

**A Simple Lumped Parameter Model
for
Unsaturated Zone Processes**

John Doherty
Watermark Numerical Computing
August, 2020

Table of Contents

1. Introduction	1
1.1 General.....	1
1.2 Installation.....	1
1.3 Source Code.....	1
1.4 Examples	1
2. Concepts	2
2.1 General.....	2
2.2 Volume of the Soil Moisture Store	3
2.3 Precipitation	3
2.4 Evapotranspiration	3
2.5 Recharge.....	4
2.6 Macropore Recharge	5
2.7 Irrigation.....	6
2.8 Runoff.....	6
2.9 Volume to Elevation.....	6
3. Solution Method.....	8
4. Using LUMPREM.....	9
4.1 Subroutine RECHMOD	9
4.2 Program LUMPREM	11
4.2.1 Running LUMPREM	11
4.2.2 Inputs.....	11
4.2.3 Outputs.....	14
5. LR2SERIES.....	16
5.1 General.....	16
5.2 LR2SERIES Input Control File	16
5.2.1 An Example	16
5.2.2 Specifications	16
5.2.3 READ_LUMPREM_OUTPUT_FILE	17
5.2.4 WRITE_MF6_TIME_SERIES_FILE	17
6. LUMPREP.....	19
6.1 General.....	19
6.2 The LUMPREP Input File.....	19
6.3 LUMPREP Keywords - General.....	21
6.4 LUMPREP Keywords – Details.....	22
6.4.1 Keywords that are Supplied Only Once	22
6.4.2 Model-Specific Keywords	23
6.4.3 Writing a Partial PEST Control File	27
7. References	29

1. Introduction

1.1 General

This document describes a simple lumped parameter model named LUMPREM which simulates water movement in the unsaturated zone. It also describes some utility programs which expedite LUMPREM setup, and provide an interface to groundwater models. The number of these LUMPREM-support programs is expected to grow over time.

LUMPREM was built to complement a groundwater model, receiving daily rainfall as its input and supplying groundwater recharge as its output. Recharge is accumulated over user-defined lengths of time which may correspond, if desired, to groundwater model stress periods. If requested by the user, the demand and application of irrigation water can also be simulated; the fraction of this demand that is met by groundwater is accumulated and reported.

LUMPREM provides basic simulation of water balance within the unsaturated zone, with inputs and outputs that include rainfall, irrigation, evapotranspiration, runoff, recharge and macropore recharge. All calculations pertaining to water movement and storage within the unsaturated zone take place within a subroutine named RECHMOD. This subroutine can be called by programs other than LUMPREM if desired. LUMPREM reads data required by RECHMOD, undertakes basic manipulation of these data, calls RECHMOD to calculate water balance components, and writes RECHMOD-calculated results to its output file.

1.2 Installation

Unzip the LUMPREM zip file in a folder of your choice. Add the name of the installation folder to the PATH environment variable. Alternatively, copy the executable programs in the LUMPREM installation folder to a folder whose name is already cited in the PATH environment variable.

1.3 Source Code

Both source code and executable versions of programs described in this document are provided. Executable programs are 64 bit. They were compiled using the Intel compiler. See *compile.bat* in the LUMPREM installation folder.

1.4 Examples

See the *examples* subfolder of the main LUMPREM installation folder. To run LUMPREM, type the following command from a command-line window open to this folder:

```
..\lumprem lumprem_test.in lumprem_test.out
```

To run LR2SERIES, type the following command, and then answer its prompt as follows:

```
..\lr2series
Enter name of LR2SERIES control file: lr2series.in
```

A set of files that exemplify the running of the LUMPREP preprocessor are provided in the *lumprep_example* folder. To run LUMPREP, type the following command in a command-line window open to this folder, and then respond to its prompt as follows:

```
..\lumprep
Enter name of LUMPREP control file: lumprep.in
```

LUMPREP produces two LUMPREM input datasets. LUMPREM can be run using these datasets by typing the following commands at the screen prompt.

```
..\lumprem lr_lm1.in lr_lm1.out
..\lumprem lr_lm2.in lr_lm2.out
```

2. Concepts

2.1 General

Figure 2.1 illustrates processes that are simulated by RECHMOD. The zone occupied by plant roots, from which water is evapotranspired, is modelled as a single “soil moisture store”. It receives water from rainfall, and maybe irrigation. It loses water through vegetative extraction, direct drainage, and macropore drainage, the latter only occurring when the store is full. Recharge and macropore drainage losses from the soil moisture store are housed in separate buffers (these representing storage within the weathered zone beneath the soil); this buffer-stored water is released as recharge and macropore recharge respectively. The delay suffered by each of these recharge waters is supplied by the user in days. Note that, although supplied in days, the delay can include fractions of a day; parameter continuity is fundamental to PEST’s ability to calibrate this model. (It was decided to account for recharge delay by supplying a direct delay parameter rather than through simulation of a sub-soil moisture store in order to retain better control over the timing of recharge with respect to rainfall than is afforded through use of the latter mechanism.)

If requested, irrigation water can be supplied to the soil moisture store to meet irrigation demand. Irrigation is provided on an as-needed basis. Only enough irrigation water is supplied during each simulation time step to prevent the volume of water in the soil moisture store from falling below a user-supplied relative level. Normally this level is set to half the volume of the store in order to emulate the behaviour of a collection of irrigators rather than any one irrigator. Where a certain percentage of irrigation water is extracted from the ground (this fraction being supplied by the user), this extraction is recorded so that it can be supplied to the same groundwater model as that for which RECHMOD calculates recharge.

The various components of the RECHMOD water balance algorithm are now described in detail. The discussion herein assumes that time units are days and that length units are meters; however any consistent system of units can be employed.

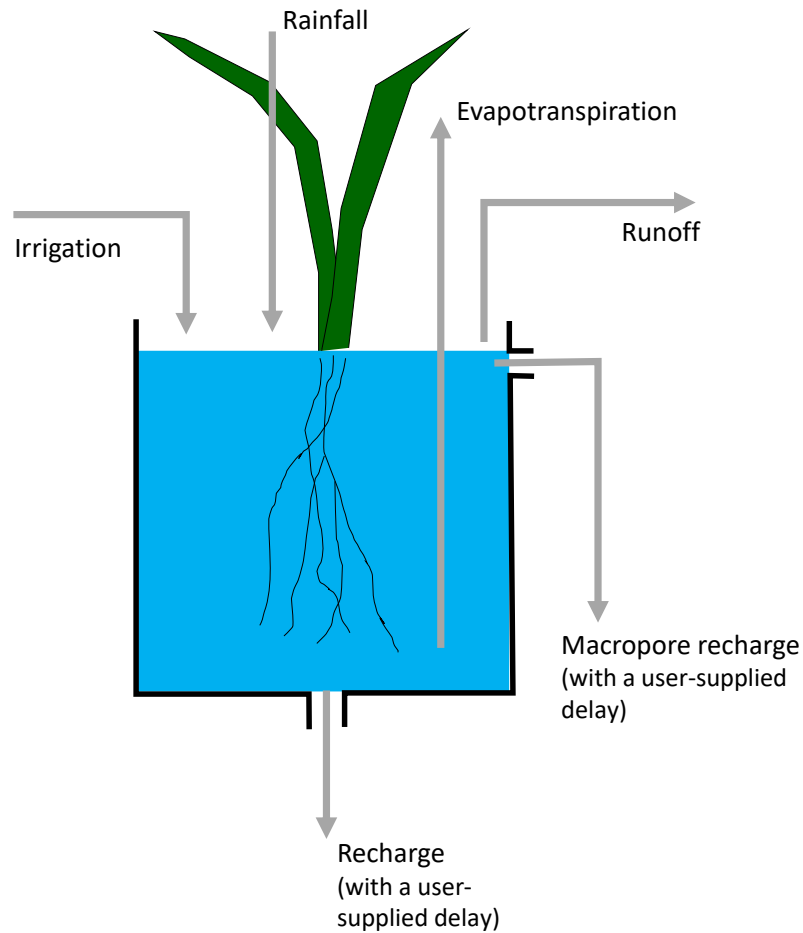


Figure 2.1. Aspects of the unsaturated zone water balance simulated by RECHMOD.

2.2 Volume of the Soil Moisture Store

The moisture store which is central to calculations undertaken by RECHMOD represents water that is stored within the plant root zone. Hence the “volume” of the store is the maximum amount of water that this root zone can hold. This is commonly referred to as the “plant available water capacity”. It is roughly equal to the depth of the root zone times the soil porosity within this zone (after correcting for that component of porosity that holds water that is incapable of either moving downwards, or of being used by plants).

