Package: tbeploads (via r-universe)

July 23, 2024

Title Calculate Loading Data to Tampa Bay

Version 0.0.0.9000

Description Loading data from major sources to Tampa Bay are calculated on a monthly or annual basis. Major sources include domestic point source (reuse, end of pipe), industrial point source, material losses, non-point sources (MS4), atmospheric deposition, and groundwater.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Depends R (>= 4.1)

Imports dplyr, lubridate, purrr, rnoaa, tibble, tidyr

LazyData true

LazyDataCompression xz

VignetteBuilder knitr

URL <https://github.com/tbep-tech/tbeploads>, <https://tbep-tech.github.io/tbeploads/>

BugReports https://github.com/tbep-tech/tbeploads/issues

Suggests knitr, mockery, rmarkdown, testthat (>= 3.0.0)

Remotes ropensci/rnoaa

Config/testthat/edition 3

Repository https://tbep-tech.r-universe.dev

RemoteUrl https://github.com/tbep-tech/tbeploads

RemoteRef HEAD

RemoteSha 94d34a16c1e39b29be375367102680c0063e3b9c

Contents

`ad_distance` *Data frame of distances of segment locations to National Weather Service (NWS) sites*

Description

Data frame of distances of segment locations to National Weather Service (NWS) sites

Usage

```
ad_distance
```

Format

A `data.frame`

Details

Used for estimating atmospheric deposition. The data frame contains the following columns:

- `segment`: Numeric identifier for the segment location
- `seg_x`: Numeric value for the x-coordinate of the segment location (WGS 84, UTM Zone 17N, CRS 32617)

- seg_y: Numeric value for the y-coordinate of the segment location (WGS 84, UTM Zone 17N, CRS 32617)
- matchsit: Numeric for the NWS site that matches the segment location
- distance: Numeric value for the distance (m) between the segment coordinate and NWS site
- invdist2: Numeric value for the inverse distance squared (1/m^2) between the segment coordinate and NWS site
- area: Numeric value for the area of the segment (ha)

Segment numbers are 1-7 for Old Tampa Bay, Hillsborough Bay, Middle Tampa Bay, Lower Tampa Bay, Boca Ciega Bay, Terra Ceia Bay, and Manatee River.

Examples

```
ad_distance
```

---

ad_rain *Data frame of daily rainfall data from NOAA NCDC National Weather Service (NWS) sites from 2017 to 2023*

Description

Data frame of daily rainfall data from NOAA NCDC National Weather Service (NWS) sites from 2017 to 2023

Usage

```
ad_rain
```

Format

A data.frame

Details

Used for estimating atmospheric deposition and created using the util_ad_getrain function. The data frame contains the following columns:

- station: Character string for the station id
- date: Date for the observation
- Year: Numeric value for the year of the observation
- Month: Numeric value for the month of the observation
- Day: Numeric value for the day of the observation
- rainfall: Numeric value for the amount of rainfall in inches

See Also

util_ad_getrain

Examples

```
ad_rain
```

---

anlz_ad *Calculate AD loads and summarize*

Description

Calculate AD loads and summarize

Usage

```
anlz_ad(
  ad_rain,
  vernafl,
  summ = c("segment", "all"),
  summtime = c("month", "year")
)
```

Arguments

ad_rain	data frame of daily rainfall data from NOAA NCDC, obtained using util_ad_getrain
vernafl	character vector of file path to Verna Wellfield atmospheric concentration data
summ	chr string indicating how the returned data are summarized, see details
summtime	chr string indicating how the returned data are summarized temporally (month or year), see details

Details

Loading from atmospheric deposition (AD) for bay segments in the Tampa Bay watershed are calculated using rainfall data and atmospheric concentration data from the Verna Wellfield site. Rainfall data must be obtained using the util_ad_getrain function before calculating loads. For convenience, daily rainfall data from 2017 to 2023 at sites in the watershed are included with the package in the ad_rain object. The Verna Wellfield data must also be obtained from https://nadp.slh.wisc.edu/sites/ntn-FL41/ as monthly observations. This file is also included with the package and can be found using system.file as in the examples below. Internally, the Verna data are converted to total nitrogen and total phosphorus from ammonium and nitrate concentration data (see util_ad_prepverna for additional information).

The function first estimates the total hydrologic load for each bay segment using daily estimates of rainfall at NWIS NCDC sites in the watershed. This is done as a weighted mean of rainfall at the measured sites relative to grid locations in each sub-watershed for the bay segments. The weights are based on distance of the grid cells from the closest site as inverse distance squared.

