Package 'VIC5'

July 21, 2025

Type Package

Title The Variable Infiltration Capacity (VIC) Hydrological Model

Version 0.2.6

Description The Variable Infiltration Capacity (VIC) model is a macroscale hydrologic model that solves full water and energy balances, originally developed by Xu Liang at the University of Washington (UW).
The version of VIC source code used is of 5.0.1 on <https://github.com/UW-Hydro/VIC/>, see Hamman et al. (2018).
Development and maintenance of the current official version of the VIC model at present is led by the UW Hydro (Computational Hydrology group) in the Department of Civil and Environmental Engineering at UW. VIC is a research model and in its various forms it has been applied to most of the major river basins around the world, as well as globally <http://vic.readthedocs.io/en/master/Documentation/References/>.
References: ``Liang, X., D. P. Lettenmaier, E. F. Wood, and S. J. Burges (1994), A simple hydrologically based model of land surface water and energy fluxes for general circulation models, J. Geophys. Res., 99(D7), 14415-14428, <doi:10.1029/94JD00483>";
``Hamman, J. J., Nijssen, B., Bohn, T. J., Gergel, D. R., and Mao, Y. (2018), The Variable Infiltration Capacity model version 5 (VIC-5): infrastructure improvements for new applications and reproducibility, Geosci. Model Dev., 11, 3481-3496, <doi:10.5194/gmd-11-3481-2018>".

License GPL-3

Copyright file inst/COPYRIGHTS

URL https://github.com/rpkgs/VIC5

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0)

Imports stats, utils, lubridate, Rcpp, foreach

Suggests testthat (>= 3.0.0), doParallel

LinkingTo RcppArmadillo, Rcpp

RoxygenNote 7.2.3

Config/testthat/edition 3

NeedsCompilation yes

Author Ruida Zhong [aut, ctb],
Dongdong Kong [aut, cre] (ORCID:
<https://orcid.org/0000-0003-1836-8172>),
Xiaohong Chen [aut, ctb],
Zhaoli Wang [aut, ctb],
Chengguang Lai [aut, ctb],
Joseph J Hamman [ctb] (Contributor and maintainer of VIC source code),
Keith Cherkauer [ctb]

Maintainer Dongdong Kong <kongdd.sysu@gmail.com>

Repository CRAN

Date/Publication 2023-02-17 08:40:02 UTC

Contents

NSE *Several metrics to evaluate the accuracy of hydrological modeling*

Description

Calculate several metrics for the evaluating of hydrological modeling accuracy, including the Nash-Sutcliffe coefficient of efficiency (NSE), Logarithmic NSE (log-NSE), relative NSE (rNSE), and relative bias (RB).

Usage

```
NSE(sim, obs)

logNSE(sim, obs)

rNSE(sim, obs)

RB(sim, obs)
```

Arguments

`sim`	Data series to be evaluated, usually are the simulated streamflow of hydrological model.
`obs`	Data series as benchmark to evaluate `sim`, usually are the observed streamflow.

Details

The Nash-Sutcliffe coefficient of efficiency (NSE) (Nash and Sutcliffe, 1970) is a widely used indicator of the accuracy of model simulations, or other estimation method with reference to a benchmark series (usually the observations), especially the hydrological modeling.

NSE is equal to one minus the normalized mean square error (ratio between the mean square error and the variation of observations):

$$NSE = 1 - \sum(sim - obs) * *2 / \sum(obs - mean(obs)) * *2$$

1 is the perfect value of NSE, and NSE < 0 indicates that the simulation results are unusable.

The conventional NSE is ususlly affected by the accuracy of high values, and would impact the low flow simulation when be taken as the objective function in hydrological model calibration. Therefore, some revised NSE were proposed.

Oudin et al. (2006) proposed the log-NSE to increase the sensitivity to the accuracy of low flow simulations:

$$log - NSE = 1 - \sum(log(sim) - log(obs)) * *2 / \sum(log(obs) - log(mean(obs))) * *2$$

Krause et al. (2005) proposed the relative NSE to reduce the impact of the magnitude of data:

$$rNSE = 1 - \sum((sim - obs)obs) * *2 / \sum((obs - mean(obs))/mean(obs)) * *2$$

Relative bias (RB) is used to quantify the relative systematic bias of the simulation results:

$$RB = sum(sim)/sum(obs) - 1$$

Positive or negative value of RB indicate the positive or negative bias of simulations respectively. Perfect value of RB is 0.

Value

The value of NSE, log-NSE, rNSE and RB.

References

Krause, P., Boyle, D. P., and Base, F., 2005, Comparison of different efficiency criteria for hydrological model assessment, Advances in Geoscience, 5, 89-97. doi: 10.5194/adgeo-5-89-2005

Nash, J. E., Sutcliffe, J. V., 1970. River flow forecasting through conceptual models part I - A discussion of principles. Journal of Hydrology. 10(3): 282-290. doi:10.1016/0022-1694(70)90255-6

Oudin, L., Andreassian, V., Mathevet, T., Perrin, C., Michel, C., 2006. Dynamic averaging of rainfall-runoff model simulations from complementary model parameterizations. Water Resources Research, 42(7). doi: 10.1029/2005WR004636

See Also

[Lohmann_UH()], [Lohmann_conv()]

`print.vic_output` *VIC model run for each gridcells*

Description

Run the VIC model for each gridcells by providing several meteorological and vegetation (optional) forcing data and land surface parameters (soil, vegetation, snowband (optional), lake (optional)).

