

# Package ‘RHMS’

January 20, 2025

**Type** Package

**Title** Hydrologic Modelling System for R Users

**Version** 1.7

**Depends** R (>= 3.0.0), graphics, stats, pso, Hmisc, network, GGally

**Date** 2021-09-22

**Author** Rezgar Arabzadeh; Shahab Araghinejad

**Maintainer** Rezgar Arabzadeh <rezgararabzadeh@ut.ac.ir>

**Description** Hydrologic modelling system is an object oriented tool for simulation and analysis of hydrologic events. The package proposes functions and methods for construction, simulation, visualization, and calibration of a hydrologic model.

**License** GPL-2

**Imports** ggplot2

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2021-09-27 15:50:02 UTC

## Contents

RHMS-package . . . . .	2
abstraction . . . . .	4
abstraction.base . . . . .	5
abstraction.default . . . . .	5
addObjectToBasin . . . . .	6
baseFlowSeparation . . . . .	8
baseFlowSeparation.base . . . . .	9
baseFlowSeparation.default . . . . .	10
createBasin . . . . .	11
createBasin.base . . . . .	12
createBasin.default . . . . .	12
createDiversion . . . . .	13
createDiversion.base . . . . .	14
createDiversion.default . . . . .	15

createJunction . . . . .	15
createJunction.base . . . . .	16
createJunction.default . . . . .	17
createReach . . . . .	18
createReach.base . . . . .	19
createReach.default . . . . .	20
createReservoir . . . . .	21
createReservoir.base . . . . .	22
createReservoir.default . . . . .	23
createSubbasin . . . . .	24
createSubbasin.base . . . . .	25
createSubbasin.default . . . . .	27
loss . . . . .	28
loss.base . . . . .	29
loss.default . . . . .	30
plot.createBasin . . . . .	31
plot.sim . . . . .	31
reachRouting . . . . .	32
reachRouting.base . . . . .	33
reachRouting.default . . . . .	34
reservoirRouting . . . . .	35
reservoirRouting.base . . . . .	36
reservoirRouting.default . . . . .	37
set.as . . . . .	38
sim . . . . .	39
sim.base . . . . .	41
sim.default . . . . .	42
summary.sim . . . . .	42
transform . . . . .	43
transform.base . . . . .	44
transform.default . . . . .	45
tune . . . . .	46
Zaab . . . . .	49

<b>Index</b>	<b>50</b>
--------------	-----------

## Description

The RHMS package provides tools to R users for simulation of hydrologic events. The packages includes functions and methods for building, simulation, visualization, and calibration of a hydrologic model.

## Details

Package: RHMS  
Type: Package  
Version: 1.7  
Date: 2021-09-27  
License: GPL-3

the package include three major types of functions as follows:

1- functions for construction and manipulatation of hydrologic features.

- `createBasin`. constructor for basin
- `createJunction`. constructor for junction
- `createReach`. constructor for reach, rivers, and channels
- `createReservoir`. constructor for reservoirs
- `createSubbasin`. constructor for sub-bains
- `createDiversion`. constructor for diversions
- `set.as`. objects connector
- `addObjectToBasin`. adds objects form above constructors to a basin inherited from class of `createBasin`

2- functions for analysis and simulation of hydrologic events.

- `reachRouting`. routes a flood in a channel or river
- `reservoirRouting`. routes a flood in a reservoir
- `transform`. trnasforms a rainfall event to runoff
- `loss`. computes excess rainfall and loss depths
- `baseFlowSeparation`. separates baseflow from a given discharge series
- `abstraction`. computes simple surface and canopy methods
- `sim`. simulates an objects inherited from class of `createBasin`

3- functions for tuning, summerizing, and visualization.

- `plot.sim`. plots the objects inherited from class of `sim`
- `plot.createBasin`. plots the objects inherited from class of `createBasin`
- `summary.sim`. summerzies the simulation results in the tabular form for every objects existing in the basin
- `tune`. calibrates an objects inherited from class of `createBasin`

#### Author(s)

Rezgar Arabzadeh ; Shahab Araghinejad

Maintainer: Rezgar Arabzadeh <rezgararabzadeh@ut.ac.ir>

**References**

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.

**See Also**

[sim](#)

---

abstraction	<i>computes surface and canopy abstractions</i>
-------------	---

---

**Description**

computes surface and canopy abstractions for a given rainfall event.

**Usage**

```
abstraction(rainfall,abstractionParams)
```

**Arguments**

rainfall            a vector : a time series of precipitation hyetograph (mm)  
abstractionParams    a list: including parameters of simple surface and simple canopy methods.

- canopyAbstraction depth of canopy abstraction in (mm). default to zero
- surfaceAbstraction depth of surface abstraction in (mm). default to zero

**Value**

a list: an object from class of abstraction

**Author(s)**

Rezgar Arabzadeh

**See Also**

[createSubbasin](#)

**Examples**

```
rainfall<-5*exp(((seq(2.5,7.5,length.out=36))-5)^2/-0.8)
abstractionParams<-list(canopyAbstraction=2,surfaceAbstraction=3.5)
abstraction(rainfall,abstractionParams)
```



**Arguments**

- rainfall            a vector : a time series of precipitation hyetograph (mm)
- abstractionParams    a list: including parameters of simple surface and simple canopy methods.
- canopyAbstraction depth of canopy abstraction in (mm). default to zero
  - surfaceAbstraction depth of surface abstraction in (mm). default to zero

**Value**

a list: an object from class of abstraction

**Author(s)**

Rezgar Arabzadeh

**See Also**

[createSubbasin](#)

---

addObjectToBasin        *adds an object to basin*

---

**Description**

adds an object inherited from either of RHMS package constructors to an object instantiated by class of createBasin.

**Usage**

```
addObjectToBasin(object, basin)
```

**Arguments**

- object            an object inherited from one of the following classes: [createReservoir](#), [createReach](#), [createSubbasin](#), [createJunction](#)
- basin            an object inherited from class of createBasin

**Value**

an object from class of createBasin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[sim](#)

**Examples**

```

storageElevationCurve<-data.frame(s=0:100*10,h=100:200)
dischargeElevationCurve<-data.frame(q=seq(0,5000,length.out=10),
                                     h=seq(180,200,length.out=10))
geometry<-list(storageElevationCurve=storageElevationCurve,
               dischargeElevationCurve=dischargeElevationCurve,
               capacity=800)
Res1<-createReservoir(name = "Reservoir1",
                      geometry=geometry,initialStorage=550)
R1<-createReach(name="Reach1",routingParams=list(k=5,x=0.3))
R2<-createReach(name="Reach2",routingParams=list(k=5,x=0.3))
R3<-createReach(name="Reach3",routingParams=list(k=5,x=0.3))
R4<-createReach(name="Reach4",routingMethod="muskingumcunge",
                routingParams=list(bedWith=100,
                                   sideSlope=2,
                                   channelSlope=0.01,
                                   manningRoughness=0.05,
                                   riverLength=120))

D1<-createDiversion(name="Diversion1",capacity=80)

Junc1<-createJunction(name = "Junc1")
S1<-createSubbasin(name="Sub1",Area=500,
                  precipitation=round(sin(seq(0,pi,length.out=24))*20),
                  transformMethod="SCS",lossMethod="SCS",BFSSMethod='recession',
                  transformParams=list(Tlag=4),lossParams=list(CN=70),BFSPParams=list(k=1.1))
S2<-createSubbasin(name="Sub2",Area=500,
                  precipitation=round(sin(seq(0,pi,length.out=24))*20),
                  transformMethod="SCS",lossMethod="SCS",BFSSMethod='recession',
                  transformParams=list(Tlag=4),lossParams=list(CN=70),BFSPParams=list(k=1.1))
S3<-createSubbasin(name="Sub3",Area=650,
                  precipitation=round(sin(seq(0,pi,length.out=24))*20),
                  transformMethod="snyder",lossMethod="horton",
                  transformParams=list(Cp=0.17,Ct=1.5,L=140,Lc=30),
                  lossParams=list(f0=5,f1=1,k=1))

S1<-set.as(R2,S1,'downstream')
R2<-set.as(Junc1,R2,'downstream')
Junc1<-set.as(R1,Junc1,'downstream')
R1<-set.as(Res1,R1,'downstream')
S3<-set.as(R3,S3,'downstream')
R3<-set.as(Junc1,R3,'downstream')
S2<-set.as(R4,S2,'downstream')
R4<-set.as(D1,R4,'downstream')
D1<-set.as(Junc1,D1,'downstream')
D1<-set.as(S1,D1,'divertTo')

basin1<-createBasin(name = "Unknown", simulation=list(start='2000-01-01',end='2000-01-10',by=7200))
basin1<-addObjectToBasin(Junc1, basin1)
basin1<-addObjectToBasin(R1, basin1)
basin1<-addObjectToBasin(R2, basin1)
basin1<-addObjectToBasin(R3, basin1)
basin1<-addObjectToBasin(R4, basin1)

