Package: hydrogeofetch (via r-universe)

October 30, 2026

Type Package

Title Hydrologic Geospatial Fabric Extraction Tool Chain

Version 2.0.5

Description Traverses and works with National Hydrography Dataset Plus (NHDPlus) data. All methods implemented in 'hydrogeofetch' are available in the NHDPlus documentation available from the US Environmental Protection Agency <https://www.epa.gov/waterdata/basic-information>. Previously published as 'nhdplusTools'.

URL https://doi-usgs.github.io/hydrogeofetch/
https://github.com/DOI-USGS/hydrogeofetch/

BugReports https://github.com/DOI-USGS/hydrogeofetch/issues/

Depends R (>= 4.1)

Imports hydroloom (>= 1.3.0), dataRetrieval, dplyr, sf, units, jsonlite, httr2, xml2, utils, data.table, methods, arrow, tools, zip, memoise (>= 2.0.0), digest

Suggests testthat, knitr, rmarkdown, ggmap, ggplot2, lwgeom, gifski, leaflet, httptest2, StreamCatTools, terra, maptiles, mapsf

License CC0

Encoding UTF-8

VignetteBuilder knitr

Config/testthat/parallel true

Config/testthat/edition 3

LazyData true

LazyDataCompression xz

Language en-US

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author David Blodgett [aut, cre] (ORCID: <https://orcid.org/0000-0001-9489-1710>), Mike Johnson [ctb] (ORCID: <https://orcid.org/0000-0002-5288-8350>), Marc Weber [ctb] (ORCID: <https://orcid.org/0000-0002-9742-4744>), Josh Erickson [ctb], Lauren Koenig [ctb] (ORCID: <https://orcid.org/0000-0002-7790-330X>)

Maintainer David Blodgett <dblodgett@usgs.gov>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-30 19:33:55 UTC

RemoteUrl https://github.com/cran/hydrogeofetch

RemoteRef HEAD

RemoteSha 2dd05207dace13aa925b333c01eab930e3df43c2

Contents

add_mainstems . . . 3
align_nhdplus_names . . . 4
check_mainstems . . . 5
discover_geoconnex_reference . . . 6
discover_nhdplus_id . . . 6
download_nhd . . . 7
download_nhdplushr . . . 8
download_nhdplusv2 . . . 9
download_rf1 . . . 10
download_vaa . . . 10
download_wbd . . . 11
get_3dhp . . . 12
get_3dhp_index . . . 14
get_boundaries . . . 15
get_catchment_characteristics . . . 16
get_characteristics_metadata . . . 17
get_drainage_area_estimates . . . 18
get_elev_along_path . . . 21
get_gagesII . . . 22
get_geoconnex_reference . . . 23
get_hr_data . . . 24
get_huc . . . 25
get_mainstem_flowlines . . . 26
get_nhdarea . . . 27
get_nhdphr . . . 28
get_nhdplus . . . 29
get_nhdplushr . . . 31
get_nldi_basin . . . 32
get_nldi_feature . . . 34
get_nldi_index . . . 34
get_nwis . . . 35

get_raindrop_trace . . . 36
get_split_catchment . . . 37
get_tocomid . . . 38
get_vaa . . . 40
get_vaa_names . . . 41
get_vaa_path . . . 42
get_waterbodies . . . 42
get_wb_outlet . . . 43
get_xs_point . . . 44
get_xs_points . . . 45
hydrogeofetch_cache_clear . . . 46
hydrogeofetch_cache_settings . . . 47
hydrogeofetch_data_dir . . . 47
make_standalone . . . 48
map_nhdplus . . . 49
navigate_nldi . . . 51
nhdplus_path . . . 52
plot_nhdplus . . . 53
rescale_catchment_characteristics . . . 56
rpu_boundaries . . . 58
subset_nhdplus . . . 58
subset_rpu . . . 61
subset_vpu . . . 62
translate_flowlines . . . 63
translate_hydrolocations . . . 64
update_mainstems . . . 66
vpu_boundaries . . . 67

Index **68**

`add_mainstems` *Add mainstem identifiers*

Description

Joins geoconnex mainstem identifiers onto a table that contains NHDPlusV2 (comid/featureid) or NHDPlusHR (nhdplusid) identifiers using a lookup table.

Usage

```
add_mainstems(x, join_col = NULL, join_col_type = NULL)
```

Arguments

x	data.frame or sf containing an identifier column joinable to NHDPlusV2 or NHDPlusHR.
join_col	character name of the identifier column in x. Detected automatically from comid, featureid, or nhdplusid (any case) if not provided.
join_col_type	character one of "nhdpv2" or "nhdphr". Detected automatically from join_col if not provided.

Details

Source data comes from the ref_rivers GitHub release here. The csv source data is downloaded once, converted to parquet, and cached in the user data dir indicated by hydrogeofetch_data_dir.

The NHDPlusV2 lookup csv is roughly 120 MB and the NHDPlusHR lookup roughly 50 MB, so the mainstem functions have no examples. The first call for a given type downloads and converts the table; later calls in the same session read a cached parquet file and are fast.

```
old_dir <- hydrogeofetch_data_dir()
hydrogeofetch_data_dir(file.path(tempdir(), "mainstems"))

add_mainstems(data.frame(comid = c(2804607, 2804621)))

hydrogeofetch_data_dir(old_dir)
```

Value

x with mainstem_uri and mainstemid columns added.

---

align_nhdplus_names *Align NHD Dataset Names*

Description

this function takes any NHDPlus dataset and aligns the attribute names with those used in hydrogeofetch.

Usage

```
align_nhdplus_names(x)
```

Arguments

x	a sf object of nhdplus flowlines

Value

data.frame renamed sf object

Examples

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

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)
```

---

check_mainstems *Check mainstem identifiers for supersession*

Description

Checks whether geoconnex mainstem identifiers have been superseded by a newer reference release. See update_mainstems to resolve a superseded mainstem id to its replacement.

Usage

```
check_mainstems(x)
```

Arguments

x	integer or character vector of geoconnex `ref/mainstems` ids, or full mainstem uri (any namespace, e.g. `ref/mainstems` or `usgs/mainstems`) vector of mainstem identifiers to check. Bare integers/character ids are assumed to be `ref/mainstems` ids; a full uri is matched literally, so ids minted under a different namespace are never confused with a `ref/mainstems` id that happens to share the same trailing number.

Value

logical vector the same length as `x`. `TRUE` indicates the mainstem id has been superseded.

Examples

```
check_mainstems(c(2086165, 2086637))
```

discover_geoconnex_reference

discover geoconnex reference feature layers

Description

Queries the geoconnex.us reference feature server for available layers and attributes.

Usage

```
discover_geoconnex_reference()
```

Value

data.frame containing layers available and fields that are available to query.

Examples

```
discover_geoconnex_reference()
```

discover_nhdplus_id

Discover NHDPlus ID

Description

Multipurpose function to find a COMID of interest.

Note that NHDPlusV2 uses "featureid" for catchment polygons and "comid" for flowline linestrings. These two identifiers are the same where a flowline/catchment pair exists. In some cases, a catchment will not have a flowline and in others, a flowline will not have a catchment.

If a point is provided, and raindrop is false, the comid/featureid integer of the catchment the point is in is returned. This options uses a web service here: https://api.water.usgs.gov/fabric/pygeoapi/collections/catchmentsp

If a point is provided, and raindrop is true, the response is the result of a call to get_raindrop_trace.

If no point is provided, the raindrop argument is ignored and the result is a comid integer derived from a call to get_nldi_feature.

Usage

```
discover_nhdplus_id(point = NULL, nldi_feature = NULL, raindrop = FALSE)
```

Arguments

point	sfc 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 ‘featureSource‘ is a known identifier from the specified ‘featureSource‘.
raindrop	logical if TRUE will call a raindrop trace web service and return will be the same as get_raindrop_trace with direction "none".

Value

integer COMID or list containing COMID and raindrop trace.

Examples

```
point <- sf::st_sfc(sf::st_point(c(-76.874, 39.482)), crs = 4326)
discover_nhdplus_id(point)

discover_nhdplus_id(point, raindrop = TRUE)

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

---

download_nhd *Download NHD*

Description

Download NHD

Usage

```
download_nhd(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_huc 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 <- get_huc(sf::st_sfc(sf::st_point(c(-73, 42)), crs = 4326),
                            type = "huc08")

(hu <- substr(hu$huc8, 1, 2))

download_nhd(tempdir(), c(hu, "0203"), download_files = FALSE)
```

`download_nhdplushr` *Download NHDPlus HiRes*

Description

Download NHDPlus HiRes

Usage

```
download_nhdplushr(nhd_dir, hu_list, download_files = TRUE, archive = FALSE)
```

Arguments

`nhd_dir`	character directory to save output into
`hu_list`	character vector of hydrologic region(s) to download. Use get_huc 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
`archive`	pull data from the "archive" folder rather than "current". The archive contains the original releases of NHDPlusHR data that were updated in subsequent processing. Not all subsets of NHDPlusHR were updated. See: https://www.usgs.gov/national-hydrography/access-national-hydrography-products for more details.

Value

character Paths to geodatabases created.

Examples

```
hu <- get_huc(sf::st_sfc(sf::st_point(c(-73, 42)), crs = 4326),
                            type = "huc08")
if(inherits(hu, "sf")) {
(hu <- substr(hu$huc8, 1, 2))

download_nhdplushr(tempdir(), c(hu, "0203"), download_files = FALSE)

download_nhdplushr(tempdir(), c(hu, "0203"), download_files = FALSE, archive = TRUE)
}
```

`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.

Default downloads lower-48 only. Pass the island archive URL to 'url' to get Hawaii, Puerto Rico, the Virgin Islands, and the Pacific Islands instead. No Alaska data are available.