Usage

```
## S3 method for class 'vic_output'
print(x, ...)

vic(
  forcing,
  soil,
  veg,
  output_info = NULL,
  veglib = NULL,
  snowband = NULL,
  lake = NULL,
  forcing_veg = NULL,
  veg_force_types = c("albedo", "LAI", "fcanopy"),
  parall = FALSE
)
```

Arguments

`x`	Return of `vic()` for print.
`...`	Other arguments to print.
`forcing`	meteorological forcing data. Must be a list of numeral matrix with the name of one of "PREC", "TEMP", "SW", "LW", "WIND", "VP" and "PRESS". See details.

soil	soil parameter data. Must be a data frame or numeral matrix.
veg	vegetation parameters. Must be a list containing several matrixs. See details.
output_info	A list containing output contents and timescales (optional). See details.
veglib	Vegetation library parameters (optional). Would using the library of the NLDAS and GLDAS when not provided.
snowband	A data frame or numeral matrix containing snow band parameters (optional). See details.
lake	A dataframe or numeric matrix containing lake parameters (optional).
forcing_veg	Vegetation forcing data (optional). See details.
veg_force_types	Type names of vegetation forcing data. Must be provided when using vegetation forcing data.
parall	Determined if run the VIC parallely. If it is TRUE, `registerDoParallel()` in package **doParallel** is need to be used before run the VIC.

Details

Function `vic` is used to run the VIC model in a "classic" style (Run the model gridcell by gridcell). Meteorological forcing data, soil parameters and vegetation parameters are the basic necessary inputs for the VIC model.

Value

A list containing output tables, time table and model settings.

VIC in R supports multiple "output tables" with different output timescales. For example, you can put soil moisture and soil temperature in a table with monthly timescale and put runoff, baseflow in another table with daily timescale.

You can use `print()` to print the summary of the returns. The returns includes:

output	One or several "output tables" and each of them containing several output variables, which is determined b
timetable	Containing initiation time, model running time and final (orgnizate outputs) time of the VIC model runnin
global_options	Global options setted for this model run.
output_infos	Output settings determined by the `output_info` parameter.

Forcing data

Parameter `forcing` must be a list containing several numeral matrixs that containing forcing data. Name of each matrix (similar to key in dictionary in Python) must be the specific type names including "PREC", "TEMP", "SW", "LW", "WIND", "VP" and "PRESS", indicating precipitation `[mm]`, air temperature `[degree C]`, shortwave radiation `[W]`, longwave radiation `[W]`, wind speed `[m/s]`, vapor pressure `[kPa]`, and atmospheric pressure `[kPa]`. All of those types are necessary to run the VIC model except "LW" and "PRESS". Each row of the matrixs is corresponding to a time step while each column of the matrixs is corresponding to a gridcell, which other must be the same as those in soil parameter.

Longwave radiation (LW) and atmospheric pressure (PRESS) could be estimated via other forcing data when not supplied. Longwave radiation would be estimated using the Konzelmann formula (Konzelmann et al., 1996) while atmospheric pressure would be estimated based on the method of VIC 4.0.6, by assuming the sea level pressure is a constant of 101.3 kPa.

Soil parameters

Parameter `soil` must be a numeric data frame or matrix that containing soil parameters. The style of this is the same as the soil parameter file of the classic VIC, that is, each row restores the parameter of a cell while each column restores one type of parameter. Detail see http://vic.readthedocs.io/en/master/Documentation/Drivers/Classic/SoilParam/ in the official VIC documentation webside.

Vegetation parameter

Parameter `veg` must be a list containing several numeral matrixs that Those matrixs restore the vegetation parameters that are corresponding to each gridcells, and those order must be the same as the soil parameters. Each row of the matrix restores the parameters of a vegetation type while each column restores a type of parameter. Each row should be like:

```
c(veg_type, area_fract, rootzone_1_depth,
  rootzone_1_fract, rootzone_2_depth, rootzone_2_fract, ...)
```

which is similar to the veg param file of the classic VIC. If the source of LAI, fcanopy or albedo is set to veg params, it must be follow by a sequence of param value for each month in a year. The rows of `veg` would be similar as:

```
c(veg_type, area_fract, rootzone_1_depth,
  rootzone_1_fract, rootzone_2_depth, rootzone_2_fract,
  LAI_Jan, LAI_Feb, LAI_Mar, ..., LAI_Dec,
  fcan_Jan, fcan_Feb, fcan_Mar, ..., fcan_Dec,
  albedo_Jan, albedo_Feb, albedo_Mar, ..., albedo_Dec)
```

Output information (Optional)

