Package ‘nhdplusTools’

August 18, 2021

Type Package

Title NHDPlus Tools

Version 0.4.3

Description Tools for traversing and working with National Hydrography Dataset Plus (NHD-Plus) data. All methods implemented in 'nhdplusTools' are available in the NHDPlus documentation available from the US Environmental Protection Agency <https://www.epa.gov/waterdata/basic-information>.

URL https://usgs-r.github.io/nhdplusTools/

https://github.com/usgs-r/nhdplusTools/

BugReports https://github.com/usgs-r/nhdplusTools/issues/

Depends R (>= 4.0)

Imports dplyr, sf, RANN, units, magrittr, jsonlite, http, igraph, xml2, R.utils, utils, tidyr, methods, rosm, prettymap, fst, dataRetrieval, tools, zip, pbapply

Suggests testthat, knitr, rmarkdown, markdown, ggmap, ggplot2, sp, lwgeom, devtools, codetools, data.table, parallel, s2

License CC0

Encoding UTF-8

RoxygenNote 7.1.1

VignetteBuilder knitr

NeedsCompilation no

Author David Blodgett [aut, cre],

Mike Johnson [ctb] (<https://orcid.org/0000-0002-5288-8350>)

Maintainer David Blodgett <dblodgett@usgs.gov>

Repository CRAN

Date/Publication 2021-08-18 10:00:07 UTC
R topics documented:

- add_plus_network_attributes ........................................ 3
- align_nhdplus_names ................................................... 4
- calculate_arbolate_sum ................................................ 5
- calculate_total_drainage_area ...................................... 5
- disambiguate_flowline_indexes .................................... 6
- discover_nhdplus_id ................................................... 8
- discover_nldi_characteristics ...................................... 8
- download_nhdplusr ................................................... 9
- download_nhdplusv2 .................................................. 10
- download_rf1 .......................................................... 11
- download_vaa .......................................................... 11
- download_wbd .......................................................... 12
- get_DD ................................................................. 13
- get_DM ................................................................. 14
- get_flowline_index .................................................... 15
- get_gagesII ............................................................ 16
- get_hr_data ............................................................ 17
- get_huc12 ............................................................. 18
- get_huc8 ............................................................... 19
- get_hydro_location ................................................... 19
- get_levelpaths ......................................................... 20
- get_nhdarea ........................................................... 21
- get_nhdplus .......................................................... 22
- get_nhdplusr .......................................................... 23
- get_nldi_basin ......................................................... 25
- get_nldi_characteristics ............................................. 26
- get_nldi_feature ...................................................... 27
- get_node ............................................................... 27
- get_nwis ............................................................... 28
- get_pathlength ......................................................... 29
- get_pfaf .............................................................. 30
- get_streamlevel ........................................................ 31
- get_streamorder ....................................................... 32
- get_terminal .......................................................... 33
- get_UM ................................................................. 34
- get_UT ................................................................. 35
- get_vaa ............................................................... 35
- get_vaa_names ........................................................ 36
- get_vaa_path .......................................................... 37
- get_waterbodies ....................................................... 38
- get_waterbody_index ................................................ 38
- make_standalone ....................................................... 39
- navigate_nldi .......................................................... 40
- nhdplusTools_data_dir .............................................. 42
- nhdplus_path ........................................................ 42
- plot_nhdplus .......................................................... 43
**add_plus_network_attributes**

Add NHDPlus Network Attributes to a provided network.

**Description**

Given a river network with required base attributes, adds the NHDPlus network attributes: hydrosequence, levelpath, terminalpath, pathlength, down levelpath, down hydroseq, total drainage area, and terminalflag. The function implements two parallelization schemes for small and large basins respectively. If a number of cores is specified, parallel execution will be used.

**Usage**

```r
add_plus_network_attributes(
  net,
  override = 5,
  cores = NULL,
  split_temp = NULL,
  status = TRUE
)
```

**Arguments**

- `net`: data.frame containing comid, tocomid, nameID, lengthkm, and areasqkm. Additional attributes will be passed through unchanged. tocomid == 0 is the convention used for outlets. If a "weight" column is provided, it will be used in `get_levelpaths` otherwise, arbolate sum is calculated for the network and used as the weight.
- `override`: numeric factor to be passed to `get_levelpaths`
- `cores`: integer number of processes to spawn if run in parallel.
- `split_temp`: character path to optional temporary copy of the network split into independent sub-networks. If it exists, it will be read from disk rather than recreated.
- `status`: logical should progress be printed?

**Value**

data.frame with added attributes
align_nhdplus_names

### Examples

```r
source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))

test_flowline <- prepare_nhdplus(walker_flowline, 0, 0, FALSE)

test_flowline <- data.frame(
  comid = test_flowline$COMID,
  tocomid = test_flowline$toCOMID,
  nameID = walker_flowline$GNIS_ID,
  lengthkm = test_flowline$LENGTHKM,
  areasqkm = walker_flowline$AreaSqKM)

add_plus_network_attributes(test_flowline)
```

---

### align_nhdplus_names

**Align NHD Dataset Names**

**Description**

This function takes any NHDPlus dataset and aligns the attribute names with those used in nhdplusTools.

**Usage**

```r
align_nhdplus_names(x)
```

**Arguments**

- `x`: a sf object of nhdplus flowlines

**Value**

data.frame renamed sf object

**Examples**

```r
source(system.file("extdata/new_hope_data.R", package = "nhdplusTools"))

names(new_hope_flowline)

names(new_hope_flowline) <- tolower(names(new_hope_flowline))

new_hope_flowline <- align_nhdplus_names(new_hope_flowline)

names(new_hope_flowline)
```
#### calculate_arbolate_sum

*Calculate Arbolate Sum*

**Description**

Calculates arbolate sum given a dendritic network and incremental lengths. Arbolate sum is the total length of all upstream flowlines.

**Usage**

```r
calculate_arbolate_sum(x)
```

**Arguments**

- `x`  
  data.frame with ID, toID, and length columns.

**Value**

numeric with arbolate sum.

**Examples**

```r
library(dplyr)
source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))
catchment_length <- select(walker_flowline, COMID, AreaSqKM) %>%
  right_join(prepare_nhdplus(walker_flowline, 0, 0,
    purge_non_dendritic = FALSE, warn = FALSE), by = "COMID") %>%
  select(ID = COMID, toID = toCOMID, length = LENGTHKM)

arb_sum <- calculate_arbolate_sum(catchment_length)
catchment_length$arb_sum <- arb_sum
catchment_length$nhd_arb_sum <- walker_flowline$ArbolateSu

mean(abs(catchment_length$arb_sum - catchment_length$nhd_arb_sum))
max(abs(catchment_length$arb_sum - catchment_length$nhd_arb_sum))
```

---

#### calculate_total_drainage_area

*Total Drainage Area*

**Description**

Calculates total drainage area given a dendritic network and incremental areas.
Usage

calculate_total_drainage_area(x)

Arguments

x
data.frame with ID, toID, and area columns.

Value

numeric with total area.

Examples

library(dplyr)
source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))
catchment_area <- select(walker_flowline, COMID, AreaSqKM) %>%
  right_join(prepare_nhdplus(walker_flowline, 0, 0,
    purge_non_dendritic = FALSE, warn = FALSE), by = "COMID") %>%
  select(ID = COMID, toID = toCOMID, area = AreaSqKM)

new_da <- calculate_total_drainage_area(catchment_area)
catchment_area$totda <- new_da
catchment_area$nhdptotda <- walker_flowline$TotDASqKM

mean(abs(catchment_area$totda - catchment_area$nhdptotda))
max(abs(catchment_area$totda - catchment_area$nhdptotda))

disambiguate_flowline_indexes

Disambiguate Flowline Indexes

Description

Given a set of flowline indexes and numeric or ascii criteria, return closest match. If numeric
criteria are used, the minimum difference in the numeric attribute is used for disambiguation. If
ascii criteria are used, the adist function is used with the following algorithm: ‘1 - adist_score / max_string_length’. Comparisons ignore case.

Usage

disambiguate_flowline_indexes(indexes, flowpath, hydro_location)
Arguments

indexes  
data.frame as output from `get_flowline_index` with more than one hydrologic location per indexed point.

flowpath  
data.frame with two columns. The first should join to the COMID field of the indexes and the second should be the numeric or ascii metric such as drainage area or GNIS Name. Names of this data.frame are not used.

hydro_location  
data.frame with two columns. The first should join to the id field of the indexes and the second should be the numeric or ascii metric such as drainage area or GNIS Name. Names of this data.frame are not used.

Value

data.frame indexes deduplicated according to the minimum difference between the values in the metric columns. If two or more result in the same "minimum" value, duplicates will be returned.

