# Package: rivernet (via r-universe)

August 22, 2024

**Type** Package

**Title** Read, Analyze and Plot River Networks

**Version** 1.2.3

**Date** 2023-08-28

**Author** Peter Reichert

**Maintainer** Peter Reichert <peter.reichert@emeriti.eawag.ch>

**Description** Functions for reading, analysing and plotting river networks. For this package, river networks consist of sections and nodes with associated attributes, e.g. to characterise their morphological, chemical and biological state. The package provides functions to read this data from text files, to analyse the network structure and network paths and regions consisting of sections and nodes that fulfill prescribed criteria, and to plot the river network and associated properties.

**License** GPL-3

**NeedsCompilation** no

**Date/Publication** 2023-08-28 11:00:03 UTC

**Repository** https://peterreichert.r-universe.dev

**RemoteUrl** https://github.com/cran/rivernet

**RemoteRef** HEAD

**RemoteSha** 58874e187dbb7e86e32fb0d2a6d0d42dc2b11bf1

## Contents

## `rivernet-package` *Read, Analyse and Plot River Networks*

### Description

Functions for reading, analysing and plotting river networks.

For this package, river networks consist of sections and nodes with associated attributes, e.g. to characterise their morphological, chemical and biological state. The package provides functions to read this data from text files, to analyse the network structure and network paths and regions consisting of sections and nodes that fulfill prescribed criteria, and to plot the river network and associated properties.

Important functions:

`rivernet.read`: Read river network and attribute data from text files.
`rivernet.write`: Write river network and attribute data to text files.
`plot.rivernet`: Plot a river network and visualize properties.
`analyze.rivernet`: Analyse the network structure and calculate network properties.
`prune.rivernet`: Prune a river network at given points.
`getreachind.rivernet`: Find the reach that is closest to a given point.
`getnodeind.rivernet`: Find the node that is closest to a given point.
`mean.rivernet`: Calculate the length and stream order weighted mean of a property.
`upstreamconnectivity.rivernet`: Find flow paths that fulfill given properties, e.g. that can be reached by fish from downstream.
`adjacentreaches.rivernet`: Find regions of river sections that fulfill given properties and are (nearly) adjacent to each other. `getregionconnections`: Find connecting paths between regions that may have been identified with the function `adjacentreaches.rivernet`.

## Details

| | |
|---|---|
| Package: | rivernet |
| Type: | Package |
| Version: | 1.2.3 |
| Date: | 2023-08-28 |
| License: | GPL-3 |

## Author(s)

Peter Reichert

Maintainer: Peter Reichert <peter.reichert@emeriti.eawag.ch>

## Examples

```
coord <- data.frame(Reach_ID=c(1,1,2,2,2,2,2,3,3,4,4),
                    X=c(5,5,5,7,8,9,10,5,0,0,2),
                    Y=c(0,2,2,4,7,6, 8,2,6,6,7),
                    Z=c(0,1,1,2,3,4, 5,1,2,2,3))
attrib.reach <- data.frame(Reach_ID=c(1,2,3,4),
                           State   =c(0,0.2,0.8,0.8),
                           Flow    =c(4,2,2,2))
attrib.node  <- data.frame(X=c(5,5,0,10,2),
                           Y=c(0,2,6, 8,7),
                           Height=c(0,0,1,0,0))
write.table(coord     ,"rivernet_example_coord.csv",sep=";",col.names=TRUE,row.names=FALSE)
write.table(attrib.reach,"rivernet_example_reach.csv",sep=";",col.names=TRUE,row.names=FALSE)
write.table(attrib.node ,"rivernet_example_node.csv" ,sep=";",col.names=TRUE,row.names=FALSE)
net <- rivernet.read("rivernet_example_coord.csv",
                     "rivernet_example_reach.csv",
                     "rivernet_example_node.csv",
                     sep=";")
plot(net,col=ifelse(net$attrib.reach$State<0.5,"red","blue"),lwd=2,pch=19,cex.nodes=1.5,
     col.nodes=ifelse(is.na(net$attrib.node$Height),
                      "black",
                      ifelse(net$attrib.node$Height<0.1,"blue","red")))
net <- splitreach(net,2,0.4)
plot(net,col=ifelse(net$attrib.reach$State<0.5,"red","blue"),lwd=2,pch=19,cex.nodes=1.5,
     col.nodes=ifelse(is.na(net$attrib.node$Height),
                      "black",
                      ifelse(net$attrib.node$Height<0.1,"blue","red")))
file.remove("rivernet_example_coord.csv")
file.remove("rivernet_example_reach.csv")
file.remove("rivernet_example_node.csv")
```