Parameter `output_info` is used to determine the output variables, output timescale (monthly, daily, sub-daily, or each 6 days, etc.), aggregration of data (mean, sum, max, min, start or end) when output timescale is larger than input timescale. It should be a list like that:

```
output_info <- list(timescale = 'timescale', aggpar = aggpar,
                    outvars = c('OUT_TYPE_1', 'OUT_TYPE_2', 'OUT_TYPE_3', ...),
                    aggtypes = c('aggtype_1', 'aggtype_2', 'aggtype_3'))
```

And a output table (a list containing the output variables in matrix form) named "output" would be returned. You can obtain the variables use code like `res$output$OUT_...`. Names of the items in the list (e.g. timescale, outvars) must be those specified as follows:

`timescale`	Output timescale, including 'never', 'step', 'second', 'minute', 'hour', 'day', 'month', 'year', 'date', and 'end'. 'n
`aggpar`	If 'timescale' is set to those except 'never', 'date' and 'end', it determined the intervals of the timescale to pass b
`outvars`	A sequence of names of output data types. The available data types please see the VIC official documentation we
`aggtypes`	Optional. A sequence of string determine how to aggregrate the output data when output timescale is larger than

If multiple output timescales are used, the outputs could be divided into several lists and take them into a list as input, e.g.:

```
out_info <- list(
  wb = list(timescale = 'hour', aggpar = 6,
            outvars = c('OUT_RUNOFF', 'OUT_BASEFLOW', 'OUT_SOIL_MOIST'),
            aggtypes = c('sum', 'sum', 'end')),
  eb = list(timescale = 'day', aggpar = 1,
            outvars = c('OUT_SWE', 'OUT_SOIL_TEMP'),
            aggtypes = c('avg', 'min'))
)
```

This would return two output tables named "wb" and "eb" respectively.

Vegetation library (Optional)

Parameter `veglib` is a matrix or a numeric dataframe of a vegetation parameter library. Each row determines a type of vegetation while each column determines a parameter, including ovetstory (or not), LAI for each month in a year, etc. If not provided, it would use the default vegetation library of the NASA Landsurface Data Assimination System (LDAS) (Rodell et al., 2004), which contains 11 types of vegetation with the vegetation classification of UMD.

Snowband (elevation band) (Optional)

Parameter `snowband` is a matrix or a numeric dataframe determines the elevation band information for each gridcells. Each row determines the band information of a gridcell while a column determines the values of the elevation band parameters. This devide a single gridcell into several parts with dfferent elevation to run individually, to further consider the sub-gridcell heterogeneity of elevation and the resulted heterogeneity air temperature in a gridcell with higher variation of elevation. The information of elevation bands includes area fraction, mean elevation and fraction of precipitation falled to the gridcell of each elevation band of the gridcell. The order of the rows must be coresponding to the gridcells determined in the soil parameters. Each row should be like:

```
c(GRID_ID, AFRAC_1, AFRAC_2, ..., AFRAC_n, ELEV_1, ELEV_2, ..., ELEV_n,
  PFRAC_1, PFRAC_2, ..., PFRAC_n)
```

`GRID_ID` is the id of the grid; AFRAC_i means area fraction of each elevation band; ELEV_i is their mean elevation; PFRAC_i is there precipitation fraction. n is the number of elevation bands for each gridcell and is determined by `'nbands'` in the global options. This can be set used `veg_param('nbands', n)`.

Vegetation forcing data (Optional)

Parameter `forcing_veg` must be a list containing several 3D numeral arrays that containing vegetation forcing data such as LAI, albedo and fraction of canopy. Different to parameter `forcing`, each 3D array in the list is the vegetation forcing data for a single gridcell, and the order of the 3D arrays must be corresponding to the order of the gridcells determined in the soil parameter.

The dimensions of a 3D array are represents: `[timestep, vegetation tile, forcing type]`

That is, the first dimension determines the data of the timesteps, the second dimension determines the data for the different vegetation tiles in this gridcell that determined by the vegetation parameters, while the third dimension determined the data of different forcing data type (LAI, albedo or fcanopy), which should be corresponding to the parameter `veg_force_types`.

References

Hamman, J. J., Nijssen, B., Bohn, T. J., Gergel, D. R., and Mao, Y. (2018), The Variable Infiltration Capacity model version 5 (VIC-5): infrastructure improvements for new applications and reproducibility, Geosci. Model Dev., 11, 3481-3496, doi:10.5194/gmd1134812018.

Konzelmann, T, Van de Wal, R.S.W., Greuell, W., Bintanja, R., Henneken, E.A.C., Abe-Ouchi, A., 1996. Parameterization of global and longwave incoming radiation for the Greenland Ice Sheet. Global Planet. Change, 9:143-164.

Liang, X., Lettenmaier, D. P., Wood, E. F., and Burges, S. J. (1994), A simple hydrologically based model of land surface water and energy fluxes for general circulation models, J. Geophys. Res., 99(D7), 14415-14428, doi:10.1029/94JD00483.

Liang, X., and Xie, Z., 2001: A new surface runoff parameterization with subgrid-scale soil heterogeneity for land surface models, Advances in Water Resources, 24(9-10), 1173-1193.