The lower-48 archive is roughly 8 GB and extraction needs 7zip installed, so this function has no example. Pass any writable directory as `outdir`; the archive is downloaded there, extracted in place, and the path to the geodatabase returned:

```
download_nhdplusv2(file.path(tempdir(), "nhdplusv2"))
```

Usage

```
download_nhdplusv2(
  outdir,
  url = paste0("https://dmap-data-commons-ow.s3.amazonaws.com/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

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.

The archive is roughly 46 MB and the server is slow, so this function has no example. Pass any writable directory as `outdir`; the gzipped e00 is downloaded there, decompressed in place, and the path to the e00 returned:

```
download_rf1(file.path(tempdir(), "rf1"))
```

Usage

```
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

download_vaa *Download NHDPlusVAA data from HydroShare*

Description

downloads and caches NHDPlusVAA data on your computer

Usage

```
download_vaa(
  path = get_vaa_path(updated_network),
  force = FALSE,
  updated_network = FALSE
)
```

Arguments

path	character path where the file should be saved. Default is a persistent system data as retrieved by hydrogeofetch_data_dir. Also see: get_vaa_path
force	logical. Force data re-download. Default = FALSE
updated_network	logical default FALSE. If TRUE, updated network attributes from E2NHD and National Water Model retrieved from doi:10.5066/P976XCVT.

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

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.

The national archive is roughly 3 GB, so this function has no example. The "hydrogeofetch Data Access Overview" article works through a download in its Watershed Boundary Dataset section.

Usage

```
download_wbd(
  outdir,
  url = paste0("https://prd-tnm.s3.amazonaws.com/StagedProducts/",
    "Hydrography/WBD/National/GDB/WBD_National_GDB.zip"),
  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_3dhp` *Get 3DHP Data*

Description

Calls the 3DHP_all web service and returns sf data.frames for the selected layers. See https://3dhp.nationalmap.gov/arcgis/res
for source data documentation.

Usage

```
get_3dhp(
  AOI = NULL,
  ids = NULL,
  type = NULL,
  universalreferenceid = NULL,
  t_srs = NULL,
  buffer = 0.5,
  page_size = 2000
)
```

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.
`ids`	character vector of id3dhp ids, mainstem uris, or workunitid prefixed ids (e.g. "workunitid:300585")
`type`	character. Type of feature to return. e.g. ("hydrolocation", "flowline", "waterbody", "drainage area", "catchment"). If NULL (default) a data.frame of available types is returned
`universalreferenceid`	character vector of hydrolocation universal reference ids such as reachcodes
`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
`page_size`	numeric default number of features to request at a time. Reducing may help if 500 errors are experienced.

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 CRS of EPSG:4269 is preserved. In all cases, a user-defined SRS can be passed to `t_srs` which will override all previous SRS (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 or valid types if no type supplied

Examples

```
AOI <- sf::st_as_sfc(sf::st_bbox(c(xmin = -89.56684, ymin = 42.99816,
                                   xmax = -89.24681, ymax = 43.17192),
                                 crs = "+proj=longlat +datum=WGS84 +no_defs"))

# get flowlines and hydrolocations
flowlines <- get_3dhp(AOI = AOI, type = "flowline")
hydrolocation <- get_3dhp(AOI = AOI, type = "hydrolocation")
waterbody <- get_3dhp(AOI = AOI, type = "waterbody")

if(!is.null(waterbody) & !is.null(flowlines) & !is.null(hydrolocation)) {
plot(sf::st_geometry(waterbody), col = "lightblue", border = "lightgrey")
plot(sf::st_geometry(flowlines), col = "blue", add = TRUE)
plot(sf::st_geometry(hydrolocation), col = "grey", pch = "+", add = TRUE) }

# given mainstem ids from any source, can query for them in ids.

SU <- get_3dhp(ids = "https://geoconnex.us/ref/mainstems/194408",
               type = "flowline")

if(!is.null(SU))
  plot(sf::st_geometry(SU), col = "blue")

# get all the waterbodies along the Susquehanna river
SU_wb <- get_3dhp(ids = unique(SU$waterbodyid3dhp), type = "waterbody")

if(!is.null(SU_wb)) {
plot(sf::st_geometry(SU_wb[grepl("Otsego", SU_wb$gnisidlabel),]),
     col = "blue", border = "NA") }

# given a workunitid, can query for all features in that work unit with
# ids = "workunitid:300585" -- not run here as a work unit is tens of
# thousands of features.

# given universalreferenceid (reachcodes), can query for them but only
# for hydrolocations. This is useful for looking up mainstem ids.

if(!is.null(hydrolocation)) {
get_3dhp(universalreferenceid =
           head(unique(hydrolocation$universalreferenceid), 5),
```

```
          type = "hydrolocation")
}
```

`get_3dhp_index` *Index Points to the 3DHP Network*

Description

Addresses (snaps) point features to the 3D Hydrography Program (3DHP) network using the HydroAdd3D web service. For each input point the service determines the nearest 3DHP flowline, the snapped location and its elevation, the mainstem of that flowline, the measure along the mainstem, and associated GNIS attributes.

Usage

```
get_3dhp_index(points, t_srs = NULL, convert_missing = TRUE, batch_size = 100)
```

Arguments

`points`	sf data.frame of POINT features to address. Must have a coordinate reference system with a resolvable EPSG code.
`t_srs`	character or object compatible with st_crs. Target coordinate reference system of the returned features. Defaults to the coordinate reference system of 'points'.
`convert_missing`	logical. If 'TRUE' (default), the service no-data values ('zsnap = -9999', 'm = -1', 'snapdistance = -1') are converted to 'NA' and a logical 'snapped' column flags points that addressed successfully. If 'FALSE', no data values are returned as-is.
`batch_size`	integer number of points sent to the service per request. Default 100.

Details

Points are sent to the service in batches. The 'mainstemid' is a geoconnex mainstem uri (e.g. 'https://geoconnex.us/ref/mainstems/312091') that can be passed to get_3dhp or used with the functions in add_mainstems.

See https://apps.usgs.gov/hydroadd3d for the web service.

Value

sf data.frame of POINT (XYZ) features in 't_srs', one row per input point. The geometry is the snapped location on the 3DHP flowline: the X and Y coordinates are the snapped position and the Z coordinate is the snapped elevation in meters. Points that did not snap carry their original input location as the geometry. Attribute columns:

- 'source_id' - input row, echoed for joins
- 'zsnap' - snapped elevation in meters (the geometry Z coordinate)

- ‘mainstemid‘ - geoconnex mainstem uri
- ‘m‘ - measure along the mainstem, 0 = downstream, 100 = upstream
- ‘gnisidlabel‘ / ‘gnisid‘ - GNIS name and id
- ‘featuredate‘ - date the 3DHP feature was loaded
- ‘snapdistance‘ - distance in meters from the original to the snapped point
- ‘snapdate‘ - timestamp of the addressing run
- ‘snapped‘ - logical, added when ‘convert_missing = TRUE‘, flagging points that addressed successfully

Examples

```
points <- sf::st_sf(
  id = c(1, 2, 3),
  geometry = sf::st_sfc(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))

indexed <- get_3dhp_index(points)

if(inherits(indexed, "sf")) {

# points that did not snap to a flowline
indexed[!indexed$snapped, ]

# pull the flowline a point was addressed to
get_3dhp(ids = indexed$mainstemid[indexed$snapped][1], type = "flowline")

}
```

`get_boundaries` *Return RPU or VPU boundaries*

Description

Return RPU or VPU boundaries

Usage

```
get_boundaries(type = "vpu")
```

Arguments

`type`	character. Either "RPU" or "VPU"

Value

An object of class `"sf"`

`get_catchment_characteristics` *Get Catchment Characteristics*

Description

Downloads (subsets of) catchment characteristics from a cloud data store. See get_characteristics_metadata for available characteristics.

When `source = "usgs"` (the default), data is retrieved from: Wieczorek, M.E., Jackson, S.E., and Schwarz, G.E., 2018, Select Attributes for NHDPlus Version 2.1 Reach Catchments and Modified Network Routed Upstream Watersheds for the Conterminous United States (ver. 3.0, January 2021): U.S. Geological Survey data release, doi:10.5066/F7765D7V.

When `source = "streamcat"`, data is retrieved from the EPA StreamCat dataset via the `StreamCatTools` package (must be installed separately). The `aoi` parameter controls the area of interest for StreamCat queries.

Usage

```
get_catchment_characteristics(
  varname,
  ids,
  reference_fabric = "nhdplusv2",
  source = "usgs",
  aoi = "cat"
)
```

Arguments

`varname`	character vector of desired variables. If repeated varnames are provided, they will be downloaded once but duplicated in the output. For `source = "streamcat"`, use StreamCat metric names (see `get_characteristics_metadata(source = "streamcat")`).
`ids`	numeric vector of identifiers (comids) from the specified fabric
`reference_fabric`	(not used) will be used to allow future specification of alternate reference fabrics
`source`	character `"usgs"` (default) or `"streamcat"`.
`aoi`	character area of interest for StreamCat queries. One of `"cat"` (local catchment, default), `"ws"` (total upstream watershed), `"catrp100"` (catchment riparian 100m buffer), `"wsrp100"` (watershed riparian 100m buffer), or `"other"` (for metrics like BankfullDepth, IWI, etc.). Ignored when `source = "usgs"` where the area of interest is encoded in the variable name prefix (e.g. CAT_, TOT_, ACC_).

Value

data.frame with columns characteristic_id, comid, characteristic_value, and percent_nodata. NULL if no requested variables were found or the data store is unavailable.

Examples

```
get_catchment_characteristics("CAT_BFI", c(5329343, 5329427))
```

`get_characteristics_metadata` *Get catchment characteristics metadata table*

Description

Download and cache table of catchment characteristics.

When `source = "usgs"` (the default), returns metadata from: Wieczorek, M.E., Jackson, S.E., and Schwarz, G.E., 2018, Select Attributes for NHDPlus Version 2.1 Reach Catchments and Modified Network Routed Upstream Watersheds for the Conterminous United States (ver. 3.0, January 2021): U.S. Geological Survey data release, doi:10.5066/F7765D7V.

When `source = "streamcat"`, returns metric metadata from the EPA StreamCat dataset accessed via the `StreamCatTools` package (must be installed separately): Weber, Marc H, Hill, Ryan A., Brookes, Allen F. 2024, StreamCatTools: Tools to work with the StreamCat API within R and access the full suite of StreamCat and LakeCat metrics, https://usepa.github.io/StreamCatTools

Usage

