Package: tbeploads (via r-universe)

October 11, 2026

Title Calculate Loading Data to Tampa Bay

Version 1.0.0.9000

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 7.3.3

Depends R (>= 4.2)

Imports dataRetrieval, dplyr, flextable, ggplot2, httr, jsonlite, lubridate, openxlsx, pdftools, purrr, readxl, sf, tbeptools, terra, tibble, tidyr, zoo

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)

SystemRequirements GDAL (>= 2.0.0)

Remotes tbep-tech/tbeptools

Config/testthat/edition 3

Config/pak/sysreqs libabsl-dev libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev libfribidi-dev libgdal-dev gdal-bin libgeos-dev make libharfbuzz-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libwebp-dev libxml2-dev libssl-dev libpoppler-cpp-dev poppler-data libproj-dev libsqlite3-dev libudunits2-dev libx11-dev

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

Date/Publication 2026-10-11 19:36:18 UTC

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

RemoteRef HEAD

RemoteSha 93ad206e4fc82f9c88082055bfb4f347ef16c2c8

Contents

aa_corrections . . . 4
ad_distance . . . 5
allflo . . . 6
allwq . . . 7
anlz_aa . . . 8
anlz_ad . . . 12
anlz_dps . . . 14
anlz_dps_facility . . . 15
anlz_gw . . . 16
anlz_ips . . . 18
anlz_ips_facility . . . 19
anlz_ml . . . 20
anlz_ml_facility . . . 21
anlz_nps . . . 22
anlz_nps_gaged . . . 24
anlz_nps_psremove . . . 26
anlz_nps_ungaged . . . 28
anlz_spr . . . 31
clucsid . . . 34
conserv_correction . . . 35
contdry . . . 36
contwet . . . 37
dbasing . . . 38
dps_allocations . . . 39
emc . . . 40
facilities . . . 40
hydro_baseline . . . 41
ml_allocations . . . 41
nps_allocations . . . 42
nps_distance . . . 43
ps_allocations . . . 44
rain . . . 45
rain_historic . . . 46
rcclucsid . . . 46
show_aaassess . . . 47
show_aaloads . . . 48
tbbase . . . 50
tbdbasin . . . 51
tbfullshed . . . 52
tbjuris . . . 53

tblu2023 . . . 54
tbmines . . . 55
tbsegdetail . . . 56
tbsoil . . . 57
tbsubshed . . . 58
usgsflow . . . 59
util_aa_npsfactors . . . 60
util_getrain . . . 61
util_gw_getcontour . . . 63
util_gw_getwq . . . 64
util_gw_grad . . . 66
util_gw_showgrad . . . 68
util_nps_fillmiswq . . . 69
util_nps_getextflow . . . 70
util_nps_getflow . . . 71
util_nps_getswfwmd . . . 73
util_nps_getusgsflow . . . 74
util_nps_getwq . . . 75
util_nps_landsoil . . . 76
util_nps_landsoilrc . . . 77
util_nps_lusumm . . . 77
util_nps_preplog . . . 78
util_nps_preprain . . . 79
util_nps_segment . . . 80
util_nps_tbbase . . . 80
util_nps_union . . . 82
util_nps_unionchunk . . . 83
util_nps_unionnochunk . . . 84
util_prepverna . . . 85
util_ps_addcol . . . 86
util_ps_checkfls . . . 87
util_ps_checkuni . . . 88
util_ps_facinfo . . . 88
util_ps_fillmis . . . 89
util_ps_fixoutfall . . . 90
util_ps_getaoc . . . 90
util_ps_getkerry . . . 92
util_ps_getmacdill . . . 95
util_ps_misc . . . 98
util_ps_mosaic . . . 100
util_ps_pascoreuse . . . 102
util_spr_getwq . . . 103
util_summ . . . 104

Index **106**

`aa_corrections` *Allocation assessment TN load corrections by bay segment and entity*

Description

Allocation assessment TN load corrections by bay segment and entity

Usage

```
aa_corrections
```

Format

A `data.frame` with 43 rows and 4 columns:

bay_seg Integer bay segment identifier

entity MS4 jurisdiction or entity name

ad_tons Atmospheric deposition TN offset (tons/yr)

project_tons Net permitted project TN offset (tons/yr); negative values indicate a load credit

Details

TN load offsets applied before hydrologic normalization in `anlz_aa` for the 2022-2024 TB-NMC assessment period. Values are sourced from the SAS script `7_Basin_assessment2224.sas` and cover two correction types: atmospheric deposition (AD) loads apportioned to each entity jurisdiction, and net permitted project (AP) load credits. FDACS agriculture entries (`entity = "All"`) carry irrigation AP reductions only (`ad_tons = 0`). Negative `project_tons` values reflect project credits that increase the allowable load.

Updating these numbers to account for additional changes in atomospheric deposition concentrations (i.e., large changes that have occurred at Bayside and Apollo Beach power plants) and the many projects completed since the late 2000s should be done for future package updates. Numbers herein have been used to date for the allocation assessment tables. They likely do not represent current conditions.

See `data-raw/aa_corrections.R` for construction.

Examples

```
aa_corrections
```

`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
```

`allflo` *Data frame of all flow data used in* `anlz_nps_gaged` *and* `anlz_nps_ungaged`

Description

Data frame of all flow data used in `anlz_nps_gaged` and `anlz_nps_ungaged`

Usage

```
allflo
```

Format

A `data.frame` of monthly mean daily flow data for select basins

Details

Monthly flow data at select stations used for estimating non-point source gaged and ungaged loads. Created using the `util_nps_getflow` function. Includes data from the USGS API using `util_nps_getusgsflow` and from external sources using `util_nps_getextflow`. The data frame contains the following columns:

- `basin`: Character string for the basin or gauge location
- `yr`: Year of the observation
- `mo`: Month of the observation
- `flow_cfs`: Numeric value for the average daily flow in cubic feet per second (cfs)

See Also

`util_nps_getusgsflow`, `util_nps_getextflow`, `util_nps_getflow`

Examples

```
allflo
```

`allwq` *Data frame of all water quality data used in* `anlz_nps_gaged`

Description

Data frame of all water quality data used in `anlz_nps_gaged`

Usage

```
allwq
```

Format

A `data.frame` of monthly mean water quality data for select stations

Details

Monthly water quality data for select stations used for estimating non-point source gaged loads. Created using the `util_nps_getwq` function. Includes data from Manatee, Pinellas, and Hillsborough (EPCHC) counties. The data frame contains the following columns:

- `basin`: Character string for the basin or station location
- `yr`: Year of the observation
- `mo`: Month of the observation
- `tn_mgl`: Numeric value for Total Nitrogen in mg/L
- `tp_mgl`: Numeric value for Total Phosphorus in mg/L
- `tss_mgl`: Numeric value for Total Suspended Solids in mg/L
- `bod_mgl`: Numeric value for Biochemical Oxygen Demand in mg/L

See Also

`util_nps_getwq`

Examples

```
allwq
```

`anlz_aa` *Allocation assessment for DPS, IPS, and NPS/MS4 entities*

Description

Allocation assessment for DPS, IPS, and NPS/MS4 entities

Usage