## adjacentreaches *Finds regions of adjacent reaches*

### Description

Finds regions of adjacent reaches.

### Usage

```
adjacentreaches(x,...)
```

### Arguments

| | |
|---|---|
| `x` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `...` | . |

### Value

Numerical vector of the same length as the number of reaches in the river network with unique numbering corresponding to regions of adjacent reaches (under the condition specified by `crit.reach` and `thresh.length`). The segment labelled 0 corresponds to reaches that do not fulfill the criteria. The other regions are labelled with 1, 2, 3, ...

### See Also

`rivernet.read`, `mean.rivernet`, `upstreamconnectivity.rivernet`, `utility`.

## adjacentreaches.rivernet *Finds Regions of Adjacent Reaches*

### Description

Finds regions of adjacent reaches that fulfill given criteria.

### Usage

```
## S3 method for class 'rivernet'
adjacentreaches(x,crit.reach,crit.node=TRUE,thresh.length=0,...)
```

## Arguments

| | |
|---|---|
| `x` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `crit.reach` | Logical vector of the same length as the number of reaches. Indicating whether the criterion for reaches is fulfilled. A single value of `TRUE` indicates the criterion is fulfilled for all reaches. |
| `crit.node` | Logical vector of the same length as the number of nodes. Indicating whether the criterion for nodes is fulfilled. A single value of `TRUE` indicates the criterion is fulfilled for all nodes. |
| `thresh.length` | Threshold length above which not fulfillment of the criterion separates reaches as not being adjacent. |
| `...` | . |

## Value

Numerical vector of the same length as the number of reaches in the river network with unique numbering corresponding to regions of adjacent reaches (under the conditions specified by `crit.reach`, `crit.node` and `thresh.length`). The segment labelled 0 corresponds to reaches which do not fulfill the criteria and are further away than `thresh.length` from reaches that fulfill the criteria or which are longer than `thresh.length`. The other regions are labelled with 1, 2, 3, ...

## See Also

`rivernet.read`, `mean.rivernet`, `upstreamconnectivity.rivernet`, `utility`.

---

`analyze` *Analyzes the Network Structure of a River Network*

---

## Description

Analyzes the network structure of a river network stored as an object of type `rivernet`.

## Usage

```
analyze(net,outlet.reach=NA,calc.streamorder=TRUE,verbose=TRUE)
```

## Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `outlet.reach` | Index or, in case of multiple subnets index vector, of outlet reach or outlet reaches. If not provided, the function identifies the outlet reach either by assuming coordinates are provided in downstream direction or from elevation information. |

| | |
|---|---|
| `calc.streamorder` | Logical variable to indicate whether streamorder should be calculated (after pruning top reaches (see `prune.rivernet`) it may be unwanted to calculate stream order by setting (new) top reaches to order one). |
| `verbose` | Logical variable to turn on and off summary output about the network structure read. |

## Value

The function returns an object of type `rivernet` that adds the following columns to the data frame `attrib.reach` of the object: `subnet`: index of sub-network,
`n_start`: number of reaches connected at the start end of the reach,
`n_end`: number of reaches connected at the end end of the reach,
`endreach`: logical variable indicating if the reach is only connected at one end,
`outlet`: logical variable indicating if the reach represents the outlet of the network,
`headwater`: logical variable indicating if the reach is a headwater,
`downstream`: logical variable indicating if the reach coordinates represent the downstream direction,
`reach_down`: index of reach downstream of the reach,
`streamorder`: stream order of the reach.