Rodell, M., Houser, P.R., Jambor, U., Gottschalck, J., Mitchell, K., Meng, C.-J., Arsenault, K., Cosgrove, B., Radakovich, J., Bosilovich, M., Entin, J.K., Walker, J.P., Lohmann, D., and Toll, D. (2004), The Global Land Data Assimilation System, Bull. Amer. Meteor. Soc., 85(3), 381-394.

Examples

```
# This is a sample data to run VIC.
data(STEHE)

forcing <- STEHE$forcing
soil <- STEHE$soil
veg <- STEHE$veg

# Set the global options for a 7-days run.
vic_params(list(
  date_start = "1949-01-01",
  date_end = "1949-01-10",
  step_per_day = 24,
  snow_step_per_day = 24,
  runoff_step_per_day = 24
))

# Run VIC.
outputs <- vic(forcing, soil, veg)
print(outputs)

# Use user defind outputs and snowband parameters.
vic_param('nbands', 5)
band <- STEHE$snowbands

out_info <- list(
  wb = list(timescale = 'hour', aggpar = 6,
            outvars = c('OUT_RUNOFF', 'OUT_BASEFLOW', 'OUT_SOIL_MOIST'),
            aggtypes = c('sum', 'sum', 'end')),
  eb = list(timescale = 'day', aggpar = 1,
            outvars = c('OUT_SWE', 'OUT_SOIL_TEMP'),
            aggtypes = c('avg', 'min'))
)
```

```
outputs <- vic(forcing, soil, veg, snowband = band, output_info = out_info)
print(outputs)

# Example of parallelization
## Not run:
library(doParallel)
registerDoParallel(cores=4)
outputs <- vic(forcing, soil, veg, snowband = band, parall = TRUE)
stopImplicitCluster()
print(outputs)

## End(Not run)
```

STEHE *Sample datasets of the Stehekin for the running of the VIC model provided by UW Hydro*

Description

Sample datasets of the Stehekin for the running of the VIC model provided by UW Hydro

Usage

```
STEHE
```

Format

A list including meteorological forcing data, soil parameters, vegetation parameters, vegetation library, snowband parameters and vegetation forcing data for 16 0.25 degree gridcells. The dataset is provided by the Computational Hydrology group, Department of Civil and Environmental Engineering at the University of Washington.

This is a list containing:

forcing 10-day hourly meteorological forcing data from 1949-01-01 to 1949-01-10, including precipitation, air temperature, longwave and shortwave radiation, wind speed, vapor pressure, and air pressure.

soil soil parameters for 16 gridcells.

veg vegetation parameters for 16 gridcells.

snowbands elevation band (snow band) parameters.

veglib vegetation library.

forcing_veg vegetation forcing data including LAI, fraction of canopy, and albedo. This is generated by simply temporally repeat the values of `veglib` according to `veg`.

runoff_table_daily Three-year daily runoff outputs of the VIC model from 1949-01-01 to 1951-12-31. This is as the sample input data of the streamflow routing model of Lohmann in the form of output table returned by `vic()`, containing two output variable: `OUT_RUNOFF` and `OUT_BASEFLOW`.

streamflow_obs Observed streamflow series to be the reference to evaluate the streamflow simulation results.

Source

https://github.com/UW-Hydro/VIC_sample_data

http://uw-hydro.github.io/

`veglib_IGBP` *IGBP vegetation library for VIC model*

Description

This library is originated from Python package `hydrate`.

Usage

```
veglib_IGBP
```

Format

An object of class `data.table` (inherits from `data.frame`) with 17 rows and 58 columns.

References

1. https://github.com/KMarkert/hydrate/blob/master/hydrate/lookups/veg_params.py
2. https://github.com/KMarkert/hydrate

`vic_params` *Get or set global parameters of the VIC model.*

Description

Get or set global parameters of the VIC model.

Usage

```
vic_params(options)

vic_param(par, val = NULL)
```

Arguments

`options`	List object, see details
`par`	Name of the VIC global parameter.
`val`	Value of the parameter to be set. Will return the current setting value when not provided.

Details

The global paremteres are as follows:

nlayers Integer. Number of moisture layers used by the model. Default = 3.

nnodes Integer. Number of thermal solution nodes in the soil column.

step_per_day Integer. Number of simulation time steps per day. Should be > 4 when full_energy=TRUE or frozen_soil=TRUE.

snow_step_per_day Integer. Number of time steps per day used to solve the snow model (if step_per_day > 1, this should = step_per_day)

runoff_step_per_day Integer. Number of time steps per day used to solve the runoff model (should be >= step_per_day)

start_year Integer. Year of model simulation starts.

start_month Integer. Month of model simulation starts

start_day Integer. Day of model simulation starts

start_sec Integer. Second of model simulation starts (e.g., start_sec = 0 means starting from the beginning of a day; start_sec = 3600 for starting from the beginning of the second hour of a day). Default = 0.