```
anlz_aa(
  yrrng,
  dps_data,
  ips_data,
  ml_data,
  nps_data,
  tbbase,
  verbose = FALSE,
  annavg = TRUE
)
```

Arguments

`yrrng`	Integer vector of length 2, start and end year, e.g., `c(2022, 2024)`.
`dps_data`	Data frame from `anlz_dps_facility`. Required columns: `Year`, `Month`, `entity`, `facility`, `coastco`, `tn_load`.
`ips_data`	Data frame from `anlz_ips_facility`. Required columns: `Year`, `Month`, `entity`, `facility`, `coastco`, `tn_load`.
`ml_data`	Data frame from `anlz_ml_facility`. Required columns: `Year`, `Month`, `entity`, `facility`, `tn_load`.
`nps_data`	Data frame from `anlz_nps` called with `summ = 'basin'` and `summtime = 'year'`. Required columns: `Year`, `source`, `segment`, `basin`, `tn_load`, `hy_load`. For gaged basins these are gauge-measured totals that include upstream point-source discharge; `anlz_aa` removes the IPS/DPS contribution from `tn_load` internally before disaggregating to MS4 entities (see Details).
`tbbase`	data frame containing polygon areas for the combined data layer of bay segment, basin, jurisdiction, land use data, and soils, see details
`verbose`	logical, if `TRUE` print a message reporting negligible unmatched NPS/MS4 entities dropped from the output (see Details). Default `FALSE`.
`annavg`	logical, if `TRUE` (default) loads are averaged over `yrrng` as in prior versions of this function. If `FALSE`, one row per entity/facility per year is returned instead, with an additional `year` column, and `load_tons`/`eff_load_tons` are that year's value rather than a `yrrng`-averaged one (see Details).

Details

Entities present in the computed loads but absent from the allocation tables are retained in the output with `NA` allocation fields so that unmatched entries are visible for troubleshooting, with one exception: unmatched NPS/MS4 entities with a mean annual load under 0.01 tons/yr are dropped, since these are negligible land-use polygon artifacts (e.g. land in `tbbase` not attributed to any jurisdiction, or a jurisdiction's boundary crossing into an adjacent basin/segment where it has no allocation) rather than real troubleshooting signal. When `verbose = TRUE`, a message reports what was dropped and why.

When `annavg = FALSE`, each of the four paths below retains one row per entity/facility per year instead of averaging over `yrrng`: a year with no measured load for an entity/facility that has data in at least one other year of `yrrng` is zero-filled (mirroring the averaged path's existing "missing years contribute zero" behavior, just applied per year rather than to the mean); an entity/facility with a real allocation but no measured load in *any* year of `yrrng` still appears, once per year, with `NA` loads (mirroring how it already appears once, with `NA` loads, when `annavg = TRUE`). `alloc_pct/alloc_tons/ishared/group_id` do not vary by year and are repeated identically on every year-row for a given entity/facility.

When `annavg = FALSE`, two additional bay-segment-level totals are computed once and repeated on every row sharing a `bay_seg/year` (analogous to how `alloc_tons` repeats for `ishared` groups): `seg_h2o_total` is the same NPS+IPS+DPS `total_h2o` already computed internally for entity-level hydrologic normalization (see the NPS/IPS paths below), summed across every basin in the segment instead of just an entity's own basins; `seg_conserv_tn` is the TN removed by the conservation-land correction (see `conserv_correction` in the NPS/MS4 path below), summed the same way. Both support downstream segment-total reporting (e.g. `show_aaloads`) that needs a whole-segment hydrologic budget rather than an entity-specific one.

DPS path

DPS facility TN loads require no hydrologic normalization. Monthly loads from `dps_data` are summed to annual totals per facility, averaged over `yrrng`, and compared directly against the `dps_allocations` table. The join key is `entity + facname + bay\_seg + source`, where `source` distinguishes direct surface water discharge (`"DPS - end of pipe"`) from reclaimed water reuse (`"DPS - reuse"`). Bay segment 5 (Boca Ciega Bay) is excluded and bayseg 6/7 are remapped to 55. Coastal subbasin 580 straddles Boca Ciega Bay and Boca Ciega Bay South, so only 41.1\ water (St. Petersburg reuse) is assigned to segment 55. The remaining 58.9\

IPS path

Raw facility loads are joined to facility metadata on `entity + facname` (not `coastco`), since several distinct permits share a single coastco. Monthly loads are summed to annual totals per permit per bay segment and averaged over `yrrng`. Hydrologic normalization is applied only to IPS facilities flagged `hydro_affected` in `ps_allocations` (mostly Mosaic mining operations); all other facilities use their raw (unnormalized) load. For `hydro_affected` permits:

$$\text{eff_tn} = \text{tn_load} \times \frac{\text{mean_h2o_9294}}{\text{basin_total_h2o}}$$

where `basin\_total\_h2o` is the annual total basin water load for the same basin and year, computed differently depending on whether the basin is gaged (per `dbasing`): for

gaged basins, NPS water is estimated from a stream gauge and so already reflects any upstream IPS + DPS discharge, so basin_total_h2o is the NPS water alone (adding IPS/DPS water again would double-count it); for ungaged basins, the modeled NPS-only water excludes point-source discharge entirely, so IPS and DPS water are added to it to reconstruct the true total. All other IPS facilities, and any facility with no ps_allocations match, use the raw (unnormalized) load.

Permits with a real ps_allocations entry but no current ips_data (permanently closed facilities) still receive their known bay_seg from facilities where available, so they appear in the output with NA loads but a real bay segment rather than NA throughout.

Permits flagged ishared = TRUE in ps_allocations carry their group's collective allocation in alloc_tons rather than an individual one, while load_tons/eff_load_tons remain each permit's own load — see ishared in Returns. ishared and hydro_affected are independent: a permit can be jointly assessed without being hydrologically normalized, or vice versa. group_id identifies which shared group a permit belongs to (NA when ishared is FALSE).

ML path

Material loss TN loads require no hydrologic normalization. Monthly loads from ml_data are summed to annual totals per facility, averaged over yrrng, and joined to the ml_allocations table on entity + facname + bay segment — one output row per facility, always. Facilities with ishared = TRUE carry their group's collective allocation in alloc_tons rather than an individual one, while load_tons/eff_load_tons remain each facility's own load — see ishared in Returns. group_id identifies which shared group a facility belongs to (NA when ishared is FALSE).

NPS/MS4 path

Gaged-basin TN loads in nps_data are gauge-measured totals and so include any upstream IPS + DPS discharge in that basin. Before disaggregation, anlz_aa subtracts the basin's IPS and DPS TN loads from gaged-basin tn_load so that only the true non-point-source contribution is assigned to MS4 entities. Ungaged-basin tn_load is already NPS-only (the modeled estimate never includes point-source discharge) and is left unchanged. Basin-level NPS loads (post-correction) are disaggregated to individual MS4 entities using the output (created internally) from util_aa_npsfactors that combines tbbase, rcclucsid, and emc into:

1. factor_tn distributes basin TN load among land use classes.
2. factor_rc distributes each land use class's load among entities proportional to area × runoff coefficient.

Before summing across CLUCSIDs, each entity's disaggregated TN load is scaled by (1 - conserv_frac) using conserv_correction, which provides entity- and CLUCSID-specific fractions of area times runoff coefficient attributable to conservation land. This removes the conservation land contribution that is absent from the tbeploads-built tbbase, and is applied using the true underlying MS4 jurisdiction since conservation land can occur within Agriculture-classified parcels too.

Only after this correction is agricultural land use (category "Agriculture") attributed to the aggregate entity "All" regardless of the underlying MS4 jurisdiction. Land under a Municipal Separate storm sewer Generic Permit (entities "MSGP COT" and "MSGP PINELLAS"

in tbbase) is not part of any individually-tracked MS4 jurisdiction and is aggregated the same way to entity "Non-MS4/Ag NPS", matching the row label used in the TBNMC draft loading tables. "PORT MANATEE"'s footprint spans both Middle Tampa Bay (bay_seg 3) and Lower Tampa Bay (bay_seg 4), so it also folds into this aggregate in both segments; Lower Tampa Bay has no other MSGP entities, so Port Manatee is its sole contributor.

After disaggregation, loads and 1992-1994 baseline water volumes are summed across basins to the segment level. TN corrections from aa_corrections (ad_tons + project_tons) are subtracted before hydrologic normalization:

$$\text{eff_tn} = (\text{tn_entity} - \text{corr_tons}) \times \frac{\text{mean_h2o_9294}}{\text{total_h2o}}$$

total_h2o is the same gaged/ungaged-gated basin water quantity described in the IPS path below (NPS water alone for gaged basins; NPS + IPS + DPS for ungaged basins).

Bay segments Terra Ceia Bay (6) and Manatee River (7) are merged into segment 55 (Remaining Lower Tampa Bay) after disaggregation, consistent with the hydro_baseline encoding and TBNMC reporting. Boca Ciega Bay (segment 5) is excluded from the allocation framework.

Value

A data frame with one row per entity (NPS/MS4) or facility (IPS) per bay segment (or per bay segment per year when annavg = FALSE):

bay_seg Integer bay segment identifier

segment Bay segment name

entity MS4 entity name or facility operator

entity_full Full entity name from nps_allocations (NPS rows only)

facname Facility name (IPS and DPS rows; also ML rows, one per facility whether shared or not)

permit NPDES permit number (IPS rows only)

source Allocation type: "MS4", "Nonpoint Source/MS4", "IPS", "DPS - end of pipe", "DPS - reuse", or "ML"

alloc_pct Fractional TN allocation (0-1)

alloc_tons Allocation in TN tons per year. For ishared rows, this is the group's collective allocation (the same value repeated on every member row), not an individual facility allocation

eff_load_tons Mean hydrologically-normalized TN load (tons/yr), averaged over yrrng; equals load_tons for DPS and ML (no normalization applied) and for IPS facilities not flagged hydro_affected in ps_allocations. NPS/MS4 rows and hydro_affected IPS facilities are normalized

load_tons Mean annual TN load (tons/yr) without hydrologic normalization, averaged over yrrng; always the facility's own individual load, even for ishared rows

ishared Logical: TRUE when the row's facility is jointly assessed against a collective allocation shared with other facilities (see alloc_tons), rather than its own individual allocation. FALSE for NPS/MS4 and DPS rows, which have no shared-group concept currently

group_id Character identifier for the shared group a row belongs to when `ishared` is `TRUE` (`NA` otherwise), from `ps_allocations`/`ml_allocations`.

year Integer year. Only present when `annavg = FALSE`, in which case there is one row per entity/facility per year in `yrrng` instead of one `yrrng`-averaged row

seg_h2o_total Total hydrologic load (NPS+IPS+DPS combined, million m3/yr) for the bay segment and year, repeated on every row sharing that `bay_seg`/`year` (see Details). Only present when `annavg = FALSE`

seg_conserv_tn Total TN (tons/yr) removed from NPS/MS4 entities by the conservation-land correction (see `conserv_correction`), summed for the bay segment and year and repeated on every row sharing that `bay_seg`/`year` (see Details). Only present when `annavg = FALSE`

Examples

```
## Not run:
fls_dps <- list.files(system.file("extdata/", package = "tbeploads"),
  pattern = "ps_dom_", full.names = TRUE)
dps <- anlz_dps_facility(fls_dps)
fls_ips <- list.files(system.file("extdata/", package = "tbeploads"),
  pattern = "ps_ind_", full.names = TRUE)
ips <- anlz_ips_facility(fls_ips)
fls_ml <- list.files(system.file("extdata/", package = "tbeploads"),
  pattern = "ps_indml", full.names = TRUE)
ml <- anlz_ml_facility(fls_ml)
nps <- anlz_nps(
  yrrng  = c("2022-01-01", "2024-12-31"),
  tbbase = tbbase,
  rain   = rain,
  allwq  = allwq,
  allflo = allflo,
  vernafl = system.file("extdata/verna-raw.csv", package = "tbeploads"),
  summ     = "basin",
  summtime = "year"
)

anlz_aa(c(2022, 2024), dps, ips, ml, nps, tbbase)

## End(Not run)
```

`anlz_ad` *Calculate AD loads and summarize*

Description

Calculate AD loads and summarize

Usage

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

Arguments

`rain`	data frame of daily rainfall data from NOAA NCDC, obtained using `util_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_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 `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_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 former case, the total excludes the northern portion of Boca Ciega Bay that is not included in the reasonable assurance boundaries. 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_getrain`, `util_prepverna`

Examples

```
vernafl <- system.file('extdata/verna-raw.csv', package = 'tbeploads')
data(rain)
anlz_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", "basin", "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'`). Options for `summ` are `'entity'` to summarize by entity, `'facility'` to summarize by facility, `'basin'` to summarize by drainage basin (retains the `basin` column for use with `anlz_nps_psremove`), `'segment'` to summarize by bay segment, and `'all'` to summarize across all segments.

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_gw — *Calculate groundwater loads to Tampa Bay segments*

Description

Calculate groundwater loads to Tampa Bay segments

Usage

```
anlz_gw(
  pot_dry,
  pot_wet,
  yrrng = c(2022, 2024),
  wqdat = NULL,
  summtime = c("month", "year")
)
```

Arguments

`pot_dry`	SpatRaster (or `PackedSpatRaster`) of Upper Floridan Aquifer potentiometric head for the dry season, as returned by `util_gw_getcontour` with `season = "dry"`. The package dataset `contdry` contains a pre-computed 2022 example.
`pot_wet`	SpatRaster (or `PackedSpatRaster`) of Upper Floridan Aquifer potentiometric head for the wet season, as returned by `util_gw_getcontour` with `season = "wet"`. The package dataset `contwet` contains a pre-computed 2022 example.
`yrrng`	integer vector of length 2, start and end year for the load estimates, e.g. `c(2022, 2024)`. The same gradients derived from `pot_dry` and `pot_wet` are applied to every year in the range.
`wqdat`	data frame of Floridan aquifer TN and TP concentrations (mg/L) as returned by `util_gw_getwq`, with columns `bay_seg`, `tn_mgl`, and `tp_mgl`. When `NULL` (default), hardcoded concentrations from the 2022-2024 loading analysis are used.
`summtime`	character, temporal summarization: `'month'` (default) returns one row per segment per month, `'year'` sums to annual totals.