2.3 Precipitation

RECHMOD receives precipitation as a sequence of daily inputs.

2.4 Evapotranspiration

RECHMOD receives potential evaporation as a sequence of daily inputs. However the rate at which plants actually evapotranspire is dependent on the volume of water within the soil moisture store, this being in accordance with the principal that when there is less water in the root zone; this water is held more tightly by the soil. The rate of water loss through evapotranspiration is calculated using the equation:

$$E = f E_p \frac{1 - e^{-\gamma'}}{1 - 2e^{-\gamma} + e^{-\gamma'}} \quad (2.1)$$

where:

E = rate of water loss through evapotranspiration;

E_p = potential evapotranspiration rate;

f = “crop factor”;

v' = relative volume of water in the soil moisture store (i.e. v/v_{max} where v is the current volume of water in the store and v_{max} is the volume of the soil moisture store); and

γ = a parameter determining the shape of the evaporation rate vs. stored water relationship (see below).

Crop factor and γ are user supplied. They can be constant over a simulation period or, if desired, can change throughout the simulation period to simulate vegetation growth, harvesting and/or clearing.

Figure 2.2 shows the dependence of evapotranspiration on γ assuming that $f \times E_p$ is unity.

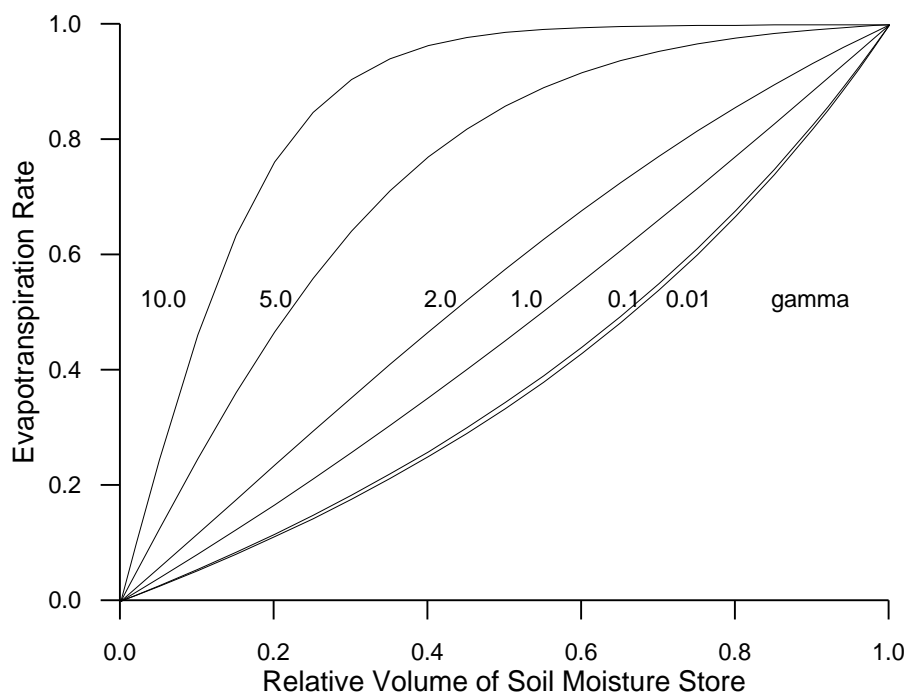


Figure 2.2. Dependence of evapotranspiration rate on γ (i.e. gamma).

2.5 Recharge

Water is lost from the soil moisture store as continuous unsaturated vertical flow to that portion of the subsurface which lies below the root zone. The rate at which water is lost in this manner is dependent on the volume of moisture currently stored, this emulating the fact that the hydraulic conductivity of unsaturated material decreases (often rapidly) with decreasing degree of saturation. Rate of water loss is expressed by the following equation (after van Genuchten, 1980):

$$R = K_s [v']^l \left[1 - \left(1 - [v']^{1/m} \right)^m \right]^2 \quad (2.2)$$

where:

R = rate of drainage;

K_s = saturated hydraulic conductivity;

v' = relative volume of water in the soil moisture store (i.e. v/v_{max}); and

- l = pore-connectivity parameter (estimated by Mualem, 1976, to be about 0.5 for many soils); and
- m = a parameter determining the shape of the drainage rate vs. stored water relationship (see below).

As the model depicted in figure 2.1 is not a true unsaturated water zone model based on Richards equation, “measurable” soil parameters such as K_s and l are not strictly useable in the above equation for soil moisture store drainage. However, values supplied to this equation are expected to be of the same order of magnitude as those which would be used to represent soil properties in a more physically exact simulator of unsaturated zone processes.

For $K_x[v]'$ equal to unity, figure 2.3 shows the variation of drainage rate with m .

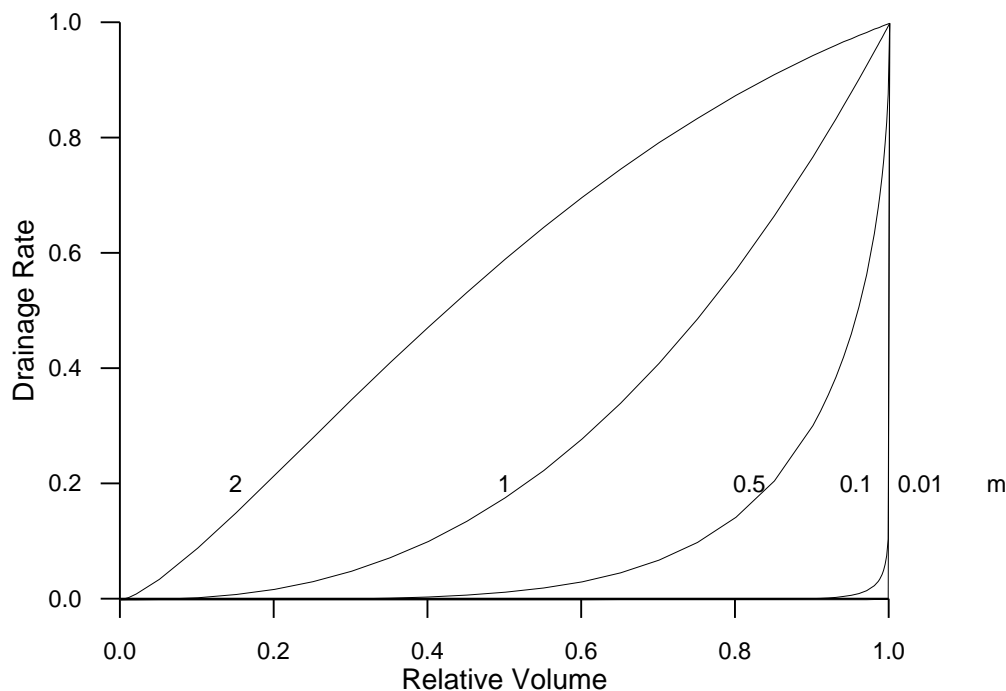


Figure 2.3. Dependence of soil moisture store on drainage rate on m .

Leakage from the soil moisture store, as calculated by RECHMOD using equation 2.2, is not released as groundwater recharge immediately. Rather it suffers a delay, the length of this delay being a user-specified parameter. The delay is normally supplied in days; fractional delays are permitted. This provides RECHMOD with the ability to replicate a phenomenon that is observed at many sites, namely that groundwater levels do not begin to rise until a certain time has elapsed since a significant rainfall event. This lag depends on the water table depth and on the hydraulic properties of the subsurface between the soil and the saturated zone.

RECHMOD records recharge at user nominated output times, these times, presumably, being those times at which recharge is required by a groundwater model. The recorded recharge represents all recharge that has occurred since the previous output time.

2.6 Macropore Recharge

In spite of the fact there is often a delay between rainfall and recharge, there are also occasions, mostly after periods of particularly heavy rain, where recharge is either immediate or is delayed very little. This is an outcome of so-called “macropore recharge” whereby temporarily saturated conditions in the upper subsurface allow water to migrate laterally to zones of preferential downward flow where

it then migrates quickly to the saturated zone. In some areas the bulk of groundwater recharge occurs as a result of macropore flow in response to heavy, sometimes cyclonic, rainfall events.