Total hydrologic load for a bay segment is then estimated by converting inches/month to m3/month using the segment area. The distance data and bay segment areas are contained in the ad_distance file included with the package.

The total nitrogen and phosphorus loads are then estimated for each bay segment by multiplying the total hydrologic load by the total nitrogen and phosphorus concentrations in the Verna data. The loading calculations also include a wet/dry deposition conversion factor to account for differences in loading during the rainy and dry seasons.

Value

A data frame with nitrogen and phosphorus loads in tons/month, hydrologic load in million m3/month, and segment, year, and month as columns if `summ = 'segment'` and `summtime = 'month'`. Total load to all segments can be returned if `summ = 'all'` and annual summaries can be returned if `summtime = 'year'`. In the latter case, loads are the sum of monthly estimates such that output is tons/yr for TN and TP and as million m3/yr for hydrologic load.

See Also

util_ad_getrain, util_ad_prepverna

Examples

```
vernafl <- system.file('extdata/verna-raw.csv', package = 'tbeploads')
data(ad_rain)
anlz_ad(ad_rain, vernafl)
```

---

anlz_dps *Calculate DPS reuse and end of pipe loads and summarize*

Description

Calculate DPS reuse and end of pipe loads and summarize

Usage

```
anlz_dps(
  fls,
  summ = c("entity", "facility", "segment", "all"),
  summtime = c("month", "year")
)
```

Arguments

fls	vector of file paths to raw entity data, one to many
summ	chr string indicating how the returned data are summarized, see details
summtime	chr string indicating how the returned data are summarized temporally (month or year), see details

Details

Input data files in `fls` are first processed by `anlz_dps_facility` to calculate DPS reuse and end of pipe for each facility and outfall. The data are summarized differently based on the `summ` and `summtime` arguments. All loading data are summed based on these arguments, e.g., by bay segment (`summ = 'segment'`) and year (`summtime = 'year'`).

Value

data frame with loading data for TP, TN, TSS, and BOD as tons per month/year and hydro load as million cubic meters per month/year

See Also

`anlz_dps_facility`

Examples

```
fls <- list.files(system.file('extdata/', package = 'tbeploads'),
  pattern = 'ps_dom', full.names = TRUE)
anlz_dps(fls)
```

---

anlz_dps_facility *Calculate DPS reuse and end of pipe loads from raw facility data*

Description

Calculate DPS reuse and end of pipe loads from raw facility data

Usage

```
anlz_dps_facility(fls)
```

Arguments

`fls`	vector of file paths to raw facility data, one to many

Details

Input data should include flow as million gallons per day, and conc as mg/L. Steps include:

1. Multiply flow by day in month to get million gallons per month
2. Multiply flow by 3785.412 to get cubic meters per month
3. Multiply conc by flow and divide by 1000 to get kg var per month
4. Multiply m3 by 1000 to get L, then divide by 1e6 to convert mg to kg, same as dividing by 1000
5. TN, TP, TSS, BOD dps reuse is multiplied by attenuation factor for land application (varies by location)
6. Hydro load (m3 / mo) is also attenuated for the reuse, multiplied by 0.6 (40% attenuation)

Value

data frame with loading data for TP, TN, TSS, and BOD as tons per month and hydro load as million cubic meters per month. Information for each entity, facility, and outfall is retained.

See Also

anlz_dps

Examples

```
fls <- list.files(system.file('extdata/', package = 'tbeploads'),
  pattern = 'ps_dom', full.names = TRUE)
anlz_dps_facility(fls)
```

---

anlz_ips *Calculate IPS loads and summarize*

Description

Calculate IPS loads and summarize

Usage

```
anlz_ips(
  fls,
  summ = c("entity", "facility", "segment", "all"),
  summtime = c("month", "year")
)
```

Arguments

fls	vector of file paths to raw entity data, one to many
summ	chr string indicating how the returned data are summarized, see details
summtime	chr string indicating how the returned data are summarized temporally (month or year), see details

Details

Input data files in fls are first processed by anlz_ips_facility to calculate IPS loads for each facility and outfall. The data are summarized differently based on the summ and summtime arguments. All loading data are summed based on these arguments, e.g., by bay segment (summ = 'segment') and year (summtime = 'year').