In addition, the list of reaches (`reaches`) and the list of nodes (`nodes`) are appended by indices `from_node`, `to_node` and `from_reach`, `to_reach`, respectively. Finally, a list of all paths from the headwaters to the outlet is provided;
`paths`: list of vectors of reach indices of the paths from all sources to the outlet.

## See Also

`rivernet.read`, `mean.rivernet`, `adjacentreaches.rivernet`,
`utility`.

---

`analyze.rivernet` *Analyzes the Network Structure of a River Network*

---

## Description

Analyzes the network structure of a river network stored as an object of type `rivernet`.

## Usage

```
## S3 method for class 'rivernet'
analyze(net,outlet.reach=NA,calc.streamorder=TRUE,verbose=TRUE)
```

## Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `outlet.reach` | Index or, in case of multiple subnets index vector, of outlet reach or outlet reaches. If not provided, the function identifies the outlet reach either by assuming coordinates are provided in downstream direction or from elevation information. |
| `calc.streamorder` | Logical variable to indicate whether streamorder should be calculated (after pruning top reaches (see `prune.rivernet`) it may be unwanted to calculate stream order by setting (new) top reaches to order one). |
| `verbose` | Logical variable to turn on and off summary output about the network structure read. |

## Value

The function returns an object of type `rivernet` that adds the following columns to the data frame `attrib.reach` of the object: `subnet`: index of sub-network,
`n_start`: number of reaches connected at the start end of the reach,
`n_end`: number of reaches connected at the end end of the reach,
`endreach`: logical variable indicating if the reach is only connected at one end,
`outlet`: logical variable indicating if the reach represents the outlet of the network,
`headwater`: logical variable indicating if the reach is a headwater,
`downstream`: logical variable indicating if the reach coordinates represent the downstream direction,
`reach_down`: index of reach downstream of the reach,
`streamorder`: stream order of the reach.

In addition, the list of reaches (`reaches`) and the list of nodes (`nodes`) are appended by indices `from_node`, `to_node` and `from_reach`, `to_reach`, respectively. Finally, a list of all paths from the headwaters to the outlet is provided;
`paths`: list of vectors of reach indices of the paths from all sources to the outlet.

## See Also

`rivernet.read`, `rivernet.write`,
`prune.rivernet`,
`mean.rivernet`, `upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`,
`utility`.

---

## `getnodeind` *Gets indices of nodes that are closest to given locations*

---

## Description

Gets indices of nodes that are closest to given locations

### Usage

```
getnodeind(net,x,y,...)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `x` | Numerical vector of x coordinates of sites to be analyzed (needs to be of the same length as argument `y`). |
| `y` | Numerical vector of y coordinates of sites to be analyzed (needs to be of the same length as argument `x`). |
| `...` | . |

### Value

A data frame with the indices in the first and the distances in the second column.

### See Also

`rivernet.read`, `getreachind.rivernet`, `upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`, `utility`.

## `getnodeind.rivernet` *Gets indices of nodes that are closest to given locations*

### Description

Gets indices of nodes that are closest to given locations

### Usage

```
## S3 method for class 'rivernet'
getnodeind(net,x,y,...)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `x` | Numerical vector of x coordinates of sites to be analyzed (needs to be of the same length as argument `y`). |
| `y` | Numerical vector of y coordinates of sites to be analyzed (needs to be of the same length as argument `x`). |
| `...` | . |

### Value

A data frame with the indices in the first and the distances in the second column.

## See Also

`rivernet.read`, `getreachind.rivernet`, `upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`, `utility`.

