6)
read (18,7203) (wudeep(mo,i),mo = 7,12)
```

gw(19): Groundwater input data (ex. rat.gw)

Results based on data from this file, evaporation from groundwater, can be replaced by results from Modflow if Swat and Modflow are coordinated.

Example: file rat.gw

```
DUMMY GROUNDWATER DATAFILE                                                              (1
    .00000     .00000     .00000     .15000   1.00000     .00000   1.00000     .00000  (2
 100.00000                                                                              (3
```

```
C       GWHT = INITIAL GROUNDWATER HEIGHT (M)
C       GWQ = INITIAL GROUNDWATER FLOW CONTRIBUTION TO STREAMFLOW
C       ABF = ALPHA FACTOR FOR GROUNDWATER
C       SYLD = SPECIFIC YIELD
C       DELAY = GROUNDWATER DELAY (DAYS)
C       RCHRGC = FRACTION (0-1) OF ROOT ZONE PERC THAT PERCOLATES
C       PAST SHALLOW GW INTO DEEP GW
C       REVAPC = REVAP COEFF-FRACTION OF RECHARGE (ROOT ZONE PERC)
C       THAT GOES TO REVAP
C       REVAPMN = REVAP STORAGE
```

```
        read (19,'(a)') title
        read (19,5301) gwht(i), gwq(i), abf(i), syld(i), delay(i),
     *      revapc(i), rchrgc(i), revapmn(i)
        read (19,5301)  deepst(i)
5301  format(8f10.4)  ! c*****format change for swat pc
        abf1(i) = exp(-abf(i))
        delay(i) = exp(-1./(delay(i)+1.e-6))
        rchrg(i) = gwq(i)
```

wgn(20): Weather generation input (ex. rat.wgn)

Example: file rat.wgn

```
  KS  PRATT KS                                                                         (1
  72.900 125.730  45.000  37.600 330.000   1                                           (2
  6.20   9.50 14.50 21.20 25.70 31.40 34.50 33.70 28.90 23.00 14.00   8.10             (3
 -6.80 -4.40   -.20   6.30 11.60 17.00 19.80 18.80 14.10   7.90    .30 -4.70           (4
   .40    .34    .27    .18    .13    .10    .07    .08    .13    .17    .23    .32    (5
249.00315.00414.00516.00563.00639.00636.00573.00486.00368.00279.00229.00               (6
  8.13   2.54   8.13 42.16 24.38 34.04 41.66 29.97 29.72 12.95 19.56   9.40            (7
   .09    .11    .13    .17    .22    .20    .19    .18    .15    .11    .09    .09    (8
   .26    .31    .33    .34    .43    .43    .37    .35    .35    .42    .37    .32    (9
   .00    .00    .00    .00    .00    .00    .00    .00    .00    .00    .00    .00    (10
  4.57   7.11   8.13   9.14 10.67 11.94 10.41 10.41 10.92 12.19   7.11   6.35          (11
  5.08   8.38   9.40 14.73 12.95 14.48 12.45 12.95 14.22 17.27   9.14   7.62           (12
  1.30   1.36    .67   4.95    .98   1.03    .98   1.71   1.58   2.65   2.32   1.08    (13
 -6.34 -3.88 -2.01   4.20 11.20 15.77 17.58 16.47 12.12   6.56   -.20 -4.31            (14
  5.57   5.65   6.47   6.27   5.65   5.91   5.11   4.94   5.32   5.47   5.46   5.46    (15
```

```
C       TP5 = TP-40 10 YEAR FREQ. .5 H RAINFALL (MM)
C       TP6 = TP-40 10 YEAR FREQ. .6 H RAINFALL (MM)
C       TP24 = NO YEARS RECORD MAX .5 H RAINFALL (MM)
C       YLT = LATITUDE
        read (20,'(a)') title                                      ! (1
        read (20,(4f8.3)) tp5(i), tp6(i), tp24(i), ylt(i)          ! (2
C       OBMX = AV MO MAX TEMP C; OBMN = AV MO MIN TEMP C
        read (20,(12f6.3)) (obmx(mo,i),mo = 1,12)                  ! (3
        read (20,(12f6.3)) (obmn(mo,i),mo = 1,12)                  ! (4
C       CVT = COEF OF VAR FOR MO TEMP
        read (20,(12f6.3)) (cvt(mo,i),mo = 1,12)                   ! (5
        do 777 mo = 1, 12
777          if (cvt(mo,i).gt.0.36) cvt(mo,i) = 0.36
C       OBSL = AV MO SOL RA
        read (20,(12f6.3)) (obsl(mo,i),mo = 1,12)                  ! (6
C       WI = MO MAX .5H RAIN FOR PERIOD OF RECORD(MM)
        read (20,(12f6.3)) wim                                     ! (7
nsim  condition
1     measured single raingage
2     simulated single raingage
3     measured multiple raingages
4     simulated multiple raingages
msim  condition
1     measured single temperature for entire basin
2     simulated single temperature for entire basin
3     measured temperature for each sub-basin
4     simulated temperature for each sub-basin
C PRW(1)= MONTHLY PROBABILITY  OF WET DAY AFTER DRY DAY.
C PRW(2)= MONTHLY PROBABILITY OF WET DAY AFTER WET DAY.
C RST = MONTHLY MEAN EVENT, ST DEV, AND SKEW COEF OF DAILY RAINFALL.
        read (20,(12f6.3)) (prw(1,mo,i),mo = 1,12)                 ! (8
        read (20,(12f6.3)) (prw(2,mo,i),mo = 1,12)                 ! (9
        read (20,(12f6.3)) (wvl(mo,i),mo = 1,12)                   !(10
        read (20,(12f6.3)) (rst(mo,1,i),mo = 1,12)                 !(11
        read (20,(12f6.3)) (rst(mo,2,i),mo = 1,12)                 !(12
        read (20,(12f6.3)) (rst(mo,3,i),mo = 1,12)                 !(13
        read (20,(12f6.3)) (dewpt(mo,i),mo = 1,12)                 !(14
        read (20,(12f6.3)) (uavm(mo,i),mo = 1,12)                  !(15
c       do 1701 j = 2, 10
c       read (20,(12f6.3)) (vobmx(mo,j),mo = 1,12)
```

```
c        read (20,(12f6.3)) (vobmn(mo,j),mo = 1,12)
```

Appendix B: coordinating Swat and Modflow programs

Running Swat and Modflow

Swat (executable file Swtmod95.exe) can be run with or without calling Modflow. Swat writes a summary of hydrologic results for each aquifer time step to a balance file, which can be read on a subsequent run of Modflow (executable file Modswb95.exe). As an example, Swat and Modflow are executed separately at the DOS prompt for a case named "corn90dc" as follows:

```
swtmod95   >corn90dc.jou
modswb95   <corn90dc.rsp   >>corn90dc.jou
```

In these command lines, files corn90dc.rsp and corn90dc.jou represent redirected input (from the keyboard) and output (to the terminal); the second command line indicates with ">>" that the redirected output is to be appended to that of the first command line.

For this case, the conditions to be simulated by Swat correspond to files listed on file corn90dc.cio; those to be simulated by Modflow correspond to files listed on file corn90dc.rsp. Just prior to executing Swat, file corn90dc.cio must be renamed "file.cio", which is read by Swat at the beginning of execution. This file associates the names of two files with case corn90dc: a standard output file corn90dc.std written by Swat and an options input file corn90dc.cod described in Appendix A. This input file specifies options for Swat-Modflow coordination and added Swat options, including the name of the balance output file, (corn90dc.bal), on which Swat's results are summarized for each aquifer time step in a format designed to be read by Modflow on a subsequent run.

The "response" file corn90dc.rsp (shown below) provides the keyboard responses to Modflow's queries. Modflow first prompts for the case name ("corn90dc" on line 1) which becomes the prefix for its standard output file (~.prn) and a "log" file (~.log). Modflow then prompts for the names of input files to be read by its various packages. File tbss.swb is shown as the input file for the MODSWB package; the last line of this file should contain the balance file name corn90dc.bal, from which Modflow obtains Swat's results for this case. Swat and Modflow can optionally be run together (option **iopswt**), in which case Swat calls Modflow at the end of each aquifer time step after completing daily watershed simulations, as previously summarized. In this case, the two commands shown above are replaced by

```
swtmod95   <corn90dc.rsp   >corn90dc.jou
```

The following is a listing of "response" file corn90dc.rsp:

```
corn90dc                                           case name (~.log, ~.prn, ~.rsp)
6061swat                                           .bas unit   1 Basic package
trans                                              .bcf unit   7 Block-centered flow
6084wr                                             .wel unit   5 Well: .7*gw rights
6084                                               .riv unit   8 River
6084                                               .evt unit  65 evaporation
```

```
tbss                                              .swb unit  66 soil water balance
6084                                              .rch unit  31 Recharge
trans                                             .sip unit   9 Strongly implicit
trans                                             .oc  unit  11 Output control
6084                                              .str unit  13 Streamflow
```

The file device numbers shown above in the response file listing identify the device numbers used in the individual package input files, especially for reading arrays as described in the Modflow manual. These device numbers are somewhat arbitrary with certain constraints. They must lie within the range allowed for the compiler (1-255 for Lahey) and must not conflict with the use of device numbers for other files such that input files are annihilated. The range 61-74 is recommended as a safe range for these device numbers, although the use of many numbers outside this range may cause no harm, as with the above case. The file device numbers used by Swat and Modflow are listed below.

Compiling and linking Swat and Modflow

Compilation under Lahey Fortran

The Lahey compiler command f77l3 is used with specified options to compile Fortran source files named "~.for"; e.g., file modmain.for is compiled by issuing the following command at the DOS prompt:

F77L3 modmain /n0/n7/**B**/nC/D/nF/H/I/nK/L/O/P/nQ1/nQ2/nQ3/R/nS/nT/V/W/X/nZ1

The compiler responds as follows, listing the definitions for the above options; options that are not defaults are shown in **boldface**.

```
Compiling modmain.for, a Standard Format Source File

OPTION  DESCRIPTION                          OPTION  DESCRIPTION
/n0   - Standard FORTRAN 77 IMPLICIT         / L   - Line-number traceback table
/n2   - Generate 387 constants and code      / P   - Protect constant arguments
/n4   - No 80486 optimizations               /nQ1 - "Unlimited" NDP stack
/n7   - Optimize inter-statement             /nQ2 - No protected-mode RPC
/nA2 - No allocatable array checking         /nQ3 - No real-mode RPC
/ B   - Check array subscript Bounds         / R   - Remember local variables
/nC   - Ignore nonstandard usage             /nS   - No SOLD file created
/nC1 - INTEGER constants 4 bytes             /nT   - INTEGER*4, LOGICAL*4 default
/ D   - DIRECT files without headers         / V   - VAX Fortran interpretation
/ H   - Hardcopy source listing              / W   - Display Warning messages
/ I   - Check subprogram Interfaces          / X   - Cross reference & allocation
/nK   - Generate 80x87 code                  /nZ1 - Better SOLD debugging

Compiling line 3:       program modmain
```

A batch file named f77log.bat, listed below, is actually used to accomplish the above with the command

```
f77log modmain
```

The name "modmain" on the command line above is passed as argument "%1" to batch file f77log.bat, which is used to direct compilation of file modmain.for and to retain results of the compilation on file modmain.log.

```
echo off
rem file f77log.bat  spp  jan 14 1993
if not "%1"=="" goto begin
echo Compile Fortran 77 source file under Lahey f77l-em/32 v.5
echo.
echo Usage: f77log file_name
echo This invokes the Lahey v.5 compiler with a particular set of switches, i.e.
echo F77L3 file_name /n0/n7/B/nC/D/nF/H/nI/nK/L/O/P/nQ1/nQ2/nQ3/R/nS/nT/V/W/X/nZ1
echo.
echo Note: specify source file name without extension ".for".
echo       Compiler diagnostics are redirected to file_name.log
echo       Compiler switch definitions will also appear there; or to see only
echo       the definitions, type:
echo          f77l3
echo.
echo Once compiled, link object modules file1.obj[, file2.obj,...] to produce
echo an executable file file1.exe as follows:
echo       386link file1[,file2,...] -symbol
echo.
echo The -symbol switch includes a symbol table in the executable module.
echo The -exe switch in the following example specifies an executable module
echo name other than the first object file's name, e.g. progrm.exe below:
echo       386link file1[,file2,...] -symbol -exe progrm.exe
echo.
echo For more information on compiling and linking see Lahey Programmer's Reference.
goto end
:begin
if exist %1.for goto compile
echo File %1.for not found.
goto end
:compile
echo Compile file %1.for under Lahey v.5; diagnostics directed to %1.log
F77L3 %1 /n0/n7/B/nC/D/nF/H/I/nK/L/O/P/nQ1/nQ2/nQ3/R/nS/nT/V/W/X/nZ1 >%1.log
:end
echo.
exit
```

LinkingSWAT and MODFLOW

To link Swat and Modflow so that Swat may optionally call Modflow, the command **lswtmd95** is issued at the DOS prompt, which produces the executable file swtmod95.exe. The batch file Lswtmd95.Bat refers to file Swtmod95.Lnk, which lists names of all object files associated with Swat and Modflow that were produced by compiling the source files and are to be linked. Files Lswtmd95.Bat and Swtmod95.Lnk are as follows.

file Lswtmd95.Bat:

```
rem file lswtmd95.bat spp  26 May 95  Link programs Swat & Modflow
rem Lahey v. 5.1:
rem note: the "@ below refers to file swtmod95.lnk, which lists object files.

386link @swtmod95 -symbol -lib graph3 -stub runb -exe swtmod95.exe
```

file swtmod95.lnk:

```
! file swtmod95.lnk  lists object files for linked Swat and Modflow, and is
! referred to by lswtmd95.bat.
!
swatmain    ! main
blockd,cliconsp,crpmdspp,readinpt,subbasin !individual Swat modules
swata-e,swatf-o,swatp-r,swats-y ! Swat modules grouped alphabetically
hydbal,soilstat !Swat hydrologic balance; pass fluxes between Swat & Modflow
penman              !independent calculation of ref. ET (Penman-Monteith).
modflo   ! Modflow's main routine, changed into a subroutine called by Swat
modbas
modbcf
```

```
modevt    ! modified to summarize evap from aquifer for each subbasin
modrch
modsrf    ! version of WELL package for surface water diversions
modsub    ! includes WELL module, modified to allow Q to change at each time step
modutl    ! utility package
modpcg2   ! more or less standard Modflow
modswb    ! module added to coordinate watershed with stream & aquifer
modstrdw  ! version of STREAM package w/ option for diffusive routing
strmrc    ! subroutines associated with diffusive routing option
pltwav    ! routine to display flood wave simulation
```

Similarly, the command **Lmdswb95** links Modflow as a stand-alone program, producing the executable file Modswb95.exe. The batch file Lmdswb95.bat refers to file Modswb95.lnk, which lists the names of all object files associated with Modflow. Files Lmdswb95.bat and Modswb95.lnk are as follows.

a) file Lmdswb95.bat:

```
386link @modswb95 -symbol -lib graph3 -stub runb -exe modswb95.exe
```

b) file Modswb95.lnk:

```
modmain    ! mainline
modflo,modbas,modbcf,modevt,modrch,modsub,modutl,modpcg2
modsrf,modswb,modstrdw,strmrc,pltwav
```

Data file device numbers used for Swat and Modflow

A large number of input and output data files are coordinated for a combined execution of Swat and Modflow. The Fortran code for these programs is compiled and linked under Lahey Fortran, which was installed with an allowable maximum of 255 data files open concurrently on device numbers from 1 to 255. Swat file device numbers run from 1 to 57, except for 70, 77, and 80-81, as listed below. Modflow currently uses device numbers 1 (Basic package input file), 116 (standard output ~.prn), 117 (log file ~.log), 150-161 (unformatted output files for "saving" results), 217-220 (summary files), and a set of device numbers specified by the model input files. For these, the range 61-74 is recommended to avoid conflict with other data files. Although many numbers within the range 1-255 could be chosen other than those already mentioned above, the range 61-74 is recommended as one safe set of choices. The main hazard is in specifying a number that coincides with an output file device number so that your input file is written over. The input files shown in the "response" file corn90do.rsp for the Rattlesnake Creek watershed model in the above example refer to numbers outside this range with no apparent conflict.

Swat and Modflow file device numbers are listed below, categorized by program location and source file in which data files were opened.

<u>Swat file device numbers:</u>

Main program (file swatmod1.h, "included" in file swatmain.for): these were added as part of changes to Swat and coordination with Modflow.

50 iobal: output to hydrologic balance file (~.bal), named in ~.cod file, written in subr Hydbal, to be read by Modflow's Modswb package.

51 iolog

52 iopcp

53 iotmp
54 iomat
55 iosol
56 ioet
57 iorad

Subr open (file swatf-o.for): mainly basin-wide input

1 rsvout
2 file.cio
3 sbsout
4 Lwqout
5 pestout
6 stdout (~.std): Swat's standard output file
7 event
8 rchout
10 codedat (~.cod)
11 basndat (~.bsn)
70 kevin.out
77 bigsub

Subr open2 (file swatf-o.for): data base (~.dat) and subbasin input files

10 subdat (~.sub)
11 routdat (~.rte)
12 ponddat (~.pnd)
13 chemdat
14 soildat (~.sol)
15 cropdat: file crop.dat, crop data base
16 napdat
17 mgtdat (~.mgt): crop management
18 mcodat (~.mco): management codes (stress factor threshholds)
19 gwdat (~.gw): groundwater, but input is superseded by results from Modflow.
20 wgendat (~.wgn): historical basis for synthesized weather data
9+j for j=1 to 18 (10 to 27): daily rainfall (mm) for up to 18 stations
27+j for j=1 to 18 (28 to 45): daily min & max temperature (deg C) for up to 18 stations.

Modflow file device numbers:

Subr Modflo (file modflo.for):

1 inbas (~.bas): input for basic package
116 iout (~.prn): standard output
117 iolog (~.log): log file
217 iodwl
218 iomeas (~.mea): residuals (calculated - measured heads)
219 iomea2 (~.me2): summary of residuals
220 iorsd1 (~.rsd): another residual file

Device numbers specified as input (array iunit in file ~.bas and in array header records): suggested numbers that should not conflict with any of those specified in Swat or Modflow as described above:

61 ~.bcf
62 ~.wel
63 ~.drn
64 ~.riv
65 ~.evt
66 ~.swb
67 ~.ghb
68 ~.rch
69 ~.sip
70 ~.pcg
71 ~.sor
72 ~.oc
73 ~.str
74 ~.div surface water diversions

Appendix C. Extensions to Modflow input instructions for modswb95

Specified-flow boundary conditions (Well package)

Imaginary wells, referred to herein as "model" wells, are used to specify boundary flow conditions. These wells are distinguished from "actual" wells that represent pumping wells. The model wells are distinguished in the input file for the Well package by an asterisk in column 41, which lies to the immediate right of the input file's field for flow rate. This is illustrated by the following excerpt from input file 6084wr.wel:

```
         1         2         3         4
12345678901234567890123456789012345678901
-----------------------------------------
     layer       row    column pump rate*
         1        44        56   -33773.9
         1        17        21   -32222.5
         1        18        21   -21959.0
         1        19         9   -56091.0
         1         3         7   112388.8*
         1         4         6   49794.62*
         1         5         5   42906.26*
         1         6         4   54999.23*
```

The first four records above identify layer, row, column, and pumping rate (ft^3/day) for actual wells; the second four records identify boundary conditions with "model" wells.

Unformatted output file options

The table below shows the scheme used to implement the unformatted output options in Modflow. These options are described in the Modflow manuals under the input instructions for the respective packages; see McDonald and Harbaugh (1988) for all but the stream package documentation, which is documented separately by Prudic (1989). Contents of the table columns are as follows.

Column	Contents of table column
input flag:	name of Modflow input flag to specify unformatted output;
device:	file device number to which unformatted output is written;
file prefix:	file name is formed by attaching the extension ".unf";
package:	the group of subroutines in Modflow associated with the flag;
subroutine:	which routine in Modflow's package's that reads the flag;
output:	description of some of the unformatted file's contents.

input flag	j: device(j)	file prefix	package	subroutine	output
ihedun	1 (151)	modhed	oc	bas1rp	heads
iddnum	2 (152)	modddn	oc	bas1rp	drawdown
ibcfcb	3 (153)	modbcf	bcf	bcf1rp	storage, flow rates
iwelcb	4 (154)	modwel	wel	well1al	
idrncb	5 (155)	moddrn	drn	drn1al	
irchcb	6 (156)	modrch	rch	rch1al	
ievtcb	7 (157)	modevt	evt	evt1al	
irivcb	8 (158)	modriv	riv	riv1al	
istcb1	9 (159)	modst1	str	str1al	streambed leakage
istcb2	10 (160)	modst2	str	str1al	streamflow
ighbcb	11 (161)	modghb	ghb	ghb1al	

Consistent with the Modflow manuals, setting any of these flags to a positive number results in unformatted output, contingent on assigning a nonzero value to the overriding flag, **icbcfl**, which is described in the output control (**oc**) instructions. Contrary to the manual, however, the positive values assigned to these flags do not serve double duty as output file unit numbers; column 2 shows the unit numbers in parentheses that are used to write these files if the corresponding input flags have a positive value (e.g., 1). Column 3 shows the names, all of which have the extension ".unf", of the resulting unformatted files.

Appendix D. Source code

HYDBAL source code

File c:\swat\for95\hydbal.for Perkins '96

Subroutine Hydbal