Value

data frame with loading data for TP, TN, TSS, and BOD as tons per month/year and hydro load as million cubic meters per month/year

See Also

anlz_ips_facility

Examples

```
fls <- list.files(system.file('extdata/', package = 'tbeploads'),
  pattern = 'ps_ind_', full.names = TRUE)
anlz_ips(fls)
```

`anlz_ips_facility` *Calculate IPS loads from raw facility data*

Description

Calculate IPS loads from raw facility data

Usage

```
anlz_ips_facility(fls)
```

Arguments

`fls`	vector of file paths to raw facility data, one to many

Details

Input data should include flow as million gallons per day, and conc as mg/L. Steps include:

1. Multiply flow by day in month to get million gallons per month
2. Multiply flow by 3785.412 to get cubic meters per month
3. Multiply conc by flow and divide by 1000 to get kg var per month
4. Multiply m3 by 1000 to get L, then divide by 1e6 to convert mg to kg, same as dividing by 1000

Value

data frame with loading data for TP, TN, TSS, and BOD as tons per month and hydro load as million cubic meters per month. Information for each entity, facility, and outfall is retained.

See Also

anlz_dps

Examples

```
fls <- list.files(system.file('extdata/', package = 'tbeploads'),
  pattern = 'ps_ind_', full.names = TRUE)
anlz_ips_facility(fls)
```

anlz_ml *Calculate material loss (ML) loads and summarize*

Description

Calculate material loss (ML) loads and summarize

Usage

```
anlz_ml(
  fls,
  summ = c("entity", "facility", "segment", "all"),
  summtime = c("month", "year")
)
```

Arguments

fls	vector of file paths to raw entity data, one to many
summ	chr string indicating how the returned data are summarized, see details
summtime	chr string indicating how the returned data are summarized temporally (month or year), see details

Details

Input data files in `fls` are first processed by `anlz_ml_facility` to calculate ML loads for each facility. The data are summarized differently based on the `summ` and `summtime` arguments. All loading data are summed based on these arguments, e.g., by bay segment (`summ = 'segment'`) and year (`summtime = 'year'`).

Value

data frame with loading data for TN as tons per month/year. Columns for TP, TSS, BOD, and hydrologic load are also returned with zero load for consistency with other point source load calculation functions.

See Also

`anlz_ml_facility`

Examples

```
fls <- list.files(system.file('extdata/', package = 'tbeploads'),
  pattern = 'ps_indml', full.names = TRUE)
anlz_ml(fls)
```

`anlz_ml_facility` *Calculate material loss (ML) loads from raw facility data*

Description

Calculate material loss (ML) loads from raw facility data

Usage

```
anlz_ml_facility(fls)
```

Arguments

`fls`	vector of file paths to raw facility data, one to many

Details

Input data should be one row per year per facility, where the row shows the total tons per year of total nitrogen loss. Input files are often created by hand based on reported annual tons of nitrogen shipped at each facility. The material losses as tons/yr are estimated from the tons shipped using an agreed upon loss rate. Values reported in the example files represent the estimated loss as the total tons of N shipped each year multiplied by 0.0023 and divided by 2000. The total N shipped at a facility each year can be obtained using a simple back-calculation (multiply by 2000, divide by 0.0023).

Value

data frame that is nearly identical to the input data except results are shown as monthly load as the annual loss estimate divided by 12. This is for consistency of reporting with other loading sources.

See Also

`anlz_ml`

Examples

```
fls <- list.files(system.file('extdata/', package = 'tbeploads'),
  pattern = 'ps_indml', full.names = TRUE)
anlz_ml_facility(fls)
```

dbasing — *Basin information for coastal subbasin codes*

Description

Basin information for coastal subbasin codes

Usage

```
dbasing
```

Format

A `data.frame`

Details

Used for domestic point source summaries, bay segments are as follows:

- 1: Hillsborough Bay
- 2: Old Tampa Bay
- 3: Middle Tampa Bay
- 4: Lower Tampa Bay
- 5: Boca Ciega Bay
- 6: Terra Ceia Bay
- 7: Manatee River
- 55: Boca Ciega Bay South

Examples

```
dbasing
```

facilities — *Domestic and industrial point source facilities, including industrial with material losses*

Description

Domestic and industrial point source facilities, including industrial with material losses

Usage

```
facilities
```

Format

A `data.frame`

Examples

```
facilities
```

`util_ad_getrain` *Get rainfall data at NOAA NCDC sites*

Description

Get rainfall data at NOAA NCDC sites

Usage

```
util_ad_getrain(yrs, station = NULL, noaa_key, ntry = 5, quiet = FALSE)
```

Arguments

`yrs`	numeric vector for the years of data to retrieve
`station`	numeric vector of station numbers to retrieve, see details
`noaa_key`	character for the NOAA API key
`ntry`	numeric for the number of times to try to download the data
`quiet`	logical to print progress in the console

Details

This function is used to retrieve a long-term record of rainfall for estimating AD loads. It is used to create an input data file for load calculations and it is not used directly by any other functions due to download time. A NOAA API key is required to use the function.