Details

Estimates groundwater loads to each Tampa Bay segment for three aquifer layers following the methodology in Zarbock et al. (1994).

Floridan aquifer: Flow is computed with Darcy's Law:

$$Q = 7.4805 \times 10^{-6} \cdot T \cdot I \cdot L$$

where T is transmissivity (ft^2/day), I is the hydraulic gradient (ft/mile), and L is the flow zone length (miles). Q is in MGD. Nutrient loads (kg/month) are:

$$\text{load} = Q \cdot C \cdot 8.342 \cdot 30.5/2.2$$

where C is the TN or TP concentration (mg/L). Hydrologic load (m^3/month) is $Q \cdot 3785 \cdot 30.5$.

Hydraulic gradients: Gradients are computed once from `pot_dry` and `pot_wet` via `util_gw_grad` and applied to every year in `yrrng`. Update `pot_dry` and `pot_wet` with fresh outputs from `util_gw_getcontour` when new FDEP potentiometric surface maps become available. See `util_gw_grad` for details on search areas, zero-gradient segments, and benchmark warnings.

Surficial and intermediate aquifers: Loads are fixed constants per segment. Surficial values are from `gwupdate95-98_final.xls` (1995-1998 SWFWMD monitoring data). Intermediate values are means from SWFWMD monitoring over 1999-2003. These have not changed since the original analysis.

Season assignment: Months 1-6 and 11-12 are dry season; months 7-10 are wet season.

Value

A data frame with columns:

- `Year`: integer
- `Month`: integer (omitted when `summtime = 'year'`)
- `source`: character, `"GW"`
- `segment`: character, bay segment name
- `tn_load`: numeric, total nitrogen load (tons/month or tons/year)
- `tp_load`: numeric, total phosphorus load (tons/month or tons/year)
- `hy_load`: numeric, hydrologic load (million m^3/month or million m^3/year)

References

Zarbock, H., A. Janicki, D. Wade, D. Heimbuch, and H. Wilson. 1994. Estimates of Total Nitrogen, Total Phosphorus, and Total Suspended Solids Loadings to Tampa Bay, Florida. Technical Publication #04-94. Prepared by Coastal Environmental, Inc. Prepared for Tampa Bay National Estuary Program. St. Petersburg, FL.

Examples

```
# contdry and contwet are pre-computed 2022 package datasets
gw <- anlz_gw(contdry, contwet, yrrng = c(2022, 2024))
head(gw)

# annual totals
anlz_gw(contdry, contwet, yrrng = c(2022, 2024), summtime = 'year')

## Not run:
# update rasters from FDEP for a new year, then compute loads
pot_dry <- util_gw_getcontour("dry", 2025)
pot_wet <- util_gw_getcontour("wet", 2025)
gw <- anlz_gw(pot_dry, pot_wet, yrrng = c(2025, 2025))

# pass concentrations from the Water Atlas API
gw <- anlz_gw(pot_dry, pot_wet, yrrng = c(2025, 2025),
              wqdat = util_gw_getwq())

## End(Not run)
```

---

anlz_ips *Calculate IPS loads and summarize*

Description

Calculate IPS loads and summarize

Usage

```
anlz_ips(
  fls,
  summ = c("entity", "facility", "basin", "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'`). Options for `summ` are

'entity' to summarize by entity, 'facility' to summarize by facility, 'basin' to summarize by drainage basin (retains the basin column for use with anlz_nps_psremove), 'segment' to summarize by bay segment, and 'all' to summarize across all segments.

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'`). Options for `summ` are 'entity' to summarize by entity only, 'facility' to summarize by facility only, 'segment' to summarize by bay segment, and 'all' to summarize total load. Options for `summtime` are 'month' to summarize by month and 'year' to summarize by year. The default is to summarize by entity and month.

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)
```

`anlz_nps` *Calculate non-point source (NPS) loads for Tampa Bay*

Description

Calculate non-point source (NPS) loads for Tampa Bay

Usage

```
anlz_nps(
  yrrng = c("2021-01-01", "2023-12-31"),
  tbbase,
  rain,
  mancopth = NULL,
  pincopth = NULL,
  lakemanpth = NULL,
  tampabypth = NULL,
  bellshlpth = NULL,
  vernafl,
  allflo = NULL,
  allwq = NULL,
  usgsflow = NULL,
  summ = c("basin", "segment", "all"),
  summtime = c("month", "year"),
  aslu = FALSE,
  verbose = TRUE
)
```

Arguments

`yrrng`	A vector of two dates in 'YYYY-MM-DD' format, specifying the date range to retrieve flow data. Default is from '2021-01-01' to '2023-12-31'.
`tbbase`	data frame containing polygon areas for the combined data layer of bay segment, basin, jurisdiction, land use data, and soils, see details
`rain`	data frame of rainfall data, see details
`mancopth`	character, path to the Manatee County water quality data file, see details
`pincopth`	character, path to the Pinellas County water quality data file, see details
`lakemanpth`	character, path to the file containing the Lake Manatee flow data, see details
`tampabypth`	character, path to the file containing the Tampa Bypass flow data, see details
`bellshlpth`	character, path to the file containing the Bell shoals data, see details
`vernafl`	character vector of file path to Verna Wellfield atmospheric concentration data

`allflo`	data frame of flow data, if already available from `util_nps_getflow`, otherwise NULL and the function will retrieve the data
`allwq`	data frame of water quality data, if already available from `util_nps_getwq`, otherwise NULL and the function will retrieve the data.
`usgsflow`	data frame of USGS flow data, if already available from `util_nps_getusgsflow`, otherwise NULL and the function will retrieve the data. Default is NULL.
`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
`aslu`	logical indicating whether to summarize by land use type (ungaged loads only), default is FALSE
`verbose`	logical indicating whether to print verbose output

Details

The function estimates non-point source (NPS) loads for Tampa Bay by combining ungaged and gaged NPS loads. Ungaged loads are estimated using rainfall, flow, event mean concentration, land use, and soils data, while gaged loads are estimated using water quality data and flow data. The function also incorporates atmospheric concentration data from the Verna Wellfield site.

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'`). Options for `summ` are 'basin' to summarize across sub-basins within bay segments, 'segment' to summarize by bay segment, and 'all' to summarize total load. Loads can also be summarized by land use type with the `summ` and `summtime` argumets by setting `aslu = TRUE`. Land use type summaries only apply to ungaged load estimates. Options for `summtime` are 'month' to summarize by month and 'year' to summarize by year. The default is to summarize by basin and month.

The following functions are used internally and are provided here for reference on the components used in the calculations:

- `anlz_nps_ungaged`: Estimates ungaged NPS loads.
- `anlz_nps_gaged`: Estimates gaged NPS loads.
- `util_nps_fillmiswq`: Fills missing water quality data with linear interpolation.
- `util_nps_getflow`: Gets flow estimates for NPS gaged and ungaged calculations.
- `util_nps_getusgsflow`: Gets USGS flow data for NPS calculations, used in `util_nps_getflow`.
- `util_nps_getextflow`: Gets external flow data (Lake Manatee, Tampa Bypass, and Bell Shoals), used in `util_nps_getflow`.
- `util_nps_getwq`: Gets water quality data for NPS gaged calculations.
- `util_nps_preprain`: Prepares and formats rainfall data.
- `util_nps_preplog`: Prepares land use data for logistic regression modeling.
- `util_nps_segment`: Assigns basins to bay segments.
- `util_prepverna`: Prepares and fills missing data with five-year means for the Verna Wellfield site data.

Value

A data frame of non-point source loads for Tampa Bay, including columns for year, month, bay segment, basin, and loads for total nitrogen (TN), total phosphorus (TP), total suspended solids (TSS), biochemical oxygen demand (BOD), and hydrology using default values for the `summ` and `summtime` arguments. TN, TP, TSS, and BOD Loads are tons per month or year depending on the `summtime` argument. Hydrologic loads are million cubic meters per month or year depending on the `summtime` argument.

Examples

```
data(tbbase)
data(rain)
data(allwq)
data(allflo)
vernafl <- system.file('extdata/verna-raw.csv', package = 'tbeploads')

nps <- anlz_nps(
  yrrng = c('2021-01-01', '2023-12-31'),
  tbbase = tbbase,
  rain = rain,
  allwq = allwq,
  allflo = allflo,
  vernafl = vernafl,
)

head(nps)
```

---

anlz_nps_gaged — *Calculate non-point source (NPS) loads for gaged basins*

Description

Calculate non-point source (NPS) loads for gaged basins

Usage

```
anlz_nps_gaged(
  yrrng = c("2021-01-01", "2023-12-31"),
  mancopth = NULL,
  pincopth = NULL,
  lakemanpth = NULL,
  tampabypth = NULL,
  bellshlpth = NULL,
  allflo = NULL,
  allwq = NULL,
  usgsflow = NULL,
  verbose = TRUE
)
```

Arguments

`yrrng`	A vector of two dates in 'YYYY-MM-DD' format, specifying the date range to retrieve flow data. Default is from '2021-01-01' to '2023-12-31'.
`mancopth`	character, path to the Manatee County water quality data file, see details
`pincopth`	character, path to the Pinellas County water quality data file, see details
`lakemanpth`	character, path to the file containing the Lake Manatee flow data
`tampabypth`	character, path to the file containing the Tampa Bypass flow data
`bellshlpth`	character, path to the file containing the Bell shoals data
`allflo`	data frame of flow data, if already available from `util_nps_getflow`, otherwise NULL and the function will retrieve the data
`allwq`	data frame of water quality data, if already available from `util_nps_getwq`, otherwise NULL and the function will retrieve the data.
`usgsflow`	data frame of USGS flow data, if already available from `util_nps_getusgsflow`, otherwise NULL and the function will retrieve the data. Default is NULL. Does not apply if `allflo` is provided.
`verbose`	logical indicating whether to print verbose output

Details

The function uses `util_nps_getflow` to retrieve flow data and `util_nps_getwq` to retrieve water quality data. It then combines these datasets and calculates loads for TN, TP, TSS, BOD, and hydrologic load. See the help files for each function for more details.