```
*
      subroutine HYDBAL (kw,iosol,title,basttl,nwshed,shed,ssub,ssb,
     1    flu,fpd,fp,tir,dtswat,da,pmpmax,tmpdeg,solrad,relhum,
     1    wndspd,etcref,priest,swtpet)
*
*             other entry points in this subroutine for Swat-Modflow connection:
* (a) pass irrigation water rights (mm) as read from Modflow's ~.wel file
*     and evaporation from water table as calculated in MODFLOW:
cccc  entry PASSFLX (kw,nwshed,shed,pmpmax)
* (b)     pass fluxes from array ssub in Swat to array shed in modflow;
cccc  entry SWT2MOD (nwshed,shed,ssub,ssb,flu,tir)
* (c)     do the reverse (pass Modflow results in array shed to ssub in Swat)
cccc  entry MOD2SWT (nwshed,shed,ssub,ssb,flu,tir)

      parameter (mb = 100) ! max number of subbasins within the big subbasins
      parameter (ml = 10)  ! max number of soil layers
      parameter (msubo = 60)   ! max. no. vectors in subbasin array ssub
*         index list for hydrologic balance components in arrays ssub & ssb:
      parameter (nsbbal=20)
      dimension shed(30,nwshed),ssub(msubo,nwshed),ssb(109),flu(nwshed)
*         List of water rights (mm/yr) for each subbasin; passed from Modflow
*         via initializing call to HYDBAL (below), made from Swatmain just
*         prior to entering annual loop.
      dimension pmpmax(nwshed)
      dimension fp(nwshed) ! pond areas as fractions of subbasins
*         note: parameter mb=max. no. subbasins (declared in file common.f)
*                         ml=max. no. soil layers (coordinate w/ max in SWB)
      character*(*) basttl ! specified as char*12 in Main0 (for soil balance
*                             file to identify time for each step.
      character*(*) title  ! specified as char*60 in Main0, and includes
*         the phrase "hydrologic balance" which serves as a search key for
*         finding the fluxes to be read.  Title is followed by time step (sec).
*
*         names & numbers for subbasin output files to be read by modflow:
      character*600 headsb
      dimension islect(20,5), select(20,5)
*         specs for hydrologic balances (subr hydbal):
      dimension idflux(nsbbal),basstg(5),stgdev(5),
     1          substg(5,mb),bascmp(nsbbal),basdev(nsbbal)
*           correspondence between subbasin array ssub and basin array ssb
*           (defined in data statement below):
      dimension idxssb(msubo)
      character*10 colhdg(msubo), namvar(msubo), balhdg(5)
      integer ilist(20), imodfl(20), idxs2m(20), idxm2s(20)
*         list of items of interest: 10=water yield,14=sediment yield,25=etpot
      data nlist /3/, (ilist(j),j=1,3) /10, 14, 25/
      data numhyd /12/ !number of hydrologic balance components
*            pass hydrologic components from Swat's ssub to Modflow's shed:
* precip,irr,et,surq,tloss,latq,perc,gwre,rev,gwq,psep,etpot:
*      1  22 12    4    13     5    11     9   7    6    16     25
      data (idflux(j),j=1,12) / 1,22,12, 4,13, 5,11, 9, 7, 6,16,25/ !components
*           corresponding list of vectors in array shed:
      data (idxs2m(j),j=1,12) /10,11,12,13,14,15,16,17,18,19,20,21/
*           data passed from MODFLOW back to SWAT:   !revap, gw return flow
      data nmodfl /2/, (imodfl(j),j=1,2) / 7, 6/,
     1                 (idxm2s(j),j=1,2) /18,19/
*           correspondence between subbasin array ssub and basin array ssb:
*           vector i in ssub(i,j) corresponds to vector idxssb(i)
*           in ssb(idxssb(i)).
      data (namvar(i),idxssb(i),i=1,21)/
     1    'subp*p2',1, 'precip',39, 'sml',36, 'qi',3, 'ssf',4,           1- 5
     1    'gwq',104, 'revap',105, 'gwseep',106, 'gwchrg',107,            6- 9
     1     'H2O yield',6, 'sep',5, 'aet',7, 'qtl',38, 'yd',12,          10-14
     1     'ev',19, 'sp',20, 'rl',21, 'qdr',22, 'pq',13,                15-19
     1     'o',23, 'yp',14/                                             20-21
      data namvar(25),idxssb(25) /'eo',108/                             25
      data (namvar(i),idxssb(i),i=34,41) /
     1     'pq',16, 'yon',40, 'yph',41, 'yno3',42, 'ssfn',45,          34-38
     1     'ysp',43, 'uno3',44, 'percn',46/                             39-41
      data namvar(54),idxssb(54) /'sw(j)',35/                           54
      data (namvar(i),idxssb(i),i=55,60) /
     1     'sp',25, 'rl',26, 'qdr',27, 'ev',24, 'o',28, 'yp',17/       55-60
*     labels for hydrologic balances basstg and substg computed by HYDBAL:
      data balhdg /'soil', 'vadose', 'aquifer', 'stream', 'combined'/
*
* basin balances: ds (change in storage) = net fluxes into control volume
*
* soil:    ds(1) =  precip + irrig - et - runoff
*                   - percolation from root zone - subsurface (lateral) flow;
* vadose:  ds(2) =  percolation from root zone - recharge;
* aquifer: ds(3) =  recharge + transmission loss + pond seepage
*                   - irrigation - revap - groundwater (base) flow;
* stream:  ds(4) =  runoff + subsurface (lateral) flow + gw (base) flow.
* combined:ds(5) =  precip - et - revap
      data itrace/1/
*         selection matrix to include proper components in balances:
      data ((islect(i,j),i=1,12),j=1,5) /
     1                        1, 1,-1,-1, 0,-1,-1, 0, 0, 0, 0, 0,      !soil
     1                        0, 0, 0, 0, 0, 0, 1,-1, 0, 0, 0, 0,      !vadose
     1                        0,-1, 0, 0, 1, 0, 0, 1,-1,-1, 1, 0,      !aquifer
     1                        0, 0, 0, 1, 0, 1, 0, 0, 0, 1,-1, 0,      !surface water
     1                        1, 0,-1, 0, 1, 0, 0, 0,-1, 0, 0, 0/      !combined
* ----------component------------- ssub idxssb idxs2m sol vad aqf sur combined
* our term   Swat var.  Swat output            (*)
*-soil-------------------------------------------------------------------
*     precip     pcp?    precip        1      1      0   1   .   .   .   1
*     irrig      tir[1]               22     tir[1]  11   1   .  -1   .   .
*     et         aet     et           12      7     12  -1   .   .   .  -1
*     qsro[5]    qd      surface Q     4      3     13  -1   .   .   1   .
*     transl     qtl     trans loss   13     38     14   .   .   1   .   1
*     qlat[5]    ssf     subsurface Q  5      4     15  -1   .   .   1   .
*     perc_rz    sep     deep perc    11      5     16  -1   1   .   .   .
*-vadose-----------------------------------------------------------------
*-aquifer----------------------------------------------------------------
*     recharge   gwchrg  gw recharge   9    107     17*  .  -1   1   .   .
*     evap[2]    revap   revap         7    105     18*  .   .  -1   .  -1
*     gw retflo  gwq     groundwater Q 6    104     19*  .   .  -1   1   .
*     pond seep  sp [3]               16     20     20   .   .   1  -1   .
*     gwperc[4]  gwseep  gw perc       8    106      0   .   .  -1   .  -1
*-stream-----------------------------------------------------------------
*     trans loss  (is this supposed to be for streams, while qtl (above)
*                 is for overland flow?)                          -1
*     evap loss                                                   -1
*------------------------------------------------------------------------
* (*) idxs2m column: (*) indicates that the component is calculated in Modflow.
* [1] tir, weighted sum of irrigation over subbasins, is calculated by Swat
*     only as avg annual value following the simulation loop on years, so
*     this weighted sum is calculated below for the irrigation variable (i=22)
*     to show it in the hydrologic balance summary for any specified time step
*     (daily, monthly, annual, avg annual).
* [2] Evaporation from water table consists of capillary flow upward into the
*     vadose zone and plant uptake by roots in the water table.  However, the
*     "revap" model bypasses the vadose and soil zones.
* [3] Seepage from ponds and wetlands are both passed to Modflow to be included
*     in aquifer recharge. (Note: either pass these as a flow rate or, if if
*     passed as a flux, fix necessary area conversions.
* [4] This term is defined in Swat as "percolation from the deep aquifer",
*     which we ignore in Modflow, so we'll quit passing it to Modflow, and
*     pass the pond and wetland seepage terms (above) instead.
*
* [5] According to Prof. Jim Koelliker, KSU, the noncontributing area fraction
*     given by fp(j) for each subbasin j affects both recharge and subbasin
*     surface outflow to streams as follows:
*     Let c = 1-fp(j) = contributing areal fraction of subbasin to streamflow.
*     1) Recharge GWRE (Modflow: shed(17,*), Swat: ssub(9,*)):
*        But MODFLOW's definition of recharge is a combination of SWAT's terms
*        as follows, which is retained as shed(25,*) in MODSWB:
```

```
*           GWRE = Qtrans_loss + c*Qlat + Qperc_rz + Qpond_seep;
*      2) subbasin outflow to streams = c*Qsurf - Qtrans_loss/c
*      These definitions are invoked by setting jkkopt=1 in the ~.cod input
*      file.  It is passed to Modflow either by Modflow's argument list (if
*      iopswt>0) or via the balance file written below (if iopswt=0).
*
*reduces both surface and subsurface
*      runoff contribution to streamflow by the subbasin contributing area
*      fraction (1-fp(j)).  The remaining fraction goes to the ponds.
*      This reduction is calculated in Modflow (module modswb, subr SWB1FM)
*      where surface and subsurface runoff from each subbasin are added to
*      stream reaches as lateral inflow.
*      This also affects the stream control volume's balance calculated below.
*-----------------------------------------------------------------------
* Note: the soil water balance as calculated in Swat (subr swbl20 called
*        by Swatmain) is shown below for comparison with balances calculated
*        here in Hydbal.  As called from Swatmain:
C      CHECKING FINAL SOIL WATER BALANCE
*      call swbl20(sm(1),sm(3),sm(7),sm(4),sm(5),ssb(35),sm(38),tir)
*      argument list
*      (swbl)  (main)  name used in Swat's final output
*      ------  ------  ------------------------------
*      p       sm(1) = precip,
*      q       sm(3) = surface Q (runoff),
*      et      sm(7) = et (actual),
*      ssfl    sm(4) = subsurface Q (interflow),
*      ol      sm(5) = deep perc (percolation out of root zone),
* ?    sxx     ssb(35)
*      tranl   sm(38)= trans loss
*      tir     tir   = irrigation
****   subroutine swbl20(p,q,et,ssfl,ol,sxx,tranl,tir) ! on file swats-y.for
C      THIS SUBROUTINE CHECKS THE SOIL WATER BALANCE AT THE END OF A SIM
****   include 'common.f'
****   sww = sww + p - q - ssfl - bst - et - ol - sxx + tranl + tir
****   return
****   end
*-----------------------------------------------------------------------
* subbasin balances:
      data initlz/0/
      if (initlz.eq.0) then
          initlz=1
          do ibal=1,5
              do icomp=1,20
                  select(icomp,ibal) = islect(icomp,ibal)
              end do
              basstg(ibal)=0.
              stgdev(ibal)=0.
              do j=1,nwshed
                  substg(ibal,j)=0.
              end do
          end do
*-------------------------------------------------------------------------
*                revised code from POSTSWAT:
          write (headsb,65010)
65010     format (6x,'SEG GIS BIG MGT MON ',' AREA KM2 ',
     *     '   PREC MM',' SNOF MM',                              ! 1, 2
     *     '   SNOM MM','   SURQ MM','   LATQ MM','   GW Q MM',  ! 3- 6
     *     '    REV MM','  GWPER MM','   GWRE MM','   WYLD MM',  ! 7-10
     *     '   PERC MM','     ET MM','  TLOSS MM','   SYLD MM',  !11-14
     *     '   PEVP MM','   PSEP MM','   PARI MM','   PINF MM',  !15-18
     *     '  PSINF MM','   POUT MM','  PSOUT MM','    IRR MM',  !19-22
     *     '   SAQU MM','   DAQU MM','  ETPOT MM','    SGW MM ', !23-26
     *     '   DGW MM  ','   WSTRS  ','TMPSTRS ','   NSTRS  ',   !27,31-33
     *     ' PSTR MM',' ORGN KH','  PYLD KH',' NSURQ KH',        !34-37
     *     '  NSUBQ KH','   SOLP KH','    NUP KH','   NO3L KH',  !38-41
     *     '    PUP KG','   LABP KH','   STAP KH','   NAPP KH',  !42-45
     *     '   PAPP KH','   NFIX KH','   DENT KH','  HMAON KH',  !46-49
     *     '   ASON KH','  HMAOP KG','  MFRSN KH','  MFRSP KH',  !50-53
     *     '     SW MM','WetlSep mm','    LAI KH','    YLD KH')  !54-57
          read (headsb(37:),'(54a)') (colhdg(j),j=1,27),
     1      (colhdg(j),j=31,55),(colhdg(j),j=57,58)
          write (kw,'(1x,a)') 'Hydbal: ' // title
          nfield = numhyd - 1 !leave last flux (etpot) out of balance & summary files
          write (iosol,'(a12,25a8)') basttl,
     1        (colhdg(idflux(k)),k=1,nfield), (balhdg(k),k=1,5),
     1         ' Temp  C',colhdg(25),'rad,ly/d',' rel.hum',' u2,m/s',
     1         ' EtRefmm',' Priest',' SwatPET'
           idx25 = idxssb(25)
           if (itrace.gt.0) then
               itrace = 0
               write (kw,'(1x,a)') 'First 60 column headings:'
               write (kw,'(2(2i5,2(1x,a)))')
     1              (j,idxssb(j),namvar(j),colhdg(j),j=1,msubo)
               write (kw,'(1x,a,i3,1x,a)')
     1               'Write',nfield,'fields to MODFLOW summary files:'
               do j=1,nfield
                   write (kw,'(i5,1x,a10)') idflux(j),colhdg(idflux(j))
               end do
           end if
           return ! from initializing call to HYDBAL.
       end if
*--------------------------------------------------------------------------
       write (kw,'(f15.0,a,t26,a)') dtswat,' (days) ',title
*          initialize balance arrays (basin and subbasins):
       do ibal=1,5
           basstg(ibal) = 0.
           stgdev(ibal) = 0.
           do j=1,nwshed
               substg(ibal,j) = 0.
           end do
       end do
*---------- accumulate daily balances:---------------------------------
       do icomp=1,nfield
           i = idflux(icomp) !i =index to hydrologic components (subbasin)
           if (i.eq.22) then   !irrigation
               bascmp(icomp) = 0.
               do j=1,nwshed
                   bascmp(icomp) = bascmp(icomp) + flu(j)*ssub(i,j)
               end do
           else
               ib= idxssb(i)   !ib=index from subbasin to basin components
               bascmp(icomp) = ssb(ib)
           end if
*              describe subbasin deviation from basin mean with population
*              standard deviation (Snedecor & Cochran, 1980, 7th ed.,
*              Iowa State Univ. Press, p.29):
           basdev(icomp)=0.
           actgrd = 0.
           do j=1,nwshed
               basdev(icomp) = basdev(icomp) +
     1             flu(j)*(ssub(i,j)-bascmp(icomp))**2
*                  active gridded area:
               actgrd = actgrd + da*shed(1,j)*shed(7,j)
           end do
           grdfrc = actgrd/da
           basdev(icomp) = SQRT(basdev(icomp))
           if (icomp.eq.1) then
               write (kw,210) 'id So Va Aq St To Sb Bas Sh component'//
     1             '   mean  std dev',(j,j=1,nwshed)
210            format (1x,a,t53,10i8/,(t53,10i8))
               write (kw,220) da,' km**2 area',
     1             'fractions:', (flu(j),j=1,nwshed)
220            format (f15.3,a,t29,a,t53,10f8.5/,(t53,10f8.5))
               write (kw,225) 'ponds (areal fractions):',
     1             fpd,(fp(j),j=1,nwshed)
               write (kw,225) 'active gridded fraction:',
     1             grdfrc,(shed(7,j),j=1,nwshed)
225            format (t2,a,t39,f8.5,6x,10f8.5/,(t53,10f8.5))
           end if
           write (kw,230) icomp, (islect(icomp,j),j=1,5),
     1         i, idxssb(i), idxs2m(icomp), colhdg(i),
     1         bascmp(icomp), basdev(icomp),(ssub(i,j),j=1,nwshed)
*              evaluate balances:
230        format (7i3,i4,i3,a,2f7.1,10f8.2/,(t53,10f8.2))
           do ibal=1,5
*                  basin balances:
               basstg(ibal) = basstg(ibal) +
     1                             select(icomp,ibal)*bascmp(icomp)
*                  subbasin balances:
               do j=1,nwshed
```

```
                substg(ibal,j) = substg(ibal,j) +
     1                           select(icomp,ibal)*ssub(i,j)
            end do
        end do
      end do
*         describe subbasin balance deviation from basin balance with
*         population standard deviation as above:
      do ibal=1,5
          stgdev(ibal)=0.
          do j=1,nwshed
              stgdev(ibal) = stgdev(ibal) +
     1            flu(j)*(substg(ibal,j)-basstg(ibal))**2
          end do
          stgdev(ibal) = SQRT(stgdev(ibal))
      end do
      write (kw,'(1x,a)') 'Balances:'
      do ibal=1,5
          write (kw,240) ibal,balhdg(ibal),
     1        basstg(ibal), stgdev(ibal), (substg(ibal,j),j=1,nwshed)
240       format (i3,t29,a,2f7.1,10f8.2/,(t53,10f8.2))
      end do
      rhpct = relhum
      write (iosol,'(a12,25f8.1)')
     1    basttl, (bascmp(k),k=1,nfield), (basstg(k),k=1,5),
     1    tmpdeg,ssb(idx25),solrad,rhpct,wndspd,etcref,priest,swtpet
      if (nlist.gt.0) then
          write (kw,'(1x,a)') 'other hydrologic terms of interest'
          do icomp=1,nlist
              i = ilist(icomp)  !i=index from list of items of interest
              ib= idxssb(i)     !ib=index from subbasin to basin components
              if (i.eq.22) then !irrigation
                  bascmp(icomp) = 0.
                  do j=1,nwshed
                      bascmp(icomp) = bascmp(icomp) + flu(j)*ssub(i,j)
                  end do
              else if (1.le.ib.and.ib.le.109) then
                  bascmp(icomp) = ssb(ib)
              else
                  bascmp(icomp) = 0.
              end if
*                 describe subbasin deviation from basin mean with population
*                 standard deviation as above:
              basdev(icomp)=0.
              do j=1,nwshed
                  basdev(icomp) = basdev(icomp) +
     1                flu(j)*(ssub(i,j)-bascmp(icomp))**2
              end do
              basdev(icomp) = SQRT(basdev(icomp))
              write (kw,250) icomp, i, idxssb(i), colhdg(i),
     1            bascmp(icomp), basdev(icomp), (ssub(i,j),j=1,nwshed)
250           format (i3,15x,i3,i4,3x,a,2f7.1,10f8.2/,(t53,10f8.2))
          end do
      end if
      return
```

Entry Passflx

```
* (a) pass irrigation water rights (mm/yr) to SWAT as read from Modflow's ~.wel file
*     to Swat just prior to entering the annual simulation loop.  shed(9,*)
*     is calculated in subr SWB1RP.  Use this as an annual limit on irrigation,
*     whether automatic (irropt=1 in the ~.mco files) or scheduled (~.mgt files).

      entry PASSFLX (kw,nwshed,shed,pmpmax)
      avgmax=0.
      flomax=0.
      do j=1,nwshed
          pmpmax(j) = shed(9,j)
          avgmax = avgmax + shed(9,j)*shed(1,j) !avg fluxes (L)
          flomax = flomax + shed(4,j) !add pumping flow rates (L**3/T)
      end do
      print '(1x,a,g12.4,1x,a,f8.2,1x,a)', 'PASSFLX: '//
     1    'pass water rights from MODFLOW (SWB1RP) to SWAT; '//
     1    'Qirrig=',flomax,'[L^3/T]',avgmax,'(mm/yr)'
      return
```

Entry Swt2Mod