Examples

```r
source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))

hydro_location <- sf::st_sf(id = c(1, 2, 3),
  geom = sf::st_sfc(list(sf::st_point(c(-76.86934, 39.49328)),
                      sf::st_point(c(-76.91711, 39.40884)),
                      sf::st_point(c(-76.88081, 39.36354))),
  crs = 4326),
  totda = c(23.6, 7.3, 427.9),
  nameid = c("Patapsco", ",", "Falls Run River")

flowpath <- dplyr::select(sample_flines,
  comid = COMID,
  totda = TotDA_sqKM,
  nameid = GNIS_NAME,
  REACHCODE,
  ToMeas,
  FromMeas)

indexes <- get_flowline_index(flowpath,
  hydro_location,
  search_radius = 0.2,
  max_matches = 10)

disambiguate_flowline_indexes(indexes,
  dplyr::select(flowpath, comid, totda),
  dplyr::select(hydro_location, id, totda))

result <- disambiguate_flowline_indexes(indexes,
  dplyr::select(flowpath, comid, nameid),
  dplyr::select(hydro_location, id, nameid))

result[result$id == 1, ]
result[result$id == 2, ]
```
discover_nldi_characteristics

result[result$id == 3, ]

discover_nhdplus_id  Discover NHDPlus ID

Description
Multipurpose function to find a COMID of interest.

Usage
discover_nhdplus_id(point = NULL, nldi_feature = NULL)

Arguments
point  An sf POINT including crs as created by: sf::st_sfc(sf::st_point(....), crs)
nldi_feature  list with names ‘featureSource’ and ‘featureID’ where ‘featureSource’ is derived
from the "source" column of the response of get_nldi_sources and the ‘feature-Source’ is a known identifier from the specified ‘featureSource’.

Value
integer COMID

Examples
point <- sf::st_sfc(sf::st_point(c(-76.87479, 39.48233)), crs = 4326)
discover_nhdplus_id(point)

nldi_nwis <- list(featureSource = "nwissite", featureID = "USGS-08279500")
discover_nhdplus_id(nldi_feature = nldi_nwis)

discover_nldi_characteristics  Discover Characteristics Metadata

Description
Provides access to metadata for characteristics that are returned by ‘get_nldi_characteristics()’.

Usage
discover_nldi_characteristics(type = "all")
download_nhdplushr

Arguments

type character "all", "local", "total", or "divergence_routed".

Value
data.frame containing available characteristics

Examples
chars <- discover_nldi_characteristics()
names(chars)
head(chars$local, 10)

download_nhdplushr Download NHDPlus HiRes

Description
Download NHDPlus HiRes

Usage
download_nhdplushr(nhd_dir, hu_list, download_files = TRUE)

Arguments

nhd_dir character directory to save output into
hu_list character vector of hydrologic region(s) to download. Use get_huc8 to find HU
codes of interest. Accepts two digit and four digit codes.
download_files boolean if FALSE, only URLs to files will be returned can be hu02s and/or
hu04s

Value
character Paths to geodatabases created.

Examples
hu <- nhdplusTools::get_huc8(sf::st_sfc(sf::st_point(c(-73, 42)), crs = 4326))
(hu <- substr(hu$huc8, 1, 2))
download_nhdplushr(tempdir(), c(hu, "0203"), download_files = FALSE)
download_nhdplusv2  Download seamless National Hydrography Dataset Version 2 (NHD-PlusV2)

Description

This function downloads and decompresses staged seamless NHDPlusV2 data. The following requirements are needed: p7zip (MacOS), 7zip (windows) Please see: https://www.epa.gov/waterdata/get-nhdplus-national-hydrography-dataset-plus-data for more information and metadata about this data.

Default downloads lower-48 only. See examples for islands. No Alaska data are available.

Usage

download_nhdplusv2(outdir, url = paste0("https://s3.amazonaws.com/edap-nhdplus/NHDPlusV21/", "Data/NationalData/NHDPlusV21_NationalData_Seamless", ",_Geodatabase_Lower48_07.7z"), progress = TRUE)

Arguments

outdir  The folder path where data should be downloaded and extracted
url  the location of the online resource
progress  boolean display download progress?

Value

character path to the local geodatabase

Examples

## Not run:
download_nhdplusV2("./data/nhd/"

download_nhdplusv2(outdir = "./inst/",
url = paste0("https://s3.amazonaws.com/edap-nhdplus/NHDPlusV21/", ",Data/NationalData/NHDPlusV21_NationalData_Seamless", ",_Geodatabase_HI_PR_VI_PI_03.7z")

## End(Not run)
download_rf1

**Description**

This function downloads and decompresses staged RF1 data. See: https://water.usgs.gov/GIS/metadata/usgswrd/XML/erf1_2.xml for metadata.

**Usage**

```r
download_rf1(
  outdir,
  url = "https://water.usgs.gov/GIS/dsdl/erf1_2.e00.gz",
  progress = TRUE
)
```

**Arguments**

- `outdir`: The folder path where data should be downloaded and extracted
- `url`: the location of the online resource
- `progress`: boolean display download progress?

**Value**

character path to the local e00 file

**Examples**

```r
## Not run:
download_wbd("./data/rf1/")
## End(Not run)
```

---

download_vaa

**Description**

downloads and caches nhdplusVAA data on your computer

**Usage**

```r
download_vaa(path = get_vaa_path(), force = FALSE)
```
Arguments

path
character path where the file should be saved. Default is a persistent system data
as retrieved by nhdplusTools_data_dir. Also see: get_vaa_path

force
logical. Force data re-download. Default = FALSE

Details

The VAA data is a aggregate table of information from the NHDPlusV2 elevslope.dbf(s), PlusFlow-
lineVAA.dbf(s); and NHDFlowlines. All data originates from the EPA NHDPlus Homepage here.
To see the location of cached data on your machine use get_vaa_path. To view aggregate data and
documentation, see here

Value

character path to cached data

download_wbd

Download the seamless Watershed Boundary Dataset (WBD)

Description

This function downloads and decompresses staged seamless WBD data. Please see: https://prd-
tnm.s3.amazonaws.com/StagedProducts/Hydrography/WBD/National/GDB/WBD_National_GDB.xml
for metadata.

Usage

download_wbd(
  outdir,
  progress = TRUE
)

Arguments

outdir
The folder path where data should be downloaded and extracted

url
the location of the online resource

progress
boolean display download progress?

Value

character path to the local geodatabase
get_DD

Examples

## Not run:
download_wbd("./data/wbd/")

## End(Not run)

get_DD

Navigate Downstream with Diversions

Description

Traverse NHDPlus network downstream with diversions NOTE: This algorithm may not scale well in large watersheds. For reference, the lower Mississippi will take over a minute.

Usage

get_DD(network, comid, distance = NULL)

Arguments

- **network**: data.frame NHDPlus flowlines including at a minimum: COMID, DnMinorHyd, DnHydroseq, and Hydroseq.
- **comid**: integer identifier to start navigating from.
- **distance**: numeric distance in km to limit how many COMIDs are returned. The COMID that exceeds the distance specified is returned. The longest of the diverted paths is used for limiting distance.

Value

integer vector of all COMIDs downstream of the starting COMID

Examples

library(sf)
start_COMID <- 11688818
source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))

DD_COMIDs <- get_DD(sample_flines, start_COMID, distance = 4)
plot(dplyr::filter(sample_flines, COMID %in% DD_COMIDs)$geom,
col = "red", lwd = 2)

DM_COMIDs <- get_DM(sample_flines, start_COMID, distance = 4)
plot(dplyr::filter(sample_flines, COMID %in% DM_COMIDs)$geom,
col = "blue", add = TRUE, lwd = 2)
**get_DM**

*Navigate Downstream Mainstem*

**Description**
Traverse NHDPlus network downstream main stem

**Usage**

```r
get_DM(network, comid, distance = NULL, sort = FALSE, include = TRUE)
```

**Arguments**

- `network` data.frame NHDPlus flowlines including at a minimum: COMID, LENGTHKM, DnHydroseq, and Hydroseq.
- `comid` integer identifier to start navigating from.
- `distance` numeric distance in km to limit how many COMIDs are returned. The COMID that exceeds the distance specified is returned.
- `sort` if TRUE, the returned COMID vector will be sorted in order of distance from the input COMID (nearest to farthest)
- `include` if TRUE, the input COMID will be included in the returned COMID vector

**Value**
integer vector of all COMIDs downstream of the starting COMID along the mainstem

**Examples**

```r
library(sf)

source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))

plot(sample_flines$geom)
start_COMID <- 11690092
DM_COMIDs <- get_DM(sample_flines, start_COMID)
plot(dplyr::filter(sample_flines, COMID %in% DM_COMIDs)$geom,
  col = "red", add = TRUE, lwd = 3)

DM_COMIDs <- get_DM(sample_flines, start_COMID, distance = 40)
plot(dplyr::filter(sample_flines, COMID %in% DM_COMIDs)$geom,
  col = "blue", add = TRUE, lwd = 2)
```
get_flowline_index  Get Flowline Index

Description

given an sf point geometry column, return COMID, reachcode, and measure for each.