```

```

basin1<-addObjectToBasin(S1, basin1)
basin1<-addObjectToBasin(S2, basin1)
basin1<-addObjectToBasin(S3, basin1)
basin1<-addObjectToBasin(Res1, basin1)
basin1<-addObjectToBasin(D1, basin1)

## Not run: plot(basin1)

object<-sim(basin1)

plot(object)

summary(object)

```

---

baseFlowSeparation      *Parametric methods for separating baseflow*

---

### Description

This function calculates baseflow for a given time series, discharge, using a number of method stated in BFSMethod.

### Usage

```
baseFlowSeparation(discharge,BFSMethod,BFSParams,plot)
```

### Arguments

discharge	a vector of flow time series (cms) or an object inherited from class of 'transform'
BFSMethod	a string: The method of base flow separation. Available methods: 'nathan', 'chapman', 'eckhardt', 'recession'
BFSParams	a list including parameters associated with the method coerced in 'BFSMethod'. <ul style="list-style-type: none"> <li>• alpha is in <math>[0, 1]</math> interval required for 'nathan', 'chapman', and 'eckhardt' methods</li> <li>• BFI is in <math>[0, 1]</math> interval required for 'eckhardt' method</li> <li>• k is in <math>[0, 1]</math> interval and timeInterval is in day required for 'recession' method</li> </ul>
plot	(optional) logical statement to plot the result or not. default to FALSE

### Value

a list: an object from class of baseFlowSeparation consisting matrix of results available at object\$operation.

### Author(s)

Rezgar Arabzadeh



**References**

Chapman, Tom. "A comparison of algorithms for stream flow recession and baseflow separation." Hydrological Processes 13.5 (1999): 701-714.

**See Also**

[baseFlowSeparation](#)

**Examples**

```
discharge<-(dnorm(seq(-3,4,length.out=200),-.3,1)+dnorm(seq(-1,7,length.out=200),4.5,1)*2)*1200
BFSEMethod<-c('nathan','chapman','eckhardt','recession')
BFSEParams<-list(alpha=0.6,BFI=0.3,k=1.1,timeInterval=15*60)
simulation<-list(start='2000-01-01',end='2000-01-02',by=400)
baseFlowSeparation(discharge,BFSEMethod[1],BFSEParams,plot=TRUE)
baseFlowSeparation(discharge,BFSEMethod[2],BFSEParams,plot=TRUE)
baseFlowSeparation(discharge,BFSEMethod[3],BFSEParams,plot=TRUE)
baseFlowSeparation(discharge,BFSEMethod[4],BFSEParams,plot=TRUE)
```

---

baseFlowSeparation.base

*base function for class of baseFlowSeparation*

---

**Description**

Methods of separating baseflow for a given flow discharge.

**Usage**

```
## S3 method for class 'base'
baseFlowSeparation(discharge,BFSEMethod,BFSEParams,plot)
```

**Arguments**

discharge	a vector of flow time series (cms) or an object inherited from class of 'transform'
BFSEMethod	a string: The method of base flow separation. Available methods: 'nathan', 'chapman', 'eckhardt', 'recession'
BFSEParams	a list including parameters associated with the method coerced in 'BFSEMethod'. <ul style="list-style-type: none"> <li>• alpha is in [0, 1] interval required for 'nathan', 'chapman', and 'eckhardt' methods</li> <li>• BFI is in [0, 1] interval required for 'eckhardt' method</li> <li>• k is in [0, 1] interval and timeInterval is in day required for 'recession' method</li> </ul>
plot	(optional) logical statement to plot the result or not. default to FALSE

**Value**

a matrix: A matrix of results including computed separated flow for Q series

**Author(s)**

Rezgar Arabzadeh

**See Also**

[baseFlowSeparation](#)

---

baseFlowSeparation.default

*default function for class of baseFlowSeparation*

---

**Description**

Methods for separating baseflow for a given flow discharge

**Usage**

```
## Default S3 method:
baseFlowSeparation(discharge,BFSMethod='none'
                    ,
                    BFSParams=list(alpha=NULL
                                   ,
                                   BFI=NULL
                                   ,
                                   k=NULL
                                   ,
                                   timeInterval=NULL),
                    plot=FALSE)
```

**Arguments**

discharge	a vector of flow time series (cms) or an object inherited from class of 'transform'
BFSMethod	a string: The method of base flow separation. Available methods: 'nathan', 'chapman', 'eckhardt', 'recession'
BFSParams	a list including parameters associated with the method coerced in 'BFSMethod'. <ul style="list-style-type: none"> <li>• alpha is in [0, 1] interval required for 'nathan', 'chapman', and 'eckhardt' methods</li> <li>• BFI is in [0, 1] interval required for 'eckhardt' method</li> <li>• k is in [0, 1] interval and timeInterval is in day required for 'recession' method</li> </ul>
plot	(optional) logical statement to plot the result or not. default to FALSE

**Value**

a list: an object from class of baseFlowSeparation consisting matrix of results available at object\$operation.

**Author(s)**

Rezgar Arabzadeh

**See Also**

[createSubbasin](#)

---

createBasin	<i>creates a basin</i>
-------------	------------------------

---

**Description**

instantiates an object from class of createBasin

**Usage**

```
createBasin(name, simulation)
```

**Arguments**

- |            |   |
|------------|---|
| name       | a string: a name for the basin  |
| simulation | a list of simulation time and dates as below: <ul style="list-style-type: none"><li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li><li>• start: the date which simulation ends, must be in 'YYYY-MM-DD' format</li><li>• by: the interval of each steps in seconds</li></ul> |

**Value**

a list: an object from class of creatBasin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

createBasin.base      *base function for class of createBasin*

---

**Description**

instantiates an object from class of createBasin

**Usage**

```
## S3 method for class 'base'  
createBasin(name, simulation)
```

**Arguments**

name                    a string: a name for the basin

simulation             a list of simulation time and dates as below:

- start: the date which simulation starts, must be in 'YYYY-MM-DD' format
- start: the date which simulation ends, must be in 'YYYY-MM-DD' format
- by: the interval of each steps in seconds