```
* (b) pass fluxes from array ssub in Swat to array shed in modflow;

      entry SWT2MOD (nwshed,shed,ssub,ssb,flu,tir)

cc    print '(1x,a)', ' k  i ssb shd              hyd. flux basin'//
cc   1                ' avg subbasin values:'
      do k=1,numhyd
          i=idflux(k)
          kshed=idxs2m(k)
          ib= idxssb(i)     !ib=index from subbasin to basin components
          if (1.le.ib.and.ib.le.109) then
              bascmp(k) = ssb(ib)
          else if (i.eq.22) then
              bascmp(k) = 0.
              do j=1,nwshed
                  bascmp(k) = bascmp(k) + flu(j)*ssub(i,j)
              end do
          else
              bascmp(k) = 0.
          end if
cc        print '(2i3,2i4,a,t29,f7.2,t53,10f8.2/,(t53,10f8.2))',
cc   1        k, i, idxssb(i),kshed, colhdg(i),
cc   1        bascmp(k), (ssub(i,j),j=1,nwshed)
          do j=1,nwshed
              shed(kshed,j) = ssub(i,j)
          end do
      end do
      return
```

Entry Mod2Swt

```
* (c) pass Modflow results in array shed to ssub in Swat

      entry MOD2SWT (nwshed,shed,ssub,ssb,flu,tir)

      print *,'MOD2SWT: pass data from array shed to ssub'
cc    print '(1x,a)', ' k  i ssb shd              hyd. flux basin'//
cc   1                ' avg subbasin values:'
      do k=1,nmodfl
          i=imodfl(k)
          kshed=idxm2s(k)
          bascmp(k)=0.
          do j=1,nwshed
              ssub(i,j) = shed(kshed,j)
*                 areally weighted average of subbasin fluxes from modflow:
              bascmp(k) = bascmp(k) + flu(j)*ssub(i,j)
          end do
          ib= idxssb(i)     !ib=index from subbasin to basin components
          if (1.le.ib.and.ib.le.109) then
              ssb(ib) = bascmp(k)
          else if (i.eq.22) then
              tir = bascmp(k)
          end if
cc        print '(2i3,2i4,a,t29,f7.2,t53,10f8.2/,(t53,10f8.2))',
cc   1        k, i, idxssb(i),kshed, colhdg(i),
cc   1        bascmp(k), (ssub(i,j),j=1,nwshed)
      end do
      return
      end
```

SWBAVG source code: spatial average of hydrologic response units

File c:\swat\for95\swbavg.for spp jan 14 1996 latest version mar 5 96
Program SWBAVG

```
    program SWBAVG
*
* compute a weighted average of soil water balance files written by Swat,
* based on a weight function for a set of conditions, and write the weighted
* avg version of the soil water balance.  This code is adapted from subr
* SWB1FM on file modswb.for, which reads a soil water balance file for
* one time step and assigns subbasin fluxes to associated grid nodes.
*
*     log of revisions:
* averages over conditions, subbasins and time steps revised, corrected  feb 28 96.
* mar 5 96: revised to accommodate latest version of SWATMOD:
* 12 hydrologic flux items, including ETPOT, are now read from the balance file
* for averaging.

C-----VERSION  1  14Jan1996 SWBAVG                                      C
C     ******************************************************************C
C     compute a weighted average of soil water balance files written by Swat.
C     ******************************************************************C
C     SPECIFICATIONS:                                                   C
C     ------------------------------------------------------------------C
      parameter (mxshed=100, mxcond=20,mxvers=30,mxvect=25)
      DIMENSION SHED(mxvect,mxshed),shedwt(mxvect,mxshed),
     1          shedav(mxvect,mxshed),irevyr(mxvers)
      dimension sumwt(mxshed),condwt(mxshed,mxcond),inpdev(mxcond)
      character*30 outsub,outsum,outbal,inpbal(mxcond), inpfil,inpdft
      character*10 condnm(mxcond)
      character*75 wthdg

*         names & numbers for subbasin output files to be read by modflow:
      character*600 headsb
      character*12 basttl  ! from subr hydbal
*         index list for hydrologic balance components in arrays ssub & ssb:
      parameter (nsbbal=20)
      character*10 balhdg(5)
*         specs for hydrologic balances (subr hydbal):
      dimension idxbal(nsbbal),basstg(5),stgdev(5),
     1          bascmp(nsbbal),basdev(nsbbal)
      character*50 fmtswb  ! format for record read by subr swbhyd (below)
      character*10 colhdg(60)  ! label read with hydrologic component from swb file
      character*10 coltxt(25)
      character*132 recinp(mxshed)
*         vector for avg basin flux: basflx (Modflow units), swtflx (Swat units)
      dimension swtflx(25), flxdev(25)
*         subbasin sums: no. active nodes, recharge,
*         evapotrans from water table, pumping, and tributary outflow:
      dimension subval(100)
*       (1) areal fraction of subbasin;
*       (2) area of subbasin [L^2];
*       (3) gridded subbasin area (see also shed(7,*) below).
*       (4) irrigation [L] specified by WELL module input (Qsub/subbasin area)*T
*       (5) irrigation flux specified by SWAT as a fraction of the irrigation
*           flux read from MODFLOW's WELL input file for the stress period
*           as calculated below.  The pumping rate for each well in the subbasin
*           is weighted by this fraction so that the irrigation flux specified
*           by SWAT's input is pumped by the wells.
*       (6) fraction  of irrigation pumping returned as recharge (percolation
*           out of the root zone); calculated below.
*               read vectors 11-25 from the flux data written by SWAT:
*       (7) gridded active subbasin area as fraction of actual subbasin area;
*           used as correction factor to gridded areal weights e.g. for
*           recharge assignment (see subr SWB1FM).
*  8        noncontributing areal fraction of subbasin, which contributes
*           instead to ponds
*       (9) irrigation flux: water rights passed back to Swat after call to SWB1RP.
* newest list (7/10/95 spp):
*   hydrologic components computed by Swat and passed to Modflow:
* 10        precipitation
* 11        applied irrigation [L] (used to scale pumping if irropt>0)
* 12        ET from ground surface [L] (maximum rate for evap from water table)
* 13        surface runoff [L] to tributaries leaving subbasin
* 14        transmission loss [L]
* 15        lateral flow [L] to tributaries leaving subbasin
* 16        percolation from the root zone
* 17        groundwater recharge = shed(14,*) + shed(15,*)*(1-shed(8,*))
*                                + shed(16,*) + shed(18,*)
*   hydrologic components also passed to Modflow, but superseded by
*   values calculated by Modflow:
* 18        evaporation from water table (used as "revap" in Swat)
* 19        streambed leakage from stream; its negative is used as return flow in Swat).
* 20       pond seepage into aquifer
* 21       tributary inflow = shed(13,*)*contrb - shed(14,*)/contrb
* 22       depth to water
* 23       potential et
      data (colhdg(i),i=1,25) /
     1      'Areal frc', ' area[L^2]','GrdAreaL^2','Irr_use[L]',         ! 1- 4
     1      'IrrSwatFrc','IrrRet_frc','AreGrd/Act','Noncontrib',         ! 5- 8
     1      'Irr_use   ','  Prcp mm',                                    ! 9-10
     1      ' Irrig mm ','    ET mm ',' SURQ mm ','XMLoss mm',           !11-14
     1      '  LATQ mm ','  PERC mm ',' GWRch mm ',' GW ET mm ',         !15-18
     1      'Basflow mm','PndSeep mm','TrbInfl mm',' DTW [L] ',          !19-22
     1      ' POT ET mm','Pmprat mm ','        '/                        !23-25
      data fmtswb /'(25x,i3,1x,a,20f8.2)'/
ccc   data timstp /86400./ ! sec/day
C**   data daystp /365./   ! day/step
C**   data cnvfac /304.8/  ! mm/ft
C**   data cnvlen /1000./  ! mm/m
      data inpdft /'swbavg.inp'/
      data nper /10000/
      data itrace /0/
*          --list also included in main (swatmod1.h) for use in annual.h.
      data (idxbal(j),j=1,12) / 1,22,12, 4,13, 5,11, 9, 7, 6,16,25/ !components
*     labels for hydrologic balances basstg and substg computed by HYDBAL:
      data balhdg /'soil', 'vadose', 'aquifer', 'stream', 'combined'/
C     -----------------------------------------------------------------
*              revised code from POSTSWAT:
          write (headsb,65010)
65010     format (6x,'SEG GIS BIG MGT MON ',' AREA KM2 ',
     *      '   PREC MM','    SNOF MM',                                  ! 1, 2
     *      '   SNOM MM','    SURQ MM','    LATQ MM','    GW Q MM',      ! 3- 6
     *      '    REV MM','   GWPER MM','    GWRE MM','    WYLD MM',      ! 7-10
     *      '   PERC MM','      ET MM','   TLOSS MM','    SYLD MM',      !11-14
     *      '   PEVP MM','    PSEP MM','    PARI MM','    PINF MM',      !15-18
     *      '  PSINF MM','    POUT MM','   PSOUT MM','     IRR MM',      !19-22
     *      '   SAQU MM','    DAQU MM','   ETPOT MM','     SGW MM ',     !23-26
     *      '   DGW MM  ','    WSTRS  ','TMPSTRS ','     NSTRS  ',       !27,31-33
     *      ' PSTR MM','   ORGN KH','    PYLD KH','  NSURQ KH',          !34-37
     *      '  NSUBQ KH','    SOLP KH','     NUP KH','    NO3L KH',      !38-41
     *      '     PUP KG','    LABP KH','    STAP KH','    NAPP KH',     !42-45
     *      '   PAPP KH','    NFIX KH','    DENT KH','   HMAON KH',      !46-49
     *      '   ASON KH','   HMAOP KG','   MFRSN KH','   MFRSP KH',      !50-53
     *      '     SW MM','Wet1Sep mm','     LAI KH','     YLD KH')       !54-57
          read (headsb(37:),'(54a)') (colhdg(j),j=1,27),
     1      (colhdg(j),j=31,55),(colhdg(j),j=57,58)
C
C1A-----IF mxshed IS LESS THAN 1 THEN MODSWB IS INACTIVE:

      print *,'Program SWBAVG: Compute weighted average of balance '//
     1        'files written by Swat.'
      print *,'(see example input file at end of source code on '//
     1        'file swbavg.for)'
      inpfil=' '
      print *,'Enter input file name; default='//inpdft(1:10)//': '
      read (*,'(a)') inpfil
      if (inpfil.eq.' ') inpfil=inpdft
*         Set mandatory name for now so that no keyboard input responses
*         are required; this makes redirecting terminal output more convenient.
      inpfil = inpdft
      print *,'Assumed input file name = '//inpfil
      print *,'Open input file: device=10, name='//inpfil
c     iolog=9
      in=8
      open (unit=in,file=inpfil,status='old')
c     open (unit=iolog,file='swbavg.log')
      print *,'read nwshed,ncond,mxper,nrevis,iopsub,outbal:'
      print *,'      nwshed = no. subbasins'
      print *,'      ncond = no. conditions'
      print *,'      mxper  = max. no. time periods'
      print *,'      nrevis = no. updates (revisions) to weight ftns'
      print *,'      iopsub = option: (y=1) specify a separate '//
     1       'weight ftn for each subbasin;'
      print *,'                            (y=0) specify one '//
     1       'weight ftn for all subbasins.'
      print *,'      outbal = avg balance output file name (30 char max)'
      read (in,*) nwshed,ncond,mxper,nrevis,iopsub,outbal
      IF(nwshed.LT.1 .or. nwshed.gt.mxshed) then
          print '(2(1x,a,i5))','no. subbasins nwshed=',nwshed,
     1        'is outside the allowed range from 1 to',mxshed
          stop
      end if
      IF(ncond.LT.1 .or. ncond.gt.mxcond) then
          print '(2(1x,a,i5))','no. weighting conditions ncond=',ncond,
     1        'is outside the allowed range from 1 to',mxcond
          stop
      end if
      outsub = 'balance.smj'
      outsum = 'balance.sum'
      idxdot = INDEX(outbal,'.')
      if (idxdot.gt.1) then
          outsum = outbal(1:idxdot) // 'sum'
          outsub = outbal(1:idxdot) // 'smj'
      end if
      iosol = 7
      open (unit=iosol,file=outsum,status='unknown')
      print '(1x,a)', 'Write avg balance summary to file '// outsum
      write (iosol,'(a12,25a8)')
     1    ' " t" "days"', (colhdg(idxbal(k)),k=1,12)
      data iosub/71/, jsub/1/
      if (jsub.gt.0) then
          open (unit=iosub,file=outsub,status='unknown')
          print '(2(1x,a,i3))', 'Write subbasin ',jsub,
     1        'avg balance summary to file '// outsub
          write (iosub,'(a12,25a8)')
     1        ' " t" "days"', (colhdg(idxbal(k)),k=1,12)
      end if
      print '(3(1x,a,i3),1x,a)', 'nwshed=',nwshed,'ncond=',ncond,
     1    'mxper=',mxper,'outbal='//outbal
      nper = MIN0(mxper,nper)
      print '(1x,a,i6)','No. time periods = MIN(mxper,nper)=',nper
      print '(1x,a,i2)','No. revisions of condition weights=',nrevis
      if (nrevis.gt.0) then
          print '(1x,a)','beginning year for each revision '//
     1        ' of weights:'
          read (in,*) (irevyr(i),i=1,nrevis)
          print '(1x,10(1x,i2,1h:,i4))', (i,irevyr(i),i=1,nrevis)
      end if
      jwtnxt = 0
      print *,'for each condition, read label (10 chars) '//
     1        'and balance file name (30 chars) in free format:'
      print *,' k Condition device file name'
      do k=1,ncond
          read (in,*) condnm(k), inpbal(k)
          inpdev(k) = 10 + k
          print '(1x,i2,1x,a,i7,1x,a)',k,condnm(k),inpdev(k),inpbal(k)
          open (unit=inpdev(k),file=inpbal(k),status='old')
      end do
*         open weighted condition output file:
      iout = 10
      open (unit=iout,file=outbal,status='unknown')
*
      daysum=0.                     !initialize cumulative time periods
      do i=1,mxvect
          do j=1,nwshed
              shedav(i,j) = 0.  !initialize avg over time periods
          end do
      end do
*         for each time period, compute a weighted average of each
*         hydrological component of each subbasin:
      idxfin=0 !continue until the avg annual balance has also been averaged.
      numavg=0 !initialize the no. of time periods to be averaged for the
*              temporal average over all years
      irevis=1
      kkstp=0
      do while (kkstp.lt.nper.or.idxfin.eq.0)
          kkstp = kkstp + 1
          do i=1,mxvect
              do j=1,nwshed
                  shedwt(i,j) = 0.  !initialize avg over crop & soil conditions
              end do
          end do
          do k=1,ncond
              if (kkstp.eq.1) then
*                 For the first time step, read header lines of flux file:
                  read (inpdev(k),'(4i5)') nyrs, iyr, lutot, jkkopt
                  if (lutot.ne.nwshed) then
                      print *,'inconsistent no. subbasins shown on'//
     1                    ' file '// inpbal(k)
                      STOP
                  end if
                  read (inpdev(k),'(a)') fmtswb
                  if (k.eq.1) then
                      write (iout,'(4i5)') nyrs,iyr,lutot,jkkopt
                      write (iout,'(a)') fmtswb
                      print '(1x,a)', 'read nyrs,iyr,lutot,
     1                    jkkopt from '// inpbal(k)
                      print '(1x,a)',
     1                    'Subbasin flux input format = '//fmtswb
                  end if
              end if
              idxhdg=0
              do while (idxhdg.eq.0)
                  read (inpdev(k),'(a)',err=500,end=500) recinp(k)
                  if (k.eq.1) write (iout,'(a)') recinp(k)
                  idxhdg = INDEX(recinp(k),'hydrologic balance')
              end do
*                 Check to see if "hydrologic balance" line is the same
*                 for each condition, indicating that the average will be
*                 based on the same time period.  Don't consider changing
*                 weights for the avg annual balance.
              if (k.eq.1) then
                  print '(1x,a)',recinp(k)(1:75)
                  idxavg = INDEX(recinp(k),'avg')
                  idxfin = MAX0(idxfin,idxavg) !flag for avg annual balance
                  if (idxavg.eq.0) then
                      read (recinp(k)(10:),'(f6.0,10x,i4)') days,ibalyr
                      iendyr = ibalyr
                      daysum = daysum + days
                  end if
                  if (kkstp.eq.1 .or.
     1                (idxavg.eq.0 .and. irevis.le.nrevis .and.
     1                irevyr(irevis).le.ibalyr)) then
                      if (kkstp.eq.1) then
                          ibgnyr = ibalyr
                          print *,'Initialize matrix of condition '//
     1                        'weights (pct) for each subbasin;'//
     1                        ' convert to fractions:'
                      else
                          irevis=irevis+1
                          print '(1x,a,i5)', 'Year',irevyr(irevis)
                          print *,'Revise matrix of condition '//
     1                        'weights(%) for each subbasin; '//
     1                        'convert to fractions:'
                      end if
                      do khdg=1,2
                          read (in,'(a)') wthdg  !2-line header for wt matrix
ccccc                     print '(1x,a)',wthdg
                      end do
                      print '(1x,a,t8,20i4/,(t8,20i4))',
     1                    ' j sum',(icol,icol=1,ncond)
                      do j=1,nwshed
                          if (j.eq.1.or.iopsub.gt.0) then
                              read (in,*) (condwt(j,ic),ic=1,ncond)
                              sumwt(j) = 0.
                              do ic=1,ncond
                                  sumwt(j) = sumwt(j) + condwt(j,ic)
                              end do
                              print '(1x,i2,21i4/,(7x,20i4))',
     1                            j,sumwt(j),(condwt(j,ic),ic=1,ncond)
                              do ic=1,ncond
                                  condwt(j,ic) = condwt(j,ic)/100.
                              end do
                              sumwt(j) = sumwt(j)/100.
                          else if (iopsub.eq.0) then
                              sumwt(j) = sumwt(1)
                              do ic=1,ncond
```

```
                                    condwt(j,ic) = condwt(1,ic)
                                end do
                            end if
                        end do
                    end if
                else
                    if (recinp(k)(1:75).ne.recinp(1)(1:75)) then
                        print '(1x,a,i3,a)','Mismatch of time periods'//
     1                         ' between condition ',k,' and 1:'
                        print '(1x,a,i3,1x,a)','Condition ',k,inpbal(k)
                        print '(1x,a)',recinp(k)(1:75)
                        k1=1
                        print '(1x,a,i3,1x,a)','Condition ',k1,inpbal(k1)
                        print '(1x,a)',recinp(k1)(1:75)
                        STOP
                    end if
                end if
                read (inpdev(k),'(t53,10i8)') (idx,j=1,nwshed)   ! column heading
                if (k.eq.1) write (iout,'(t53,10i8)') (j,j=1,nwshed)
                read (inpdev(k),220) avgval, (subval(j),j=1,nwshed)
220             format (f15.3,t53,10f8.5/,(t53,10f8.5))
                if (k.eq.1) then
                    write (iout,220) avgval, (subval(j),j=1,nwshed)
                    bsarea = avgval
                    do j=1,nwshed
                        shed(1,j) = subval(j)
                    end do
                end if
                read (inpdev(k),225) fpd,(shed(8,j),j=1,nwshed)
225             format (t39,f8.5,6x,10f8.5/,(t53,10f8.5))
                if (k.eq.1) write (iout,225) fpd,(shed(8,j),j=1,nwshed)
                read (inpdev(k),225) avgval,(subval(j),j=1,nwshed)
                if (k.eq.1)
     1              write (iout,225) avgval,(subval(j),j=1,nwshed)
*
*                   read SHED vectors 10-20 containing fluxes written by SWAT.
                do i=10,20
                    read (inpdev(k),fmtswb) idx, coltxt(i),
     1                  swtflx(i), flxdev(i), (shed(i,j),j=1,nwshed)
                    if (idx.ne.i) print '(1x,a,i3,1x,a)',
     1                  'SWB1FM: '//colhdg(i)//'i=',i,
     1                  'mismatch to ~.swb file idx=',idx
                    if (itrace.gt.0) print fmtswb, idx, coltxt(i),
     1                  swtflx(i), flxdev(i), (shed(i,j),j=1,nwshed)
*                       accumulate weighted flux over each condition and subbasin:
                    do j=1,nwshed
                        shedwt(i,j) = shedwt(i,j) + condwt(j,k)*shed(i,j)
                    end do
                end do
            end do
            if (idxavg.eq.0) then
                write (basttl,'(i5,f6.0)') ibalyr, days
                numavg = numavg + 1
                do i=10,20
                    do j=1,nwshed
*                           accumulate sum for temporal avg:
                        shedav(i,j) = shedav(i,j) + days*shedwt(i,j)
                    end do
                end do
            else
                write (basttl,'(a5,f6.0)') '"av" ',daysum
                do i=10,20
                    do j=1,nwshed
*                           temporally weighted avg for each flux and subbasin:
                        shedwt(i,j) = shedav(i,j)/daysum
                    end do
                end do
            end if
            do i=10,20
                swtflx(i)=0.
                flxdev(i)=0.
                do j=1,nwshed
*                       spatially weighted avg (over subbasins):
                     swtflx(i) = swtflx(i) + shed(1,j)*shedwt(i,j)
                end do
```