Usage

get_flowline_index(
  flines,
  points,
  search_radius = 0.1,
  precision = NA,
  max_matches = 1
)

Arguments

flines  sf data.frame of type LINESTRING or MULTILINESTRING including COMID, REACHCODE, ToMeas, and FromMeas. Can be "download_nhdplusv2" and remote nhdplusv2 data will be downloaded for the bounding box surround the submitted points. NOTE: The download option may not work for large areas, use with caution.
points  sf or sfc of type POINT
search_radius  numeric the distance for the nearest neighbor search to extend. See RANN nn2 documentation for more details.
precision  numeric the resolution of measure precision in the output in meters.
max_matches  numeric the maximum number of matches to return if multiple are found in search_radius

Details

Note 1: Inputs are cast into LINESTRINGS. Because of this, the measure output of inputs that are true multipart lines may be in error.

Note 2: This algorithm finds the nearest node in the input flowlines to identify which flowline the point should belong to. As a second pass, it can calculate the measure to greater precision than the nearest flowline geometry node.

Note 3: Offset is returned in units consistent with the projection of the flowlines.

Note 4: See ‘dfMaxLength’ input to sf::st_segmentize() for details of handling of precision parameter.

Note 5: “from” is downstream – 0 is the outlet "to" is upstream – 100 is the inlet
get_gagesII

Value

data.frame with five columns, id, COMID, REACHCODE, REACH_meas, and offset. id is the row
or list element in the point input.

Examples

source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))

get_flowline_index(sample_flines,
   sf::st_sfc(sf::st_point(c(-76.87479,
   39.48233)),
   crs = 4326))

get_flowline_index("download_nhdplusv2",
   sf::st_sfc(sf::st_point(c(-76.87479,
   39.48233)),
   crs = 4326))

get_flowline_index(sample_flines,
   sf::st_sfc(sf::st_point(c(-76.87479,
   39.48233)),
   crs = 4326), precision = 30)

get_flowline_index(sample_flines,
   sf::st_sfc(list(sf::st_point(c(-76.86934, 39.49328)),
   sf::st_point(c(-76.91711, 39.40884)),
   sf::st_point(c(-76.88081, 39.36354)) ),
   crs = 4326),
   search_radius = 0.2,
   max_matches = 10)

get_gagesII Find GAGESII Features

Description

Subsets the gagesII dataset by location (POINT), area (POLYGON), or set of IDs.

Usage

get_gagesII(AOI = NULL, id = NULL, t_srs = NULL, buffer = 0.5)
Arguments

AOI sf (MULTI)POINT or (MULTI)POLYGON. An ‘area of interest’ can be provided as either a location (sf POINT) or area (sf POLYGON) in any Spatial Reference System.

id NWIS Gage ID(s)

t_srs character (PROJ string or EPSG code) or numeric (EPSG code). A user specified - target - Spatial Reference System (SRS/CRS) for returned objects. Will default to the CRS of the input AOI if provided, and to 4326 for ID requests.

buffer numeric. The amount (in meters) to buffer a POINT AOI by for an extended search. Default = 0.5

Details

The returned object(s) will have the same Spatial Reference System (SRS) as the input AOI. If a individual or set of IDs are used to query, then the default geoserver CRS of EPSG:4326 is preserved. In all cases, a user-defined SRS can be passed to t_srs which will override all previous SRS’s (either input or default). All buffer and distance operations are handled internally using in EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object

---

**get_hr_data**  
*Get NHDPlus HiRes Data*

**Description**

Use to remove unwanted detail NHDPlusHR data See *get_nhdplushr* for examples.

**Usage**

```r
get_hr_data(
  gdb,
  layer = NULL,
  min_size_sqkm = NULL,
  simp = NULL,
  proj = NULL,
  rename = TRUE
)
```
get_huc12

Arguments

gdb  character path to geodatabase to get data from.
layer character layer name from geodatabase found with \texttt{st\_layers}
min\_size\_sqkm numeric minimum basin size to be included in the output
simp numeric simplification tolerance in units of projection
proj a projection specification compatible with \texttt{st\_crs}
rename boolean if TRUE, nhdplusTools standard attribute values will be applied.

Value

\begin{verbatim}
sf data.frame containing requested data
\end{verbatim}

Description

Subsets the WBD level 12 features by location (POINT), area (POLYGON), or set of IDs.

Usage

\begin{verbatim}
get_huc12(AOI = NULL, id = NULL, t_srs = NULL, buffer = 0.5)
\end{verbatim}

Arguments

\begin{itemize}
\item \texttt{AOI} \begin{verbatim}sf\end{verbatim} (MULTI)POINT or (MULTI)POLYGON. An 'area of interest' can be provided as either a location (sf POINT) or area (sf POLYGON) in any Spatial Reference System.
\item \texttt{id} WBD HUC12 ID(s)
\item \texttt{t\_srs} character (PROJ string or EPSG code) or numeric (EPSG code). A user specified target - Spatial Reference System (SRS/CRS) for returned objects. Will default to the CRS of the input AOI if provided, and to 4326 for ID requests.
\item \texttt{buffer} numeric. The amount (in meters) to buffer a POINT AOI by for an extended search. Default = 0.5
\end{itemize}

Details

The returned object(s) will have the same Spatial Reference System (SRS) as the input AOI. If a individual or set of IDs are used to query, then the default geoserver CRS of EPSG:4326 is preserved. In all cases, a user-defined SRS can be passed to \texttt{t\_srs} which will override all previous SRS's (either input or default). All buffer and distance operations are handled internally using in EPSG:5070 Albers Equal Area projection

Value

\begin{verbatim}
a simple features (sf) object
\end{verbatim}
**get_huc8**

*Find WBD HUC 08 unit subsets*

**Description**

Subsets the WBD level 08 features by location (POINT), area (POLYGON), or set of IDs.

**Usage**

`get_huc8(AOI = NULL, id = NULL, t_srs = NULL, buffer = 0.5)`

**Arguments**

- **AOI**: sf (MULTI)POINT or (MULTI)POLYGON. An ’area of interest’ can be provided as either a location (sf POINT) or area (sf POLYGON) in any Spatial Reference System.
- **id**: WBD HUC08 ID(s)
- **t_srs**: character (PROJ string or EPSG code) or numeric (EPSG code). A user specified target Spatial Reference System (SRS/CRS) for returned objects. Will default to the CRS of the input AOI if provided, and to 4326 for ID requests.
- **buffer**: numeric. The amount (in meters) to buffer a POINT AOI by for an extended search. Default = 0.5

**Details**

The returned object(s) will have the same Spatial Reference System (SRS) as the input AOI. If an individual or set of IDs are used to query, then the default geoserver CRS of EPSG:4326 is preserved. In all cases, a user-defined SRS can be passed to `t_srs` which will override all previous SRS’s (either input or default). All buffer and distance operations are handled internally using in EPSG:5070 Albers Equal Area projection.

**Value**

a simple features (sf) object

---

**get_hydro_location**

*get hydro location*

**Description**

given a flowline index, returns the hydrologic location (point) along the specific linear element referenced by the index.

**Usage**

`get_hydro_location(indexes, flowpath)`
get_levelpaths

Arguments

indexes data.frame as output from get_flowline_index.
flowpath data.frame with two columns. The first should join to the COMID field of the indexes and the second should be linear geometry.

Examples

source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))

indexes <- get_flowline_index(sample_flines,
                               sf::st_sfc(sf::st_sfc(list(sf::st_point(c(-76.86934, 39.49328)),
                                                    sf::st_point(c(-76.91711, 39.40884)),
                                                    sf::st_point(c(-76.88081, 39.36354))),
                                    crs = 4326)))

get_hydro_location(indexes, sample_flines)

get_levelpaths  Get Level Paths

Description

Calculates level paths using the stream-leveling approach of NHD and NHDPlus. In addition to a levelpath identifier, a topological sort and levelpath outlet identifier is provided in output. If arbolate sum is provided in the weight column, this will match the behavior of NHDPlus. Any numeric value can be included in this column and the largest value will be followed when no nameID is available.

Usage

get_levelpaths(x, override_factor = NULL, status = FALSE, cores = NULL)

Arguments

x data.frame with ID, toID, nameID, and weight columns.
override_factor numeric factor to use to override nameID. If 'weight' is 'numeric_factor' times larger on a path, it will be followed regardless of the nameID indication.
status boolean if status updates should be printed.
cores numeric number of cores to use in initial path ranking calculations.
Details

1. levelpath provides an identifier for the collection of flowlines that make up the single mainstem flowpath of a total upstream aggregate catchment.