---

## `getreachind` *Gets indices of reaches that are closest to given locations*

---

### Description

Gets indices of reaches that are closest to given locations

### Usage

```
getreachind(net,x,y,...)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `x` | Numerical vector of x coordinates of sites to be analyzed (needs to be of the same length as argument `y`). |
| `y` | Numerical vector of y coordinates of sites to be analyzed (needs to be of the same length as argument `x`). |
| `...` | . |

### Value

A data frame with the indices in the first and the distances in the second column.

### See Also

`rivernet.read`, `getnodeind.rivernet`, `upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`, `utility`.

---

## `getreachind.rivernet` *Gets indices of reaches that are closest to given locations*

---

### Description

Gets indices of reaches that are closest to given locations

### Usage

```
## S3 method for class 'rivernet'
getreachind(net,x,y,...)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `x` | Numerical vector of x coordinates of sites to be analyzed (needs to be of the same length as argument `y`). |
| `y` | Numerical vector of y coordinates of sites to be analyzed (needs to be of the same length as argument `x`). |
| `...` | . |

### Value

A data frame with the indices in the first and the distances in the second column.

### See Also

`rivernet.read`, `getnodeind.rivernet`, `upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`, `utility`.

---

## `getregionconnections` *Gets indices of reaches that are closest to given locations*

---

### Description

Gets indices of reaches that are closest to given locations

### Usage

```
getregionconnections(net,regions)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `regions` | Numerical vector of the length of the number of regions containing the region coding as calculated e.g. by the function `adjacentreaches.rivernet`: Reaches not belonging to a region should be encoded by the value of zero; reaches belonging to regions should be coded with the index of the region, a natural number from 1 to the number of reaches. |

### Value

A list with an entry for each region. The entry for each region is again a list with the folowing entries (some may be empty):
`downstream.path`: A single vector of indices of reaches describing the path downstream of the given region to the outlet of the river network irrespective of potential other regions to be crossed. Note that the vector will be of length zero if the given region extends to the outlet reach.
`upstream.paths`: A list of numeric vectors of the paths from all upstream headwaters to the given region irrespective of potential other regions to be crossed. Note that the list will be of length zero

if the given region includes all upstream headwaters.
`downstream.region`: If there on the path downstream to the outlet, this will be an empty list. If there is a downstream region, this is a list with the following elements: `region`: index of downstream region; `path`: vector of reach indices of the path from the given region to the downstream region; `dist`: length of the path to the downstream region.
`upstream.regions`: Empty list if there is no upstream region to the given region. Otherwise list of upstream regions with list entries as in `downstream.region` for each upstream regtion.
`downupstream.regions`: List of regions that can be reached from the given region by first moving downstream and then upstream without crossing another region. If there are no such regions, this is an empty list. Otherwise this is a list of such regions and for each of these regions again a list with the following entries: `region`: index of (down-upstream) region; `downstream.path`: vector of reach indices of the downstream path from the given region to the junction from which the region can be reached upstream; `upstream.path`: vector of reach indices of the path from thejunction where the downstream path ends to the region; `dist`: length of the path between the regions.

## See Also

`rivernet.read`, `adjacentreaches.rivernet`,
`utility`.

---

`mean.rivernet` *Calculates the length and stream order - weighted average of a given reach property*

---

## Description

Calculates the length and stream order - weighted average of a given reach property.

## Usage

```
## S3 method for class 'rivernet'
mean(x,y=NA,na.rm=FALSE,...)
```

## Arguments

| | |
|---|---|
| `x` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `y` | Numerical vector of properties of reaches to be averaged. If no values are provided, the function returns the average length of the reaches. |
| `na.rm` | Indication whether NAs should be removed. |
| `...` | . |

## Value

A scalar representing the weighted mean.

## See Also

rivernet.read, upstreamconnectivity.rivernet, adjacentreaches.rivernet, utility.

---

## plot.rivernet *Plots a River Network Stored in an Object of Class "rivernet"*

---

### Description

Plots a river network.

### Usage