**Value**

a list: an object from class of creatBasin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

createBasin.default      *default function for class of createBasin*

---

**Description**

instantiates an object from class of createBasin

**Usage**

```
## Default S3 method:  
createBasin(name = "Untittled", simulation=list(start=NULL,end=NULL,by=NULL))
```

**Arguments**

name	a string: a name for the basin
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• start: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

a list: an object from class of creatBasin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

createDiversion	<i>creates a diversion object</i>
-----------------	-----------------------------------

---

**Description**

instantiates an object from class of createDiversion

**Usage**

```
createDiversion(name, downstream, divertTo, capacity)
```

**Arguments**

name	(optional) a string: the name of diversion to be instantiated
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
divertTo	an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
capacity	diversion capacity (cms)

**Value**

a list: an object from class of createDiversion

**Author(s)**

Rezgar Arabzadeh

**See Also**[addObjectToBasin](#)

---

createDiversion.base    *base function for class of createDiversion*

---

**Description**

instantiates an object from class of createDiversion

**Usage**

```
## S3 method for class 'base'  
createDiversion(name,downstream,divertTo,capacity)
```

**Arguments**

name	(optional) a string: the name of diversion to be instantiated
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
divertTo	an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
capacity	diversion capacity (cms)

**Value**

a list: an object from class of createDiversion

**Author(s)**

Rezgar Arabzadeh

**See Also**[addObjectToBasin](#)

---

```
createDiversion.default
```

*default function for class of createDiversion*

---

**Description**

instantiates an object from class of createDiversion

**Usage**

```
## Default S3 method:  
createDiversion(name="Untitled",downstream=NA,divertTo,capacity)
```

**Arguments**

name	(optional) a string: the name of diversion to be instantiated
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
divertTo	an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
capacity	diversion capacity (cms)

**Value**

a list: an object from class of createDiversion

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

```
createJunction
```

*creates a junction object*

---

**Description**

instantiates an object from class of createJunction

**Usage**

```
createJunction(name, downstream,  
              inflow, delayInflow)
```

**Arguments**

name	(optional) a string: the name of junction to be instantiated
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
inflow	(optional): a vector of direct inflow/lateral flow (cms)
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series

**Value**

a list: an object from class createJunction

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

createJunction.base     *base function for class of createJunction*

---

**Description**

instantiates an object from class of createJunction

**Usage**

```
## S3 method for class 'base'
createJunction(name , downstream,
               inflow , delayInflow )
```

**Arguments**

name	(optional) a string: the name of junction to be instantiated
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
inflow	(optional): a vector of direct/lateral (cms)
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series

**Value**

a list: an object from class of createJunction



**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

`createJunction.default`  
*default function for class of createJunction*

---

**Description**

instantiates an object from class of createJunction

**Usage**

```
## Default S3 method:  
createJunction(name = "Untitled", downstream=NA,  
               inflow = NA, delayInflow = 1)
```

**Arguments**

name	(optional) a string: the name of junction to be instantiated
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
inflow	(optional): a vector of direct/lateral inflow (cms)
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series

**Value**

a list: an object from class of createJunction

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

createReach                      *creates a reach object*

---

### Description

instantiates an object from class of createReach

### Usage

```
createReach(name, routingMethod, inflow,
            routingParams, delayInflow, downstream)
```

### Arguments

name	(optional) a string: the name of reach to be instantiated
routingMethod	a string: the method of channel routing. available types: "muskingum", and "muskingumcunge". default to "muskingum"
inflow	(optional): a vector of direct/lateral inflow (cms)
routingParams	a list : parameters associated to the routingMethod: <ul style="list-style-type: none"> <li>• k and x for "muskingum",</li> <li>• bedWith (m), sideSlope (m/m), channelSlope (m/m), manningRoughness, riverLength (Km) for "muskingumcunge"</li> </ul>
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.

### Value

a list: an object from class of createReach

### Author(s)

Rezgar Arabzadeh

### See Also

[addObjectToBasin](#)

---

createReach.base      *base function for class of createReach*

---

### Description

instantiates an object from class of createReach

### Usage

```
## S3 method for class 'base'
createReach(name, routingMethod, inflow,
            routingParams,
            delayInflow, downstream)
```

### Arguments

name	(optional) a string: the name of reach to be instantiated
routingMethod	a string: the method of channel routing. available types: "muskingum", and "muskingumcunge". default to "muskingum"
inflow	(optional): a vector of lateral inflow (cms)
routingParams	a list : parameters associated to the routingMethod: <ul style="list-style-type: none"> <li>• k and x for "muskingum",</li> <li>• bedWith (m), sideSlope (m/m), channelSlope (m/m), manningRoughness, riverLength (Km) for "muskingumcunge"</li> </ul>
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.

### Value

a list: an object from class of createReach

### Author(s)

Rezgar Arabzadeh

### See Also

[addObjectToBasin](#)

---

createReach.default    *default function for class of createReach*

---

### Description

instantiates an object from class of createReach

### Usage

```
## Default S3 method:
createReach(name="Unttitled",routingMethod="muskingum",inflow=NA,
            routingParams=list(k=3,x=0.2,bedWith=NULL,
                               sideSlope=2,channelSlope=NULL,
                               manningRoughness=0.025,riverLength=NULL),
            delayInflow=1,downstream=NA)
```

### Arguments

name	(optional) a string: the name of reach to be instantiated
routingMethod	a string: the method of channel routing. available types: "muskingum", and "muskingumcunge". default to "muskingum".
inflow	(optional): a vector of direct/lateral (cms)
routingParams	a list : parameters associated to the routingMethod: <ul style="list-style-type: none"> <li>• k and x for "muskingum",</li> <li>• bedWith (m), sideSlope (m/m), channelSlope (m/m), manningRoughness, riverLength (Km) for "muskingumcunge"</li> </ul>
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional) an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.

### Value

a list: an object from class of createReach

### Author(s)

Rezgar Arabzadeh

### See Also

[addObjectToBasin](#)

---

createReservoir      *creates a reservoir object*

---

**Description**

instantiates an object from class of createReservoir

**Usage**

```
createReservoir(name , inflow , geometry, initialStorage,  
                delayInflow , downstream )
```

**Arguments**

name	(optional): a string: the name of reservoir to be instantiated
inflow	(optional) : a vector of direct/lateral inflow (cms)
geometry	a list of geometric specifications of the reservoir: <ul style="list-style-type: none"><li>• storageElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent volume to the height at first column (MCM)</li><li>• dischargeElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent discharge rate to the height at first column (cms)</li><li>• storage: the maximum volume of reservoir capacity (MCM)</li></ul>
initialStorage	(optional) the initial storage of reservoir at the first time step of simulation (MCM)
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional): an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.

**Value**

a list: an object from class of createReservoir

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

createReservoir.base *base function for class of createReservoir*

---

### Description

instantiates an object from class of createReservoir

### Usage

```
## S3 method for class 'base'
createReservoir(name , inflow , geometry,
                initialStorage, delayInflow , downstream )
```

### Arguments

name	(optional): a string: the name of reservoir to be instantiated
inflow	(optional) : a vector of direct/lateral inflow (cms)
geometry	a list of geometric specifications of the reservoir: <ul style="list-style-type: none"> <li>• storageElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second collums presents equivalent volume to the height at first collumn (MCM)</li> <li>• dischargeElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second collums presents equivalent discharge rate to the height at first collumn (cms)</li> <li>• storage: the maximum volume of reservoir capacity (MCM)</li> </ul>
initialStorage	(optional): the initial storage of reservoir at the first time step of simulation (MCM)
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional): an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.