```
get_characteristics_metadata(search, source = "usgs", cache = TRUE)
```

Arguments

`search`	character string of length 1 to free search the metadata table. If no search term is provided the entire table is returned.
`source`	character `"usgs"` (default) or `"streamcat"`.
`cache`	logical should cached metadata be used?

Value

data.frame of characteristic metadata with columns ID, description, units, datasetLabel, datasetURL, themeLabel, themeURL, watershedType, sbid, end, s3_url, and http_url. NULL if the metadata service is unavailable.

Examples

```
# the metadata table is cached in hydrogeofetch_data_dir(); point it at a
# temporary directory so this example does not write to user space.
old_dir <- hydrogeofetch_data_dir()
hydrogeofetch_data_dir(file.path(tempdir(check = TRUE), "meta_demo"))

get_characteristics_metadata()

hydrogeofetch_data_dir(old_dir)
```

get_drainage_area_estimates *Estimate drainage area from HUC and catchment data*

Description

Combines HUC12 areas upstream of HUC outlets with NHDPlusV2 catchment areas for the portion of the basin between the outlet and HUC12 outlets to produce a drainage area estimate. Non-contributing areas captured in HUC12 boundaries are included.

Usage

```
get_drainage_area_estimates(
  start,
  catchments = FALSE,
  nhdplushr = TRUE,
  local_navigation = FALSE,
  huc12_data = NULL,
  huc12_outlets = NULL,
  waterbody_data = NULL,
  catchment_data = NULL,
  HU_inclusion_override = NULL,
  outlet_split_threshold_m = 100
)
```

Arguments

`start`	list with `featureSource` and `featureID` compatible with `get_nldi_feature`. May also include optional `reachcode` and `measure` fields; when present these override the NLDI-supplied values used to position the outlet-catchment split point, without affecting upstream network navigation. NLDI can be bypassed entirely by passing only `reachcode` and `measure` (no `featureSource`/`featureID`); in that case the start COMID is resolved from the reachcode via the VAA when `local_navigation = TRUE`, otherwise via the NHD OGC API, and the start point geometry is computed from the flowline at the supplied measure.
`catchments`	logical. If TRUE, fetch and return NHDPlusV2 catchment polygons for the full upstream network. Default FALSE.
`nhdplushr`	logical. If TRUE (the default), compute a drainage area estimate from NHDPlusHR catchments. Set to FALSE to skip this step, which avoids the HR web service calls and speeds up computation.
`local_navigation`	logical. If TRUE, use `get_vaa` for network navigation and flowline attributes instead of NLDI/OGC API web services. Only HUC12 pour points are fetched from the NLDI. Default FALSE.

`huc12_data` sf data.frame or NULL. In-memory HUC12 polygon table to use instead of fetching from web services. Column names are lowercased internally; must include at minimum `huc_12` and `ncontrb_a` (case-insensitive). When provided, all HUC12 polygon queries are resolved by subsetting this table. Default NULL (use web services).

`huc12_outlets` sf data.frame, character path, or NULL. HUC12 pour points to use instead of the NLDI `huc12pp` service. Accepts either a preloaded sf data.frame or a path to a GPKG, which is read with `read_sf`. Columns `COMID` and `FinalWBD_HUC12` are renamed to `comid` and `identifier`; any other columns are preserved. When provided, the outlets are filtered to the upstream network COMID set and no NLDI `huc12pp` queries are issued. Pair with `local_navigation = TRUE` and `huc12_data` to run fully offline. National CONUS outlets GPKGs are available from Blodgett, D.L., 2022, Mainstem Rivers of the Conterminous United States (ver. 3.0, February 2026): U.S. Geological Survey data release, doi:10.5066/P13LNDDQ (layer `hu_points`). Default NULL.

`waterbody_data` sf data.frame or NULL. In-memory NHDWaterbody polygon table to use instead of fetching from web services. Column names are lowercased internally; must include `comid` (case-insensitive). When provided and `start` is an `sfc_POINT`, the waterbody containing the point is resolved by spatial filter on this table. Default NULL (use web services).

`catchment_data` sf data.frame or NULL. In-memory NHDPlusV2 CatchmentSP polygon table. Column names are lowercased internally; must include `featureid` (case-insensitive). Used in three ways: (1) when `catchments = TRUE`, catchment polygons for the upstream network are subsetted from this table instead of fetched from the OGC API; (2) the gap-zone catchment polygons (between the outlet and the immediate HUC12 outlets) are subsetted from this table instead of fetched; (3) when `start` is an `sfc_POINT` and no waterbody is found, the COMID is resolved by spatial join on this table. Default NULL (use web services).

`HU_inclusion_override` character vector or NULL. Eight- or ten-digit HUC codes whose HUC12s should be kept even when the parent HUC outlet is not in the on-network set. Useful for regions like the Prairie Potholes where landscape-connected HUCs lack a network outlet (e.g. `"10130106"` in South Dakota). Default NULL (no overrides).

`outlet_split_threshold_m` numeric. Minimum distance in meters from the gage to the outlet of its catchment before the catchment is split. When the gage is at least this far upstream, the outlet catchment is split at the gage point and only the upstream portion is included. Default 100.

Details

By default, network navigation is performed via the NLDI web service and flowline attributes are retrieved from the NHDPlusV2 OGC API. When `local_navigation = TRUE`, the NHDPlusV2 Value Added Attributes (`get_vaa`) are used for network navigation instead, and only HUC12 pour points are fetched from the NLDI.

HUC drainage area is used upstream of the nearest HUC12 outlet. Between the outlet (e.g. gage) and the HUC outlet(s), catchment areas are used. For large upstream areas the largest HUC level is

used to define connectedness, but drainage estimates are derived from HUC12 because that is where the non-contributing area attribute lives.

Three pairs of drainage area estimates are returned: one using only NLDI-identified HUC12s (HUC12-level), one using HUC10-level queries, and one using HUC08-level queries for basins spanning multiple HUC08s. Each pair includes a total and a contributing-only estimate derived from the `ncontrb_a` (non-contributing acres) attribute on HUC12 features.

Value

list with elements, or NULL with a warning when a required web service is unavailable:

da_huc12_sqkm numeric. Total DA using NLDI-identified HUC12s only.

da_huc10_sqkm numeric or NA. Total DA using HUC10-level queries. NA when basin is within a single HUC10.

da_huc08_sqkm numeric or NA. Total DA using HUC08-level queries. NA when basin is within a single HUC08.

contrib_da_huc12_sqkm numeric. Contributing DA (HUC12-only).

contrib_da_huc10_sqkm numeric or NA. Contributing DA (HUC10-level).

contrib_da_huc08_sqkm numeric or NA. Contributing DA (HUC08-level).

network_da_sqkm numeric. Network-derived total DA for comparison.

basin_da_sqkm numeric or NA. Area of `drainage_basin`. Not the same quantity as `network_da_sqkm`: that one is the NHDPlusV2 `totdasqkm` attribute for the outlet flowline and covers the whole outlet catchment, while this is a geometric area with the portion below the submitted point removed. The two differ by that sliver plus the usual drift between attribute and geometry. NA when `drainage_basin` is NULL.

nhdplushr_network_dasqkm numeric or NA. Drainage area from NHDPlusHR catchments upstream of the matched HR flowline. NA with a warning when the HR web service is unavailable or fails.

nhdplushr_boundary sfc_GEOMETRY or NULL. Dissolved boundary of upstream NHDPlusHR catchments. NULL when the HR estimate is unavailable.

start_feature sf data.frame. The resolved NLDI start feature.

hu12_by_huc12 sf data.frame. NLDI-identified upstream HUC12 polygons (EPSG:5070).

hu12_by_huc10 sf data.frame or NULL. Upstream HUC12 polygons (HUC10 query). NULL when basin is within a single HUC10.

hu12_by_huc08 sf data.frame or NULL. Upstream HUC12 polygons (HUC08 query). NULL when basin is within a single HUC08.

extra_catchments sf data.frame. Catchments between outlet and HUC12 outlets.

split_catchment sf data.frame. Split catchment at HUC12 outlet(s).

drainage_basin sf data.frame or NULL. NHDPlusV2 catchments for the upstream network dissolved into a single boundary (EPSG:5070, with a `dasqkm` column), with the outlet catchment replaced by the portion above the submitted point. No HUC boundaries are involved. Available when `catchments = TRUE` or `catchment_data` was supplied, and always for headwater starts, where the upstream network is a single catchment; NULL otherwise. Also NULL when a split of the outlet catchment was due but the service returned none, since the basin would otherwise carry area below the submitted point.

all_network data.frame. Full upstream flowline attributes.

all_catchments sf data.frame or NULL. NHDPlusV2 catchment polygons for the full upstream network. NULL when `catchments = FALSE`.

outlet_flowline_measure numeric or NULL. Flowline measure (0–100) for the gage on its outlet flowline. NULL for non-gage starts.

outlet_split_catchment sf data.frame or NULL. Split catchment at the gage point. Contains "catchment" and "splitCatchment" rows with areas. NULL when no split is needed (gage at outlet or below threshold).

hu12_outlet sf data.frame. HUC12 pour points upstream of outlet.

Examples

```
# Black Earth Creek
start <- list(featureSource = "nwissite", featureID = "USGS-05406500")
result <- get_drainage_area_estimates(start)
result$da_huc10_sqkm
result$network_da_sqkm
```

`get_elev_along_path` *Get Elevation Along Path (experimental)*

Description

Uses a cross section retrieval web services to retrieve elevation along a path.

Usage

