Package ‘HBV.IANIGLA’

August 29, 2019

Type Package

Title Decoupled Hydrological Model for Research and Education Purposes

Version 0.1.1

Date 2019-08-29

Author Ezequiel Toum <etoum@mendoza-conicet.gob.ar>

Maintainer Ezequiel Toum <etoum@mendoza-conicet.gob.ar>

Description The HBV (Hydrologiska Byråns Vattenbalansavdelning) hydrological model is decoupled to allow the user
to build his/her own model. This version was developed by the author in IANIGLA-CONICET (Instituto Argentino de Nivologia, Glaciologia
y Ciencias Ambientales - Consejo Nacional de Investigaciones Cientificas y Tecnicas) for hydro-climatic studies in the Andes. HBV.IANIGLA
incorporates modules for precipitation and temperature interpolation, and also for clean and debris covered ice melt estimations.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports Rcpp (>= 0.12.0)

LinkingTo Rcpp

RoxygenNote 6.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-08-29 15:20:09 UTC

R topics documented:

`Glacier_Disch` *Glacier discharge conceptual model*

Description

Conceptual storage reservoir model of glacier discharge. It follows the concepts described in *Jansson et al. 2002* that were applied in the Bridge glacier by *Stahl et al. 2008* (hereafter S08).

Usage

```
Glacier_Disch(model, inputData, initCond, param)
```

Arguments

`model` choose model's option as numeric:

1. S08 model.

`inputData` **S08 model**: numeric matrix with two columns:

`SWE`: snow water equivalent above the glacier $[mm/\Delta t]$.

`Total`: melted snow + melted ice + rainfall $[mm/\Delta t]$.

NOTE: you can obtain the `SWE` and `Total` series from the `SnowGlacier_HBV` function.

`initCond` **S08 model**: numeric value with the initial glacier reservoir value `SG` $[mm]$.

`param` **S08 model**: numeric vector with three parameters.

1. `KGmin`: minimal outflow rate $[1/\Delta t]$.
2. `dKG`: maximum outflow rate increase $[1/\Delta t]$.
3. `AG`: scale factor $[mm]$.

Value

Returns a numeric matrix with two columns for every time step.

1. `Q`: glacier discharge $[mm/\Delta t]$.
2. `SG`: glacier's water storage content $[1/\Delta t]$.

Author(s)

J. Ezequiel Toum A.

References

Jansson, P., Hock, R., Schneider, T., 2003. The concept of glacier storage: a review. J. Hydrol., Mountain Hydrology and Water Resources 282, 116–129. https://doi.org/10.1016/S0022-1694(03)00258-0

Stahl, K., Moore, R.D., Shea, J.M., Hutchinson, D., Cannon, A.J., 2008. Coupled modelling of glacier and streamflow response to future climate scenarios. Water Resour. Res. 44, W02422. https://doi.org/10.1029/2007WR005956

Examples

```
DataMatrix <- cbind(runif(n = 100, min = 0, max = 50),
                    runif(n = 100, min = 0, max = 200))

dischGl    <- Glacier_Disch(model = 1, inputData = DataMatrix,
              initCond = 100, param = c(0.1, 0.9, 10))
```

PET — *Calculates potential evapotranspiration*

Description

Choose one of the model's options to obtain a series of potential evapotranspiration

Usage

```
PET(model, hemis, inputData, elev, param)
```

Arguments

`model` choose model's option as numeric:

1. Calder's model.

`hemis` choose hemisphere option as numeric.

1. southern hemisphere.
2. northern hemisphere.

This option is harmless when using other model than *Calder's*, nevertheless a numeric value must be provided.

`inputData` **Calder's model**: numeric matrix with julian dates. e.g: `as.matrix(c(1:365))`.

`elev` **Calder's model**: numeric vector of length two `c(zref,ztopo)`. `zref`: the reference heigh where potential evapotranspiration or input data to calculate PET is known. `ztopo`: target PET's topographic height.

`param` **Calder's model**: numeric vector of length two `c(PET,gradPET)`. `PET`: climatological daily mean potential evapotranspiration [mm]. `gradPET`: evapotranspiration decrease gradient [mm/100 m].

Value

Returns a numeric vector class with the potential evapotranspiration (in $[mm/\Delta t]$) for every time step.