2. outletID is the catchment ID (COMID in the case of NHDPlus) for the catchment at the outlet of the levelpath the catchment is part of.

3. topo_sort is similar to Hydroseq in NHDPlus in that large topo_sort values are upstream of small topo_sort values. Note that there are many valid topological sort orders of a directed graph. The sort order output by this function is generated using ‘igraph::topo_sort’.

Value

data.frame with ID, outletID, topo_sort, and levelpath columns. See details for more info.

Examples

```r
source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))

test_flowline <- prepare_nhdplus(walker_flowline, 0, 0, FALSE)

test_flowline <- data.frame(
ID = test_flowline$COMID,
toID = test_flowline$toCOMID,
nameID = walker_flowline$GNIS_ID,
weight = walker_flowline$ArbolateSu,
stringsAsFactors = FALSE)

get_levelpaths(test_flowline)
```

---

get_nhdarea  Find NHD Areas

Description

Subsets NHD Area features by location (POINT), area (POLYGON), or set of IDs.

Usage

```r
get_nhdarea(AOI = NULL, id = NULL, t_srs = NULL, buffer = 0.5)
```

Arguments

- **AOI**: sf (MULTI)POINT or (MULTI)POLYGON. An 'area of interest' can be provided as either a location (sf POINT) or area (sf POLYGON) in any Spatial Reference System.
- **id**: NHD Area COMID(s)
get_nhdplus

t_srs character (PROJ string or EPSG code) or numeric (EPSG code). A user specified target -Spatial Reference System (SRS/CRS) for returned objects. Will default to the CRS of the input AOI if provided, and to 4326 for ID requests.

buffer numeric. The amount (in meters) to buffer a POINT AOI by for an extended search. Default = 0.5

Details

The returned object(s) will have the same Spatial Reference System (SRS) as the input AOI. If a individual or set of IDs are used to query, then the default geoserver CRS of EPSG:4326 is preserved. In all cases, a user-defined SRS can be passed to t_srs which will override all previous SRS’s (either input or default). All buffer and distance operations are handled internally using in EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object

get_nhdplus

Get National Hydrography Dataset V2 Subsets (Multirealization)

Description

Subsets NHDPlusV2 features by location (POINT), area (POLYGON), or set of COMIDs. Multi realizations are supported allowing you to query for flowlines, catchments, or outlets.

Usage

get_nhdplus(
  AOI = NULL,
  comid = NULL,
  nwis = NULL,
  realization = "flowline",
  streamorder = NULL,
  t_srs = NULL
)

Arguments

AOI sf (MULTI)POINT or (MULTI)POLYGON. An 'area of interest' can be provided as either a location (sf POINT) or area (sf POLYGON) in any Spatial Reference System.

comid numeric or character. Search for NHD features by COMID(s)

nwis numeric or character. Search for NHD features by collocated NWIS identifiers

realization character. What realization to return. Default is flowline and options include: outlet, flowline, catchment, and all
streamorder numeric or character. Only return NHD flowlines with a streamorder greater then or equal to this value for input value and higher. Only usable with AOI and flowline realizations.

t_srs character (PROJ string or EPSG code) or numeric (EPSG code). A user specified - target -Spatial Reference System (SRS/CRS) for returned objects. Will default to the CRS of the input AOI if provided, and to 4326 for ID requests.

Details
The returned object(s) will have the same Spatial Reference System (SRS) as the input AOI. If a individual or set of IDs are used to query, then the default geoserver CRS of EPSG:4326 is preserved. In all cases, a user-defined SRS can be passed to t_srs which will override all previous SRS's (either input or default). All buffer and distance operations are handled internally using in EPSG:5070 Albers Equal Area projection

Value
sfc a single, or list, of simple feature objects

Examples

point <- sf::st_sfc(sf::st_point(c(-119.845, 34.4146)), crs = 4326)
get_nhdplus(point)
get_nhdplus(point, realization = "catchment")
get_nhdplus(point, realization = "all")
get_nhdplus(comid = 101)
get_nhdplus(nwis = c(11120000, 11120500))
area <- sf::st_as_sfc(sf::st_bbox(c(xmin = -119.8851, xmax =-119.8361,
ymax = 34.42439, ymin = 34.40473), crs = 4326))
get_nhdplus(area)
get_nhdplus(area, realization = "flowline", streamorder = 3)
get_nhdplushr

```r
check_terminals = TRUE,
overwrite = FALSE,
keep_cols = NULL,
```

Arguments

- `hr_dir`: character directory with geodatabases (gdb search is recursive)
- `out_gpkg`: character path to write output geopackage
- `pattern`: character optional regex to select certain files in `hr_dir`
- `check_terminals`: boolean if TRUE, run `make_standalone` on output.
- `overwrite`: boolean should the output overwrite? If false and the output layer exists, it will be read and returned so this function will always return data even if called a second time for the same output. This is useful for workflows. Note that this will NOT delete the entire Geopackage. It will overwrite on a per layer basis.
- `keep_cols`: character vector of column names to keep in the output. If NULL, all will be kept.
- `...`: parameters passed along to `get_hr_data` for "NHDFlowline" layers.

Details

NHDFlowline is joined to value added attributes prior to being returned. Names are not modified from the NHDPlusHR geodatabase. Set layers to "NULL" to get all layers.

Value

sf data.frames containing output that may also be written to a geopackage for later use.

Examples

```r
## Not run:
# Note this will download a lot of data to a temp directory.
# Change 'temp_dir' to your directory of choice.
temp_dir <- file.path(nhdplusTools_data_dir(), "temp_hr_cache")

download_dir <- download_nhdplushr(temp_dir, c("0302", "0303"))

get_nhdplushr(download_dir, file.path(download_dir, "nhdplus_0302-03.gpkg"))

get_nhdplushr(download_dir,}
```
## get_nldi_basin

Get NLDI Basin Boundary

### Description

Get a basin boundary for a given NLDI feature.

### Usage

```r
get_nldi_basin(nldi_feature)
```

### Arguments

- `nldi_feature` list with names ‘featureSource’ and ‘featureID’ where ‘featureSource’ is derived from the "source" column of the response of `dataRetrieval::get_nldi_sources()` and the ‘featureSource’ is a known identifier from the specified ‘featureSource’.

### Details

Only resolves to the nearest NHDPlus catchment divide. See: https://waterdata.usgs.gov/blog/nldi-intro/ for more info on the nldi.

### Value

sf data.frame with result basin boundary

### Examples

```r
library(sf)
library(dplyr)

nldi_nwis <- list(featureSource = "nwissite", featureID = "USGS-05428500")

basin <- get_nldi_basin(nldi_feature = nldi_nwis)
```
get_nldi_characteristics

Get Catchment Characteristics

Description

Retrieves catchment characteristics from the Network Linked Data Index. Metadata for these characteristics can be found using `discover_nldi_characteristics()`.

Usage

```r
get_nldi_characteristics(nldi_feature, type = "local")
```

Arguments

- `nldi_feature` list with names `featureSource` and `featureID` where `featureSource` is derived from the "source" column of the response of `dataRetrieval::get_nldi_sources()` and the `featureSource` is a known identifier from the specified `featureSource`.
- `type` character "all", "local", "total", or "divergence_routed".

Value

`data.frame` containing requested characteristics

Examples

```r
chars <- get_nldi_characteristics(list(featureSource = "nwissite", featureID = "USGS-05429700"))
names(chars)
head(chars$local, 10)
```
get_nldi_feature

Get NLDI Feature

Description

Get a single feature from the NLDI

Usage

get_nldi_feature(nldi_feature)

Arguments

nldi_feature list with names 'featureSource' and 'featureID' where 'featureSource' is derived from the "source" column of the response of dataRetrieval::get_nldi_sources() and the 'featureSource' is a known identifier from the specified 'featureSource'.

Value

sf data.frame with one feature

Examples

get_nldi_feature(list("featureSource" = "nwissite", featureID = "USGS-05428500"))

generate_feature

Get flowline node

Description

Given one or more flowlines, returns a particular node from the flowline.

Usage

get_node(x, position = "end")

Arguments

x sf data.frame with one or more flowlines
position character either "start" or "end"

Value

sf data.frame containing requested nodes
**get_nwis**

*Discover USGS NWIS Stream Gages*

**Description**

Returns a POINT feature class of active, stream network, NWIS gages for an Area of Interest. If a POINT feature is used as an AOI, then the returned sites within the requested buffer, are sorted by distance (in meters) from that POINT.