```
get_elev_along_path(points, num_pts, res = 1, status = TRUE)
```

Arguments

`points`	sf data.frame containing a point column.
`num_pts`	numeric number of points to retrieve along the cross section.
`res`	integer resolution of 3D Elevation Program data to request. Must be one of: 1, 3, 5, 10, 30, 60.
`status`	logical

Value

sf data.frame containing points retrieved. Names include "id", "distance_m", "elevation_m", "spatial_ref", "geometry", and ".group". .group tracks which input point each set of output points belongs to.

Examples

```
point1 <- sf::st_sfc(sf::st_point(x = c(-105.9667, 36.17602)), crs = 4326)
point2 <- sf::st_sfc(sf::st_point(x = c(-105.97768, 36.17526)), crs = 4326)
point3 <- sf::st_sfc(sf::st_point(x = c(-105.98869, 36.17450)), crs = 4326)

points <- sf::st_as_sf(c(point1, point2, point3))

(xs <- get_elev_along_path(points, 100))

if(inherits(xs, "sf")) {

bbox <- sf::st_bbox(xs) + c(-0.005, -0.005, 0.005, 0.005)

hydrogeofetch::plot_nhdplus(bbox = bbox, cache_data = FALSE)

plot(sf::st_transform(sf::st_geometry(xs), 3857), pch = ".", add = TRUE, col = "red")
plot(sf::st_transform(sf::st_sfc(point1, crs = 4326), 3857), add = TRUE)
plot(sf::st_transform(sf::st_sfc(point2, crs = 4326), 3857), add = TRUE)
plot(sf::st_transform(sf::st_sfc(point3, crs = 4326), 3857), add = TRUE)

plot(xs$distance_m, xs$elevation_m)
}
```

---

get_gagesII *Find gagesII Features*

Description

Subsets the gagesII dataset by location (POINT), area (POLYGON), or set of IDs. See <doi:10.5066/P96CPHOT> for documentation of source data.

Usage

```
get_gagesII(AOI = NULL, id = NULL, t_srs = NULL, buffer = 0.5, basin = FALSE)
```

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	character 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
basin	logical should the gagesII basin also be returned? If True, return value will be a list with "site" and "basin" elements.

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 server 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 (either input or default). All buffer and distance operations are handled internally using an EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object, or a data.frame when `skip_geometry = TRUE`

`get_geoconnex_reference`

get geoconnex reference feature layers

Description

Queries the geoconnex reference feature server for features of interest.

Usage

```
get_geoconnex_reference(
  AOI,
  type = NULL,
  t_srs = NULL,
  buffer = 0.5,
  status = TRUE
)
```

Arguments

AOI	bbox, sf polygon or point, or a URL that will return an sf object when passed to read_sf
type	character the feature type to query. Must match one of the ‘id‘ entries returned by discover_geoconnex_reference (the ‘type‘ column of that output is not a valid value here).
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
status	boolean print status or not

Value

sf data.frame containing requested reference features

Examples

```
dplyr::distinct(discover_geoconnex_reference()[c("id", "title")])

AOI <- sf::st_as_sfc(sf::st_bbox(c(xmin = -89.56684, ymin = 42.99816,
                                   xmax = -89.24681, ymax = 43.17192),
                                 crs = "+proj=longlat +datum=WGS84 +no_defs"))

get_geoconnex_reference(AOI, type = "hu04")

get_geoconnex_reference("https://geoconnex.us/ref/mainstems/315626", type = "hu04", )

AOI <- sf::st_sfc(sf::st_point(c(-89.56684, 42.99816)),
                  crs = "+proj=longlat +datum=WGS84 +no_defs")

get_geoconnex_reference(AOI, type = "hu04", buffer = 100000)
```

---

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, hydrogeofetch standard attribute values will be applied.

Value

sf data.frame containing requested data

`get_huc` *Find WBD HUC unit subsets*

Description

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

Usage

```
get_huc(AOI = NULL, id = NULL, t_srs = NULL, buffer = 0.5, type = 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.
`id`	WBD HUC 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
`type`	character. Type of feature to return. If 'NULL' (default) and 'id' is provided, the HUC level is autodetected from the character length of the IDs (e.g. 2-character IDs → 'huc02', 12-character → 'huc12'). A version suffix alone (e.g. '"_2020"', '"_nhdplusv2"') can also be provided to combine autodetected HUC level with a specific version. If 'NULL' and no 'id' is provided, defaults to 'huc12'. Bare types ('huc02'-'huc12') default to the 2025 WBD version. Versioned types are also available with suffixes '_2025', '_2020', '_nhdplusv2', and '_nhdplushr' (e.g. 'huc12_nhdplusv2'). See https://api.water.usgs.gov/fabric/pygeoapi for the web service.

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 server 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 (either input or default). All buffer and distance operations are handled internally using an EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object, or a data.frame when `skip_geometry = TRUE`

`get_mainstem_flowlines` *Get the flowlines of mainstems*

Description

Returns the flowlines that realize one or more geoconnex mainstems in NHDPlusV2, NHDPlusHR, or 3DHP. The result carries the identifier, topology, and length attributes that hy recognizes, with `mainstemid` holding the full mainstem uri, so it can go directly to the hydroloom functions that work on aggregate flowpaths such as get_confluences and index_points_to_lines with `aggregate_ids`.

Usage

```
get_mainstem_flowlines(
  mainstems,
  network = c("nhdpv2", "nhdphr", "3dhp"),
  include_confluences = FALSE
)
```

Arguments

`mainstems`	character vector of geoconnex mainstem uris, or `ref/mainstems` integer ids.
`network`	character one of "nhdpv2", "nhdphr", or "3dhp".
`include_confluences`	logical. If TRUE, append the outlet flowline of each identified mainstem that flows into a requested one.

Details

Each network realizes a mainstem differently, so each is fetched differently.

For `"nhdpv2"`, comids come from the ref_rivers lookup table (see add_mainstems), topology and length from the updated network attributes (see get_vaa with `updated_network = TRUE`), and geometry from get_nhdplus. Columns are `comid`, `tocomid`, `lengthkm`, `hydroseq`, and `mainstemid`. `tocomid` is already dendritic so no `divergence` is carried.

For `"nhdphr"`, NHDPlusIDs come from the ref_rivers NHDPlusHR lookup table and the flowlines from get_nhdphr. Columns are `nhdplusid` (as character), `fromnode`, `tonode`, `hydroseq`, `lengthkm`, `divergence`, and `mainstemid`. Flowlines link by node rather than by `toid`. The NHDPlusHR lookup does not cover every mainstem.

For `"3dhp"`, 3DHP carries mainstem uris natively and flowlines come from get_3dhp. Columns are `id3dhp`, `toid` (built from `dnhydrosequence`, `""` where it leaves the result), `hydroseq` (renamed from `hydrosequence`), `lengthkm`, `divergence`, `featuredate`, and `mainstemid`. 3DHP identifiers are not persistent across releases; `featuredate` records which release the flowlines came from.

With `include_confluences = TRUE`, the outlet flowline of each other identified mainstem that flows into a requested mainstem is appended. Those are the rows get_confluences needs to find

where tributary mainstems join. They are found by attribute queries on the flowlines' downstream connections, not by area of interest, and inflowing flowlines with no mainstem uri are left out.

The NHDPlusV2 lookup and updated network attribute tables are large downloads on first use (see add_mainstems and get_vaa), so this function has no example.

```
fl <- get_mainstem_flowlines("https://geoconnex.us/ref/mainstems/2408056",
                             "3dhp", include_confluences = TRUE)

hydroloom::get_confluences(fl)
```

Value

sf data.frame of LINESTRING flowlines, or NULL if none were found. Mainstems with no flowlines are left out with a warning.

---

get_nhdarea *Find NHDPlusV2 Areas*

Description

Subsets NHDPlusV2 Area features by location (POINT), area (POLYGON), or set of IDs. See download_nhdplusv2 for source data documentation.

Usage

```
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)
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 server 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 (either input or default). All buffer and distance operations are handled internally using an EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object, or a data.frame when `skip_geometry = TRUE`

`get_nhdphr` *Get NHDPlusHR Data*

Description

Calls the NHDPlus_HR web service and returns sf data.frames for the selected layers. See https://hydro.nationalmap.gov/arcg for source data documentation.

Usage

```
get_nhdphr(
  AOI = NULL,
  ids = NULL,
  type = NULL,
  reachcode = NULL,
  t_srs = NULL,
  buffer = 0.5,
  page_size = 2000
)
```

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.
`ids`	character vector of nhdplusid ids
`type`	character. Type of feature to return e.g. c("networknhdflowline", nonnetworknhdflowline", nhdwaterbody", "nhdpluscatchment"). If NULL (default) a data.frame of available types is returned
`reachcode`	character vector of reachcodes NOTE: performance of this query is currently very poor, spatial queries are the primary use of this function.
`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
`page_size`	numeric default number of features to request at a time. Reducing may help if 500 errors are experienced.

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 CRS of EPSG:4269 is preserved. In all cases, a user-defined SRS can be passed to `t_srs` which will override all previous SRS (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 or valid types if no type supplied

Examples

```
AOI <- sf::st_as_sfc(sf::st_bbox(c(xmin = -89.56684, ymin = 42.99816,
                                   xmax = -89.24681, ymax = 43.17192),
                                 crs = "+proj=longlat +datum=WGS84 +no_defs"))

# get flowlines and hydrolocations
flowlines <- get_nhdphr(AOI = AOI, type = "networknhdflowline")
point <- get_nhdphr(AOI = AOI, type = "nhdpoint")
waterbody <- get_nhdphr(AOI = AOI, type = "nhdwaterbody")

if(!is.null(waterbody) & !is.null(flowlines) & !is.null(point)) {
plot(sf::st_geometry(waterbody), col = "lightblue", border = "lightgrey")
plot(sf::st_geometry(flowlines), col = "blue", add = TRUE)
plot(sf::st_geometry(point), col = "grey", pch = "+", add = TRUE) }

# given universalreferenceid (reachcodes), can query for them but only
# for hydrolocations. This is useful for looking up mainstem ids.

