Efcarson

Simulation, Model, and Parameter Documentation

Efcarson: Simulation, Model, and Parameter Documentation

Publication date Tue, Jun 8 2010 13:50:12 MDT

Table of Contents

List of Figures

List of Equations

Chapter 1. Model Component

1. Component 'PrmsDdJh'

Name
`model.PrmsDdJh`

1.1. Parameter

adjmix_rain - `double[]`
Adjustment factor for rain in a rain/snow mix Monthly factor to adjust rain proportion in a mixed rain/snow event

albset_rna - `double`
Albedo reset - rain, accumulation stage Proportion of rain in a rain-snow precipitation event above which the snow albedo is not reset. Applied during the snowpack accumulation stage.

albset_rnm - `double`
Albedo reset - rain, melt stage Proportion of rain in a rain-snow precipitation event above which the snow albedo is not reset. Applied during the snowpack melt stage

albset_sna - `double`
Albedo reset - snow, accumulation stage Minimum snowfall, in water equivalent, needed to reset snow albedo during the snowpack accumulation stage

albset_snm - `double`
Albedo reset - snow, melt stage Minimum snowfall, in water equivalent, needed to reset snow albedo during the snowpack melt stage

basin_area - `double`
Total basin area

basin_solsta - `int`
Index of main solar radiation station Index of solar radiation station used to compute basin radiation values

basin_tsta - `int`
Index of main temperature station Index of temperature station used to compute basin temperature values

carea_max - `double[]`
Maximum contributing area Maximum possible area contributing to surface runoff expressed as a portion of the HRU area

cecn_coef - `double[]`
Convection condensation energy coefficient, varied monthly

cov_type - `int[]`
Vegetation cover type designation for HRU (0=bare soil; 1=grasses; 2=shrubs; 3=trees)

covden_sum - `double[]`
Summer vegetation cover density for the major vegetation type on each HRU. [intcp]

covden_win - `double[]`
Winter vegetation cover density for the major vegetation type on each HRU

dday_intcp - `double[]`
Intercept in temperature degree-day relationship Intercept in relationship: dd-coef = dday_intcp + dday_slope*(tmax)+1.

dday_slope - `double[]`
Slope in temperature degree-day relationship Coefficient in relationship: dd-coef = dday_intcp + dday_slope*(tmax)+1.

den_init - `double`
Initial density of new-fallen snow

den_max - `double`
Average maximum snowpack density

dprst_flag - `int`
Selection flag for depression storage computation. 0=No; 1=Yes

emis_noppt - `double`
Average emissivity of air on days without precipitation

endTime - `Calendar`
Ending date of the simulation.

epan_coef - `double[]`
Evaporation pan coefficient Evaporation pan coefficient

freeh2o_cap - `double`
Free-water holding capacity of snowpack expressed as a decimal fraction of the frozen water content of the snowpack (pk_ice)

frozen - `int[]`
Flag for frozen ground (0=no; 1=yes)

glacier_flag - `int`
Flag incicating presence of a glacier, (0=no; 1=yes)

groundmelt - `double[]`
Amount of snowpack-water that melts each day to soils

gwflow_coef - `double[]`
Groundwater routing coefficient Groundwater routing coefficient - is multiplied by the storage in the groundwater reservoir to compute groundwater flow contribution to down-slope flow

gwsink_coef - `double[]`
Groundwater sink coefficient Groundwater sink coefficient - is multiplied by the storage in the groundwater reservoir to compute the seepage from each reservoir to the groundwater sink

gwstor_init - `double[]`
Initial storage in each gw reservoir Storage in each groundwater reservoir at the beginning of a simulation

hru_area - `double[]`
HRU area

hru_deplcrv - `int[]`
Index number for the snowpack areal depletion curve associated with an HRU

hru_elev - `double[]`
Mean elevation for each HRU

hru_gwres - `int[]`
Index of groundwater reservoir assigned to HRU Index of groundwater reservoir receiving excess soil water from each HRU

hru_percent_dprst - `double[]`
HRU depression storage area as a decimal percent of the total HRU area

hru_percent_imperv - `double[]`
Proportion of each HRU area that is impervious

hru_psta - `int[]`
Index of the base precipitation station used for lapse rate calculations for each HRU.

hru_radpl - `int[]`
Index of radiation plane for HRU Index of radiation plane used to compute solar radiation for each HRU

hru_solsta - `int[]`
Index of solar radiation station associated with each HRU

hru_ssres - `int[]`
Index of subsurface reservoir receiving excess water from HRU soil zone

hru_tsta - `int[]`
Index of the base temperature station used for lapse rate calculations

hru_type - `int[]`
Type of each HRU (0=inactive; 1=land; 2=lake; 3=swale)

imperv_stor_max - `double[]`
HRU maximum impervious area retention storage Maximum impervious area retention storage for each HRU

inputFile - `File`

jh_coef - `double[]`
Monthly air temperature coefficient used in Jensen -Haise potential evapotranspiration computations, see PRMS manual for calculation method

jh_coef_hru - `double[]`
Jensen-Haise Air temperature coefficient used in Jensen-Haise potential evapotranspiration computations for each HRU. See PRMS manual for calculation method

melt_force - `int`
Julian date to force snowpack to spring snowmelt stage; varies with region depending on length of time that permanent snowpack exists

melt_look - `int`
Julian date to start looking for spring snowmelt stage. Varies with region depending on length of time that permanent snowpack exists

ndays - `int`
Number of HRUs.

ndepl - `int`
Number of snow cover depletion curves.

ndeplval - `int`
Number of values in each snow cover depletion curve.

nevap - `int`
Number of evaporation pan stations.

ngw - `int`
Number of Ground water reservoirs.

nhru - `int`
Number of HRUs.

nobs - `int`
Number of streamflow (runoff) measurement stations.

nradpl - `int`
Number of radiation planes.

nrain - `int`
Number of precipitation stations.

nsol - `int`
Number of solar radiation stations.

nssr - `int`
Number of subsurface reservoirs.

nstorm - `int`
Number of storms.

ntemp - `int`
Number of temperature stations.

objfunc_q - `int`
Index of the runoff station used as the measured runoff variable in the objective function calculation

outFile - `File`

potet_sublim - `double`
Proportion of potential ET that is sublimated from the snow surface

ppt_rad_adj - `double[]`
Radiation reduced if basin precip above this value If basin precip exceeds this value, radiation is mutiplied by summer or winter precip adjustment

precip_units - `int`
Units for measured precipitation Units for measured precipitation (0=inches; 1=mm)

print_freq - `int`
Frequency for output data file (0=none; 1=run totals; 2=yearly; 4=monthly; 8=daily; or additive combinations) For combinations, add index numbers, e.g., daily plus yearly = 10; yearly plus total = 3

print_objfunc - `int`
Switch to turn objective function printing off and on (0=no; 1=yes)

print_type - `int`
Type of output data file (0=measured and simulated flow only; 1=water balance table; 2=detailed output)

rad_conv - `double`
Conversion factor to langleys for measured radiation Conversion factor to langleys for measured radiation

rad_trncf - `double[]`
Transmission coefficient for short-wave radiation through the winter vegetation canopy

radadj_intcp - `double`
Intercept in temperature range adjustment to solar radiation Intercept in equation: adj = radadj_intcp + radadj_slope*(tmax-tmax_index)

radadj_slope - `double`
Slope in temperature range adjustment to solar radiation Slope in equation: adj = radadj_intcp + radadj_slope * (tmax - tmax_index)

radj_sppt - `double`
Adjustment factor for computed solar radiation for summer day with greater than ppt_rad_adj inches precip

radj_wppt - `double`
Adjustment factor for computed solar radiation for winter day with greater than ppt_rad_adj inches precip

radmax - `double`
The maximum portion of the potential solar radiation that may reach the ground due to haze, dust, smog, etc.

radpl_aspect - `double[]`
Aspect for each radiation plane

radpl_lat - `double[]`
Latitude of each radiation plane

radpl_slope - `double[]`
Slope of each radiation plane, specified as change in vertical length divided by change in horizontal length

rain_adj - `double[][]`
Monthly factor to adjust measured precipitation on each HRU to account for differences in elevation, etc

rain_code - `int[]`
Code indicating rule for precip station use (1=only precip if the regression stations have precip; 2=only precip if any station in the basin has precip; 3=precip if xyz says so; 4=only precip if rain_day variable is set to 1; 5=only precip if psta_freq_nuse stations see precip)

runoff_units - `int`
Measured runoff units (0=cfs; 1=cms)

settle_const - `double`
Snowpack settlement time constant

smidx_coef - `double[]`
Coefficient in contributing area computations Coefficient in non-linear contributing area algorithm. Equation used is: contributing area = smidx_coef * 10.**(smidx_exp*smidx) where smidx is soil_moist + .5 * ppt_net

smidx_exp - `double[]`
Exponent in contributing area computations Exponent in non-linear contributing area algorithm. Equation used is: contributing area = smidx_coef * 10.**(smidx_exp*smidx) where smidx is soil_moist + .5 * ppt_net

snarea_curve - `double[][]`
Snow area depletion curve values, 11 values for each curve (0.0 to 1.0 in 0.1 increments)

snarea_thresh - `double[]`
Maximum threshold water equivalent for snow depletion The maximum threshold snowpack water equivalent below which the snow-covered-area curve is applied. Varies with elevation.

snow_adj - `double[][]`
Monthly factor to adjust measured precipitation on each HRU to account for differences in elevation, etc

snow_intcp - `double[]`
Snow interception storage capacity for the major vegetation type in each HRU

snowinfil_max - `double[]`
Maximum snow infiltration per day Maximum snow infiltration per day

soil2gw_max - `double[]`
The maximum amount of the soil water excess for an HRU that is routed directly to the associated groundwater reservoir each day

soil_moist_init - `double[]`
Initial value of available water in soil profile

soil_moist_max - `double[]`
Maximum available water holding capacity of soil profile. Soil profile is surface to bottom of rooting zone

soil_rechr_init - `double[]`
Initial value for soil recharge zone (upper part of soil_moist). Must be less than or equal to soil_moist_init

soil_rechr_max - `double[]`
Maximum value for soil recharge zone (upper portion of soil_moist where losses occur as both evaporation and transpiration). Must be less than or equal to soil_moist

soil_type - `int[]`
HRU soil type (1=sand; 2=loam; 3=clay)

srain_intcp - `double[]`
Summer rain interception storage capacity for the major vegetation type in each HRU

ssr2gw_exp - `double[]`
Coefficient in equation used to route water from the subsurface reservoirs to the groundwater reservoirs: ssr_to_gw = ssr2gw_rate * ((ssres_stor / ssrmax_coef)**ssr2gw_exp); recommended value is 1.0

ssr2gw_rate - `double[]`
Coefficient to route water from subsurface to groundwater Coefficient in equation used to route water from the subsurface reservoirs to the groundwater reservoirs: ssr_to_gw = ssr2gw_rate * ((ssres_stor / ssrmax_coef)**ssr2gw_exp)

ssr_gwres - `int[]`
Index of gw reservoir to receive flow from ss reservoir Index of the groundwater reservoir that will receive flow from each subsurface or gravity reservoir

ssrcoef_lin - `double[]`
Coefficient to route subsurface storage to streamflow using the following equation: ssres_flow = ssrcoef_lin * ssres_stor + ssrcoef_sq * ssres_stor**2

ssrcoef_sq - `double[]`
Coefficient to route subsurface storage to streamflow using the following equation: ssres_flow = ssrcoef_lin * ssres_stor + ssrcoef_sq * ssres_stor**2

ssrmax_coef - `double[]`
Coefficient in equation used to route water from the subsurface reservoirs to the groundwater reservoirs: ssr_to_gw = ssr2gw_rate * ((ssres_stor / ssrmax_coef)**ssr2gw_exp); recommended value is 1.0

ssstor_init - `double[]`
Initial storage in each subsurface reservoir; estimated based on measured flow

startTime - `Calendar`
Starting date of the simulation.

storm_scale_factor - `double[]`
Adjustment factor for each storm

strain_adj - `double[][]`
Monthly factor to adjust measured precipitation to each HRU to account for differences in elevation, etc. This factor is for the rain gage used for kinematic or storm routing

sumFile - `File`
Summary file name for user selected summary output.

temp_units - `int`
Units for measured temperature (0=Fahrenheit; 1=Celsius)

tmax_adj - `double[]`
Adjustment to maximum temperature for each HRU, estimated based on slope and aspect

tmax_allrain - `double[]`
If maximum temperature of an HRU is greater than or equal to this value (for each month, January to December), precipitation is assumed to be rain

tmax_allsnow - `double`
If maximum temperature of an HRU is less than or equal to this value, precipitation is assumed to be snow

tmax_index - `double[]`
Monthly index temperature Index temperature used to determine precipitation adjustments to solar radiation, deg F or C depending on units of data

tmax_lapse - `double[]`
Array of twelve values representing the change in maximum temperature per 1000 elev_units of elevation change for each month, January to December

tmin_adj - `double[]`
Adjustment to minimum temperature for each HRU, estimated based on slope and aspect

tmin_lapse - `double[]`
Array of twelve values representing the change in minimum temperture per 1000 elev_units of elevation change for each month, January to December

transp_beg - `int[]`
Month to begin summing tmaxf for each HRU; when sum is >= to transp_tmax, transpiration begins

transp_end - `int[]`
Month to stop transpiration computations; transpiration is computed thru end of previous month

transp_tmax - `double[]`
Temperature index to determine the specific date of the start of the transpiration period. Subroutine sums tmax for each HRU starting with the first day of month transp_beg. When the sum exceeds this index, transpiration begins

tsta_elev - `double[]`
Elevation of each temperature measurement station

tstorm_mo - `double[]`

Monthly indicator for prevalent storm type (0=frontal storms prevalent; 1=convective storms prevalent)

wrain_intcp - `double[]`

Winter rain interception storage capacity for the major vegetation type in the HRU

Chapter 2. Sub Component

1. 'prms2008'

1.1. Component 'Basin'

Basin setup. Check for validity of basin parameters and compute reservoir areas.

Name
: `prms2008.Basin`

Author
: George H. Leavesley - ghleavesley@colostate.edu

Keyword
: Routing

Version
: $Id: Basin.java 861 2010-01-21 01:54:38Z ghleavesley $

Source
: $URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Basin.java $

License
: http://www.gnu.org/licenses/gpl-2.0.html

1.1.1. Parameter

basin_area `[acres] - double`
: Total basin area

dprst_flag `- int`
: Selection flag for depression storage computation. 0=No; 1=Yes

dprst_pct_open `[decimal fraction] - double[]`
: Decimal fraction of depression storage area that can flow to a stream channel. Amount of flow is a function of storage.

hru_area `[acres] - double[]`
: Area of each HRU

hru_gwres `- int[]`
: Index of groundwater reservoir receiving excess soil water from each HRU

hru_percent_dprst `[decimal fraction] - double[]`
: HRU depression storage area as a decimal percent of the total HRU area

hru_percent_imperv `[decimal fraction] - double[]`
: Proportion of each HRU area that is impervious

hru_ssres `- int[]`
: Index of subsurface reservoir receiving excess water from HRU soil zone

hru_type `- int[]`
: Type of each HRU (0=inactive; 1=land; 2=lake; 3=swale)

ngw - `int`
Number of Ground water reservoirs.

nhru - `int`
Number of HRUs.

nssr - `int`
Number of subsurface reservoirs.

1.1.2. Variables (Out)

active_gwrs - `int`
Number of active GWRs

active_hrus - `int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

dem_dprst `[acres] - double[]`
HRU depression storage area defined by DEM

dprst_clos `[acres] - double[]`
HRU depression storage area that is closed and can only spill

dprst_open `[acres] - double[]`
HRU depression storage area that can flow to a stream

gwr_route_order - `int[]`
Routing order for ground-water reservoirs

gwres_area `[acres] - double[]`
Area of each groundwater reservoir. Computed by summing areas of HRUs that contribute to it

hru_dprst `[acres] - double[]`
HRU depression storage area

hru_imperv `[acres] - double[]`
Impervious area of each HRU

hru_percent_impv `[decimal fraction] - double[]`
Proportion of each HRU area that is impervious

hru_percent_perv `[decimal fraction] - double[]`
Proportion of each HRU area that is pervious

hru_perv `[acres] - double[]`
Pervious area of each HRU

hru_route_order - `int[]`
Routing order for HRUs

land_area `[acres] - double`
Basin area composed of land.

ssres_area `[acres] - double[]`
Area of each subsurface reservoir; computed by summing areas of HRUs that contribute to it

water_area `[acres] - double`
Basin area composed of water bodies

1.1.3. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.1.4. Basin

1.1.4.1. Description

All computations for this component are done in the initialize method. There is no execute method. The HRU pervious and impervious areas are computed using the parameters `hru_percent_imperv` and `hru_area`. Subsurface and groundwater reservoir areas are computed by summing the areas of the HRU's that contribute to them.

1.1.4.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.2. Component 'BasinSum'

Basin summary.

Name
`prms2008.BasinSum`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Summary

Version
$Id: BasinSum.java 367 2009-08-28 22:21:52Z odavid $

Source
$HeadURL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/BasinSum.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.2.1. Parameter

endTime `[yyyy-mm-dd] - Calendar`
Ending date of the simulation.

nhru `- int`
Number of HRUs.

nobs `- int`
Number of streamflow (runoff) measurement stations.

objfunc_q - `int`
Index of the runoff station used as the measured runoff variable in the objective function calculation

print_freq - `int`
Frequency for output data file (0=none; 1=run totals; 2=yearly; 4=monthly; 8=daily; or additive combinations) For combinations, add index numbers, e.g., daily plus yearly = 10; yearly plus total = 3

print_objfunc - `int`
Switch to turn objective function printing off and on (0=no; 1=yes)

print_type - `int`
Type of output data file (0=measured and simulated flow only; 1=water balance table; 2=detailed output)

runoff_units - `int`
Measured runoff units (0=cfs; 1=cms)

startTime `[yyyy-mm-dd] - Calendar`
Starting date of the simulation.

sumFile - `File`
Summary file name for user selected summary output.