**Usage**

```r
get_nwis(AOI = NULL, t_srs = NULL, buffer = 20000)
```

**Arguments**

- **AOI**
  - `sf` (MULTI)POINT or (MULTI)POLYGON. An 'area of interest' can be provided as either a location (sf POINT) or area (sf POLYGON) in any Spatial Reference System.

- **t_srs**
  - character (PROJ string or EPSG code) or numeric (EPSG code). A user specified target Spatial Reference System (SRS/CRS) for returned objects. Will default to the CRS of the input AOI if provided, and to 4326 for ID requests.

- **buffer**
  - numeric. The amount (in meters) to buffer a POINT AOI by for an extended search. Default = 20,000. Returned results are arrange by distance from POINT AOI.

**Examples**

```r
source(system.file("extdata/sample_data.R", package = "nhdplusTools"))

fline <- sf::read_sf(sample_data, "NHDFlowline_Network")

start <- get_node(fline, "start")
end <- get_node(fline, "end")

plot(sf::st_zm(fline$geom), lwd = fline$StreamOrde, col = "blue")
plot(sf::st_geometry(start), add = TRUE)

plot(sf::st_zm(fline$geom), lwd = fline$StreamOrde, col = "blue")
plot(sf::st_geometry(end), add = TRUE)
```
get_pathlength

Details

The returned object(s) will have the same Spatial Reference System (SRS) as the input AOI. If a individual or set of IDs are used to query, then the default geoserver CRS of EPSG:4326 is preserved. In all cases, a user-defined SRS can be passed to t_srs which will override all previous SRS's (either input or default). All buffer and distance operations are handled internally using in EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object

| get_pathlength | Get path length |

Description

Generates the main path length to a basin’s terminal path.

Usage

get_pathlength(x)

Arguments

x data.frame with ID, toID, length columns.

Value

data.frame containing levelpaths for each ID

Examples

source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))

fl <- dplyr::select(prepare_nhdplus(walker_flowline, 0, 0),
  ID = COMID, toID = toCOMID, length = LENGTHKM)

get_pathlength(fl)
get_pfaf \hspace{2cm} \textit{Get Pfafstetter Codes (Experimental)}

\textbf{Description}

Determines Pfafstetter codes for a dendritic network with total drainage area, levelpath, and topo_sort attributes.

\textbf{Usage}

\texttt{get_pfaf(x, max_level = 2, status = FALSE)}

\textbf{Arguments}

- \texttt{x} \hspace{1cm} sf data.frame with ID, toID, totda, outletID, topo_sort, and levelpath attributes.
- \texttt{max_level} \hspace{1cm} integer number of pfaf levels to attempt to calculate. If the network doesn’t have resolution to support the desired level, unexpected behavior may occur.
- \texttt{status} \hspace{1cm} boolean print status or not

\textbf{Value}

data.frame with ID and pfaf columns.

\textbf{Examples}

\begin{verbatim}
library(dplyr)
source(system.file("extdata/nhdplushr_data.R", package = "nhdplusTools"))
hr_flowline <- align_nhdplus_names(hr_data$NHDFlowline)

fl <- select(hr_flowline, COMID, AreaSqKM) %>%
    right_join(prepare_nhdplus(hr_flowline, 0, 0,
        purge_non_dendritic = FALSE,
        warn = FALSE),
        by = "COMID") %>%
    sf::st_sf() %>%
    select(ID = COMID, toID = toCOMID, area = AreaSqKM)
fl$nameID = ""
fl$totda <- calculate_total_drainage_area(sf::st_set_geometry(fl, NULL))
fl <- left_join(fl, get_levelpaths(rename(sf::st_set_geometry(fl, NULL),
        weight = totda)), by = "ID")

pfaf <- get_pfaf(fl, max_level = 3)
fl <- left_join(fl, pfaf, by = "ID")
plot(fl[, "pf_level_3", lwd = 2])
\end{verbatim}
get_streamlevel

Description

Applies a topological sort and calculates stream level. Algorithm: Terminal level paths are assigned level 1 (see note 1). Paths that terminate at a level 1 are assigned level 2. This pattern is repeated until no paths remain.

If a TRUE/FALSE coastal attribute is included, coastal terminal paths begin at 1 and internal terminal paths begin at 4 as is implemented by the NHD stream leveling rules.

Usage

get_streamlevel(x)

Arguments

x data.frame with levelpathi, dnlevelpat, and optionally a coastal flag. If no coastal flag is included, all terminal paths are assumed to be coastal.
get_streamorder

Value

numeric stream order in same order as input

Examples

```r
source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))

test_flowline <- data.frame(
  levelpathi = walker_flowline$LevelPathI,
  dnlevelpat = walker_flowline$DnLevelPat)

test_flowline$dnlevelpat[1] <- 0

(level <- get_streamlevel(test_flowline))

walker_flowline$level <- level

plot(sf::st_geometry(walker_flowline), lwd = walker_flowline$level, col = "blue")

test_flowline$coastal <- rep(FALSE, nrow(test_flowline))

(level <- get_streamlevel(test_flowline))

test_flowline$coastal[!test_flowline$dnlevelpat %in% test_flowline$levelpathi] <- TRUE

(level <- get_streamlevel(test_flowline))
```

get_streamorder  Get Streamorder

Description

Applies a topological sort and calculates strahler stream order. Algorithm: If more than one upstream flowpath has an order equal to the maximum upstream order then the downstream flowpath is assigned the maximum upstream order plus one. Otherwise it is assigned the max upstream order.

Usage

```r
get_streamorder(x, status = FALSE)
```

Arguments

- `x` : data.frame with dendritic ID and toID columns.
- `status` : logical emit progress update messages?

Value

numeric stream order in same order as input
get_terminal

Examples

source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))

test_flowline <- prepare_nhdplus(walker_flowline, 0, 0, FALSE)

test_flowline <- data.frame(
  ID = test_flowline$COMID,
  toID = test_flowline$toCOMID)

(order <- get_streamorder(test_flowline))

walker_flowline$order <- order

plot(sf::st_geometry(walker_flowline), lwd = walker_flowline$order, col = "blue")

get_terminal(x, outlets)

Arguments

x two column data.frame with IDs and toIDs. Names are ignored.
outlets IDs of outlet flowlines

Value
data.frame containing the terminal ID for each outlet

Examples

source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))

fl <- dplyr::select(prepare_nhdplus(walker_flowline, 0, 0),
  ID = COMID, toID = toCOMID)

outlet <- fl$ID[which(!fl$toID %in% fl$ID)]

get_terminal(fl, outlet)

Description

Get the ID of the basin outlet for each flowline.

Usage

get_terminal(x, outlets)

Arguments

x two column data.frame with IDs and toIDs. Names are ignored.
outlets IDs of outlet flowlines

Value
data.frame containing the terminal ID for each outlet

Examples

source(system.file("extdata", "walker_data.R", package = "nhdplusTools"))

fl <- dplyr::select(prepare_nhdplus(walker_flowline, 0, 0),
  ID = COMID, toID = toCOMID)

outlet <- fl$ID[which(!fl$toID %in% fl$ID)]

get_terminal(fl, outlet)
**get_UM**

*Navigate Upstream Mainstem*

**Description**

Traverse NHDPlus network upstream main stem

**Usage**

```
get_UM(network, comid, distance = NULL, sort = FALSE, include = TRUE)
```

**Arguments**

- `network`: data.frame NHDPlus flowlines including at a minimum: COMID, Pathlength, LevelPathI, and Hydroseq.
- `comid`: integer identifier to start navigating from.
- `distance`: numeric distance in km to limit how many COMIDs are
- `sort`: if TRUE, the returned COMID vector will be sorted in order of distance from the input COMID (nearest to farthest)
- `include`: if TRUE, the input COMID will be included in the returned COMID vector returned. The COMID that exceeds the distance specified is returned.

**Value**

integer vector of all COMIDs upstream of the starting COMID along the mainstem

**Examples**

```
library(sf)

source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))

plot(sample_flines$geom)
start_COMID <- 11690196
UM_COMIDs <- get_UM(sample_flines, start_COMID)
plot(dplyr::filter(sample_flines, COMID %in% UM_COMIDs)$geom,
    col = "red", add = TRUE, lwd = 3)

UM_COMIDs <- get_UM(sample_flines, start_COMID, distance = 50)
plot(dplyr::filter(sample_flines, COMID %in% UM_COMIDs)$geom,
    col = "blue", add = TRUE, lwd = 2)
```
**get_UT**

*Navigate Upstream with Tributaries*

**Description**

Traverse NHDPlus network upstream with tributaries

**Usage**

```r
get_UT(network, comid, distance = NULL)
```

**Arguments**

- `network` - data.frame NHDPlus flowlines including at a minimum: COMID, Pathlength, LENGTHKM, and Hydroseq.
- `comid` - integer Identifier to start navigating from.
- `distance` - numeric distance in km to limit how many COMIDs are returned. The COMID that exceeds the distance specified is returned.