Required external data inputs are Lake Manatee, Tampa Bypass, and Alafia River Bell Shoals flow data. These are not available from the USGS API and must be obtained from the contacts listed in `util_nps_getextflow`. USGS flow data are for stations 02299950, 02300042, 02300500, 02300700, 02301000, 02301300, 02301500, 02301750, 02303000, 02303330, 02304500, 02306647, 02307000, 02307359, and 02307498. The USGS flow data are from the NWIS database as returned by `read_waterdata_daily` using `util_nps_getusgsflow`. A preprocessed USGS flow data frame can be provided using the `usgsflow` argument to avoid re-downloading the data. All inputs for flow can be superseded by providing a complete flow data frame using the `allflo` argument.

Water Quality data are obtained from the FDEP WIN database API, tbeptools, or local files as described in `util_nps_getwq`. Chosen stations are ER2 and UM2 for Manatee County and station 06-06 for Pinellas County. Environmental Protection Commission (EPC) of Hillsborough County stations retained are 105, 113, 114, 132, 141, 138, 142, and 147. Manatee or Pinellas County data can be imported from local files using the `mancopth` and `pincopth` arguments, respectively. If these are not provided, the function will attempt to retrieve data from the FDEP WIN database using `read_importwqwin` from tbeptools. The EPC data are retrieved using `read_importepc` from tbeptools. All inputs for water quality can be superseded by providing a complete water quality data frame using the `allwq` argument.

The function assumes that the water quality data are in mg/L and flow data are in cfs. Missing water quality data are filled with previous five year averages for the end months, then linearly interpolated using `util_nps_fillmiswq`.

Value

A data frame with columns for basin, year, month, TN in mg/L, TP in mg/L, TSS in mg/L, BOD in mg/L, flow in liters/month, hydrologic load in m3/month, TN load in kg/month, TP load in kg/month, TSS load in kg/month, and BOD load in kg/month.

Examples

```
data(allwq)
data(allflo)

nps_gaged <- anlz_nps_gaged(
  yrrng = c('2021-01-01', '2023-12-31'),
  allflo = allflo,
  allwq = allwq
)

head(nps_gaged)
```

---

anlz_nps_psremove *Remove point source loads from non-point source load estimates*

Description

Subtract gaged industrial and domestic point source loads from NPS model output to isolate true non-point source loads.

Usage

```
anlz_nps_psremove(
  nps,
  ips,
  dps,
  ad_ap = TRUE,
  summ = c("segment", "basin"),
  summtime = c("month", "year")
)
```

Arguments

nps	data frame of NPS loads from anlz_nps with summ = 'basin' and summtime = 'month'
ips	data frame of IPS loads from anlz_ips with summ = 'basin' and summtime = 'month'
dps	data frame of DPS loads from anlz_dps with summ = 'basin' and summtime = 'month'
ad_ap	logical, whether to apply fixed monthly AD/AP TN reductions from the 2007 RA allocation analysis. Default TRUE.

`summ` character, one of `'segment'` or `'basin'`. Controls whether output is aggregated to the bay segment level (`'segment'`, default) or retained at the basin level (`'basin'`).

`summtime` character, one of `'month'` or `'year'`. Controls whether the output is monthly or annual. Default is `'month'`.

Details

Gaged NPS loads (estimated from stream gauges) include point source loads discharged upstream of the gauge. This function subtracts IPS and DPS loads in gaged basins from the combined NPS model output so that point source contributions are not double-counted.

Only IPS and DPS records in gaged basins (identified via `dbasing`) are subtracted. Nested basin identifiers (02301000, 02301300 → 02301500; 02303000, 02303330 → 02304500; 02299950 → LMANATEE) are reassigned to their parent basins before summing, consistent with the handling in `anlz_nps`.

When `ad_ap = TRUE`, fixed monthly TN reductions from the 2007 RA allocation analysis (AD/AP) are subtracted from the segment-level NPS totals. These values represent the annual reduction divided into monthly increments:

Old Tampa Bay -2.41 short tons/month

Hillsborough Bay -4.31 short tons/month

Middle Tampa Bay -2.29 short tons/month

Lower Tampa Bay -0.36 short tons/month

Manatee River -2.74 short tons/month (representing the combined reduction for segments 55, 6, and 7 as applied in the 2022-2024 RA)

Other segments (Boca Ciega Bay, Boca Ciega Bay South, Terra Ceia Bay) receive no AD/AP adjustment.

Value

data frame with columns for `Year`, `Month` (if `summtime = 'month'`), `source` (always `"NPS"`), `segment`, `basin` (if `summ = 'basin'`), `tn_load`, `tp_load`, `tss_load`, `bod_load`, and `hy_load`. Loads are in short tons per month or year; hydrologic load is in cubic meters per month or year.

See Also

`anlz_nps`, `anlz_ips`, `anlz_dps`

Examples

```
## Not run:
nps <- anlz_nps(yrrng = c('2021-01-01', '2023-12-31'), tbbase = tbbase,
  rain = rain, allwq = allwq, allflo = allflo, vernafl = vernafl,
  summ = 'basin', summtime = 'month')

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

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

ips <- anlz_ips(ipsfls, summ = 'basin', summtime = 'month')
dps <- anlz_dps(dpsfls, summ = 'basin', summtime = 'month')

anlz_nps_psremove(nps, ips, dps)

## End(Not run)
```

`anlz_nps_ungaged` *Estimated non-point source (NPS) ungaged loads*

Description

Estimated non-point source (NPS) ungaged loads

Usage

```
anlz_nps_ungaged(
  yrrng = c("2021-01-01", "2023-12-31"),
  tbbase,
  rain,
  lakemanpth = NULL,
  tampabypth = NULL,
  bellshlpth = NULL,
  allflo = NULL,
  usgsflow = NULL,
  verbose = TRUE
)
```

Arguments

`yrrng`	A vector of two dates in 'YYYY-MM-DD' format, specifying the date range to retrieve flow data. Default is from '2021-01-01' to '2023-12-31'.
`tbbase`	data frame containing polygon areas for the combined data layer of bay segment, basin, jurisdiction, land use data, and soils, see details
`rain`	data frame of rainfall data, see details
`lakemanpth`	character, path to the file containing the Lake Manatee flow data, see details
`tampabypth`	character, path to the file containing the Tampa Bypass flow data, see details
`bellshlpth`	character, path to the file containing the Bell shoals data, see details
`allflo`	data frame of flow data, if already available from `util_nps_getflow`, otherwise NULL and the function will retrieve the data

`usgsflow`	data frame of USGS flow data, if already available from `util_nps_getusgsflow`, otherwise NULL and the function will retrieve the data. Default is NULL. Does not apply if `allflo` is provided.
`verbose`	logical indicating whether to print verbose output

Details

This function estimates pollutant loads from non-point sources in ungaged (unmonitored) basins within the Tampa Bay watershed. The approach combines spatial land use data, rainfall patterns, hydrologic modeling, and empirical relationships to estimate monthly nutrient and sediment loads. Several steps are followed:

1. **Data Preparation**: Processes land use data for logistic regression modeling and calculates inverse distance-weighted rainfall data for each sub-basin.
2. **Flow Estimation**: Uses a logistic regression model to predict monthly streamflow in ungaged basins based on:
   - Current month rainfall and 2-month lagged rainfall
   - Land use percentages (urban, agriculture, wetlands, forest)
   - Seasonal patterns (wet season: July-October, dry season: November-June)
   - Urban development intensity (Group A: <19% urban, Group B: 19% urban)
   - Hydrologic soil group characteristics
3. **Runoff Coefficient Application**: Applies land use and soil-specific runoff coefficients to distribute predicted flows across different landscape types within each basin.
4. **Load Calculation**: Estimates pollutant loads using Event Mean Concentrations (EMCs) for different land use categories (CLUCSID), calculating:
   - Total Nitrogen (TN) loads
   - Total Phosphorus (TP) loads
   - Total Suspended Solids (TSS) loads
   - Biochemical Oxygen Demand (BOD) loads
   - Stormwater-specific loads (with different EMCs for certain categories)

Spatial Framework:

The analysis uses a nested spatial hierarchy:

- **Bay Segments**: Major subdivisions of Tampa Bay (1-7, with 55 for Lower Boca Ciega Bay)
- **Basins**: Hydrologic sub-basins, including both USGS gaged and ungaged areas
- **Land Use Polygons**: Detailed spatial units combining jurisdiction, land use (FLUCCS codes), and soil characteristics

Flow Prediction Models:

The function uses season- and development-specific regression equations:

- Separate models for wet vs. dry seasons
- Separate models for low-development (Group A) vs. high-development (Group B) areas

- Models account for antecedent moisture conditions through lagged rainfall terms

Load Estimation:

Pollutant loads are calculated as: `Load = Flow × EMC × Unit Conversions`

Where EMCs vary by land use category (CLUCSID). Special handling is applied for water bodies and certain wetland types (CLUCSIDs 18, 20), which are assigned zero stormwater loads.

Requires the following inputs:

- `tbbase`: A data frame containing polygon areas for the combined data layer of bay segment, basin, jurisdiction, land use data, and soils, Stored as `tbbase` or created (takes an hour or so) with `util_nps_tbbase`.
- `rain`: A data frame of rainfall data. See `rain`.
- `lakemanpth`: character, path to the file containing the Lake Manatee flow data. See `util_nps_getextflow`. Only applies if `allflo` is not provided.
- `tampabypth`: character, path to the file containing the Tampa Bypass flow data. See `util_nps_getextflow`. Only applies if `allflo` is not provided.
- `bellshlpth`: character, path to the file containing the Bell shoals data. See `util_nps_getextflow`. Only applies if `allflo` is not provided.

USGS gaged flows are also used, as returned by `util_nps_getusgsflow` and combined with the external flow data using `util_nps_getextflow` and `util_nps_getflow`.