1.2.2. Variables (In)

active_hrus - `int`
Number of active HRUs

basin_actet `[inches] - double`
Weighted average actual evapotranspiration for the basin. [smba]

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

basin_cfs `[cfs] - double`
Total streamflow for the basin.

basin_gwflow `[inches] - double`
Weighted average groundwater contribution to streamflow for the basin. [gwflow]

basin_gwsink `[inches] - double`
Basin area weighted average of groundwater reservoir storage to the groundwater sink

basin_gwstor `[inches] - double`
Weighted average groundwater storage for the basin. [gwflow]

basin_imperv_evap `[inches] - double`
Basin area-weighted average for evaporation from impervious area

basin_imperv_stor `[inches] - double`
Basin area-weighted average for storage on impervious area

basin_intcp_evap `[inches] - double`
Weighted average basin evaporation and sublimation loss from interception storage. [intcp]

basin_intcp_stor `[inches] - double`
Weighted average interception storage for the basin. [intcp]

basin_lakeevap `[inches] - double`
Basin area weighted average of lake evaporation

basin_net_ppt `[inches] - double`
Weighted average net precipitation for the basin. [intcp]

basin_perv_et `[inches] - double`
Basin area weighted average of pervious area ET

basin_potet `[inches] - double`
Weighted average potential evapotranspiration for basin. [potet]

basin_ppt `[inches] - double`
Average basin precipitation. [precip]

basin_pweqv `[inches] - double`
Average snowpack water equivalent for total basin area. [snow]

basin_snowevap `[inches] - double`
Average evaporation and sublimation for total basin area. [snow]

basin_snowmelt `[inches] - double`
Average snowmelt for total basin area. [snow]

basin_soil_moist `[inches] - double[]`
Weighted average soil moisture content for the basin. [smbal]

basin_sroff `[inches] - double`
Weighted average surface runoff for the basin. [srunoff]

basin_ssflow `[inches] - double`
Weighted average of contribution to streamflow from subsurface reservoirs for the basin. [ssflow]

basin_ssstor `[inches] - double`
Weighted average of storage in subsurface reservoirs for the basin. [ssflow]

basin_stflow `[inches] - double`
The sum of basin_sroff, basin_gwflow and basin_ssflow.

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

deltim `[hours] - double`
Length of the time step

hru_actet `[inches] - double[]`
Actual evapotranspiration on HRU, pervious + impervious

hru_route_order `- int[]`
Routing order for HRUs

last_intcp_stor `[inches] - double`
Basin area-weighted average changeover interception

orad `[langleys] - double`
Measured or computed solar radiation on a horizontal surface

route_on - `int`
Kinematic routing switch (0=daily; 1=storm period)

runoff `[cfs] - double[]`
Observed runoff for basin. [obs]

solrad_tmax `[degrees] - double`
Basin daily maximum temperature for use with solrad radiation

solrad_tmin `[degrees] - double`
Basin daily minimum temperature for use with solrad radiation

tmaxf `[degrees F] - double[]`
HRU adjusted daily maximum temperature

tminf `[degrees F] - double[]`
HRU adjusted daily minimum temperature

1.2.3. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.2.4. BasinSum

1.2.4.1. Description

This component writes a summary of pre-defined PRMS simulation variables at a daily time step. There are three types of summaries available. The first is a listing of the observed and predicted streamflow only. The second provides a table with values that will allow water balance computations and includes the basin-weighted averages for net precipitation, evapotranspiration from all sources, storage in all reservoirs and the predicted and observed streamflows. The third is a detailed summary of the rainfall, outflow and state variables.

Any of the summaries may be requested in any combination of the available time increments whic are daily, monthly, yearly or total for the run.

1.2.4.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.3. Component 'Ddsolrad'

Solar radiation distribution algorithm and estimation procedure for missing radiation data.Procedures for distributing solar radiation to each HRU and for estimating missing solar radiation data using a maximum temperature / degree-day relationship.

Name
`prms2008.Ddsolrad`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Radiation

Version
$Id: Ddsolrad.java 1128 2010-04-07 19:43:29Z ghleavesley $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Ddsolrad.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.3.1. Parameter

basin_solsta - `int`
Index of main solar radiation station Index of solar radiation station used to compute basin radiation values

dday_intcp `[dday] - double[]`
Intercept in temperature degree-day relationship Intercept in relationship: dd-coef = dday_intcp + dday_slope*(tmax)+1.

dday_slope `[dday/degree] - double[]`
Slope in temperature degree-day relationship Coefficient in relationship: dd-coef = dday_intcp + dday_slope*(tmax)+1.

hru_area `[acres] - double[]`
HRU area , Area of each HRU

hru_radpl - `int[]`
Index of radiation plane for HRU Index of radiation plane used to compute solar radiation for each HRU

hru_solsta - `int[]`
Index of solar radiation station associated with each HRU

nhru - `int`
Number of HRUs.

nradpl - `int`
Number of radiation planes.

nsol - `int`
Number of solar radiation stations.

ppt_rad_adj `[inches] - double[]`
Radiation reduced if basin precip above this value If basin precip exceeds this value, radiation is mutiplied by summer or winter precip adjustment

rad_conv - `double`
Conversion factor to langleys for measured radiation Conversion factor to langleys for measured radiation

radadj_intcp `[dday] - double`
Intercept in temperature range adjustment to solar radiation Intercept in equation: adj = radadj_intcp + radadj_slope*(tmax-tmax_index)

radadj_slope `[dday/degree] - double`
Slope in temperature range adjustment to solar radiation Slope in equation: adj = radadj_intcp + radadj_slope * (tmax - tmax_index)

radj_sppt `[decimal fraction] - double`
Adjustment factor for computed solar radiation for summer day with greater than ppt_rad_adj inches precip

radj_wppt `[decimal fraction] - double`
Adjustment factor for computed solar radiation for winter day with greater than ppt_rad_adj inches precip

radmax `[decimal fraction] - double`
The maximum portion of the potential solar radiation that may reach the ground due to haze, dust, smog, etc.

tmax_allrain `[degrees] - double[]`
If maximum temperature of an HRU is greater than or equal to this value (for each month, January to December), precipitation is assumed to be rain

tmax_index `[degrees] - double[]`
Monthly index temperature Index temperature used to determine precipitation adjustments to solar radiation, deg F or C depending on units of data

1.3.2. Variables (In)

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

basin_obs_ppt `[inches] - double`
Area-weighted measured average precipitation for basin. [precip]

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

hemisphere `- int`
Flag to indicate in which hemisphere the model resides (0=Northern; 1=Southern)

hru_route_order `- int[]`
Routing order for HRUs

newday `- int`
Switch signifying the start of a new day (0=no; 1=yes)

radpl_cossl `- double[]`
Cosine of the radiation plane slope [soltab]

radpl_soltab `[langleys] - double[][]`
Potential shortwave radiation for each radiation plane for each timestep [soltab]

solrad `[langleys] - double[]`
Observed solar radiation [obs]

solrad_tmax `[degrees F] - double`
Basin daily maximum temperature adjusted to elevation of solar radiation station

1.3.3. Variables (Out)

basin_horad `[langleys] - double`
Potential shortwave radiation for the basin centroid

basin_potsw `[langleys] - double`
Area-weighted average of potential shortwave radiation for the basin

orad `[langleys] - double`
Measured or computed solar radiation on a horizontal surface

swrad `[langleys] - double[]`
Computed shortwave radiation for each HRU

1.3.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.3.5. Ddsolrad

1.3.5.1. Description

This component has two functions. One is to distribute measured or estimated solar radiation values for a horizontal surface to the slope and aspect combination of each HRU. The second is to estimate missing solar radiation data.

Observed daily shortwave radiation (`solrad`) expressed in langleys per day (ly/d) can be input directly or estimated from daily air-temperature and precipitation data for watersheds where it is not available. `solrad`, measured on a horizontal surface, is adjusted to estimate `swrad`, the daily shortwave radiation received on the slope-aspect combination of each HRU. `swrad` is computed by:

$$\mathrm{swrad} = \left(\mathrm{solrad} \times \frac{\mathrm{radpl\_potsw}}{\mathrm{horad}}\right) \Big/ \mathrm{radpl\_cossl}$$

where

radpl_potsw
the daily potential solar radiation for the slope and aspect of a radiation plane (ly),

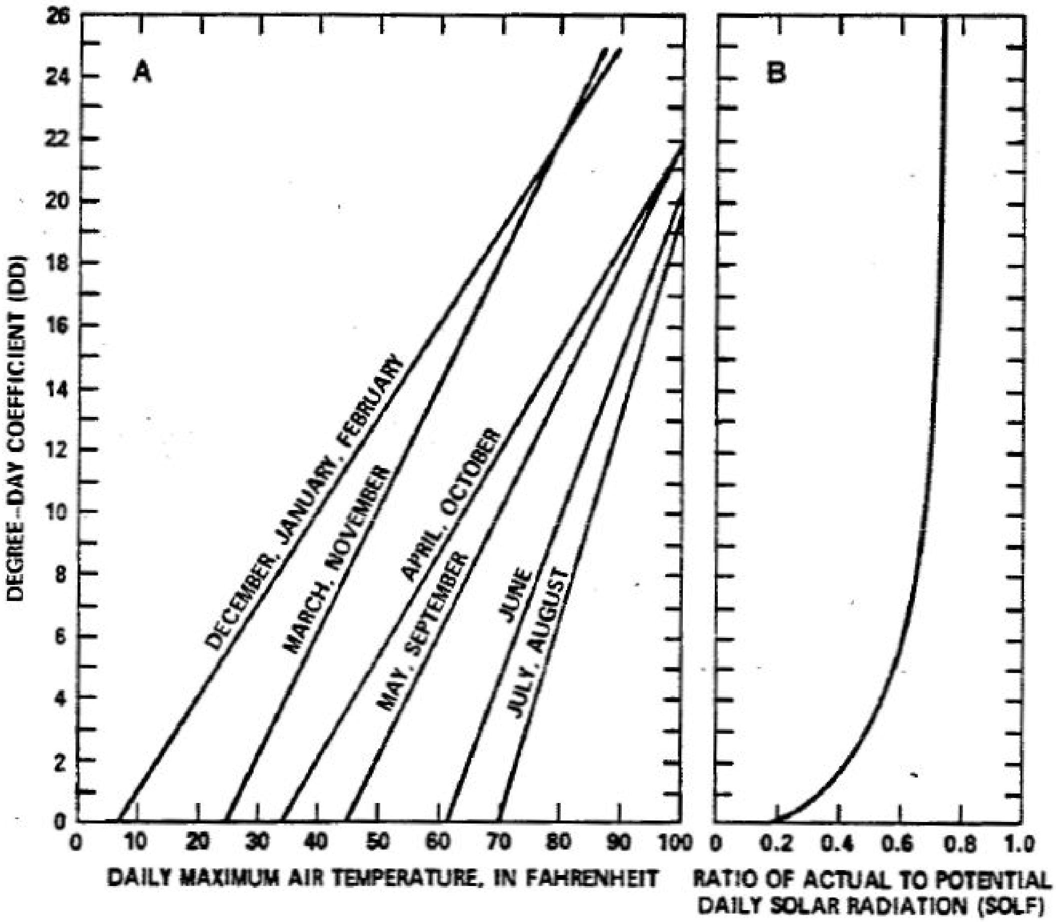
horad
daily potential solar radiation for a horizontal surface (ly), and

radpl_cossl
the cosine of the radiation plane slope

Tables consisting of daily estimates of the potential (clear sky) short-wave solar radiation for each radiation plane (`radpl_potsw`) are computed on the basis of hours between sunrise and sunset for each Julian day of the year in module soltab_prms. The potential short-wave solar radiation also is computed for each Julian day of the year for a horizontal plane at the centroid of the modeled basin (`horad`). daily values of `radpl_potsw` and `horad` are calculated using a combination of methods described in Meeus (1999), Lee (1963), and Swift (1976).

For missing days or periods of record, `solrad` can be estimated using an air temperature degree-day relationship described by Leaf and Brink (1973). This method was developed for a section of the Rocky Mountain Region of the United States. It appears most applicable to regions where predominantly clear skies prevail on days without precipitation. The method is shown graphically in the coaxial relation of Figure 2.1, "Example of coaxial relationship for estimating shortwave solar radiation from maximum daily air temperature developed for northwestern Colorado.". A daily maximum temperature is entered in the X-axis of part A and intersects the appropriate month curve. From this intersection point, one moves horizontally across the degree-day coefficient axis and intersects the curve in part B. From this point, the ratio of actual-to-potential radiation for a horizontal surface (`solf`) can be obtained.

Figure 2.1. Example of coaxial relationship for estimating shortwave solar radiation from maximum daily air temperature developed for northwestern Colorado.



An estimate of solrad is then computed by:

$$solrad = solf \times \ horad$$

The ratio `solf` is developed for days without precipitation; thus, the computed `solrad` is for dry days. `solrad` for days with precipitation is computed by multiplying the dry day `solrad` times a precipitation correction factor `ppt_adj`. `ppt_adj` is determined based on the maximum air temperature (`tmax`) measured at the basin index temperature station (`basin_tsta`) on the day with precipitation and the current month. If `tmax` is greater than or equal to the monthly parameter `tmax_index`, then `ppt_adj` is computed by:

$$ppt\_adj = (radadj\_slope \times tdif) + radadj\_intcp$$

where tdif is the difference between `tmax` and `tmax_index`.

If `tmax` is less than `tmax_index` then `ppt_adj` is set equal to a user-defined constant `radj_wppt` for the period October through April or `radj_sppt` for the period May through September. The use of `tmax_index` is an attempt to distinguish between days where precipitation is convective in origin and days where precipitation is frontal in origin. Days with typically short convective storms may have more solar radiation than those days with frontal storms. The assumption is that for each month a maximum temperature threshold value can be used to distinguish between these storm types.

The input parameters required to use this procedure are the slope (`dday_slope`) and the y-intercept (`dday_intcp`) of the line that expresses the relationship between monthly maximum air temperature and a degree-day coefficient (dd) (Figure 2.1, “Example of coaxial relationship for estimating shortwave solar radiation from maximum daily air temperature developed for northwestern Colorado.”). dd is computed by:

$$dd = (dday\_slope \times tmax) + dday\_intcp$$

where tmax is the observed daily maximum air temperature. The dd-solf relationship of Figure 2.1, “Example of coaxial relationship for estimating shortwave solar radiation from maximum daily air temperature developed for northwestern Colorado.” is assumed constant.

Monthly values of dday_slope and dday_intcp can be estimated from historic air- temperature and solar-radiation data. One method is to make monthly plots of tmax versus their daily degree-day coefficients, dd, for days without precipitation. The dd values for this plot are computed using Figure 2.1, “Example of coaxial relationship for estimating shortwave solar radiation from maximum daily air temperature developed for northwestern Colorado.” and the daily solf ratios computed from historic data. A set of monthly lines then can be drawn through these points either visually or with linear- regression techniques. If there is a large difference in elevation between the climate station on the watershed and the station with radiation data, the air-temperature data associated with the radiation data should be adjusted to the elevation of the study-basin climate station.

A more rapid and coarse procedure is to establish two points for each monthly line using some average values. One point for each month is estimated using the average solf and average maximum temperature for days without precipitation. The second point is estimated using the maximum observed temperature for each month and a dd value of 15. Using this second procedure, curves shown in part A of Figure 2.1, “Example of coaxial relationship for estimating shortwave solar radiation from maximum daily air temperature developed for northwestern Colorado.” were estimated for a region in northwest Colorado. Estimates of radj_wppt and radj_sppt are obtained from the radiation record. radj_wppt is the ratio of solf for days with precipitation to days without precipitation over the October through April period. radj_sppt is the ratio of solf for days with precipitation to days without precipitation over the May through September period

1.3.5.2. References

Leaf, C. F., and Brink, G. E., 1973, Hydrologic simulation model of Colorado subalpine forest: U.S. Department of Agriculture, Forest Service Research Paper RM-107, 23 p.

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

Lee, R., 1963, Evaluation of solar beam irradiation as a climatic parameter of mountain watersheds: Colorado State University Hydrology Papers, no. 2, 50 p.

Meeus, J., 1999, Astronomical Algorithms: Richmond, Va., Willmann-Bell, Inc., 477 p.

Swift, Lloyd W., Jr., 1976, Algorithm for solar radiation on mountain slopes: Water Resources Research, v. 12, no. 1, p. 108-112.

1.4. Component 'Gwflow'

Groundwater Flow.Sums inflow to groundwater reservoirs and computes outflow tostreamflow and to a groundwater sink if specified.

Name

prms2008.Gwflow

Author

George H. Leavesley - ghleavesley@colostate.edu

Keyword
Groundwater

Version
$Id: Gwflow.java 861 2010-01-21 01:54:38Z ghleavesley $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Gwflow.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.4.1. Parameter

basin_area `[acres] - double`
Total basin area.

gwflow_coef `[1/day] - double[]`
Groundwater routing coefficient Groundwater routing coefficient - is multiplied by the storage in the groundwater reservoir to compute groundwater flow contribution to down-slope flow

gwsink_coef `[1/day] - double[]`
Groundwater sink coefficient Groundwater sink coefficient - is multiplied by the storage in the groundwater reservoir to compute the seepage from each reservoir to the groundwater sink

gwstor_init `[inches] - double[]`
Initial storage in each gw reservoir Storage in each groundwater reservoir at the beginning of a simulation

hru_area `[acres] - double[]`
HRU area, Area of each HRU

hru_gwres `- int[]`
Index of groundwater reservoir assigned to HRU Index of groundwater reservoir receiving excess soil water from each HRU

ngw `- int`
Number of Ground water reservoirs.

nhru `- int`
Number of HRUs.

nssr `- int`
Number of subsurface reservoirs.

ssr_gwres `- int[]`
Index of gw reservoir to receive flow from ss reservoir Index of the groundwater reservoir that will receive flow from each subsurface or gravity reservoir

1.4.2. Variables (In)

active_gwrs `- int`
Number of active GWRs

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

deltim `[hours] - double`
Length of the time step

gwr_route_order `- int[]`
Routing order for ground-water reservoirs

gwres_area `[acres] - double[]`
Groundwater reservoir area.

hru_perv `[acres] - double[]`
HRU pervious area. [basin]

hru_route_order `- int[]`
Routing order for HRUs

soil_to_gw `[inches] - double[]`
The amount of water transferred from the soil zone to a groundwater reservoir for each HRU. [smbal]

ssr_to_gw `[inches] - double[]`
Flow from each subsurface reservoir to its associated groundwater reservoir. [ssflow]

ssres_area `[acres] - double[]`
Subsurface reservoir area. [ssflow]

1.4.3. Variables (Out)

basin_gwflow `[inches] - double`
Basin area weighted average of groundwater flow

basin_gwin `[inches] - double`
Basin area weighted average of inflow to groundwater reservoirs

basin_gwsink `[inches] - double`
Basin area weighted average of groundwater reservoir storage to the groundwater sink

basin_gwstor `[inches] - double`
Basin area weighted average of groundwater storage

gw_in_soil `[acre-inches] - double[]`
Sum of inflows to each groundwater reservoir from the soil-water excess of associated HRUs

gw_in_ssr `[acre-inches] - double[]`
Sum of inflows to each groundwater reservoir from associated subsurface or gravity reservoirs

gwres_flow `[inches] - double[]`
Outflow from each groundwater reservoir to streams

gwres_in `[acre-inches] - double[]`
Sum of inflows to each groundwater reservoir from all associated soil-zone reservoirs

gwres_sink `[inches] - double[]`
Amount of water transferred from groundwater reservoirs to the groundwater sink. This water is effectively routed out of the basin and will not be included in streamflow

gwres_stor `[inches] - double[]`
Storage in each groundwater reservoir

1.4.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.4.5. Gwflow

1.4.5.1. Description

The groundwater system is conceptualized as a linear reservoir and is assumed to be the source of all baseflow. Inflow to the groundwater reservoir is from excess soil moisture, soil_to_gw, and from seepage from a subsurface reservoir, ssr_to_gw. The shape of the baseflow recession of the simulated hydrograph will be influenced by the relative proportion of groundwater recharge from the two sources. Recharge from soil_to_gw occurs only on days when soil_moist_max is exceeded by infiltration, while ssr_to_gw occurs at any time there is water available in the subsurface reservoir. Therefore, the use of ssr_to_gw to recharge groundwater preferentially over soil_to_gw could decrease subsurface flow and increase groundwater contributions to the simulated hydrograph.