```
          end do
*                 Compute spatially weighted std deviation of temporally- and
*                 condition-averaged fluxes:
          do i=10,20
              do j=1,nwshed
                  flxdev(i) = flxdev(i)
     1                      + shed(1,j)*(shedwt(i,j)-swtflx(i))**2
              end do
              flxdev(i) = SQRT(flxdev(i))
              write (iout,fmtswb) i, coltxt(i),
     1                swtflx(i), flxdev(i), (shedwt(i,j),j=1,nwshed)
          end do
*                 code for write statement takend from hydbal:
          write (iosol,'(a12,25f8.1)') basttl, (swtflx(k),k=10,20)
cccc 1            , (basstg(k),k=1,5),tmpdeg,ssb(idx25),solrad,rhpct,
cccc 1              wndspd,etcref,priest,swtpet
          if (jsub.gt.0) write (iosub,'(a12,25f8.1)')
     1        basttl, (shedwt(k,jsub),k=10,20)
      end do
500   print *,'End of balance file reached.'
      do k=1,ncond
          close (unit=inpdev(k))
      end do
      close (unit=iout)
      close (unit=iosol)
      STOP
      end
c-------------------------------------------------------------------------------
c  example of control input file for the above program (ignore c in 1st column):
c  taken from Table XX, "Matrix of land use/soil type conditions.. for 1990":
c29,15,26,1,1,'test15ag.bal',    nwshed,ncond,nper,nrevis,iopsub,outfil swbavg.inp
c1974
c'Harney-wsf','ncase01g.bal',       condnm(k),inpfil(k)
c'Harney-irc','ncase01a.bal',       condnm(k),inpfil(k)
c'Harney-r/p','ncase01a.bal',       condnm(k),inpfil(k)
c'Pratt -wsf','ncase01g.bal',       condnm(k),inpfil(k)
c'Pratt -irc','ncase01a.bal',       condnm(k),inpfil(k)
c'Pratt -r/p','ncase01a.bal',       condnm(k),inpfil(k)
c'Tivoli-wsf','ncase01g.bal',       condnm(k),inpfil(k)
c'Tivoli-irc','ncase01a.bal',       condnm(k),inpfil(k)
c'Tivoli-r/p','ncase01a.bal',       condnm(k),inpfil(k)
c'Naron -wsf','ncase01g.bal',       condnm(k),inpfil(k)
c'Naron -irc','ncase01a.bal',       condnm(k),inpfil(k)
c'Naron -r/p','ncase01a.bal',       condnm(k),inpfil(k)
c'Carwil-wsf','ncase01g.bal',       condnm(k),inpfil(k)
c'Carwil-irc','ncase01a.bal',       condnm(k),inpfil(k)
c'Carwil-r/p','ncase01a.bal',       condnm(k),inpfil(k)
cnormalized weights from beginning year:
c  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
c 75  5 20  0  0  0  0  0  0  0  0  0  0  0  0                     1
c 75  5 20  0  0  0  0  0  0  0  0  0  0  0  0                     2
c 25  2  8 14  7 14 12  6 12  0  0  0  0  0  0                     3
c 36  2 12  4  8 13  0  0  0  4  8 13  0  0  0                     4
c  0  0  0  6 20 24  0  0  0  6 20 24  0  0  0                     5
c  0  0  0 10  5 25  6  2 12 10  5 25  0  0  0                     6
c 28  2 10 15  6  9  0  0  0 15  6  9  0  0  0                     7
c  0  0  0 14  8 18  0  0  0 27 12 21  0  0  0                     8
c  0  0  0 10 10 10  0  0  0 17 16 16  7  7  7                     9
c  0  0  0 10 16 14  0  0  0 15 24 21  0  0  0                    10
c  0  0  0 10 18 17  0  0  0 12 22 21  0  0  0                    11
c  0  0  0 15 10 25  9  6 15  0  0  0  8  4  8                    12
c  0  0  0  5 10  5  0  0  0 12 22 16  8 14  8                    13
c  0  0  0  8 12 10  0  0  0 11 15 14  8 12 10                    14
c  0  0  0  6  6  8  0  0  0 10 14 16 10 14 16                    15
c  0  0  0  8  6 16  0  0  0 12 10 28  5  4 11                    16
c  0  0  0  0  0  0  8  4  8 24 10 16 18  6  6                    17
c  0  0  0  0  0  0  4  4 12 10 10 30  6  6 18                    18
c  0  0  0  0  0  0  0  0  0 74  5 21  0  0  0                    19
c  0  0  0  0  0  0  7  5  8 18 14 18 12  8 10                    20
c  0  0  0  0  0  0  0  0  0 16 20 64  0  0  0                    21
c  0  0  0  8  2 10  0  0  0 24  6 30  8  2 10                    22
c  0  0  0 12  5 33  6  2 17  0  0  0  6  3 16                    23
c  0  0  0  9  3 18  0  0  0 15  5 30  6  2 12                    24
c  0  0  0 10  6 14  0  0  0 14  8 18 10  6 14                    25
c  0  0  0 22  5 22  0  0  0  0  0  0 23  5 23                    26
```

```
c  0  0  0  5  0 30  4  0 26  0  0  0  5  0 30                     27
c  0  0  0  5  0 45  5  0 45  0  0  0  0  0  0                     28
c  0  0  0  5  0 45  5  0 45  0  0  0  0  0  0                     29
cnormalized weights beginning in 1974:
c  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
c 75  5 20  0  0  0  0  0  0  0  0  0  0  0  0                      1
c 75  5 20  0  0  0  0  0  0  0  0  0  0  0  0                      2
c 25  2  8 14  7 14 12  6 12  0  0  0  0  0  0                      3
c 36  2 12  4  8 13  0  0  0  4  8 13  0  0  0                      4
c  0  0  0  6 20 24  0  0  0  6 20 24  0  0  0                      5
c  0  0  0 10  5 25  6  2 12 10  5 25  0  0  0                      6
c 28  2 10 15  6  9  0  0  0 15  6  9  0  0  0                      7
c  0  0  0 14  8 18  0  0  0 27 12 21  0  0  0                      8
c  0  0  0 10 10 10  0  0  0 17 16 16  7  7  7                      9
c  0  0  0 10 16 14  0  0  0 15 24 21  0  0  0                     10
c  0  0  0 10 18 17  0  0  0 12 22 21  0  0  0                     11
c  0  0  0 15 10 25  9  6 15  0  0  0  8  4  8                     12
c  0  0  0  5 10  5  0  0  0 12 22 16  8 14  8                     13
c  0  0  0  8 12 10  0  0  0 11 15 14  8 12 10                     14
c  0  0  0  6  6  8  0  0  0 10 14 16 10 14 16                     15
c  0  0  0  8  6 16  0  0  0 12 10 28  5  4 11                     16
c  0  0  0  0  0  0  8  4  8 24 10 16 18  6  6                     17
c  0  0  0  0  0  0  4  4 12 10 10 30  6  6 18                     18
c  0  0  0  0  0  0  0  0  0 74  5 21  0  0  0                     19
c  0  0  0  0  0  0  7  5  8 18 14 18 12  8 10                     20
c  0  0  0  0  0  0  0  0  0 16 20 64  0  0  0                     21
c  0  0  0  8  2 10  0  0  0 24  6 30  8  2 10                     22
c  0  0  0 12  5 33  6  2 17  0  0  0  6  3 16                     23
c  0  0  0  9  3 18  0  0  0 15  5 30  6  2 12                     24
c  0  0  0 10  6 14  0  0  0 14  8 18 10  6 14                     25
c  0  0  0 22  5 22  0  0  0  0  0  0 23  5 23                     26
c  0  0  0  5  0 30  4  0 26  0  0  0  5  0 30                     27
c  0  0  0  5  0 45  5  0 45  0  0  0  0  0  0                     28
c  0  0  0  5  0 45  5  0 45  0  0  0  0  0  0                     29
```

Modswb: MODFLOW's interface to watershed simulation results

File c:\swat\for95\modswb.for perkins Feb 15 95

Subroutine Swb1al: allocate arrays SHED and ISHED for the MODSWB package

```
Allocate space in X array for subbasins (shed,ished), tributaries (ishout) and
correspondence between watershed subbasins and aquifer grid (ibshed).

      SUBROUTINE SWB1AL(ISUM,LENX,LCSHED,ICSHED,LCOUTF,LCBSHD,
     1     lczsol,lcwsol,Lcevtr,LCdtw,LCevap,LCrchi,nwshed,noutfl,
     1     nsoils,mxslay,irropt,ievopt,ioprch,ncol,nrow,
     1     iunit,IOUT,filnam)
C usage:
C     IF(IUNIT(6).GT.0) call SWB1AL(ISUM,LENX,LCSHED,ICSHED,LCOUTF,     !swb1
C    1     LCBSHD,LCZSOL,LCWSOL,LCEVTR,nwshed,noutfl,nsoils,mxslay,
C    1     irropt,ncol,nrow,iunit,IOUT,filnam)

C-----VERSION    1 01MAR1995 SWB1AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR WATERSHEDS
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
        character *(*) filnam
        dimension iunit(24)
C     ------------------------------------------------------------------
C
      in = iunit(6)
C1------IDENTIFY PACKAGE AND INITIALIZE SUBBASINS.
      WRITE(IOUT,1) IN,filnam
    1 FORMAT(1H0,'SWB  -- SOIL WATER BALANCE PACKAGE V.1, 3/01/95',
     1'INPUT READ FROM UNIT',I3,' file ',a)
      NBASIN=0
C
C2------ READ:
*    nwshed = no. subbasins in watershed;
*    noutfl = total no. streams leaving subbasins (min. 1/subbasin);
*    nsoils = no. soils; each subbasin is associated with 1 soil type.
*    mxslay = max. no. soil layers
*    irropt = option (y=1,n=0) to use irrigation flux as determined
*             by SWAT for each subbasin to scale the pumping rates
*             specified by the WELL module input data.
*    ievopt = option (y=1,n=0) to specify potential evap from ground
*             surface with the potential evaporation flux calculated by SWAT,
*             rather than (ievopt < or = 0) with standard input to array evtr
*             in Modflow's EVT module.
*    ioprch = option (y=1,n=0) to distribute recharge within each subbasin
*             according to the recharge input file array RECH.
*         change from free to fixed format (3, below) for std version?
  100 read (in,*) nwshed,noutfl,nsoils,mxslay,irropt,ievopt,ioprch
    3 FORMAT(5I10,F10.0,2I10)
      nwshed = MAX0 (nwshed,0)
      noutfl = MAX0 (noutfl,0)
      nsoils = MAX0 (nsoils,0)
      write (iout,'(2(i5,1x,a))') nwshed,'subbasins,',
     1      noutfl,'exits from subbasins to streams,'
      write (iout,'(i5,1x,a)') nsoils,'soil types '//
     1    '(each subbasin is associated with one soil type)'
C
C3------SET LCSHED EQUAL TO ADDRESS OF FIRST UNUSED SPACE IN X.
  200 LCSHED=ISUM
C3------SPACE NEEDED FOR real watershed subbasin array shed(30,mxshed):
*      (1) areal fraction of subbasin;
*      (2) area of subbasin [L^2];
*      (3) gridded subbasin area (see also shed(7,*) below).
*      (4) irrigation [L] specified by WELL module input (Qsub/subbasin area)*T
*      (5) not currently used; previously,
*          irrigation flux specified by SWAT as a fraction of the irrigation
*          flux read from MODFLOW's WELL input file for the stress period
*          as calculated below.  The pumping rate for each well in the subbasin
*           is weighted by this fraction so that the irrigation flux specified
*           by SWAT's input is pumped by the wells.
*       (6) not currently used; previously,
*           fraction  of irrigation pumping returned as recharge (percolation
*           out of the root zone); calculated below.
*               read vectors 11-25 from the flux data written by SWAT:
*       (7) gridded active subbasin area as fraction of actual subbasin area;
*           used as correction factor to gridded areal weights e.g. for
*           recharge assignment (see subr SWB1FM).
*       (8) noncontributing areal fraction of subbasin, which contributes
*           instead to ponds
*       (9) irrigation flux: water rights passed back to Swat after call to SWB1RP.
* newest list (7/10/95 spp):
*   hydrologic components computed by Swat and passed to Modflow:
* 10        precipitation
* 11        applied irrigation [L] (used to scale pumping if irropt>0)
* 12        ET from ground surface [L] (maximum rate for evap from water table)
* 13        surface runoff [L] to tributaries leaving subbasin
* 14        transmission loss [L]
* 15        lateral flow [L] to tributaries leaving subbasin
* 16        percolation from the root zone
* 17        groundwater recharge (SWAT's definition); see shed(17,*) for
*           MODFLOW's definition.
*   shed vectors 18 and 19 are passed from SWAT to Modflow, but are superseded
*   by values based on Modflow's aquifer solution and are evaluated in SWB1BD:
* 18        evaporation from water table (used as "revap" in Swat)
* 19        streambed leakage from stream; its negative is used as return flow in Swat).
* 20        pond seepage into aquifer
* 21        tributary inflow = contrb*(shed(13,*) + shed(14,*))
* 22        depth to water
* 23        potential et
* 24        assigned irrigation flux (mm), calculated as a check: if irropt>0,
*           it should equal shed(11,j), from SWAT; otherwise, it should equal
*           shed(9,j), based on MODFLOW's ~.WEL file input.
* 25        groundwater recharge = shed(14,*) + shed(15,*)*(1-shed(8,*))
* 26        recharge flux (mm) corresponding to array RECH as read from
*           Modflow input file; flux is for initial time step duration, delt0,
*           and with respect to area of active nodes in subbasin.
*                                 + shed(16,*) + shed(18,*)
      ISPA=30*nwshed
      ISUM=ISUM+ISPA
C
C4------SPACE NEEDED FOR integer watershed array ished(10,mxshed):
*         (1) not used; formerly, no. streams out of watershed;
*         (2) not used; formerly, index to list of streams out of watershed.
*         (3) not used; formerly, device no. for reading subbasin hydrograph file
*         (4) soil id for subbasin (read but not used)
*         (5) not used; formerly, no. soil layers for subbasin
      ICSHED=ISUM
      ISPB=10*nwshed
      ISUM=ISUM+ISPB
C
C5------SPACE NEEDED FOR watershed outflow array ishout(5,mxshed):
*         (1) not used; formerly, list of streams out of subbasins;
*         (2) itrshd: watershed from which stream flows;
*         (3) itrseg: segment of stream into which tributary flows;
*         (4) itrrch: reach of stream within a segment into which tributary flows.
*         (5) idxrch: index to order in which a given (segment,reach) was read
*                     from the STREAM input file (~.str); found by a search
*                     in subr SWB1RP (below).
      LCOUTF=ISUM
      ISPC=5*noutfl
      ISUM=ISUM+ISPC
C6------CALCULATE AMOUNT OF SPACE NEEDED FOR watershed-aquifer grid index.
      LCBSHD=isum
      write (iout,'(1x,a,i8)')
     1    'watershed-aquifer grid ibshed index LCBSHD =',lcbshd
      ispd=ncol*nrow
      isum=isum+ispd
*         space for soils data:
*         (1) zsoil(i,j): depth to bottom of each layer i of soil j
      lczsol=isum
      ispe=mxslay*nsoils
      isum=isum+ispe
*         (2) wcsoil(i,j): water content (volumetric fraction of soil layer)
*             for each layer i of subbasin j:
      lcwsol=isum
      ispf=mxslay*nwshed
      isum=isum+ispf
*         Add space for max evap rate, dtw and evap
*         if evap module isn't invoked:
      if (iunit(5).eq.0) then
          ispg = 3
          lcevtr = isum        !x(lcevtr): evtr(ncol,nrow)
          isum = isum + 1
          lcdtw  = isum        !x(lcdtw):  dtw(ncol,nrow)
          isum = isum + 1
          lcevap = isum        !x(lcevap): evap(ncol,nrow)
          isum = isum + 1
      else
          ispg = 0
      end if
*         space for initial recharge matrix whose distribution is to be
*         maintained for each time step by scaling element values by
*         (psi_s/psi_r), where psi_s = subbasin recharge flux specified
*         by Swat and psi_r = subbasin recharge flux corresponding to the
*         initial recharge matrix specified in the recharge input file;
*         psi_r = Q_r/A, where:
*             Q_r = sum over active nodes i,j in subbasin [rech(i,j)*area(i,j)]
*             A   = sum over active nodes i,j in subbasin [area(i,j)]
      lcrchi=isum
      ispj = ncol*nrow
      isum=isum+ispj
C         total space used in x by soil water balance package so far:
      ISP=ISPA+ISPB+ISPC+ISPD+ispe+ispf+ispg+ispj
C
C9------PRINT AMOUNT OF SPACE USED BY WATERSHED INTERFACE PACKAGE.
      WRITE (IOUT,8)ISP
    8 FORMAT(1X,I8,' ELEMENTS IN X ARRAY ARE USED FOR SWB')
      ISUM1=ISUM-1
      WRITE(IOUT,9)ISUM1,LENX
    9 FORMAT(1X,I8,' ELEMENTS OF X ARRAY USED OUT OF',I7)
      IF(ISUM1.GT.LENX) WRITE(IOUT,10)
   10 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C
C10-----RETURN.
      RETURN
      END
```

Swb1rp: map subbasin areas onto grid, and subbasin outflows ongo stream network

Read subbasin, tributary and watershed correspondence to aquifer.

