Package 'nhdplusTools'

September 15, 2020

Type Package

Title NHDPlus Tools

Version 0.3.15

Description Tools for traversing and working with National Hydrography Dataset Plus (NHDPlus) 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 (>= 3.5.0)

Imports dplyr, sf, RANN, units, magrittr, jsonlite, httr, igraph, xml2, R.utils, utils, tidyr, methods, rosm, prettymapr

Suggests testthat, knitr, rmarkdown, ggmap, ggplot2, sp, lwgeom, devtools, codetools

License CC0

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

VignetteBuilder knitr

NeedsCompilation no

Author David Blodgett [aut, cre], Mike Johnson [ctb]

Maintainer David Blodgett <dblodgett@usgs.gov>

Repository CRAN

Date/Publication 2020-09-15 15:30:02 UTC

R topics documented:

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

```
align_nhdplus_names(x)
```

Arguments

x	a sf object of nhdplus flowlines

Value

a renamed sf object

Examples

```
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

```
calculate_arbolate_sum(x)
```

Arguments

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

Value

numeric with arbolate sum.

Examples

```
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))
```

---

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 discover_nldi_sources() and the ‘featureSource‘ 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", tier = "prod")
```

Arguments

type	character "all", "local", "total", or "divergence_routed".
tier	character optional "prod" or "test"

Examples

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

discover_nldi_navigation

Discover NLDI Navigation Options

Description

Discover available navigation options for a given feature source and id.

Usage

```
discover_nldi_navigation(nldi_feature, tier = "prod")
```

Arguments

nldi_feature	length 2 list list with optional names ‘featureSource‘ and ‘featureID‘ where ‘featureSource‘ is derived from the "source" column of the response of discover_nldi_sources() and the ‘featureSource‘ is a known identifier from the specified ‘featureSource‘. e.g. list("nwissite", "USGS-08279500")
tier	character optional "prod" or "test"

Value

data.frame with three columns "source", "sourceName" and "features"

Examples

```
discover_nldi_sources()

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

discover_nldi_navigation(nldi_nwis)

discover_nldi_navigation(list("nwissite", "USGS-08279500"))
```

---

discover_nldi_sources *Discover NLDI Sources*

Description

Function to retrieve available feature and data sources from the Network Linked Data Index.

Usage

```
discover_nldi_sources(tier = "prod")
```

Arguments

tier	character optional "prod" or "test"

Value

data.frame with three columns "source", "sourceName" and "features"

Examples

```
discover_nldi_sources()
```

---

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
download_files	boolean if FALSE, only URLs to files will be returned can be hu02s and/or hu04s

Value

Paths to geodatabases created.

Examples

```
download_nhdplushr(tempdir(), c("01", "0203"), download_files = FALSE)
```

---

download_nhdplusv2 *Download seamless National Hydrography Dataset Version 2 (NHDPlusV2)*

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.

Usage

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

Arguments

outdir	The folder path where data should be downloaded and extracted
url	the location of the online resource

Value

the path to the local geodatabase

Examples

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

## End(Not run)
```

download_rf1 *Download the seamless Reach File (RF1) Database*

Description

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

Usage

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

Arguments

outdir	The folder path where data should be downloaded and extracted
url	the location of the online resource

Value

the path to the local e00 file

Examples

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

## End(Not run)
```

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,
  url = paste0("https://prd-tnm.s3.amazonaws.com/StagedProducts/",
    "Hydrography/WBD/National/GDB/WBD_National_GDB.zip")
)
```

Arguments

outdir	The folder path where data should be downloaded and extracted
url	the location of the online resource

Value

the path to the local geodatabase

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
sample_flines <- read_sf(system.file("extdata",
                                     "petapsco_flowlines.gpkg",
                                     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

```
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

```
library(sf)
sample_flines <- read_sf(system.file("extdata",
                                     "petapsco_flowlines.gpkg",
                                     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.

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

```
sample_flines <- sf::read_sf(system.file("extdata",
                                         "petapsco_flowlines.gpkg",
                                         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_hr_data *Get NHDPlus HiRes Data*

Description

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

Usage

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

Arguments

gdb	character path to geodatabase to get data from.
layer	character layer name from geodatabase found with 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 st_crs
rename	boolean if TRUE, nhdplusTools standard attribute values will be applied.