The flow from each groundwater reservoir (gwres_flow), expressed in acre-inches is computed by:

Equation 2.1.

$$\text{gwres\_flow} = \text{gwflow\_coef} \times \text{gwres\_stor}$$

where

gwflow_coef
: the groundwater routing coefficient to obtain groundwater flow contribution to streamflow, and

gwres_stor
: the total storage in each groundwater reservoir.

gwflow_coef and the initial value of gwres_stor, gwstor_init, can be estimated from available streamflow records using the hydrograph separation technique described by Linsley, Kohler and Paulhus (1958). Integrating the characteristic depletion equation:

$$q_t = q_0 \times K_r^t$$

where

qt, q0
: streamflow at times t and 0, and

Kr
: a recession constant

They show a relationship between gwres_flow and gwres_stor that is expressed as:

Equation 2.2.

$$\text{gwres\_stor} = -\frac{\text{gwres\_flow}}{\log_e K_r}$$

where Kr is the slope of the groundwater flow recession obtained from the semi-log plot for discharge versus time.

Rewriting this equation as

Equation 2.3.

$$gwres\_flow = -\log_e K_r \times gwres\_stor$$

shows that -loge Kr is equivalent to gwflow_coef in the first equation.

The movement of water through the groundwater reservoir to points beyond the area of interest or measurement is treated using a groundwater sink. The accretion to gwres_sink is computed by:

Equation 2.4.

$$gwres\_sink = gwsink\_coef \times gwres\_stor$$

One or more groundwater reservoirs can be delineated in a watershed. More than one reservoir requires sufficient data to estimate initial storage volumes and routing coefficients. On small watersheds, only one groundwater reservoir is normally specified.

This module also computes weighted averages for gwres_stor, gwres_flow and gwres_sink for the basin.

1.4.5.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

Linsley, R. K., JR., Kohler, M. A., and Paulhus, J. L., 1958, Hydrology for Engineers: New York, McGraw-Hill, p.151-155.

1.5. Component 'Intcp'

Interception calculation.Computes amount of intercepted rain and snow, evaporationfrom intercepted rain and snow, and net rain and snow thatreaches the soil or snowpack.

Name
`prms2008.Intcp`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Interception

Version
$Id: Intcp.java 1059 2010-03-11 21:05:20Z odavid $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Intcp.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.5.1. Parameter

cov_type - `int[]`
Vegetation cover type designation for each HRU (0=bare soil; 1=grasses; 2=shrubs; 3=trees)

covden_sum `[decimal fraction] - double[]`
Summer vegetation cover density for the major vegetation type on each HRU

covden_win `[decimal fraction] - double[]`
Winter vegetation cover density for the major vegetation type on each HRU

epan_coef `- double[]`
Evaporation pan coefficient Evaporation pan coefficient

hru_area `[acres] - double[]`
Area of each HRU

hru_type `- int[]`
Type of each HRU (0=inactive; 1=land; 2=lake; 3=swale)

nevap `- int`
Number of evaporation pan stations.

nhru `- int`
Number of HRUs.

potet_sublim `[decimal fraction] - double`
Proportion of potential ET that is sublimated from snow surface

snow_intcp `[inches] - double[]`
Snow interception storage capacity for the major vegetation type in each HRU

srain_intcp `[inches] - double[]`
Summer rain interception storage capacity for the major vegetation type in each HRU

wrain_intcp `[inches] - double[]`
Winter rain interception storage capacity for the major vegetation type in the HRU

1.5.2. Variables (In)

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

basin_ppt `- double`

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

deltim `[hours] - double`
Length of the time step

hru_ppt `[inches] - double[]`
Precipitation on HRU, rain and snow. [precip]

hru_rain `[inches] - double[]`
Rain on HRU. [precip]

hru_route_order `- int[]`
Routing order for HRUs

hru_snow `[inches] - double[]`
Snow on HRU. [precip]

newsnow `- int[]`
New snow on HRU (0=no; 1=yes)

pan_evap `[inches] - double[]`
Measured pan evaporation. [obs]

pkwater_equiv `[inches] - double[]`
Psuedo parameter, snow pack water equivalent from previous time step.

potet `[inches] - double[]`
Potential evapotranspiration for each HRU. [potet]

pptmix `- int[]`
Precipitation is mixture of rain and snow (0=no; 1=yes)

route_on `- int`
Kinematic routing switch (0=daily; 1=storm period)

swrad `[calories/cm2] - double[]`
The computed solar radiation for each HRU [solrad]

tavgc `[deg C] - double[]`
Average HRU temperature. [temp]

tmaxf `[deg F] - double[]`
Maximum HRU temperature. [temp]

transp_on `- int[]`
Indicator for whether transpiration is occurring, 0=no, 1=yes. [potet]

1.5.3. Variables (Out)

basin_intcp_evap `[inches] - double`
Basin area-weighted evaporation from interception

basin_intcp_stor `[inches] - double`
Basin area-weighted average interception storage

basin_net_ppt `[inches] - double`
Basin area-weighted average net_ppt

hru_intcpevap `[inches] - double[]`
Evaporation from interception on each HRU

hru_intcpstor `[inches] - double[]`
Storage in canopy on each HRU

intcp_evap `[inches] - double[]`
Evaporation from interception on canopy of each HRU

intcp_form `- int[]`
Form (rain or snow) of interception (0=rain; 1=snow)

intcp_on `- int[]`
Whether there is interception in the canopy (0=no; 1=yes)

intcp_stor `[inches] - double[]`
Current interception storage on each HRU

last_intcp_stor `[inches] - double`
Basin area-weighted average changeover interception

net_ppt `[inches] - double[]`
HRU precipitation (rain and/or snow) with interception removed

net_rain `[inches] - double[]`
hru_rain minus interception

net_snow `[inches] - double[]`
hru_snow minus interception

newsnow `- int[]`
New snow on HRU (0=no; 1=yes)

pptmix `- int[]`
Precipitation is mixture of rain and snow (0=no; 1=yes)

1.5.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.5.5. Intcp

1.5.5.1. Description

Interception of precipitation is computed as a function of the cover density (`covden_sum` and `covden_win`) and the storage available (`snow_intcp`, `srain_intcp`, and `wrain_intcp`) for the predominant vegetation on an HRU. HRU precipitation is obtained from a precipitation distribution module in the form of total precipitation (`hru_ppt`) and the amounts of `hru_ppt` that are in the form of rain (`hru_rain`) and snow (`hru_snow`). Net rain (`net_rain`) during the summer period is computed by:

$$\text{net\_rain} = [\text{hru\_rain} \times (1.0 - \text{covden\_sum})] + (\text{thrufall} \times \text{covden\_sum})$$

where `thrufall` is the summer period cover density, and

`thrufall` is computed by

Equation 2.5.

$$\text{thrufall} = \text{hru\_rain} - (\text{srain\_intcp} - \text{intcp\_stor})$$

where

srain_intcp
rain interception storage capacity for the major vegetation type during the summer period (in.), and

intcp_stor
current depth of interception storage (in.)

`net_rain` for the winter period is computed as above but with the winter cover density (`covden_win`) substituted for `covden_sum` and the winter interception storage capacity for rain (`wrain_intcp`) substituted for `srain_intcp`.

net_snow is also computed in the same manner but with the substitution of hru_snow for hru_rain, winter cover density (covden_win) for covden_sum, and the interception storage capacity for snow (snow_intcp) for srain_intcp.

The existence of intercepted precipitation is denoted by setting intcp_on to a value of 1. A value of 0 indicates no intercepted precipitation. The form of the intercepted precipitation is denoted by intcp_form which is set to 0 for rain and 1 for snow. If precipitation is a mixture of rain and snow, rain is assumed to occur first and interception is computed for each precipitation form. net_ppt is the sum of net_rain plus net_snow. When snow falls on intercepted rain, intcp_form is changed to 1 and net_snow is computed as above. snow_intcp is assumed to always be greater than or equal to wrain_intcp.

The potential evaporation rate for intercepted precipitation is computed as a function of interception form. Intercepted rain is assumed to evaporate at a free-water surface rate. If pan-evaporation data are used, then the rain evaporation rate (evrn) equals the pan loss rate. If potential evapotranspiration (potet) is computed from meteorological variables, evrn is computed by:

Equation 2.6.

$$evrn = \frac{potet}{epan\_coef}$$

where epan_coef is the monthly evaporation-pan coefficient.

Sublimation of intercepted snow (evsn) is assumed to occur at a rate proportional to potet and is computed by:

Equation 2.7.

$$evsn = potet\_sublim \times potet$$

where potet_sublim is the proportion of potet that is sublimated from the snow surface.

Actual loss from interception (intcp_evap) is equal to the smaller value of intcp_stor or the computed evaporation loss (evrn or evsn). If intcp_stor is not depleted in one time step, the remainder is carried over to the next time step. intcp_evap represents loss from the percentage of an HRU expressed in covden_sum or covden_win. For water balance computations, intcp_evap is adjusted to represent an HRU average value.

1.5.5.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.6. Component 'Obs'

Read input variables.Reads input variables from the designated data file.

Name
prms2008.Obs

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
IO

Version
$Id: Obs.java 1300 2010-06-08 17:17:37Z odavid $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Obs.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.6.1. Parameter

endTime `[yyyy-mm-dd] - Calendar`
Ending date of the simulation.

inputFile `- File`

rain_code `- int[]`
Code indicating rule for precip station use (1=only precip if the regression stations have precip; 2=only precip if any station in the basin has precip; 3=precip if xyz says so; 4=only precip if rain_day variable is set to 1; 5=only precip if psta_freq_nuse stations see precip)

startTime `[yyyy-mm-dd] - Calendar`
Starting date of the simulation.

1.6.2. Variables (Out)

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

deltim `[hours] - double`
Length of the time step

moreData `- boolean`

newday `- int`
Switch signifying the start of a new day (0=no; 1=yes)

pan_evap `[inches] - double[]`
Measured pan evaporation at each measurement station

precip `[inches] - double[]`
Measured precipitation at each rain gage

rain_day `- int`
Flag to force rain day

route_on `- int`
Kinematic routing switch (0=daily; 1=storm period)

runoff `[cfs] - double[]`
Measured runoff for each stream gage

solrad `[langleys] - double[]`
Measured solar radiation at each measurement station

tmax `[degrees] - double[]`
Measured daily maximum temperature at each measurement station

tmin `[degrees] - double[]`
Measured daily minimum temperature at each measurement station

1.6.3. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.7. Component 'Output'

Name
`prms2008.Output`

1.7.1. Parameter

outFile - `File`

1.7.2. Variables (In)

basin_cfs - `double`

date - `Calendar`

runoff - `double[]`

1.8. Component 'PotetJh'

Potential ET - Jensen Haise.Determines whether current time period is one of activetranspiration and computes the potential evapotranspirationfor each HRU using the Jensen-Haise formulation.

Name
`prms2008.PotetJh`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Evapotranspiration

Version
$Id: PotetJh.java 1132 2010-04-08 19:54:26Z ghleavesley $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/PotetJh.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.8.1. Parameter

hru_area `[acres] - double[]`
HRU area , Area of each HRU

jh_coef `[per degrees] - double[]`
Monthly air temperature coefficient used in Jensen -Haise potential evapotranspiration computations, see PRMS manual for calculation method

jh_coef_hru `[per degrees] - double[]`
Jensen-Haise Air temperature coefficient used in Jensen-Haise potential evapotranspiration computations for each HRU. See PRMS manual for calculation method

nhru `- int`
Number of HRUs.

nsol `- int`
Number of solar radiation stations.

1.8.2. Variables (In)

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

deltim `[hours] - double`
Length of the time step

hru_route_order `- int[]`
Routing order for HRUs

newday `- int`
Switch signifying the start of a new day (0=no; 1=yes)

route_on `- int`
Kinematic routing switch (0=daily; 1=storm period)

swrad `[calories/cm2] - double[]`
The computed solar radiation for each HRU. [solrad]

tavgc `[deg C] - double[]`
Average HRU temperature. [temp]

tavgf `[deg F] - double[]`
Average HRU temperature. [temp]

1.8.3. Variables (Out)

basin_potet `[inches] - double`
Basin area-weighted average of potential et

basin_potet_jh `[inches] - double`
Basin area-weighted average of potential et

potet `[inches] - double[]`
Potential evapotranspiration on an HRU

1.8.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.8.5. PotetJh

1.8.5.1. Description

The potential evapotranspiration for each HRU (`potet`) for each time period is computed by:

$$potet = jh\_coef \times (tavgf - jh\_coef\_hru) \times rin$$

where

jh_coef
: the monthly air temperature coefficient used in Jensen-Haise potential evapotranspiration computations,

jh_coef_hru
: the air temperature coefficient used in Jensen-Haise potential evapotranspiration computations for each HRU,

rin
: the daily solar radiation expressed in inches of evaporation potential, and

tavgf
: the average HRU temperature, in °F.

For aerodynamically rough crops, which are assumed to include forests, `jh_coef` can be computed each month for the watershed by

$$jh\_coef = [C1 + (13.0 \times CH)]^{-1}$$

`C1` is an elevation correction factor computed by:

$$C1 = 68.0 - \left[3.6 \times \frac{E1}{1000}\right]$$

where `E1` is the median elevation of the watershed in feet.

`CH` is a humidity index computed by:

$$CH = \frac{50.0}{e2 - e1}$$

where

e2
: the saturation vapor pressure (mb) for the mean maximum air temperature for the warmest month of the year, and

e1
: the saturation vapor pressure (mb) for the mean minimum air temperature for the warmest month of the year

`jh_coef_hru` is computed for each HRU by:

Equation 2.8.

$$jh\_coef\_hru = 27.5 - (0.25 \times (e2 - e1)) - \frac{E2}{1000}$$

where `E2` is the median elevation of the HRU in feet.

The basin weighted average potential evapotranspiration, `basin_potet`, is also computed in this module.

1.8.5.2. References

Jensen, M. E., and Haise, H. R., 1963,Estimating evapotranspiration from solar radiation: Proceedings of the American Society of Civil Engineers, Journal of Irrigation and Drainage, v.89, no. IR4, p. 15-41.

Jensen, M. E., Rob, D. C. N., and Franzoy, C. E., 1969, Scheduling irrigations using climate-crop-soil data: National Conference on Water Resources Engineering of the American Society of Civil Engineers, New Orleans, LA., 1969, Proceedings, 20 p.

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.9. Component 'Precip'

Precipitation form and distribution.This component determines whether measured precipitationis rain or snow and distributes it to the HRU's.

Name
`prms2008.Precip`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Precipitation

Version
$Id: Precip.java 861 2010-01-21 01:54:38Z ghleavesley $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Precip.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.9.1. Parameter

adjmix_rain `[decimal fraction] - double[]`
Adjustment factor for rain in a rain/snow mix Monthly factor to adjust rain proportion in a mixed rain/snow event

hru_area `[acres] - double[]`
Area of each HRU

hru_psta `- int[]`
Index of the base precipitation station used for lapse rate calculations for each HRU.

nhru `- int`
Number of HRUs.

nrain `- int`
Number of precipitation stations.

nstorm `- int`
Number of storms.

ntemp `- int`
Number of temperature stations.

precip_units `- int`
Units for measured precipitation Units for measured precipitation (0=inches; 1=mm)

rain_adj `[decimal fraction] - double[][]`
Monthly factor to adjust measured precipitation on each HRU to account for differences in elevation, etc

snow_adj `[decimal fraction] - double[][]`
Monthly factor to adjust measured precipitation on each HRU to account for differences in elevation, etc

storm_scale_factor `[percent] - double[]`
Adjustment factor for each storm

strain_adj `[decimal fraction] - double[][]`
Monthly factor to adjust measured precipitation to each HRU to account for differences in elevation, etc. This factor is for the rain gage used for kinematic or storm routing

temp_units `- int`
Units for measured temperature (0=Fahrenheit; 1=Celsius)

tmax_allrain `[degrees] - double[]`
If maximum temperature of an HRU is greater than or equal to this value (for each month, January to December), precipitation is assumed to be rain, in deg C or F, depending on units of data

tmax_allsnow `[degrees] - double`
If HRU maximum temperature is less than or equal to this value, precipitation is assumed to be snow, in deg C or F, depending on units of data

1.9.2. Variables (In)

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

hru_route_order `- int[]`
Routing order for HRUs

precip `[inches] - double[]`
Observed precipitation at each measurement station. [obs]

route_on `- int`
Kinematic routing switch (0=daily; 1=storm period)

solrad_tmax `- double`
Basin daily maximum temperature for use with solrad radiation component

tempc `[deg C] - double[]`
HRU adjusted temperature for timestep < 24

tempf `[deg F] - double[]`
HRU adjusted temperature for timestep < 24

tmaxc `[deg C] - double[]`
Maximum HRU temperature. [temp]

tmaxf `[deg F] - double[]`
Maximum HRU temperature. [temp]

tminc `[deg C] - double[]`
Minimum HRU temperature. [temp]

tminf `[deg F] - double[]`
Minimum HRU temperature. [temp]

1.9.3. Variables (Out)

basin_obs_ppt `[inches] - double`
Area weighted measured average precip for basin

basin_ppt `[inches] - double`
Area weighted adjusted average precip for basin

basin_rain `[inches] - double`
Area weighted adjusted average rain for basin

basin_snow `[inches] - double`
Area weighted adjusted average snow for basin

hru_ppt `[inches] - double[]`
Adjusted precipitation on each HRU

hru_rain `[inches] - double[]`
Computed rain on each HRU

hru_snow `[inches] - double[]`
Computed snow on each HRU

newsnow `- int[]`
New snow on HRU (0=no; 1=yes)

pptmix `- int[]`
Precipitation mixture (0=no; 1=yes)

prmx `[decimal fraction] - double[]`
Proportion of rain in a mixed event

1.9.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.9.5. Precip

1.9.5.1. Description

Total daily precipitation depth (hru_ppt) received on an HRU is computed by:

$$hru\_ppt = precip \times pcor$$

where

pcor
: rain_adj if precipitation is rain, or

pcor
: snow_adj if precipitation is snow, and

precip
: observed precipitation at the measurement station corresponding to the HRU.