Author(s)

J. Ezequiel Toum A.

References

Calder, I.R., Harding, R.J., Rosier, P.T.W., 1983. An objective assessment of soil-moisture deficit models. J. Hydrol. 60, 329–355. https://doi.org/10.1016/0022-1694(83)90030-6

Examples

```
## Run the model for a year in the southern hemisphere
potEvap <- PET(model = 1, hemis = 1, inputData = as.matrix(1:365),
elev = c(1000, 1500), param = c(4, 0.5))
```

`Precip_model` *Gradient precipitation models*

Description

Contains a linear precipitation gradient model to extrapolate raingauge measurements from a fixed elevation to other sites with different elevation. Also included is a modified version which incorporates an extra parameter to avoid increasing precipitation beyond a certain altitude.

Usage

```
Precip_model(model, inputData, zmeteo, ztopo, param)
```

Arguments

`model` choose model's option as numeric:

1. Linear precip. gradient (LP).
2. Linear precip. gradient with upper threshold (LPM).

`inputData` **LP**: numeric vector with precipitation serie $[mm/\Delta t]$.
LPM: numeric vector with precipitation serie $[mm/\Delta t]$.

`zmeteo` altitude of the raingauge $[masl]$.

`ztopo` target heigh $[masl]$.

`param` **LP**: numeric vector with precipitation gradient `gradP` [%/100m].
LPM: numeric vector with two values. `gradP`: precip. gradient [%/100m].
`maxALT`: threshold heigh where precip. does not increase any more $[masl]$.

Value

Numeric vector with the extrapolated precipitation.

Author(s)

J. Ezequiel Toum A.

References

For some interesting work on precipitation gradients in basin and synoptic scale see:

Immerzeel, W.W., Petersen, L., Ragettli, S., Pellicciotti, F., 2014. The importance of observed gradients of air temperature and precipitation for modeling runoff from a glacierized watershed in the Nepalese Himalayas. Water Resour. Res. 50, 2212–2226. https://doi.org/10.1002/2013WR014506

Viale, M., Nuñez, M.N., 2010. Climatology of Winter Orographic Precipitation over the Subtropical Central Andes and Associated Synoptic and Regional Characteristics. J. Hydrometeorol. 12, 481–507. https://doi.org/10.1175/2010JHM1284.1

Examples

```
## LP case
set.seed(369)

precLP <- Precip_model(model = 1, inputData = runif(n = 365, max = 30, min = 0),
          zmeteo = 3000, ztopo = 4700, param = c(5))

## LPM case
set.seed(369)

precLPM <- Precip_model(model = 2, inputData = runif(n = 365, max = 30, min = 0),
          zmeteo = 3000, ztopo = 4700, param = c(5, 4500))
```

`Routing_HBV` *Routing bucket type models*

Description

Contains the classical HBV routing scheme and adds four more bucket models to route water.

Usage

```
Routing_HBV(model, lake, inputData, initCond, param)
```

Arguments

`model` choose model's option as numeric.

1. Three series of reservoirs. Lake option is allowed.
2. Two series of reservoirs. Lake option is allowed.
3. Two reservoirs and three outlets. Lake option is allowed.
4. One reservoir and two outlets. Lake is NOT allowed.
5. One reservoir and three outlets. Lake is NOT allowed.

`lake` logical argument. `TRUE` will enable lake option.

`inputData` numeric matrix with three columns (two of them depends if you are using a lake).

- `Ieff`: effective rainfall/melted snow serie $[mm/\Delta T]$. NOTE: output of `Soil_HBV` module.
- `precip`: precipitation serie falling in the lake.Note that this option should be provided when working with lakes. Your precip. serie should be appropriately scaled $[mm/\Delta T]$.
- `evap`: lake evaporation serie. As in the precipitation teh values should be appropriately scaled $[mm/\Delta T]$.

`initCond` numeric vector with the following initial state variables.

- `SLZ0`: initial water content of the lower reservoir $[mm]$.
- `SUZ0`: initial water content of the intermediate reservoir. It could be optional regarding model's option $[mm]$.
- `STZ0`: initial water content of the upper reservoir. It could be optional regarding model's option $[mm]$.

`param` numeric vector which sizes depends upon model's choice.