nrecs Integer. Number of time steps over which to run model. The number of records must be defined such that the model completes the last day. Either??nrecs or end_year, end_month, and end_day must be specified, but??not both.

end_year Integer. Year of model simulation ends.

end_month Integer. Month of model simulation ends

end_day Integer. Day of model simulation ends

calendar Boolean. Calendar to use. Calendar to use. Must be one of: "standard", "gregorian", "proleptic_gregorian", "noleap", "365_day", "360_day", "julian", "all_leap", "366_day."

time_units Boolean. Units for output time variables. Units for output time variables. Should be one of "days", "hours", "minutes", "seconds". Default = days.

full_energy Boolean. Option for computing land surface temperature (soil or snowpack surface). TRUE??= compute (via iteration) the temperature that balances the surface energy budget. FALSE??= set surface temperature equal to air temperature. Default = False.

close_energy Boolean. Option for controlling links between the energy balances of the surface and the canopy. TRUE??= iterate between the canopy and surface energy balances until they are consistent. FALSE??= compute the surface and canopy energy balances separately, once per time step.?? Default = FALSE.

frozen_soil Boolean. Option for handling the water/ice phase change in frozen soils. TRUE = account for water/ice phase change (including latent heat). FALSE = soil moisture always remains liquid, even when below 0 C; no latent heat effects and ice content is always 0. Default = FALSE. To activate this option, the user must also set the FS_ACTIVE flag to 1 in the soil parameters (parameter `soil` of `vic()`) for each grid cell where this option is desired. In other words, the user can choose for some grid cells (e.g. cold areas) to compute ice contents and for others (e.g. warm areas) to skip the extra computation.

quick_flux Boolean. Option for computing the soil vertical temperature profile. TRUE = use the approximate method described by Liang et al. (1999) to compute soil temperatures and ground heat flux; this method ignores water/ice phase changes. FALSE = use the finite element method described in Cherkauer and Lettenmaier (1999) to compute soil temperatures and ground heat flux; this method is appropriate for accounting for water/ice phase changes. Default = FALSE (i.e. use Cherkauer and Lettenmaier, 1999) when running FROZEN_SOIL; and TRUE (i.e. use Liang et al. ,1999) in all other cases.

implicit Boolean. If TRUE the model will use an implicit solution for the soil heat flux equation of Cherkauer and Lettenmaier (1999) (quick_flux is FALSE), otherwise uses original explicit solution. When quick_flux is TRUE the implicit solution has no effect. The user can override this option by setting implicit to FALSE in the global parameters. The implicit solution is guaranteed to be stable for all combinations of time step and thermal node spacing; the explicit solution is only stable for some combinations. If the user sets implicit to FALSE, VIC will check the time step, node spacing, and soil thermal properties to confirm stability. If the explicit solution will not be stable, VIC will exit with an error message. Default = TRUE.

quick_solve Boolean. This option is a hybrid of quick_flux. If TRUE, model will use the method described by Liang et al. (1999) to compute ground heat flux during the surface energy balance iterations, and then will use the method described in Cherkauer and Lettenmaier (1999) for the final solution step. Default = FALSE.

no_flux Boolean. If TRUE model will use a no flux bottom boundary with the finite difference soil thermal solution (i.e. quick_flux = FALSE or full_energy = TRUE or frozen_soil = TRUE). Default = FALSE (i.e., use a constant temperature bottom boundary condition).

exp_trans Boolean. If TRUE the model will exponentially distributes the thermal nodes in the Cherkauer and Lettenmaier (1999) finite difference algorithm, otherwise uses linear distribution. (This is only used if frozen_soil = TRUE). Default = TRUE.

grnd_flux_type Integer. Options for handling ground flux: 0 = use (flawed) formulas for ground flux, delta H, and fusion as in VIC 4.0.6 and earlier. 1 = use formulas from VIC 4.1.0. This option exists for backwards compatibility with earlier releases and likely will be removed in later releases. Default = 1.

Tfallback Boolean. Options for handling failures of T iterations to converge. FALSE = if T iteration fails to converge, report an error. TRUE = if T iteration fails to converge, use the previous time step's T value. This option affects the temperatures of canopy air, canopy snow, ground snow pack, ground surface, and soil T nodes. If TFALLBACK is TRUE, VIC will report the total number of instances in which the previous step's T was used, at the end of each grid cell's simulation. In addition, a time series of when these instances occurred (averaged across all veg tile/snow band combinations) can be written to the outputs, using the following output variables: OUT_TFOL_FBFLAG = time series of T fallbacks in canopy snow T solution. OUT_TCAN_FBFLAG = time series of T fallbacks in canopy air T solution. OUT_SNOWT_FBFLAG = time series of T fallbacks in snow pack surface T solution. OUT_SURFT_FBFLAG = time series of T fallbacks in ground surface T solution. OUT_SOILT_FBFLAG = time series of T fallbacks in soil node T solution (one time series per node). Default = TRUE.

share_layer_moist Boolean. If TRUE, then??if??the soil moisture in the layer that contains more than half of the roots is above the critical point, then the plant's roots in the drier layers can access the moisture of the wetter layer so that the plant does not experience moisture limitation. If FALSE or all of the soil layer moistures are below the critical point, transpiration in each layer is limited by the layer's soil moisture. Default: TRUE.