RECHMOD allows macropore drainage to occur when the soil moisture store is full (and saturated conditions are therefore assumed to prevail in the shallow subsurface). The user must supply the maximum amount of macropore drainage allowed to occur on any one day. Denoting this quantity as R_m , any overflow of the container illustrated in figure 2.1 occurring during periods of very wet weather is partitioned between macropore recharge and runoff; the first R_m units (or all of the overflow if the amount is less than R_m) becomes macropore recharge.

Like recharge from the bottom of the soil moisture store, macropore recharge is delayed by a user-specified amount. This will normally be considerably smaller than the delay incurred by normal recharge.

At user-requested output times LUMPREM records drainage and macropore recharge (accumulated since the last output time) both individually and summed.

2.7 Irrigation

The user must supply the value of a variable named F_i , this being the “fractional irrigation threshold”. F_i must lie between 0.0 and 1.0. On any day on which irrigation is decreed to be active, the volume of water in the soil moisture store is raised to F_i times the total volume of the store, this compensating for water losses incurred through evapotranspiration and drainage on that day. Normally F_i is set to 0.5. This accommodates the fact that while irrigation may not be applied to any one paddock unless the water content in that paddock’s soil moisture store is low, the water content averaged over many paddocks is a half of the composite soil moisture store of those paddocks. That is, the irrigation water that is demanded by all of these paddocks is that which is required to maintain all of their moisture stores at an average of half full. Daily irrigation demand therefore balances daily evapotranspiration plus daily deep drainage on a regional scale.

Where RECHMOD is used as a recharge model for a groundwater model, the demand for groundwater that is used for irrigation must be supplied as an extraction term to the groundwater model. RECHMOD does not deduct irrigation water from the recharge that it calculates for the groundwater system. Rather it records groundwater-irrigation-based extraction so that the user can arrange for the extraction of the necessary amount of water to take place from the groundwater model him/herself. The user informs RECHMOD of the fraction of irrigation water that is sourced from the ground. If desired, this fraction can change over time.

2.8 Runoff

Overflow of the soil moisture store is partitioned between macropore recharge and runoff. After macropore recharge requirements have been met, the residual overflow is assigned to runoff. Like recharge, runoff is summed over user-supplied accumulation intervals so that it can be recorded on an output file at the end of each such interval.

2.9 Volume to Elevation

RECHMOD calculates, and continually updates, the volume of water in the soil moisture store. In many instances of RECHMOD usage, RECHMOD-calculated recharge, irrigation demand and groundwater extraction are the items of principle interest. There may be occasions, however, when the volume of water in the container representing the soil moisture store is also of some interest. After appropriate transformation, this may be used to set transient boundary conditions for a groundwater model; in a case such as this, RECHMOD is presumably being used to simulate storage of water in a part of the groundwater system that is outside of the domain of the groundwater model itself.

LUMPREM records the volume of water stored in the RECHMOD container in its output file, along with other RECHMOD-calculated quantities such as recharge and irrigation requirements. In one of the final

columns of this output file it also records a time-series of “elevations” that are calculated from stored water volumes. These elevations are calculated using the formula:

$$h = a + f_1 v + f_2 v^p \quad (2.3)$$

where:

- h is the calculated elevation;
- v is the volume of water in the container;
- a is an offset;
- f_1 is a factor (referred to as “factor 1” in LUMPREM input);
- f_2 is a factor (referred to as “factor 2” in LUMPREM input); and
- p is a power.

As well as elevation, LUMPREM also records depth to water. This is calculated by subtracting the water elevation from a reference elevation (for example the elevation of the topographic surface); the latter is supplied by the user. Let the reference (topographic) elevation be denoted by s . The depth to water d is calculated using the formula:

$$d = s - h \quad (2.4)$$

where:

- d is the depth to water; and
- s is the reference elevation.

3. Solution Method

RECHMOD assumes that rainfall, E_p , γ , and f are constant over each day. However RECHMOD actually performs its calculations over intervals of time that are less than this, the solution time step being a user-supplied fraction of a day. Within each such time step, RECHMOD calculates all aspects of the water balance (as well as irrigation demand and groundwater extraction to support this demand) in accordance with the equations presented in the previous section. However because the volume of water in the soil moisture store varies with time as a result of the subtractions and additions that contribute to the conceptual water balance, an iterative solution procedure is adopted so that the calculated outcomes at the end of each time step do not depend on the order in which the various components are calculated and added (or subtracted). In this way updated water volumes and recharge/runoff/evaporation/irrigation rates are all in accordance with the discretised form of the differential equation describing instantaneous water balance within the soil moisture store.

Both a fully implicit scheme and a Crank-Nicholson method were tested within the iterative solution process. In the latter method E and R are calculated using equations 2.1 and 2.2 on the basis of a v' that is half way between the initial and final volumes for any time step. In the former method the final v' figure is employed. Iteration is required in both cases because the final v' is not known at the time E and R are calculated.

Experience shows that convergence is very fast for both methods. However the Crank-Nicholson scheme performs better in terms of agreement between water balance components calculated on the basis of different lengths of the solution time step (i.e. it converges more rapidly with respect to time step length). It is thus retained in the RECHMOD model.

As a check on its calculations, LUMPREM carries out a global water audit at each of its output times. Ideally, mass balance errors should be minimal (of the same order as numerical errors). All water balance components are written to its output file at each user-requested output time.

4. Using LUMPREM

As was discussed above, iterative water balance calculations take place within subroutine RECHMOD. This is called by the LUMPREM main program. However RECHMOD can also be called by another program (for example a groundwater model) if desired. In order to support flexibility of its usage, this section provides a description of its arguments.

4.1 Subroutine RECHMOD

Subroutine RECHMOD is called using the statement:

```
call rechmod(mxiter,tol,ndays,subdim,nstep,maxvol,
+ irrigvolfrac,cropfac_day,gamma_day,ks,l,m,mflowmax,rdelay,
+ mdelay,vol,drainsub,macsub,rain,epot,recharge,mrecharge,
+ runoff,evapn,rainfall,potevapn,irrigation,gwithdrawal,
+ irrigcode,gwirrigfac)
```

Subroutine arguments are listed below:

<i>mxiter</i>	The number of permissible iterations per time step. RECHMOD stops with an error message if convergence has not been attained after <i>mxiter</i> iterations. [integer, intent in]
<i>tol</i>	The convergence tolerance. The solution is assumed to be calculated when two successive estimates of v' differ by less than <i>tol</i> . [integer, intent in]
<i>ndays</i>	The number of days to be simulated on the present subroutine call. [integer, intent in]
<i>subdim</i>	Dimension of the recharge and macropore recharge delay buffer arrays in the calling program. [integer, intent in]
<i>nstep</i>	The number of steps into which each day is subdivided for the purpose of iterative soil moisture store computation. [integer, intent in]
<i>maxvol</i>	The volume of the soil moisture store. [real, intent in]
<i>irrigvolfrac</i>	The fractional soil moisture volume F_i which irrigation water must maintain. [double precision, intent in]
<i>cropfac_day</i>	An array of dimension <i>ndays</i> . The crop factor (f of equation 2.1) on each of the <i>ndays</i> days simulated on the current call to RECHMOD. [double precision, intent in]
<i>gamma_day</i>	An array of dimension <i>ndays</i> . Gamma (γ of equation 2.1) on each of the <i>ndays</i> days simulated on the current call to RECHMOD. [double precision, intent in]
<i>ks</i>	K_s from equation 2.2. [double precision, intent in]
<i>l</i>	l from equation 2.2. [double precision, intent in]
<i>m</i>	m from equation 2.2. [double precision, intent in]
<i>mflowmax</i>	Maximum macropore recharge allowed per day. [double precision, intent in]
<i>rdelay</i>	Delay, in days, between water draining from the soil moisture store (as calculated using equation 2.1) and being assigned to groundwater recharge. [double precision, intent in]
<i>mdelay</i>	Delay, in days, between water leaving the soil moisture store (calculated as overflow up to <i>mflowmax</i>) and being assigned to macropore groundwater recharge [double precision, intent in]