**Value**

integer vector of all COMIDs upstream with tributaries of the starting COMID.

**Examples**

```r
library(sf)
source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))
plot(sample_flines$geom)
start_COMID <- 11690196
UT_COMIDs <- get_UT(sample_flines, start_COMID)
plot(dplyr::filter(sample_flines, COMID %in% UT_COMIDs)$geom, col = "red", add = TRUE)

UT_COMIDs <- get_UT(sample_flines, start_COMID, distance = 50)
plot(dplyr::filter(sample_flines, COMID %in% UT_COMIDs)$geom, col = "blue", add = TRUE)
```

**get_vaa**

*NHDPlusV2 Attribute Subset*

**Description**

Return requested NHDPlusV2 Attributes
get_vaa_names

Usage

get_vaa(atts = NULL, path = get_vaa_path(), download = TRUE)

Arguments

atts character The variable names you would like, always includes comid
path character path where the file should be saved. Default is a persistent system data
as retrieved by nhdplusTools_data_dir. Also see: get_vaa_path
download logical if TRUE, the default, will download VAA table if not found at path.

Details

The VAA data is a aggregate table of information from the NHDPlusV2 elevslope.dbf(s), PlusFlowlineVAA.dbf(s); and NHDFlowlines. All data originates from the EPA NHDPlus Homepage here. To see the location of cached data on your machine use get_vaa_path. To view aggregate data and documentation, see here

Value
data.frame containing requested VAA data

Examples

## Not run:
# This will download the vaa file to the path from get_vaa_path()

get_vaa("slope")
get_vaa(c("slope", "lengthkm"))

#cleanup if desired
unlink(dirname(get_vaa_path()), recursive = TRUE)

## End(Not run)

get_vaa_names

Available NHDPlusV2 Attributes

Description

Find variables available from the NHDPlusV2 attribute data.frame

Usage

get_vaa_names()
Details

The VAA data is an aggregate table of information from the NHDPlusV2 elevslope.dbf(s), PlusFlowlineVAA.dbf(s); and NHDFlowlines. All data originates from the EPA NHDPlus Homepage here. To see the location of cached data on your machine use `get_vaa_path`. To view aggregate data and documentation, see here

Value

character vector

Examples

```r
# Not run:
get_vaa_names()

cleanup if desired
unlink(dirname(get_vaa_path()), recursive = TRUE)
```

---

**get_vaa_path**  
File path to value added attribute (vaa) Cache

Description

nhdplusTools will download and cache an ‘fst’ file with NHDPlusV2 attribute data sans geometry. This function returns the file path to the cached file. Will use the user data dir indicated by `nhdplusTools_data_dir`.

Usage

```r
get_vaa_path()
```

Details

The VAA data is an aggregate table of information from the NHDPlusV2 elevslope.dbf(s), PlusFlowlineVAA.dbf(s); and NHDFlowlines. All data originates from the EPA NHDPlus Homepage here. To see the location of cached data on your machine use `get_vaa_path`. To view aggregate data and documentation, see here

Value

character file path

Examples

```r
get_vaa_path()
```
get_waterbodies

**Find NHD Water Bodies**

**Description**

Subsets NHD waterbody features by location (POINT), area (POLYGON), or set of IDs.

**Usage**

```r
get_waterbodies(AOI = NULL, id = NULL, t_srs = NULL, buffer = 0.5)
```

**Arguments**

- **AOI** sf (MULTI)POINT or (MULTI)POLYGON. An 'area of interest' can be provided as either a location (sf POINT) or area (sf POLYGON) in any Spatial Reference System.
- **id** NHD Waterbody COMID(s)
- **t_srs** character (PROJ string or EPSG code) or numeric (EPSG code). A user specified - target - Spatial Reference System (SRS/CRS) for returned objects. Will default to the CRS of the input AOI if provided, and to 4326 for ID requests.
- **buffer** numeric. The amount (in meters) to buffer a POINT AOI by for an extended search. Default = 0.5

**Details**

The returned object(s) will have the same Spatial Reference System (SRS) as the input AOI. If a individual or set of IDs are used to query, then the default geoserver CRS of EPSG:4326 is preserved. In all cases, a user-defined SRS can be passed to `t_srs` which will override all previous SRS's (either input or default). All buffer and distance operations are handled internally using in EPSG:5070 Albers Equal Area projection

**Value**

a simple features (sf) object

get_waterbody_index

**Get Waterbody Index**

**Description**

given an sf point geometry column, return waterbody id, and COMID of dominant artificial path

**Usage**

```r
get_waterbody_index(waterbodies, points, flines = NULL, search_radius = 0.1)
```
make_standalone

**Arguments**

- `waterbodies` sf data.frame of type POLYGON or MULTIPOLYGON including COMID attributes.
- `points` sfc of type POINT
- `flines` sf data.frame of type LINESTRING or MULTILINESTRING including COMID, WBAREACOMI, and Hydroseq attributes
- `search_radius` numeric how far to search for a waterbody boundary in units of provided projection

**Value**

data.frame with two columns, COMID, in_wb_COMID, near_wb_COMID, near_wb_dist, and outlet_fline_COMID. Distance is in units of provided projection.

**Examples**

```r
source(system.file("extdata/sample_data.R", package = "nhdplusTools"))

waterbodies <- sf::read_sf(sample_data, "NHDWaterbody")

gwbody_index(waterbodies,
    sf::st_sfc(sf::st_point(c(-89.356086, 43.079943)),
    crs = 4326, dim = "XY")
```

---

**Description**

Cleans up and prepares NHDPlusHR regional data for use as complete NHDPlus data. The primary modification applied is to ensure that any flowpath that exits the domain is labeled as a terminal path and attributes are propagated upstream such that the domain is independently complete.

**Usage**

```r
make_standalone(flowlines)
```

**Arguments**

- `flowlines` sf data.frame of NHDPlusHR flowlines.

**Value**

sf data.frame containing standalone network
Examples

```
library(dplyr)
library(sf)
source(system.file("extdata/nhdplushr_data.R", package = "nhdplusTools"))

(outlet <- filter(hr_data$NHDFlowline, Hydroseq == min(Hydroseq)))
nrow(filter(hr_data$NHDFlowline, TerminalPa == outlet$Hydroseq))

hr_data$NHDFlowline <- make_standalone(hr_data$NHDFlowline)

(outlet <- filter(hr_data$NHDFlowline, Hydroseq == min(Hydroseq)))
nrow(filter(hr_data$NHDFlowline, TerminalPa == outlet$Hydroseq))

source(system.file("extdata/nhdplushr_data.R", package = "nhdplusTools"))

# Remove mainstem and non-dendritic stuff.
subset <- filter(hr_data$NHDFlowline,
                 StreamLeve > min(hr_data$NHDFlowline$StreamLeve) &
                 StreamOrde == StreamCalc)

subset <- subset_nhdplus(subset$COMID, nhdplus_data = hr_gpkg)$NHDFlowline

plot(sf::st_geometry(hr_data$NHDFlowline))

flowline_mod <- make_standalone(subset)

terminals <- unique(flowline_mod$TerminalPa)

colors <- sample(hcl.colors(length(terminals), palette = "Zissou 1"))

for(i in 1:length(terminals)) {
  fl <- flowline_mod[flowline_mod$TerminalPa == terminals[i], ]
  plot(st_geometry(fl), col = colors[i], lwd = 2, add = TRUE)
}

ol <- filter(flowline_mod, TerminalFl == 1 & TerminalPa %in% terminals)

plot(st_geometry(ol), lwd = 2, add = TRUE)
```

navigate_nldi

Navigate NLDI

Description

Navigate the Network Linked Data Index network.
navigate_nldi

Usage

navigate_nldi(
  nldi_feature,
  mode = "upstreamMain",
  data_source = "flowlines",
  distance_km = 10
)

Arguments

nldi_feature  list with names ‘featureSource’ and ‘featureID’ where ‘featureSource’ is derived from the "source" column of the response of dataRetrieval::get_nldi_sources() and the ‘featureSource’ is a known identifier from the specified ‘featureSource’.

mode          character chosen from ("UM", "UT", "DM", "DD"). See examples.

data_source   character chosen from "source" column of the response of dataRetrieval::get_nldi_sources() or empty string for flowline geometry.

distance_km   numeric distance in km to stop navigating.