```
## S3 method for class 'rivernet'
plot(x,margin=0,
                        main=NA,cex.main=1,pos="topleft",
                        col=NA,lwd=1,
                        pch.nodes=NA,cex.nodes=0.2,col.nodes="black",
                        ...)
```

### Arguments

| | |
|---|---|
| x | Object of class "rivernet" e.g. read by the function rivernet.read. |
| margin | Relative margin size. |
| main | Optional title of the plot. |
| cex.main | Font scaling factor of the title. |
| pos | Position of legend. Either "topleft" or "topright". |
| col | Optional single color or vector of colors of the different reaches either in the same order as the reaches in the rivernet object or labeled by the reach identifiers. |
| lwd | Optional single line width or vector of line widths of the different reaches either in the same order as the reaches in the rivernet object or labeled by the reach identifiers. |
| pch.nodes | Plot marker for nodes. See points for an explanation of codes. |
| cex.nodes | Scaling of markers for nodes. |
| col.nodes | Single color or vector of colors used for node markers. The order must be the same as in the rivernet object. |
| ... | Further arguments are passed to the plotting function. |

### See Also

rivernet.read, utility.

## prune — *Prune a river network at specified reaches*

### Description

Prune a river network at specified reaches.

### Usage

```
prune(net,reach.up=numeric(0),reach.dn=numeric(0),verbose=TRUE)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `reach.up` | Numerical vector of reach indices from which to prune all upstream reaches (the indicated reaches to prune from will be kept in the pruned network). Note that you can get reach indices from coordinates with the function `getreachind.rivernet`. |
| `reach.dn` | Numerical vector of reach indices from which to prune all downstream reaches and branches from downstream reaches (the indicated reaches to prune from will be kept in the pruned network). Note that you can get reach indices from coordinates with the function `getreachind.rivernet`. |
| `verbose` | Logical argument to specify whether there should be (minimal) output over what has been done. |

### Value

Returns pruned object of class "rivernet".

### See Also

`rivernet.read`, `rivernet.write`,
`upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`,
`utility`.

## prune.rivernet — *Prune a river network at specified reaches*

### Description

Prune a river network at specified reaches.

### Usage

```
## S3 method for class 'rivernet'
prune(net,reach.up=numeric(0),reach.dn=numeric(0),verbose=TRUE)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `reach.up` | Numerical vector of reach indices from which to prune all upstream reaches (the indicated reaches to prune from will be kept in the pruned network). Note that you can get reach indices from coordinates with the function `getreachind.rivernet`. |
| `reach.dn` | Numerical vector of reach indices from which to prune all downstream reaches and branches from downstream reaches (the indicated reaches to prune from will be kept in the pruned network). Note that you can get reach indices from coordinates with the function `getreachind.rivernet`. |
| `verbose` | Logical argument to specify whether there should be (minimal) output over what has been done. |

### Value

Returns pruned object of class "rivernet".

### See Also

`rivernet.read`, `rivernet.write`,
`upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`,
`utility`.

---

## `rivernet.read` *Reads Geographical Information of River Network*

---

### Description

Reads a river network and attributes from text files.

### Usage