Value

A data frame with monthly pollutant load estimates containing the following columns:

- `bay_seg`: Bay segment identifier (1: Old Tampa Bay, 2: Hillsborough Bay, 3: Middle Tampa Bay, 4: Lower Tampa Bay, 5: Upper Boca Ciega Bay, 6: Terra Ceia Bay, 7: Manatee River, 55: Lower Boca Ciega Bay)
- `basin`: Basin identifier (USGS gage number or internal code)
- `yr`: Year
- `mo`: Month (1-12)
- `clucsid`: Consolidated Land Use Classification System ID
- `h2oload`: Water load (cubic meters)
- `tnload`: Total nitrogen load (kg)
- `tpload`: Total phosphorus load (kg)
- `tssload`: Total suspended solids load (kg)
- `bodload`: Biochemical oxygen demand load (kg)
- `area`: Land use area (hectares)
- `bas_area`: Total basin area (hectares)

Examples

```
# external flow sources
data(tbbase)
data(rain)
data(allflo)

nps_ungaged <- anlz_nps_ungaged(
  yrrng = c('2021-01-01', '2023-12-31'),
  tbbase = tbbase,
  rain = rain,
  allflo = allflo
)

head(nps_ungaged)
```

---

anlz_spr *Calculate spring loads to Hillsborough Bay*

Description

Calculate spring loads to Hillsborough Bay

Usage

```
anlz_spr(
  tbwxlpth,
  wqpth = NULL,
  yrrng = c(2022, 2024),
  summ = c("spring", "basin", "segment"),
  summtime = c("month", "year"),
  sulphurflow = NULL,
  verbose = TRUE
)
```

Arguments

`tbwxlpth` character string, file path to the Tampa Bay Water discharge Excel workbook (.xlsx) for Lithia and Buckhorn springs. The workbook must contain one sheet per device, named by device ID: 3381 (Lithia Minor), 4586 (Lithia Major), 3388 (Buckhorn Upper), and 3649 (Buckhorn Lower). Each sheet must contain columns `DeviceID`, `MeasureDateTime`, `Value`, `MeasureType`, and `Units`.

`wqpth` character string or `NULL` (default). File path to spring water quality data (.csv), in one of two formats (see Details): FDEP IWR sample data, or annual output from `util_spr_getwq`. Spring names must match `"Lithia"`, `"Buckhorn"`, and `"Sulphur"`. When `NULL`, water quality data are retrieved automatically from external APIs via `util_spr_getwq`. Lithia and Buckhorn concentrations are obtained from the Water Atlas API

(SWFWMD stations 17805 and 18276) and Sulphur Spring concentrations are obtained via `read_importepc` (EPC station 174).

`yrrng` integer vector of length 2, start and end year for the analysis, e.g. `c(2022, 2024)`.

`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

`sulphurflow` data frame of daily Sulphur Spring discharge or `NULL` (default) to obtain from the USGS API.

`verbose` logical, if `TRUE` (default) progress messages are printed when water quality data are retrieved via API (`wqpth = NULL`).

Details

Loads are calculated for Lithia, Buckhorn, and Sulphur springs, all of which discharge to Hillsborough Bay (bay segment 2).

Discharge data (Lithia and Buckhorn): The Excel workbook supplied in `tbwxlpth` contains one sheet per device. Device IDs map to sub-springs as follows: 3381 = Lithia Minor, 4586 = Lithia Major, 3388 = Buckhorn Upper, 3649 = Buckhorn Lower. Flow values in MGD are converted to CFS (1 MGD = 1.547 CFS); values already in CFS are used as-is. Lithia total flow is the sum of Minor and Major. Buckhorn total flow is Lower minus Upper, because the two gauges bracket the spring input on the same stream reach.

Contact for gage data is Cathleen Jonas, cjonas@tampabaywater.org. Device IDs 3381, 4586, 3388, and 3649 should be bundled with requests for Tampa Bypass Canal data (device ID 957) and Bell Shoals data (device ID 4626) used in the NPS workflow.

Discharge data (Sulphur Springs): Daily CFS values for station 02306000 are retrieved from the USGS NWIS API via `util_nps_getusgsflow`. A data frame can also be supplied via the `sulphurflow` argument.

Interpolation: Because springs are assumed never to have zero discharge, all gaps in the daily discharge record are filled by linear interpolation between observed values (`na.approx` with `rule = 2`). Leading or trailing gaps are filled with the nearest observed value.

Water quality data (file path, FDEP IWR format): When `wqpth` is supplied and the CSV contains a `sta` column, it is treated as raw FDEP Impaired Waters Rule (IWR) sample data (available at `https://publicfiles.dep.state.fl.us/dear/iwr/`), with sample concentrations (mg/L) for TN and TP read from the CSV. Annual mean concentrations are computed per spring and joined to monthly flow estimates. A spring-year is considered complete when its samples span all four calendar quarters (Jan-Mar, Apr-Jun, Jul-Sep, Oct-Dec). Spring-years that are entirely missing or whose samples do not cover all four quarters are filled by carrying forward the most recent complete year's mean. The file should therefore include data from years prior to the focal period so that every spring has at least one complete reference year available. If the earliest year in the file is already incomplete for a spring, there is no prior year to carry forward and an error is raised.

Water quality data (file path, Water Atlas format): When `wqpth` is supplied and the CSV does *not* contain a `sta` column, it is treated as already-summarized annual output from `util_spr_getwq` (`raw = FALSE`, the default), with columns `spring`, `yr`, `tn_mgl`, `tp_mgl`,

and `tss_mgl` used as-is (no additional summarization or quarter-completeness check is applied; see note below). Supplying the raw, per-observation output of `util_spr_getwq` (`raw = TRUE`, identified by a `date` column instead of `yr`) raises an error.

Water quality data (API, `wqpth = NULL`): When `wqpth` is `NULL`, water quality data are obtained using `util_spr_getwq` (`raw = FALSE`), which is preferred over the FDEP IWR file in most cases because the Water Atlas API data are usually more current. Lithia (SWFWMD station 17805, Lithia Main Spring) and Buckhorn (SWFWMD station 18276, Buckhorn Main Spring) concentrations are retrieved from the Water Atlas API (`WIN_21FLSWFD` data source). These are probably the same quarterly SWFWMD observations included in the FDEP IWR file. Sulphur Spring (EPC station 174) is retrieved via `read_importepc`, providing monthly observations from the Environmental Protection Commission of Hillsborough County.

Note on quarter-completeness: the four-quarter completeness check described above applies only to the FDEP IWR CSV format. Annual means from `util_spr_getwq` (whether obtained via `wqpth = NULL` or supplied as a Water Atlas-format CSV) are not checked for quarter coverage before averaging.

TSS concentrations: When `wqpth` is supplied, TSS concentrations are assigned from a fixed lookup table derived from the historical SAS-based loading model (SPRMOD2). When `wqpth = NULL`, TSS values from the API or EPC source are used where available. Any spring-year with no observed TSS falls back to the same fixed values. The fixed values are: Sulphur Springs (02306000) = 4.4 mg/L, Buckhorn Springs (02301695) = 4.0 mg/L, Lithia Springs (02301600) = 4.0 mg/L.

BOD concentrations: BOD loads are returned as zero because BOD is not measured at the springs.

Load calculation: Monthly mean flows (CFS) are computed from the complete daily discharge series. Loads are then:

$$hy_load\,(m^3/month) = \overline{Q}_{cfs} \times 86400 \times \frac{365}{12} \times 28.32 \times 10^{-3}$$

$$load\,(tons/month) = hy_load \times C_{mg/L} \times 10^{-3}/907.1847$$

`hy_load` is converted to million m3 in the final output.

Spatial summaries: 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'`). For springs, valid options for `summ` are `'spring'` (one row per spring per time period), `'basin'` (loads summed within drainage basins. Lithia and Buckhorn combined into `"Alafia River"`, Sulphur into `"Hillsborough River"`), and `'segment'` (all springs summed to bay segment Hillsborough Bay).

Value

A data frame whose structure depends on `summ`:

- `'spring'`: one row per spring per time period, with columns `Year`, `Month` (dropped for annual), `source`, `segment`, `spring`, `tn_load` (tons), `tp_load` (tons), `tss_load` (tons), `bod_load` (tons), and `hy_load` (1e6 m3).

- `'basin'`: one row per drainage basin per time period, with columns `Year`, `Month` (dropped for annual), `source`, `segment`, `basin`, `tn_load` (tons), `tp_load` (tons), `tss_load` (tons), `bod_load` (tons), and `hy_load` (1e6 m3).
- `'segment'`: one row per bay segment per time period, with columns `Year`, `Month` (dropped for annual), `source`, `segment`, `tn_load` (tons), `tp_load` (tons), `tss_load` (tons), `bod_load` (tons), and `hy_load` (1e6 m3).

For annual output (`summtime = 'year'`), load columns are summed over months.

Examples

```
tbwxlpth <- system.file('extdata/sprflow2224.xlsx', package = 'tbeploads')
wqpth    <- system.file('extdata/sprwq2224.csv',    package = 'tbeploads')

# monthly per-spring loads using a local water quality file
anlz_spr(tbwxlpth = tbwxlpth, wqpth = wqpth, yrrng = c(2022, 2024))

# annual basin-level totals
anlz_spr(tbwxlpth = tbwxlpth, wqpth = wqpth, yrrng = c(2022, 2024),
          summ = 'basin', summtime = 'year')

# monthly segment-level totals
anlz_spr(tbwxlpth = tbwxlpth, wqpth = wqpth, yrrng = c(2022, 2024),
          summ = 'segment')

## Not run:
# retrieve water quality from APIs automatically (no local file needed)
anlz_spr(tbwxlpth = tbwxlpth, yrrng = c(2022, 2024))

## End(Not run)
```

`clucsid` *Lookup table for FLUCCSCODE conversion to CLUCSID and IMPROVED*

Description

Lookup table for FLUCCSCODE conversion to CLUCSID and IMPROVED

Usage

```
clucsid
```

Format

A data frame

Details

Used to create the `tbbase` combined layer with jurisdictions, land use, soils, and sub-basins, used in `util_nps_tbbase`.

- `FLUCCSCODE`: Numeric value for the Florida Land Use, Cover and Forms Classification System (FLUCCS) code
- `CLUCSID`: Numeric value for the coastal land use code, from JEI
- `IMPROVED`: Numeric value for whether the code is improved (1) or not (0)
- `DESCRIPTION`: Character description of the CLUCSID code

Examples

```
## Not run:
clucsid <- read.csv('data-raw/clucsid.csv', stringsAsFactors = F, header = T)

save(clucsid, file = 'data/clucsid.RData', compress = 'xz')

## End(Not run)
clucsid
```

`conserv_correction` *Conservation land correction fractions for NPS/MS4 allocation assessment*

Description

Conservation land correction fractions for NPS/MS4 allocation assessment

Usage

```
conserv_correction
```

Format

A data frame with one row per unique bay segment, basin, entity, and CLUCSID combination where conservation land is present:

bay_seg Integer bay segment identifier (1 = OTB, 2 = HB, 3 = MTB, 4 = LTB, 55 = RALTB).

basin Character drainage basin identifier.

entity MS4 jurisdiction name.

clucsid Integer Coastal Land Use Classification System identifier.

conserv_frac Fraction of entity area x runoff-coefficient attributable to conservation land within that bay segment / basin / CLUCSID combination. Computed as conservation area x RC divided by total entity area x RC (conservation + non-conservation) for that group.