get_nhdphr(reachcode = "13020101021927", type = "networknhdflowline")
```

`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.

NOTE: not all flowlines have catchments and not all catchment have flowlines. Flowlines without catchments will have a 0 areasqkm attribute. Catchments without flowlines terminate in a sink and will have a negative valued comid.

Usage

```
get_nhdplus(
  AOI = NULL,
  comid = NULL,
  nwis = NULL,
  realization = "flowline",
  streamorder = NULL,
  t_srs = NULL,
  properties = NULL,
  skip_geometry = FALSE
)
```

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.
properties	character vector. Column names to return. When NULL (default), all columns are returned.
skip_geometry	logical. If TRUE, omit geometry from the response and return a plain data.frame. Default FALSE.

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 server 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 (either input or default). All buffer and distance operations are handled internally using an EPSG:5070 Albers Equal Area projection

Value

sf single, or named list of, simple feature objects. `NULL` if no realization succeeded (e.g. degraded service).

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` *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

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

Examples

```
# a small NHDPlusHR sample, downloaded to a temporary directory
work_dir <- file.path(tempdir(check = TRUE), "hr_example")
try(source(system.file("extdata/nhdplushr_data.R", package = "hydrogeofetch")))

if(file.exists(file.path(work_dir, "03_sub.gpkg"))) {

get_nhdplushr(work_dir, pattern = "03_sub.gpkg")

get_nhdplushr(work_dir, pattern = "03_sub.gpkg", layers = NULL)

get_nhdplushr(work_dir, pattern = "03_sub.gpkg",
              layers = "NHDFlowline",
              min_size_sqkm = 10, simp = 10, proj = 5070)

}

# For real data, download subregions first -- see download_nhdplushr().
```

---

get_nldi_basin *Get NLDI Basin Boundary*

Description

Get a basin boundary for a given NLDI feature.

Usage

```
get_nldi_basin(nldi_feature, simplify = TRUE, split = FALSE)
```

Arguments

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 ‘featureID‘ is a known identifier from the specified ‘featureSource‘.
simplify	logical should response geometry be simplified for visualization and performance?
split	logical should response resolve precisely to the location of the ‘nldi_feature‘? Setting ‘TRUE‘ calls an additional service and will be slower and less robust.

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")

site <- get_nldi_feature(nldi_nwis)

basin <- get_nldi_basin(nldi_feature = nldi_nwis)

plot(st_geometry(basin))

basin

basin2 <- get_nldi_basin(nldi_feature = nldi_nwis,
                         simplify = FALSE, split = TRUE)

if(inherits(basin, "sf") & inherits(basin2, "sf")) {

length(st_coordinates(basin))
length(st_coordinates(basin2))

plot(st_geometry(st_buffer(st_transform(site, 5070),
                           units::set_units(3000, "m"))), border = NA)

plot(st_geometry(st_transform(site, 5070)), add = TRUE)
plot(st_geometry(st_transform(basin2, 5070)), add = TRUE)

plot(st_geometry(st_transform(basin, 5070)), border = "red", add = TRUE)

}
```

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 get_nldi_sources and the ‘featureID‘ 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"))
```

get_nldi_index *Get NLDI Index*

Description

uses the Network Linked Data Index to retrieve an estimated network location for the given point. If not within a grid cell of a flowline, will use a raindrop trace service to find the nearest downslope flowline location.

Usage

```
get_nldi_index(location)
```

Arguments

location	numeric WGS84 lon/lat pair (X, Y)

Value

sf data.frame of estimated network locations for the given point. NULL if the NLDI service is unavailable.

Examples

```
index <- get_nldi_index(c(-89.276, 42.988))

if(inherits(index, "sf")) {

plot_nhdplus(
  bbox = sf::st_bbox(
    sf::st_buffer(
      sf::st_transform(index[1,], 5070), units::set_units(1000, "m")
      )
    )
)
plot(sf::st_geometry(sf::st_transform(index, 3857)), add = TRUE)

}
```

---

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

```
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 arranged by distance from POINT AOI

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 server 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 (either input or default). All buffer and distance operations are handled internally using an EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object, or a data.frame when `skip_geometry = TRUE`

`get_raindrop_trace` *Get Raindrop Trace*

Description

Uses a raindrop trace web service to trace the nhdplus digital elevation model to the nearest downslope flowline.

Usage

```
get_raindrop_trace(point, direction = "down")
```

Arguments

`point`	sfc POINT including crs as created by: `sf::st_sfc(sf::st_point(.. ,..), crs)`
`direction`	character `"up"`, `"down"`, or `"none"`. Controls the portion of the split flowline that is returned along with the raindrop trace line.

Value

sf data.frame containing raindrop trace and requested portion of flowline.

Examples

```
point <- sf::st_sfc(sf::st_point(x = c(-89.2158, 42.9561)), crs = 4326)

(trace <- get_raindrop_trace(point))

if(inherits(trace, "sf")) {
bbox <- sf::st_bbox(trace) + c(-0.005, -0.005, 0.005, 0.005)

hydrogeofetch::plot_nhdplus(bbox = bbox, cache_data = FALSE)

plot(sf::st_transform(sf::st_sfc(point, crs = 4326), 3857), add = TRUE)
plot(sf::st_transform(sf::st_geometry(trace)[1], 3857), add = TRUE, col = "red")
plot(sf::st_transform(sf::st_geometry(trace)[2], 3857), add = TRUE, col = "black")
}
```

`get_split_catchment` *Get split catchment*

Description

Uses a catchment splitting web service to retrieve the portion of a catchment upstream of the point provided.

Usage

```
get_split_catchment(point, upstream = TRUE)
```

Arguments

`point`	scf POINT including crs as created by: `sf::st_sfc(sf::st_point(.. ,..), crs)`.
`upstream`	logical If TRUE, the entire drainage basin upstream of the point provided is returned in addition to the local catchment.

Details

This service works within the conterminous US NHDPlusV2 domain. If the point provided falls on an NHDPlusV2 flowline as retrieved from get_raindrop_trace the catchment will be split across the flow line. IF the point is not along the flowline a small sub catchment will typically result. As a result, most users of this function will want to use get_raindrop_trace prior to calls to this function.

An attempt is made to eliminate polygon shards if they exist in the output. However, there is a chance that this function will return a multipolygon data.frame.

Value

sf data.frame containing the local catchment, the split portion and optionally the total drainage basin.

Examples

```
point <- sf::st_sfc(sf::st_point(x = c(-89.2158, 42.9561)), crs = 4326)

trace <- get_raindrop_trace(point)

if(inherits(trace, "sf")) {

(snap_point <- sf::st_sfc(sf::st_point(trace$intersection_point[[1]]),
                          crs = 4326))

(catchment <- get_split_catchment(snap_point))

if(inherits(catchment, "sf")) {
bbox <- sf::st_bbox(catchment) + c(-0.005, -0.005, 0.005, 0.005)
```

```
hydrogeofetch::plot_nhdplus(bbox = bbox, cache_data = FALSE)

if(length(sf::st_geometry(catchment)) > 1) {
  plot(sf::st_transform(sf::st_geometry(catchment)[2], 3857), add = TRUE, col = "black")
}
plot(sf::st_transform(sf::st_geometry(catchment)[1], 3857), add = TRUE, col = "red")
plot(sf::st_transform(sf::st_sfc(point, crs = 4326), 3857), add = TRUE, col = "white")
}

(catchment <- get_split_catchment(snap_point, upstream = FALSE))


if(inherits(catchment, "sf")) {
bbox <- sf::st_bbox(catchment) + c(-0.005, -0.005, 0.005, 0.005)

hydrogeofetch::plot_nhdplus(bbox = bbox, cache_data = FALSE)

plot(sf::st_transform(sf::st_geometry(catchment)[1], 3857), add = TRUE, col = "red")
if(length(sf::st_geometry(catchment)) > 1) {
  plot(sf::st_transform(sf::st_geometry(catchment)[2], 3857), add = TRUE, col = "black")
}
plot(sf::st_transform(sf::st_sfc(point, crs = 4326), 3857), add = TRUE, col = "white")
}

pour_point <- sf::st_sfc(sf::st_point(x = c(-89.25619, 42.98646)), crs = 4326)

(catchment <- get_split_catchment(pour_point, upstream = FALSE))

if(inherits(catchment, "sf")) {
bbox <- sf::st_bbox(catchment) + c(-0.005, -0.005, 0.005, 0.005)

hydrogeofetch::plot_nhdplus(bbox = bbox, cache_data = FALSE)

plot(sf::st_transform(sf::st_geometry(catchment)[1], 3857), add = TRUE, col = "red")
if(length(sf::st_geometry(catchment)) > 1) {
  plot(sf::st_transform(sf::st_geometry(catchment)[2], 3857), add = TRUE, col = "black")
}
plot(sf::st_transform(sf::st_sfc(pour_point, crs = 4326), 3857), add = TRUE, col = "white")
}

}
```

`get_tocomid` *Get tocomid*

Description

Given flowlines with fromnode and tonode attributes, will return a toid attribute that is the result of joining tonode and fromnode attributes. In the case that a terminalpa attribute is included, the join is executed by terminalpa group. This is done grouped by terminalpathID because duplicate node ids have been encountered across basins in some datasets. If 'remove_coastal' is 'TRUE' (the default) either ftype or fcode are required. Uses the add_toids function.

Usage

```
get_tocomid(
  x,
  return_dendritic = TRUE,
  missing = 0,
  remove_coastal = TRUE,
  add = TRUE
)
```

Arguments

x	data.frame with comid, tonode, fromnode, and (optionally) divergence and terminalpa attributes.
return_dendritic	logical if TRUE, a divergence attribute is required (2 indicates diverted path, 1 is main) and diverted paths will be treated as headwaters. If this is FALSE, the return value is a data.frame including the comid and tocomid attributes.
missing	integer value to use for terminal nodes.
remove_coastal	logical remove coastal features prior to generating tocomid values? ftype or fcode are required if 'TRUE'. fcode == 56600 or fcode == "Coastline" will be removed.
add	logical if TRUE, a tocomid column will be added, otherwise a data.frame with two columns will be returned.