```
rivernet.read(file.reachcoord,
              file.reachattrib = NA,
              file.nodeattrib  = NA,
              colnames         = c(reach = "Reach_ID",
                                    node  = "Node_ID",
                                    x     = "X",
                                    y     = "Y",
                                    z     = "Z"),
              sep             ="\t",
              tol              = 1,
              analyze          = FALSE,
              verbose          = TRUE,
              ...)
```

## Arguments

| | |
|---|---|
| `file.reachcoord` | Name of text file or vector of names that contain(s) columns with reach id, x, y, and z coordinates. An arbitrary number of rows per reach is possible to allow for a reasonable geographical resolution of the river reach, but the rows corresponding to the same reach id are interpreted as a sequential series of points between which the river is constructed by linear interpolation. If multiple file names are provided, the data frames read from different files are combined with `rbind`. |
| `file.reachattrib` | Name of text file or vector of names that contain(s) a column with the same reach reach ids as used in the file `file.reachcorrd` and an arbitrary number of attributes of the reaches. If multiple file names are provided, the data frames read from different files are combined by `merge` so that either additional attributes of existing reaches (same reach identifier) or attributes for additional reaches (new reach identifiers) can be provided. |
| `file.nodeattrib` | Name of text file or vector of names that contain(s) columns with x and y coordinates of the node (must be identical with start or end coordinats of the connecting reaches) and an arbitrary number of attributes of the nodes. If multiple file names are provided, the data frames read from different files are combined by `merge` so that either additional attributes of existing nodes (same values for x and y coordinates) or attributes for additional nodes (new x and y coordinates) can be provided. |
| `colnames` | Labelled vector containing strings to identify the headers of the colums for reach and node identifiers and x, y and z coordinates along the river reaches. |
| `sep` | Column separator used for the files. |
| `tol` | Spatial tolerance for identifying nodes. |
| `analyze` | Logical variable to indicate whether the function `analyze.rivernet` should be called after reading. |
| `verbose` | Logical variable to turn on and off summary output about the network structure read. |
| `...` | Optional further arguments are passed to read.table. |

## Value

The function returns an object of type "rivernet" that contains the geographical representation of the river. This object contains the following elements

`reaches`: list of river reaches with the following elements:
`n`: number of coordinates,
`x`: vector of x-coordinates,
`y`: vector of y-coordinates,
`z`: vector of z-coordinates,
`length`: length of the reach),

`nodes`: list of river nodes with the following elements:

x: x-coordinate,
y: y-coordinate,

xlim: range of river network in x direction.

ylim: range of river network in y direction.

zlim: vertical range of river network (z direction).

htow: ratio of y to x extension; to be used for the height to widht ratio of network plots.

total.length: sum of length of all river reaches in the network.

attrib.reach: data frame of reaches with columns
Reach_ID: reach identifier,
Reach: reach index,
x_start: start x-coordinate of reach,
y_start: start y-coordinate of reach,
z_start: start elevation of reach,
x_end: end x-coordinate of reach,
y_end: end y-coordinate of reach,
z_end: end elevation of reach,
node_start: index of node at the start of the reach,
node_end: index of node at the end of the reach,
length: length of reach.
If a file file.reachattrib was provided, its columns are added to the colums of this data frame.

attrib.node: data frame of nodes with columns
node: node index,
x: x-coordinate of the node,
y: y-coordinate of the node.

If a file file.nodeattrib was provided, its columns are added to the colums of this data frame.

If the argument analyze is true, the data frame attrib.reach contains the additional columns:
subnet: index of sub-network,
n_start: number of reaches connected at the start end of the reach,
n_end: number of reaches connected at the end end of the reach,
endreach: logical variable indicating if the reach is only connected at one end,
outlet: logical variable indicating if the reach represents the outlet of the network,
headwater: logical variable indicating if the reach is a headwater,
downstream: logical variable indicating if the reach coordinates represent the downstream direction,
reach_down: index of reach downstream of the reach,
streamorder: stream order of the reach.
paths: list of vectors of reach indices of the paths from all headwaters to the outlet.
In addition, the list of reaches (reaches) and the list of nodes (nodes) are appended by indices from_node, to_node and from_reach, to_reach, respectively.

### See Also

analyze.rivernet
plot.rivernet
rivernet.write
prune.rivernet
getreachind.rivernet
getnodeind.rivernet
mean.rivernet
upstreamconnectivity.rivernet
adjacentreaches.rivernet.

---

## rivernet.write *Write a river network to three data files*

---

### Description

Writes a river network to three data files: coordinates of river reaches, attributes of reaches, and attributes of nodes.

### Usage

```
rivernet.write(x,
               file.reachcoord  = NA,
               file.reachattrib = NA,
               file.nodeattrib  = NA,
               sep              = "\t",
               subnets          = NA)
```

### Arguments

| | |
|---|---|
| x | Object of class "rivernet" e.g. read by the function rivernet.read. |
| file.reachcoord | File name for writing reach coordinates. |
| file.reachattrib | File name for writing reach attributes. |
| file.nodeattrib | File name for writing node attributes. |
| sep | Separator on data files. |
| subnets | Optional vector of sub-networks to write (with the default NA all sub-networks are written.. |

### Value

No return value.

### See Also

`rivernet.read`,
`prune.rivernet`,
`upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`,
`utility`.

## splitreach — *Split a reach into two sub-reaches*

### Description

Splits a reach at a given part of its length into two subreaches

### Usage

```
splitreach(net,reachind,fract,...)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `reachind` | Index of a reach in the rivernet object. |
| `fract` | Fraction of the length at which the reach should be split. Needs to be between 0 and 1. The fraction is counted from the logical start point of the reach. |
| `...` | . |

### Value

The complete rivernet with the splitted reach.