Precipitation form (rain, snow, or a mixture of both) on each HRU is estimated from the HRU maximum and minimum daily air temperatures and their relationship to a base temperature (tmax_allsnow). Precipitation is all snow if the maximum temperature is less than or equal to the tmax_allsnow and all rain if the minimum temperature is greater than or equal to tmax_allsnow. If the maximum temperature is greater than tmax_allsnow and the minimum temperature is below tmax_allsnow, then the precipitation is considered a mixture, and the rain is assumed to occur first. The portion of the total precipitation occurring as rain (prmx) is computed by:

$$prmx = \left[\frac{tmax - tmax\_allsnow}{tmax - tmin}\right] \times adjmix\_rain$$

where

tmax
: the maximum HRU temperature,

tmin
: the minimum HRU temperature, and

adjmix_rain
: a monthly factor to adjust the rain proportion in a mixed rain/snow event.

This mixture algorithm can be overridden in two ways. One is the use of the parameter tmax_allrain, which is an air temperature value that when exceeded by tmax, forces the precipitation to be considered all rain. This parameter is useful for periods, such as in the spring, when the minimum daily temperatures may be below tmax_allsnow but precipitation is predominantly convective afternoon storms. The form of the precipitation may also be explicitly specified by including the variable form_data in the observed data file.

1.9.5.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

Willen, D. W., Shumway, C. A., and Reid, J. E., 1971, Simulation of daily snow water equivalent and melt: Western Snow Conference, Billings, Montana, 1971, Proceedings, v. 39, p1-8.

1.10. Component 'Smbal'

Soil moisture accounting.This module does soil moisture accounting, including additionof infiltration, computation of actual evapotranspiration, andseepage to subsurface and groundwater reservoirs.

Name
: prms2008.Smbal

Author
: George H. Leavesley - ghleavesley@colostate.edu

Keyword
Soilwater

Version
$Id: Smbal.java 1266 2010-05-25 20:52:52Z odavid $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Smbal.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.10.1. Parameter

basin_area `[acres] - double`
Total basin area. [basin]

cov_type `- int[]`
Vegetation cover type designation for HRU (0=bare soil; 1=grasses; 2=shrubs; 3=trees)

dprst_flag `- int`
Selection flag for depression storage computation. 0=No; 1=Yes

frozen `- int[]`
Flag for frozen ground (0=no; 1=yes)

hru_area `[acres] - double[]`
Area of each HRU

hru_type `- int[]`
Type of each HRU (0=inactive; 1=land; 2=lake; 3=swale)

nhru `- int`
Number of HRUs.

soil2gw_max `[ inches] - double[]`
The maximum amount of the soil water excess for an HRU that is routed directly to the associated groundwater reservoir each day

soil_moist_init `[inches] - double[]`
Initial value of available water in soil profile

soil_moist_max `[inches] - double[]`
Maximum available water holding capacity of soil profile. Soil profile is surface to bottom of rooting zone

soil_rechr_init `[inches] - double[]`
Initial value for soil recharge zone (upper part of soil_moist). Must be less than or equal to soil_moist_init

soil_rechr_max `[inches] - double[]`
Maximum value for soil recharge zone (upper portion of soil_moist where losses occur as both evaporation and transpiration). Must be less than or equal to soil_moist

soil_type `- int[]`
HRU soil type (1=sand; 2=loam; 3=clay)

1.10.2. Variables (In)

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

deltim `[hours] - double`
Length of the time step

dprst_evap_hru `[inches] - double[]`
Evaporation from depression storage for each HRU

gmelt_to_soil `[ inches] - double[]`
Ground-melt of snowpack, goes to soil

hru_impervevap `[inches] - double[]`
Evaporation from impervious area for each HRU

hru_intcpevap `[inches] - double[]`
Evaporation from interception on each HRU

hru_percent_perv `[decimal fraction] - double[]`
Proportion of each HRU area that is pervious

hru_perv `[acres] - double[]`
HRU pervious area. [basin]

hru_ppt `[inches] - double[]`
Adjusted precipitation on each HRU

hru_route_order `- int[]`
Routing order for HRUs

infil `[inches] - double[]`
Infiltration for each HRU. [sroff]

potet `[inches] - double[]`
Potential evapotranspiration for each HRU. [potet]

route_on `- int`
Kinematic routing switch (0=daily; 1=storm period)

snow_evap `[inches] - double[]`
Evaporation and sublimation from snowpack. [snow]

snowcov_area `[decimal fraction] - double[]`
Snow-covered area on an HRU, in decimal fraction of total HRU area. [snow]

transp_on `- int[]`
Indicator for whether transpiration is occurring. [potet]

1.10.3. Variables (Out)

basin_actet `[inches] - double`
Basin area weighted average of hru_actet for land HRUs

basin_gmelt2soil `[inches] - double`
Basin area weighted average of glacier melt to soil

basin_lakeevap `[inches] - double`
Basin area weighted average of lake evaporation

basin_perv_et `[inches] - double`
Basin area weighted average of pervious area ET

basin_soil_moist `[inches] - double[]`
Basin area weighted average for soil_moist

basin_soil_rechr `[inches] - double`
Basin area weighted average for soil_rechr

basin_soil_to_gw `[inches] - double`
Basin average excess soil water that flows directly to groundwater reservoirs

hru_actet `[inches] - double[]`
Actual evapotranspiration on HRU, pervious + impervious

perv_actet `[inches] - double[]`
Actual evapotranspiration from pervious areas of HRU

soil_moist `[inches] - double[]`
Current moisture content of soil profile to the depth of the rooting zone of the major vegetation type on the HRU

soil_rechr `[inches] - double[]`
Current moisture content of soil recharge zone, ie, the portion of the soil profile from which evaporation can take place

soil_to_gw `[inches] - double[]`
Portion of excess soil water from an HRU that flows to its associated groundwater reservoir

soil_to_ssr `[inches] - double[]`
Portion of excess soil water from an HRU that flows to its associated subsurface reservoir

1.10.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.10.5. Smbal

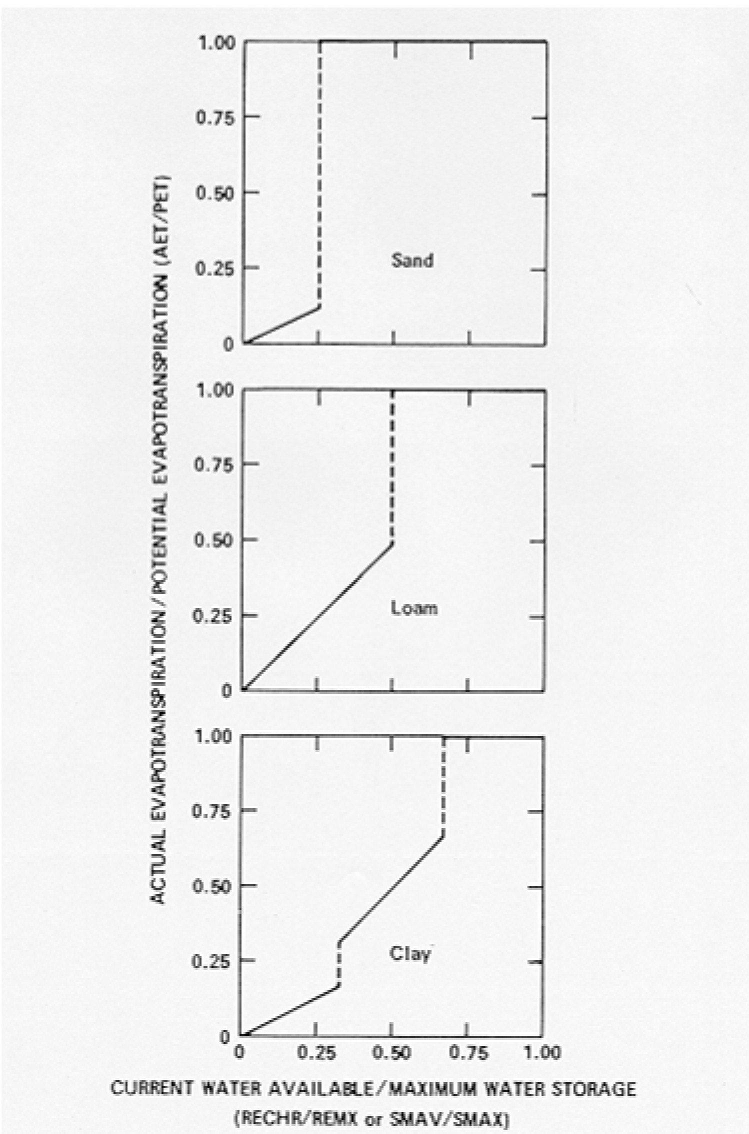
1.10.5.1. Description

Soil moisture accounting is performed as the algebraic summation of all moisture accretions to, and depletions from, the active soil profile. Depletions include evapotranspiration and recharge to the subsurface and groundwater reservoirs. Accretions are rainfall and snowmelt infiltration. The depth of the active soil profile is considered to be the average rooting depth of the predominant vegetation on the HRU. The maximum available water holding capacity of the soil zone, `soil_moist_max`, is the difference between field capacity and wilting point of the profile. The active soil profile is divided into two layers. The upper layer is termed the recharge zone and the lower layer is termed the lower zone. The recharge zone is user-definable as to depth and maximum available water-holding capacity, `soil_rechr_max`. The maximum available water-holding capacity of the lower zone is the difference between `soil_moist_max` and `soil_rechr_max`. Losses from the recharge zone occur from evaporation and transpiration; they occur only as transpiration from the lower zone.

First, infiltration from rainfall or snowmelt, `infil`, is added to the soil zone. Water in excess of `soil_moist_max` is distributed to the subsurface and groundwater reservoirs. The excess is first used to satisfy the maximum groundwater-recharge, soil_to_gw, and any remaining excess (soil_to_ssr) is added to the subsurface reservoir associated with the HRU.

Actual evapotranspiration, hru_actet, is the computed rate of water loss, which reflects the availability of water to satisfy potet. When available water is non-limiting, hru_actet equals potet. Evaporation of intercepted water, intcp_evap, evaporation from impervious area retention storage, imperv_evap, and evaporation/sublimation from a snowpack, snow_evap, are first used to satisfy potet, in that order. Remaining potet demand then is applied to the soil-zone storage. perv_actet is computed separately for the recharge zone and the lower zone using the unsatisfied demand and the ratio of currently available water in the soil zone to its maximum available water-holding capacity. For the recharge zone, this ratio is $\frac{\text{soil\_rechr}}{\text{soil\_rechr\_max}}$. For the lower zone, the ratio $\frac{\text{soil\_moist}}{\text{soil\_moist\_max}}$ is used. The perv_actet for the recharge zone is first used to satisfy potet; any remaining demand is attempted to be met from the lower zone. HRU soils are designated as being predominantly sand, loam or clay, using parameter soil_type. The potential-actual relations for these soil types as a function of the soil-water ratio are shown in (Zahner, 1967). hru_actet is computed as the area weighted average of perv_actet and imperv_evap.

Figure 2.2. Soil-water withdrawal functions for evapotranspirtation.



This module also computes basin weighted averages, `basin_actet`, `basin_soil_moist`, and `basin_soil_rechr`.

1.10.5.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

Zahner, R., 1967, Refinement in empirical functions for realistic soil-moisture regimes under forest cover, in Sopper, W. E., and Lull, H. W., eds., International Symposium of Forest Hydrology: New York, Pergamon Press, p. 261-274

1.11. Component 'Snowcomp'

Snow accounting component.Initiates development of a snowpack and simulates snowaccumulation and depletion processes using an energy-budget approach.

Name
`prms2008.Snowcomp`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Snow

Version
$Id: Snowcomp.java 1240 2010-05-18 22:51:59Z ghleavesley $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Snowcomp.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.11.1. Parameter

albset_rna `[decimal fraction] - double`
Albedo reset - rain, accumulation stage Proportion of rain in a rain-snow precipitation event above which the snow albedo is not reset. Applied during the snowpack accumulation stage.

albset_rnm `[decimal fraction] - double`
Albedo reset - rain, melt stage Proportion of rain in a rain-snow precipitation event above which the snow albedo is not reset. Applied during the snowpack melt stage

albset_sna `[inches] - double`
Albedo reset - snow, accumulation stage Minimum snowfall, in water equivalent, needed to reset snow albedo during the snowpack accumulation stage

albset_snm `[inches] - double`
Albedo reset - snow, melt stage Minimum snowfall, in water equivalent, needed to reset snow albedo during the snowpack melt stage

basin_area `[acres] - double`
Total basin area. [basin]

cecn_coef `[calories per degree C above 0] - double[]`
Convection condensation energy coefficient, varied monthly

cov_type `- int[]`
Cover type designation for HRU Vegetation cover type designation for HRU (0=bare soil; 1=grasses; 2=shrubs; 3=trees)

covden_sum `- double[]`
Summer vegetation cover density for the major vegetation type on each HRU. [intcp]

covden_win `[decimal fraction] - double[]`
Winter vegetation cover density for the major vegetation type on each HRU

den_init `[gm/cm3] - double`
Initial density of new-fallen snow

den_max `[gm/cm3] - double`
Average maximum snowpack density

emis_noppt `[decimal fraction] - double`
Average emissivity of air on days without precipitation

freeh2o_cap `[decimal fraction] - double`
Free-water holding capacity of snowpack expressed as a decimal fraction of the frozen water content of the snowpack (pk_ice)

glacier_flag `- int`
Flag incicating presence of a glacier, (0=no; 1=yes)

groundmelt `[inches/day] - double[]`
Amount of snowpack-water that melts each day to soils

hru_area `[acres] - double[]`
Area of each HRU

hru_deplcrv `- int[]`
Index number for the snowpack areal depletion curve associated with an HRU

hru_type `- int[]`
Type of each HRU (0=inactive; 1=land; 2=lake; 3=swale)

melt_force `[Julian day] - int`
Julian date to force snowpack to spring snowmelt stage; varies with region depending on length of time that permanent snowpack exists

melt_look `[Julian day] - int`
Julian date to start looking for spring snowmelt stage. Varies with region depending on length of time that permanent snowpack exists

ndepl `- int`
Number of snow cover depletion curves.

ndeplval `- int`
Number of values in each snow cover depletion curve.

nhru `- int`
Number of HRUs.

nradpl - `int`
Number of radiation planes.

nsol - `int`
Number of solar radiation stations.

potet_sublim `[decimal fraction] - double`
Proportion of potential ET that is sublimated from the snow surface

rad_trncf `[decimal fraction] - double[]`
Transmission coefficient for short-wave radiation through the winter vegetation canopy

settle_const `[decimal fraction] - double`
Snowpack settlement time constant

snarea_curve `[decimal fraction] - double[][]`
Snow area depletion curve values, 11 values for each curve (0.0 to 1.0 in 0.1 increments)

snarea_thresh `[inches] - double[]`
Maximum threshold water equivalent for snow depletion The maximum threshold snowpack water equivalent below which the snow-covered-area curve is applied. Varies with elevation.

tmax_allsnow `[degrees] - double`
If maximum temperature of an HRU is less than or equal to this value, precipitation is assumed to be snow

tstorm_mo - `double[]`
Monthly indicator for prevalent storm type (0=frontal storms prevalent; 1=convective storms prevalent)

1.11.2. Variables (In)

active_hrus - `int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

basin_horad `[langleys] - double`
Potential shortwave radiation for the basin centroid

basin_ppt `[inches] - double`
Average basin precipitation. [precip]

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

deltim `[hours] - double`
Length of the time step

hru_intcpevap `[inches] - double[]`
Evaporation from interception on each HRU

hru_route_order - `int[]`
Routing order for HRUs

net_ppt `[inches] - double[]`
HRU net precipitation, the sum of net_rain and net_snow. [intcp]

net_rain `[inches] - double[]`
Rain on an HRU (hru_rain) minus interception. [intcp]

net_snow `[inches] - double[]`
Snow on an HRU (hru_snow) minus interception. [intcp]

newsnow `- int[]`
Indicator for new snow during time step. [precip]

orad `[langleys] - double`
Measured or computed solar radiation on a horizontal surface

potet `[inches] - double[]`
Potential evapotranspiration for each HRU. [potet]

pptmix `- int[]`
Indicator for mixed rain and snow during time step. [precip]

prmx `- double[]`
The proportion of rain in a mixture of rain and snow. [precip]

route_on `- int`
Kinematic routing switch (0=daily; 1=storm period)

swrad `[langleys] - double[]`
The computed solar radiation for each HRU. [solrad].

tavgc `[deg C] - double[]`
Average HRU temperature in xb0 C. [temp]

tmaxf `[deg F] - double[]`
Maximum HRU temperature. [temp]

tminf `[deg F] - double[]`
Minimum HRU temperature. [temp]

transp_on `- int[]`
Indicator for whether transpiration is occurring, 0=no, 1=yes. [potet]