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, status = FALSE)
```

Arguments

x	data.frame with ID, toID, nameID, and weight columns.
status	boolean if status updates should be printed.

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

```
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_nhdplushr *Get NHDPlus HiRes*

Description

Get NHDPlus HiRes

Usage

```
get_nhdplushr(
  hr_dir,
  out_gpkg = NULL,
  layers = c("NHDFlowline", "NHDPlusCatchment"),
  pattern = ".*GDB.gdb$",
  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
layers	character vector with desired layers to return. c("NHDFlowline", "NHDPlusCatchment") is default. Choose from: c("NHDFlowline", "NHDPlusCatchment", "NHDWaterbody", "NHDArea", "NHDLine", "NHDPlusSink", "NHDPlusWall", "NHDPoint", "NHDPlusBurnWaterbody", "NHDPlusBurnLineEvent", "HYDRO_NET_Junctions", "WBDHU2", "WBDHU4","WBDHU6", "WBDHU8" "WBDHU10", "WBDHU12", "WBDLine") Set to NULL to get all available.
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

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

Examples

```
# Note this will download a lot of data to a temp directory.
# Change 'tempdir()' to your directory of choice.
download_dir <- download_nhdplushr(tempdir(), c("0302", "0303"))

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

get_nhdplushr(download_dir,
              file.path(download_dir, "nhdplus_0302-03.gpkg"),
              layers = NULL, overwrite = TRUE)

get_nhdplushr(download_dir,
              file.path(download_dir, "nhdplus_0302-03.gpkg"),
              layers = "NHDFlowline", overwrite = TRUE,
              min_size_sqkm = 10, simp = 10, proj = "+init=epsg:5070")
```

get_nldi_basin *Get NLDI Basin Boundary*

Description

Get a basin boundary for a given NLDI feature.

Usage

```
get_nldi_basin(nldi_feature, tier = "prod")
```

Arguments

nldi_feature	list with names 'featureSource' and 'featureID' where 'featureSource' is derived from the "source" column of the response of discover_nldi_sources() and the 'featureSource' is a known identifier from the specified 'featureSource'.
tier	character optional "prod" or "test"

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

```
library(sf)
library(dplyr)

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

basin <- get_nldi_basin(nldi_feature = nldi_nwis)

basin %>%
 st_geometry() %>%
 plot()

basin
```

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

```
get_nldi_characteristics(nldi_feature, type = "local", tier = "prod")
```

Arguments

nldi_feature	list with names ‘featureSource‘ and ‘featureID‘ where ‘featureSource‘ is derived from the "source" column of the response of discover_nldi_sources() and the ‘featureSource‘ is a known identifier from the specified ‘featureSource‘.
type	character "all", "local", "total", or "divergence_routed".
tier	character optional "prod" or "test"

Examples

```
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, tier = "prod")
```

Arguments

nldi_feature	list with names ‘featureSource‘ and ‘featureID‘ where ‘featureSource‘ is derived from the "source" column of the response of discover_nldi_sources() and the ‘featureSource‘ is a known identifier from the specified ‘featureSource‘.
tier	character optional "prod" or "test"

Value

sf feature collection with one feature

Examples

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

---

get_node *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"

Examples

```
fline <- sf::read_sf(system.file("extdata/sample_natseamless.gpkg",
                                 package = "nhdplusTools"),
                     "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 *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.

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 *Get Pfafstetter Codes (Experimental)*

Description

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

Usage

```
get_pfaf(x, max_level = 2, status = FALSE)
```

Arguments

x	sf data.frame with ID, toID, totda, outletID, topo_sort, and levelpath attributes.
max_level	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.
status	boolean print status or not

Value

data.frame with ID and pfaf columns.