spatial_frost Integer. Option to allow spatial heterogeneity in soil temperature. 0 = Assume soil temperature is horizontally constant (only varies with depth). > 0 = Assume soil temperatures at each given depth are distributed horizontally with a uniform (linear) distribution and the value would be the number of frost sub-areas (each having a distinct temperature). In this case, even when the mean temperature is below freezing, some portion of the soil within the grid cell at that depth could potentially be above freezing. This requires specifying a frost slope value as an extra field in the soil parametes, so that the minimum/maximum temperatures can be computed from the mean value. The maximum and minimum temperatures will be set to mean temperature +/- frost_slope. Default = 0.

snow_den Boolean. Options for computing snow density. 0 = Use algorithm taken from SNTHRM model.1 = Use traditional VIC algorithm taken from Bras (1990). Default = 0.

blowing Boolean. If TRUE, compute evaporative fluxes due to blowing snow. Default = FALSE.

blowing_var_threshold Boolean. If TRUE, a variable shear stress threshold is used to determine the blowing snow flux. If FALSE, then a fixed threshold is used. See Li and Pomeroy (1997) for details. Default: TRUE.

blowing_calc_prob Boolean. If TRUE, the probability of occurrence of blowing snow is calculated as a function of environmental conditions. If FALSE, then the probability is set to 1. See Lu and Pomeroy (1997) for details. Default: TRUE.

blowing_simple Boolean. If TRUE, the sublimation flux of blowing snow is calculated as a function vapor pressure and wind speed. If FALSE, then additional calculations are made to account for a saltation and suspension layer. See Lu and Pomeroy (1997) for details. Default: FALSE.

blowing_fetch Boolean. This option is only used when BLOWING_SIMPLE is set to FALSE. When this option is set to TRUE, the fetch is accounted for in the calculation of the sublimation flux from blowing snow. If FALSE then the fetch is not used. See Lu and Pomeroy (1997) for details. Default: TRUE.

blowing_spatial_wind Boolean. If TRUE, multiple wind speed ranges, calculated according to a probability distribution, are used to determine the sublimation flux from blowing snow. If FALSE, then a single wind speed is used. See Lu and Pomeroy (1997) for details. Default: TRUE.

compute_treeline Integer. Options for handling above-treeline vegetation. 0 = Do not compute treeline or replace vegetation above the treeline. > 0 means compute the treeline elevation based on average July temperatures; for those elevation bands with elevations above the treeline (or the entire grid cell if nbands = 1 when the grid cell elevation is above the tree line), if they contain vegetation tiles having overstory, replace that vegetation with the vegetation having id that same as the value of this parameter in the vegetation library. You must supply VIC with a July average air temperature in the optional July_Tavg field of the soil parameters (parameter `soil` of `vic()`), and set the july_tavg option to TRUE so that VIC can read the soil parameters correctly. If lakes=TRUE, compute_treeline must be FALSE. Default = FALSE.

corrprec Boolean. If TRUE correct precipitation for gauge undercatch. This option is not supported when using snow/elevation bands. Default = FALSE.

spatial_snow Boolean. Option to allow spatial heterogeneity in snow water equivalent (yielding partial snow coverage) when the snow pack is melting. FALSE = Assume snow water equivalent is constant across grid cell. TRUE = Assume snow water equivalent is distributed horizontally with a uniform (linear) distribution, so that some portion of the grid cell has 0 snow pack.

This requires specifying the max_snow_distrib_slope value as an extra field in the soil parameters (parameter `soil` of `vic()`). max_snow_distrib_slope should be set to twice the desired minimum spatial average snow pack depth [m]. I.e., if we define depth_thresh to be the minimum spatial average snow depth below which coverage < 1.0, then max_snow_distrib_slope = 2*depth_thresh. Partial snow coverage is only computed when the snow pack has started melting and the spatial average snow pack depth <= max_snow_distrib_slope/2. During the accumulation season, coverage is 1.0. Even after the pack has started melting and depth <= max_snow_distrib_slope/2, new snowfall resets coverage to 1.0, and the previous partial coverage is stored. Coverage remains at 1.0 until the new snow has melted away, at which point the previous partial coverage is recovered. Default = FALSE.

AERO_resist_cansnow Integer. Options for aerodynamic resistance in snow-filled canopy. 0??= Multiply by 10 for latent heat, but do NOT multiply by 10 for sensible heat. When no snow in canopy, use surface aero_resist instead of overstory aero_resist. (As in VIC 4.0.6). 1 = Multiply by 10 for both latent and sensible heat. When no snow in canopy, use surface aero_resist instead of overstory aero_resist. 2 = Multiply by 10 for both latent and sensible heat. Always use overstory aero_resist (snow or bare). 3??= Apply stability correction, instead of multiplying by 10, for both latent and sensible heat. Always use overstory aero_resist (snow or bare).?? NOTE: this option exists for backwards compatibility with earlier releases and likely will be removed in later releases.??Default = 2.

carbon Boolean. Options for simulating carbon cycle or not. FALSE??= do not simulate carbon cycle. TRUE??= simulate carbon cycle.??Default = FALSE.