1.11.3. Variables (Out)

albedo `[decimal fraction] - double[]`
Snow surface albedo on an HRU or the fraction of radiation reflected from the snowpack surface

basin_pk_precip `[inches] - double`
Basin area-weighted average precip added to snowpack

basin_pweqv `[inches] - double`
Average snowpack water equivalent for total basin area

basin_snowcov `[decimal fraction] - double`
Average snow-covered area for total basin area

basin_snowdepth `[inches] - double`
Basin area-weighted average snow depth

basin_snowevap `[inches] - double`
Average evaporation and sublimation for total basin area

basin_snowmelt `[inches] - double`
Average snowmelt for total basin area

gmelt_to_soil `[inches] - double[]`
Ground-melt of snowpack, goes to soil

pk_depth `[inches] - double[]`
Snowpack depth on an HRU

pkwater_ante `[inches] - double[]`
Antecedent snowpack water equivalent on an HRU

pkwater_equiv `[inches] - double[]`
Snowpack water equivalent on an HRU

pptmix_nopack `- int[]`
Indicator that a rain-snow mix event has occurred with no snowpack present on an HRU (1), otherwise (0)

snow_evap `[inches] - double[]`
Evaporation and sublimation from snowpack on an HRU

snowcov_area `[decimal fraction] - double[]`
Snow-covered area on an HRU

snowmelt `[inches] - double[]`
Snowmelt from snowpack on an HRU

tcal `[Langleys] - double[]`
Net snowpack energy balance on an HRU

1.11.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

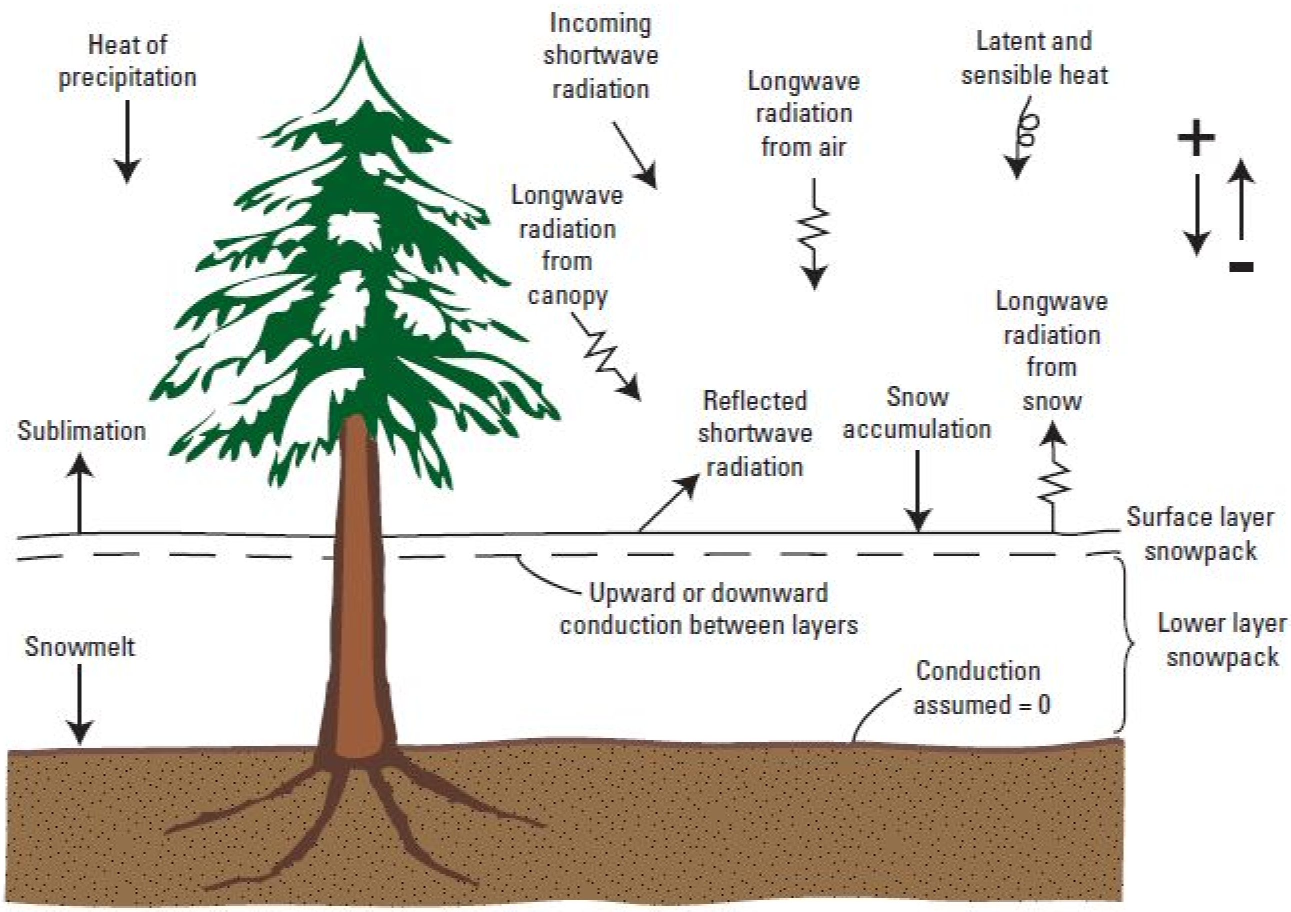
1.11.5. Snowcomp

1.11.5.1. Description

The snow module simulates the initiation, accumulation, and depletion of a snowpack on each HRU. A snowpack is maintained and modified both on a water-equivalent basis and as a dynamic-heat reservoir. A snowpack-water balance is computed daily, and an energy balance is computed twice each day for two 12-hour periods (designated day and night).

The conceptual model for the snowpack system and its energy relationships is one described by Obled and Rosse (1977). The snowpack is assumed to be a two-layered system. The surface layer consists of the upper 3 to 5 centimeters of the snowpack, and the lower layer is the remaining snowpack. Heat transfer between the surface layer and the lower layer occurs by conduction when the temperature of the surface layer (ts) is less than 0 °C (Celsius). When ts equals 0 °C, heat transfer occurs as conduction when the net energy balance at the air-snow interface is negative, and as mass transfer by surface melting when the net energy balance is positive. Heat transfer from precipitation occurs as a mass-transfer process. Conduction of heat across the soil-snow interface is assumed to be zero, since it is negligible compared to the energy exchange at the air-snow interface. The conceptual snowpack system and the components of the snowpack energy-balance equations are shown in ???.

Figure 2.3. Components of the snowpack energy balance.



The snow accumulation and melt processes are simulated using an execute method that calls a series of seven methods, each of which addresses one or more snow related processes and is described below:

Method `ppt_to_pack`

This method initiates the development of a new snowpack if one does not exist during a time step that receives precipitation input in the form of snow. If a snow pack already exists, the precipitation received during that time step is added to the snowpack.

If precipitation is in the form of rain, it is added to the snowpack and the heat content of the rain is used to modify the temperature and free-water content of the snowpack. If the snowpack temperature is less than 0 °C, the heat content of the rain is used to warm the snowpack. If the heat content is not sufficient to warm the snowpack to 0 °C, part or all of the rain is frozen, releasing 80 calories per gram of water frozen, to further warm the snowpack. If the snowpack reaches 0 °C, then any remaining liquid water is used to satisfy the free-water-holding capacity of the snowpack. Liquid water available in excess of the free-water-holding capacity leaves the bottom of the snowpack as snowmelt.

If the precipitation is in the form of snow, it is added to the snowpack and its heat content is used to recompute the snowpack temperature and cold content. If free water is available, part or all is frozen using the cold content of the new snow.

If the precipitation occurs as a mixture of rain and snow, rain is assumed to occur first. If no snowpack existed prior to the storm, the rain amount is input to the soil module and the snow amount is used to initiate a new snowpack. If a snowpack existed, rain and snow are added to the snowpack in sequential order and snowpack conditions are computed as described above.

Method `snalbedo`

This subroutine computes the albedo of the snowpack surface. For computational purposes, the snowpack is considered to be in either an accumulation phase or a melt phase. A new snowpack begins in the accumulation phase and continues in this phase until the temperature of the lower layer is 0 °C for 5 consecutive days or the Julian date exceeds a forcing date (`melt_force`). At this point, the snowpack shifts to the melt phase. The date to begin checking for the start of the melt phase (`melt_look`) and `melt_force` are user-specified.

Snow surface albedo (`albedo`) is computed as a function of the snowpack phase and the number of days since the last snowfall (`slst`). A separate albedo-slst relationship (U.S. Army, 1956) is used for each phase. The albedo of new snow decays daily according to the albedo-slst relationship in effect at the time of the snowfall. `slst` is reset to zero when new snow water equivalent equals or exceeds the parameter `albset_sna` during the accumulation phase, or `albset_snm` during the melt phase. For precipitation that is a mixture of rain and snow, `slst` is not reset if the percentage rain in the mixture (`prmx`) is greater than or equal to the parameter `albset_rna` for the accumulation phase or `albset_rnm` for the melt phase.

Subroutine `snowbal`

This subroutine computes the energy balance for a snowpack for a 12 hour period. It is called twice from the run function using the variable `niteda` to define the day and night periods.

The temperature of the surface layer (`ts`) for the daylight period is computed as the mean of the maximum and mean-daily air temperature for an HRU. For the night period, it is computed as the mean of the minimum and mean-daily air temperature for a HRU. Snow surface temperature equals 0 °C when these means exceed 0 °C. When `ts` is less than 0 °C, the mean air temperature for that 12-hour period is assumed to integrate the effects of the radiation, latent heat, sensible heat, and diffusion processes expressed in the complete equation for the surface temperature. See Anderson (1968) for a complete discussion of the equation.

The conduction of heat between the snow surface and the lower layer of the snowpack (`qcond`) is computed by:

Equation 2.9.

$$qcond = 2.0 \times pk\_den \times cs \times \sqrt{\frac{keff}{pk\_den \times cs} \times \frac{\Delta t}{\Pi}} \times (ts - pk\_temp)$$

where

pk_den
: snowpack density (g/cm3),

cs
: the specific heat of ice in calories per gram per °C (cal/g/°C),

keff
: the effective thermal conductivity of the snowpack (cal/s/cm/°C),

#t
: the time interval(s),

#
: pi, and

pk_temp
    the temperature of the lower layer of the snowpack (°C).

keff is assumed equal to $0.0077 \times \text{pk\_den}^2$ (Anderson, 1968) where pk_den is computed by:

$$\text{pk\_den} = \frac{\text{pkwater\_equiv}}{\text{pk\_depth}}$$

where

pkwater_equiv
    snowpack water equivalent (in.), and

pk_depth
    snowpack depth (in.).

pk_depth is computed using a finite difference scheme to solve the equation (Riley, Israelsen, and Eggleston, 1973):

$$\frac{d\left[\text{pk\_depth(t)}\right]}{d\,t} + \left[\text{settle\_const} \times \text{pk\_depth(t)}\right] = \frac{\text{net\_snow}}{\text{den\_init}} + \left[\frac{\text{settle\_const}}{\text{den\_max}} \times \text{pss}\right]$$

where

settle_const
    a settlement-time constant,

new_snow
    net daily snowfall (in. water equivalent),

den_init
    initial density of new-fallen snow,

den_max
    the assumed average maximum snowpack density,

t
    time in days (t=0 at date of snowpack initiation), and

pss
    accumulated sum of net_snow from time of snowpack initiation (in. water equivalent).

The temperature of the lower layer of the snowpack, pk_temp, is recomputed each 12-hour period by:

$$\text{pk\_temp} = \frac{\text{-pk\_def}}{(\text{pkwater\_equiv} \times 1.27)}$$

where

pk_def
    the number of calories of heat required to bring the lower layer to an isothermal state at 0 °C, and

constant 1.27
    the product of the specific heat of ice (0.5 calories/ gram/°C) and the conversion factor for inches to centimeters (2.54).

When the temperature of the surface layer equals 0 °C, an energy balance at the air-snow interface is computed. For each 12-hour period the energy balance cal in calories is computed by:

$$cal = swn + lwn + cecsub$$

where

swn
net shortwave radiation (cal),

lwn
net longwave radiation (cal),

cecsub
an approximation of the convection-condensation energy terms (cal).

An additional energy term for the heat content of precipitation (`calpr`) is computed in method `ppt_to_pack` and is added to the snowpack before `cal` is computed.

Net shortwave radiation `swn` is computed by:

$$swn = swrad \times (1.0 - albedo) \times rad\_trncf$$

where

swrad
the computed shortwave solar radiation received (ly),

albedo
the albedo of the snowpack surface, and

rad_trncf
the transmission coefficient for the vegetation canopy over the snowpack.

`swrad` for each HRU is obtained as an external variable. `albedo` is computed in method `snalbedo`. `rad_trncf` is estimated using a winter forest cover density (`covden_win`)-`rad_trncf` relationship for the vegetation canopy over the snowpack. Miller (1959) developed a `covden_win-rad_trncf` relationship for several species of pine, and Vézina and Péch (1964) developed a similar relationship for both jack pine and balsam fir. An average of the pine relationships has given reasonable results in model applications (Leavesley and Striffler, 1978).

Longwave net radiation (`lwn`) has two components--the longwave exchange between the air and snowpack surface (`sky`) and the longwave exchange between the vegetation canopy and the snowpack surface (`can`). `lwn` is computed by:

$$lwn = sky + can$$

where

$$sky = (1.0 - covden\_win) \times [(emis \times air) - sno]$$

$$can = covden\_win \times (air - sno)$$

where

covden_win
the winter cover density of the predominant vegetation above the snowpack,

emis
the emissivity of air,

air
: the longwave energy emitted from a perfect black body at the average air temperature for the 12-hour period (cal), and

sno
: the longwave energy emitted from the snowpack surface at the surface temperature (ts) for the 12-hour period (cal).

`air` and `sno` are computed using the Stefan-Boltzmann law. The temperature used for `air` is the average of the daily-minimum and daily-mean air temperatures for the night period, and the average of the daily-maximum and daily-mean air temperature for the daylight period. The snowpack and the vegetation canopy are assumed to radiate as perfect black bodies and, thus, have an emissivity of 1.0. Emissivity of the air (`emis`) is a function of its moisture content and ranges between 0.757 and 1.0 (U.S. Army, 1956). For days without precipitation, `emis` is user specified as `emis_noppt`. For days with precipitation, `emis` is computed separately for the day and night periods as a function of storm type and observed solar radiation. A period of months during which storms are predominantly convective in origin can be user-specified by setting the `tstorm_mo` parameter to 1 for each month in the period. Storms occurring outside these months are assumed to be frontal in origin with an associated `emis` of 1.0 for both day and night periods. During the convective storm period, days with precipitation are assumed to have an `emis` of 0.85 for the night period. For the associated day period, `emis` is assumed to vary between `emis_noppt` and 1.0 as a function of the ratio of observed to potential solar radiation received on a horizontal surface. If solar radiation is computed from temperature data, the day period `emis` is assumed to be 1.0.

The full equation for computing latent and sensible heat flux includes terms for temperature, vapor pressure, wind speed, and diffusivities of heat and vapor (U.S. Army, 1956). However, wind and vapor pressure or humidity data generally are not available. Therefore, the full equation is simplified to a temperature index form to estimate latent and sensible heat terms. It is computed by:

$$\mathrm{cecsub} = \mathrm{cec(mo)} \times \mathrm{temp}$$

where

cecsub
: an estimate of latent and sensible heat,

cec
: a parameter for month mo, and

temp
: the mean air temperature for the 12-hour period (°C).

To provide a measure of the influence of wind in the full equation for cecsub, areas of forest cover are assumed to receive only one-half of cecsub computed. Vapor pressure influences are considered by computing cecsub only on days of rainfall or when the ratio of observed to potential shortwave solar radiation is less than or equal to 0.33.

When the 12-hour energy balance (`cal`) is negative, heat flow occurs by conduction only. The amount and direction of heat flow is computed from the `qcond` equation above. Snowpack water equivalent and energy conditions are recomputed by method `calin` when the heat flow is from the surface to the pack, and by method `caloss` when the heat flow is from the pack to the surface. When the 12-hour energy balance is positive, this energy is assumed to melt snow in the surface layer. Snowmelt transports heat into the snowpack by mass transfer. The amount of snowmelt and subsequent thermal and water equivalent changes in the snowpack are computed in method `calin`.

Method `calin`

This subroutine computes the change in snowpack thermal and water equivalent conditions resulting from a net gain of energy (`cal` is positive). If `pk_temp` is less than 0 °C, that is `pk_def` is greater than zero, then `cal` is first used to satisfy `pk_def`. If `cal` is less than `pk_def`, `pk_def` is reduced by `cal` and a new `pk_temp` is computed using the pk_temp equation above. When pk_def reaches zero, any remaining calories in cal are used to melt snow. Snowmelt (pmelt) is computed by:

$$pmelt = \frac{cal}{203.2}$$

where 203.2 is the number of calories required to melt 1 inch of water- equivalent ice at 0 °C.

pkwater_equiv is the sum of two components, ice (pk_ice) and liquid water (freeh2o). When the temperature of the lower snowpack reaches 0 °C, snowmelt is removed from pk_ice and is first used to satisfy the free water-holding capacity (freeh2o_cap) of the snowpack. freeh2o_cap ranges from 2 to 5 percent of the water equivalent of the snowpack existing in the ice phase (U.S. Army, 1956; Leaf, 1966) and is user-specified. Once freeh2o_cap is satisfied, the remaining snowmelt moves out of the snowpack and becomes available for infiltration and surface runoff.

The volume of snowmelt is computed as a function of the snow-covered area of an HRU. pmelt is multiplied times a value of snow-covered area (snowcov_area) that is computed in method snowcov.

Method caloss

This method computes the change in snowpack thermal and water equivalent conditions resulting from a net loss of energy (cal is negative). When pk_temp is warmer than ts, heat is conducted from the lower snowpack to the snowpack surface. The amount of heat conducted to the surface is computed using the qcond equation above. If pk_temp is 0 °C, the heat loss from the lower snowpack first causes any freeh2o held in the pack to freeze. The calorie loss (calnd) required to freeze the freeh2o is computed by:

$$calnd = freeh2o \times 203.2$$

Any loss available after freeh2o is frozen is accumulated as a heat deficit in variable pk_def, and a new temperature for the lower snowpack is computed using the pk_temp equation above.

Method snowevap

Evaporation and sublimation from the snow surface are assumed to occur only when there is no transpiration occurring from vegetation above the snowpack. The loss from the snow surface (snow_evap) is computed daily as a percentage of the computed potential evapotranspiration (potet). The equation used is:

$$snow\_evap = (potet\_sublim \times potet) - (intcp\_evap \times cov)$$

where

potet_sublim
: a loss coefficient,

intcp_evap
: the evaporation and sublimation loss from interception (in.), and

cov
: the vegetation cover density for the date of computation.

If snow_evap is less than zero, then snow_evap is assumed to be zero. The volume of snow_evap is also computed as a function of the snow-covered area of an HRU. snow_evap is multiplied times a value of snow-covered area (snowcov_area) that is computed in method snowcov.

Method snowcov

This method computes the snow-covered area of an HRU using an approach developed by Anderson (1973). Snow-covered area is computed using a user-defined areal depletion curve that is a plot of the areal extent of snow cover versus the ratio of pkwater_equiv to an index value, ai. ai is the smaller of either the maximum of pkwater_equiv since snow began to accumulate or the parameter `snarea_thresh`. `snarea_thresh` is the value of `pkwater_equiv` above which an HRU is 100 percent snow covered.

Up to 10 separate areal depletion curves may be defined. Each curve is input as a set of 11 areal coverage values, one for each `ai` value ranging from 0.0 to 1.0 in increments of 0.1. An `ai` value is computed daily and a corresponding areal coverage value, `snowcov_area`, is computed by interpolation from the areal depletion curve defined for each HRU.

If new snow occurs during periods when `ai` is less than 1.0, `snowcov_area` is increased to 100 percent and remains at 100 percent until 25 percent of the new snow melts. When 25 percent has melted, `snowcov_area` is then decreased linearly in successive time steps as a function of the ratio of melt water equivalent to 75 percent of the water equivalent in the new snow.