By default, rainfall data is retrieved for the following stations:

- `228`: ARCADIA
- `478`: BARTOW
- `520`: BAY LAKE
- `940`: BRADENTON EXPERIMENT
- `945`: BRADENTON 5 ESE
- `1046`: BROOKSVILLE CHIN HIL
- `1163`: BUSHNELL 2 E
- `1632`: CLEARWATER
- `1641`: CLERMONT 7 S
- `2806`: ST PETERSBURG WHITTD

- 3153: FORT GREEN 12 WSW
- 3986: HILLSBOROUGH RVR SP
- 4707: LAKE ALFRED EXP STN
- 5973: MOUNTAIN LAKE
- 6065: MYAKKA RIVER STATE P
- 6880: PARRISH
- 7205: PLANT CITY
- 7851: ST LEO
- 7886: ST PETERSBURG WHITTD
- 8788: TAMPA INTL ARPT
- 8824: TARPON SPNGS SWG PLT
- 9176: VENICE
- 9401: WAUCHULA 2 N

Value

a data frame with the following columns:

- station: numeric, the station id
- date: Date, the date of the observation
- Year: numeric, the year of the observation
- Month: numeric, the month of the observation
- Day: numeric, the day of the observation
- rainfall: numeric, the amount of rainfall in inches

See Also

ad_rain

Examples

```
## Not run:
noaa_key <- Sys.getenv('NOAA_KEY')
util_ad_getrain(2021, 228, noaa_key)

## End(Not run)
```

`util_ad_prepverna` *Prep Verna Wellfield data for use in AD calculations*

Description

Prep Verna Wellfield data for use in AD calculations

Usage

```
util_ad_prepverna(fl, fillmis = T)
```

Arguments

`fl`	text string for the file path to the Verna Wellfield data
`fillmis`	logical indicating whether to fill missing data with monthly means

Details

Raw data can be obtained from https://nadp.slh.wisc.edu/sites/ntn-FL41/ as monthly observations. Total nitrogen and phosphorus concentrations are estimated from ammonium and nitrate concentrations (mg/L) using the following relationships:

$$TN = NH_4^+ * 0.78 + NO_3^- * 0.23$$

$$TP = 0.01262 * TN + 0.00110$$

The first equation corrects for the % of ions in ammonium and nitrate that is N, and the second is a regression relationship between TBADS TN and TP, applied to Verna.

Value

A data frame with total nitrogen and phosphorus estimates as mg/l for each year and month of the input data

Examples

```
fl <- system.file('extdata/verna-raw.csv', package = 'tbeploads')
util_ad_prepverna(fl)
```

util_ps_addcol *Add column names for point source data from raw entity data*

Description

Add column names for point source from raw entity data

Usage

```
util_ps_addcol(dat)
```

Arguments

dat	data frame from raw entity data as data.frame

Details

The function checks for TN, TP, TSS, and BOD. If any of these are missing, the columns are added with empty values including a column for units. If BOD is missing but CBOD is present, the CBOD column is renamed to BOD.

Value

Input data frame from pth as is if column names are correct, otherwise additional columns are added as needed.

Examples

```
pth <- system.file('extdata/ps_dom_hillsco_falkenburg_2019.txt', package = 'tbeploads')
dat <- read.table(pth, skip = 0, sep = '\t', header = TRUE)
util_ps_addcol(dat)
```

util_ps_checkfls *Create a data frame of formatting issues with point source input files*

Description

Create a data frame of formatting issues with point source input files

Usage

```
util_ps_checkfls(fls)
```

Arguments

fls	vector of file paths to raw facility data, one to many

Details

The chk column indicates the issue with the file and will indicate "ok" if no issues are found, "read error" if the file cannot be read, and "check columns" if the column names are not as expected. Any file not showing "ok" should be checked for issues.

All files are checked with util_ps_checkuni if a file does not have a read error.

The function cannot be used with files for material losses.

Value

A data.frame with three columns indicating name for the file name, chk for the file issue, and nms for a concatenated string of column names for the file

Examples

```
fls <- system.file('extdata/ps_dom_hillsco_falkenburg_2019.txt', package = 'tbeploads')
util_ps_checkfls(fls)
```

---

util_ps_checkuni *Check units for point source from raw entity data*

Description

Check units for point source from raw entity data

Usage

```
util_ps_checkuni(dat)
```

Arguments

dat	data frame from raw entity data as data.frame

Details

Input data should include flow as million gallons per day, and concentration as mg/L.