Value

sf data.frame with result

Examples

library(sf)
library(dplyr)

nldi_nwis <- list(featureSource = "nwissite", featureID = "USGS-05428500")

navigate_nldi(nldi_feature = nldi_nwis,
  mode = "upstreamTributaries")$UT %>%
  st_geometry() %>%
  plot()

navigate_nldi(nldi_feature = nldi_nwis,
  mode = "UM")$UM %>%
  st_geometry() %>%
  plot(col = "blue", add = TRUE)

nwissite <- navigate_nldi(nldi_feature = nldi_nwis,
  mode = "UT",
  data_source = "nwissite")$UT_nwissite

st_geometry(nwissite) %>%
  plot(col = "green", add = TRUE)

nwissite
**nhdplusTools_data_dir**  
*get or set nhdplusTools data directory*

**Description**

if left unset, will return the user data dir as returned by `R_user_dir` for this package.

**Usage**

```r
nhdplusTools_data_dir(dir = NULL)
```

**Arguments**

- **dir** path of desired data directory

**Value**

character path of data directory (silent when setting)

**Examples**

```r
nhdplusTools_data_dir()
nhdplusTools_data_dir("demo")
nhdplusTools_data_dir(tools::R_user_dir("usgs_r/nhdplusTools"))
```

---

**nhdplus_path**  
*NHDPlus Data Path*

**Description**

Allows specification of a custom path to a source dataset. Typically this will be the national seamless dataset in geodatabase or geopackage format.

**Usage**

```r
nhdplus_path(path = NULL, warn = FALSE)
```

**Arguments**

- **path** character path ending in `.gdb` or `.gpkg`
- **warn** boolean controls whether warning an status messages are printed
Value

0 (invisibly) if set successfully, character path if no input.

Examples

nhdplus_path("/data/NHDPlusV21_National_Seamless.gdb")

nhdplus_path("/data/NHDPlusV21_National_Seamless.gdb", warn=FALSE)

nhdplus_path()

plot_nhdplus

Plot NHDPlus

Description

Given a list of outlets, get their basin boundaries and network and return a plot in EPSG:3857 Web Mercator Projection.

Usage

```
plot_nhdplus(
  outlets = NULL,
  bbox = NULL,
  streamorder = NULL,
  nhdplus_data = NULL,
  gpkg = NULL,
  plot_config = NULL,
  add = FALSE,
  actually_plot = TRUE,
  overwrite = TRUE,
  flowline_only = NULL,
  ...
)
```

Arguments

- `outlets`: list of nldi outlets. Other inputs are coerced into nldi outlets, see details.
- `bbox`: object of class bbox with a defined crs. See examples.
- `streamorder`: integer only streams of order greater than or equal will be returned
- `nhdplus_data`: geopackage containing source nhdplus data (omit to download)
- `gpkg`: path and file with .gpkg ending. If omitted, no file is written.
- `plot_config`: list containing plot configuration, see details.
- `add`: boolean should this plot be added to an already built map.
actually_plot boolean actually draw the plot? Use to get data subset only.
overwrite passed on the subset_nhdplus.
flowline_only boolean only subset and plot flowlines?
... parameters passed on to rosm.

Details

plot_nhdplus supports several input specifications. An unexported function "as_outlet" is used to convert the outlet formats as described below.

1. if outlets is omitted, the bbox input is required and all nhdplus data in the bounding box is plotted.
2. If outlets is a list of integers, it is assumed to be NHDPlus IDs (comids) and all upstream tributaries are plotted.
3. if outlets is an integer vector, it is assumed to be all NHDPlus IDs (comids) that should be plotted. Allows custom filtering.
4. If outlets is a character vector, it is assumed to be NWIS site ids.
5. if outlets is a list containing only characters, it is assumed to be a list of nldi features and all upstream tributaries are plotted.
6. if outlets is a data.frame with point geometry, a point in polygon match is performed and upstream with tributaries from the identified catchments is plotted.

The plot_config parameter is a list with names "basin", "flowline" and "outlets". The following shows the defaults that can be altered.

1. basin list(lwd = 1, col = NA, border = "black")
2. flowline list(lwd = 1, col = "blue")
3. outlets
   list(default = list(col = "black", border = NA, pch = 19, cex = 1),
        nwissite = list(col = "grey40", border = NA, pch = 17, cex = 1),
        huc12pp = list(col = "white", border = "black", pch = 22, cex = 1),
        wqp = list(col = "red", border = NA, pch = 20, cex = 1))

If adding additional layers to the plot, data must be projected to EPSG:3857 with `sf::st_transform(x, 3857)` prior to adding to the plot.

Value
data.frame plot data is returned invisibly in NAD83 Lat/Lon.

Examples

```r
options("rgdal_show_exportToProj4Warnings"="none")
plot_nhdplus("05428500")
```
plot_nhdplus("05428500", streamorder = 2)

plot_nhdplus(list(13293970, 13293750))

source(system.file("extdata/sample_data.R", package = "nhdplusTools"))

plot_nhdplus(list(13293970, 13293750), streamorder = 3, nhdplus_data = sample_data)

plot_nhdplus(list(list("comid", "13293970"),
list("nwissite", "USGS-05428500"),
list("huc12pp", "070900020603"),
list("huc12pp", "070900020602")),
streamorder = 2,
nhdplus_data = sample_data)

plot_nhdplus(sf::st_as_sf(data.frame(x = -89.36083,
y = 43.08944),
coords = c("x", "y"), crs = 4326),
streamorder = 2,
nhdplus_data = sample_data)

plot_nhdplus(list(list("comid", "13293970"),
list("nwissite", "USGS-05428500"),
list("huc12pp", "070900020603"),
list("huc12pp", "070900020602")),
streamorder = 2,
nhdplus_data = sample_data,
plot_config = list(basin = list(lwd = 2),
outlets = list(huc12pp = list(cex = 1.5),
comid = list(col = "green"))))

bbox <- sf::st_bbox(c(xmin = -89.43, ymin = 43, xmax = -89.28, ymax = 43.1),
crs = "+proj=longlat +datum=WGS84 +no_defs")

fline <- sf::read_sf(sample_data, "NHDFlowline_Network")
comids <- nhdplusTools::get_UT(fline, 13293970)

plot_nhdplus(comids)

#' # With Local Data
plot_nhdplus(bbox = bbox, nhdplus_data = sample_data)

# With downloaded data
plot_nhdplus(bbox = bbox, streamorder = 3)

# Can also plot on top of the previous!
plot_nhdplus(bbox = bbox, nhdplus_data = sample_data,
plot_config = list(flowline = list(lwd = 0.5)))
plot_nhdplus(comids, nhdplus_data = sample_data, streamorder = 3, add = TRUE,
plot_config = list(flowline = list(col = "darkblue")))
prepare_nhdplus

Prep NHDPlus Data

Description

Function to prep NHDPlus data for use by nhdplusTools functions

Usage

```r
prepare_nhdplus(
  flines,
  min_network_size,
  min_path_length,
  min_path_size = 0,
  purge_non_dendritic = TRUE,
  warn = TRUE,
  error = TRUE,
  skip_toCOMID = FALSE
)
```

Arguments

- `flines` data.frame NHDPlus flowlines including: COMID, LENGTHKM, FTYPE (or FCODE), TerminalFl, FromNode, ToNode, TotDASqKM, StartFlag, StreamOrde, StreamCalc, TerminalPa, Pathlength, and Divergence variables.
- `min_network_size` numeric Minimum size (sqkm) of drainage network to include in output.
- `min_path_length` numeric Minimum length (km) of terminal level path of a network.
- `min_path_size` numeric Minimum size (sqkm) of outlet level path of a drainage basin. Drainage basins with an outlet drainage area smaller than this will be removed.
- `purge_non_dendritic` boolean Should non dendritic paths be removed or not.
- `warn` boolean controls whether warning an status messages are printed
- `error` boolean controls whether to return potentially invalid data with a warning rather than an error
- `skip_toCOMID` boolean if TRUE, toCOMID will not be added to output.

Value

data.frame ready to be used with the refactor_flowlines function.
rescale_measures

Examples

source(system.file("extdata", "sample_flines.R", package = "nhdplusTools"))

prepare_nhdplus(sample_flines,
    min_network_size = 10,
    min_path_length = 1,
    warn = FALSE)

<table>
<thead>
<tr>
<th>rescale_measures</th>
<th>Rescale reachcode measure to comid flowline measure.</th>
</tr>
</thead>
</table>

Description

Given a reachcode measure and the from and to measure for a comid flowline, returns the measure along the comid flowline. This is a utility specific to the NHDPlus data model where many comid flowlines make up a single reachcode / reach. "Measures" are typically referenced to reaches. Flowlines have a stated from-measure / to-measure. In some cases it is useful to rescale the measure such that it is relative only to the flowline.

from is downstream – 0 is the outlet to is upstream – 100 is the inlet

Usage

rescale_measures(measure, from, to)

Arguments

<table>
<thead>
<tr>
<th>measure</th>
<th>numeric reach measure between 0 and 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>numeric flowline from-measure relative to the reach</td>
</tr>
<tr>
<td>to</td>
<td>numeric flowline to-measure relative to the reach</td>
</tr>
</tbody>
</table>

Value

numeric rescaled measure

Examples

rescale_measures(40, 0, 50)
rescale_measures(60, 50, 100)
stage_national_data  

**Stage NHDPlus National Data (deprecated)**

---

**Description**

Breaks down the national geo database into a collection of quick to access R binary files.