1.11.5.2. References

Anderson, E.A., 1968, Development and testing of snowpack energy balance equations: Water Resources Research, v. 4, n. 1, p. 19-38.

Anderson, E.A., 1973, National Weather Service River Forecast System - Snow accumulation and ablation model: U.S. Department of Commerce, NOAA Technical Memorandum NWS Hydro-17.

Leaf, C.F., 1966, Free water content of snowpack in subalpine areas: Western Snow Conference, Seattle, Wash., 1966, Proceedings, v. 34, p.17-24.

Leavesley, G.H., and Striffler, W.D., 1978, A mountain watershed simulation model, in Colbeck, S.C., and Ray, M., eds., Modeling of snow cover runoff, Hanover, New Hampshire, 1978, Proceedings: U.S. Army Corps of Engineers, Cold Regions Engineering and Research Laboratory, p. 379-386.

Miller, D.H., 1959, Transmission of insolation through pine forest canopy as it effects the melting of snow: Mitteilungen der Schweizerischen Anstalt für das forstliche Versuchwesen, Versuchsw. Mitt., v. 35, p35-79.

Obled, Charles, and Rosse, B.B., 1977, Mathematical models of a melting snowpack at an index plot: Journal of Hydrology, no. 32, p. 139-163.

Riley, J.P., Israelsen, E.K., and Eggleston, K.O., 1973, Some approaches to snowmelt prediction, in The role of snow and ice in hydrology: International Association of Hydrological Sciences Publication 107, p. 956-971.

U.S. Army, 1956, Snow hydrology: Portland, Oreg., U.S. Army Corps of Engineers, North Pacific Division, 437 p.

Vézina, P.E., and Péch, G.Y., 1964, Solar radiation beneath conifer canopies in relation to crown closure: Forest Science, v. 10, no. 4, p. 443-451.

Willen, D.W., Schumway, C.A., and Reid, J.E., 1971, Simulation of daily snow water equivalent and melt: Western Snow Conference, Billings, Montana, 1971, Proceedings, v. 39, p. 1-8.

1.12. Component 'Soltab'

Solar Radiation Computation.This module computes the potential solar radiation and the sunrise and sunset times for a horizontal surface and for any slope/aspect combination.

Name

`prms2008.Soltab`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Radiation

Version
$Id: Soltab.java 1055 2010-03-11 18:28:30Z odavid $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Soltab.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.12.1. Parameter

ndays - `int`
Number of HRUs.

nradpl - `int`
Number of radiation planes.

radpl_aspect `[degrees] - double[]`
Aspect for each radiation plane

radpl_lat `[degrees] - double[]`
Latitude of each radiation plane

radpl_slope `[decimal percent] - double[]`
Slope of each radiation plane, specified as change in vertical length divided by change in horizontal length

1.12.2. Variables (Out)

basin_lat `[degrees] - double`
Latitude of the basin, computed as an average of the radiation plane latitudes.

hemisphere - `int`
Flag to indicate in which hemisphere the model resides (0=Northern; 1=Southern)

radpl_cossl - `double[]`
Cosine of each radiation plane slope

radpl_soltab `[langleys] - double[][]`
Potential daily shortwave radiation for each radiation plane

sunhrs_soltab `[hours] - double[][]`
Hours between sunrise and sunset for each radiation plane

1.12.3. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.12.4. Soltab

1.12.4.1. Description

Tables consisting of daily estimates of the potential (clear sky) short-wave solar radiation for each radiation plane are computed on the basis of hours between sunrise and sunset for each Julian day of the year. The potential short-wave solar radiation also is computed for each Julian day of the year for a horizontal plane at the surface of the centroid of the modeled basin. The computations of the solar tables use double precision constants, such as decimal days per year, rotational degrees per day, eccentricity of the Earth’s orbit, and the number pi and constants based on pi such as radians and radians per year.

Daily estimates of obliquity are computed from (Meeus, 1999):

Equation 2.10.

$$E^{m} = 1.0 - \left[\mathrm{EC} \times \cos(\mathrm{jd} - 3.0) \times \mathrm{rad}\right]$$

where

E^m
: the obliquity of the Sun's ecliptic for time step m in angular degrees,

EC
: the eccentricity of the Earth's orbit (~0.01671), in radians,

jd
: the Julian day number (3 is subtracted as the solar year begins on December 29) in days, and

rad
: the revolution speed of the Earth (~0.0172) in radians per day.

Daily estimates of solar declination are computed from (Meeus, 1999):

Equation 2.11.

$$\mathrm{DM}^{m} = 0.006918 - 0.399912\cos(E_{\mathrm{rt}}) + 0.070257\sin(E_{\mathrm{rt}}) - 0.006758\cos(E_{\mathrm{rt}}) + 0.000907\sin(2E_{\mathrm{rt}}) - 0.002697\cos(3E_{\mathrm{rt}}) + 0$$

where

DM^m
: the solar declination for time step m, in angular degrees

E_{rt}
: `rad` * (`jd`-1)

Sunset and sunrise times are computed for each radiation plane with a three-step procedure. First, the sunset and sunrise times are computed for a horizontal plane at the centroid of the radiation plane. Next, a horizontal surface on the terrestrial spheroid is found which is parallel to the slope and aspect of the surface of the radiation plane. This is called the equivalent-slope surface (Lee, 1963). Sunset and sunrise times also are computed for this surface. Finally, sunset time on the sloped surface of the radiation plane is taken as the earliest of the computed horizontal surface sunset times of the radiation plane and equivalent-slope surface. Likewise, sunrise time is taken as the latest of the computed horizontal surface sunrise times. Daylight length for each radiation plane is computed from these sunset and sunrise times.

The angle between the local meridian and the sunset (or sunrise) meridian, referred to as the hour angle of sunset (or sunrise), for the horizontal surface of both the radiation plane and equivalent-slope surface is calculated according to (Swift, 1976):

Equation 2.12.

$$ss^m = \cos^{-1}\left[-\tan(lat)\tan\left(DM^m\right)\right]$$
$$sr^m = -ss^m$$
$$sunhrs\_soltab^m_{RP} = \frac{\left(ss^m_{RP} - sr^m_{RP}\right)24}{2\Pi}$$

where

ss^m
: the hour angle of sunset, measured from the local meridian of a horizontal surface (radiation plane or equivalent-slope surface) for time step m, in radians;

sr^m
: the hour angle of sunrise, measured from the local meridian of a horizontal surface (radiation plane or equivalent-slope surface) for time step m, in radians;

ss^m_{RP}
: the hour angle of sunset on the sloped surface of the radiation plane for time step m, in radians;

sr^m_{RP}
: the hour angle of sunrise on the sloped surface of the radiation plane for time step m ,in radians;

$sunhrs\_soltab^m_{RP}$
: the daylight length on the radiation plane for time step m, in hours;

#
: the constant pi (~3.1415926535898) dimensionless; and

lat
: the latitude of the horizontal surface (basin centroid parameter `basin_lat`, radiation plane centroid parameter `radpl_lat` or equivalent-slope surface), positive values are in the northern hemisphere and negative values are in the southern hemisphere,in radians.

Daily estimates of potential solar radiation (`radpl_soltab`) for each radiation plane are calculated as described in Frank and Lee (1966), and Swift (1976):

Equation 2.13.

$$radpl\_soltab^m_{RP} = sc^m\left(c1_{PSR} + c2_{PSR}\right)$$
$$c1_{PSR} = \sin\left(DM^m\right)\sin\left(lat'_{RP}\right)sunhrs\_soltab^m_{RP}$$
$$c2_{PSR} = \left\{\cos\left(DM^m\right)\cos\left(lat'_{RP}\right)\left[\sin\left(ss^m_{RP} + long'_{RP}\right) - \sin\left(sr^m_{RP} + long'_{RP}\right)\right]24.0\right\}/2\Pi$$

where

$radpl\_soltab^m_{RP}$
: the potential solar radiation on the radiation plane (RP) during time step m, in calories per square centimeter per day;

lat'_{RP}
: the latitude of the equivalent-slope surface of the radiation plane, in radians;

$long'_{RP}$
: the longitude offset between the equivalent-slope surface and the radiation plane, in radians; and

sc^m
: the 60-minute period solar constant for time step m, in calories per square centimeter per hour.

1.12.4.2. References

Frank, E.C., and Lee, R., 1966, Potential solar beam irradiation on slopes: U. S. Department of Agriculture, Forest Service Research Paper RM-18, 116 p.

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

Lee, R., 1963, Evaluation of solar beam irradiation as a climatic parameter of mountain watersheds: Colorado State University Hydrology Papers, no. 2, 50 p.

Meeus, J., 1999, Astronomical Algorithms: Richmond, Va., Willmann-Bell, Inc., 477 p.

Swift, Lloyd W., Jr., 1976, Algorithm for solar radiation on mountain slopes: Water Resources Research, v. 12, no. 1, p. 108-112.

1.13. Component 'SrunoffSmidx'

Surface runoff.Computes surface runoff and infiltration for each HRU usinga non-linear variable-source-area method.

Name
: `prms2008.SrunoffSmidx`

Author
: George H. Leavesley - ghleavesley@colostate.edu

Keyword
: Runoff, Surface

Version
: $Id: SrunoffSmidx.java 1128 2010-04-07 19:43:29Z ghleavesley $

Source
: $URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/SrunoffSmidx.java $

License
: http://www.gnu.org/licenses/gpl-2.0.html

1.13.1. Parameter

carea_max `[decimal fraction] - double[]`
: Maximum contributing area Maximum possible area contributing to surface runoff expressed as a portion of the HRU area

hru_area `[acres] - double[]`
: Area of each HRU

hru_type `- int[]`
: Type of each HRU (0=inactive; 1=land; 2=lake; 3=swale)

imperv_stor_max `[inches] - double[]`
HRU maximum impervious area retention storage Maximum impervious area retention storage for each HRU

nhru `- int`
Number of HRUs.

smidx_coef `[decimal fraction] - double[]`
Coefficient in contributing area computations Coefficient in non-linear contributing area algorithm. Equation used is: contributing area = smidx_coef * 10.**(smidx_exp*smidx) where smidx is soil_moist + .5 * ppt_net

smidx_exp `[1/inch] - double[]`
Exponent in contributing area computations Exponent in non-linear contributing area algorithm. Equation used is: contributing area = smidx_coef * 10.**(smidx_exp*smidx) where smidx is soil_moist + .5 * ppt_net

snowinfil_max `[inches/day] - double[]`
Maximum snow infiltration per day Maximum snow infiltration per day

soil_moist_init `[inches] - double[]`
Initial value of available water in soil profile

soil_moist_max `[inches] - double[]`
Maximum value of water for soil zone Maximum available water holding capacity of soil profile. Soil profile is surface to bottom of rooting zone

1.13.2. Variables (In)

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

hru_imperv `[acres] - double[]`
HRU impervious area. [basin]

hru_intcpevap `[inches] - double[]`
Evaporation from interception on each HRU

hru_percent_impv `[decimal fraction] - double[]`
Proportion of each HRU area that is impervious

hru_perv `[acres] - double[]`
HRU pervious area. [basin]

hru_ppt `[inches] - double[]`
Precipitation on HRU, rain and snow. [precip]

hru_route_order `- int[]`
Routing order for HRUs

net_ppt `[inches] - double[]`
HRU precipitation (rain and/or snow) with interception removed

net_rain `[inches] - double[]`
Rain on an HRU (hru_rain) minus interception. [intcp]

net_snow `[inches] - double[]`
Snow on an HRU (hru_snow) minus interception. [intcp]

pkwater_equiv `[inches] - double[]`
Snowpack water equivalent on an HRU. [snow]

potet `[inches] - double[]`
Potential evapotranspiration for each HRU. [potet]

pptmix_nopack `- int[]`
Indicator that a rain-snow mix event has occurred with no snowpack present on an HRU. [snow]

route_on `- int`
Kinematic routing switch (0=daily; 1=storm period)

snow_evap `[inches] - double[]`
Evaporation and sublimation from snowpack on an HRU. [snow]

snowcov_area `[decimal fraction] - double[]`
Snow-covered area on an HRU, in decimal fraction of total HRU area. [snow]

snowmelt `[inches] - double[]`
Snowmelt from snowpack on an HRU. [snow]

soil_moist `[inches] - double[]`
Pseudo parameter. Soil moisture content for each HRU. [smbal]

1.13.3. Variables (Out)

basin_imperv_evap `[inches] - double`
Basin area-weighted average for evaporation from impervious area

basin_imperv_stor `[inches] - double`
Basin area-weighted average for storage on impervious area

basin_infil `[inches] - double`
Basin area-weighted average for infiltration

basin_sroff `[inches] - double`
Basin area-weighted average of surface runoff

dprst_evap_hru `[inches] - double[]`
Evaporation from depression storage for each HRU

dt_sroff `[inches] - double`
Total basin surface runoff for a storm timestep

hru_impervevap `[inches] - double[]`
Evaporation from impervious area for each HRU

hru_impervstor `[inches] - double[]`
Storage on impervious area for each HRU

imperv_evap `[inches] - double[]`
Evaporation from impervious area

imperv_stor `[inches] - double[]`
Current storage on impervious area for each HRU

infil `[inches] - double[]`
: Amount of water infiltrating the soil on each HRU

sroff `[inches] - double[]`
: Surface runoff to streams for each HRU

upslope_hortonian `[inches] - double[]`
: Hortonian surface runoff received from HRUs upslope

1.13.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.13.5. SrunoffSmidx

1.13.5.1. Description

This module computes the surface runoff for rain or snowmelt on pervious and impervious areas and keeps track of the retention storage on impervious areas.

Surface runoff from rainfall on pervious, snow-free HRU's is computed using a contributing area concept (Dickenson and Whitely, 1970; Hewlett and Nutter, 1970). The percent of an HRU contributing to surface runoff is computed as a non-linear function of antecedent soil moisture and rainfall amount. The contributing area, `ca_percent` is computed by:

$$\text{ca\_percent} = \text{smidx\_coef} \times 10.^{(\text{smidx\_exp} \times \text{smidx})}$$

where

smidx_coef
: the coefficient in non-linear contributing area algorithm, and

smidx_exp
: the exponent in non-linear contributing area algorithm.

`smidx` is computed by:

$$\text{smidx} = \text{soil\_moist} \times \frac{\text{net\_rain}}{2.0}$$

where

soil_moist
: the soil moisture content for each HRU, in inches, and

net_rain
: the rain on an HRU minus interception, in inches.

A maximum value is specified for `ca_percent` using the parameter `carea_max`. Surface runoff for the pervious area (`srp`) is then computed as:

$$\text{srp} = \text{ca\_percent} \times \text{net\_rain}$$

Infiltration (`infil`) is computed by:

$$\text{infil} = \text{net\_rain} - \text{srp}$$

Estimates of smidx_coef, smidx_exp, carea_max, and direct-surface runoff can be made from observed runoff and soil-moisture data. Where soil moisture data are not available, estimates of soil-moisture values can be obtained from preliminary model runs. A regression of log ca_percent versus smidx can be done for these data to determine the coefficients. Using the equation:

$$\log(\text{ca\_percent}) = a + b \times \text{smidx}$$

then

$$\text{smidx\_coef} = 10.^{a}$$
$$\text{smidx\_exp} = b$$

Surface runoff from snowmelt on pervious areas is assumed to occur only when the soil zone of an HRU reaches field capacity. At field capacity, maximum infiltration amount, snowinfil_max, is user defined. Any snowmelt in excess of snowinfil_max becomes surface runoff. Snowmelt generated by rain on a snowpack is treated as all snowmelt if the snowpack is not totally depleted by the rain. If the snowpack is totally depleted by the rain, the resulting rain and snowmelt mix is treated as if it were all rain on a snow-free HRU.

Surface runoff from impervious areas is computed identically for both rainfall and snowmelt in subroutine imperv_sroff. Total surface runoff for each HRU is then computed by taking an area-weighted average of the pervious and impervious area surface runoff. This subroutine also computes a basin weighted average for surface runoff and infiltration. Evaporation from impervious areas is computed in subroutine imperv_et.

On impervious areas, rainfall or snowmelt first satisfies available retention storage, and the remainder becomes surface runoff. Available retention storage (avail_stor) is computed by:

$$\text{avail\_stor} = \text{imperv\_stor\_max} - \text{imperv\_stor}$$

where

imperv_stor
: the storage on impervious area, in inches, and

imperv_stor_max
: the maximum impervious area retention storage for HRU, in inches.

Impervious area retention storage, imperv_stor, on snow-free HRUs is depleted by evaporation, imperv_evap, at the potential rate, potet. On snow-covered HRUs, the evaporation on impervious areas is set equal to snow_evap.

1.13.5.2. References

Dickenson, W. T., and Whitely, H. Q., 1970, Watershed areas contributing to runoff: International Association of Hydrologic Sciences Publication 96, p. 1.12-1.28.

Hewlett, J. D., and Nutter, W. L., 1970, The varying source area of streamflow from upland basins, in Symposium on Interdisciplinary Aspects of Watershed Management, Montana State University, Bozeman, Montana, 1970, Proceedings, p. 65-83.

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.14. Component 'Ssflow'

Subsurface flow.Adds inflow to subsurface reservoirs and computesoutflow to groundwater and to streamflow.

Name
`prms2008.Ssflow`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Runoff, Subsurface

Version
$Id: Ssflow.java 861 2010-01-21 01:54:38Z ghleavesley $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Ssflow.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.14.1. Parameter

basin_area `[acres] - double`
Total basin area [basin]

frozen `- int[]`
Flag for frozen ground (0=no; 1=yes)

hru_area `[acres] - double[]`
Area of each HRU

hru_ssres `- int[]`
Index of subsurface reservoir receiving excess water from HRU soil zone

hru_type `- int[]`
Type of each HRU (0=inactive; 1=land; 2=lake; 3=swale)

nhru `- int`
Number of HRUs.

nssr `- int`
Number of subsurface reservoirs.

ssr2gw_exp `- double[]`
Coefficient in equation used to route water from the subsurface reservoirs to the groundwater reservoirs: ssr_to_gw = ssr2gw_rate * ((ssres_stor / ssrmax_coef)**ssr2gw_exp); recommended value is 1.0

ssr2gw_rate `[1/day] - double[]`
Coefficient to route water from subsurface to groundwater Coefficient in equation used to route water from the subsurface reservoirs to the groundwater reservoirs: ssr_to_gw = ssr2gw_rate * ((ssres_stor / ssrmax_coef)**ssr2gw_exp)

ssrcoef_lin `[1/day] - double[]`
Coefficient to route subsurface storage to streamflow using the following equation: ssres_flow = ssrcoef_lin * ssres_stor + ssrcoef_sq * ssres_stor**2

ssrcoef_sq `- double[]`
Coefficient to route subsurface storage to streamflow using the following equation: ssres_flow = ssrcoef_lin * ssres_stor + ssrcoef_sq * ssres_stor**2

ssrmax_coef `[inches] - double[]`
Coefficient in equation used to route water from the subsurface reservoirs to the groundwater reservoirs: ssr_to_gw = ssr2gw_rate * ((ssres_stor / ssrmax_coef)**ssr2gw_exp); recommended value is 1.0

ssstor_init `[inches] - double[]`
Initial storage in each subsurface reservoir; estimated based on measured flow

1.14.2. Variables (In)

active_hrus `- int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

deltim `[hours] - double`
Length of the time step

hru_perv `[acres] - double[]`
HRU pervious area. [basin]

hru_route_order `- int[]`
Routing order for HRUs

soil_to_ssr `[inches] - double[]`
The amount of water transferred from the soil zone to a subsurface reservoir for each HRU. [smbal]

ssres_area `[acres] - double[]`
Subsurface reservoir area.