```
      subroutine SWB1RP(SHED,ISHED,ISHOUT,IBSHED,ibound,DELR,DELC,
     1  ISTRM,ZSOIL,well,dtw,evap,mxwell,nwells,nwshed,noutfl,nsoils,
     1  mxslay,irropt,itmuni,cnvtim,cnvlen,bsarea,actfrc,kkper,mxstrm,
     1  nstrem,nrow,ncol,nlay,iunit,iout,iolog)
c usage:
cc    IF(IUNIT(6).GT.0) call SWB1RP (x(LCSHED),x(ICSHED),x(LCOUTF),      swb1
cc   1  x(LCBSHD),X(LCDELR),X(LCDELC),x(ICSTRM),x(LCZSOL),x(LCWELL),
cc   1  mxwell,nwells,nwshed,noutfl,nsoils,mxslay,irropt,itmuni,
cc   1  cnvlen,bsarea,kkper,mxstrm,nstrem,nrow,ncol,iunit,iout,iolog)
C
C-----VERSION  1   1May1995 SWB1RP                                     C
C     ****************************************************************C
C     READ watershed DATA:
*     a) list of tributaries; corresp. watershed (source) and
*        stream reach (destination);
*     b) list of watersheds; corresponding tributaries (outflows);
*     c) watershed-aquifer grid correspondence (array ibshed(ncol,nrow))
C     ****************************************************************C
C                                                                     C
C     SPECIFICATIONS:                                                 C
C     ----------------------------------------------------------------C
      DIMENSION SHED(30,nwshed),ISHED(10,nwshed),ISHOUT(5,noutfl),
     1    istrm(5,mxstrm),ibshed(ncol,nrow),ibound(ncol,nrow,nlay),
     1    DELR(ncol),DELC(nrow),zsoil(mxslay,nsoils),WELL(6,mxwell),
     1    dtw(ncol,nrow),evap(ncol,nrow),iunit(24)
      dimension numshd(50)
```

```
C     --------------------------------------------------------------C
      character recinp*132, fmtbnd*20
      dimension tsec(0:5)
*         time period (sec) for each time unit specified in Modflow by itmuni
*         (use 1 here for unspecified option itmuni=0); itmuni is returned
*         from initializing call to Modflow.
*                    undefined, sec, min, hour,  day,   year(365 days)
      data (tsec(j),j=0,5) /1.,   1., 60., 3600., 86400., 31536000./
C                                                                     C
      IF(nwshed.LT.1) RETURN
      in = iunit(6)
C                                                                     C
*         For now, this routine initializes variables that are to remain
*         constant over all stress periods:
      if (kkper.eq.1) then
*             read subbasin-tributary correspondence from SWB input file:
*             this list is now unnecessary since only one outflow per
*             subbasin is allowed; retain the list in order to avoid
*             changing the input. Read list heading and then list of
*             nwshed records numbered 1 through nwshed, where nwshed= no.
*             subbasins:
        read (in,'(1x,a)') recinp   !header
        do i=1,nwshed
          read (in,'(a)') recinp    !number these records 1 to nwshed.
        end do
*            read length conversion factor cnvlen, which is used to convert
*            SWAT flux units into MODFLOW flux units:
        read (in,*) cnvlen      ! (e.g. cnvlen=1000 mm/m, or 304.8 mm/ft)
        cnvtim = tsec(itmuni)/86400.  !days/Modflow time unit
        write (iolog,'(1x,a,i5,1x,a,f10.3)') 'SWB1RP: nwshed=',nwshed,
     1      'cnvlen=',cnvlen
*           basin area and areal fractions for each subbasin:
*           these should be the same as read by SWAT, except that SWAT reads
*           the basin area in km^@, and bsarea should be in units of [L^2].
        read (in,*) bsarea, (shed(1,i),i=1,nwshed)
*           soil id for each subbasin (read but not used):
        read (in,*) (ished(4,i),i=1,nwshed)
        do i=1,nwshed
          shed(2,i) = bsarea*shed(1,i)       !area [L^2] of each subbasin
        end do
*           for each soil, no. layers and depth to bottom of each layer:
*           (not currently used)
        do i=1,nsoils
          read (in,*) nlsoil, (zsoil(j,i),j=1,nlsoil)
          write (iout,'(2i5,12f10.2)') i, nlsoil,
     1        (zsoil(j,i),j=1,nlsoil)
        end do
        write (iout,'(1x,a)') 'SWB1RP: Subbasin outflow'
        write (iout,'(1x,a)') 'location in stream network (idx='//
     1    'input order of reach in stream file)'
        write (iout,'(1x,a)') ' sub  seg  rch  idx  row  col'//
     1                      ' name      subbas fraction'
*             read two lines of headings for subbasin outflow table:
        do j=1,2
          read (in,'(1x,a)') recinp
        end do
*         For each subbasin, read the following:
*         subbasin index, tributary index, stream segment, and stream reach;
*         make sure that subbasin index and tributary index are equal, and
*         in ascending order from 1 to nwshed, so that i = idxsub = idxtrb.
        do i=1,nwshed
          read (in,*) idxsub,idxtrb,idxseg,idxrch
          ishout(2,i)=idxsub  !subbasin index (= tributary index)
          ishout(3,i)=idxseg  !index to stream segment
          ishout(4,i)=idxrch  !index to stream reach
*             Find the index to the order in which this (segment,reach)
*             was read from the STREAM (~.str) input file, to allow direct
*             indexing of the (segment,reach) for subbasin outflow; set up a
*             default to the first reach in case no match is found.
          ishout(5,i) = 1
          do k=1,nstrem
              if (idxseg.eq.istrm(4,k)) then
                  if (idxrch.eq.istrm(5,k)) then
*                         match was found on (segment,reach) for subbasin outflow:
                      ishout(5,i) = k
```

```
                    go to 30
                 else if (istrm(4,k).eq.1) then
*                       this is better than nothing: first reach of same segment
                    ishout(5,i) = k
                 end if
              end if
           end do
*          the (segment,reach) identified for watershed outflow wasn't found.
           print *,'SWB1RP: Stream reach and segment mismatch '//
     1        'between subbasin outflow and stream segment:'
           print '(5(1x,a,i5))', 'SWB1RP: subbasin',i,
     1        'tributary outflow to stream segment',idxseg,
     1        'reach',idxrch,'not found in stream network.'
30         continue
           write (iout,'(6i5)') (ishout(j,i),j=2,5),istrm(2,k),istrm(3,k)
         end do
         numact=0 !total no. active nodes (from ibound array)
         numzon=0 !total no. nodes within subbasins (zones > 0)
         numaz=0  !total no. active nodes within subbasins
         activa=0.
         do izone=1,nwshed
           numshd(izone)=0    !initialize no. active nodes in each subbasin
           shed(3,izone)=0.   !initialize active gridded area in each subbasin
         end do
*            read watershed zones array (ibshed) following the same way that the
*            ibound array is read from the BAS input file (see Modflow manual):
         read (in,'(2i10,a20,i10)') locat, iconst, fmtbnd, iprn
         write (iout,'(2i10,a20,i10)') locat, iconst, fmtbnd, iprn
         do ir=1,nrow
           read (in,fmtbnd) (ibshed(ic,ir),ic=1,ncol)
           write (iout,'(1x,i4,25i3/,(5x,25i3))')
     1        ir,(ibshed(ic,ir),ic=1,ncol)
*             accumulate gridded area by subbasin:
           do ic=1,ncol
              izone = ibshed(ic,ir)
              if (ibound(ic,ir,1).gt.0) numact=numact+1
              if (izone.gt.0) then
                 numzon=numzon+1
                 if (ibound(ic,ir,1).gt.0) then
                    numshd(izone)=numshd(izone)+1
                    numaz=numaz+1
                    acell = delr(ic)*delc(ir)
                    activa = activa + acell
                    shed(3,izone) = shed(3,izone) + acell
                 end if
              end if
           end do
         end do
         actfrc = activa/bsarea
         cnvday = cnvlen/cnvtim
         cnvyr = cnvlen*(365./cnvtim)
         do izone=1,nwshed
*             gridded active area as a fraction of actual area:
           shed(7,izone) = shed(3,izone)/shed(2,izone)
         end do
*
         write (iolog,'(1x,a,i3,1x,a,f12.1)')
     1       'SWB1RP: Modflow itmuni=',itmuni,
     1       'tsec(itmuni)=',tsec(itmuni)
         write (iolog,'(1x,a,g15.7)') 'days/Modflow time unit = '//
     1       'tsec(itmuni)/86400 =',cnvtim
         write (iolog,'(1x,a,3(i5,1x,a))') 'SWB1RP: ',
     1     numact, 'active nodes (ibound>0)',
     1     numzon, 'nodes in zones (izone>0)',
     1     numaz,  'active nodes in zones (ibound>0 and izone>0)'
         write (iolog,'(1x,a10,12x,9i12/,(23x,9i12))')
     1     'subbasin',(izone,izone=1,nwshed)
         write (iolog,'(1x,a10,12x,9i12/,(23x,9i12))')
     1     'no. cells',(numshd(izone),izone=1,nwshed)
         write (iolog,'(1x,a10,10e12.4/,(23x,9e12.4))')
     1     'actual',bsarea, (shed(2,izone),izone=1,nwshed)
         write (iolog,'(1x,a10,10e12.4/,(23x,9e12.4))')
     1     'active',activa, (shed(3,izone),izone=1,nwshed)
         write (iolog,'(1x,a10,10f12.4/,(23x,9f12.4))')
     1     'gridded',actfrc, (shed(7,izone),izone=1,nwshed)
```

```
*
*           initial evaporation rate from water table:
        if (iunit(5).ne.0) then
          evtflx=0.
          dtwavg=0.
          do izone=1,nwshed
              shed(18,izone)=0.  !evap from water table
              shed(23,izone)=0.  !avg depth to water
          end do
          do ir=1,nrow
              do ic=1,ncol
                  izone = ibshed(ic,ir)
                  if (izone.gt.0 .and. ibound(ic,ir,1).gt.0) then
                      acell = delr(ic)*delc(ir)
*                             Note: at this point, et has already been
*                             multiplied by grid cell area via array evtr
*                             in subr EVT1RP.  Calculate subbasin et
*                             as Swat daily flux [mm/day]:
                      shed(18,izone) = shed(18,izone) + evap(ic,ir)
*                             (active cell) area-weighted avg depth to water [L]:
                      shed(23,izone) = shed(23,izone) + acell*dtw(ic,ir)
                  end if
              end do
          end do
          evtflx = 0.
          do izone=1,nwshed
              shed(18,izone) = ((cnvlen/cnvtim)/shed(2,izone))
     1            *shed(18,izone)
              evtflx = evtflx + shed(1,izone)*shed(18,izone)
              shed(23,izone) = shed(23,izone)/shed(3,izone)
              dtwavg = dtwavg + shed(1,izone)*shed(23,izone)
          end do
          write (iolog,'(1x,a)') 'Evaporation rate from initial WT:'//
     1        ' (Fluxes based on total subbasin areas)'
          i=18
          write (iolog,'(25x,i3,1x,a,f7.1,7x,10f8.2/,(t53,10f8.2))')
     1            i,'WTev,mm/d', evtflx, (shed(i,j),j=1,nwshed)
          write (iolog,'(1x,a)') 'Depth to water:'//
     1        ' (Fluxes based on active gridded areas)'
          i=23
          write (iolog,'(25x,i3,1x,a,f7.1,7x,10f8.2/,(t53,10f8.2))')
     1            i,'avg DTW[L]', dtwavg, (shed(i,j),j=1,nwshed)
        end if
      end if  !kkper=1
*
*         For each stress period (assumed to be 1 yr), initialize total pumping
*         rate for each subbasin as specified in MODFLOW's WELL input file (~.wel).
*             The negative sign is introduced to cancel the sign difference
*         between the SWAT irrigation flux (positive) and MODFLOW's
*         well flow rate (negative, i.e. pumping out of the aquifer).
*         NOTE: on a combined SWAT-MODFLOW run, the corresponding flux, shed(9,*),
*         is assumed to be available in MODFLOW's SHED array just prior to
*         entering the daily loop in SWAT's mainline, where shed(9,*)
*         is retrieved by SWAT's subr PASSFLX.
      do i=1,nwshed
          shed(4,i)=0. ! accumulator for negative of subbasin well pumping rates
      end do
      qbc=0.
      qsum=0.
      qact=0.
      qzone=0.
      qaz=0.
      do L=1,nwells
          il = well(1,L)
          ir = well(2,L)
          ic = well(3,L)
          iwtype = well(6,L) !well type(0:irrig; 1:b.c. flux or recharge)
          if (iwtype.eq.1) then
              qbc = qbc + well(5,L)
          else
              izone = ibshed(ic,ir)
              qsum = qsum + well(5,L)
              if (ibound(ic,ir,1).gt.0) qact = qact + well(5,L)
              if (izone.gt.0) then
                  qzone=qzone+well(5,L)
                  if (ibound(ic,ir,1).gt.0) then
                     qaz = qaz + well(5,L)
*                        total pumping rate for each subbasin:
*                        add the negative of the value read from ~.wel.
                     shed(4,izone) = shed(4,izone) - well(5,L)
                  end if
               end if
            end if
         end do
         write (iolog,'(1x,a,i3,(1x,g15.7,1x,a))')
     1      'SWB1RP: stress period',kkper,qsum,'total pumping',
     1      qact,'pumping in active nodes (ibound>0)',
     1      qzone,'pumping in watershed zones (izone>0)',
     1      qaz,'pumping in active watershed zones '//
     1          '(ibound>0 and izone>0)',
     1          qbc,'total for wells representing flux b.c. and recharge'
         write (iolog,'(1x,a)') 'SWB1RP: total pumping by subbasin '//
     1        'specified by Modflow ~.wel input:'
         sumpmp=0.
         do j=1,nwshed
            sumpmp=sumpmp+shed(4,j)
            shed(9,j) = cnvyr*shed(4,j)/shed(2,j)
         end do
         pmpflx = cnvyr*sumpmp/bsarea
         i=4
         write (iolog,
     1      '(i3,1x,a,g11.3,8(i3,g11.3)/,(25x,8(i3,g11.3)))')
     1               i,'Pump, L3/T', sumpmp, (j,shed(i,j),j=1,nwshed)
         write (iolog,'(1x,a,t53,10i8/,(t53,10i8))')
     1      ' basin  ',(j,j=1,nwshed)
210      format (1x,a,t53,10i8/,(t53,10i8))
         i=9
         write (iolog,'(1x,a)') 'Pumping fluxes (based on actual '//
     1      'subbasin areas; excludes flux b.c. and recharge wells):'
         write (iolog,'(25x,i3,a,f7.1,7x,10f8.2/,(t53,10f8.2))')
     1               i,'Pump,mm/yr', pmpflx, (shed(i,j),j=1,nwshed)
         RETURN
         END
*----------------------------------------------------------------------
```

Swb1fm: specify tributary flow, recharge, potential evaporation, and irrigation

```
Read subbasin flux for one time step; set up corresponding conditions for solution.

      subroutine SWB1FM (SHED,ISHED,ISHOUT,IBSHED,EVTR,dtw,evap,IRCH,
     1   RECH,rchinp,ibound,DELR,DELC,istrm,strm,zsoil,wcsoil,mxstrm,
     1   nstrem,well,mxwell,nwells,nwshed,noutfl,nsoils,mxslay,irropt,
     1   ievopt,ioprch,iopswt,jkkopt,cnvtim,cnvlen,delt,bsarea,actfrc,
     1   kkper,kkstp,iss,nrow,ncol,nlay,iunit,iout,iolog)
C
* usage:  call from mainline moddw to map watershed hydrologic fluxes
*         from subbasins onto corresponding aquifer grid cells.
c
C-----VERSION  1  26May1995 SWB1FM                                    C
C     ****************************************************************C
C     READ results for current time step from SWAT output file.
C     ****************************************************************C
C     SPECIFICATIONS:                                                 C
C     ----------------------------------------------------------------C
      DIMENSION SHED(30,nwshed),ISHED(10,nwshed),ISHOUT(5,noutfl),
     1    IBSHED(ncol,nrow), EVTR(ncol,nrow),dtw(ncol,nrow),
     1    evap(ncol,nrow),IRCH(ncol,nrow),RECH(ncol,nrow),
     1    rchinp(ncol,nrow),ibound(ncol,nrow,nlay),
     1    DELR(ncol),DELC(nrow),
     1    ISTRM(5,mxstrm),STRM(20,mxstrm),zsoil(mxslay,nsoils),
     1    wcsoil(mxslay,nwshed),WELL(6,mxwell),iunit(24)

      character*50 fmtswb  ! format for record read by subr swbhyd (below)
      character*30 nambal ! file name for hydrologic balance file if specified
      character*10 colhdg(30)  ! label read with hydrologic component from swb file
      character recinp*132, coltxt*10
*         vector for avg basin flux: basflx (Modflow units), swtflx (Swat units)
      dimension swtflx(30), flxdev(30)
*         subbasin sums: no. active nodes, recharge,
*         evapotrans from water table, pumping, and tributary outflow:
      dimension numshd(50)
      dimension subval(100)
      data (colhdg(i),i=1,26) /
     1     'Areal frc', ' area[L^2]','GrdAreaL^2','Irr_use[L]',         ! 1- 4
     1     'IrrSwatFrc','IrrRet_frc','AreGrd/Act','Noncontrib',         ! 5- 8
     1     'Irr_use   ','  Prcp mm',                                    ! 9-10
     1     ' Irrig mm ','    ET mm ','  SURQ mm ','XMLoss mm',          !11-14
     1     '  LATQ mm ','  PERC mm ',' GWRch mm ',' GW ET mm ',         !15-18
     1     'Basflow mm','PndSeep mm',' POT ET mm','TrbInfl mm',         !19-22
     1     '  DTW [L] ','Pmprat mm ',' GWRch mm ','RchInp     '/        !23-26
      data fmtswb /'(25x,i3,1x,a,20f8.2)'/
ccc   data timstp /86400./ ! sec/day
C**   data daystp /365./   ! day/step
C**   data cnvfac /304.8/  ! mm/ft
C**   data cnvlen /1000./  ! mm/m
      data itrace /1/
C     ---------------------------------------------------------------
C
C1A-----IF mxshed IS LESS THAN 1 THEN MODSWB IS INACTIVE:
      IF(nwshed.LT.1) RETURN
      in = iunit(6)
*
*         read array SHED vectors 10-20 from the flux data written by SWAT.

      write (iolog,'(6(2x,a,i5))') 'SWB1FM: kkper=',kkper,
     1    'kkstp=',kkstp,'irropt=',irropt,'nwells=',nwells,
     1    'nwshed=',nwshed,'iopswt=',iopswt
      if (kkstp.eq.1) then
*         For only the first time step of each stress period, initialize total
*         pumping rate for each subbasin as specified in MODFLOW's WELL input
*         file (~.wel).
*             The negative sign is introduced to cancel the sign difference
*         between the SWAT irrigation flux (positive) and MODFLOW's
*         well flow rate (negative, i.e. pumping out of the aquifer).
          do i=1,30
              swtflx(i)=0.
              flxdev(i)=0.
          end do
*             If Modflow is running in stand-alone mode (not called by SWAT,
*             i.e. iopswt=0), then read header lines from the hydrologic flux
*             file:
*         For only the first time step of the first stress period, either:
*         (a) read header lines of flux file written by SWAT appended to ~.swb;
*         (b) read the hydrologic balance file name (~.bal) following the
*         ibshed array, close the ~.swb file, open the ~.bal file, and
*         proceed to read the ~.bal file as if it were appended to the ~.swb
*         file.
          if (kkper.eq.1) then
              if (iopswt.eq.0) then
                  read (in,'(a)') recinp
                  read (recinp,'(4i5)',err=100,iostat=iostat)
     1                nyrs,iyr,lutot,jkkopt
100               if (iostat.eq.0) then
                      write (iolog,'(1x,a)') 'read hydrologic '//
     1                    'balances appended to file ~.swb:'
                  else
                      read (recinp,'(a)') nambal  !hydrologic flux file (~.bal)
                      write (iolog,'(1x,a)')
     1                    'read hydrologic balances from file '// nambal
                      close (unit=in)             !close the ~.swb file
                      open (unit=in,file=nambal,status='old',
     1                    err=110,iostat=iostat) !open the ~.bal file
110                   if (iostat.eq.0) then
                          read (in,*) nyrs, iyr, lutot, jkkopt  !~.bal file rec. 1
                      else
                          write (iolog,'(1x,a)') 'SWB1FM error: '//
     1                    'balance file not found: '//nambal
                          write (iolog,'(1x,a)') '(name read from '//
     1                    'end of ~.swb file following ibshed array)'
                          STOP
                      end if
                  end if
                  write (iolog,'(4i5)') nyrs,iyr,lutot,jkkopt
                  read (in,'(a)') fmtswb     !~.bal file rec. 2
```

```
                 write (iolog,'(1x,a)')
     1                'Subbasin flux input format = '//fmtswb
             end if
             write (iolog,'(3(1x,a,g15.7),1x,a)')
     1           'cnvlen=',cnvlen,'mm/m, Basin area=',bsarea,'m^2'
             write (iolog,'(1x,a,i2,1x,a)') 'SWB1FM: Tributary flow'//
     1           ' and recharge are defined as follows for jkkopt=',
     1           jkkopt, '(default=0):'
             if (jkkopt.eq.0) then
                 write (iolog,'(1x,a)')
     1                'Q_trib(22) = c*[Q_surf(13) + Q_subsurf(15)]'
                 write (iolog,'(1x,a)') 'GW Rech(25) = XMLoss(14) +'//
     1                ' Perc_rz(16) + Pond seepage(20)'
             else
                 write (iolog,'(1x,a)') 'Q_trib(22) = '//
     1                'c*[Q_surf(13) - (1/c)*XMloss(14)]'
                 write (iolog,'(1x,a)') 'GW Rech(25) = XMLoss(14) +'//
     1                ' c*Q_subsurf(15) + Perc_rz(16) + Pond seepage(20)'
             end if
             write (iolog,'(1x,a)')
     1           '(c = 1 - noncontributing fraction)'
         end if
      end if
      if (iopswt.eq.0) then
         idxhdg=0
         do while (idxhdg.eq.0)
             read (in,'(a)') recinp
             idxhdg = INDEX(recinp,'hydrologic balance')
         end do
         read (recinp,'(f15.0)') dtswat
         delt = dtswat/cnvtim
         read (in,'(t53,10i8)') (idx,j=1,nwshed)   ! column heading
         read (in,220) avgval, (subval(j),j=1,nwshed)
220          format (f15.3,t53,10f8.5/,(t53,10f8.5))
221      format (f15.3,a,t53,10f8.5/,(t53,10f8.5))
         read (in,225) fpd,(shed(8,j),j=1,nwshed)
225          format (t39,f8.5,6x,10f8.5/,(t53,10f8.5))
226          format (t2,a,t39,f8.5,6x,10f8.5/,(t53,10f8.5))
         read (in,225) avgval,(subval(j),j=1,nwshed)
ccc      write (iolog,221) avgval,' km**2; fractions:', ! subbasins: fractions of basin
area
ccc  1       (subval(j),j=1,nwshed)
ccc      write (iolog,226) 'ponds (areal fractions):', ! ponds: subbasin fractions
ccc  1           fpd,(shed(8,j),j=1,nwshed)
ccc      write (iolog,226) 'gridded area fractions: ',
ccc  1           avgval,(subval(j),j=1,nwshed)
*     Note: pond seepage is passed as a flux, but with respect to the subbasin
*     area rather than only the subbasin's noncontributing area (that which
*     drains to the ponds); so divide the pond flux by shed(8,izone) to
*     convert to a flux with respect to the subbasin's noncontributing area.
*(25x,i3,a,2f7.1,10f8.2/,(t53,10f8.2))
* 1  1  0  0  0  1  1   1 10   PREC MM  607.7    19.7  650.90  650.90  650.90  593.34
593.34  593.34  593.34  596.53  596.53  596.14
*                                                    596.14  596.53  596.53  596.53
596.14  596.53  596.53  596.53  596.53  601.06
*                                                    607.96  607.96  607.96  607.96
601.06  607.96  607.96  663.46  607.96
********* read (in,'(a)') recinp  !subbasin area fractions (already read
*            as shed(1,*) in swb1rp)
         write (iolog,'(1x,a,11a)')
     1       'Read fluxes for ',(colhdg(i),i=10,20)
         do i=10,20
             read (in,fmtswb) idx, coltxt,
     1           swtflx(i), flxdev(i), (shed(i,j),j=1,nwshed)
             if (idx.ne.i) write (iolog,'(1x,a,i3,1x,a)')
     1           'SWB1FM: '//colhdg(i)//'i=',i,
     1           'mismatch to ~.swb file idx=',idx
c            write (iolog,fmtswb) idx, coltxt,
c    1           swtflx(i), flxdev(i), (shed(i,j),j=1,nwshed)
         end do
      else
         do i=10,20
             swtflx(i)=0.
             flxdev(i)=0.
             do j=1,nwshed
```