Details

The tbeploads-built `tbbase` is derived from routinely updated GIS sources (land use, soils, jurisdictions) and does not include a conservation land spatial overlay. The conservation layer was available only for prior SAS workflows and cannot reproduced.

`conserv_correction` provides entity-specific fractions derived from the SAS land cover file (`npsag_3_2224_25Sep25.sas7bdat`), which includes a binary `conservation` column (0/1) indicating conservation land while retaining the original MS4 jurisdiction in `entity`. Within `anlz_aa`, each MS4 entity's disaggregated TN load for a given basin and CLUCSID is scaled by `(1 - conserv_frac)` to remove the conservation land contribution before hydrologic normalization.

Preprocessing matches `util_aa_npsfactors`: non-contributing drainage (`drnfeat = "NONCON"`) and water / tidal CLUCSIDs (17, 21, 22) are excluded, compound hydrologic soil groups are simplified, and nested basins are remapped. Only entity, basin, CLUCSID combinations with `conserv_frac > 0` are retained.

Source

Derived from `data-raw/npsag_3_2224_25Sep25.sas7bdat`, the SAS NPS land cover file. Built by `data-raw/conserv_correction.R`.

See Also

`anlz_aa`, `tbbase`, `util_aa_npsfactors`

`contdry` *Upper Floridan Aquifer potentiometric surface raster, dry season 2022*

Description

Upper Floridan Aquifer potentiometric surface raster, dry season 2022

Usage

```
contdry
```

Format

A `PackedSpatRaster` (see `wrap`) representing a 1-mile resolution grid of potentiometric head (ft above MSL) for the dry season. Unwrap with `terra::unwrap(contdry)` to obtain a `SpatRaster`.

Details

Interpolated from Upper Floridan Aquifer potentiometric surface contour lines (May 2022) downloaded from the FDEP / Florida Geological Survey ArcGIS REST service using util_gw_getcontour. The spatial extent covers the Tampa Bay watershed (tbfullshed) buffered outward by 40 miles. Interpolation used inverse distance weighting (IDW, 5-mile radius, power = 2) followed by five passes of a 3x3 focal mean gap fill. Projection is NAD83(2011) / Florida GDL Albers (ftUS), CRS 6443.

Wet season equivalent is contwet.

Examples

```
## Not run:
pot_dry <- util_gw_getcontour("dry", 2022)
contdry <- terra::wrap(pot_dry)
save(contdry, file = "data/contdry.RData", compress = "xz")

## End(Not run)
terra::unwrap(contdry)
```

---

contwet — *Upper Floridan Aquifer potentiometric surface raster, wet season 2022*

Description

Upper Floridan Aquifer potentiometric surface raster, wet season 2022

Usage

```
contwet
```

Format

A PackedSpatRaster (see wrap) representing a 1-mile resolution grid of potentiometric head (ft above MSL) for the wet season. Unwrap with terra::unwrap(contwet) to obtain a SpatRaster.

Details

Interpolated from Upper Floridan Aquifer potentiometric surface contour lines (September 2022) downloaded from the FDEP / Florida Geological Survey ArcGIS REST service using util_gw_getcontour. The spatial extent covers the Tampa Bay watershed (tbfullshed) buffered outward by 40 miles. Interpolation used inverse distance weighting (IDW, 5-mile radius, power = 2) followed by five passes of a 3x3 focal mean gap fill. Projection is NAD83(2011) / Florida GDL Albers (ftUS), CRS 6443.

Dry season equivalent is contdry.

Examples

```
## Not run:
pot_wet <- util_gw_getcontour("wet", 2022)
contwet <- terra::wrap(pot_wet)
save(contwet, file = "data/contwet.RData", compress = "xz")

## End(Not run)
terra::unwrap(contwet)
```

---

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

See "data-raw/dbasing.R" for creation.

Examples

```
dbasing
```

`dps_allocations` *TBNMC TN load allocations for DPS domestic wastewater facilities*

Description

TBNMC TN load allocations for DPS domestic wastewater facilities

Usage

```
dps_allocations
```

Format

A `data.frame`

Details

TN load allocations assigned to individual domestic point source (DPS) facilities under the Tampa Bay Nitrogen Management Consortium (TBNMC) framework.

- `entity`: Short entity name matching the `facilities` table convention (e.g., `"Clearwater"`, `"Hillsborough Co."`)
- `entity_full`: Full entity name as listed in the source allocation file (e.g., `"City of Clearwater"`, `"Hillsborough County"`)
- `facname`: Facility name matching the `facilities` table convention
- `bay_seg`: Integer bay segment identifier (1 = Old Tampa Bay, 2 = Hillsborough Bay, 3 = Middle Tampa Bay, 4 = Lower Tampa Bay, 55 = Remaining Lower Tampa Bay)
- `source`: DPS discharge type; one of `"DPS - end of pipe"` (direct surface water discharge) or `"DPS - reuse"` (reclaimed water reuse)
- `alloc_tons`: Allocation in tons TN per year

TECO Big Bend and Tropicana are not included: TECO is an industrial reuse customer rather than a direct discharger, and Tropicana is classified as an industrial point source in the `facilities` table. Neither can be matched to DPS load data from `anlz_dps_facility`.

Examples

```
dps_allocations
```

emc — *Event Mean Concentration (EMC) data for CLUCSID in Tampa Bay*

Description

Event Mean Concentration (EMC) data for CLUCSID in Tampa Bay

Usage

```
emc
```

Format

A `data.frame`

Details

Used for non-point source (NPS) ungaged estimates summaries.

Values are grouped by CLUCSID and include mean TN, TP, TSS, and BOD.

See "data-raw/emc.R" for creation.

Examples

```
emc
```

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
```

hydro_baseline — *Historic 1992-1994 mean total water load baseline by bay segment and basin*

Description

Historic 1992-1994 mean total water load baseline by bay segment and basin

Usage

```
hydro_baseline
```

Format

A `data.frame`

Details

Mean total water load for the 1992-1994 baseline period, used for hydrologic normalization in the allocation assessment. Values are in million cubic meters per year.

- `bay_seg`: Integer bay segment identifier
- `basin`: Drainage basin identifier
- `mean_h2o_9294`: Mean 1992-1994 total water load (million m3/yr)

See "data-raw/hydro_baseline.R" for creation.

Examples

```
hydro_baseline
```

ml_allocations — *TBNMC TN load allocations for industrial material loss (ML) facilities*

Description

TBNMC TN load allocations for industrial material loss (ML) facilities

Usage

```
ml_allocations
```

Format

A `data.frame`

Details

TN load allocations assigned to industrial material loss facilities under the Tampa Bay Nitrogen Management Consortium (TBNMC) framework. One row per facility, always.

- `entity`: Entity name matching the `facilities` table convention
- `facname`: Facility name matching the `facilities` table convention
- `bay_seg`: Integer bay segment identifier (2 = Hillsborough Bay, 4 = Lower Tampa Bay)
- `alloc_tons`: Allocation in tons TN per year. For `ishared` facilities, this is the group's collective allocation (the same value repeated on every member row), not an individual facility allocation
- `ishared`: Logical; `TRUE` when the facility is jointly assessed against a collective allocation shared with other facilities (see `alloc_tons`)
- `group_id`: Character identifier for the shared group a facility belongs to (`NA` when `ishared` is `FALSE`). Provided so shared-group membership can be recovered directly rather than inferred from matching `entity` + `alloc_tons`

The three Mosaic material loss facilities (Big Bend, Riverview, Tampa Marine) share a single 9.9 ton/year allocation in Hillsborough Bay (`ishared = TRUE` on all three rows, `group_id = "ml_mosaic_hb"`). Kinder Morgan Port Sutton and Tampaplex Material Losses are each assessed individually against their own distinct allocation (`ishared = FALSE`), despite the misleadingly similar names to the shared IPS Kinder Morgan group; Kinder Morgan Port Manatee is likewise a separate, non-shared facility.

Examples

```
ml_allocations
```

`nps_allocations` *TBNMC TN load allocations for NPS/MS4 entities*

Description

TBNMC TN load allocations for NPS/MS4 entities

Usage

```
nps_allocations
```

Format

A `data.frame`

Details

TN load allocations assigned to non-point source and MS4 jurisdictions under the Tampa Bay Nitrogen Management Consortium (TBNMC) framework.

- `bay_seg`: Integer bay segment identifier (1, 2, 3, 4, 55)
- `entity`: Short entity name used for joining
- `entity_full`: Full entity name
- `type`: Allocation type (e.g., MS4, Agriculture, Other)
- `alloc_pct`: Fractional allocation share (0-1)
- `alloc_tons`: Allocation in tons TN per year

Examples

```
nps_allocations
```

`nps_distance` *Data frame of distances of drainage basin locations to National Weather Service (NWS) sites*

Description

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

Usage

```
nps_distance
```

Format

A `data.frame`

Details

Used for estimating non-point source (NPS) ungaged loads. The data frame contains the following columns:

- `target`: Numeric identifier for the drainage basin
- `targ_x`: Numeric value for the x-coordinate of the drainage basin location (WGS 84, UTM Zone 17N, CRS 32617)
- `targ_y`: Numeric value for the y-coordinate of the drainage basin location (WGS 84, UTM Zone 17N, CRS 32617)
- `matchsit`: Numeric for the NWS site that matches the drainage basin location
- `distance`: Numeric value for the distance (m) between the drainage basin coordinate and NWS site
- `invdist2`: Numeric value for the inverse distance squared (1/m^2) between the drainage basin coordinate and NWS site

Examples

```
nps_distance
```

---

ps_allocations — *TBNMC TN load allocations for IPS point source facilities*

Description

TBNMC TN load allocations for IPS point source facilities

Usage