Model 1

1. `K0`: `STZ` storage constant $[1/\Delta t]$.
2. `K1`: `SUZ` storage constant $[1/\Delta t]$.
3. `K2`: `SLZ` storage constant $[1/\Delta t]$.
4. `UZL`: maximum rate flux between `STZ` and `SUZ` $[mm/\Delta t]$.
5. `PERC`: maximum rate flux between `STZ` and `SUZ` $[mm/\Delta t]$.

Model 2

1. `K1`: `SUZ` storage constant $[1/\Delta t]$.
2. `K2`: `SLZ` storage constant $[1/\Delta t]$.
3. `PERC`: maximum rate flux between `STZ` and `SUZ` $[mm/\Delta t]$.

Model 3

1. `K0`: `SUZ` storage constant $[1/\Delta t]$.
2. `K1`: `SUZ` storage constant $[1/\Delta t]$.
3. `K2`: `SLZ` storage constant $[1/\Delta t]$.
4. `UZL`: minimum water content of `SUZ` to get fast runoff `Q0` $[mm]$.
5. `PERC`: maximum rate flux between `STZ` and `SUZ` $[mm/\Delta t]$.

Model 4

1. `K1`: `SLZ` storage constant $[1/\Delta t]$.
2. `K2`: `SLZ` storage constant $[1/\Delta t]$.
3. `PERC`: minimum water content of `SLZ` to get Q1 runoff $[mm]$.

Model 5

1. `K0`: `SLZ` storage constant $[1/\Delta t]$.
2. `K1`: `SLZ` storage constant $[1/\Delta t]$.
3. `K2`: `SLZ` storage constant $[1/\Delta t]$.
4. `UZL`: minimum water content of `SLZ` to get fast runoff `Q0` $[mm]$.
5. `PERC`: minimum water content of `SLZ` to get Q1 runoff $[mm]$.

Value

in every case this function returns a matrix with outlet discharges and reservoir(s) water levels series.

Author(s)

J. Ezequiel Toum A.

References

Bergström, S., Lindström, G., 2015. Interpretation of runoff processes in hydrological modelling—experience from the HBV approach. Hydrol. Process. 29, 3535–3545. https://doi.org/10.1002/hyp.10510

Beven, K.J., 2012. Rainfall - Runoff Modelling, 2 edition. ed. Wiley, Chichester.

Seibert, J., Vis, M.J.P., 2012. Teaching hydrological modeling with a user-friendly catchment-runoff-model software package. Hydrol Earth Syst Sci 16, 3315–3325. https://doi.org/10.5194/hess-16-3315-2012

Examples

```
inputMatrix <- cbind(runif(n = 200, max = 100, min = 0), runif(n = 200, max = 50, min = 5),
               runif(n = 100, max = 3, min = 1))

routeMod1   <- Routing_HBV(model = 1, lake = TRUE, inputData = inputMatrix,
               initCond = c(10, 15, 20), param = c(0.1, 0.05, 0.001, 1, 0.8))
```

`SnowGlacier_HBV` *Snowmelt and ice-melt models*

Description

Simulates the accumulation and melt processes of snow and ice using a temperature index model.

Usage

```
SnowGlacier_HBV(model, inputData, initCond, param)
```

Arguments

`model` choose model's option as numeric.

1. Temperature index model.
2. Temperature index model with snow cover area.
3. Temperature index model with variable glacier area.

`inputData` numeric matrix with columns as input data.

Model 1

1. `airT`: air temperature $[C/\Delta t]$.
2. `precip`: precipitation $[mm/\Delta t]$.

Model 2

1. `airT`: air temperature $[C/\Delta t]$.
2. `precip`: precipitation $[mm/\Delta t]$.
3. `SCA`: snow cover area. Values between [0 ; 1] $[-]$.

Model 3

1. `airT`: air temperature $[C/\Delta t]$.
2. `precip`: precipitation $[mm/\Delta t]$.
3. `GCA`: glacier cover area. This area values are relative to the total surface area of the basin $[-]$.

`initCond` numeric vector with the following values.

- `SWE0`: initial snow water equivalent $[mm]$.
- numeric integer indicating the surface type. *1*: clean ice; *2*: soil; *3*: debris-covered ice.
- area of the glacier(s) (in the elevation band) relative to the basin; e.g.: 0.1 $[-]$. This option is required in *Model 1* and *Model 2* when surface is a glacier.