<i>vol</i>	On input, this is the current volume of water in the soil moisture store. This is updated by RECHMOD to represent current water volume after <i>ndays</i> on RECHMOD exit. [double precision, intent inout]
<i>drainsub</i>	An array of dimension <i>subdim</i> . On input, this array contains current contents of the recharge delay buffer. Updated by RECHMOD. [double precision, intent inout]
<i>macsub</i>	An array of dimension <i>subdim</i> . On input, this array contains current contents of the macropore recharge delay buffer. Updated by RECHMOD. [double precision, intent inout]
<i>rain</i>	An array of dimension <i>ndays</i> containing rainfall for each day simulated during the current RECHMOD call. [double precision, intent in]
<i>epot</i>	An array of dimension <i>ndays</i> containing potential evaporation for each day simulated during the current RECHMOD call. [double precision, intent in]
<i>recharge</i>	Cumulative recharge (after passing through the recharge delay buffer) for <i>ndays</i> simulated by the current RECHMOD call. [double precision, intent out]
<i>mrecharge</i>	Cumulative macropore recharge (after passing through the macropore recharge delay buffer) for <i>ndays</i> simulated by the current RECHMOD call. [double precision, intent out]
<i>runoff</i>	Cumulative runoff for <i>ndays</i> simulated by the current RECHMOD call. [double precision, intent out]
<i>evapn</i>	Cumulative evaporation for <i>ndays</i> simulated by the current RECHMOD call. [double precision, intent out]
<i>rainfall</i>	Cumulative rainfall for <i>ndays</i> supplied on the current RECHMOD call [double precision, intent out]
<i>potevapn</i>	Cumulative potential evaporation for <i>ndays</i> supplied on the current RECHMOD call. [double precision, intent out]
<i>irrigation</i>	Cumulative irrigation for <i>ndays</i> simulated over the current RECHMOD call. Note that irrigation only occurs on days for which the <i>irrigcode</i> array entry (see below) is 1. [double precision, intent out]
<i>gwithdrawal</i>	Cumulative groundwater withdrawal simulated over the current RECHMOD call. On any given day, groundwater withdrawal occurs only if irrigation occurs, and only if the pertinent element of the <i>gwirrigfrac</i> array is greater than zero. [double precision, intent out]
<i>irrigcode</i>	An array of dimension <i>ndays</i> containing the irrigation code for each day simulated during the current RECHMOD call. A value of zero disables irrigation; a value of 1 enables irrigation. [integer, intent in]
<i>gwirrigfrac</i>	An array of dimension <i>ndays</i> containing the fraction of irrigation water that is sourced from groundwater for each day simulated during the current RECHMOD call; this must be between (and including) zero and one. [double precision, intent in]

On each occasion that RECHMOD is called, it carries out calculations over a simulation period of *ndays* full days. The simulation period is assumed to begin at midnight preceding day 1 and finish at midnight following day *ndays*. The crop factor, rainfall, potential evaporation, gamma, irrigation code and groundwater irrigation fraction are assumed to be constant for each day. Calculated recharge, macropore recharge, runoff, evapotranspiration, irrigation and groundwater-sourced irrigation are accumulated over the *ndays* simulated days. RECHMOD returns the current volume of the soil

moisture storage, as well as the volume of water that is currently stored within each of the recharge and macropore recharge delay buffers.

4.2 Program LUMPREM

4.2.1 Running LUMPREM

LUMPREM is a simple program which reads climatic, soils, vegetation and irrigation data for the use of subroutine RECHMOD. It records the outcomes of RECHMOD's calculations on its output file. At each output time it also computes a complete water balance, thus checking that all of RECHMOD's calculations are in order.

LUMPREM can be run in either of two ways. The first is to type its name at the command-line prompt. On commencement of execution, it then prompts for the name of its input control file, and for the name of the output file which it must write. Alternatively, the name of its input and output files can be supplied on the LUMPREM command line.

4.2.2 Inputs

LUMPREM reads up to five input files and generates one output file. As presently programmed, input/output is very rudimentary; it may be improved over time. Its main input file (i.e. its "control" file) is easily prepared using a text editor. However rainfall, potential evaporation and irrigation data files may be more difficult to prepare, especially for long simulation times. These will normally need to be written using simple, specially-coded software. LUMPREM's output file can be read by any spreadsheet or plotting program.

The structure of the LUMPREM control file is illustrated in figure 4.1; examples are provided in figures 4.2a and 4.2b. Items on each line can be comma, tab or space delimited; rigid formatting is unnecessary. At present, error-checking is rudimentary; it may be improved over time.

```
* earth properties
maxvol irrigvolfrac
rdelay mdelay
ks m l mflowmax
* volume to elevation
offset factor1 factor2 power [elevmin elevmax]
* topographic surface
surface
* initial conditions
vol
nrbuf nmbuf
(rbuf(i), i=1,nrbuf)
(mbuf(i), i=1,nmbuf)
* solution parameters
nstep mxiter tol
* timing information
numdays noutdays
(outday(i),i=1,noutdays)
data filenames
vegfile OR cropfac_all gamma_all
rainfile
epotfile
irrigfile OR irrigcode_all gwirrigfrac_all
```

Figure 4.1. Structure of the LUMPREM input control file. The *elevmin* and *elevmax* variables are optional.

```

* earth properties
.3 0.5
10.8 0.4
.015 0.7 0.5 0.2
* volume to elevation
1.0 2.0 3.0 0.5
* topographic surface
100.0
* initial conditions
.1
1 1
0.0
0.0
* solution parameters
1 100 1.0e-5
* timing information
100 10
9 11 32 33 40 41 42 43 90 100
* data filenames
vegfile.dat
rainfile.dat
epotfile.dat
irrigfile.dat

```

Figure 4.2a. A typical LUMPREM input file. Vegetation and irrigation time-series data are read from files *vegfile.dat* and *irrigfile.dat*.

```

* earth properties
.3 0.5
10.8 0.4
.015 0.7 0.5 0.2
* volume to elevation
1.0 2.0 3.0 0.5
* topographic surface
120.5
* initial conditions
.1
1 1
0.0
0.0
* solution parameters
1 100 1.0e-5
* timing information
100 10
9 11 32 33 40 41 42 43 90 100
* data filenames
0.8 0.45
rainfile.dat
epotfile.dat
1 0.5

```

Figure 4.2b. A typical LUMPREM input file. Global values of vegetation and irrigation parameters are read from this file.

In figure 4.1, the variables *maxvol*, *irrigvolfrac*, *rdelay*, *mdelay*, *ks*, *m*, *l*, *mflowmax*, *vol*, *nstep*, *mxiter* and *tol* are identical to those discussed in the preceding section. Those variables that were not discussed in that section are now described.

elevmin Elevation values computed using equation 2.3 are clipped if they are below *elevmin*.

elevmax Elevation values computed using equation 2.3 are clipped if they are above *elevmax*.

<i>offset</i>	a of equation 2.3.
<i>factor1</i>	f_1 of equation 2.3.
<i>factor2</i>	f_2 of equation 2.3.
<i>power</i>	p of equation 2.3.
<i>surface</i>	s of equation 2.4.
<i>nrbuf</i>	The number of elements of the <i>rbuf</i> array to be supplied on the next line.
<i>nmbuf</i>	The number of elements of the <i>nmbuf</i> array to be supplied two lines down.
<i>rbuf</i>	The recharge delay buffer. Supply initial values for the elements of this array. The content of the first element is assumed to have left the soil moisture store on the previous day; the content of the second element is assumed to have drained from the soil moisture store two days previously, etc.
<i>mbuf</i>	The macropore recharge delay buffer. Supply initial values for the elements of this array. The content of the first element is assumed to have left the soil moisture store on the previous day; the content of the second element is assumed to have left the soil moisture store two days earlier, etc.
<i>numdays</i>	The total number of days to be simulated by LUMPREM.
<i>noutdays</i>	The number of days for which output is required (equivalent to the number of times that LUMPREM calls RECHMOD).
<i>outday</i>	An integer array with <i>noutdays</i> elements listing the simulation days at which output is required.
<i>vegfile</i>	The name of the “vegetation file” which LUMPREM must read to obtain crop factor and γ data. Alternatively, time-invariant values of the crop factor and gamma can be read from this line.
<i>rainfile</i>	The name of the “rainfall file” which LUMPREM must read to obtain rainfall data.
<i>epotfile</i>	The name of the “evaporation file” which LUMPREM must read to obtain potential evaporation data.
<i>irrigfile</i>	The name of the “irrigation file” which LUMPREM must read to obtain irrigation data. Alternatively, time-invariant values for the irrigation code and groundwater extraction fraction can be read from this line.