Examples

```
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)

pfaf <- get_pfaf(fl, max_level = 4)

hr_catchment <- left_join(hr_data$NHDPlusCatchment, pfaf, by = c("FEATUREID" = "ID"))

colors <- data.frame(pf_level_4 = unique(hr_catchment$pf_level_4),
                color = sample(terrain.colors(length(unique(hr_catchment$pf_level_4)))),
                    stringsAsFactors = FALSE)
hr_catchment <- left_join(hr_catchment, colors, by = "pf_level_4")
plot(hr_catchment["color"], border = NA, reset = FALSE)
plot(sf::st_geometry(hr_flowline), col = "blue", add = TRUE)

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

fl <- select(walker_flowline, COMID, AreaSqKM) %>%
  right_join(prepare_nhdplus(walker_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 = 2)
```

```
fl <- left_join(fl, pfaf, by = "ID")

plot(fl["pf_level_2"], lwd = 2)
```

`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

```
get_streamorder(x)
```

Arguments

`x`	data.frame with dendritic ID and toID columns.

Value

numeric stream order in same order as input

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 *Get Terminal ID*

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

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, UpHydroseq, 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)
sample_flines <- read_sf(system.file("extdata",
                                     "petapsco_flowlines.gpkg",
                                     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

```
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

```
library(sf)
sample_flines <- read_sf(system.file("extdata",
                                     "petapsco_flowlines.gpkg",
                                     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_waterbody_index *Get Waterbody Index*

Description

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

Usage

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

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

```
sample <- system.file("extdata/sample_natseamless.gpkg",
                      package = "nhdplusTools")

waterbodies <- sf::read_sf(sample, "NHDWaterbody")
get_waterbody_index(waterbodies,
                    sf::st_sfc(sf::st_point(c(-89.356086, 43.079943)),
                               crs = 4326, dim = "XY"))
```

---

make_standalone *Make isolated NHDPlusHR region a standalone dataset*

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

```
make_standalone(flowlines)
```

Arguments

flowlines	sf data.frame of NHDPlusHR flowlines.

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.

Usage

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

Arguments

`nldi_feature`	list with names ‘featureSource‘ and ‘featureID‘ where ‘featureSource‘ is derived from the "source" column of the response of discover_nldi_sources() and the ‘featureSource‘ is a known identifier from the specified ‘featureSource‘.
`mode`	character chosen from names, URLs, or url parameters returned by discover_nldi_navigation(nldi_feature) See examples.
`data_source`	character chosen from "source" column of the response of discover_nldi_sources() or empty string for flowline geometry.
`distance_km`	numeric distance in km to stop navigating.
`tier`	character optional "prod" or "test"

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") %>%
  st_geometry() %>%
  plot()

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



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

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

nwissite
```

---

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

```
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

1 if set successfully, the 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.

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))
```

Value

plot data is returned invisibly.

Examples

```
options("rgdal_show_exportToProj4_warnings"="none")
rosm::set_default_cachedir(tempfile())

plot_nhdplus("05428500")

plot_nhdplus("05428500", streamorder = 2)

plot_nhdplus(list(13293970, 13293750))
```

```
sample_data <- system.file("extdata/sample_natseamless.gpkg", 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

```
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, 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.

Examples

```
flines_in <- sf::read_sf(system.file("extdata/petapsco_flowlines.gpkg",
                                     package = "nhdplusTools"))
prepare_nhdplus(flines_in,
                min_network_size = 10,
                min_path_length = 1,
                warn = FALSE)
```

`stage_national_data` *Stage NHDPlus National Data*

Description

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

Usage

```
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

```
sample_data <- system.file("extdata/sample_natseamless.gpkg",
                           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.

Usage

```
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
)
```

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.
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 NHDPlusV2.1 data. Not required if nhdplus_path has been set or the default has been adopted. See details for more.
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. Not relevant to the "download" option or NHDPlusHR data.
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.

Details

If stage_national_data has been run in the current session, this function will use the staged national data automatically.

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 permenently.

Value

path to the saved subset geopackage

Examples

```
sample_data <- system.file("extdata/sample_natseamless.gpkg",
                           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")

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)

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?

Examples

```
sample_data <- system.file("extdata/sample_natseamless.gpkg",
                           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