`param` numeric vector with the following values:

1. `SFCF`: snowfall correction factor $[-]$.
2. `Tr`: solid and liquid precipitation threshold temperature $[C]$.
3. `Tt`: melt temperature $[C]$.
4. `fm`: snowmelt factor $[mm/C.\Delta t]$.
5. `fi`: ice-melt factor $[mm/C.\Delta t]$.
6. `fic`: debris-covered ice-melt factor $[mm/C.\Delta t]$.

Value

Numeric matrix with the following columns:

Model 1

** if surface is soil,

1. `Prain`: precip. as rainfall.
2. `Psnow`: precip. as snowfall.

3. SWE: snow water equivalent.
4. Msnow: melted snow.
5. Total: Prain + Msnow.

** if surface is ice,

1. Prain: precip. as rainfall.
2. Psnow: precip. as snowfall.
3. SWE: snow water equivalent.
4. Msnow: melted snow.
5. Mice: melted ice.
6. Mtot: Msnow + Mice.
7. Cum: Psnow - Mtot.
8. Total: Prain + Mtot.
9. TotScal: Total * initCond[3].

Model 2

** if surface is soil,

1. Prain: precip. as rainfall.
2. Psnow: precip. as snowfall.
3. SWE: snow water equivalent.
4. Msnow: melted snow.
5. Total: Prain + Msnow.
6. TotScal: Msnow * SCA + Prain.

** if surface is ice -> as in *Model 1*

Model 3

** if surface is soil -> as in *Model 1*

** if surface is ice -> as in *Model 1*

Author(s)

J. Ezequiel Toum A.

References

Bergström, S., Lindström, G., 2015. Interpretation of runoff processes in hydrological modelling—experience from the HBV approach. Hydrol. Process. 29, 3535–3545. https://doi.org/10.1002/hyp.10510

Parajka, J., Merz, R., Blöschl, G., 2007. Uncertainty and multiple objective calibration in regional water balance modelling: case study in 320 Austrian catchments. Hydrol. Process. 21, 435–446. https://doi.org/10.1002/hyp.6253

Seibert, J., Vis, M.J.P., 2012. Teaching hydrological modeling with a user-friendly catchment-runoff-model software package. Hydrol Earth Syst Sci 16, 3315–3325. https://doi.org/10.5194/hess-16-3315-2012

Examples

```
## Debris-covered ice
ObsTemp   <- sin(x = seq(0, 10*pi, 0.1))
ObsPrecip <- runif(n = 315, max = 50, min = 0)
ObsGCA    <- seq(1, 0.8, -0.2/314)

## Fine debris covered layer assumed. Note that the ice-melt factor is cumpulsory but harmless.
DebrisCovGlac <- SnowGlacier_HBV(model = 3, inputData = cbind(ObsTemp, ObsPrecip, ObsGCA),
                 initCond = c(10, 3, 1), param = c(1, 1, 0, 3, 1, 6))
```

Soil_HBV *Conceptual soil moisture routine to account for abstractions and effective runoff.*

Description

Calculates effective runoff to be routed in the buckets (Routing_HBV).

Usage

```
Soil_HBV(model, inputData, initCond, param)
```

Arguments

`model` choose model's option as numeric.

1. Classical HBV soil moisture routine.
2. HBV soil moisture routine with varying area. This option should be used with SnowGlacier_HBV's *model 3*.

`inputData` numeric matrix with columns as input data.

Model 1

1. `Total`: `Mice` + `Msnow` + `Prain` $[mm/\Delta t]$.
2. `PET`: potential evapotranspiration serie $[mm/\Delta t]$.

Model 2

1. `Total`: `Mice` + `Msnow` + `Prain` $[mm/\Delta t]$.
2. `PET`: potential evapotranspiration serie $[mm/\Delta t]$.
3. `SoCA` : serie of relative soil area (ratio of soil surface over basin area).

`initCond` numeric vector with the following values:

1. `SM`: initial soil water content $[mm]$.
2. `realtArea`: relative area $[-]$.

`param` numeric parameter's vector:

1. `FC`: fictitious soil field capacity $[mm]$.
2. `LP`: parameter to get actual ET $[-]$.
3. β: parameter that allows non-linear relations between input-output runoff $[-]$.