If supplied, a vegetation file must have three columns of data. On each line the first entry is the simulation day (a simulation starts at midnight on the day preceding day 1 and finishes at midnight on day *numdays*). The crop factor and gamma value corresponding to that day follow the day itself. For a vegetation file:

- Simulation days are integers.
- The first simulation day must be day 1.
- The last simulation day must be equal to or greater than *numdays*.
- Days must be arranged in increasing order.
- Crop factors and gamma are real numbers.
- Not every day needs to be represented. For those days which are not represented in the vegetation file, the crop factor and gamma are computed by LUMPREM using linear interpolation.

The rainfall file must contain two columns of data. On each line the first entry is the simulation day while the second entry is the rainfall corresponding to that day (a real number). For missing days the rainfall is assumed to be zero.

Like the rainfall file, the evaporation file must contain two columns of data. The first entry on each line contains the simulation day; the second entry contains potential evaporation for that day (a real number). For an evaporation file:

- The simulation first day must be day 1.
- Days must be provided in increasing order.
- Not every day needs to be represented. Where a day is absent, the potential evaporation is assumed to be that pertaining to the preceding listed day.

If supplied, the irrigation file must contain three columns of data. The first entry on each line specifies the simulation day. The second entry specifies the irrigation code for that day (an integer), while the third entry contains the fraction of irrigation water that is supplied by groundwater (a real number). An irrigation code of zero signals that irrigation is not operative for that day. The groundwater irrigation fraction must lie between zero and one. A value of zero signifies that no irrigation water is sourced from groundwater. A value of one signifies that all irrigation water is sourced from groundwater.

For an irrigation file:

- The first simulation day must be day 1.
- Days must be provided in increasing order.
- Not every day needs to be represented. Where a day is absent, both the irrigation code and the groundwater irrigation fraction are assumed to be those pertaining to the preceding listed day.

4.2.3 Outputs

LUMPREM produces one output file. Data recorded on this file are arranged in columns; it is too wide for depiction in this document. Each column is labelled. Columns are, in order:

1. simulation day,
2. current soil moisture store volume (pertaining to midnight following the named simulation day),
3. current volume of water stored in the recharge delay buffer,
4. current volume of water stored in the macropore recharge delay buffer,
5. change in water stored in soil moisture store since last output time,
6. change in water stored in recharge delay buffer since last output time,
7. change in water stored in macropore delay buffer since last output time,
8. rainfall since last output time,
9. irrigation since last output time,
10. drainage recharge since last output time,
11. macropore recharge since last output time,
12. total recharge since last output time,
13. groundwater irrigation withdrawal since last output time,
14. net recharge since last output time (i.e. total recharge – groundwater irrigation withdrawal),
15. runoff since last output time,
16. potential evaporation since last output time,
17. evapotranspiration since last output time,
18. groundwater potential evapotranspiration since last output time (i.e. potential evaporation minus actual evapotranspiration),

-
19. water imbalance since last output time (this should be a very small number),
 20. elevation computed using equation 2.3,
 21. depth to water computed using equation 2.4.

LUMPREM's output file is easily imported into a spreadsheet or graphing package for plotting and display.

5. LR2SERIES

5.1 General

“LR2SERIES” stands for “LUMPPREM output file to time series file”. This utility reads one or a number of LUMPREM output files and writes one or a number of MODFLOW 6 time series files. The suite of time series that it writes to the latter files are the same as, or modified from, those read from LUMPREM output files.

5.2 LR2SERIES Input Control File

5.2.1 An Example

LR2SERIES obtains instructions from an input control file. Figure 5.1 shows an example of such a file.

```

READ_LUMPREM_OUTPUT_FILE lr_lm1.out 5
#  my_name      LUMPREM_name      divide_by_delta_t?

    rech1       total_rech         div_delta_t
    netrech1     net_recharge       div_delta_t
    gw1          gw_withdrawal      div_delta_t
    gwevap1      gw_pot_evap        div_delta_t
    elev1        elevation          no_div_delta_t

READ_LUMPREM_OUTPUT_FILE "another.out" 5
#  my_name      LUMPREM_name      divide_by_delta_t?

    "rech2"      total_rech         div_delta_t
    "netrech2"   net_recharge       div_delta_t
    gw2          gw_withdrawal      div_delta_t
    gwevap2      gw_pot_evap        div_delta_t
    elev2        elevation          no_div_delta_t

WRITE_MF6_TIME_SERIES_FILE temp.out 6
#  ts_name      scale      offset      mf6_method

    rech1        1.0        0.0        linearend
    elev1         1.0        0.0        linearend
    "gwevap1"     1.0        0.0        linearend
    rech2         1.0        0.0        linearend
    elev2         1.0        0.0        linearend
    gwevap2       1.0        0.0        linear

WRITE_MF6_TIME_SERIES_FILE temp1.out 7 1.0
    rech1        1.0        1.0        linearend  next
    elev1         1.0        1.0        stepwise  next
    gwevap1       1.0        1.0        linearend  3.0-5
    rech2         1.0        1.0        linearend  0.0
    elev2         1.0        1.0        linear    120.0
    elev2         1.0        2.0        linear    120.0
    gwevap2       1.0        1.0        linear    next

```

Figure 5.1 An LR2SERIES input control file.

5.2.2 Specifications

LR2SERIES actions are instigated by keywords within the LR2SERIES input file. Two keywords are supported, these being:

- READ_LUMPREM_OUTPUT_FILE

- WRITE_MF6_TIME_SERIES_INPUT_FILE

The LR2SERIES actions which are instigated by these keywords are obvious from their names. Information which follows these keywords in a LR2SERIES input file specifies the details of these actions.

As is apparent from the above example, a blank line can be placed anywhere within a LR2SERIES input control file. Lines beginning with the “#” character can also be placed anywhere; these are comment lines.

Each of these keywords is now discussed in detail.

5.2.3 READ_LUMPREM_OUTPUT_FILE

The READ_LUMPREM_OUTPUT_FILE keyword must be followed on the same line by two other entries. The first is the name of the LUMPREM output file that LR2SERIES must read. The second is an integer greater than zero specifying the number of time series that must be read from that file. The name of the LUMPREM output file can optionally be enclosed in quotes.

Suppose that the number of time series to be read from the LUMPREM output file is N . Then N lines must follow the line which contains the READ_LUMPREM_OUTPUT_FILE keyword. Each of these lines must contain three entries. These entries are in order:

- The user-supplied name of the time series. This name must be twenty characters or less in length and contain no blanks.
- The text header to a time series recorded in the LUMPREM output file.
- The *div_delta_t* string. This must be supplied as either “div_delta_t” or “no_div_delta_t”.

If *div_delta_t* is supplied as “div_delta_t”, then a time series is modified as it is read from the LUMPREM output file. The value corresponding to time i is divided by $t_i - t_{i-1}$ where t_i and t_{i-1} are the times (in days) corresponding to entries i and $i-1$ of the time series. Meanwhile, the time series value ascribed to time zero is set to the same value as the ensuing time. This is appropriate for quantities such as recharge and evapotranspiration. LUMPREM records the volume of water that is lost to the unsaturated zone system through each of these mechanisms since the last output time step; the rate is obtained by dividing the volume by the pertinent time interval.

The name given to each time series by the user must be unique. This name is used to refer to the time series in later LR2SERIES processing.

The LUMPREM text header uniquely identifies each time series recorded on a LUMPREM output file. This name is not case sensitive; it can be repeated if desired. However the name provided to LR2SERIES must be in exact accordance with that provided in the LUMPREM output file.

Multiple READ_LUMPREM_OUTPUT_FILE keywords can exist in the same LR2SERIES input control file. The number of time series that are stored in LR2SERIES’s memory is increased accordingly on each occasion that this keyword is encountered. However LR2SERIES will cease execution with an error message if it is asked to read LUMPREM output files which employ different time bases. Thus all LUMPREM output files read by LR2SERIES must have identical first columns. This column is headed “days”; it records LUMPREM simulation times. LR2SERIES will also cease execution with an appropriate error message if the user-supplied name ascribed to a time series is repeated on subsequent READ_LUMPREM_OUTPUT_FILE calls.

5.2.4 WRITE_MF6_TIME_SERIES_FILE

The WRITE_MF6_TIME_SERIES_FILE keyword must be followed on the same line by two other text strings. Optionally a third text string can follow.

The first text string is the name of the MODFLOW 6 time series file that LR2SERIES must write. The second is an integer greater than zero specifying the number of time series that must be written to that file. The name of the MODFLOW 6 time series file can optionally be enclosed in quotes. The optional third text string can be a real number specifying the time that must be added to all time series elements before the time series is recorded in the MODFLOW 6 input file. This number can be any number that is zero or greater. The default value (which is ascribed to this time series delay if the time delay is not provided by the user) is 0.0.