Value

data.frame containing comid and tocomid attributes or all attributes provided with comid and tocomid in the first and second columns..

Examples

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

tocomid <- get_tocomid(sample_flines)

tocomid <- get_tocomid(sample_flines, return_dendritic = FALSE)
```

get_vaa *NHDPlusV2 Attribute Subset*

Description

Return requested NHDPlusV2 Attributes.

Calling this function downloads the attribute table to get_vaa_path if it is not already cached there. The base table is roughly 260 MB and the updated network table another 170 MB, so this function has no example. To try it without writing to your user data directory, point hydrogeofetch_data_dir at a temporary path first:

```
old_dir <- hydrogeofetch_data_dir()
hydrogeofetch_data_dir(file.path(tempdir(), "vaa"))

get_vaa("slope")
get_vaa(c("slope", "lengthkm"))
get_vaa("reachcode", updated_network = TRUE)

hydrogeofetch_data_dir(old_dir)
```

Usage

```
get_vaa(
  atts = NULL,
  path = get_vaa_path(),
  download = TRUE,
  updated_network = FALSE
)
```

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 hydrogeofetch_data_dir. Also see: get_vaa_path
download	logical if TRUE, the default, will download VAA table if not found at path.
updated_network	logical default FALSE. If TRUE, updated network attributes from E2NHD and National Water Model retrieved from doi:10.5066/P976XCVT.

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

`get_vaa_names` *Available NHDPlusV2 Attributes*

Description

Find variables available from the NHDPlusV2 attribute data.frame

Calling this function downloads the attribute table to get_vaa_path if it is not already cached there. The table is roughly 260 MB, so this function has no example.

```
old_dir <- hydrogeofetch_data_dir()
hydrogeofetch_data_dir(file.path(tempdir(), "vaa"))

get_vaa_names()

hydrogeofetch_data_dir(old_dir)
```

Usage

```
get_vaa_names(updated_network = FALSE)
```

Arguments

`updated_network`
: logical default FALSE. If TRUE, updated network attributes from E2NHD and National Water Model retrieved from doi:10.5066/P976XCVT.

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

character vector

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

Description

hydrogeofetch will download and cache a parquet 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 hydrogeofetch_data_dir.

Usage

```
get_vaa_path(updated_network = FALSE)
```

Arguments

updated_network	logical default FALSE. If TRUE, returns path to updated network parameters. See get_vaa for more.

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

character file path

Examples

```
get_vaa_path()

get_vaa_path(updated_network = TRUE)
```

get_waterbodies *Find NHDPlusV2 Water Bodies*

Description

Subsets NHDPlusV2 waterbody features by location (POINT), area (POLYGON), or set of IDs. See download_nhdplusv2 for source data documentation.

Usage

```
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 server 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 (either input or default). All buffer and distance operations are handled internally using an EPSG:5070 Albers Equal Area projection

Value

a simple features (sf) object, or a data.frame when `skip_geometry = TRUE`

`get_wb_outlet` *Get Waterbody Outlet*

Description

Get Waterbody Outlet

Usage

```
get_wb_outlet(lake_id, network)
```

Arguments

lake_id	integer COMID (or character permanent identifier for hi res) of lake.
network	data.frame of network features containing wbareacomi, and Hydroseq

Value

sf data.frame with single record of network COMID associated with most-downstream reach in the NHD Waterbody

Examples

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

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

lake_COMID <- wtbdy$COMID[wtbdy$GNIS_NAME=='Lake Mendota 254']

get_wb_outlet(13293262, fline)
```

---

get_xs_point *Get Cross Section From Point (experimental)*

Description

Uses a cross section retrieval web services to retrieve a cross section given a point and specified width. Orientation is determined based on direction of the flowline found near point. This function uses a 10m National Elevation Dataset request on the back end.

Usage

```
get_xs_point(point, width, num_pts)
```

Arguments

point	sfc POINT including crs as created by: `sf::st_sfc(sf::st_point(.. ,..), crs)`
width	Cross section width in meters.
num_pts	numeric number of points to retrieve along the cross section.

Value

sf data.frame containing points retrieved.

Examples

```
point <- sf::st_sfc(sf::st_point(x = c(-105.97218, 36.17592)), crs = 4326)

(xs <- get_xs_point(point, 300, 100))

if(inherits(xs, "sf")) {

bbox <- sf::st_bbox(xs) + c(-0.005, -0.005, 0.005, 0.005)

hydrogeofetch::plot_nhdplus(bbox = bbox, cache_data = FALSE)
```

```
plot(sf::st_transform(sf::st_geometry(xs), 3857), pch = ".", add = TRUE, col = "red")
plot(sf::st_transform(sf::st_sfc(point, crs = 4326), 3857), add = TRUE)

plot(xs$distance_m, xs$elevation_m)
}
```

`get_xs_points` *Get Cross Section Endpoints (experimental)*

Description

Uses a cross section retrieval web services to retrieve a cross section between two endpoints.

Usage

```
get_xs_points(point1, point2, num_pts, res = 1)
```

Arguments

`point1`	sfc POINT including crs as created by: `sf::st_sfc(sf::st_point(.. ,..), crs)`
`point2`	sfc POINT including crs.
`num_pts`	numeric number of points to retrieve along the cross section.
`res`	integer resolution of 3D Elevation Program data to request. Must be one of: 1, 3, 5, 10, 30, 60.

Value

sf data.frame containing points retrieved.

Examples

```
point1 <- sf::st_sfc(sf::st_point(x = c(-105.9667, 36.17602)), crs = 4326)
point2 <- sf::st_sfc(sf::st_point(x = c(-105.97768, 36.17526)), crs = 4326)

(xs <- get_xs_points(point1, point2, 100))

if(inherits(xs, "sf")) {

bbox <- sf::st_bbox(xs) + c(-0.005, -0.005, 0.005, 0.005)

hydrogeofetch::plot_nhdplus(bbox = bbox, cache_data = FALSE)

plot(sf::st_transform(sf::st_geometry(xs), 3857), pch = ".", add = TRUE, col = "red")
plot(sf::st_transform(sf::st_sfc(point1, crs = 4326), 3857), add = TRUE)
```

```
plot(sf::st_transform(sf::st_sfc(point2, crs = 4326), 3857), add = TRUE)

plot(xs$distance_m, xs$elevation_m)
}
```

`hydrogeofetch_cache_clear` *Clear cached hydrogeofetch data*

Description

Removes files that hydrogeofetch has written to hydrogeofetch_data_dir. Use this to reclaim space or to force downloaded data to be refreshed.

Downloaded datasets are kept until removed with this function – nothing expires them automatically – so it is worth calling periodically if you use get_vaa or add_mainstems on many different areas.

Usage

```
hydrogeofetch_cache_clear(what = "all", ask = interactive())
```

Arguments

`what`	character which cached data to remove. One or more of 'responses' (the 'memoise' filesystem cache of web service responses), 'tiles' (basemap tiles cached by plot_nhdplus), 'vaa' (value added attribute parquet files), 'mainstems' (mainstem lookup tables), or 'all' (the default) for everything in the data directory.
`ask`	logical if TRUE, the default in an interactive session, asks for confirmation before deleting.

Value

character vector of paths removed, invisibly.

Examples

```
# pointed at an empty temporary directory here so the example does not
# remove anything you have actually cached
old_dir <- hydrogeofetch_data_dir()
hydrogeofetch_data_dir(file.path(tempdir(), "hgf_cache_demo"))

hydrogeofetch_cache_clear(ask = FALSE)

hydrogeofetch_data_dir(old_dir)
```

`hydrogeofetch_cache_settings` *hydrogeofetch cache settings*

Description

Provides an interface to adjust hydrogeofetch ‘memoise‘ cache.

Mode and timeout can also be set using environment variables. ‘HYDROGEOFETCH_MEMOISE_CACHE‘ and ‘HYDROGEOFETCH_MEMOISE_TIMEOUT‘ are used unless overridden with this function. The old ‘NHDPLUSTOOLS_*‘ names are still recognized as a fallback.

The default mode is ’memory’, so cached responses live only as long as the R session. Set ’filesystem’ to persist them in hydrogeofetch_data_dir, and see hydrogeofetch_cache_clear to remove them again.

Usage

```
hydrogeofetch_cache_settings(mode = NULL, timeout = NULL)
```

Arguments

`mode`	character ’memory’ (default) or ’filesystem’
`timeout`	numeric number of seconds until caches invalidate

Value

list containing settings at time of calling. If inputs are NULL, current settings. If settings are altered, previous setting values.

`hydrogeofetch_data_dir` *Get or set hydrogeofetch data directory*

Description

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

Usage

```
hydrogeofetch_data_dir(dir = NULL)
```

Arguments

`dir`	path of desired data directory

Value

character path of data directory (silent when setting)

Examples

```
hydrogeofetch_data_dir()

# set it somewhere else, then put it back
old_dir <- hydrogeofetch_data_dir()

hydrogeofetch_data_dir(file.path(tempdir(check = TRUE), "demo"))

hydrogeofetch_data_dir()

hydrogeofetch_data_dir(old_dir)
```

---

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.

Value

sf data.frame containing standalone network

Examples

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

(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 = "hydrogeofetch"))

# 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)
```

`map_nhdplus` *Make Interactive Map of NHDPlus*

Description

Given a list of outlets, get their basin boundaries and network and return a leaflet map in EPSG:4326.

Usage

```
map_nhdplus(
  outlets = NULL,
  bbox = NULL,
  streamorder = NULL,
  nhdplus_data = NULL,
  gpkg = NULL,
  flowline_only = NULL,
  plot_config = NULL,
  overwrite = TRUE,
  cache_data = NULL,
  return_map = FALSE
)
```

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.
flowline_only	boolean only subset and plot flowlines only, default=FALSE
plot_config	list containing plot configuration, see details.
overwrite	passed on to subset_nhdplus.
cache_data	character path to rds file where all plot data can be cached. If file doesn't exist, it will be created. If set to FALSE, all caching will be turned off – this includes basemap tiles.
return_map	if FALSE (default), a data.frame of plot data is returned invisibly in NAD83 Lat/Lon, if TRUE the leaflet object is returned

Details

map_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.