### Value

a list: an object from class of createReservoir

### Author(s)

Rezgar Arabzadeh

### See Also

[addObjectToBasin](#)

---

```
createReservoir.default
      default function for class of createReservoir
```

---

### Description

instantiates an object from class of createReservoir

### Usage

```
## Default S3 method:
createReservoir(name = "Untitled", inflow = NA,
                geometry=list(storageElevationCurve=NULL,
                              dischargeElevationCurve=NULL,
                              capacity=NULL),
                initialStorage = NA,
                delayInflow = 1, downstream = NA)
```

### Arguments

name	(optional): a string: the name of reservoir to be instantiated
inflow	(optional): a vector of direct/lateral inflow (cms)
geometry	a list of geometric specifications of the reservoir: <ul style="list-style-type: none"> <li>• storageElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent volume to the height at first column (MCM)</li> <li>• dischargeElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent discharge rate to the height at first column (cms)</li> <li>• storage: the maximum volume of reservoir capacity (MCM)</li> </ul>
initialStorage	(optional): the initial storage of reservoir at the first time step of simulation (MCM)
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional): an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.

### Value

a list: an object from class of createReservoir

### Author(s)

Rezgar Arabzadeh

**See Also**[addObjectToBasin](#)


---

createSubbasin	<i>creates a sub-basin object</i>
----------------	-----------------------------------

---

**Description**

instantiates an object from class of createSubbasin

**Usage**

```
createSubbasin(name,precipitation,
               inflow,Area,delayInflow,downstream,
               transformMethod,lossMethod,BFSMethod,UH,
               abstractionParams,transformParams,lossParams,BFSParams)
```

**Arguments**

name	(optional): a string: the name of sub-basin to be instantiated
precipitation	a vector : a time series of precipitation hytograph (mm)
inflow	(optional): a vector of direct inflow rather than flows comming from upstream (cms)
Area	the area of basin (Km^2)
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional): an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
transformMethod	a string: the type of transformation method. Available types: "SCS", "snyder", and "user" for user defined unit hydrograph. default to "SCS"
lossMethod	a string: the type of loss method. Available types: "SCS" and "horton"
BFSMethod	a string: The method of base flow separation. Available methods: 'nathan', 'chapman', 'eckhardt', 'recession'
UH	a data.frame: including the ordinates of user UH. the HU first collumn indicates time (Hr) and second collumn include flow rates (cms)
abstractionParams	a list: including parameters of simple surface and simple canopy methods. <ul style="list-style-type: none"> <li>• canopyAbstaction depth of canopy abstraction in (mm)</li> <li>• surfaceAbstaction depth of surface abstraction in (mm)</li> </ul>
BFSParams	a list including parameters associated with the method coerced in 'BFSMethod'. <ul style="list-style-type: none"> <li>• alpha is in [0, 1] interval required for 'nathan', 'chapman', and 'eckhardt' methods</li> </ul>



- BFI is in  $[0, 1]$  interval required for 'eckhardt' method
  - k is in  $[0, 1]$  interval and timeInterval is in day required for 'recession' method
- transformParams      a list of parameters associated to the selected type of transformMethod:
- Tlag for "SCS" method in (Hours)
  - Ct, Cp, L, and Lc for "snyder" method
- lossParams          a list of parameters associated to the selected type of lossMethod:
- CN for "SCS" method
  - $f_0, f_1, k$  other for "horton" method

**Value**

a list: an object from class of createSubbasin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

createSubbasin.base      *base function for class of createSubbasin*

---

**Description**

instantiates an object from class of createSubbasin

**Usage**

```
## S3 method for class 'base'
createSubbasin(name,precipitation,
               inflow,Area,delayInflow,downstream,
               transformMethod,lossMethod,BFSMethod,UH,
               abstractionParams,transformParams,lossParams,BFSParams)
```

**Arguments**

name                    (optional): a string: the name of sub-basin to be instantiated

precipitation          a vector : a time series of precipitation hytograph (mm)

inflow                  (optional): a vector of direct inflow/lateral (cms)

Area                    the area of basin (Km<sup>2</sup>)

delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional): an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
transformMethod	a string: the type of transformation method. Available types: "SCS", "snyder", and "user" for user defined unit hydrograph. default to "SCS"
lossMethod	a string: the type of loss method. Available types: "SCS" and "horton"
BFSMethod	a string: The method of base flow separation. Available methods: 'nathan', 'chapman', 'eckhardt', 'recession'
UH	a data.frame: including the ordinates of user UH. the HU first collumn indicates time (Hr) and second collumn include flow rates (cms)
abstractionParams	a list: including parameters of simple surface and simple canopy methods. <ul style="list-style-type: none"> <li>• canopyAbstaction depth of canopy abstraction in (mm)</li> <li>• surfaceAbstaction depth of surface abstraction in (mm)</li> </ul>
BFSParams	a list including parameters associated with the method coerced in 'BFSMethod'. <ul style="list-style-type: none"> <li>• alpha is in [0, 1] interval required for 'nathan', 'chapman', and 'eckhardt' methods</li> <li>• BFI is in [0, 1] interval required for 'eckhardt' method</li> <li>• k is in [0, 1] interval and timeInterval is in day required for 'recession' method</li> </ul>
transformParams	a list of parameters associated to the selcted type of transformMethod: <ul style="list-style-type: none"> <li>• Tlag for "SCS" method in (Hours)</li> <li>• Ct, Cp, L, and Lc for "snyder" method</li> </ul>
lossParams	a list of parameters associated to the selcted type of lossMethod: <ul style="list-style-type: none"> <li>• CN for "SCS" method</li> <li>• f0, f1, k other for "horton" method</li> </ul>

**Value**

a list: a list features for the constructed sub-basin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

```
createSubbasin.default
```

*default function for class of createSubbasin*

---

## Description

instantiates an object from class of createSubbasin

## Usage

```
## Default S3 method:
createSubbasin(name="Untitled",
               precipitation, inflow=NA, Area, delayInflow=1,
               downstream=NA,
               transformMethod="SCS",
               lossMethod="none",
               BFSSMethod='none',
               UH=NA,
               abstractionParams=list(canopyAbstraction=NULL, surfaceAbstraction=NULL),
               transformParams=list(Tlag=NULL, Cp=NULL, Ct=NULL, L=NULL, Lc=NULL),
               lossParams=list(CN=NULL, f0=NULL, f1=NULL, k=NULL),
               BFSSParams=list(alpha=NULL, BFI=NULL, k=NULL))
```

## Arguments

name	(optional): a string: the name of sub-basin to be instantiated
precipitation	a vector : a time series of precipitation hytograph (mm)
inflow	(optional): a vector of direct/lateral inflow (cms)
Area	the area of basin (Km <sup>2</sup> )
delayInflow	(optional): an integer presenting the time steps to delay direct/lateral inflow time series
downstream	(optional): an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach.
transformMethod	a string: the type of transformation method. Available types: "SCS", "snyder", and "user" for user defined unit hydrograph. default to "SCS"
lossMethod	a string: the type of loss method. Available types: "SCS" and "horton"
BFSSMethod	a string: The method of base flow separation. Available methods: 'nathan', 'chapman', 'eckhardt', 'recession'
UH	a data.frame: including the ordinates of user UH. the HU first collumn indicates time (Hr) and second collumn include flow rates (cms)
abstractionParams	a list: including parameters of simple surface and simple canopy methods. <ul style="list-style-type: none"> <li>canopyAbstaction depth of canopy abstraction in (mm)</li> </ul>