```
                  swtflx(i) = swtflx(i) + shed(1,j)*shed(i,j)
              end do
              do j=1,nwshed
                  flxdev(i) = flxdev(i) +
     1                shed(1,j)*(shed(i,j)-swtflx(i))**2
              end do
              flxdev(i) = SQRT(flxdev(i))
          end do
      end if
      activa = bsarea*actfrc
      write (iolog,'(3(1x,a,g15.7))')
     1     'SWB1FM: delt=dtswat/cnvtim=',delt
cc    write (iolog,'(1x,a,t40,i7,t53,10i8/,(t53,10i8))')
cc   1      'No. active nodes',nactiv,(numshd(j),j=1,nwshed)
cc    write (iolog,'(1x,a,t40,f7.4,t53,10f8.4/,(t53,10f8.4))')
cc   1      'Active/subbasin area ratios',actfrc,(shed(7,j),j=1,nwshed)
*             tributary flow rate = (runoff+subsurface flux)*
*                 contributing subbasin area;
*             enter tributary flow as lateral inflow to this reach;
*             see above for calculation of (22), contribution to streamflow
*             from subbasin:
*             13: surface runoff; 14: transmission loss; 15: lat. subsurf. flow
      do izone=1,nwshed
*             contributing fraction of subbasin:
          contrb = (1.-shed(8,izone))
*             jkkopt is read from Swat's ~.cod file and passed to here:
          if (jkkopt.eq.0) then
*                 Default definitions for recharge and tributary flow:
*                 (25) groundwater recharge includes (14) transmission loss,
*                 (16) percolation through the root zone, and (20) pond seepage.
              shed(25,izone) = shed(14,izone) + shed(16,izone)
     1            + shed(20,izone)

*                 (22) tributary flow rate (default) includes (13) surface
*                 runoff and (15) subsurface lateral flow, both reduced by the
*                 contributing area fraction of the subbasin; the remainder of
*                 these two terms ends up in the ponds.
              shed(22,izone) = contrb*(shed(13,izone) + shed(15,izone))
*
*             Definitions for (25) recharge and (22) tributary flow rate are
*             modified if the option jkkopt>0 is invoked.  Revised definitions
*             are based on Prof. Jim Koelliker's conceptual model for the
*             Rattlesnake basin; hence the option's name.  --spp jan 4 96
          else
*                 (25) recharge also includes lateral subsurface flow (15),
*                 reduced by the contributing area fraction;
              shed(25,izone) = shed(14,izone) + contrb*shed(15,izone)
     1            + shed(16,izone) + shed(20,izone)

*                 (22) tributary flow rate [L/T] is given by
*                 contrb*(surface runoff) - (1/contrb)*(transmission loss):
              shed(22,izone) = contrb*shed(13,izone)
     1           - (1./contrb)*shed(14,izone)
          end if
      end do
*         mean for tributary flow(22) & recharge(25):
      do i=22,25,3
          swtflx(i)=0.
          do j=1,nwshed
              swtflx(i) = swtflx(i) + shed(1,j)*shed(i,j)
          end do
      end do
*         std deviation for tributary flow(22) & recharge(25):
      do i=22,25,3
          flxdev(i)=0.
          do j=1,nwshed
              flxdev(i) = flxdev(i) +
     1            shed(1,j)*(shed(i,j)-swtflx(i))**2
          end do
          flxdev(i) = SQRT(flxdev(i))
      end do
*
      do j=1,nwshed
          numshd(j)=0
      end do
```

```
*         For option ioprch=1: set up initial recharge matrix whose
*         distribution will be maintained for each time step by scaling
*         element values by the product of time and recharge flux ratios
*         (delt0/delt)*(shed(25,j)/shed(26,j)), where
* delt0 = initial time step duration, Modflow units;
* delt = time step i duration, Modflow units;
* shed(17,j) = subbasin recharge flux (mm) specified by Swat for time step i.
* shed(25,j) = subbasin recharge (mm) for MODFLOW as sum of SWAT terms (above)
* shed(26,j) = recharge flux (mm) corresponding to array RECH as read from
*         Modflow input file; flux is for initial time step duration, delt0,
*         and with respect to area of active nodes in subbasin;
      if (ioprch.gt.0 .and. kkper.eq.1 .and. kkstp.eq.1) then
          delt0 = delt
          do ir=1,nrow
              do ic=1,ncol
                  izone = ibshed(ic,ir)
*                         rech was already multiplied by cell area in RCH1RP
*                         to convert from flux to flow rate.
                  rchinp(ic,ir) = rech(ic,ir)

                  if (izone.gt.0.and.ibound(ic,ir,1).gt.0) then
                      shed(26,izone) = shed(26,izone) + rech(ic,ir)
                  end if
              end do
          end do
          rchflx=0.
          do j=1,nwshed
              shed(26,j) = delt*cnvlen*shed(26,j)/shed(3,j)
              rchflx = rchflx + shed(1,j)*shed(26,j)
          end do
          i=26
          write (iolog,'(1x,a,g15.7)') 'Recharge flux (Modflow input'//
     1        ' ~.rch); initial delt =',delt0
          write (iolog,'(25x,i3,a,f7.1,7x,10f8.2/,(t53,10f8.2))')
     1        i,colhdg(i), rchflx, (shed(i,j),j=1,nwshed)
      end if
*         root zone percolation component from precipitation is distributed
*         equally over subbasin zone of recharge array:
*             For each grid cell:
*         Recharge (L**3/T) is obtained by multiplying recharge flux by
*         grid cell area; similarly for evaporation at ground surface.
*        areal weight is adjusted for gridded subbasin discrepancy:
*            Note regarding area discrepancy between watershed shed(2,*)
*        and active gridded aquifer shed(3,*): --spp 12/30/95
*        Flux over active area was previously increased by ratio of subbasin's
*        area to its active gridded area; this was done by dividing recharge
*        and evaporative fluxes by the ratio shed(3,izone)/shed(2,izone).
      qrech=0.
      nactiv=0
      do ir=1,nrow
          do ic=1,ncol
              izone = ibshed(ic,ir)
              if (izone.gt.0.and.ibound(ic,ir,1).gt.0) then
                  nactiv=nactiv+1
                  numshd(izone) = numshd(izone) + 1
                  rcarea = delr(ic)*delc(ir)
                  if (ioprch.gt.0) then
                      rech(ic,ir) = rchinp(ic,ir)*(delt0/delt)*
     1                             (shed(25,izone)/shed(26,izone))
                  else
                      rech(ic,ir) = shed(25,izone)*rcarea/(delt*cnvlen)
                  end if
                  qrech = qrech + rech(ic,ir)
*                     evaporation at ground surface: EVT module is used;
*                     shed(21,*) is supposed to be potential et.
*                     NOTE:
*                     if ievopt > 0, evaporation from the water
*                         table is controlled by potential evaporation at
*                         ground surface as calculated by Swat; however,
*                         it is coded below as the "actual" et as calculated
*                         by SWAT.  The next version of Swat-Modflow, i.e.
*                         swatmod96, is a little further along than this version
*                         on properly representing evapotranspiration from the
*                         water table.
*                     otherwise (ievopt < or = 0), evap at ground surface is
*                           specified by Modflow input to the EVT module.
*
                    if (iunit(5).ne.0.and.ievopt.gt.0) then
                        evtr(ic,ir) = shed(12,izone)*rcarea/(delt*cnvlen)
                    end if
                else if (izone.eq.0) then
                    rech(ic,ir) = 0.
                end if
            end do
        end do
*           shed(4,j):     total pumping rate [L^3/T] for each subbasin as
*                          accumulated over wells read from ~.WEL file (MODFLOW)
*           shed(9,j):     flux(mm) corresponding to total pumping, shed(4,j),
*                          specified by MODFLOW's ~.WEL file for current time step
*                          (e.g. mm/day, mm/month, mm/yr).
*           shed(11,j):    irrigation flux (mm) specified by SWAT for current time step.
*           shed(24,j):    assigned irrigation flux (mm), calculated as a check:
*                          if irropt>0, it should equal shed(11,j), from SWAT;
*                          otherwise, it should equal shed(9,j), based on MODFLOW's
*                          ~.WEL file.
        do j=1,nwshed
            shed(9,j)=0.
            shed(24,j)=0.
            if (shed(4,j).gt.0.)
       1        shed(9,j) = shed(4,j)*cnvlen*delt/shed(2,j)
        end do
        if (nwells.gt.0 .and. irropt.gt.0)
       1    write (iolog,'(1x,a)') 'SWB1FM: irropt > 0; adjust '//
       1    ' ~.wel pumping rates to match SWAT irrigation fluxes.'
        qirrig=0.
        qbc=0.
        do L=1,nwells
            il = well(1,L)
            ir = well(2,L)
            ic = well(3,L)
            well(4,L)=0.
            iwtype = well(6,L)
            j = ibshed(ic,ir)
*               include only wells in active cells:
            if (j.gt.0.and.ibound(ic,ir,1).gt.0) then
*               If irropt > 0, then scale the rate of each pumping well in the
*               subbasin by the ratio shed(11,i)/shed(9,i) so that the subbasin's
*               total pumping flow rate corresponds to the flux specified by SWAT;
*               otherwise (irropt=0), use the rate specified in Modflow's well
*               input file ~.wel.
*                   Note: wells are now distinguished by type:
                if (iwtype.eq.0) then
                    if (irropt.gt.0) then    !pumping rate controlled by SWAT:
                        well(4,L) = well(5,L)*(shed(11,j)/shed(9,j))
                    else                     !pumping rate as specified by MODFLOW input
                        well(4,L) = well(5,L)
                    end if
*                       irrigation pumping rate as flux [Lsw/Tsw] for subbasin:
                    qirrig = qirrig - well(4,L)
                    shed(24,j) = shed(24,j) - well(4,L)
                else
                    well(4,L) = well(5,L)       !b.c., recharge wells
                    qbc = qbc + well(4,L)
                end if
            end if
        end do
        write (iolog,'(1x,a)')
       1      'Irrigation specified by Modflow ~.wel input file:'
        pmpflx=0.
        asgflx=0.
        do j=1,nwshed
            pmpflx = pmpflx + shed(1,j)*shed(9,j)    !irrig. flux (mm) based on Modflow input
            shed(24,j) = shed(24,j)*delt*cnvlen/shed(2,j)
            asgflx = asgflx + shed(1,j)*shed(24,j)
        end do
        i=9
        write (iolog,'(25x,i3,a,f7.1,7x,10f8.2/,(t53,10f8.2))') i,
       1            'Approp,mm ', pmpflx, (shed(i,j),j=1,nwshed)
        i=11
        write (iolog,'(25x,i3,a,f7.1,7x,10f8.2/,(t53,10f8.2))') i,
     1           colhdg(i), swtflx(i), (shed(i,j),j=1,nwshed)
      if (irropt.gt.0) then
          write (iolog,'(1x,a)')
     1        'irrig. flux should match shed(11,*) from SWAT:'
      else
          write (iolog,'(1x,a)') 'irrig. flux should match '//
     1        'shed(9,*) based on MODFLOW input:'
      end if
      i=24
      write (iolog,'(25x,i3,a,f7.1,7x,10f8.2/,(t53,10f8.2))') i,
     1           'Asg.pmp,mm', asgflx, (shed(24,j),j=1,nwshed)
*         initialize the lateral inflow array if it wasn't initialized
*         already in subr SRF1FM:
      if (iunit(14).eq.0) then
          do i=1,nstrem
              strm(18,i)=0.
          end do
      end if
      write (iolog,'(1x,a)') 'Outflow from subbasin 1 to stream '//
     1    'reach in subbasin 2:'
ccc   write (iolog,'(a)') ' sub1 sub2  row  col Qtrib,mm  Rech,mm'
      qtrib=0.
      do j=1,nwshed
*             stream reach associated with subbasin j outflow: idxrch is the
*             index to stream (segment,reach) as read from stream file ~.str.
          idxrch = ishout(5,j)
          ir = istrm(2,idxrch)  !row of stream reach location
          ic = istrm(3,idxrch)  !column of stream reach location
          izone = ibshed(ic,ir) !subbasin in which stream reach lies
*             tributary outflow from subbasin j:
          qtrb = shed(22,j)*shed(2,j)/(delt*cnvlen)
*             accumulate tributary outflow as lateral inflow to stream reach
*             identified by input to SWB1RP as segment iseg, reach ireach,
*             located at row ir, column ic, in subbasin izone.
*             Note: diversions have already been subtracted from strm(18,*)
*             if the MODSRF package is used to specify surface water use.
          strm(18,idxrch) = strm(18,idxrch) + qtrb
          qtrib = qtrib + qtrb
ccc       if (itrace.gt.0) write (iolog,'(4i5,2f9.2)')
ccc  1        j,izone,ir,ic,shed(22,j),shed(25,j)
      end do
      write (iolog,'(4(1x,a,g15.7))') 'SWB1FM: Qirrig=',qirrig,
     1    'Qbc=',qbc,'Qrech=',qrech,'Qtrib=',Qtrib
      RETURN
```

Swb1bd: summarize baseflow and evaporation from shallow ground water

```
Summarize aquifer solution for watershed model (baseflow, evap_wt)

      entry SWB1BD (SHED,ISHED,ISHOUT,IBSHED,EVTR,dtw,evap,IRCH,RECH,
     1    rchinp,ibound,DELR,DELC,istrm,strm,zsoil,wcsoil,mxstrm,nstrem,
     1    well,mxwell,nwells,nwshed,noutfl,nsoils,mxslay,irropt,ievopt,
     1    ioprch,iopswt,cnvtim,cnvlen,delt,bsarea,actfrc,kkper,kkstp,
     1    nrow,ncol,nlay,iunit,iout,iolog)
C
C-----VERSION  1  26May1995 SWB1BD                                  C
C1A-----IF mxshed IS LESS THAN 1 THEN MODSWB IS INACTIVE:
      IF(nwshed.LT.1) RETURN
      in = iunit(6)
*
      do izone=1,nwshed
          shed(18,izone)=0.  !evap from water table
          shed(19,izone)=0.  !baseflow (negative of streambed leakage)
          shed(23,izone)=0.  !avg depth to water
      end do
*           evaporation rate from water table:
      dtwavg = 0.
      if (iunit(5).ne.0) then
          do ir=1,nrow
              do ic=1,ncol
                  izone = ibshed(ic,ir)
                  if (izone.gt.0 .and. ibound(ic,ir,1).gt.0) then
*                             Note: at this point, et has already been
*                             multiplied by grid cell area via array evtr
*                              in subr EVT1RP.  Accumulate et in shed(18) as a
*                              flux rate with respect to total subbasin area:
                       shed(18,izone) = shed(18,izone) + evap(ic,ir)
*                              area-weighted avg depth to water (active cells):
                       wtarea = delr(ic)*delc(ir)/shed(3,izone)
                       shed(23,izone) = shed(23,izone)+wtarea*dtw(ic,ir)
                   end if
               end do
           end do
           do j=1,nwshed
*                  convert accumulated evap from water table [L^3/T]
*                  to flux (mm) w.r.t. subbasin area, shed(2,j):
               shed(18,j) = shed(18,j)*delt*cnvlen/shed(2,j)
*                  avg depth to water for active cell area in subbasin:
               dtwavg = dtwavg + shed(1,j)*shed(23,j)
           end do
           write (iolog,'(1x,a)') 'avg depth to water:'
           i=23
           write (iolog,'(25x,i3,a,f7.1,7x,10f8.2/,(t53,10f8.2))')
     1             i,'avg DTW[L]', dtwavg, (shed(i,j),j=1,nwshed)
      end if
*          Baseflow (from groundwater to stream) is the negative of streambed
*          leakage, which is calculated by the STREAM package (Prudic, 1989):
      do L=1,nstrem
           ir = istrm(2,L)
           ic = istrm(3,L)
           if (ir.le.0.or.ic.le.0) then
               print '(3(1x,a,i6))','stream reach L=',L,
     1             'grid cell not defined: ic=',ic,'ir=',ir
               stop
           end if
           izone = ibshed(ic,ir)
           if (izone.gt.0)
     1         shed(19,izone) = shed(19,izone) - strm(11,L)
      end do
*          convert the sum of baseflow for each subbasin to a flux rate (mm)
*          with respect to the subbasin's total area.
      do j=1,nwshed
           shed(19,j) = shed(19,j)*delt*cnvlen/shed(2,j)
      end do
*          total basin flow rate for GW Et (18) and streambed leakage (19):
      write (iolog,'(1x,a,f6.1,1x,a,f12.0,1x,a,g15.7)')
     1    'cnvlen(mm/L)=',cnvlen,'delt(T)=',delt,'bsarea(L^2)=',bsarea
      write (iolog,210) 'id So Va Aq St To Sb Bas Sh component'//
     1    ' basin  ',(j,j=1,nwshed)
210   format (1x,a,t53,10i8/,(t53,10i8))
      if (itrace.gt.0) then
           write (iolog,'(1x,a)')
     1         'SWB1BD: Results based on Modflow solution:' //
     1                  '(18): REV; (19): GWQ; (23): DTW'
           do i=18,19
               swtflx(i)=0.
               do j=1,nwshed
                   swtflx(i) = swtflx(i) + shed(1,j)*shed(i,j) !basin flow rate (Modflow
units)
               end do
               write (iolog,'(25x,i3,a,f7.1,7x,10f8.2/,(t53,10f8.2))')
     1             i, colhdg(i), swtflx(i), (shed(i,j),j=1,nwshed)
           end do
      end if
      RETURN
      END
```

SURFACE package: representing surface water diversions in MODFLOW

The following source code for the SURFACE package is linked into SWATMOD for the Rattlesnake Creek Basin model. A subsequent version of the package was linked into SWATMOD2 and applied to simulate the Lower Republican River Basin (Sophocleous and Perkins, 1997b; Perkins and Sophocleous, 1997). Because the linking of the SURFACE package into SWATMOD2 was more fully developed and tested as part of the Lower Republican River Basin study, SWATMOD2 was used for a limited number of simulations to examine surface water diversions in the Rattlesnake Creek Basin; the use of SWATMOD2 for this purpose is summarized above in this report.

SRF1AL: Allocate arrays for surface water diversions