See plot_nhdplus for details on plot configuration.

Value

data.frame or leaflet map (see return_map)

Examples

```
map_nhdplus("05428500")

map_nhdplus("05428500", streamorder = 2)

map_nhdplus(list(13293970, 13293750))
```

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

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

#return leaflet object
map_nhdplus("05428500", return_map = 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
)
```

Arguments

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 ‘featureID‘ 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 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
```

---

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

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,
  basemap = "Esri.NatGeoWorldMap",
  zoom = NULL,
  add = FALSE,
  actually_plot = TRUE,
  overwrite = TRUE,
  flowline_only = NULL,
  cache_data = 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.
basemap	character indicating which basemap type to use. Chose from: get_tiles.
zoom	integer passed on to get_tiles. This value will override the default set by the package.
add	boolean should this plot be added to an already built map.
actually_plot	boolean actually draw the plot? Used to get data subset only.
overwrite	passed on to subset_nhdplus.
flowline_only	boolean only subset and plot flowlines only, default=FALSE
cache_data	character path to rds file where all plot data can be cached. If file doesn't exist, it will be created. If set to FALSE, all caching will be turned off – this includes basemap tiles.

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", "outlets", "network_wtbd", and "off_network_wtbd". 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))
```

4. network_wtbd

```
list(lwd = 1, col = "lightblue", border = "black")
```

5. off_network_wtbd

```
list(lwd = 1, col = "darkblue", border = "black")
```

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

```
# Beware plot_nhdplus caches data to the default location.
# If you do not want data in "user space" change the default.
old_dir <- hydrogeofetch::hydrogeofetch_data_dir()
hydrogeofetch_data_dir(tempdir())

plot_nhdplus("05428500")

plot_nhdplus("05428500", streamorder = 2)

plot_nhdplus(list(13293970, 13293750))

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

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 <- hydroloom::navigate_hydro_network(fline, 13293970, "UT")

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")))

hydrogeofetch::hydrogeofetch_data_dir(old_dir)
```

`rescale_catchment_characteristics` *Rescale Catchment Characteristics*

Description

Given catchment characteristics to retrieve or process will aggregate and / or split the characteristics according to a lookup table.

Usage

```
rescale_catchment_characteristics(
  vars,
  lookup_table,
  refactored_areas = NULL,
  catchment_characteristics = NULL,
  catchment_areas = NULL,
  source = "usgs",
  aoi = "cat"
)
```

Arguments

`vars`	data.frame containing ‘characteristic_id‘ retrieved from get_characteristics_metadata and ‘summary_statistic‘ indicating which summary statistic should be applied to rescale each characteristic. Accepted values are "sum," "length_weighted_mean," "area_weighted_mean," "min," and "max."
`lookup_table`	data.frame containing ‘id‘ numeric vector of identifiers at the desired scale; "comid" is a numeric vector of NHDPlusV2 identifiers; "member_comid" contains formatted NHDPlusV2 COMIDs indicating that the catchments in question need to be split. If catchments have not been split, the columns "comid" and "member_comid" should be identical.
`refactored_areas`	data.frame containing columns "featureid" and "areasqkm." Used to retrieve adjusted catchment areas in the case of split catchments. If not provided, either no split catchments can be considered or the ‘catchment_areas‘ parameter is required.

`catchment_characteristics`
: data.frame containing columns "characteristic_id", "comid", "characteristic_value", and "percent_nodata". If not provided, it will be retrieved from get_catchment_characteristics using the characteristic ids from ‘vars‘ and the comids from ‘lookup_table‘.

`catchment_areas`
: data.frame containing columns "comid", "areasqkm", "split_catchment_areasqkm", and "split_area_prop". If not provided, it will be retrieved from ‘refactored_areas‘ and/or get_vaa.

`source`
: character `"usgs"` (default) or `"streamcat"`. Passed through to get_catchment_characteristics when `catchment_characteristics` is not provided.

`aoi`
: character area of interest, passed through to get_catchment_characteristics when `catchment_characteristics` is not provided. See get_catchment_characteristics for details.

Details

NOTE: Since this algorithm works on catchment characteristics that are spatial averages, when splitting, the average condition is apportioned evenly to each split. In some cases, such as with land cover or elevation, this may not be appropriate and source data should be used to derive new characteristics. In addition, this function handles catchment areas for split catchments but makes no adjustments for the length of flowlines in those catchments. Therefore, requests for length-weighted mean values may not be appropriate when working with split catchments.

Value

data.frame with one row per unique ‘id‘ in ‘lookup_table‘. Always contains ‘id‘, ‘areasqkm_sum‘ (total catchment area contributing to that ‘id‘), and ‘lengthkm_sum‘ (total flowline length). Each characteristic named in ‘vars‘ adds a column called ‘<characteristic_id>_<statistic>‘, where the statistic suffix is one of "area_wtd", "length_wtd", "sum", "min", or "max" following the ‘summary_statistic‘ requested for it – for example ‘CAT_IMPV11_area_wtd‘ or ‘CAT_BASIN_AREA_sum‘. Where the source reports a no-data fraction, a matching ‘percent_nodata_<characteristic_id>_area_wtd‘ column is included. Returns ‘NULL‘ if ‘catchment_characteristics‘ was not supplied and could not be retrieved from the web service.

Examples

```
vars <- data.frame(characteristic_id = c("CAT_IMPV11","CAT_BASIN_AREA"),
                   summary_statistic = c("area_weighted_mean","sum"))
lookup_table <- data.frame(id = rep(10012268, 2),
                           comid = c(4146596, 4147382),
                           member_comid = c(4146596, 4147382))
rescale_catchment_characteristics(vars, lookup_table)

vars <- data.frame(characteristic_id = c("CAT_ELEV_MIN","CAT_ELEV_MAX"),
                   summary_statistic = c("min","max"))
lookup_table <- data.frame(id = rep(10012268, 2),
                           comid = c(4146596, 4147382),
                           member_comid = c(4146596, 4147382))
rescale_catchment_characteristics(vars, lookup_table)
```

```
vars <- data.frame(characteristic_id = c("CAT_EWT","CAT_TWI", "CAT_BASIN_AREA"),
                   summary_statistic = c("area_weighted_mean", "area_weighted_mean","sum"))
lookup_table <- data.frame(id = c(10012268, 10012268, 10024047, 10024048),
                           comid = c(4146596, 4147382, 4147396, 4147396),
                         member_comid = c("4146596", "4147382", "4147396.1", "4147396.2"))
comid_areas <- data.frame(featureid = c("4146596", "4147382", "4147396.1", "4147396.2"),
                              areasqkm = c(0.9558, 11.9790, 6.513294, 1.439999))
rescale_catchment_characteristics(vars, lookup_table, refactored_areas = comid_areas)
```

`rpu_boundaries` *RPU Boundaries Raster Processing Unit boundaries*

Description

RPU Boundaries Raster Processing Unit boundaries

Usage

```
rpu_boundaries
```

Format

An object of class `"sf"`

`subset_nhdplus` *Subset NHDPlus*

Description

Saves a subset of the National Seamless database or other hydrogeofetch 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

```
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.
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.
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

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

nhdplus_path(sample_data)
```

```
sample_flines <- sf::st_zm(sf::read_sf(nhdplus_path(), "NHDFlowline_Network"))

plot(sf::st_geometry(sample_flines),
     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 <- hydroloom::navigate_hydro_network(sample_flines, start_comid, "UT")

plot(sf::st_geometry(dplyr::filter(sample_flines, COMID %in% comids)),
     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(sf::st_geometry(catchment), add = TRUE)

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

plot(sf::st_geometry(waterbody),
     col = rgb(0, 0, 1, alpha = 0.5), add = TRUE)

# Cleanup temp
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 = "hydrogeofetch"))

up_ids <- hydroloom::navigate_hydro_network(hr_data$NHDFlowline, 15000500028335, "UT")
```

```
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, strict = FALSE)
```

Arguments

fline	sf data.frame NHD Flowlines with comid, pathlength, lengthkm, hydroseq, levelpathi, rpuid, and arbolatesu (dnhydroseq is required if tocomid is not provided).
rpu	character e.g. "01a"
run_make_standalone	logical default TRUE should the run_make_standalone function be run on result?
strict	logical if TRUE, paths that extend outside the RPU but have no tributaries in the upstream RPU will be included in the output.

Value

data.frame containing subset network

Examples

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

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

subset_rpu(sample_flines, rpu = "07b")
```

---

subset_vpu *Subset by Vector Processing Unit*

Description

Calls subset_rpu for all raster processing units for the requested vector processing unit.