Suppose that the number of time series to be written to a MODFLOW 6 time series file is N . Then N lines must follow the line which contains the WRITE_MF6_TIME_SERIES_FILE keyword. Each of these lines must contain four entries, and an optional fifth entry. The first four entries are in order:

- The user-supplied name of a previously read time series.
- A number by which all elements of this time series are multiplied; this is referred to as the *scale*.
- A number which is added to all elements of the time series after multiplication by the *scale*; this is referred to as the *offset*.
- The MODFLOW 6 *method* which specifies how MODFLOW 6 interpolates entries of the time series to its own time base. In accordance with MODFLOW 6 protocol, this must be “stepwise”, “linear” or “linearend”; it is case-insensitive.

If the time delay ascribed to a WRITE_MF6_TIME_SERIES_FILE block is greater than zero, then LR2SERIES records an additional element for each time series cited in this block as it records these time series in the MODFLOW 6 time series file. The time associated with this additional element is 0.0. The value associated with this new element for each recorded time series is determined by the optional fifth element supplied on each line of the WRITE_MF6_TIME_SERIES_FILE block. This element must be provided if the series time delay is greater than zero. It can be supplied as a number, in which case it is the actual fill-in value for this new first time series element. Note that this value is NOT multiplied by the user-supplied scale ascribed to the time series; nor is the user-supplied offset added. Alternatively, this fifth string can be supplied as “next”. This instructs LR2SERIES to re-use the same value as that associated with the following element of that time series – the element which used to be the first element before the new element was inserted. Note that, as stated above, the time associated with all time series elements other than the newly inserted element for a time of zero, is shifted back by the time series delay ascribed to the current WRITE_MF6_TIME_SERIES_FILE block. Note also that if the time series delay is zero, LR2SERIES does look for a fifth entry on each line of the WRITE_MF6_TIME_SERIES_FILE block.

The user-supplied name for a time series is recorded in the ATTRIBUTES block of the MODFLOW 6 time series input file. If desired, multiple instances of the same time series can be written to the same MODFLOW 6 time series file using the same WRITE_MF6_TIME_SERIES_FILE keyword, possibly with a different *scale* and/or *offset* for each instance of the time series. If this occurs, LR2SERIES appends the suffix *_J* to the name of the time series on each occasion that it is repeated in order to ensure naming uniqueness in the MODFLOW 6 time series file.

Multiple WRITE_MF6_TIME_SERIES_FILE keywords can occur in the same LR2SERIES input file. Thus a number of different MODFLOW 6 time series files can be written during a single LR2SERIES run.

6. LUMPREP

6.1 General

As the name implies, LUMPREP is a pre-processor for LUMPREM. It supports creation of:

- input files for multiple LUMPREM models;
- PEST template files of these LUMPREM input files;
- a partial PEST control file that cites LUMPREM parameters;
- rainfall and irrigation files required by the LUMPREM models;
- a model batch file that runs multiple incidences of LUMPREM.

LUMPREP obtains daily rainfall and irrigation data from files that are easily downloadable from the SILO climate database. See: <https://www.longpaddock.qld.gov.au/silo/> See the LUMPREP example installed with LUMPREM for an example of this type of file.

LUMPREP does not prepare irrigation and/or vegetation files for a LUMPREM model. However these files are not always required. As is explained in previous sections of this manual, a LUMPREM control file can cite single values for the *cropfactor* and *gamma* variables instead of the name of a vegetation file. Similarly, it can cite single values for the *irrigcode* and *gwirrigfrac* variables instead of the name of an irrigation file. If desired, values for all of these variables can be provided for one or many LUMPREM models through the LUMPREP input file.

After it has written one or a number of input datasets for LUMPREM models, LUMPREP optionally writes a partial PEST control file which cites estimable parameters pertaining to these models, together with PEST control variables; it is the user's responsibility to add observations to this file. If it writes a partial PEST control file, LUMPREP also writes part of the batch file which this control file cites in its "model command line" section. Commands within this batch file run all LUMPREM models created by LUMPREP.

6.2 The LUMPREP Input File

The LUMPREP input file is comprised of a set of keywords and their respective values. Each keyword is listed on a new line; its user-supplied value must follow its name. Keywords represent variables which must be given either a single value or, in the case of some keywords, two values. The "value" of a keyword can be filename, a number, or a text string.

A representative LUMPREP input file is depicted in figure 6.1.

```

START_DATE      31/1/1990
END_DATE        5/1/2000
NDAY_OUT        monthly
STEPS_PER_DAY   5

# The first LUMPREM dataset

SILOFILE        "39104pp.txt"  evap
RAINFIL        rain.dat
EPOTFILE        epot.dat
VEGFIL        "veg file.dat"
VEGFIL        0.2 1.5
IRRIGFILE       1 0.4
MAXVOL          0.5
IRRIGVOLFRAC    0.6
RDELAY         15.0
MDELAY         2.0
KS              0.9
M              0.2
L              0.55
MFLOWMAX       0.1
OFFSET         15.0
FACTOR1        1.0
FACTOR2        1.11
POWER          2.0
VOL            0.15
SURFACE        120.0
LUMPREM_MODEL_NAME  lm1

# -- The second LUMPREM dataset

SILOFILE        "39104pp.txt"  evap
RAINFIL        rain.dat
EPOTFILE        epot.dat
OFFSET         0.0
FACTOR1        1e-5
FACTOR2        1e-5
POWER          0.0
IRRIGFILE       irrig.dat
VEGFIL        veg.dat
LUMPREM_MODEL_NAME  lm2

# -- Write a partial PEST dataset

BATCH_FILE_NAME  "model.bat"
PEST_CONTROL_FILE temp.pst

```

Figure 6.1 An example of a LUMPREP input file.

As is apparent from figure 6.1, blank lines and comment lines can be interspersed with lines that contain keywords and their values in a LUMPREP input file. A comment line begins with the “#” character.

Each keyword in a LUMPREP input file instigates an action. In some cases the action can be comprised simply of assignment of a value to a keyword. In other cases this can be accompanied by a more complex action such as the reading of a file and/or the writing of one or more files.

Certain keywords (namely START_DATE, END_DATE and NDAY_OUT) can be provided only once in a LUMPREP input file. These keywords must reside near the top of this file as the values that are associated with them are required for actions that are instigated by other keywords. Some of these other keywords can be featured many times in a LUMPREP input file - once for each LUMPREM model that is created by LUMPREP. Some keywords are optional; where they are omitted LUMPREP provides default values.

6.3 LUMPREP Keywords - General

Table 6.1 lists the keywords that can be supplied on a LUMPREP input file. A more detailed description of each keyword is presented in ensuing subsections.

Keyword	Mandatory or Optional	Number of times the keyword can be supplied	Action
START_DATE	mandatory	1	Assigns a value
END_DATE	mandatory	1	Assigns a value
NDAY_OUT	mandatory	1	Assigns a value
STEPS_PER_DAY	optional	1	Assigns a value
SILOFILE	mandatory	multiple	A SILO data file is read. Rainfall and evaporation data are stored
RAINFIL	mandatory	multiple	A LUMPREM rainfall file is written based on the contents of the most recently read SILO file; the name of this file is provided with the keyword
EPOTFILE	mandatory	multiple	A LUMPREM potential evaporation file is written based on the contents of the most recently read SILO file; the name of this file is provided with the keyword
VEGFILE	mandatory	multiple	Assigns a name to <i>vegfile</i> , or assigns values to <i>cropfac_all</i> and <i>gamma_all</i>
IRRIGFILE	mandatory	multiple	Assigns a name to <i>irrigfile</i> , or assigns values to <i>irrigcode_all</i> and <i>gwirrigfrac_all</i>
MAXVOL	mandatory	multiple	Assigns a value
IRRIGVOLFRAC	optional	multiple	Assigns a value
RDELAY	optional	multiple	Assigns a value
MDELAY	optional	multiple	Assigns a value
KS	optional	multiple	Assigns a value
M	optional	multiple	Assigns a value
L	optional	multiple	Assigns a value
MFLOWMAX	optional	multiple	Assigns a value
OFFSET	optional	multiple	Assigns a value
FACTOR1	optional	multiple	Assigns a value
FACTOR2	optional	multiple	Assigns a value
POWER	optional	multiple	Assigns a value
SURFACE	optional	multiple	Assigns a value
VOL	optional	multiple	Assigns a value

LUMPREM_MODEL_NAME	mandatory	multiple	LUMPREP writes a LUMPREM input file and corresponding template file
BATCH_FILE_NAME	optional	multiple (but is normally supplied only once)	Assigns a name to a file
PEST_CONTROL_FILE	optional	multiple (but is normally supplied only once)	LUMPREP writes a partial PEST control file and model batch file based on all LUMPREM models that were built through previous LUMPREM_MODEL_NAME keywords

Table 6.1 Keywords supported by LUMPREP.