```
      SUBROUTINE SRF1AL(ISUM,LENX,LCDVRS,MXSURF,NSURFS,IN,IOUT,
     1                         ISRFCB,filnam)
* usage:
ccc   IF(IUNIT(14).GT.0) CALL SRF1AL(ISUM,LENX,Lcsurf,mxsurf,nsurfs,
ccc  1                  IUNIT(14),IOUT,isrfcb,filnam(14))
C
C-----VERSION 1538 12MAY1987 SRF1AL
C     ******************************************************************
C     ALLOCATE ARRAY STORAGE FOR SURF PACKAGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
        character *(*) filnam
C     ------------------------------------------------------------------
C
C1------IDENTIFY PACKAGE AND INITIALIZE NSURFS
      WRITE(IOUT,1)IN,filnam
    1 FORMAT(1H0,'SRF1 -- SURF PACKAGE, VERSION 1, 9/1/87',
     1' INPUT READ FROM',I3,' file ',a)
      NSURFS=0
C
C2------READ MAX NUMBER OF SURFS AND
C2------UNIT OR FLAG FOR CELL-BY-CELL FLOW TERMS.
      READ(IN,2) MXSURF,ISRFCB
    2 FORMAT(2I10)
      WRITE(IOUT,3) MXSURF
    3 FORMAT(1H ,'MAXIMUM OF',I5,' SURFACE WATER DIVERSION PTS')
      IF(ISRFCB.GT.0) WRITE(IOUT,9) ISRFCB
    9 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE RECORDED ON UNIT',I3)
      IF(ISRFCB.LT.0) WRITE(IOUT,8)
    8 FORMAT(1X,'CELL-BY-CELL FLOWS WILL BE PRINTED WHEN ICBCFL NOT 0')
C
C3------SET LCDVRS EQUAL TO LOCATION OF SURF LIST IN X ARRAY.
      LCDVRS=ISUM
C
C4------ADD AMOUNT OF SPACE USED BY SURF LIST TO ISUM.
*            change from 4*mxsurf to 5*mxsurf to accommodate variation in
*            flow rate with time step, to be determined in the watershed
*            module MODSWB, subr SWBHYD.
*            (1) layer
*            (2) row
*            (3) column
*            (4) segment
*            (5) reach from which flow is diverted.
*            (6) index to order in which stream reaches were read.
*            (7) flow rate as orig. unless irropt>0 (MODSWB), in which
*                case surf(6)=rate read for stress period and surf(7) is
*                scaled according to subbasin surface water use.
*            (8) flow rate (described above);
      ISP=8*MXSURF
      ISUM=ISUM+ISP
C
C5------PRINT NUMBER OF SPACES IN X ARRAY USED BY SURF PACKAGE.
      WRITE(IOUT,4) ISP
    4 FORMAT(1X,I8,' ELEMENTS IN X ARRAY ARE USED FOR SURFACE ',
     1               'WATER DIVERSIONS.')
      ISUM1=ISUM-1
      WRITE(IOUT,5) ISUM1,LENX
    5 FORMAT(1X,I8,' ELEMENTS OF X ARRAY USED OUT OF',I8)
C
C6------IF THERE ISN'T ENOUGH SPACE IN THE X ARRAY THEN PRINT
C6------A WARNING MESSAGE.
      IF(ISUM1.GT.LENX) WRITE(IOUT,6)
    6 FORMAT(1X,'   ***X ARRAY MUST BE DIMENSIONED LARGER***')
C7------RETURN
      RETURN
      END
```

SRF1RP: define surface water diversions for a stress period

```
      SUBROUTINE SRF1RP(SURF,NSURFS,MXSURF,istrm,mxstrm,IN,IOUT)
* usage:
ccc   IF(IUNIT(14).GT.0) CALL SRF1RP(X(LCsurf),nsurfs,mxsurf,
ccc  1             X(icstrm),mxstrm,IUNIT(14),IOUT)
C
C
C-----VERSION 1544 22DEC1982 SRF1RP
C     ******************************************************************
C     READ NEW SURFACE WATER DIVERSION LOCATIONS AND FLOW RATES
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
*         This is a variation on WEL1RP with an indexing scheme taken from
*         SWB1RP.
      DIMENSION SURF(8,MXSURF),istrm(5,mxstrm)
C     ------------------------------------------------------------------
C
C1------READ ITMP(NUMBER OF SURFS OR FLAG SAYING REUSE SURF DATA)
      READ (IN,1) ITMP
    1 FORMAT(I10)
      IF(ITMP.GE.0) GO TO 50
C
C1A-----IF ITMP LESS THAN ZERO REUSE DATA. PRINT MESSAGE AND RETURN.
      WRITE(IOUT,6)
    6 FORMAT(1H0,'Use surface water diversions',
     1            ' FROM LAST STRESS PERIOD')
      RETURN
C
C1B-----ITMP=>0.  SET NSURFS EQUAL TO ITMP.
   50 NSURFS=ITMP
      IF(NSURFS.LE.MXSURF) GO TO 100
C
C2------NSURFS>MXSURF.  PRINT MESSAGE. STOP.
      WRITE(IOUT,99) NSURFS,MXSURF
   99 FORMAT(1H0,'NSURFS(',I4,') IS GREATER THAN MXSURF(',I4,')')
      STOP
C
C3------PRINT NUMBER OF surface water diversions IN CURRENT STRESS PERIOD.
  100 WRITE (IOUT,2) NSURFS
    2 FORMAT(1H0,10X,I4,' SURFS')
C
C4------IF THERE ARE NO ACTIVE surface water diversions IN THIS STRESS
*            PERIOD THEN RETURN
      IF(NSURFS.EQ.0) GO TO 260
C
C5------READ AND PRINT LAYER,ROW,COLUMN, segment, reach AND flow RATE.
      if (isrfcb.ne.0) WRITE(IOUT,3)
    3 FORMAT(1H ,' LAY  ROW  COL  seg  rch  idx        Q (cfs)   ii')
      qsum=0.
      DO II=1,NSURFS
         READ (IN,4) K,I,J,isegmt,ireach,Q
    4    FORMAT(5I5,F15.0)
         SURF(1,II)=K
         SURF(2,II)=I
         SURF(3,II)=J
         surf(4,ii)=isegmt
         surf(5,ii)=ireach
```

```
          SURF(7,II)=Q
          qsum=qsum+q
*             set surf(jj,ii) to q for both jj=7 and 8; see note above. --spp
          surf(8,ii)=q

*             Find the index to the order in which this (segment,reach)
*             was read from the STREAM (~.str) input file, to allow direct
*             indexing of the (segment,reach) for subbasin outflow.

*             set up a default to the first reach in case no match is found:
          idxrch = 1
          do idx=1,mxstrm
              if (isegmt.eq.istrm(4,idx)) then
                  if (ireach.eq.istrm(5,idx)) then
*                         match was found on (segment,reach) for subbasin outflow:
                      idxrch = idx
                      go to 30
                  else if (istrm(4,idx).eq.1) then
*                         this is better than nothing: first reach of same segment
                      idxrch = idx
                  end if
              end if
          end do
*             the (segment,reach) identified for watershed outflow wasn't
*             found; let the user know.
          print *,'SRF1RP: Mismatch between surface water '//
     1        'diversion pt. and stream segment:'
          print '(5(1x,a,i5))', 'surface diversion',ii,
     1        'outflow to',isegmt,'reach',ireach,'index',idx
30        continue
          surf(6,ii) = idxrch
          if (isrfcb.ne.0) write (iout,'(6i5,f15.3,i5)')
     1        K,I,J,isegmt,ireach,idxrch,Q,ii
      end do
      write (iout,'(1x,a,g15.7)') 'SRF1RP: total: qsum=',qsum
C
C6------RETURN
  260 RETURN
      END
```

SRF1FM: surface water diversions as lateral outflow from streams

```
      SUBROUTINE SRF1FM(NSURFS,MXSURF,mxstrm,SURF,STRM,IBOUND,
     1        NCOL,NROW,NLAY,iout)
* usage:
cccc  IF(IUNIT(14).GT.0) CALL SRF1FM(nsurfs,mxsurf,mxstrm,x(LCsurf),
cccc 1     X(LCSTRM),x(LCIBOU),ncol,nrow,NLAY,IOUT)
C
C-----VERSION 1233 12MAY1987 SRF1FM
C
C     ******************************************************************
C     SUBTRACT Q FROM RHS
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
      DIMENSION SURF(8,MXSURF),STRM(20,mxstrm),
     1            IBOUND(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
*         initialize lateral inflow for each reach to zero.
*         Note: this won't be done in SWB1FM if iunit(14) > 0.
      do i=1,mxstrm
          strm(18,i)=0.
      end do
C1------IF NUMBER OF surface water diversion <= 0 THEN RETURN.
      IF(NSURFS.LE.0) RETURN
C
C2------PROCESS EACH surface water diversion IN THE SURF LIST.
      DO 100 L=1,NSURFS
      IR=SURF(2,L)
      IC=SURF(3,L)
      IL=SURF(1,L)
cccc  isegmt = surf(4,L)
cccc  ireach = surf(5,L)
```

```
      idxrch = surf(6,L)
      Q=SURF(7,L)
C
C2A-----IF THE CELL IS INACTIVE THEN BYPASS PROCESSING.
      IF(IBOUND(IC,IR,IL).LE.0) GO TO 100
C
C2B-----IF THE CELL IS VARIABLE HEAD THEN SUBTRACT Q FROM
C         THE stream reach lateral inflow:
      strm(18,idxrch) = strm(18,idxrch) + q
ccccc RHS(IC,IR,IL)=RHS(IC,IR,IL)-Q
  100 CONTINUE
C
C3------RETURN
      RETURN
      END
```

Appendix E: Potential evaporation (subroutine PENMAN)

Wind speed measurement height

An option has been added to SWAT (**iopet**, in options input file ~.cod) to calculate potential evaporation over land according to the Penman equation 4.2.31 recommended by Shuttleworth (1993) and implemented by subr PENMAN, listed below.

Penman calculations are based on wind speed at 2 m above land surface (ALS), denoted u_2, which is estimated as follows, using eq. 4.4.9 in Shuttleworth (1993). Wind speed measured at 10 m, denoted u_{10}, is reduced by a factor f given by $f = 34.9648/(f_h \cdot f_w)$, where $f_h = \ln[(z_h - 0.08)/0.001476]$, $f_w = \ln[(z_w - 0.08)/0.01476]$; and z_h and z_w are heights (ALS) of humidity and wind speed measurements, respectively. For $z_h = 2$ m and $z_w = 10$ m, wind speed measured at 10 m is reduced by a factor $f = 0.749$; and wind speed at 2 m ALS is estimated to be given by $u_2 = f \cdot u_{10}$.

Source code

```
* file penman.for   spp aug 94  compare methods of calculating evapotranspiration

      subroutine PENMAN (itrace,method,iemiss,julday,albedo,elev,patm,
     1    phi,tmin,tmax,tavg,dtsoil,stj,u2,relhum,pcpmm,hc,dxleaf,s0j,
     1    vpdef,rs,ra,snj,rlong,shflux,etref)

ccc  ,dltchg,eschg)
*
* Compute evaporation estimates (mm/day); numbered equations refer to
* Handbook of Hydrology, 1993, Maidment, ed., especially Ch. 4, Evaporation,
* Shuttleworth; see also sec. 13.2, pp. 13.22-13.25.  This routine is designed
* to aid comparison of various methods of calculating evapotranspiration.
*     Note: relative humidity value is assumed to be available.
*
* argument list
*     INPUT:
* int itrace option (no=0, yes otherwise) to print most calculated values
*              just prior to returning (see end of routine).
* re  patm   atmospheric pressure (kPa)
* re  phi    site latitude (radians)
* int julday Julian day of year
* re  tmin   day's minimum temperature (deg C)
* re  tmax   day's maximum temperature (deg C)
* re  tavg   day's avg temperature (deg C)
* re  dtsoil change from previous day in soil temperature (deg C) at 0.1 m depth
* re  stj    incoming solar radiation (insolation) at land surface (MJ/day)
* re  u2     day's avg wind speed (m/s)  NOTE: set minimum at 1 m/s (arbitrary)
* re  relhum relative humidity (pct)
* re  hc     crop height (m): used to evaluate actual et by (4.2.27): see et (below).
* re  dxleaf leaf area index; if input as zero, calculated by 4.2.23 for grass.
*
*     OUTPUT: intermediate calculations
* re  cfcref coefficient replacing the value 1.26 in (4.2.38) req'd for eprta
*               to equal epmref (above).
* re  s0j     day's extraterrestrial incoming radiation (MJ/m^2/day) through a
*               plane tangent to a sphere enclosing the earth's atmosphere.
*
*     OUTPUT: evaporation results
* re  epmref reference crop evaporation (4.2.31 or 4.4.14), version of the
*(method 1)     Penman-Montheith equation recommended by the Handbook of Hydrology
* re  epot    potential open water evaporation (4.2.30)
*(method 2)
* re  eprta reference crop evaporation (4.2.38-39), Priestley-Taylor eqn.,
*(method 3)  radiation based; use cfrad=1.74 for arid locations (def. avg
*             relative humidity < 60% for month of peak evaporation);
*             otherwise, cfrad=1.26.  Here, cfrad=1.26 is assumed.
* re  ekimb reference crop evaporation (4.2.33), Penman combination eqn;
*(method 4)
* re  epnmon actual evapotranspiration, the Penman-Monteith eqn (4.2.27);
*(method 5)     see also p. 13.25.
*                   This model accounts directly for the physical mechanisms of
*               turbulent and molecular diffusion of water vapor as follows.
*                   For evaporation, i.e. transfer of water vapor from plant and
*               soil surfaces, turbulent diffusion dominates, and is represented
*               by aerodynamic resistance ra (4.2.25), a function of wind speed
*               and crop height.
*                   For transpiration, i.e. transfer of water vapor from leaf
*               stomata to plant surface, molecular diffusion dominates, but is
*               controlled by the plant's stomatal response to soil moisture and
*               other factors.  This controlled transfer is represented by
*               surface resistance rs, for which some empirical relationships
*               are available, e.g. (4.2.22), which depends on crop height.
*                   When plants are wet, this surface resistance is effectively
*               zero, since the water vapor source is on the plant surface rather
*               than inside the leaves (see sec. 4.2.5).  But evaporation
*               during precipitation (and irrigation) can be represented by
*               interception (and depression storage) losses, which can be
*               estimated separately.
*                   This is a one-step method to calculate actual evapotranspiration,
*               in contrast to all of the preceding methods, which estimate only
*               reference evaporation, which must be multiplied by empirically
*               based restrictive factors for soil moisture and vegetation to
*               estimate actual evapotranspiration, and hence are referred to as
*               two-step methods (sec. 4.4.8, Handbook of Hydrology).
*
* Checks and notes on internal variables:
* ra          aerodynamic resistance (4.2.25; see also eqn 13.2.1) checks
*             for wind speed u2=5 m/s:
*                 crop      mean height hc (m)    ra (s/m)
*                 grass           0.1             45
*                 agricultural    1.              18
*                 forest          10.              6.5
* daymax      max. possible daylight hours (N, eqn 4.4.1) accuracy is within 0.1 h.
*
* s0m         extraterrestrial solar radiation S0 in MJ/m^2/day was backed out
*             from (4.4.4) by multiplying by lambda (i.e. htvap): calculated
*             to compare with s0j above.  Note that (4.4.4) must assume a
*             nominal value for heat of vaporization, so that s0j (above)
*             should be more rigorous than s0m.  A comparison of these two
*             for 1993 Sandyland data (actually it's independent of these data)
*             showed a mean error 100*(s0m-s0j)/s0j of -1.2 pct ranging from
*             -2.6 pct to 0.6 pct (small compared to other error sources
*             in evaporation calculation). S0 (mm/day) by (4.4.4) is accurate
*             to within 0.1 mm.
*                 Checks from Maidment (p.4.31) for April 15 (Julian day=105):
*                 latitude (deg)    N (daymax, h)    S0 (mm/day)
*                     30 N              12.7            15.0
*                      0                12.0            15.1
*                     30 S              11.3            11.2
*-----------------------------------------------------------------------------
      dimension etref(5)
      character*60 refnam(5)
      data (refnam(j),j=1,5) /
     1    '(4.2.31) Penman-Monteith reference crop evap',
     1    '(4.2.30) Penman-Monteith potential open water evap',
     1    '(4.2.39) Priestley-Taylor reference crop evap',
     1    '(4.2.33) Kimberly Penman reference crop evap',
     1    '(4.2.27) Penman-Monteith'/
*         coefficients for estimating incoming solar radiation at given
*         extraterrestrial solar radiation s0 and fraction of cloud cover n/N:
      data as, bs /0.25, 0.50/
*         est. albedo (reflectance): (now pass as argument)
cccc  data albedo /0.23/
*         Stefan-Boltzmann constant (MJ/m^2/K^4) for long-wave radiation:
      data stefan /4.903e-9/
*         solar constant, langleys/min:
      data gsc /1.957/
*         convert radiation in langleys (cal/cm^2) to MJ/m^2:
*             (note: based on 1 J = 0.239 cal)
      data radcnv /23.9/
*         soil heat capacity (MJ/(m^3*C)) for average moist soil (Handbook,p.4.10):
      data cpsoil /2.1/
*         presumed soil depth (m) for temperature change dTsoil:       (4.2.17)
      data dpsoil /0.15/
*             specific heat of moist air (kJ/(kg*C)) (p. 4.13):
      data cp /1.013/
*             mixing ratio (check term): ratio of molecular weight of water
*             vapor to that for dry air:
      data epsiln /0.622/
*         coefficient for psychrometric constant gamma: pscoef=1.e-3*cp/epsiln
      data pscoef /1.6286e-3/
      data iopchk /0/ !option to write out table with entries like T. 4.4.3
      data initlz /0/
*-------------------------------
*         saturation vapor pressure (kPa) given temperature (deg C):    (4.2.2)
*         (approximation for Clausius-Clapeyron eqn (3.3.3))
      satvp(T) = 0.6108*EXP(17.27*T/(237.3+T))
*             evaluate derivative delta = d(es)/dT at T (kPa/degC):      (4.2.3)
      desdt(es,T) = 4098.*es/(237.3+T)**2
*         estimated density (kg/m^3) of moist air (eqn 4.2.4, Shuttleworth,
*         Ch. 4, Evaporation, Handbook of Hydrology, ed. by Maidment '93), for
*         atmospheric pressure P (kPa) and air temperature T (deg C):
      airden(P,T) = 3.486*P/(275.+T)
*         estimated atmospheric pressure P (kPa) as a function of elevation z (m)
*         leaf area index for clipped grass of height 0.05 < hc < 0.15 m:
      dxgras(hc) = 24.*hc
*         leaf area index for alfalfa of height 0.10 < hc < 0.50 m:
      dxalfa(hc) = 5.5 + 1.5*ALOG(hc)
*         program petsnd.bas (KSU) vapor pressure calculation:
      vpksu(RH,T) = 6.11*(RH/100.)*10.**(7.5*T/(T+237.3))
*         nominal atm. pressure at site (kPa): z=elev (m)              (4.4.12)
      pnom(z) = 101.3*((293.-0.0065*z)/293.)**5.256
*         (The above function is used in the calling routine ETSAND)
*         pressure conversions:
*         1 atm = 101.32 kPa (kPa = kilopascal; 1 pascal = 1 N/m^2)
*               = 1013.2 millibars = 760 mm Hg = 10.33 m water
*               = 14.7 psia      = 29.92 in Hg = 33.9 ft water
*
      if (initlz.eq.0) then
          initlz = 1
          pi = 4.*ATAN(1.)
          twopi = 2.*pi
          gscday = gsc*24.*60.
*             solar constant (MJ/m^2/day):
          gscj = gsc*24.*60./radcnv
          govrpi=gscj/pi
*             coef. used in (4.2.18) for daily change in soil heat content:
          cpdp = cpsoil*dpsoil
          print 110,'solar constant (MJ/m^2/day) gscj=',gscj,
     1        'gscj/pi=',govrpi
          print 110,'soil heat flux coef cp*dp in (4.2.17) =',cpdp
      end if
*         choose 1st method if not specified (1-5):
      if (method.lt.1.or.method.gt.5) method=1
*
*             nominal atmospheric pressure (kPa):
      patm = PNOM(elev)
*             mean daily air temperature (deg K):
      tavgk = tavg + 273.2
*             julday = Julian day
      djulia = julday
      yrfrac = djulia/365.
```

```
*         dr = earth-sun distance as fraction of (mean?)            (4.4.5)
      dr = 1.+0.033*COS(twopi*yrfrac)
*             solar declination (radians)                            (4.4.3)
      dlt = 0.4093*SIN(twopi*yrfrac - 1.405)
*             sunset hour angle (radians):                           (4.4.2)
      ws = ACOS(-TAN(phi)*TAN(dlt))
*             max. possible daylight hours (N):                      (4.4.1)
      daymax = 24.*(ws/pi)
*
*             extraterrestrial solar radiation S0 (MJ/m^2/day):      (ce751)
      s0j = (ws*SIN(phi)*SIN(dlt) + COS(phi)*COS(dlt)*SIN(ws))*
     1      dr*gscday/(pi*radcnv)
*
*             htvap = latent heat of vaporization (MJ/kg) (lambda):   (4.2.1)
*                     base this on mean daily temperature tavg (deg C),
*                     but is defined in terms of Ts, water surface temperature.
      htvap = 2.501 - 0.002361*tavg
*             psychrometric constant (kPa/deg C):                     (4.2.28)
      gamma = pscoef*patm/htvap
*         day's avg saturated water vapor pressure (kPa) (see p. 4.33):
      esmin = SATVP(tmin)
      esmax = SATVP(tmax)
      esavg = (esmin + esmax)/2.
*             estimate avg vapor pressure deficit (kPa):              (4.4.6)
      vpdef = esavg*(1.-relhum/100.)
*             sat. water vapor pressure (kPa) at day's avg temp (deg C):
      es = SATVP(tavg)
cccc  eschg = esavg/es
ccccc desdt(es,T) = 4098.*es/(237.3+T)**2    !(copy of statement function above)
*             evaluate avg derivative delta = d(es)/dT at T (kPa/degC)  (4.2.3)
      delta = (DESDT(esmin,Tmin) + DESDT(esmax,Tmax))/2.
cccc  dltold= 4098.*es/(237.3+tavg)**2
cccc  dltchg = delta/dltold
*
*             To convert solar radiation from MJ/m^2/day to mm/day, divide by
*             htvap, the latent heat of vaporization (MJ/kg), (4.2.1).
*                 Explanation (p. 4.8): for net radiation Rn (MJ/m^2/day), the
*             equivalent depth of evaporated water is Rn/(rho_w*lambda) (m),
*             where rho_w= density of water (approx. 1000 kg/m^3) and
*                   lambda=latent heat of vaporization (htvap, above, MJ/kg).
*             so equivalent evaporated depth (mm) is (1000/rho_w)*Rn/lambda,
*             or approximately Rn/lambda.
*             extraterrestrial solar radiation s0m in MJ/m^2/day backed out
*             from (4.4.4) by multiplying by lambda (i.e. htvap); see
*             notes on s0m in argument list description.
cccc  s0m = htvap*15.392*dr*
cccc 1          (ws*SIN(phi)*SIN(dlt) + COS(phi)*COS(dlt)*SIN(ws))