RC_mode Integer. Determines how canopy resistance is computed. 0??= VIC computes canopy resistance by applying resistance factors to the veg class's minimum canopy resistance listed in the veg library. 1??= VIC computes canopy resistance by applying resistance factors to the canopy resistance corresponding to photosynthetic demand (in the absence of moisture limitation).?? Default = 0.

veglib_photo Boolean. Tells VIC about the contents of the veg library. Options for VEGLIB_PHOTO: FALSE??= veg library does not contain photosynthesis parameters. TRUE = veg library contains photosynthesis parameters. Default = FALSE

continue_error Boolean. Options for handling fatal errors. FALSE??= if simulation of a grid cell encounters an error, exit VIC. TRUE??= if simulation of a grid cell encounters an error, move to next grid cell.??Default = TRUE.

wind_h Numeric. Height [m] of wind speed measurement over bare soil and snow cover. Wind measurement height over vegetation is now read from the vegetation library for all types, the value in the global options only controls the wind height over bare soil and over the snow pack when a vegetation canopy is not defined.

canopy_layers Integer. Number of canopy layers in the model. Default: 3.

baseflow Integer. This option describes the form of the baseflow parameters in the soil parameters (parameter `soil` of `vic()`): 0 = fields 5-8 of the soil parameters are the standard VIC baseflow parameters; 1 = fields 5-8 of the soil parameters are the baseflow parameters from Nijssen et al (2001) Default = 0.

july_tavg Boolean. If TRUE then VIC will expect an additional column (July_Tavg) in the soil parameters (parameter `soil` of `vic()`) to contain the gridcell's average July temperature. If your soil parameters contains this optional column, you MUST set this parameter to TRUE so that VIC can read the soil parameters correctly. NOTE: Supplying July average temperature is only required if the compute_treeline option is set to TRUE. Default = FALSE.

organic Boolean. TRUE = the soil parameters (parameter `soil` of `vic()`) contains 3*Nlayer extra columns. For each layer: the organic fraction, and the bulk density and soil particle density of the organic matter in the soil layer. FALSE = the soil parameters do not contain any information about organic soil, and organic fraction should be assumed to be 0. Default = FALSE.

nrootzones Integer. Number of defined root zones defined for root distribution.

vegpar_albedo Boolean. If TRUE the vegetation parameters (parameter `veg` of `vic()`) contains the columns for each vegetation type that defines monthly albedo values for each vegetation type for each grid cell. Default = FALSE.

albedo_src Integer. This option tells VIC where to look for ALBEDO values: 1 = Use the ALBEDO values listed in the vegetation library. 2 = Use the ALBEDO values listed in the vegetation parameters (parameter `veg` of `vic()`). Note: for this to work, vegpar_albedo must be TRUE. 3= Use the albedo values in the parameter `forcing_veg` of `vic()`. For this to work, albedo must be supplied in codeforcing_veg. Default = 1.

vegpar_LAI Boolean. If TRUE the vegetation parameters (parameter `veg` of `vic()`) contains an extra line for each vegetation type that defines monthly LAI values for each vegetation type for each gridcell. Default = FALSE.

LAI_src Boolean. This option tells VIC where to look for LAI values: 1 = Use the LAI values listed in the vegetation library. 2 = Use the LAI values listed in the vegetation parameters (parameter `veg` of `vic()`). Note: for this to work, vegpar_LAI must be TRUE. 3= Use the LAI values in the parameter `forcing_veg` of `vic()`. For this to work, albedo must be supplied in codeforcing_veg. Default = 1.

veglib_fcan Boolean. If TRUE the vegetation library contains monthly FCANOPY values for each vegetation type for each grid cell (between the LAI and ALBEDO values). Default = FALSE.

vegpar_fcan Boolean. If TRUE the vegetation parameters (parameter `veg` of `vic()`) contains an extra line for each vegetation type that defines monthly FCANOPY values for each vegetation type for each grid cell. Default = FALSE.

fcan_src Boolean. This option tells VIC where to look for FCANOPY values: 0 = Set FCANOPY to 1.0 for all veg classes, all times, and all locations. 1 = Use the FCANOPY values listed in the vegetation library. Note: for this to work, veglib_fcan must be TRUE. 2 = Use the FCANOPY values listed in the vegetation parameters (parameter `veg` of `vic()`). Note: for this to work, vegpar_fcan must be TRUE. 3= Use the FCANOPY values in the parameter `forcing_veg` of `vic()`. For this to work, FCANOPY must be supplied in codeforcing_veg Default = 0.

nbands Integer. Maximum number of snow elevation bands to use. Parameter snowband should be provided for the `vic()` function when > 1. Default = 1.

lakes Boolean. Options for if simulate lakes lake parameter or not. Default = FALSE.

lake_profile Boolean. Options for describing lake profile: FALSE??= VIC computes a parabolic depth-area profile for the lake basin. TRUE??= VIC reads user-specified depth-area profile from the lake parameters. Default = FALSE.

resolution Numeric. Width of grid cells, in decimal degree latitude or longitude. Default = none, but must be set by the user to match the grid cell size if the lake model is running.