```
ps_allocations
```

Format

A `data.frame`

Details

TN load allocations assigned to individual industrial point source facilities under the Tampa Bay Nitrogen Management Consortium (TBNMC) framework.

- `entity`: Entity name (owner/operator)
- `facname`: Facility name as used in `facilities`
- `permit`: NPDES permit number
- `alloc_pct`: Fractional allocation share (0-1)
- `alloc_tons`: Allocation in tons TN per year. For `ishared` permits, this is the group's collective allocation (the same value repeated on every member row), not an individual permit allocation
- `hydro_affected`: Logical; `TRUE` for permits whose IPS load `anlz_aa` hydrologically normalizes
- `ishared`: Logical; `TRUE` when the permit is jointly assessed against a collective allocation shared with other permits (see `alloc_tons`)
- `group_id`: Character identifier for the shared group a permit belongs to (`NA` when `ishared` is `FALSE`). Provided so shared-group membership can be recovered directly rather than inferred from matching `entity` + `alloc_tons`

The 19 Mosaic facilities in Hillsborough Bay (Bartow, Bonnie, Ft. Lonesome, Green Bay, Hookers Prairie, Mulberry Phosphogypsum Stack, Mulberry Plant, New Wales Chemical Plant, Nichols Mine, Plant City, Riverview, Riverview Stack Closure, South Pierce, Tampa Ammonia Terminal, Tampa Marine Terminal, Hopewell, Kingsford, Port Sutton, Black Point) share a single 124.1 ton/year allocation (`ishared = TRUE`, `group_id = "ips_mosaic_hb"`). Kinder Morgan Tampaplex, Port Sutton, and Hartford Terminal likewise share a single 25.0 ton/year allocation (`group_id = "ips_kinder_morgan"`). All other permits are non-shared (`ishared = FALSE`, `group_id = NA`).

Examples

```
ps_allocations
```

`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

```
rain
```

Format

A `data.frame`

Details

Used for estimating atmospheric deposition and non-point source ungaged loads. Created using the `util_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_getrain`

Examples

```
rain
```

rain_historic — *Data frame of historical daily rainfall datas*

Description

Data frame of historical daily rainfall datas

Usage

```
rain_historic
```

Format

A `data.frame`

Details

Historical daily rain fall data for 388 stations through 2021. Columns are:

- `COOPID`: Character string for the station id
- `date`: Date for the observation
- `Prcp`: Numeric value for the amount of rainfall in inches

Examples

```
## Not run:
pth1 <- 'T:/03_BOARDS_COMMITTEES/05_TBNMC/2022_RA_Update/01_FUNDING_OUT/DELIVERABLES/TO-9/'
pth2 <- 'datastick_deliverables/LoadingCodes&Datasets/2021/AtmosphericDeposition2021/'
fl <- 'fl_rain_por_220223v93.sas7bdat'
pth <- file.path(pth1, pth2, fl)
rain_historic <- haven::read_sas(pth)

save(rain_historic, file = 'data/rain_historic.RData', compress = 'xz')

## End(Not run)
```

rcclucsid — *Lookup table for CLUCSID runoff coefficients*

Description

Lookup table for CLUCSID runoff coefficients

Usage

```
rcclucsid
```

Format

A data frame

Details

Used to create the land use runoff coefficient data used in util_nps_landsoilrc.

- clucsid: Numeric value for CLUCSID
- hsg: Numeric value for the hydrologic soil group
- dry_rc: Numeric value for dry weather runoff coefficient
- wet_rc: Numeric value for wet weather runoff coefficient

Examples

```
## Not run:
rcclucsid <- read.csv('data-raw/rc_clucsid.csv', stringsAsFactors = F, header = T)

save(rcclucsid, file = 'data/rcclucsid.RData', compress = 'xz')

## End(Not run)
rcclucsid
```

---

show_aaassess — *Bay segment table of allocation assessment by entity and facility*

Description

Bay segment table of allocation assessment by entity and facility

Usage

```
show_aaassess(aa_data, bay_seg, digits = 1, family = "Arial", txtsz = 11)
```

Arguments

aa_data	data frame returned by anlz_aa called with annavg = TRUE
bay_seg	integer bay segment identifier, one of 1 (Old Tampa Bay), 2 (Hillsborough Bay), 3 (Middle Tampa Bay), 4 (Lower Tampa Bay), or 55 (Remaining Lower Tampa Bay).
digits	numeric indicating decimal precision for the Allocated Tons and Effective Load columns. Default 1. Allocation % is always shown to two decimal places with a percent sign, independent of digits.
family	chr string indicating font family for text labels
txtsz	numeric indicating font size

Details

Rows are organized by **Entity**. An entity's block gathers every row that applies to it in `bay_seg` regardless of `source` (e.g. an entity with both IPS and Material Losses facilities gets one combined block and one combined Total).

Facility column: The displayed `Facility` text combines `facname` and `source`: IPS rows show `facname` as-is; DPS rows append `" (end of pipe)"` or `" (reuse)"`; Material Losses rows append `" (Material Losses)"`; MS4 rows (no `facname`) show `"MS4"`.

Entity Total row: a "unit" is either one shared group or one standalone facility/entity row. An entity gets a bolded `"Total"` row when it has more than one unit (regardless of how many display rows those units span), summing `alloc_tons/eff_load_tons` once per unit. Entities with exactly one unit (every MS4 row, and any entity with only one facility) get no Total row. Allocation % is left blank on the Total row.

Value

A `flextable` object with one row per facility/source nested under its owning entity, and a bolded "Total" row for any entity with more than one facility/source in `bay_seg`.

Examples

```
## Not run:
aa_data <- anlz_aa(c(2022, 2024), dps, ips, ml, nps, tbbase)
show_aaassess(aa_data, bay_seg = 2L)

## End(Not run)
```

---

show_aaloads *Bay segment table of TN loads by year*

Description

Bay segment table of TN loads by year

Usage

```
show_aaloads(
  aa_data,
  bay_seg,
  gw_data,
  spr_data,
  ad_data,
  yrrng = NULL,
  digits = 1,
  family = "Arial",
  txtsz = 11
)
```

Arguments

aa_data	data frame returned by anlz_aa called with annavg = FALSE
bay_seg	integer bay segment identifier, one of 1L (Old Tampa Bay), 2L (Hillsborough Bay), 3L (Middle Tampa Bay), 4L (Lower Tampa Bay), or 55L (Remaining Lower Tampa Bay).
gw_data	data frame returned by anlz_gw called with summtime = 'year'.
spr_data	data frame returned by anlz_spr called with summ = 'segment' and summtime = 'year'.
ad_data	data frame returned by anlz_ad called with summ = 'segment' and summtime = 'year'.
yrrng	optional integer vector of length 2 restricting the displayed years to a subset of those already present in aa_data. Default NULL shows all years present.
digits	numeric indicating decimal precision for the year columns. Default 1.
family	chr string indicating font family for text labels
txtsz	numeric indicating font size

Details

Rows are grouped into sections using aa_data's source column: "MS4" (source of "MS4" or "Nonpoint Source/MS4"), "Industrial Point Source" (source == "IPS"), "Domestic Point Source - end of pipe" and "Domestic Point Source - reuse" (source == "DPS - end of pipe"/"DPS - reuse"), "Material Losses" (source == "ML"), and "Nonpoint Source" (the "All" (FDACS) and "Non-MS4/Ag NPS" aggregate rows from aa_data, plus two rows built from gw_data/spr_data/ ad_data - see below; other unmatched source = NA rows in aa_data are negligible land-use slivers dropped by anlz_aa at its 0.01 tons/yr threshold in most years but not all, and are excluded here rather than shown as a spurious partial row). Each facility/entity keeps its own row, even for ishared shared-allocation groups (see anlz_aa), except Material Losses shared groups. Their members' loads are an even split of one combined load, so they are summed into a single row labeled with every member (e.g., "Mosaic - Big Bend, Riverview, Tampa Marine"), consistent with show_aaassess. Other row labels combine the owning entity with the facility name (e.g., "Mosaic - Bartow").

Atmospheric Deposition and Other (Groundwater, Springs, Conservation): gw_data, spr_data, and ad_data are mapped to bay_seg the same way anlz_aa does internally (Terra Ceia Bay and Manatee River summed into segment 55; Boca Ciega Bay variants dropped, consistent with the allocation framework's existing exclusion) and filtered to the requested bay_seg. ad_data's tn_load becomes the "Atmospheric Deposition" row. gw_data's and spr_data's tn_load (spr_data only has rows for Hillsborough Bay - other segments that have no contribution receive 0) plus aa_data's seg_conserv_tn (TN removed by the conservation-land correction already computed inside anlz_aa(), exposed as a segment total rather than sourced separately) become the "Other (Groundwater, Springs, Conservation)" row. Both are zero-filled for years with no matching input, same as facility/entity rows.

A facility/entity with a real allocation but no load data for a given year (or any year at all) is NA in aa_data by design (see anlz_aa); here it displays as 0 rather than blank.

Values shown are always `load_tons` (raw, unnormalized loads), never `eff_load_tons`.

A bolded `"Total Load"` row sums every displayed row for each year (including the AD and Other rows). A bolded `"Normalized Load"` row below it applies one hydrologic-normalization ratio to the whole segment's Total Load:

$$\text{Normalized Load} = \text{Total Load} \times \frac{\text{baseline_h2o}}{\text{seg_h2o_total} + \text{gw_hy_load} + \text{spr_hy_load} + \text{ad_hy_load}}$$

where the denominator is `aa_data`'s `seg_h2o_total` (NPS+IPS+DPS combined) plus `gw_data`/`spr_data`/`ad_data` own `hy_load`, and `baseline_h2o` is a hardcoded 1992-1994 baseline hydrologic load (million m3/yr) per bay segment:

bay_seg	baseline_h2o
1 Old Tampa Bay	449.44
2 Hillsborough Bay	895.62
3 Middle Tampa Bay	645.25
4 Lower Tampa Bay	361.19
55 Remaining Lower Tampa Bay	422.709

An additional `"Average"` column, appended after the year columns, gives each row's own mean across the displayed years.

Value

A `flextable` object with one row per entity/facility in `bay_seg`, grouped into sections by source, and one column per year.

Examples

```
## Not run:
aa_data <- anlz_aa(c(2022, 2024), dps, ips, ml, nps, tbbase, annavg = FALSE)
gw_data <- anlz_gw(contdry, contwet, yrrng = c(2022, 2024), summtime = 'year')
spr_data <- anlz_spr(tbwxlpth, wqpth, yrrng = c(2022, 2024),
  summ = 'segment', summtime = 'year')
ad_data <- anlz_ad(rain, vernafl, summ = 'segment', summtime = 'year')
show_aaloads(aa_data, bay_seg = 2L, gw_data, spr_data, ad_data)

## End(Not run)
```

`tbbase` *Combined spatial data required for non-point source (NPS) ungaged estimate*

Description

Combined spatial data required for non-point source (NPS) ungaged estimate

Usage