*             absorbed short-wave radiation:                          (4.2.5)
      snj = (1.-albedo)*stj

*         intermediate calculations for net long-wave radiation:
*             est. fraction of available extraterrestrial solar radiation:
      radfrc = stj/s0j
*             estimate fraction of cloud cover dayfrc (n/N) based on st and s0:
*             rearrange equation used to estimate st = s0*(as + bs*n/N) (4.2.6)
*             into the form  [(st/s0) - as]/bs = n/N; nominal values for
*             coefficients as, bs are in data statement.  Limit this fraction
*             to the range [0,1].
      dayfrc = AMIN1(1., AMAX1(0., (radfrc - as)/bs ))
*             cloudiness factor f using nominal coefficients:          (4.2.12)
      f = 0.9*dayfrc + 0.1
*         use the alternate calculation (4.2.9) if relative humidity isn't
*         available; e.g. for Swat, temperature comes from data, but relative
*         humidity was simulated.
      if (iemiss.eq.0) then
*             unsaturated vapor pressure: from def. RH = e(T)/es(T):
         eunsat= esavg - vpdef
*             est. net emissivity from vapor pressure (kPa):          (4.2.8)
         emiss = 0.34 - 0.14*SQRT(eunsat)
      else
*            alt: est. net emissivity from avg temperature (deg C):   (4.2.9)
         emiss = -0.02 + 0.261*EXP(-7.77e-4*tavg*tavg)
      end if
*             net long-wave radiation (MJ/m^2/day):                   (4.2.7)
```

```
      rlong = -f*emiss*stefan*tavgk**4

*             est. soil heat flux (MJ/m^2/day):                       (4.2.18)
      shflux = cpdp*dtsoil
*         energy available for evaporation (mm/day) = net radiation - soil heat flux
*             net radiation (MJ/m^2/day) = snj + rlong;               (4.2.13)
      avail = (snj + rlong - shflux)/htvap
*             available energy coefficient for potential evaporation from
*             open water (4.2.30), for the Priestley-Taylor eqn (4.2.38-39),
*             and the Kimberly Penman combination equation (4.2.33):
      f1p = delta/(delta+gamma)

      if (method.eq.1 .or. iopchk.gt.0 .or. itrace.gt.0) then
*             evaluate reference crop evaporation:                    (4.2.31)
*             coef's f1rc and f2rc are both functions of (T,u2,z).
*             variation on psychrometric constant gamma (above):      (4.2.32)
         gstar = gamma*(1.+0.33*u2)
*             available energy coef.:                                 (4.2.15)
         f1rc = delta/(delta+gstar)
*             vapor pressure deficit coef.:                           (4.2.16)
         f2rc = (gamma/(delta+gstar))*900.*u2/(tavg+275.)
*             est. reference crop evaporation (mm/day):               (4.4.14)
         etref(1) = f1rc*avail + f2rc*vpdef
      end if
*             est. evaporation rate from open water: potential evap by
*             Penman eqn (4.2.30), rewritten as (4.4.10):
      if (method.eq.2 .or. iopchk.gt.0 .or. itrace.gt.0) then
*             energy available for evaporation from open water (mm/day),
*             assuming that data to estimate advected energy Ah (4.2.20) is
*             not available (see flowchart, Table 4.4.1, item 5b);
*             Note that albedo for open water from T. 4.2.2 is used here:
         data albh2o /0.08/
         avlopn = (stj*(1.-albh2o) + rlong)/htvap
         f2p = gamma/(delta+gamma)*6.43*(1.+0.536*u2)/htvap
(4.4.13)
         etref(2) = f1p*avlopn + f2p*vpdef                            (4.4.10)
      end if
*
*        always calculate this for now to pass back comparison w/ method 1.
cccc  if (method.eq.3 .or. iopchk.gt.0 .or. itrace.gt.0) then
*       compute coef. for comparing with radiation-based eqn 4.2.38-39:
*       use cfrad=1.74 for arid locations (def. avg relative humidity < 60%
*       for month of peak evaporation); cfrad=1.26 otherwise.  Here,
*       cfrad=1.26 is assumed.
*             Priestley-Taylor eqn (radiation-based evap):
         etref(3) = 1.26*f1p*avail                                    (4.2.39)
*             compute coef. req'd for radiation-based eqn (4.2.38-39) to match
*             value given by etref(1) from (4.2.31):
         cfcref = etref(1)*(1.+gamma/delta)/avail
cccc  end if
*
      if (method.eq.4 .or. iopchk.gt.0 .or. itrace.gt.0) then
*             est. ref. crop evap based on the Kimberly Penman combination
*             eqn (4.2.33), so named for using a seasonal adjustment based on
*             Kimberly, Idaho data.
*                 wind ftn for the Kimberly Penman combination eqn:
*             coef. aw for wind function (below):                     (4.2.35)
         awexp = ((djulia-173.)/58.)**2
         aw = 0.4 + 1.4*EXP(-awexp)
*             coef. bw for wind function (below) for Northern latitudes:(4.2.36)
*             (for Southern latitudes see variation given in Handbook)
         bwexp = ((djulia-243.)/80.)**2
         bw = 0.605 + 0.345*EXP(-bwexp)
*             wind function for Kimberly Penman eqn:                  (4.2.34)
         wndftn = aw + bw*u2
         f2k = gamma/(delta+gamma)*6.43*wndftn/htvap                  (4.4.13)
         etref(4) = f1p*avail + f2k*vpdef                             (4.2.33)
      end if
*
*             resistance-based model, the Penman-Monteith eqn(4.2.27)
      if (method.eq.5 .or. iopchk.gt.0 .or. itrace.gt.0) then
*             if leaf area index is not given (i.e. input as zero),
*             use 4.2.23 to estimate leaf area index for grass:
         if (dxleaf.eq.0.) dxleaf = DXGRAS(hc)
*             est. fraction of day that canopy is wet; use to reduce surface
*             resistance.
          frcwet = AMIN1(1.,0.045*pcpmm)
*             approximate surface resistance:                            (4.2.22)
          rs = frcwet*200./dxleaf
*             von Karman constant:
          data vonkar /0.41/
*             assumed heights (m) of wind speed and humidity measurements:
          data zu, ze /2., 2./
*             intermediate values for aerodynamic resistance (below)
*             for a mean crop height hc (m):
*             roughness length (c.f. (13.2.2)):
          d = 0.67*hc
*             zero plane displacement for wind and humidity factors (c.f. (13.2.2):
          zom = 0.123*hc
          zov = 0.0123*hc
          wfactr = ALOG((zu-d)/zom)
          hfactr = ALOG((ze-d)/zov)
*             assume wind speed uz(zu) (m/s) measurement height zu = 2 m:
          uz = u2
*             aerodynamic resistance:                                    (4.2.25)
*             (see also eqn (13.2.1))
          ra = wfactr*hfactr/(vonkar*vonkar*uz)
*             compare eqn given for the reference crop (for std u2)      (4.2.26)
          rasc = 208./u2
*             air density:                                               (4.2.4)
          rhoair = AIRDEN(patm,tavg)
*             resistance-based model for evap: the Penman-Monteith eqn  (4.2.27)
          etref(5) = ((delta*avail + rhoair*cp*vpdef/ra)/
     1        (delta+gamma*(1.+rs/ra)))/htvap
      end if
*
*             try to duplicate KSU calculations of program PETSND.BAS, from
*             which the following equations were taken.  Since 8am air
*             temperature is not included in the Sandyland output file,
*             estimate t8am as weighted average of min and max.
c     tfavg = 32.+(9./5.)*tavg
*             est. temp. at 8am based on daily min and max temp's:
c     t8am = 0.75*tmin + 0.25*tmax
c     tf8am = 32.+(9./5.)*t8am
c     esksu = VPKSU(100.,tfavg)
c     e8am  = VPKSU(relhum,tf8am)
*             conversion factor from m/s to mi/day:
cccccc    cnvmpd = 24.*(3600./5280.)/0.3048
*             wind run (mi/day):
c     u2mpd = cnvmpd*u2
c     etksu = (3.1*(0.837*st-2.2) + (0.262*(1.+0.0061*u2mpd))*
c    1        (esksu-e8am))/4.1
c     etksu = AMAX1(0.,etksu)
*
      data inithd/0/
      if (iopchk.gt.0.or.itrace.gt.0) then
          if (inithd.eq.0) then
          inithd=1
              print '(1x,a)', 'elev,m P,kPa Tmp,C u2,m/s'//
     1        '    F1p     F2p     F1rc     F2rc     F2k'//
     1        '   delta   gamma   htvap   esavg'
          end if
          print '(1x,f6.0,3f6.1,1x,10f8.3)',
     1        elev,patm,tavg,u2,f1p,f2p,f1rc,f2rc,f2k,
     1        delta,gamma,htvap,esavg
      end if
      if (itrace.gt.0) then
          do j=1,5
              print '(i3,f8.2,1x,a)', j,etref(j),refnam(j)
          end do
          print 110,'cfcref=',cfcref,'(Priestley-Taylor coef reqd'//
     1        ' to match Penman ref et)'
          print 110,'es=',es
          print 110,'s0j=',s0j,'MJ/m^2/d, daymax=',daymax
          print 110,'vpdef=',vpdef,'stj=',stj,'MJ/m^2/d'
          print 110,'stj/s0j=',radfrc,'dayfrc=',dayfrc,
     1            'cloudiness factor f=',f,'albedo=',albedo
          print 110,'emiss(e)=',emiss,'iemiss=',iemiss
          print 110,'snj=',snj,'MJ/m^2/d, rlong=',rlong,'MJ/m^2/d'
          print 110,'shflux=',shflux,'MJ/m^2/d, avail=',avail,'mm/d'
```

```
110       format (5(1x,a,f10.3))
          print *,'Terms for resistance-based evaporation model:'
          print 110,'uz=',uz,'hc=',hc
          print 110,'d=',d,'zom=',zom,'zov=',zov
          print 110,'wfactr=',wfactr,'hfactr=',hfactr
          print 110,'rs=',rs,'ra=',ra,'rasc=',rasc
          print 110,'rhoair=',rhoair
          print 110,'frcwet=',frcwet
      end if
      return
      end
```

Appendix F. MODFLOW postprocessing programs

Programs POSTMOD and MODHYD were written to extract results of interest from MODFLOW's standard output for further data analysis and display. POSTMOD extracts heads or drawdowns for specified time steps of interest. MODHYD extracts a time series of results at an aquifer node (heads or drawdowns) or a stream node (streamflow, streambed leakage, or stream stage); or a time series of results along a row, column, or stream as described below.

POSTMOD and MODHYD are both based on FORTRAN's intrinsic INDEX character function, used to search for character strings that are part of standard MODFLOW output and serve as guideposts to locate the desired data. The MODFLOW manual (McDonald & Harbaugh, 1988) provides details of output format. Source code excerpts from MODFLOW were adapted for use in POSTMOD and MODHYD by changing WRITE statements into READ statements.

POSTMOD with example to extract drawdown arrays

Program POSTMOD (executable file postmod.exe) reads a standard MODFLOW output file to copy arrays of heads or drawdowns (or others listed below if written by MODFLOW) to separate files for specified time steps. Heads or drawdowns may be extracted (i.e. copied to a separate file) for any given time step and stress period for which the heads or drawdowns of interest were printed. Other arrays listed below are not printed in MODFLOW's standard output except through use of nonstandard options added to MODFLOW.

The file extension (shown as "ext" below) depends on the file format chosen, with the following options. First, the format may be that of a spreadsheet (ext = "srf"), in which each array element is described by a file record specifying grid coordinates and array element value (row i, column j, a_{ij}). Second, the format may be a wrapped matrix format (ext = "dat") as defined in the Utilities chapter of MODFLOW's manual (McDonald et al., 1988). File name prefixes correspond to the type of extracted array as follows:

HEADn.ext - heads from stress period n;
DRAWn.ext - drawdowns from stress period n.
THCKn.ext - saturated thickness, stress period n;
VSTGn.ext - volumetric storage, stress period n.
DSDTn.ext - rate of change of storage, stress period n.
CFLOn.ext - constant head cell flow rates, stress period n.
FFLOn.ext - front face cell flow rates, stress period n.
RFLOn.ext - right face cell flow rates, stress period n.
LFLOn.ext - lower face cell flow rates, stress period n.

As shown in an example below, the program prompts the user to specify the stress period and layer, whether an array of heads, drawdowns or other array is to be extracted, and the format in which the array is written, according to the MODFLOW output control codes. POSTMOD is normally able to find these control codes from MODFLOW's output; if not, POSTMOD asks for the format.

Example: use POSTMOD to extract drawdown arrays.

Run POSTMOD at the DOS prompt with redirected input from file postmod.rsp and redirected output to file postmod.jou as follows:

```
   postmod <postmod.rsp >postmod.jou
```

Redirected input file postmod.rsp is as follows:

```
corn90cp.prn
c              continue or exit
y              subst. value for dry nodes ok?
1,1            enter time step, stress period of interest
h              extract heads or drawdowns?
s              [S]urfer, [A]rc-Info, or [M]atrix (wrapped) format?
y              exclude zero-valued array elements?
y              use unit cell dimensions?
n              extract another array?
```

POSTMOD redirected output file postmod.jou with queries and responses (in **bold**):

```
Program POSTMOD
Extract head or drawdown arrays from MODFLOW output file.

 Enter MODFLOW output file name [MODFLOW.PRN]: corn90cp.prn
0Pre-dev transient '60-'84 DWR Rattlesnake Creek 1/12/96 w/ Swat: 6061swat.bas
  1 layer(s),  47 rows, 190 columns,   2 stress periods,
Head format code    2 = (5x,9g14.6)
Drawdown format code   2 = (5x,9g14.6)
Note: Cell-by-cell flow format code not found.
Scan for dry nodes; write list to file corn90cp.DRY
   14 instances of dry nodes are listed on file corn90cp.DRY
C[ontinue POSTMOD] or E[xit]?[C]: C
Default Substitution value =  0.0000000     for dry nodes.
Is substitute dry node value ok?[Y]: Y

           2 stress periods.
Enter time step, stress period (negative no. to STOP):    1    1
Find STRESS PERIOD  1
Options:
HEADn [H]eads
DRAWn [D]rawdowns
Enter choice: [H]: H
nput format = (5x,9g14.6)
Output format: [S]urfer input, [A]rc/info input, [M]atrix[S]: S
Exclude zero values from output?[Y]: Y
Use unit cell dimensions (row,col widths)?[Y]: Y
  layer timstp strper period:
     1     1     1     1

HEAD IN LAYER                  1 AT END OF TIME STEP  1 IN STRESS PERIOD  1
write to file HEAD1.srf
Use format (5x,9g14.6)
Write file HEAD1.srf

Extract row:
   1   2   3   4   5   6   7   8   9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
Array written to file HEAD1.srf
Extract another array? [Y]: N
```

Excerpt from results on file head1.srf, giving (row, column, head) for each record:

```
     7.500        46.50        2365.030
     8.500        46.50        2364.720
     9.500        46.50        2364.190
     10.50        46.50        2363.520
     11.50        46.50        2362.760
     12.50        46.50        2362.000
     13.50        46.50        2361.310
     14.50        46.50        2360.700
     15.50        46.50        2360.240
     52.50        46.50        2235.240
```

MODHYD with example to extract streamflow time series

Program MODHYD (executable file modhyd95.exe) reads a standard MODFLOW output file and extracts (i.e., copies to a separate file) a time series of results for either individual nodes or for a profile of a row, column or stream. Each element in the time series corresponds to printed results for a time step in the MODFLOW output.

For an aquifer MODHYD will extract either heads or drawdowns, and profiles may be extracted along either a row or a column.

For a stream MODHYD will extract stage, depth, leakage or aquifer head at a stream node. If stream profiles are specified then all the above stream results will be extracted.

If time series for individual nodes are specified, then up to 20 such series may be extracted in succession. They will be written to a file after all have been extracted, such that columns in the file correspond to nodes, and rows correspond to time steps. Optional columns at the left hand side label time step and stress periods.

If profiles (for aquifer rows or columns, or for stream reaches) are specified, then the file is written as follows:

```
do for each time step:
   do for each node along the row, column or stream:
      write time, location and corresponding results;
   end do
   write a blank line to separate profiles for successive time steps (*);
end do.
```

(*) The files are written to be imported into a spreadsheet for analysis and plotting. In the case of Quattro Pro™, the blank line between time steps produces the equivalent of a raised pen directive between coordinate pairs so that successive profiles may be superimposed on a plot.

Example: use MODHYD95 to extract streamflow time series

Stream reaches are designated by (row,column) grid coordinates.

Run modhyd95 at the DOS prompt with redirected input from file modhyd95.rsp and redirected output to file modhyd95.jou as follows:

```
   modhyd95 <modhyd95.rsp >modhyd95.jou
```

Redirected input file modhyd95.rsp is as follows:

```
corn90cp.prn                   MODFLOW output file name
y                              use unit cell dimensions?
s                              extract heads, drawdowns, or stream
o                              write [S]:streambed leakage or [O] streamflow?
corn90cp.hot                   enter the hydrograph output file name
p                              extract row, column, stream or point values?
3                              enter no. series (max=20, min=1)
2,25                           enter (row, col) for time series 1
3,36                                                     series 2
4,45                                                     series 3
y                              include time step, stress period columns on left?
n                              run again?
```

MODHYD95 redirected output file modhyd95.jou with queries and responses (in **bold**):

```
Program MODHYD
Extract head or drawdown arrays from MODFLOW output file.

NOTE: Arrays must be WRAPPED (rather than STRIPPED).
 (See MODFLOW manual, Ch. 14, "Utility Modules")

 Enter MODFLOW output file name [MODFLOW.PRN]: corn90cp.prn
0Pre-dev transient '60-'84 DWR Rattlesnake Creek 1/12/96 w/ Swat: 6061swat.bas
  1 layer(s),  47 rows, 190 columns,   2 stress periods,
Head format code =   2 Drawdown format code =   2
Use unit cell dimensions (row,col widths)?[Y]: Y
 Extract H[eads] D[rawdowns] S[tream] or Q[uit]: [H]: S
   775 STREAM NODES  nstrem=    775
Write S[tream flow into aquifer] or O[utflow into reach]? [S]: O
outfmt = (2(1x,i3),1x,11g11.3)
fmthdg = (1x,a6,t10,11(2x,i5,4x))
Enter the hydrograph output file name corn90cp.hot
Extract R[ow], C[olumn] S[tream] or P[oint values]?[P]: P
Enter no. series (max=20): 3

Enter (row,col) for time series  1:    2   25
Enter (row,col) for time series  2:    3   36
Enter (row,col) for time series  3:    4   45
Extract   3 hydrographs:
   2   25STRM
   3   36STRM
   4   45STRM
 layer  row  col  str  rchinto aquifer  outflow
    8 ITERATIONS FOR TIME STEP   1 IN STRESS PERIOD  1 WITH SIP SOLVER
    8 ITERATIONS FOR TIME STEP   2 IN STRESS PERIOD  1 WITH SIP SOLVER
    7 ITERATIONS FOR TIME STEP   3 IN STRESS PERIOD  1 WITH SIP SOLVER
   13 ITERATIONS FOR TIME STEP   4 IN STRESS PERIOD  1 WITH SIP SOLVER
end of file: stream reach index L=    0 time step    4    4    1
Extracted    4 series (mxser=   20 ) from corn90cp.prn
Include time step, stress period columns on left? (y/n)[Y]: Y
Write hydrograph output file corn90cp.hot
Again?[Y]: N
```

Output file corn90cp.hot specified by response file modhyd95.rsp is as follows:

```
----------------------------------------------------------------------
0Pre-dev transient '60-'84 DWR Rattlesnake Creek 1/12/96 w/ Swat: 6061swat.bas

               1          2          3
               2          3          4
              25         36         45
   1    1  0.000      0.000      0.000
   1    2  0.000      0.000      0.000
   1    3  0.000      0.000      0.000
   1    4  0.000      0.000      0.000
```