1.14.3. Variables (Out)

basin_ssflow `[inches] - double`
Basin weighted average for subsurface reservoir outflow

basin_ssin `[inches] - double`
Basin weighted average for inflow to subsurface reservoirs

basin_ssr2gw `[inches] - double`
Basin average drainage from soil added to groundwater

basin_ssstor `[inches] - double`
Basin weighted average for subsurface reservoir storage

basin_ssvol `[acre-inches] - double`
Basin weighted average for subsurface reservoir storage volume

ssr_to_gw `[inches] - double[]`
Seepage from subsurface reservoir storage to its associated groundwater reservoir each time step

ssres_flow `[inches] - double[]`
Outflow from each subsurface reservoir

ssres_in `[inches] - double[]`
Sum of inflow to subsurface reservoir from all associated HRUs

ssres_stor `[inches] - double[]`
Storage in each subsurface reservoir

1.14.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.14.5. Ssflow

1.14.5.1. Description

Subsurface flow is considered to be the relatively rapid movement of water from the unsaturated zone to a stream channel. Subsurface flow occurs during, and for a period after, rainfall and snowmelt. The source of subsurface flow is soil water in excess of field capacity. This excess percolates to shallow groundwater zones or moves downslope at shallow depths from point of infiltration to some point of discharge above the water table. Subsurface flow in this module is computed using a reservoir routing system.

A subsurface reservoir may receive inflow from one or several HRUs. The number of subsurface reservoirs delineated in a basin can range from one to the number of HRUs delineated. The subsurface reservoir associated with an HRU is designated by the parameter, `hru_ssres`. Inflow to a subsurface reservoir, `ssres_in`, is determined by summing the `soil_to_ssr` values from all of the HRU's associated with the reservoir.

The continuity of mass equation for the subsurface-flow system is expressed as:

$$\mathrm{ssres\_flow} = \mathrm{ssres\_in} - \frac{d\,(\mathrm{ssres\_stor})}{d\,t}$$

where

ssres_flow
: the contribution to streamflow from each subsurface reservoir,

ssres_in
: the total inflow to each subsurface reservoir, and

ssres_stor
: the storage in each subsurface reservoir.

The variable `ssres_flow` can be expressed in terms of `ssres_stor` using the relationship:

$$\mathrm{ssres\_flow} = (\mathrm{ssrcoef\_lin} \times \mathrm{ssres\_stor}) + (\mathrm{ssrcoef\_sq} \times \mathrm{ssres\_stor}^{2})$$

where

ssrcoef_lin
: the linear subsurface routing coefficient to route subsurface storage to streamflow, and

ssrcoef_sq
: the non-linear subsurface routing coefficient to route subsurface storage to streamflow.

This is a non-linear relationship which can be made linear, if desired, by setting `ssrcoef_sq` equal to 0.

One reason for selecting a second degree polynomial equation for computing subsufrace flow is that the equation has an analytical solution for application at any time step t. This makes the equation applicable for both daily mode and storm mode computations in PRMS.

Substituting the equation above into the continuity of mass equation above produces:

$$\frac{d\,(\text{ssres\_stor})}{d\,t} = \text{ssres\_in} - (\text{ssrcoef\_lin} \times \text{ssres\_stor}) - (\text{ssrcoef\_sq} \times \text{ssres\_stor}^{2})$$

This equation is solved for `ssres_stor` with the initial conditions `ssres_stor` = $\texttt{ssres\_stor}_0$.

This solution is combined with the continuity equation, producing;

$$\text{ssres\_flow} \times \Delta t = \text{ssres\_in} \times \Delta t + \text{SOS} \times \frac{\left[1.0 + \frac{\text{ssrcoef\_sq}}{\text{XK3}} \times \text{SOS}\right] \times \left(1.0 - e^{(-\text{XK3} \times \Delta t)}\right)}{1.0 + \frac{\text{ssrcoef\_sq}}{\text{XK3}} \times \text{SOS} \times \left(1.0 - e^{(-\text{XK3} \times \Delta t)}\right)}$$

where

#t is the time interval,

$$\text{SOS} = \text{ssres\_stor}_{0} - \left[\frac{\text{XK3} - \text{ssrcoef\_lin}}{2.0 \times \text{ssrcoef\_sq}}\right]\text{, and}$$

$$\text{XK3} = \sqrt{(\text{ssrcoef\_lin})^{2} + 4.0 \times \text{ssrcoef\_sq} \times \text{ssres\_in}}$$

A second discharge point from the subsurface reservoir provides recharge, `ssr_to_gw`, to the groundwater reservoir, which is computed as:

$$\text{ssr\_to\_gw} = \text{ssrgw\_rate} \times \left[\frac{\text{ssres\_stor}}{\text{ssr2gw\_max}}\right]^{\text{ssr2gw\_exp}}$$

where

ssr2gw_exp and ssr2gw_rate
: coefficients to route water from subsurface to groundwater, and

ssr2gw_max
: maximum value for water routed from subsurface to groundwater.

The coefficients `ssr2gw_max` and `ssr2gw_exp` are used to define the routing characteristics of `ssr_to_gw`. Setting them equal to 1.0 makes the routing a linear function of `ssres_stor`.

Initial storage volumes and routing coefficients must be determined for each subsurface reservoir. The initial estimate of storage normally is zero. Values of `ssrcoef_lin` and `ssrcoef_sq` can be fitted from historic streamflow data. For the nonlinear routing scheme, there are no procedures currently developed for making initial estimates of `ssrcoef_lin` and `ssrcoef_sq`. However, for the linear case, `ssrcoef_lin` can be estimated using the hydrograph separation technique on semilogarithmic paper described by Linsley, Kohler, and Paulhus (1958). Integrating the characteristic depletion equation:

$$q_{t} = q_{0} \times \text{Kr}^{t}$$

where

qt and q0
: streamflow at times t and 0, and

Kr
: a recession constant.

They show a relationship between `ssres_flow` and `ssres_stor` that is expressed as:

$$ssres\_stor = -\frac{ssres\_flow}{\log_e Kr}$$

where `Kr` is the slope of the subsurface flow recession obtained from the semilog plot for t=1 day.

Rewriting equation 7 as:

$$ssres\_flow = -\log_e Kr \times ssres\_stor$$

shows that `-logeKr` is equivalent to `ssrcoef_lin` in the computation of `ssres_flow` above when `ssrcoef_sq` is zero.

This module also computes weighted averages for `ssres_stor`, `ssres_in` and `ssres_flow` for the basin.

1.14.5.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

Linsley, R. K., JR., Kohler, M. A., and Paulhus, J. L., 1958, Hydrology for Engineers: New York, McGraw-Hill, p.151-155.

1.15. Component 'Strmflow'

Calculates daily streamflow, individual storm flows, and daily reservoir routing.Procedures to compute (1) daily streamflow as the sum of surface, subsurface,and ground-water flow contributions, (2) storm runoff totals for storm periods,and (3) daily reservoir routing.

Name
    `prms2008.Strmflow`

Author
    George H. Leavesley - ghleavesley@colostate.edu

Keyword
    Runoff

Version
    $Id: Strmflow.java 1293 2010-06-07 21:58:43Z ghleavesley $

Source
    $URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Strmflow.java $

License
    http://www.gnu.org/licenses/gpl-2.0.html

1.15.1. Parameter

basin_area `[acres] - double`
    Total basin area

hru_area `[acres] - double[]`
    HRU area

ngw `- int`
    Number of Ground water reservoirs.

nhru - `int`
Number of HRUs.

nssr - `int`
Number of subsurface reservoirs.

1.15.2. Variables (In)

active_hrus - `int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

basin_gwflow `[inches] - double`
Basin area-weighted average for ground-water flow [gwflow]

basin_sroff `[inches] - double`
Basin area-weighted average of surface runoff [srunoff]

basin_ssflow `[inches] - double`
Basin area-weighted average for subsurface flow [ssflow]

deltim `[hours] - double`
Length of the time step

hru_route_order - `int[]`
Routing order for HRUs

route_on - `int`
Kinematic routing switch - 0= non storm period, 1=storm period [obs]

1.15.3. Variables (Out)

basin_cfs `[cfs] - double`
Streamflow from basin

basin_cms `[cms] - double`
Streamflow from basin

basin_gwflow_cfs `[cfs] - double`
Basin ground-water flow for timestep

basin_sroff_cfs `[cfs] - double`
Basin surface runoff for timestep

basin_ssflow_cfs `[cfs] - double`
Basin subsurface flow for timestep

basin_stflow `[inches] - double`
Sum of basin_sroff, basin_ssflow and basin_gwflow for timestep

1.15.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.15.5. Strmflow

1.15.5.1. Description

For daily timesteps, simulated streamflow at a basin outflow point, `basin_cfs`, is computed as:

$$basin\_cfs = basin\_sroff + basin\_ssflow + basin\_gwflow$$

where

basin_sroff
: basin area-weighted average of surface runoff, (in),

baisn_ssflow
: basin area-weighted average for subsurface flow, (in), and

basin_gwflow
: basin area-weighted average for ground-water flow, (in).

Basin streamflow in inches (`basin_stflow`) is computed by converting `basin_cfs` to inches depth over the basin.

For storm-mode computations with timesteps of 60 minutes or less, `basin_cfs` for the simulation timestep is computed as:

$$basin\_cfs = qchan(outlet\_chan)$$

where `qchan(outlet_chan)` is the timestep flow rate computed for the channel segment at the outlet of the basin. `qchan` for each channel segment is computed in the `krout_chan` component.

1.15.5.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.16. Component 'Temp1sta'

Temperature distribution.Distributes temperatures to HRU's using temperature data measured at a stationand a monthly parameter based on the lapse rate with elevation.

Name
: `prms2008.Temp1sta`

Author
: George H. Leavesley - ghleavesley@colostate.edu

Keyword
: Temperature

Version
: $Id: Temp1sta.java 861 2010-01-21 01:54:38Z ghleavesley $

Source
: $URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/Temp1sta.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.16.1. Parameter

basin_tsta - `int`
Index of main temperature station Index of temperature station used to compute basin temperature values

hru_area `[acres] - double[]`
Area of each HRU

hru_elev `[elev_units] - double[]`
Mean elevation for each HRU

hru_tsta - `int[]`
Index of the base temperature station used for lapse rate calculations

nhru - `int`
Number of HRUs.

ntemp - `int`
Number of temperature stations.

temp_units - `int`
Units for measured temperature (0=Fahrenheit; 1=Celsius)

tmax_adj `[degrees] - double[]`
Adjustment to maximum temperature for each HRU, estimated based on slope and aspect

tmax_lapse `[degrees] - double[]`
Array of twelve values representing the change in maximum temperature per 1000 elev_units of elevation change for each month, January to December

tmin_adj `[degrees] - double[]`
Adjustment to minimum temperature for each HRU, estimated based on slope and aspect

tmin_lapse `[degrees] - double[]`
Array of twelve values representing the change in minimum temperture per 1000 elev_units of elevation change for each month, January to December

tsta_elev `[elev_units] - double[]`
Elevation of each temperature measurement station

1.16.2. Variables (In)

active_hrus - `int`
Number of active HRUs

basin_area_inv `[1/acres] - double`
Inverse of total basin area as sum of HRU areas

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

hru_route_order - `int[]`
Routing order for HRUs

route_on - `int`
Kinematic routing switch - 0= non storm period, 1=storm period [obs]

tmax - `double[]`
Measured maximum temperature at each temperature measurement station, F or C depending on units of data. [obs]

tmin - `double[]`
Measured minimum temperature at each temperature measurement station, F or C depending on units of data. [obs]

1.16.3. Variables (Out)

basin_temp `[degrees] - double`
Basin area-weighted temperature for timestep < 24

basin_tmax `[degrees] - double`
Basin area-weighted daily maximum temperature

basin_tmin `[degrees] - double`
Basin area-weighted daily minimum temperature

solrad_tmax `[degrees] - double`
Basin daily maximum temperature for use with solrad radiation

solrad_tmin `[degrees] - double`
Basin daily minimum temperature for use with solrad radiation

tavgc `[degrees Celsius] - double[]`
HRU adjusted daily average temperature

tavgf `[degrees F] - double[]`
HRU adjusted daily average temperature

tempc `[degrees Celsius] - double[]`
HRU adjusted temperature for timestep < 24

tempf `[degrees F] - double[]`
HRU adjusted temperature for timestep < 24

tmaxc `[degrees Celsius] - double[]`
HRU adjusted daily maximum temperature

tmaxf `[degrees F] - double[]`
HRU adjusted daily maximum temperature

tminc `[degrees Celsius] - double[]`
HRU adjusted daily minimum temperature

tminf `[degrees F] - double[]`
HRU adjusted daily minimum temperature

1.16.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.16.5. Temp1sta

1.16.5.1. Description

Measured daily maximum (`tmax`) and minimum (`tmin`) air temperatures are adjusted to account for differences in elevation and slope-aspect between the climate station and each HRU. Air temperature data and the adjustment parameters must be in the same units as were declared in the `` `basin' `` component. If the computation time step is less than daily, then `tmax` should equal `tmin` and represent the average temperature for the time step.

The monthly correction factors for maximum (`tcrx`) and minimum (`tcrn`) temperature for each HRU are computed by:

$$tcrx = (tmax\_lapse \times elfac) - tmax\_adj$$
$$tcrn = (tmin\_lapse \times elfac) - tmin\_adj$$

where

tmax_lapse
: the monthly maximum temperature lapse rate,

tmin_lapse
: the monthly minimum temperature lapse rate,

tmax_adj
: the HRU maximum temperature adjustment, and

tmin_adj
: the HRU minimum temperature adjustment,

`elfac` is an elevation effect correction factor that is computed as:

$$elfac = hru\_elev - \frac{tsta\_elev}{1000.}$$

where

hru_elev
: the median elevation for each HRU, and

tsta_elev
: the elevation of each temperature measurement station.

The temperatures for each time step are then computed by subtracting `tcrx` and `tcrn` from `tmax` and `tmin` respectively. All temperatures are computed in both °F and °C units for use by other components.

This component also computes a weighted average maximum (`basin_tmax`)and minimum (`basin_tmin`) temperature for the basin for each time step

1.16.5.2. References

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.17. Component 'TranspTindex'

TranspTindexDetermines whether current time period is one of activetranspiration based on temperature index method

Name
`prms2008.TranspTindex`

Author
George H. Leavesley - ghleavesley@colostate.edu

Keyword
Evapotranspiration

Version
$Id: PotetJh.java 861 2010-01-21 01:54:38Z ghleavesley $

Source
$URL: http://svn.javaforge.com/svn/oms/branches/oms3.prj.prms2008/src/prms2008/PotetJh.java $

License
http://www.gnu.org/licenses/gpl-2.0.html

1.17.1. Parameter

nhru - `int`
Number of HRUs.

temp_units - `int`
Units for measured temperature (0=Fahrenheit; 1=Celsius)

transp_beg `[month] - int[]`
Month to begin summing tmaxf for each HRU; when sum is >= to transp_tmax, transpiration begins

transp_end `[month] - int[]`
Month to stop transpiration computations; transpiration is computed thru end of previous month

transp_tmax `[degrees] - double[]`
Temperature index to determine the specific date of the start of the transpiration period. Subroutine sums tmax for each HRU starting with the first day of month transp_beg. When the sum exceeds this index, transpiration begins

1.17.2. Variables (In)

active_hrus - `int`
Number of active HRUs

date `[yyyy mm dd hh mm ss] - Calendar`
Date of the current time step

deltim `[hours] - double`
Length of the time step

hru_route_order - `int[]`
Routing order for HRUs

newday - `int`
Switch signifying the start of a new day (0=no; 1=yes)

route_on - `int`
Kinematic routing switch (0=daily; 1=storm period)

tmaxc `[deg C] - double[]`
Maximum HRU temperature. [temp]

tmaxf `[deg F] - double[]`
Maximum HRU temperature. [temp]

1.17.3. Variables (Out)

basin_transp_on `- int`
Switch indicating whether transpiration is occurring anywhere in the basin (0=no; 1=yes)

transp_on `- int[]`
Switch indicating whether transpiration is occurring (0=no; 1=yes)

1.17.4. Bibliography

- Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

1.17.5. PotetJh

1.17.5.1. Description

The potential evapotranspiration for each HRU (`potet`) for each time period is computed by:

$$\text{potet} = \text{jh\_coef} \times (\text{tavgf} - \text{jh\_coef\_hru}) \times \text{rin}$$

where

jh_coef
the monthly air temperature coefficient used in Jensen-Haise potential evapotranspiration computations,

jh_coef_hru
the air temperature coefficient used in Jensen-Haise potential evapotranspiration computations for each HRU,

rin
the daily solar radiation expressed in inches of evaporation potential, and

tavgf
the average HRU temperature, in °F.

For aerodynamically rough crops, which are assumed to include forests, `jh_coef` can be computed each month for the watershed by

$$\text{jh\_coef} = \left[\text{C1} + (13.0 \times \text{CH})\right]^{-1}$$

`C1` is an elevation correction factor computed by:

$$\text{C1} = 68.0 - \left[3.6 \times \frac{\text{E1}}{1000}\right]$$

where `E1` is the median elevation of the watershed in feet.

`CH` is a humidity index computed by:

$$CH = \frac{50.0}{e2 - e1}$$

where

e2
: the saturation vapor pressure (mb) for the mean maximum air temperature for the warmest month of the year, and

e1
: the saturation vapor pressure (mb) for the mean minimum air temperature for the warmest month of the year

`jh_coef_hru` is computed for each HRU by:

Equation 2.14.

$$jh\_coef\_hru = 27.5 - (0.25 \times (e2 - e1)) - \frac{E2}{1000}$$

where `E2` is the median elevation of the HRU in feet.

The basin weighted average potential evapotranspiration, `basin_potet`, is also computed in this module.