BFSParams	<ul style="list-style-type: none"> <li>• surfaceAbstraction depth of surface abstraction in (mm)</li> </ul> a list including parameters associated with the method coerced in 'BFSTMethod'. <ul style="list-style-type: none"> <li>• alpha is in [0, 1] interval required for 'nathan', 'chapman', and 'eckhardt' methods</li> <li>• BFI is in [0, 1] interval required for 'eckhardt' method</li> <li>• k is in [0, 1] interval and timeInterval is in day required for 'recession' method</li> </ul>
transformParams	a list of parameters associated to the selected type of transformMethod: <ul style="list-style-type: none"> <li>• Tlag for "SCS" method in (Hours)</li> <li>• Ct, Cp, L, and Lc for "snyder" method</li> </ul>
lossParams	a list of parameters associated to the selected type of lossMethod: <ul style="list-style-type: none"> <li>• CN for "SCS" method</li> <li>• f0, f1, k other for "horton" method</li> </ul>

**Value**

a list: an object from class of createSubbasin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

loss

*Excess rainfall computation*

---

**Description**

this function provides methods (e.g. "horton" and "SCS") to compute loss and direct runoff depths

**Usage**

```
loss(precipitation,lossMethod,lossParams)
```

**Arguments**

precipitation	a vector of precipitation time series(mm)
lossMethod	a string including the type of lossMethod: "SCS" and "horton". default to "SCS" method
lossParams	a list of parameters associated to the selected type of lossMethod: <ul style="list-style-type: none"> <li>• the curve number, CN, and imperviousness in percentage for "SCS" method</li> <li>• f0, f1, k for "horton" method</li> <li>• timeInterval: the interval of each steps in seconds needed for "horton" method</li> </ul>

**Value**

a dataframe: including precipitation, loss, and excess rainfall depth

**Author(s)**

Rezgar Arabzadeh

**See Also**

[transform](#)

**Examples**

```
precipitation<-sin(seq(0.1,pi-0.1,length.out=20))*30
lossParams<-list(f0=20,f1=5,k=2,timeInterval=3600,CN=65)
lossMethod<-c("horton","SCS")
(Horton_loss<-loss(precipitation,lossMethod[1],lossParams))
(SCS_loss<-loss(precipitation,lossMethod[2],lossParams))
```

---

loss.base

*base function for class of reachRouting*

---

**Description**

this function provides methods (e.g. "horton" and "SCS") to compute loss and direct runoff depths

**Usage**

```
## S3 method for class 'base'
loss(precipitation,lossMethod,lossParams)
```

**Arguments**

precipitation	a vector of precipitation time series(mm)
lossMethod	a string including the type of lossMethod: "SCS" and "horton". default to "SCS" method
lossParams	a list of parameters associated to the selcted type of lossMethod: <ul style="list-style-type: none"> <li>• the curve number, CN, and imperviousness in precentage for "SCS" method</li> <li>• f0, f1, k for "horton" method</li> <li>• timeInterval: the interval of each steps in seconds needed for "horton" method</li> </ul>

**Value**

a dataframe: including precipitation, loss, and excess rainfall depth

**Author(s)**

Rezgar Arabzadeh

**See Also**[loss](#)


---

loss.default	<i>default function for class of loss</i>
--------------	---

---

**Description**

this function provides methods (e.g. "horton" and "SCS") to compute loss and direct runoff depths

**Usage**

```
## Default S3 method:
loss(precipitation, lossMethod,
     lossParams=list(f0=NULL,
                    f1=NULL,
                    k=NULL,
                    timeInterval=NULL,
                    CN=NULL,
                    imperviousness=NULL))
```

**Arguments**

**precipitation** a vector of precipitation time series(mm)

**lossMethod** a string including the type of lossMethod: "SCS" and "horton". default to "SCS" method

**lossParams** a list of parameters associated to the selected type of lossMethod:

- the curve number, CN, and imperviousness in percentage for "SCS" method
- f0, f1, k for "horton" method
- timeInterval: the interval of each steps in seconds needed for "horton" method

**Value**

a dataframe: including precipitation, loss, and excess rainfall depth

**Author(s)**

Rezgar Arabzadeh

**See Also**[loss](#)

---

plot.createBasin	<i>plots basin layout</i>
------------------	---------------------------

---

**Description**

plot method for objects inherited from class of createBasin

**Usage**

```
## S3 method for class 'createBasin'  
plot(x,...)
```

**Arguments**

x	an object from class of createBasin
...	other objects that can be passed to plot function

**Author(s)**

Rezgar Arabzadeh

**See Also**

[sim](#)

---

plot.sim	<i>plot method for an RHMS object</i>
----------	---------------------------------------

---

**Description**

plot method for objects inherited from class of sim

**Usage**

```
## S3 method for class 'sim'  
plot(x,...)
```

**Arguments**

x	an object from class of sim
...	other objects that can be passed to plot function

**Author(s)**

Rezgar Arabzadeh

**See Also**[sim](#)


---

reachRouting	<i>channel routing computation</i>
--------------	------------------------------------

---

**Description**

function for flood routing using parameteric Muskingum and muskingum-cunge techniques.

**Usage**

```
reachRouting(inflow, routingMethod,
             routingParams, simulation)
```

**Arguments**

inflow	a vector of runoff (cms) presenting a runoff event generated by excess rainfall computed by loss methods or an object inherited from any of the following classes :transform ; reachRouting ; reservoirRouting.
routingMethod	a string: the type of channel routing method: "muskingum" or "muskingumcunge". default to "muskingum"
routingParams	a list : parameters associated to the routingMethod: <ul style="list-style-type: none"> <li>• k and x for "muskingum",</li> <li>• bedWith (m), sideSlope (m/m), channelSlope (m/m), manningRoughness, riverLength (Km) for "muskingumcunge"</li> </ul>
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• end: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

a data.frame: including inflow time series routing results and simulation details

**Author(s)**

Rezgar Arabzadeh

**References**

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.

**See Also**[reservoirRouting](#)



**Examples**

```

inflow<-c(100,500,1500,2500,5000,11000,22000,28000,28500,26000,
          22000,17500,14000,10000,7000,4500,2500,1500,1000,500,100)
routingMethod<-c("muskingum","muskingumcunge")
routingParams<-list(k=3,x=0.2,bedWith=50,sideSlope=2,channelSlope=0.0001,
                   manningRoughness=0.01,riverLength=100)
simulation<-list(start='2000-01-01',end='2000-01-04',by=3600)

reachRouting(inflow,routingMethod[1],routingParams,simulation)
reachRouting(inflow,routingMethod[2],routingParams,simulation)

```

---

reachRouting.base      *base function for class of reachRouting*

---

**Description**

function for flood routing using Muskingum and muskingum-cunge techniques.

**Usage**

```

## S3 method for class 'base'
reachRouting(inflow,routingMethod,
             routingParams,simulation)

```

**Arguments**

inflow	a vector of runoff (cms) or an object inherited from any of the following classes :transform ; reachRouting ; reservoirRouting.
routingMethod	a string: the type of channel routing method: "muskingum" or "muskingumcunge". default to "muskingum"
routingParams	a list : parameters associated to the routingMethod: <ul style="list-style-type: none"> <li>• k and x for "muskingum",</li> <li>• bedWith (m), sideSlope (m/m), channelSlope (m/m), manningRoughness, riverLength (Km) for "muskingumcunge"</li> </ul>
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• start: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

a data.frame: including inflow time series routing resaults and simulation details

**Author(s)**

Rezgar Arabzadeh

**References**

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.

**See Also**

[reachRouting](#)

---

reachRouting.default *default function for class of reachRouting*

---

**Description**

function for flood routing in channels using Muskingum and muskingum-cunge techniques.