6.4 LUMPREP Keywords – Details

Each of LUMPREM's keywords is now discussed in detail.

6.4.1 Keywords that are Supplied Only Once

START_DATE

This keyword must be placed near the top of a LUMPREP input file. It provides the starting date of all LUMPREM simulations for which input files are written by LUMPREP. Each simulation starts at midnight just preceding the nominated starting day. The date must be provided in the format *dd/mm/yyyy*. Only a single *START_DATE* keyword can be supplied in a LUMPREP input file.

END_DATE

This keyword must be placed near the top of a LUMPREP input file. It provides the finishing date of all LUMPREM simulations for which input control files are written by LUMPREP. Each simulation finishes at midnight on the nominated day. The date must be provided in the format *dd/mm/yyyy*. Only a single *END_DATE* keyword can be supplied in a LUMPREP input file.

NDAY_OUT

LUMPREP will cease execution with an error message if this keyword is supplied prior to either the *START_DATE* or *END_DATE* keywords.

The value supplied for this keyword must be either a positive integer or the word "monthly". This value is used to calculate the value of the LUMPREM *noutdays* variable, and to fill the elements of the LUMPREM *outday* array. Suppose that the value supplied for this keyword is the positive integer *N*. Then all LUMPREM input files written by LUMPREP during its current run will request that LUMPREM records its calculated outputs after every *N* simulation days. On the other hand, suppose that the value of this keyword is supplied as "monthly". Then the filling of the *outday* array will be such as to request LUMPREM output at the end of each month, and at the end of the LUMPREM simulation if this does not coincide with the end of a month.

STEPS_PER_DAY

This keyword is optional. It can be supplied only once in a LUMPREP input file. It is the value of the *nstep* control variable required in a LUMPREP input file. As presently programmed, the default value for this variable is 1.

As presently programmed, LUMPREP does not allow the user to provide values for the LUMPREM *mxiter* and *tol* control variables. These are set internally to 500 and 1E-5 respectively.

6.4.2 Model-Specific Keywords

General

The keywords that are discussed next can be repeated as many times as desired in a LUMPREP input file. Some of them may be repeated for each LUMPREP input control file that is written by LUMPREP. Others may not be repeated, or may be repeated on fewer occasion than LUMPREP model input files are written; they are thus re-used during the writing of subsequent LUMPREP input datasets. A LUMPREP input dataset is written whenever a LUMPREM_MODEL_NAME keyword is encountered. The details of this input dataset depend on the most recently-supplied values for model-specific keywords.

SILOFILE

Two text strings must follow the name of this keyword on a LUMPREP input file. The first is the name of a SILO file that must be read to obtain rainfall and potential evapotranspiration data. The second is the character string that is used to identify the source of potential evapotranspiration data in this file. Often this string will be “evap” (it is case insensitive). However there may be occasions where a user may prefer the “evsp” or “fao56” options. Meanwhile, daily rainfall is assumed to lie in the column that is headed by the “rain” string.

On encountering the SILOFILE keyword, LUMPREP reads the nominated SILO file. If an error occurs while reading this file, LUMPREP reports this error to the screen and ceases execution. If it does not encounter an error, it stores rainfall and potential evaporation data pertaining to the LUMPREM model simulation time within its memory for future use.

RAINFIL

The name of the daily rainfall input file for a LUMPREM model must be supplied with this keyword. On encountering the RAINFILE keyword, LUMPREP writes the LUMPREM rainfall file based on the contents of its most recently read SILO file.

EPOTFILE

The name of a daily potential evaporation input file for a LUMPREM model must be supplied with this keyword. On encountering the EPOTFILE keyword, LUMPREP writes the LUMPREM potential evaporation file based on the contents of its most recently read SILO file.

VEGFILE

LUMPREP provides the same options as does LUMPREM in providing a value for the *vegfile* control variable.

The first option is to provide the name of a file. It is the user’s responsibility to create and fill this file him/herself; LUMPREP will then use the name of this file as the value of the LUMPREM *vegfile* variable on the next occasion that it writes a LUMPREM input file (that is, on the next occasion that it encounters a LUMPREM_MODEL_NAME keyword).

Alternatively, global values of crop factor and gamma can be provided with this keyword. These are the LUMPREM *crofac_all* and *gamma_all* control variables. If this VEGFILE option is taken, then global crop factor and global gamma are declared as PEST-adjustable parameters. They are given the names *crfac_name* and *gamma_name*, where *name* is the name of the LUMPREM model supplied with the next LUMPREM_MODEL_NAME keyword. Values supplied for crop factor and gamma must both be greater than zero.

IRRIGFILE

LUMPREP provides the same options as does LUMPREM in providing a name for the *irrigfile* control variable.

The first option is to provide the name of a file. It is the user's responsibility to create this file him/herself; LUMPREP will then use the name of this file as the value of the LUMPREM *irrigfile* variable on the next occasion that it writes a LUMPREM input file (that is, on the next occasion that it encounters a LUMPREM_MODEL_NAME keyword).

Alternatively, global values of the irrigation code and groundwater irrigation extraction fraction can be provided with this keyword. These are the LUMPREM *irrigcode_all* and *gwirrigfrac_all* control variables. If this IRRIGFILE option is taken, then the global groundwater irrigation extraction fraction is declared as a PEST parameter. It is given the name *gwirfr_name*, where *name* is the name of the LUMPREM model supplied with the next LUMPREM_MODEL_NAME keyword. The groundwater irrigation extraction fraction parameter is declared to be PEST-adjustable if the global irrigation code is assigned a value of 1; it is declared to be a fixed parameter if the global irrigation code is assigned a value of 0. The value supplied for groundwater irrigation extraction fraction must be greater than zero and less than one.

MAXVOL

Provide the value of the LUMPREM *maxvol* variable. This is declared as a PEST-adjustable parameter with name *maxvol_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Its value must be positive.

IRRIGVOLFRAC

Provide the value of the LUMPREM *irrigvolfrac* variable. If values of *irrigcode_all* and *gwirrigfrac_all* are supplied for the IRRIGFILE keyword, and if *irrigcode_all* is set to 1, then this is declared as a PEST-adjustable parameter with name *irrigvf_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Alternatively, if *irrigcode_all* is set to 0, or if the name of an irrigation file is supplied with the IRRIGFILE keyword, then *irrigvf_name* is declared as a fixed PEST parameter. It is the user's responsibility to declare *irrigvf_name* as adjustable in the PEST control file used for joint calibration of LUMPREP-created LUMPREM models if the irrigation file provided by the user allows irrigation (and if, indeed, the user wishes that a value be estimated for this LUMPREM variable).

The value supplied with the IRRIGVOLFRAC keyword must be greater than zero and less than one.

RDELAY

Provide the value of the LUMPREM *rdelay* variable. This is declared as a PEST-adjustable parameter with name *rdelay_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Its value must be positive. This is an optional keyword whose default value is 5 days.

MDELAY

Provide the value of the LUMPREM *mdelay* variable. This is declared as a PEST-adjustable parameter with name *mdelay_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Its value must be positive. This is an optional keyword whose default value is 1 day.

KS

Provide the value of the LUMPREM *ks* variable. This is declared as a PEST-adjustable parameter with name *ks_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Its value must be positive. This is an optional keyword whose default value is 0.1.

M

Provide the value of the LUMPREM *m* variable. This is declared as a PEST-adjustable parameter with name *m_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Its value must be positive. This is an optional keyword whose default value is 0.5.

L

Provide the value of the LUMPREM *l* variable. This is declared as a PEST-adjustable parameter with name *l_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Its value must be positive. This is an optional keyword whose default value is 0.5.

MFLOWMAX

Provide the value of the LUMPREM *mflowmax* variable. This is declared as a PEST-adjustable parameter with name *mfmax_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. Its value must be positive. This is an optional keyword whose default value is 0.1.