1.17.5.2. References

Jensen, M. E., and Haise, H. R., 1963,Estimating evapotranspiration from solar radiation: Proceedings of the American Society of Civil Engineers, Journal of Irrigation and Drainage, v.89, no. IR4, p. 15-41.

Jensen, M. E., Rob, D. C. N., and Franzoy, C. E., 1969, Scheduling irrigations using climate-crop-soil data: National Conference on Water Resources Engineering of the American Society of Civil Engineers, New Orleans, LA., 1969, Proceedings, 20 p.

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G., 1983, Precipitation-runoff modeling system--user's manual: U. S. Geological Survey Water Resources Investigations report 83-4238, 207 p.

Chapter 3. Parameter Set

adjmix_rain

`{0.5, 0.6000000238419, 0.6999999880791, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.5}`

bound - nmonths

albset_rna

`0.8000000119209`

albset_rnm

`0.6000000238419`

albset_sna

`0.05000000074506`

albset_snm

`0.2000000029802`

basin_area

`0.0`

basin_solsta

`0`

basin_tsta

`2`

carea_max

`{0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.0, 0.0, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.0, 0.0, 0.0, 0.0, 0.2000000029802, 0.2000000029802}`

bound - nhru

cecn_coef

`{5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0}`

bound - nmonths

cov_type

`{3, 1, 1, 3, 3, 3, 3, 3, 0, 0, 3, 3, 3, 3, 3, 3, 3, 0, 0, 0, 0, 1, 3}`

bound - nhru

covden_sum

`{0.1000000014901, 0.2000000029802, 0.2000000029802, 0.25, 0.25, 0.25, 0.3499999940395, 0.4000000059605, 0.0, 0.0, 0.25, 0.3499999940395, 0.4000000059605, 0.4000000059605, 0.3499999940395, 0.3499999940395, 0.3499999940395, 0.0, 0.0, 0.0, 0.0, 0.2000000029802, 0.25}`

bound - nhru

covden_win

`{0.1000000014901, 0.2000000029802, 0.2000000029802, 0.25, 0.25, 0.25, 0.3499999940395, 0.4000000059605, 0.0, 0.0, 0.25, 0.3499999940395, 0.4000000059605, 0.4000000059605, 0.3499999940395, 0.3499999940395, 0.3499999940395, 0.0, 0.0, 0.0, 0.0, 0.2000000029802, 0.25}`

bound - nhru

dday_intcp

`{-10.0, -10.0, -10.0, -10.0, -10.0, -10.0, -10.0, -10.0, -10.0, -10.0, -10.0, -10.0}`

bound - nmonths

dday_slope

`{0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605, 0.4000000059605}`

bound - nmonths

den_init

`0.1000000014901`

den_max

`0.5`

dprst_flag

`0`

emis_noppt

`0.9750000238419`

endTime

`1986-09-30`

epan_coef

`{1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0}`

bound - nmonths

freeh2o_cap

`0.05000000074506`

frozen

`{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}`

bound - nhru

glacier_flag

`0`

groundmelt

`{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}`

bound - nhru

gwflow_coef

`{0.01269999984652}`

bound - ngw

gwsink_coef

`{0.0}`

bound - ngw

gwstor_init

```
{1.0}
```

bound - ngw

hru_area

```
{10650.0, 5540.0, 13340.0, 30070.0, 7450.0, 5280.0, 7940.0, 10530.0, 5490.0, 3830.0, 8610.0,
7310.0, 14160.0, 12370.0, 6820.0, 9840.0, 14410.0, 2970.0, 5860.0, 4170.0, 3130.0, 10770.0,
17710.0}
```

bound - nhru

hru_deplcrv

```
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
```

bound - nhru

hru_elev

```
{5600.0, 5800.0, 7400.0, 7100.0, 6100.0, 6800.0, 7900.0, 7900.0, 8500.0, 8400.0, 6800.0,
8400.0, 8300.0, 8600.0, 8800.0, 9000.0, 8400.0, 8900.0, 9800.0, 9700.0, 8500.0, 7400.0,
6700.0}
```

bound - nhru

hru_gwres

```
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
```

bound - nhru

hru_percent_dprst

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

bound - nhru

hru_percent_imperv

```
{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, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0}
```

bound - nhru

hru_psta

```
{1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 1, 3, 3, 4, 5, 5, 4, 3, 5, 5, 4, 1, 1}
```

bound - nhru

hru_radpl

```
{7, 2, 2, 3, 4, 8, 8, 4, 8, 4, 5, 3, 5, 5, 8, 9, 8, 6, 5, 8, 7, 7, 3}
```

bound - nhru

hru_solsta

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

bound - nhru

hru_ssres

`{2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2}`

bound - nhru

hru_tsta

`{2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2}`

bound - nhru

hru_type

`{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}`

bound - nhru

imperv_stor_max

`{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, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0}`

bound - nhru

inputFile

`/od/projects/oms3.prj.prms2008/data/efcarson/data.csv`

jh_coef

`{0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213, 0.01400000043213}`

bound - nmonths

jh_coef_hru

`{15.0, 14.80000019073, 13.19999980927, 13.5, 14.5, 13.80000019073, 12.69999980927, 12.69999980927, 12.10000038147, 12.19999980927, 13.80000019073, 12.19999980927, 12.30000019073, 12.0, 11.80000019073, 11.60000038147, 12.19999980927, 11.69999980927, 10.80000019073, 10.89999961853, 12.10000038147, 13.19999980927, 13.89999961853}`

bound - nhru

melt_force

`120`

melt_look

`90`

moyrsum

`0`

mxnsos

`1`

role - dimension

ncascade

`0`

role - dimension

ncascdgw

```
0
```

role - dimension

nchan

```
0
```

role - dimension

ndays

```
366
```

role - dimension

ndepl

```
1
```

role - dimension

ndeplval

```
11
```

role - dimension

nevap

```
0
```

role - dimension

nform

```
0
```

role - dimension

ngate

```
1
```

role - dimension

ngw

```
1
```

role - dimension

nhru

```
23
```

role - dimension

nlapse

```
3
```

role - dimension

nmonths

```
12
```

role - dimension

nnode

    0

role - dimension

nobjfunc

    5

role - dimension

nobs

    1

role - dimension

nradpl

    9

role - dimension

nrain

    5

role - dimension

nsegment

    0

role - dimension

nsfres

    1

role - dimension

nsnow

    0

role - dimension

nsol

    0

role - dimension

nssr

    2

role - dimension

nstorm

    0

role - dimension

ntemp

    2

role - dimension

objfunc_q

    0

out

    summary.txt

outFile

    out.csv

outlet_sta

    0

pkwater_equiv_intcp

    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}

bound - nhru

pmo

    0

potet_sublim

    0.75

ppt_rad_adj

    {0.01999999955297, 0.01999999955297, 0.01999999955297, 0.01999999955297, 0.01999999955297,
    0.01999999955297, 0.01999999955297, 0.01999999955297, 0.01999999955297, 0.01999999955297,
    0.01999999955297, 0.01999999955297}

bound - nmonths

precip_units

    0

print_debug

    0

print_freq

    3

print_objfunc

    0

print_type

    1

rad_conv

    1.0

rad_trncf

    {0.7699999809265, 0.589999973774, 0.589999973774, 0.5099999904633, 0.5099999904633,
    0.5099999904633, 0.3899999856949, 0.3400000035763, 1.0, 1.0, 0.5099999904633,
    0.3899999856949, 0.3400000035763, 0.3400000035763, 0.3899999856949, 0.3899999856949,
    0.3899999856949, 1.0, 1.0, 1.0, 1.0, 0.589999973774, 0.5099999904633}

bound - nhru

radadj_intcp

    1.0

radadj_slope

0.0

radj_sppt

0.4399999976158

radj_wppt

0.5

radmax

0.8000000119209

radpl_aspect

{0.0, 90.0, 90.0, 45.0, 45.0, 45.0, 135.0, 135.0, 90.0}

bound - nradpl

radpl_lat

{38.5, 38.5, 38.5, 38.5, 38.5, 38.5, 38.5, 38.5, 38.5}

bound - nradpl

radpl_slope

{0.0, 0.2000000029802, 0.3000000119209, 0.2000000029802, 0.3000000119209, 0.4000000059605, 0.2000000029802, 0.3000000119209, 0.4000000059605}

bound - nradpl

rain_adj

{{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}}

bound - nmonths,nhru

rain_code

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}

bound - nmonths

runoff_units

0

settle_const

0.1000000014901

smidx_coef

{5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 0.0, 0.0, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 5.000000237487E-4, 0.0, 0.0, 0.0, 0.0, 5.000000237487E-4, 5.000000237487E-4}

bound - nhru

smidx_exp

{0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.0, 0.0, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.2000000029802, 0.0, 0.0, 0.0, 0.0, 0.2000000029802, 0.2000000029802}

bound - nhru

snarea_curve

{{0.05000000074506, 0.2399999946356, 0.4000000059605, 0.5299999713898, 0.6499999761581, 0.75, 0.8199999928474, 0.8799999952316, 0.9300000071526, 0.9900000095367, 1.0}}

bound - ndepl, ndeplval

snarea_thresh

{26.10000038147, 26.10000038147, 26.10000038147, 26.10000038147, 26.10000038147, 26.10000038147, 43.40000152588, 43.40000152588, 52.09999847412, 52.09999847412, 26.10000038147, 52.09999847412, 52.09999847412, 52.09999847412, 52.09999847412, 52.09999847412, 52.09999847412, 69.5, 70.0, 70.0, 69.40000152588, 26.10000038147, 26.10000038147}

bound - nhru

snow_adj

{{0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.19000005722, 1.30999994278, 1.149999976158, 1.149999976158, 1.44000005722, 1.379999995232, 1.19000005722, 1.049999952316, 1.0, 1.539999961853, 1.200000047684, 1.299999952316, 1.44000005722, 1.299999952316, 1.679999947548, 1.639999985695, 1.480000019073, 0.660000026226, 1.30999994278}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.19000005722, 1.30999994278, 1.149999976158, 1.149999976158, 1.44000005722, 1.379999995232, 1.19000005722, 1.049999952316, 1.0, 1.539999961853, 1.200000047684, 1.299999952316, 1.44000005722, 1.299999952316, 1.679999947548, 1.639999985695, 1.480000019073, 0.660000026226, 1.30999994278}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.19000005722, 1.30999994278, 1.149999976158, 1.149999976158, 1.44000005722, 1.379999995232, 1.19000005722, 1.049999952316, 1.0, 1.539999961853, 1.200000047684, 1.299999952316, 1.44000005722, 1.299999952316, 1.679999947548, 1.639999985695, 1.480000019073, 0.660000026226, 1.30999994278}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.19000005722, 1.30999994278, 1.149999976158, 1.149999976158, 1.44000005722, 1.379999995232, 1.19000005722, 1.049999952316, 1.0, 1.539999961853, 1.200000047684, 1.299999952316, 1.44000005722, 1.299999952316, 1.679999947548, 1.639999985695, 1.480000019073, 0.660000026226, 1.30999994278}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.080000042915, 1.19000005722, 1.019999980927, 1.019999980927, 1.27999997139, 1.230000019073, 1.080000042915, 0.9399999976158, 0.8899999856949, 1.360000014305, 1.05999994278, 1.149999976158, 1.27999997139, 1.149999976158, 1.490000009537, 1.450000047684, 1.320000052452, 0.6000000238419, 1.19000005722}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.080000042915, 1.19000005722, 1.019999980927, 1.019999980927, 1.27999997139, 1.230000019073, 1.080000042915, 0.9399999976158, 0.8899999856949, 1.360000014305, 1.05999994278, 1.149999976158, 1.27999997139, 1.149999976158, 1.490000009537, 1.450000047684, 1.320000052452, 0.6000000238419, 1.19000005722}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.080000042915, 1.19000005722, 1.019999980927, 1.019999980927, 1.27999997139, 1.230000019073, 1.080000042915, 0.9399999976158, 0.8899999856949, 1.360000014305, 1.05999994278, 1.149999976158, 1.27999997139, 1.149999976158, 1.490000009537, 1.450000047684, 1.320000052452, 0.6000000238419, 1.19000005722}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.080000042915, 1.19000005722, 1.019999980927, 1.019999980927, 1.27999997139, 1.230000019073, 1.080000042915, 0.9399999976158, 0.8899999856949, 1.360000014305, 1.05999994278, 1.149999976158, 1.27999997139, 1.149999976158, 1.490000009537, 1.450000047684, 1.320000052452, 0.6000000238419, 1.19000005722}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.080000042915, 1.19000005722, 1.019999980927, 1.019999980927, 1.27999997139, 1.230000019073, 1.080000042915, 0.9399999976158, 0.8899999856949, 1.360000014305, 1.05999994278, 1.149999976158, 1.27999997139, 1.149999976158, 1.490000009537, 1.450000047684, 1.320000052452, 0.6000000238419, 1.19000005722}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.19000005722, 1.30999994278, 1.149999976158, 1.149999976158, 1.44000005722, 1.379999995232, 1.19000005722, 1.049999952316, 1.0, 1.539999961853, 1.200000047684, 1.299999952316, 1.44000005722, 1.299999952316, 1.679999947548, 1.639999985695, 1.480000019073, 0.660000026226, 1.30999994278}, {0.5, 0.5, 0.6000000238419, 0.6000000238419, 1.19000005722, 1.30999994278, 1.149999976158, 1.149999976158, 1.44000005722, 1.379999995232, 1.19000005722, 1.049999952316, 1.0, 1.539999961853, 1.200000047684, 1.299999952316, 1.44000005722, 1.299999952316, 1.679999947548, 1.639999985695, 1.480000019073, 0.660000026226, 1.30999994278}}

bound - nmonths,nhru

snow_intcp

{0.1000000014901, 0.0, 0.0, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.0, 0.0, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.1000000014901, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1000000014901}

bound - nhru

snowinfil_max

{1.5, 1.5, 1.5, 1.5, 1.5, 1.5, 2.0, 2.0, 2.0, 2.0, 1.5, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 1.5, 1.5}

bound - nhru

soil2gw_max

{0.1819999963045, 0.1819999963045}

bound - nhru

soil_moist_init

{2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 1.0, 1.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 1.0, 1.0, 1.0, 1.0, 2.0, 2.0}

bound - nhru

soil_moist_max

{7.710000038147, 6.159999847412, 6.159999847412, 7.710000038147, 7.710000038147, 7.710000038147, 9.159999847412, 9.159999847412, 3.089999914169, 3.089999914169, 7.710000038147, 9.159999847412, 9.159999847412, 9.159999847412, 9.159999847412, 9.159999847412, 9.159999847412, 3.089999914169, 3.089999914169, 3.089999914169, 3.089999914169, 6.159999847412, 7.710000038147}

bound - nhru

soil_moist_srunoff

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}

bound - nhru

soil_rechr_init

{1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0}

bound - nhru

soil_rechr_max

{2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0, 2.0}

bound - nhru

soil_type

`{1, 1, 1, 2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 2}`

bound - nhru

srain_intcp

`{0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.009999999776483, 0.009999999776483, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.009999999776483, 0.009999999776483, 0.009999999776483, 0.009999999776483, 0.05000000074506, 0.05000000074506}`

bound - nhru

ssr2gw_exp

`{1.0, 1.0}`

bound - nssr

ssr2gw_rate

`{0.01999999955297, 0.01999999955297}`

bound - nssr

ssr_gwres

`{1, 1}`

bound - nssr

ssrcoef_lin

`{0.1010000035167, 0.1000000014901}`

bound - nssr

ssrcoef_sq

`{0.08449999988079, 0.04500000178814}`

bound - nssr

ssrmax_coef

`{1.0, 1.0}`

bound - nssr

ssstor_init

`{0.0, 0.0}`

bound - nssr

startTime

`1980-10-01`

storm_scale_factor

`{}`

bound - nstorm

strain_adj

{{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 },{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 }, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 },{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 }, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 },{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 }, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 },{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 }, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 },{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 }, {0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581 },{0.4000000059605, 0.4000000059605, 0.5, 0.5, 0.8999999761581, 0.8999999761581, 0.8600000143051, 0.8600000143051, 1.0, 1.0, 0.8999999761581, 0.7799999713898, 0.7400000095367, 1.0, 0.8999999761581, 0.8999999761581, 1.0, 0.8999999761581, 1.0, 1.080000042915, 1.0, 0.5, 0.8999999761581}}

bound - nmonths,nhru

sumFile

basinsum.csv

temp_units

0

tmax_adj

{1.0, 0.0, 0.0, 0.0, -1.0, 1.0, 1.0, -1.0, 1.0, -1.0, -1.0, 0.0, -1.0, -1.0, 1.0, 0.0, 1.0, -1.0, -1.0, 1.0, 1.0, 1.0, 0.0}

bound - nhru

tmax_allrain

`{60.0, 60.0, 60.0, 50.0, 50.0, 50.0, 40.0, 40.0, 40.0, 50.0, 60.0, 60.0}`

bound - nmonths

tmax_allsnow

`31.63999938965`

tmax_index

`{50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0}`

bound - nmonths

tmax_lapse

`{3.200000047684, 4.199999809265, 4.900000095367, 5.300000190735, 5.699999809265, 5.800000190735, 6.300000190735, 4.0, 4.900000095367, 4.199999809265, 3.299999952316, 3.0}`

bound - nmonths

tmin_adj

`{1.0, 0.0, 0.0, 0.0, -1.0, 1.0, 1.0, -1.0, 1.0, -1.0, -1.0, 0.0, -1.0, -1.0, 1.0, 0.0, 1.0, -1.0, -1.0, 1.0, 1.0, 1.0, 0.0}`

bound - nhru

tmin_lapse

`{3.200000047684, 4.199999809265, 4.900000095367, 5.300000190735, 5.699999809265, 5.800000190735, 6.300000190735, 4.0, 4.900000095367, 4.199999809265, 3.299999952316, 3.0}`

bound - nmonths

transp_beg

`{1, 3, 3, 1, 1, 1, 4, 4, 4, 4, 1, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 1}`

bound - nhru

transp_end

`{12, 10, 10, 12, 12, 12, 11, 11, 10, 10, 12, 11, 11, 11, 11, 11, 11, 10, 10, 10, 10, 10, 12}`

bound - nhru

transp_tmax

`{500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0, 500.0}`

bound - nhru

tsta_elev

`{7829.0, 5670.0}`

bound - ntemp

tstorm_mo

`{0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0}`

bound - nmonths

wrain_intcp

{0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.009999999776483, 0.009999999776483, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.05000000074506, 0.009999999776483, 0.009999999776483, 0.009999999776483, 0.009999999776483, 0.05000000074506, 0.05000000074506}

bound - nhru

Index

A

B

C

I

O

P

R

S

T

U

W