**Usage**

```
## Default S3 method:
reachRouting(inflow, routingMethod="muskingum",
             routingParams=list(k=3,
                               x=0.2,
                               bedWith=NULL,
                               sideSlope=2,
                               channelSlope=NULL,
                               manningRoughness=0.025,
                               riverLength=NULL),
             simulation=list(start=NULL, end=NULL, by=NULL))
```

**Arguments**

inflow	a vector of runoff (cms) or an object inherited from any of the following classes :transform ; reachRouting ; reservoirRouting.
routingMethod	a string: the type of channel routing method: "muskingum" or "muskingumcunge". default to "muskingum"
routingParams	a list : parameters associated to the routingMethod: <ul style="list-style-type: none"> <li>• k and x for "muskingum",</li> <li>• bedWith (m), sideSlope (m/m), channelSlope (m/m), manningRoughness, riverLength (Km) for "muskingumcunge"</li> </ul>
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• start: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

a list: including inflow time series routing results and simulation details

**Author(s)**

Rezgar Arabzadeh

**References**

Chow, V. T., Maidment, D. R., &amp; Mays, L. W. (1988). Applied hydrology.

**See Also**[reachRouting](#)


---

 reservoirRouting      *reservoir routing*


---

**Description**

function for routing flood through a reservoir using classical Muskingum technique

**Usage**

reservoirRouting(inflow,geometry,initialStorage,simulation)

**Arguments**

- |                |  |
|----------------|--|
| inflow         | a vector of in (cms) presenting a runoff event generated by excess rainfall computed by loss methods or an object inherited from any of the following classes :transform ; reachRouting ; reservoirRouting.  |
| geometry       | a list of geometric specifications of the reservoir: <ul style="list-style-type: none"> <li>• storageElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent volume to the height at first column (MCM)</li> <li>• dischargeElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent discharge rate to the height at first column (cms)</li> <li>• storage: the maximum volume of reservoir capacity (MCM)</li> </ul> |
| initialStorage | (optional) the initial storage of reservoir at the first time step of simulation (MCM). default to the capacity.   |
| simulation     | a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• end: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>  |

**Value**

a data.frame: including inflow time series and routing results

**Author(s)**

Rezgar Arabzadeh

**References**

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.

**See Also**

[reachRouting](#)

**Examples**

```
inflow<-sin(seq(0,pi,length.out=50))*1000
storageElevationCurve<-data.frame(s=0:49*2,h=100:149)
dischargeElevationCurve<-data.frame(q=0:9*250,h=140:149)
geometry<-list(storageElevationCurve=storageElevationCurve,
               dischargeElevationCurve=dischargeElevationCurve,
               capacity=80)
simulation<-list(start='2000-01-01',end='2000-01-05',by=1800)
reservoir_sim<-reservoirRouting(inflow=inflow,
                                geometry=geometry,
                                simulation=simulation)
plot(reservoir_sim$operation[,1],typ="o",
     ylab="Discharge rate (cms)",
     xlab="Time step")
lines(reservoir_sim$operation[,3],col=2)
```

---

reservoirRouting.base *base function for class of reservoirRouting*

---

**Description**

function for routing flood through a reservoir using classical Muskingum technique

**Usage**

```
## S3 method for class 'base'
reservoirRouting(inflow, geometry,initialStorage,simulation)
```

**Arguments**

inflow	a vector of in (cms) presenting a runoff event generated by excess rainfall computed by loss methods or an object inherited from any of the following classes :transform ; reachRouting ; reservoirRouting.
geometry	a list of geometric specifications of the reservoir:

- storageElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent volume to the height at first column (MCM)
  - dischargeElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second column presents equivalent discharge rate to the height at first column (cms)
  - storage: the maximum volume of reservoir capacity (MCM)
- initialStorage (optional) the initial storage of reservoir at the first time step of simulation (MCM). default to the capacity.
- simulation a list of simulation time and dates as below:
- start: the date which simulation starts, must be in 'YYYY-MM-DD' format
  - end: the date which simulation ends, must be in 'YYYY-MM-DD' format
  - by: the interval of each steps in seconds

**Value**

a data.frame: including inflow time series and routing results

**Author(s)**

Rezgar Arabzadeh

**References**

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.

**See Also**

[reservoirRouting](#)

---

reservoirRouting.default

*default function for class of reservoirRouting*

---

**Description**

function for routing flood through a reservoir using classical Muskingum technique

**Usage**

```
## Default S3 method:
reservoirRouting(inflow,
                 geometry=list(storageElevationCurve=NULL,
                               dischargeElevationCurve=NULL,
                               capacity=NULL),
                 initialStorage=NA,
                 simulation=list(start=NULL, end=NULL, by=NULL))
```

**Arguments**

inflow	a vector of in (cms) presenting a runoff event generated by excess rainfall computed by loss methods or an object inherited from any of the following classes :transform ; reachRouting ; reservoirRouting.
geometry	a list of geometric specifications of the reservoir: <ul style="list-style-type: none"> <li>• storageElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second collums presents equivalent volume to the height at first collumn (MCM)</li> <li>• dischargeElevationCurve: a data frame: a data frame at which its first column includes height (masl) and second collums presents equivalent discharge rate to the height at first collumn (cms)</li> <li>• storage: the maximum volume of reservoir capacity (MCM)</li> </ul>
initialStorage	(optional) the initial storage of reservoir at the first time step of simulation (MCM). default to the capacity.
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• end: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

a data.frame: including inflow time series and routing results

**Author(s)**

Rezgar Arabzadeh

**References**

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.

**See Also**

[reservoirRouting](#)

---

set.as

*RHMS objects connector*

---

**Description**

this function connects a base object as a either of: 'downstream' or 'divertTo' to a target object, which are both instantiated by RHMS constructors.

**Usage**

```
set.as(base, target, type='downstream')
```

**Arguments**

base	An object; from either of classes of <a href="#">createReservoir</a> , <a href="#">createJunction</a> , <a href="#">createDiversion</a> , <a href="#">createSubbasin</a> , or <a href="#">createReach</a>
target	An object; from either of classes of <a href="#">createReservoir</a> , <a href="#">createJunction</a> , <a href="#">createDiversion</a> , <a href="#">createSubbasin</a> , or <a href="#">createReach</a>
type	the type of base object to be set as to the target object: 'downstream', or 'divertTo'

**Value**

an object from class of target object.

**Author(s)**

Rezgar Arabzadeh

**See Also**

[addObjectToBasin](#)

---

 sim

---

*RHMS simulation function*


---

**Description**

simulates an object inherited form class of createBasin

**Usage**

sim(object)

**Arguments**

object            an object from class of createBasin

**Value**

a list: the same as objects inherited from class of createBasin

**Author(s)**

Rezgar Arabzadeh

**References**

- NRCS, U. (1986). Urban hydrology for small watersheds-Technical Release 55 (TR55). Water Resources Learning Center. Washington DC.
- Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.





```
end = '2000-01-15',  
by = 3600))  
  
ZaabRB<-addObjectToBasin(R1,ZaabRB)  
ZaabRB<-addObjectToBasin(R2,ZaabRB)  
ZaabRB<-addObjectToBasin(R3,ZaabRB)  
ZaabRB<-addObjectToBasin(R4,ZaabRB)  
ZaabRB<-addObjectToBasin(R5,ZaabRB)  
ZaabRB<-addObjectToBasin(R6,ZaabRB)  
ZaabRB<-addObjectToBasin(R7,ZaabRB)  
ZaabRB<-addObjectToBasin(R8,ZaabRB)  
ZaabRB<-addObjectToBasin(J1,ZaabRB)  
ZaabRB<-addObjectToBasin(J2,ZaabRB)  
ZaabRB<-addObjectToBasin(D1,ZaabRB)  
ZaabRB<-addObjectToBasin(SilvehDam,ZaabRB)  
ZaabRB<-addObjectToBasin(GordebinDam,ZaabRB)  
ZaabRB<-addObjectToBasin(KanisibDam,ZaabRB)  
ZaabRB<-addObjectToBasin(Pardanan,ZaabRB)  
ZaabRB<-addObjectToBasin(Zangabad,ZaabRB)  
ZaabRB<-addObjectToBasin(Darbekaykhaneh,ZaabRB)  
  
## Not run:  
plot(ZaabRB)  
  
plot(sim(ZaabRB))  
  
## End(Not run)
```