### See Also

`rivernet.read`, `getnodeind.rivernet`, `upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`, `utility`.

## `splitreach.rivernet` *Split a reach into two sub-reaches*

### Description

Splits a reach at a given part of its length into two subreaches

### Usage

```
## S3 method for class 'rivernet'
splitreach(net,reachind,fract,...)
```

### Arguments

| | |
|---|---|
| `net` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `reachind` | Index of a reach in the rivernet object. |
| `fract` | Fraction of the length at which the reach should be split. Needs to be between 0 and 1. The fraction is counted from the logical start point of the reach. |
| `...` | . |

### Value

The complete rivernet with the splitted reach.

### See Also

`rivernet.read`, `getnodeind.rivernet`, `upstreamconnectivity.rivernet`, `adjacentreaches.rivernet`, `utility`.

## `upstreamconnectivity` *Calculates connectivity from outlet to upstream reaches*

### Description

Calculates connectivity from the outlet to upstream reaches, in particular to stream order 1 reaches. Criteria can be provided for reaches as well as nodes.

### Usage

```
upstreamconnectivity(x,...)
```

### Arguments

| | |
|---|---|
| `x` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `...` | . |

### Value

List with the following entries:
`paths.reachable`: List of numeric vectors specifying the indices of the reaches of a reachable path. Note that this is a subset of the reaches given in the same component of the element `pahts` of the river network.
`firstorder.reachable`: Vector of logicals indicating if the corresponding paths given above reach first order rivers. `fract.firstorder.reachable`: fraction of first order segments that can be reached (under the conditions specified by `crit.reach`, `crit.node` and `thresh.length`).
`streamorder.reachable`: vector of reachable rivers of order 1, 2, etc.

### See Also

`rivernet.read`, `mean.rivernet`, `adjacentreaches.rivernet`,
`utility`.

---

## upstreamconnectivity.rivernet

*Calculates connectivity from outlet to upstream reaches*

---

### Description

Calculates connectivity from the outlet to upstream reaches, in particular to stream order 1 reaches. Criteria can be provided for reaches as well as nodes.

### Usage

```
## S3 method for class 'rivernet'
upstreamconnectivity(x,crit.reach,crit.node,thresh.length=0,...)
```

### Arguments

| | |
|---|---|
| `x` | Object of class "rivernet" e.g. read by the function `rivernet.read`. |
| `crit.reach` | Logical vector of the same length as the number of reaches. Indicates whether the reach can be counted as connecting between adjacent reaches. |
| `crit.node` | Logical vector of the same length as the number of (internal) nodes of the river network. Indicates wheter the node can be counted as connecting between adjacent reaches (e.g. small drop height that allows for fish migration). |
| `thresh.length` | Threshold lenght below which a reach (or a sequence of reaches) can be accepted as connecting even if their criterion `crit.reach` is not fulfilled. |
| `...` | . |

## Value

List with the following entries:
`paths.reachable`: List of numeric vectors specifying the indices of the reaches of a reachable path. Note that this is a subset of the reaches given in the same component of the element `paths` of the river network.
`firstorder.reachable`: Vector of logicals indicating if the corresponding paths given above reach first order rivers. `fract.firstorder.reachable`: fraction of first order segments that can be reached (under the conditions specified by `crit.reach`, `crit.node` and `thresh.length`).
`streamorder.reachable`: vector of reachable rivers of order 1, 2, etc.

## See Also

`rivernet.read`, `mean.rivernet`, `adjacentreaches.rivernet`,
`utility`.

# Index