Usage

```
subset_vpu(fline, vpu, include_null_rpuid = TRUE, run_make_standalone = TRUE)
```

Arguments

fline	sf data.frame NHD Flowlines with comid, pathlength, lengthkm, hydroseq, levelpathi, rpuid, vpuid, and arbolatesu (dnhydroseq is required if tocomid is not provided).
vpu	character e.g. "01"
include_null_rpuid	logical default TRUE. Note that there are some flowlines that may have a NULL rpuid but be included in the vector processing unit.
run_make_standalone	logical default TRUE 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 = "hydrogeofetch"))

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

subset_vpu(sample_flines, "07")
```

`translate_flowlines` *Translate flowlines between networks*

Description

Finds the stretch of another network's flowlines (NHDPlusV2, NHDPlusHR, or 3DHP) that covers each given flowline, on the same mainstem. Each source flowline comes back as one or more pieces of target flowlines, each cut by measure.

Usage

```
translate_flowlines(
  ids,
  from,
  to,
  snap_m = 50,
  search_radius = units::set_units(1000, "m"),
  precision = 10
)
```

Arguments

`ids`	vector of source flowline identifiers: comids for `"nhdpv2"`, NHDPlusIDs for `"nhdphr"`, or id3dhp for `"3dhp"`.
`from`	character network `x` is indexed to, one of "nhdpv2", "nhdphr", or "3dhp".
`to`	character network to place `x` on, one of "nhdpv2", "nhdphr", or "3dhp".
`snap_m`	numeric meters, passed to get_subpath.
`search_radius`	units distance passed to index_points_to_lines, in a linear unit since indexing is in EPSG:5070. Only flowlines of the point's own mainstem are searched, so the radius can be wide enough to span differences between networks, such as a reservoir that one network routes differently from another.
`precision`	numeric meters, passed to index_points_to_lines. Lines near each point are segmentized to this spacing so the match is to the nearest point on a line, not the nearest vertex; `NA` matches vertices only.

Details

The mainstem of each source flowline comes from its identifier, as described in translate_hydrolocations. The source and target flowlines of that mainstem come from get_mainstem_flowlines.

The downstream (end) and upstream (start) nodes of each source flowline are indexed to the target flowlines of its mainstem with index_points_to_lines and `aggregate_ids`, and the stretch between them is taken with get_subpath. An end within `snap_m` meters of a target flowline boundary moves to that boundary, so two source flowlines that share a node share one cut. `line_from_measure` and `line_to_measure` place each piece along the source flowline, 0 at its downstream end, by the pieces' lengths. `from_offset` and `to_offset` are the distances from the source flowline's

downstream and upstream nodes to the target flowlines, in meters (indexing is in EPSG:5070, as in translate_hydrolocations).

`status` is one of: `placed`; `inverted` (the source flowline's downstream node is upstream of its upstream node on the target); `collapsed` (both ends land on one point, one row with `NA id`); `not_placed` (an end has no target flowline within `search_radius`); `no_mainstem`; `mainstem_superseded`; `source_not_found` (the source flowline is not among the fetched flowlines of its mainstem); and `mainstem_not_in_target`. Rows that are not placed have `NA` pieces. `measured_on` is as described in translate_hydrolocations. The reverse translation is the same call with `from` and `to` swapped.

The ref_rivers lookup tables are large downloads on first use (see add_mainstems), so this function has no example.

```
translate_flowlines(8894344, "nhdpv2", "3dhp")
```

Value

data.frame with `line_id` (the source id), `segment_order` (1 at the downstream end), `id` (target flowline), `id_from_measure`, `id_to_measure`, `line_from_measure`, `line_to_measure`, `length_km`, `from_offset`, `to_offset`, `status`, and `measured_on`; an sf data.frame of piece geometry when **lwgeom** is installed.

`translate_hydrolocations` *Translate hydro locations between networks*

Description

Places points that are indexed to one network (NHDPlusV2, NHDPlusHR, or 3DHP) on the flowlines of another, keeping each point on the mainstem it is on in the source network. The mainstem comes from the point's source identifier, not from its geometry, so a point near a confluence stays on its own river.

Usage

```
translate_hydrolocations(
  x,
  from,
  to,
  method = c("geometry", "hydro_location"),
  search_radius = units::set_units(1000, "m"),
  precision = 10
)
```

Arguments

x	sf POINT data.frame with the source identifier column for from (comid, nhdplusid or id3dhp) or a mainstem_uri column.
from	character network x is indexed to, one of "nhdpv2", "nhdphr", or "3dhp".
to	character network to place x on, one of "nhdpv2", "nhdphr", or "3dhp".
method	character "geometry" or "hydro_location". See details.
search_radius	units distance passed to index_points_to_lines, in a linear unit since indexing is in EPSG:5070. Only flowlines of the point's own mainstem are searched, so the radius can be wide enough to span differences between networks, such as a reservoir that one network routes differently from another.
precision	numeric meters, passed to index_points_to_lines. Lines near each point are segmentized to this spacing so the match is to the nearest point on a line, not the nearest vertex; NA matches vertices only.

Details

The mainstem of each point comes from, in order: a mainstem_uri column on x; for "3dhp", a mainstemid column; otherwise the identifier column for from (comid, nhdplusid or id3dhp), looked up in the ref_rivers tables (see add_mainstems) or the 3DHP flowlines. A superseded mainstem (see check_mainstems) is reported and not placed; update_mainstems resolves it.

The target flowlines of each mainstem come from get_mainstem_flowlines. There are two ways to place a point on them.

method = "geometry" indexes the point to the target flowlines of its mainstem with index_points_to_lines and aggregate_ids. offset is the distance from the point to the target flowline.

method = "hydro_location" indexes the point on the source flowlines, records it as a distance from the nearest confluence that both networks have on that mainstem (get_confluences and add_hydro_location_distance), and lays the distance out on the target (index_hydro_location_distance). The placement is approximate in absolute terms but stays on the same side of the confluence, which is what matters for upstream drainage. offset is the distance from the point to the source flowline.

status is one of: placed; not_placed (no flowline within search_radius); no_mainstem; mainstem_superseded; mainstem_not_in_target (no target flowlines for the mainstem); and, for "hydro_location", no_shared_hydro_location (no confluence on the mainstem is in both networks, so the point is known only to be on mainstem_uri), past_end and hydro_location_not_in_x as described in index_hydro_location_distance. measured_on names the target network and the ref_rivers release, or for 3DHP the range of featuredate. 3DHP identifiers are not persistent, so a 3DHP placement holds for that release.

Points and flowlines are indexed in EPSG:5070 (CONUS Albers), so offset is in meters.

The ref_rivers lookup tables are large downloads on first use (see add_mainstems), so this function has no example.

```
# a point on New Hope River just above the Northeast Creek mouth
pt <- sf::st_sf(comid = 8894344,
                geometry = sf::st_sfc(sf::st_point(c(-78.9763, 35.8355)),
                                      crs = 4326))

translate_hydrolocations(pt, "nhdpv2", "3dhp", method = "hydro_location")
```

Value

x with mainstem_uri, id (target flowline), id_measure, offset, past_end_km, status, method, and measured_on added, and for "hydro_location" also hydro_location_id and distance_km.

---

update_mainstems *Update superseded mainstem identifiers*

Description

Given point features with a known geoconnex mainstem identifier, checks whether the mainstem has been superseded and, if so, re-indexes the point to the replacement mainstem using index_points_to_lines. Points whose mainstem has no replacement are left unchanged and flagged "unresolved". See check_mainstems for a lightweight superseded/current check.

Usage

```
update_mainstems(x, mainstem_col = NULL, search_radius = NULL)
```

Arguments

x	sf POINT features with a mainstem identifier column.
mainstem_col	character name of the mainstem identifier column in x. Detected automatically from a "mainstemid" or "mainstem_id" column if not provided. Values may be bare ref/mainstems ids or full mainstem uris (any namespace); see check_mainstems.
search_radius	units distance for the nearest neighbor search passed to index_points_to_lines. If NULL, the default in index_points_to_lines is used.

Value

x with mainstem_col updated in place and a mainstem_update_status column added with values "unchanged", "updated", or "unresolved".

Examples

```
pt <- sf::st_sf(mainstemid = 2086165,
                geometry = sf::st_sfc(sf::st_point(c(-75.567, 43.176)),
                                      crs = 4326))
update_mainstems(pt)
```

vpu_boundaries *VPU Boundaries Vector Processing Unit boundaries*

Description

VPU Boundaries Vector Processing Unit boundaries

Usage

```
vpu_boundaries
```

Format

An object of class `"sf"`

Index

∗ **data**
    rpu_boundaries, 58
    vpu_boundaries, 67

add_hydro_location_distance, *65*
add_mainstems, 3, *14*, *26, 27*, *46*, *64, 65*
add_toids, *39*
align_nhdplus_names, 4

check_mainstems, 5, *65, 66*

discover_geoconnex_reference, 6, *23*
discover_nhdplus_id, 6
download_nhd, 7
download_nhdplushr, 8
download_nhdplusv2, 9, *27*, *42*
download_rf1, 10
download_vaa, 10
download_wbd, 11

get_3dhp, 12, *14*, *26*
get_3dhp_index, 14
get_boundaries, 15
get_catchment_characteristics, 16, *57*
get_characteristics_metadata, *16*, 17, *56*
get_confluences, *26*, *65*
get_drainage_area_estimates, 18
get_elev_along_path, 21
get_gagesII, 22
get_geoconnex_reference, 23
get_hr_data, 24, *32*
get_huc, *7, 8*, 25
get_mainstem_flowlines, 26, *63*, *65*
get_nhdarea, 27
get_nhdphr, *26*, 28
get_nhdplus, *26*, 29, *58*
get_nhdplushr, *24*, 31
get_nldi_basin, 32
get_nldi_feature, *6*, *18*, 34
get_nldi_index, 34
get_nldi_sources, *7*, *33, 34*, *51*
get_nwis, 35
get_raindrop_trace, *6, 7*, 36, *37*
get_split_catchment, 37
get_subpath, *63*
get_tiles, *53*
get_tocomid, 38
get_vaa, *18, 19*, *26, 27*, 40, *42*, *46*, *57*
get_vaa_names, 41
get_vaa_path, *11*, *40, 41*, 42, *42*
get_waterbodies, 42
get_wb_outlet, 43
get_xs_point, 44
get_xs_points, 45

hy, *26*
hydrogeofetch_cache_clear, 46, *47*
hydrogeofetch_cache_settings, 47
hydrogeofetch_data_dir, *4*, *11*, *40*, *42*, *46*, 47, *47*

index_hydro_location_distance, *65*
index_points_to_lines, *26*, *63*, *65, 66*

make_standalone, *31*, 48
map_nhdplus, 49

navigate_nldi, 51
nhdplus_path, 52, *59*

plot_nhdplus, *46*, *50*, 53

read_sf, *19*, *23*
rescale_catchment_characteristics, 56
rpu_boundaries, 58

st_crs, *14*, *24*
st_layers, *24*
subset_nhdplus, *50*, *53*, 58
subset_rpu, 61, *62*
subset_vpu, 62

translate_flowlines, 63
translate_hydrolocations, *63*, 64, *64*

update_mainstems, *5*, *65*, 66

vpu_boundaries, 67