---

sim.base

*base function for class of sim*

---

## Description

simulates an object inherited from class of createBasin

## Usage

```
## S3 method for class 'base'  
sim(object)
```

## Arguments

object            an object from class of createBasin

## Author(s)

Rezgar Arabzadeh

## See Also

[sim](#)

---

sim.default	<i>default function for class of sim</i>
-------------	--

---

**Description**

simulates an object inherited form class of createBasin

**Usage**

```
## Default S3 method:
sim(object)
```

**Arguments**

object            an object from class of createBasin

**Author(s)**

Rezgar Arabzadeh

**See Also**

[sim](#)

---

summary.sim	<i>summary method for RHMS objects</i>
-------------	--

---

**Description**

summary method for objects inherited from class of sim

**Usage**

```
## S3 method for class 'sim'
summary(object,...)
```

**Arguments**

object            an object from class of sim  
 ...               other objects that can be passed to summary function

**Value**

a matrix: including inflow and outflow volumes and peaks rates respectively

**Author(s)**

Rezgar Arabzadeh

**See Also**[sim](#)


---

transform	<i>Transforms a rainfall event to runoff</i>
-----------	--

---

**Description**

This function transforms an excess rainfall event to a direct runoff hydrograph.

**Usage**

```
transform(rainfall, transformMethod, transformParams, Area, UH, simulation)
```

**Arguments**

rainfall	an object inherited from loss function
transformMethod	a string: the type of transformation method. available types: "SCS", "snyder", and "user". default to "SCS"
transformParams	a list of parameters associated to the selected type of transformMethod: <ul style="list-style-type: none"> <li>• Tlag for "SCS" method</li> <li>• Ct, Cp, L, and Lc for "snyder" method</li> </ul>
Area	the area of drainage basin (Km <sup>2</sup> )
UH	a data.frame: must be provided when transformMethod is set to "user". UH is the ordinates of a user defined UH by the which its first column is time (Hr) and the second column includes flow rates (cms)
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• end: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

Hydrograph of direct runoff

**Author(s)**

Rezgar Arabzadeh

**See Also**[sim](#)**Examples**

```

Area=200
lossMethod<-"SCS"
lossParams<-list(CN=65)
transformMethod<-c("snyder","SCS","user")
simulation<-list(start='2000-01-01',end='2000-01-7',by=7200)
precipitation<-sin(seq(0.1,pi-0.1,length.out=10))*20
transformParams=list(Tlag=4,Cp=0.15,Ct=2,L=100,Lc=15)
UH<-data.frame(t=1:20,q=sin(seq(0,pi,length.out=20))*1)

SCS_loss<-loss(precipitation,lossMethod,lossParams)

snyder_transformation<-transform(rainfall=SCS_loss,
                                transformMethod=transformMethod[1],
                                transformParams,Area,UH=NA,simulation)
SCS_transformation  <-transform(rainfall=SCS_loss,
                                transformMethod=transformMethod[2],
                                transformParams,Area,UH=NA,simulation)
user_transformation <-transform(rainfall=SCS_loss,
                                transformMethod=transformMethod[3],
                                transformParams,Area,UH,simulation)

```

---

transform.base	<i>base function for class of transform</i>
----------------	---

---

**Description**

This function transforms an excess rainfall event to a direct runoff hydrograph.

**Usage**

```

## S3 method for class 'base'
transform(rainfall,transformMethod,transformParams,Area,UH,simulation)

```

**Arguments**

rainfall	an object inherited from loss function
transformMethod	a string: the type of transformation method. available types: "SCS", "snyder", and "user". default to "SCS"
transformParams	a list of parameters associated to the selected type of transformMethod: <ul style="list-style-type: none"> <li>• Tlag for "SCS" method</li> <li>• Ct, Cp, L, and Lc for "snyder" method</li> </ul>

Area	the area of drainage basin (Km <sup>2</sup> )
UH	a data.frame: must be provided when transformMethod is set to "user". UH is the ordinates of a user defined UH by the which its first column is time (Hr) and the second column includes flow rates (cms)
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• end: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

Hydrograph of direct runoff

**Author(s)**

Rezgar Arabzadeh

**See Also**

[transform](#)

---

transform.default      *default function for class of transform*

---

**Description**

This function transforms an excess rainfall event to a direct runoff hydrograph.

**Usage**

```
## Default S3 method:
transform(rainfall, transformMethod='SCS',
         transformParams=list(Tlag=NULL,
                              Cp =NULL,
                              Ct =NULL,
                              L  =NULL,
                              Lc =NULL),
         Area,UH=NA,
         simulation=list(start=NULL,end=NULL,by=NULL))
```

**Arguments**

rainfall	an object inherited from loss function
transformMethod	a string: the type of transformation method. available types: "SCS", "snyder", and "user". default to "SCS"
transformParams	a list of parameters associated to the selected type of transformMethod: <ul style="list-style-type: none"> <li>• Tlag for "SCS" method</li> <li>• Ct, Cp, L, and Lc for "snyder" method</li> </ul>
Area	the area of drainage basin (Km <sup>2</sup> )
UH	a data.frame: must be provided when transformMethod is set to "user". UH is the ordinates of a user defined UH by the which its first column is time (Hr) and the second column includes flow rates (cms)
simulation	a list of simulation time and dates as below: <ul style="list-style-type: none"> <li>• start: the date which simulation starts, must be in 'YYYY-MM-DD' format</li> <li>• end: the date which simulation ends, must be in 'YYYY-MM-DD' format</li> <li>• by: the interval of each steps in seconds</li> </ul>

**Value**

Hydrograph of direct runoff

**Author(s)**

Rezgar Arabzadeh

**See Also**

[transform](#)

---

tune

*tuning an RHMS model*

---

**Description**

a function for tuning an RHMS model based on a set of observed time series, using *particle swarm optimization*

**Usage**

```
tune(object, targetObject, decisionObjects,
      observationTS, delay=0,
      transformBandWith=list(ct=c(1 , 2.5),
                             cp=c(0.1, 0.3),
                             cn=c(25 , 85 ),
                             k =c(0.1, 2 )),
      routingBandWith=list(manning = c(0.0001, 0.1),
                           x       = c(0.2 , 0.6),
                           k       = c(1 , 5 )),
      maxiter=NA, update=FALSE, plot=FALSE)
```