Value

No return value. Set global options for VIC model.

References

Bras, R. F., 1990: Hydrology, an introduction to hydrologic science, Addison-Wesley.

Cherkauer, K. A. and D. P. Lettenmaier, 1999: Hydrologic effects of frozen soils in the upper Mississippi River basin, J. Geophys. Res., 104(D16), 19,599-19,610.

Liang, X., E. F. Wood, and D. P. Lettenmaier, 1999: Modeling ground heat flux in land surface parameterization schemes, J. Geophys. Res., 104(D8), 9581-9600.

Nijssen, B.N., R. Schnur and D.P. Lettenmaier, 2001: Global retrospective estimation of soil moisture using the Variable Infiltration Capacity land surface model, 1980-1993, J. Clim., 14(8), 1790-1808, doi:10.1175/1520-0442(2001)014<1790:GREOSM>2.0.CO;2.

`vic_version` *Display version of VIC model and this package.*

Description

Display version of VIC model and this package.

Usage

```
vic_version()
```

`XAJ` *Run Xinanjiang (XAJ) model (three sources, lumped).*

Description

An R implementation of three-source Xinanjiang model by Renjun Zhao, used for daily streamflow simulation.

Usage

```
XAJ(PREC, EVAP, params, area = dt * 3.6, dt = 24, full.UH = FALSE)
```

Arguments

`PREC`	Time series of precipitation (daily)
`EVAP`	Time series of pan evaporation or potential evapotranspiration (daily), length must equal to `PREC`
`params`	parameters `XAJ.param.range()`
`area`	Basin area (km^2).
`dt`	time step (in hour) of the simulation
`full.UH`	Use the unit hydrograph defined by user, rather than the instantaneous unit hydrograph (IUH) of Nash, for routing of surface runoff. Default FALSE.

Value

This function returns a data frame of some common variables of the XAJ model at each time step, such as evaporation, soil moisture, surface and underground runoff.

The variables of the table including::

- `E` : Total evaporation (mm)
- `EU` : Evaporation (mm) of upper soil layer
- `EL` : Evaporation (mm) of lower soil layer
- `ED` : Evaporation (mm) of deep soil layer
- `W` : Total soil moisture (mm)
- `WU` : Soil moisture (mm) of upper soil layer
- `WL` : Soil moisture (mm) of lower soil layer
- `WD` : Soil moisture (mm) of deep soil layer
- `R` : Total runoff (mm) of each time step
- `RS` : Surface runoff (mm) of each time step
- `RI` : Interflow (mm) of each time step
- `RG` : Underground runoff (mm) of each time step
- `Q` : Total runoff (m^3/s) at the outlet of the basin
- `QS` : Surface runoff (m^3/s) at the outlet of the basin
- `QI` : Interflow runoff (m^3/s) at the outlet of the basin
- `QG` : Underground runoff (m^3/s) at the outlet of the basin

References

Zhao and Liu, 1995. The Xinanjiang model, Computer Models of Watershed Hydrology, Water Resources Publication, Highlands Ranch, CO (1995), pp. 215-232

`XAJ.param.range` *XAJ parameters*

Description

The lumped XAJ model has 13 parameters, including:

- If `full.UH = FALSE`:

The parameter `params` must be a numeric vector looks like: `c(KC, IM, WUM, WLM, WDM, C, B, SM, EX, KI, KG, CI, CG, N, NK)`

- If `full.UH = TRUE`:

when use the instantaneous unit hydrograph of Nash, or looks like: `c(KC, IM, WUM, WLM, WDM, C, B, SM, EX, KI, KG, CI, CG, UH_1, UH_2, ..., UH_n)` UH_1, UH_2, ..., UH_n means the series of the unit hydrograph.

Usage

```
XAJ.param.range
```

Format

An object of class `data.frame` with 15 rows and 4 columns.

Details

Parameters::

1. `KC` : Ratio of potential evap to pan evap
2. `IM` : Fraction of impermeable area
3. `WUM` : Soil moisture capacity of upper layer
4. `WLM` : Soil moisture capacity of lower layer
5. `WDM` : Soil moisture capacity of deep layer
6. `C` : Coefficient of deep evap
7. `B` : Exponent of the soil moisture storage capacity curve
8. `SM` : Areal mean free water capacity of the surface soil layer
9. `EX` : Exponent of the free water capacity curve
10. `KI` : outflow coefficients of the free water storage to interflow
11. `KG` : outflow coefficients of the free water storage to groundwater
12. `CI` : recession constant of the lower interflow storage
13. `CG` : recession constant of groundwater storage.
14. `N`: (optional) number of reservoirs in the instantaneous unit hydrograph
15. `NK`: (optional) common storage coefficient in the instantaneous unit hydrograph

If `full.UH = TRUE`: 14~end: is the full unit hydrograph defined by user.

Index