Value

Input data frame from pth with relevant data and columns renamed, otherwise an error is returned if units are not correct. Only year, month, outfall, flow, TN, TP, TSS, and BOD are returned.

Examples

```
pth <- system.file('extdata/ps_dom_hillsco_falkenburg_2019.txt', package = 'tbeploads')
dat <- read.table(pth, skip = 0, sep = '\t', header = TRUE)
util_ps_checkuni(dat)
```

util_ps_facinfo *Get point source entity information from file name*

Description

Get point source entity information from file name

Usage

```
util_ps_facinfo(pth, asdf = FALSE)
```

Arguments

pth	path to raw entity data
asdf	logical, if TRUE return as data.frame

Details

Bay segment is an integer with values of 1, 2, 3, 4, 5, 6, 7, and 55 for Old Tampa Bay, Hillsborough Bay, Middle Tampa Bay, Lower Tampa Bay, Boca Ciega Bay, Terra Ceia Bay, Manatee River, and Boca Ciega Bay South, respectively.

Value

A list or data.frame (if asdf = TRUE) with entity, facility, permit, facility id, coastal id, and coastal subbasin code

Examples

```
pth <- system.file('extdata/ps_dom_hillsco_falkenburg_2019.txt', package = 'tbeploads')
util_ps_facinfo(pth)
```

util_ps_fillmis *Fill missing point source data with annual average*

Description

Fill missing point source data with annual average

Usage

```
util_ps_fillmis(dat)
```

Arguments

dat	data frame from raw entity data as data.frame

Details

Missing concentration data are replaced with the average for the outfall in a given year. All flow data are also floored at zero. Rows with missing flow data are assigned 0 for all data. Rows with zero flow are assigned concentration of zero.

Value

Input data frame as is if no missing values, otherwise missing data filled as described above.

Examples

```
pth <- system.file('extdata/ps_dom_hillsco_falkenburg_2019.txt', package = 'tbeploads')
dat <- read.table(pth, skip = 0, sep = '\t', header = TRUE)
dat <- util_ps_checkuni(dat)
util_ps_fillmis(dat)
```

---

util_ps_fixoutfall *Light edits to the outfall ID column for point source data*

Description

Light edits to the outfall ID column for point source data

Usage

```
util_ps_fixoutfall(dat)
```

Arguments

dat	data frame from raw entity data as data.frame

Details

The outfall ID column is edited lightly to remove any leading or trailing white space, a hyphen is added between letters and numbers if missing, and "Outfall" prefix is removed if presenn.

Value

Input data frame as is, with any edits to the outfall ID column.

Examples

```
pth <- system.file('extdata/ps_ind_busch_busch_2020.txt', package = 'tbeploads')
dat <- read.table(pth, skip = 0, sep = '\t', header = TRUE)
util_ps_fixoutfall(dat)
```

util_ps_summ *Summarize point source load estimates*

Description

Summarize point source load estimates

Usage

```
util_ps_summ(
  dat,
  summ = c("entity", "facility", "segment", "all"),
  summtime = c("month", "year")
)
```

Arguments

dat	Pre-processed data frame of point source load estimates, see examples
summ	chr string indicating how the returned data are summarized, see details
summtime	chr string indicating how the returned data are summarized temporally (month or year), see details

Details

The data are summarized differently based on the summ and summtime arguments. All loading data are summed based on these arguments, e.g., by bay segment (summ = 'segment') and year (summtime = 'year').

Value

Data frame with summarized loading data based on user-supplied arguments

Examples

```
fls <- list.files(system.file('extdata/', package = 'tbeploads'),
  pattern = 'ps_ind_', full.names = TRUE)

ipsbyfac <- anlz_ips_facility(fls)

# add bay segment and source, there should only be loads to hills, middle, and lower tampa bay
ipsld <- ipsbyfac  |>
  dplyr::arrange(coastco) |>
  dplyr::left_join(dbasing, by = "coastco") |>
  dplyr::mutate(
    segment = dplyr::case_when(
      bayseg == 1 ~ "Old Tampa Bay",
      bayseg == 2 ~ "Hillsborough Bay",
      bayseg == 3 ~ "Middle Tampa Bay",
```

```
      bayseg == 4 ~ "Lower Tampa Bay",
      TRUE ~ NA_character_
    ),
    source = 'IPS'
  ) |>
  dplyr::select(-basin, -hectare, -coastco, -name, -bayseg)

util_ps_summ(ipsld, summ = 'entity', summtime = 'year')
```

Index