Value

Returns a numeric matrix with the following columns:

- `Rech`: recharge serie $[mm/\Delta t]$. This is the input to `Routing_HBV` module.
- `Eact`: actual evaporation/evapotranspiration serie $[mm/\Delta t]$.
- `SM`: soil moisture serie $[mm/\Delta t]$.

Author(s)

J. Ezequiel Toum A.

References

Bergström, S., Lindström, G., 2015. Interpretation of runoff processes in hydrological modelling—experience from the HBV approach. Hydrol. Process. 29, 3535–3545. https://doi.org/10.1002/hyp.10510

Examples

```
# HBV soil routine with variable area
  ## Calder's model
potEvap <- PET(model = 1, hemis = 1, inputData = as.matrix(1:315), elev = c(1000, 1500),
param = c(4, 0.5))

  ## Debris-covered ice
ObsTemp   <- sin(x = seq(0, 10*pi, 0.1))
ObsPrecip <- runif(n = 315, max = 50, min = 0)
ObsGCA    <- seq(1, 0.8, -0.2/314)

 ## Fine debris covered layer assumed. Note that the ice-melt factor is cumpulsory but harmless.
DebrisCovGlac <- SnowGlacier_HBV(model = 3, inputData = cbind(ObsTemp, ObsPrecip, ObsGCA),
                initCond = c(10, 3, 1), param = c(1, 1, 0, 3, 1, 6))

  ## Soil routine
  ObsSoCA     <- 1 - ObsGCA
  inputMatrix <- cbind(DebrisCovGlac[ , 9], potEvap, ObsSoCA)

soil <- Soil_HBV(model = 2, inputData = inputMatrix, initCond = c(50), param = c(200, 0.5, 2))
```

`Temp_model` *Linear temperature gradient model*

Description

Contains a linear temperature gradient model to extrapolate air temperature measurements from fixed stations to other sites.

Usage

```
Temp_model(model, inputData, zmeteo, ztopo, param)
```

Arguments

- `model` — choose model's option as numeric:
  1. linear gradient model.
- `inputData` — numeric vector with the air temperature serie $[C/\Delta t]$.
- `zmeteo` — altitude of the air temperature sensor $[masl]$.
- `ztopo` — target heigh $[masl]$.
- `param` — numeric vector with the following values,
  Model 1
  1. `gradT`: air temperature gradient factor $[C/100m]$.

Value

Returns a numeric vector with the extrapolated air temperature serie.

Author(s)

J. Ezequiel Toum A.

References

Immerzeel, W.W., Petersen, L., Ragettli, S., Pellicciotti, F., 2014. The importance of observed gradients of air temperature and precipitation for modeling runoff from a glacierized watershed in the Nepalese Himalayas. Water Resour. Res. 50, 2212–2226. https://doi.org/10.1002/2013WR014506

Examples

```
airTemp <- Temp_model(model = 1, inputData = runif(200, max = 25, min = -10),
           zmeteo = 2000, ztopo = 3500, param = c(-0.65))
```

UH — *Unit hydrograph*

Description

Transfer function to adjust the simulated hydrograph timing.

Usage

```
UH(model, Qg, param)
```

Arguments

- `model` — choose model's option as numeric:
  1. Static triangular unit hydrograph.
- `Qg` — numeric vector with the input data. It is the output of the `Routing_HBV` module.
- `param` — numeric vector with the following values, **Model 1**
  1. `Bmax`: base of the UH triangle $[timestep]$.

Value

Returns a numeric vector with the hydrograph.

Author(s)

J. Ezequiel Toum A.

References

Bergström, S., Lindström, G., 2015. Interpretation of runoff processes in hydrological modelling—experience from the HBV approach. Hydrol. Process. 29, 3535–3545. https://doi.org/10.1002/hyp.10510

Examples

```
## For a real world example see Manual.pdf file.
  ## Routing example
inputMatrix <- cbind(runif(n = 200, max = 100, min = 0), runif(n = 200, max = 50, min = 5),
               runif(n = 100, max = 3, min = 1))

routeMod1   <- Routing_HBV(model = 1, lake = TRUE, inputData = inputMatrix,
               initCond = c(10, 15, 20), param = c(0.1, 0.05, 0.001, 1, 0.8))

  ## UH
  dischOut <- UH(model = 1, Qg = routeMod1[ , 1], param = 2.2)
```

Index