**Arguments**

object	an object from class of createBasin
targetObject	an object from either of classes: createDiversion, createReservoir, createSubbasin, createJunction, createReach associated to the observationTS
decisionObjects	A list of objects, also, already existing in the object which their parameters needed to be optimized. They objects must be from either of classes: createSubbasin, createReach
observationTS	a vector: an observed flow time series (cms)
delay	(optional) an integer presenting the number of time steps to delay observationTS time series
transformBandWith	an list: a list of vector(s), including upper and lower limit of parameters of transformation methods. Each parameter search domain is set as a two-value vector, whose first element indicates lower limit and second elemnt is upper limit. <ul style="list-style-type: none"> <li>• Ct=[1, 2.5] and Cp=[0.1, 0.3] are parameters for "Snyder" Unit Hydrograph (SUH)</li> <li>• cn=[25, 85] curve number for "SCS" loss method</li> <li>• k for "horton" loss method</li> </ul>
routingBandWith	an list: a list of vector(s), including upper and lower limit of parameters of routing methods. Each parameter search domain is set as a two-value vector, whose first element indicates lower limit and second elemnt is upper limit. <ul style="list-style-type: none"> <li>• manning=[0.0001, 0.1] is a parameter used "muskingumcunge" method</li> <li>• x = [0.2, 0.6] and k=[1, 5] belong to "muskingum" channel routing method</li> </ul>
maxiter	(optional) an integer: maximum number of iterations. default to the square of dimension of decision variables
plot	(optional) logical: plots the optimization results
update	(optional) logical: If FALSE, the optimized parameter(s) are returned, If TRUE, the calibrated object from class of createBasin is returned

**Value**

a vector of tuned parameters or an object from class of createBasin

**Author(s)**

Rezgar Arabzadeh

**References**

Kennedy, J. (1997). "The particle swarm: social adaptation of knowledge". Proceedings of IEEE International Conference on Evolutionary Computation. pp. 303-308

**Examples**

```
J1<-createJunction (name="J1")
R1<-createReach(name="R1",routingMethod="muskingum",
                routingParams=list(k=3,x=0.2),
                downstream=J1)
R2<-createReach(name="R2",routingMethod="muskingumcunge",
                routingParams=list(bedWith=50,
                                sideSlope=2,
                                channelSlope=0.0005,
                                manningRoughness=0.025,
                                riverLength=100),
                downstream=J1)
S1<-createSubbasin(name = "S1",
                  precipitation=sin(seq(0,pi,length.out=20))*40,
                  Area=100,downstream=R1,
                  transformMethod="SCS",lossMethod="SCS",
                  transformParams=list(Tlag=4),lossParams=list(CN=60))
S2<-createSubbasin(name = "S2",
                  precipitation=sin(seq(0,pi,length.out=20))*30,
                  Area=300,downstream=R2,
                  transformMethod="snyder",lossMethod="horton",
                  transformParams=list(Cp=0.17,Ct=2,L=30,Lc=15),
                  lossParams=list(f0=10,f1=4,k=1))

basin1<-createBasin(name = "Ghezil_Ozan",
                   simulation=list(start='2000-01-01',
                                end  ='2000-01-05',
                                by   =3600))

basin1<-addObjectToBasin(S1, basin1)
basin1<-addObjectToBasin(S2, basin1)
basin1<-addObjectToBasin(R1, basin1)
basin1<-addObjectToBasin(R2, basin1)
basin1<-addObjectToBasin(J1, basin1)

## Not run: plot(basin1)

simulated<-sim(basin1)
plot(simulated)
observationTS1<-simulated$operation$junctions[[1]]$outflo[,1]
```



```

set.seed(1)
observationTS1<-observationTS1+rnorm(length(observationTS1),0,25)
y<-observationTS1; x<-1:length(observationTS1)
observationTS1<-predict(loess(y~x),x)
observationTS1[which(observationTS1<0)]<-0
observationTS<-observationTS1
plot(simulated$operation$junctions[[1]]$outflow[,1],typ='o',ylab='flow rate (cms)',xlab='time step')
lines(observationTS,col=2)

transformBandWith=list(ct=c(1,2.5),
                      cp=c(0.1,0.3),
                      cn=c(25,85),
                      k=c(0.1,2))
routingBandWith=list(manning=c(0.0001,0.1),
                    x=c(0.2,0.6),
                    k=c(1,5))
targetObject<-J1
decisionObjects<-list(R1,R2,S1,S2)
## Not run:
tune(object=basin1,
     targetObject=targetObject,
     decisionObjects=decisionObjects,
     observationTS=observationTS,
     routingBandWith=routingBandWith,
     transformBandWith=transformBandWith,
     plot=TRUE)

## End(Not run)

```

---

Zaab

*datasets for Zaab subbasin, a subbasin in Kurdistan, Iran.*


---

### Description

an object inherited from class of createBasin. including features, of a sub-basin in Kurditan known as Zaab, such as: reservoirs, reaches, subbasins, and junctions.

### Usage

```
data(Zaab)
```

### Source

Iran Water Resources Management Company (2015)

### Examples

```
data(Zaab)
```

# Index

- \* **array**
  - baseFlowSeparation, 8
  - baseFlowSeparation.base, 9
  - summary.sim, 42
  - transform, 43
  - transform.base, 44
  - transform.default, 45
- \* **datasets**
  - Zaab, 49
- \* **graphs**
  - plot.createBasin, 31
  - plot.sim, 31
- \* **iteration**
  - addObjectToBasin, 6
- \* **list**
  - abstraction, 4
  - abstraction.base, 5
  - abstraction.default, 5
  - baseFlowSeparation.default, 10
  - createBasin, 11
  - createBasin.base, 12
  - createBasin.default, 12
  - createDiversion, 13
  - createDiversion.base, 14
  - createDiversion.default, 15
  - createJunction, 15
  - createJunction.base, 16
  - createJunction.default, 17
  - createReach, 18
  - createReach.base, 19
  - createReach.default, 20
  - createReservoir, 21
  - createReservoir.base, 22
  - createReservoir.default, 23
  - createSubbasin, 24
  - createSubbasin.base, 25
  - createSubbasin.default, 27
  - reachRouting, 32
  - reachRouting.base, 33
  - reachRouting.default, 34
  - reservoirRouting, 35
  - reservoirRouting.base, 36
  - reservoirRouting.default, 37
  - set.as, 38
  - sim, 39
  - sim.base, 41
  - sim.default, 42
- \* **matrix**
  - loss, 28
  - loss.base, 29
  - loss.default, 30
- \* **optimize**
  - tune, 46
- \* **package**
  - RHMS-package, 2
- abstraction, 3, 4
- abstraction.base, 5
- abstraction.default, 5
- addObjectToBasin, 3, 6, 11–22, 24–26, 28, 39
- baseFlowSeparation, 3, 8, 9, 10
- baseFlowSeparation.base, 9
- baseFlowSeparation.default, 10
- createBasin, 3, 11
- createBasin.base, 12
- createBasin.default, 12
- createDiversion, 3, 13, 39
- createDiversion.base, 14
- createDiversion.default, 15
- createJunction, 3, 6, 15, 39
- createJunction.base, 16
- createJunction.default, 17
- createReach, 3, 6, 18, 39
- createReach.base, 19
- createReach.default, 20
- createReservoir, 3, 6, 21, 39
- createReservoir.base, 22

createReservoir.default, 23  
createSubbasin, 3–6, 11, 24, 39  
createSubbasin.base, 25  
createSubbasin.default, 27

loss, 3, 28, 30  
loss.base, 29  
loss.default, 30

plot.createBasin, 3, 31  
plot.sim, 3, 31

reachRouting, 3, 32, 34–36  
reachRouting.base, 33  
reachRouting.default, 34  
reservoirRouting, 3, 32, 35, 37, 38  
reservoirRouting.base, 36  
reservoirRouting.default, 37  
RHMS (RHMS-package), 2  
RHMS-package, 2

set.as, 3, 38  
sim, 3, 4, 6, 31, 32, 39, 41–44  
sim.base, 41  
sim.default, 42  
summary.sim, 3, 42

transform, 3, 29, 43, 45, 46  
transform.base, 44  
transform.default, 45  
tune, 3, 46

Zaab, 49