**Usage**

```r
stage_national_data(
  include = c("attribute", "flowline", "catchment"),
  output_path = NULL,
  nhdplus_data = NULL,
  simplified = TRUE
)
```

**Arguments**

- `include` character vector containing one or more of: "attributes", "flowline", "catchment".
- `output_path` character path to save the output to defaults to the directory of the nhdplus_data.
- `nhdplus_data` character path to the .gpkg or .gdb containing the national seamless dataset. Not required if `nhdplus_path` has been set.
- `simplified` boolean if TRUE (the default) the CatchmentSP layer will be included.

**Details**

"attributes" will save ‘NHDFlowline_Network’ attributes as a separate data.frame without the geometry. The others will save the ‘NHDFlowline_Network’ and ‘Catchment’ or ‘CatchmentSP’ (per the ‘simplified’ parameter) as sf data.frames with superfluous Z information dropped.

The returned list of paths is also added to the nhdplusTools_env as "national_data".

**Value**

list containing paths to the .rds files.

**Examples**

```r
source(system.file("extdata/sample_data.R", package = "nhdplusTools"))
stage_national_data(nhdplus_data = sample_data, output_path = tempdir())
```
**subset_nhdplus**

**Subset NHDPlus**

**Description**

Saves a subset of the National Seamless database or other nhdplusTools compatible data based on a specified collection of COMIDs. This function uses `get_nhdplus` for the "download" data source but returns data consistent with local data subsets in a subset file.

**Usage**

```r
subset_nhdplus(
  comids = NULL,
  output_file = NULL,
  nhdplus_data = NULL,
  bbox = NULL,
  simplified = TRUE,
  overwrite = FALSE,
  return_data = TRUE,
  status = TRUE,
  flowline_only = NULL,
  streamorder = NULL,
  out_prj = 4269
)
```

**Arguments**

- **comids**: integer vector of COMIDs to include.
- **output_file**: character path to save the output to defaults to the directory of the `nhdplus_data`. Not required if `nhdplus_path` has been set or the default has been adopted. See details for more.
- **nhdplus_data**: character path to the .gpkg or .gdb containing the national seamless database, a subset of NHDPlusHR, or "download" to use a web service to download NHD-PlusV2.1 data. Not relevant to the "download" option or NHDPlusHR data.
- **bbox**: object of class "bbox" as returned by `sf::st_bbox` in Latitude/Longitude. If no CRS is present, will be assumed to be in WGS84 Latitude Longitude.
- **simplified**: boolean if TRUE (the default) the CatchmentSP layer will be included.
- **overwrite**: boolean should the output file be overwritten.
- **return_data**: boolean if FALSE path to output file is returned silently otherwise data is returned in a list.
- **status**: boolean should the function print status messages.
- **flowline_only**: boolean WARNING: experimental if TRUE only the flowline network and attributes will be returned.
- **streamorder**: integer only streams of order greater than or equal will be downloaded. Not implemented for local data.
- **out_prj**: character override the default output CRS of NAD83 lat/lon (EPSG:4269)
**Details**

This function relies on the National Seamless Geodatabase or Geopackage. It can be downloaded here.

The "download" option of this function should be considered preliminary and subject to revision. It does not include as many layers and may not be available permanently.

**Value**

character path to the saved subset geopackage

**Examples**

```r
source(system.file("extdata/sample_data.R", package = "nhdplusTools"))
nhdplus_path(sample_data)
staged_nhdplus <- stage_national_data(output_path = tempdir())
sample_flines <- readRDS(staged_nhdplus$flowline)
geom_col <- attr(sample_flines, "sf_column")
plot(sample_flines[[geom_col]], lwd = 3)
start_point <- sf::st_sfc(sf::st_point(c(-89.362239, 43.090266)), crs = 4326)
plot(start_point, cex = 1.5, lwd = 2, col = "red", add = TRUE)
start_comid <- discover_nhdplus_id(start_point)
comids <- get_UT(sample_flines, start_comid)
plot(dplyr::filter(sample_flines, COMID %in% comids)[[geom_col]], add = TRUE, col = "red", lwd = 2)
output_file <- tempfile(fileext = ".gpkg")
subset_nhdplus(comids = comids, 
               output_file = output_file, 
nhdplus_data = sample_data, 
               overwrite = TRUE, 
               status = TRUE)

sf::st_layers(output_file)
catchment <- sf::read_sf(output_file, "CatchmentSP")
```
subset_rpu

plot(catchment[[attr(catchment, "sf_column")]], add = TRUE)

waterbody <- sf::read_sf(output_file, "NHDWaterbody")

plot(waterbody[[attr(waterbody, "sf_column")]],
     col = rgb(0, 0, 1, alpha = 0.5), add = TRUE)

# Cleanup temp
sapply(staged_nhdplus, unlink)
unlink(output_file)

# Download Option:
subset_nhdplus(comids = comids,
                 output_file = output_file,
                 nhdplus_data = "download",
                 overwrite = TRUE,
                 status = TRUE, flowline_only = FALSE)

sf::st_layers(output_file)

# NHDPlusHR
source(system.file("extdata/nhdplushr_data.R", package = "nhdplusTools"))

up_ids <- get_UT(hr_data$NHDFlowline, 15000500028335)

sub_gpkg <- file.path(work_dir, "sub.gpkg")
sub_nhdhr <- subset_nhdplus(up_ids, output_file = sub_gpkg,
                            nhdplus_data = hr_gpkg, overwrite = TRUE)

sf::st_layers(sub_gpkg)
names(sub_nhdhr)

plot(sf::st_geometry(hr_data$NHDFlowline), lwd = 0.5)
plot(sf::st_geometry(sub_nhdhr$NHDFlowline), lwd = 0.6, col = "red", add = TRUE)

unlink(output_file)
unlink(sub_gpkg)

---

subset_rpu  Subset by Raster Processing Unit.

Description

Given flowlines and an rpu_code, performs a network-safe subset such that the result can be used in downstream processing. Has been tested to work against the entire NHDPlusV2 domain and satisfies a number of edge cases.
Usage

subset_rpu(fline, rpu, run_make_standalone = TRUE)

Arguments

fline sf data.frame NHD Flowlines with COMID, Pathlength, LENGTHKM, and Hydroseq. LevelPathI, RPUID, ToNode, FromNode, and ArbolateSu.

rpu character e.g. "01a"

run_make_standalone boolean should the run_make_standalone function be run on result?

Value
data.frame containing subset network

Examples

source(system.file("extdata/sample_data.R", package = "nhdplusTools"))
nhdplus_path(sample_data)
staged_nhdplus <- stage_national_data(output_path = tempdir())
sample_flines <- readRDS(staged_nhdplus$flowline)
subset_rpu(sample_flines, rpu = "07b")
Index

add_plus_network_attributes, 3
adist, 6
align_nhdplus_names, 4
calculate_arbolate_sum, 5
calculate_total_drainage_area, 5
disambiguate_flowline_indexes, 6
discover_nhdplus_id, 8
discover_nldi_characteristics, 8
download_nhdplus, 9
download_nhdplusv2, 10
download_rf1, 11
download_vaa, 11
download_wbd, 12
get_DD, 13
get_DM, 14
get_flowline_index, 7, 15, 20
get_gagesII, 16
get_hr_data, 17, 24
get_huc12, 18
get_huc8, 9, 19
get_hydro_location, 19
get_levelpaths, 3, 20
get_nhdarea, 21
get_nhdplus, 22, 49
get_nhdplushr, 17, 23
get_nldi_basin, 25
get_nldi_characteristics, 26
get_nldi_feature, 27
get_nldi_sources, 8
get_node, 27
get_nwis, 28
get_pathlength, 29
get_pfaf, 30
get_streamlevel, 31
get_streamorder, 32
get_terminal, 33
get_UT, 35
get_vaa, 35
get_vaa_names, 36
get_vaa_path, 12, 36, 37, 37
get_waterbodies, 38
get_waterbody_index, 38
make_standalone, 24, 39
navigate_nldi, 40
nhdplus_path, 42, 48, 49
nhdplusTools_data_dir, 12, 36, 37, 42
plot_nhdplus, 43
prepare_nhdplus, 46
R_user_dir, 42
rescale_measures, 47
st_crs, 18
st_layers, 18
stage_national_data, 48
subset_nhdplus, 44, 49
subset_rpu, 51