OFFSET

Provide the value of the LUMPREM *offset* variable. This is declared as a PEST parameter with name *offset_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword.

OFFSET is an optional keyword whose default value is 0.0. If OFFSET and POWER are both zero, then the PEST parameters *offset_name*, *fac1_name*, *fac2_name* and *power_name* are all fixed in the partial PEST control file which LUMPREP writes when it encounters a PEST_CONTROL_FILE keyword.

FACTOR1

Provide the value of the LUMPREM *factor1* variable. This is declared as a PEST parameter with name *fac1_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. As presently programmed, its value must be positive.

FACTOR1 is an optional keyword whose default value is 1.0. If OFFSET and POWER are both zero, then the PEST parameters *offset_name*, *fac1_name*, *fac2_name* and *power_name* are all fixed in the partial PEST control file which LUMPREP writes when it encounters a PEST_CONTROL_FILE keyword.

FACTOR2

Provide the value of the LUMPREM *factor2* variable. This is declared as a PEST parameter with name *fac2_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword. As presently programmed, its value must be positive.

FACTOR2 is an optional keyword whose default value is 1.0. If OFFSET and POWER are both zero, then the PEST parameters *offset_name*, *fac1_name*, *fac2_name* and *power_name* are all fixed in the partial PEST control file which LUMPREP writes when it encounters a PEST_CONTROL_FILE keyword.

POWER

Provide the value of the LUMPREM *power* variable. This is declared as a PEST parameter with name *power_name*, where *name* is the name of the LUMPREM model supplied with the ensuing LUMPREM_MODEL_NAME keyword.

POWER is an optional keyword whose default value is 0.0. If OFFSET and POWER are both zero, then the PEST parameters *offset_name*, *fac1_name*, *fac2_name* and *power_name* are all fixed in the partial PEST control file which LUMPREP writes when it encounters a PEST_CONTROL_FILE keyword.

SURFACE

Provide the elevation of the topographic surface. This is s of equation 2.4. This is an optional keyword whose default value is 100.0.

VOL

Provide the value of the LUMPREP *vol* variable. This is not declared as an adjustable parameter by LUMPREP. Its default value is a half of *mflowmax*.

LUMPREP_MODEL_NAME

The value supplied for this keyword must be a text string of three characters or less in length. This name is used as a suffix for PEST-adjustable parameters associated with this particular LUMPREP model instance. It is also used as a suffix for the filename base of the LUMPREP model input file and corresponding template file, both of which are written by LUMPREP when it encounters this keyword. These files are named *lr_name.in* and *lr_name.tpl*, where *name* is the value ascribed to the LUMPREP_MODEL_NAME keyword.

Details of the LUMPREP input file (and corresponding template file) written by LUMPREP depend on the most-recently supplied values for model-specific keywords provided on the LUMPREP input file. Note the following in particular.

- If values of crop factor and gamma were supplied with the last-encountered VEGFILE keyword, these values are passed to the LUMPREP input file instead of the name of a vegetation file. Alternatively, if a filename was supplied with the most recent VEGFILE keyword, then the name of this file is recorded on the LUMPREP input file. In the former case, *crfac_name* and *gamma_name* are declared as adjustable parameters. (The former is the global crop factor for the current LUMPREP model instance while the latter is the global gamma value.) These parameter names are featured in the *lr_name.tpl* template file which LUMPREP writes, and in the partial PEST control file which LUMPREP writes on encountering the PEST_CONTROL_FILE keyword. On the other hand, if a filename is supplied with the VEGFILE keyword, then these parameters are not featured in either the template file for this LUMPREP model (because there is no place to write them), nor in the partial PEST control file. If a user wishes to introduce, and adjust, crop factors and gammas that pertain to his/her own *irrigfile* file, then he/she must take responsibility for this aspect of PEST input dataset construction him/herself.
- If values for the irrigation code and groundwater irrigation extraction fraction were supplied with the last-encountered IRRIGFILE keyword, these values are passed to the LUMPREP input file instead of the name of an irrigation file. Alternatively, if a filename was supplied with the last-encountered IRRIGFILE keyword, then the name of this file is recorded on the LUMPREP input file which LUMPREP writes. In the former case, *gwirfr_name* is declared as a parameter; it is adjustable if the global irrigation code is set to 1, but fixed otherwise (its value then becomes redundant). The *gwirfr_name* parameter is featured in the *lr_name.tpl* template file which LUMPREP writes, and in the partial PEST control file which LUMPREP writes when it encounters a PEST_CONTROL_FILE keyword. On the other hand, if a filename was supplied with the last-encountered IRRIGFILE keyword, then the *gwirfr_name* parameter is not featured in either the template file (because there is no place to write it), nor in the partial PEST control file which LUMPREP writes. If a user wishes to adjust irrigation fractions which he/she supplies in his/her own *irrigfile* file, then he/she must take responsibility for this aspect of PEST input dataset construction him/herself.
- If a global irrigation code and global groundwater irrigation fraction are supplied with the IRRIGFILE keyword, and if the global irrigation code is zero, then the *irrigvf_name* parameter (i.e. the value of the LUMPREP *irrigvolfrac* variable) for the current LUMPREP model instance is declared as fixed. Under these circumstances this variable has no meaning. Nevertheless, a value for this variable must be recorded on a LUMPREP input file. LUMPREP thus records a

corresponding parameter space in the template file for the current LUMPREM input file, and cites the parameter in the partial PEST control file which it writes when it encounters a PEST_CONTROL_FILE keyword.

- The names of the *rainfile* and *epotfile* filenames recorded on the current LUMPREM input file are those written by LUMPREP on the last occasions that it encountered RAINFILE and EPOTFILE keywords.
- If *offset* and *power* are both supplied as zero (these are their default values if OFFSET and POWER keywords are not provided in a LUMPREM input file), then all of the *offset_name*, *fac1_name*, *fac2_name* and *power_name* parameters are declared as fixed in the partial PEST control file which LUMPREP writes when it encounters a PEST_CONTROL_FILE keyword.

LUMPREP will cease execution with an appropriate error message if any of the following keywords do not precede a LUMPREM_MODEL_NAME keyword.

- START_DATE
- END_DATE
- NDAY_OUT
- MAXVOL
- VEGFILE
- RAINFILE
- EPOTFILE
- IRRIGFILE

6.4.3 Writing a Partial PEST Control File

General

Once it has written one or a number of LUMPREM input files and corresponding template files, LUMPREP can be asked to write as much of a PEST control file as it can, based on parameters associated with these LUMPREM models. The LUMPREM input file requires two keywords to complete this action, these being the BATCH_FILE_NAME and PEST_CONTROL_FILE keywords. Both are optional. If provided, the BATCH_FILE_NAME keyword should be the second last keyword in a LUMPREP control file, while the PEST_CONTROL_FILE keyword should be the last.

These keywords are now discussed in detail

BATCH_FILE_NAME

Provide the name of the batch file which will be cited in the PEST control file which LUMPREP writes when it encounters the PEST_CONTROL_FILE keyword. This batch file will contain the following:

- commands to delete the output files of all LUMPREM models for which LUMPREP has constructed input datasets;
- commands to run LUMPREM on the basis of each of the LUMPREM input files that LUMPREP has written.

The commands to run LUMPREM employ the LUMPREM command line option for supplying the names of its input and output files. Output files are named *lr_name.out* where *name* is the pertinent LUMPREM_MODEL_NAME.

BATCH_FILE_NAME is an optional keyword. However it is mandatory if a PEST_CONTROL_FILE keyword is supplied. In that case it must be provided before the latter keyword.

PEST_CONTROL_FILE

Provide the name of a PEST control file which LUMPREP must write. LUMPREP writes the following sections of this file:

-
- control data
 - singular value decomposition
 - parameter groups
 - parameter data
 - model input/output
 - model command line

It also writes the batch file whose name was provided with the BATCH_FILE_NAME keyword.

Normally, PEST is used to calibrate a composite model that is comprised of one or a number of LUMPREM models that are run prior to a groundwater model. Hence both the partial PEST control file and the batch file which are written by LUMPREP will require supplementation by the user before composite model calibration is undertaken.

7. References

Mualem, Y., 1976. A new model for predicting the hydraulic conductivity of unsaturated porous media. *Water Resources Research*. 12:513-522.

van Genuchten, M, Th. 1980. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil Sci. Soc. Am. J.* 44:892-898.