```
tbbase
```

Format

A summarized data frame containing the union of all inputs showing major bay segment, sub-basin (basin), drainage feature (drnfeat), jurisdiction (entity), land use/land cover (FLUCCSCODE), CLUCSID, IMPROVED, hydrologic group (hydgrp), and area in hectures. These represent all relevant spatial combinations in the Tampa Bay watershed.

See "data-raw/tbbase.R" for creation.

Examples

```
tbbase
```

---

tbdbasin — *Simple feature polygons major drainage basins in the Tampa Bay Estuary Program boundary*

Description

Simple feature polygons major drainage basins in the Tampa Bay Estuary Program boundary

Usage

```
tbdbasin
```

Format

A `sf` object

Details

Used for estimating ungaged non-point source (NPS) loads. The data includes the following columns.

- `basin`: Numeric value for the basin
- `drnfeat`: Numeric for the drainage feature
- `geometry`: The geometry column

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

Examples

```
## Not run:
prj <- 6443

tbdbasin <- sf::st_read("./data-raw/gis/TBEP_dBasins_FIPS0902_Projection.shp") |>
  sf::st_transform(prj) |>
  sf::st_buffer(dist = 0) |>
  dplyr::group_by(NEWGAGE, DRNFEATURE) |>
  dplyr::summarise(geometry = sf::st_union(geometry), .groups = "drop") |>
  dplyr::select(
    basin = NEWGAGE,
    drnfeat = DRNFEATURE
  ) |>
  dplyr::arrange(basin, drnfeat)

save(tbdbasin, file = 'data/tbdbasin.RData', compress = 'xz')

## End(Not run)
tbdbasin
```

---

tbfullshed — *Simple features polygon for the Tampa Bay Estuary Program boundary*

Description

Simple features polygon for the Tampa Bay Estuary Program boundary

Usage

```
tbfullshed
```

Format

A `sf` object

Details

Used for estimating ungaged non-point source (NPS) loads. The data includes the following columns.

- `Name`: Character for the layer name
- `Hectares`: Numeric value for area of the polygon
- `geometry`: The geometry column

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

Examples

```
## Not run:
prj <- 6443

tbfullshed <- sf::st_read("./data-raw/gis/TBEP_Watershed_Correct_Projection.shp") |>
  st_transform(prj) |>
  st_union(by_feature = T) |>
  st_buffer(dist = 0) |>
  dplyr::select(Name, Hectares)

save(tbfullshed, file = 'data/tbfullshed.RData', compress = 'xz')

## End(Not run)
tbfullshed
```

---

tbjuris — *Simple feature polygons of jurisdictional boundaries in the Tampa Bay Estuary Program boundary*

Description

Simple feature polygons of jurisdictional boundaries in the Tampa Bay Estuary Program boundary

Usage

```
tbjuris
```

Format

A `sf` object

Details

Used for estimating ungaged non-point source (NPS) loads. The data includes the following columns.

- `entity`: Character for the entity name
- `geometry`: The geometry column

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

Examples

```
## Not run:
prj <- 6443

tbjuris <- sf::st_read("./data-raw/gis/TB_Juris.shp") |>
  sf::st_transform(prj) |>
  sf::st_buffer(dist = 0) |>
  dplyr::rename(entity = NAME_FINAL) |>
  dplyr::select(entity) |>
  dplyr::group_by(entity) |>
  dplyr::summarise()

save(tbjuris, file = 'data/tbjuris.RData', compress = 'xz')

## End(Not run)
tbjuris
```

---

tblu2023 — *Simple feature polygons of 2023 land use in the Tampa Bay Estuary Program boundary*

Description

Simple feature polygons of 2023 land use in the Tampa Bay Estuary Program boundary

Usage

```
tblu2023
```

Format

A `sf` object

Details

Used for estimating ungaged non-point source (NPS) loads. The data includes the following columns.

- `FLUCCSCODE`: Numeric value for the Florida Land Use, Cover and Forms Classification System (FLUCCS) code
- `FLUCCSDESC`: Character describing the FLUCCS description
- `geometry`: The geometry column

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

Examples

```
## Not run:
# import from local source
prj <- 6443

tblu2023 <- sf::st_read("T:/05_GIS/SWFWMD/LULC_2023/LANDUSELANDCOVER2023.shp") |>
  sf::st_transform(prj) |>
  sf::st_intersection(tbfullshed) |>
  sf::st_buffer(dist = 0) |>
  dplyr::group_by(FLUCCSCODE, FLUCSDESC) |>
  dplyr::summarise() |>
  dplyr::ungroup()

# or use SWFWMD API
tblu2023 <- util_nps_getswfwmd('lulc2023')

save(tblu2023, file = 'data/tblu2023.RData', compress = 'xz')

## End(Not run)
```

---

tbmines — *Simple feature polygons of NPDES-permitted phosphate mine boundaries in the Tampa Bay watershed*

Description

Simple feature polygons of NPDES-permitted phosphate mine boundaries in the Tampa Bay watershed

Usage

```
tbmines
```

Format

A `sf` object

Details

Used by `util_nps_tbbase` to flag land under an active NPDES discharge permit so it can be excluded from non-point source (NPS) load estimation (see `util_aa_npsfactors`, which excludes `CLUCSID = 22`). Without this exclusion, a permitted facility's land area would be counted twice: once through the land-use-based NPS model and again through its actual point-source (IPS/DPS) discharge data. The data include the following columns.

- `Location`: Character for the mine name
- `Owner`: Character for the operating company
- `geometry`: The geometry column

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

Covers 14 Central Florida phosphate mine sites (13 operated by Mosaic, 1 by CF Industries).

Examples

```
## Not run:
prj <- 6443

tbmines <- sf::st_read("./data-raw/gis/TB_mines_NPDES.shp") |>
  sf::st_make_valid() |>
  sf::st_transform(prj)

save(tbmines, file = 'data/tbmines.RData', compress = 'xz')

## End(Not run)
tbmines
```

---

tbsegdetail — *Simple feature polygons of Tampa Bay segments with shoreline detail*

Description

Simple feature polygons of Tampa Bay segments with shoreline detail

Usage

```
tbsegdetail
```

Format

A `sf` object

Details

Detailed shoreline polygons for the major bay segments, including a North/South split for Boca Ciega Bay. Note that the Boca Ciega Bay segment is only the northern portion. The data includes the following columns.

Layer was clipped to retain only the main open-water bay areas, excluding tidal rivers and creek arms. The clipping mask is available in `data-raw/tbsegdetail_clip_mask.RData` for reproducibility. See `data-raw/tbsegdetail_clip.R` for the clipping workflow.

- `bay_seg`: Integer, numeric segment identifier matching
- `geometry`: The geometry column

Bay segments included:

- 1: Old Tampa Bay

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

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

Examples

```
tbsegdetail
```

`tbsoil` *Simple feature polygons of soil data in the Tampa Bay Estuary Program boundary*

Description

Simple feature polygons of soil data in the Tampa Bay Estuary Program boundary

Usage

```
tbsoil
```

Format

A `sf` object

Details

Used for estimating ungaged non-point source (NPS) loads. The data includes the following columns.

- `hydgrp`: Character for the hydrologic group (A, B, C, D, etc.) of the soil, `NA` for areas with no SSURGO classification (e.g., open water)
- `geometry`: The geometry column

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

`NA` `hydgrp` values are treated as group D by `util_nps_tbbase` rather than excluded, matching RP's own convention (see the `"NOSOIL"` handling in their SAS workflow) of retaining such areas instead of dropping them from the combined base layer.

Examples

```
## Not run:
usdasoil <- st_read('T:/05_GIS/TBEP/TBLOADS/USDA_SSURGO_CLIP_FIPS0902.shp')
tbsoil <- usdasoil |>
  select(hydgrp = SDV_Hydr_1) |>
  group_by(hydgrp) |>
  summarise() |>
  st_transform(crs = 6443)
save(tbsoil, file = 'data/tbsoil.RData', compress = 'xz')

## End(Not run)
```

---

tbsubshed — *Simple feature polygons of sub-watersheds in the Tampa Bay Estuary Program boundary*

Description

Simple feature polygons of sub-watersheds in the Tampa Bay Estuary Program boundary

Usage

```
tbsubshed
```

Format

A `sf` object

Details

Used for estimating ungaged non-point source (NPS) loads. The data includes bay segment as follows:

- 1: Old Tampa Bay
- 2: Hillsborough 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

Projection is NAD83(2011) / Florida West (ftUS), CRS 6443.

Examples

```
## Not run:
prj <- 6443

tbsubshed <- sf::st_read("./data-raw/gis/TBEP_Major_Basins_NAD1983_SP_FIPS0902_FT.shp") |>
  sf::st_transform(prj) |>
  sf::st_buffer(dist = 0) |>
  dplyr::mutate(
    bay_seg = dplyr::case_when(
      BASINNAME %in% c('Coastal Old Tampa Bay') ~ 1,
      BASINNAME %in% c('Alafia River', 'Coastal Hillsborough Bay', 'Hillsborough River') ~ 2,
      BASINNAME %in% c('Coastal Middle Tampa Bay', 'Little Manatee River') ~ 3,
      BASINNAME %in% c('Coastal Lower Tampa Bay') ~ 4,
      BASINNAME %in% c('Upper Boca Ciega Bay') ~ 5,
      BASINNAME %in% c('Coastal Terra Ceia Bay') ~ 6,
      BASINNAME %in% c('Manatee River') ~ 7,
      BASINNAME %in% c('Lower Boca Ciega Bay') ~ 55,
    )
  ) |>
  dplyr::group_by(bay_seg) |>
  dplyr::summarise(geometry = sf::st_union(geometry), .groups = "drop")

save(tbsubshed, file = 'data/tbsubshed.RData', compress = 'xz')

## End(Not run)
tbsubshed
```

---

usgsflow — *Data frame of USGS stream flow data from the USGS NWIS database for 2021 to 2023*

Description

Data frame of USGS stream flow data from the USGS NWIS database for 2021 to 2023

Usage

```
usgsflow
```

Format

A `data.frame`

Details

Daily flow data at select stations used for estimating non-point source gaged and ungaged loads. Created using the `util_nps_getusgsflow` function. The file is provided to reduce calls to the USGS API. The data frame contains the following columns:

- `site_no`: Character string for the site number

- `date`: Date for the observation
- `flow_cfs`: Numeric value for the daily flow in cubic feet per second (cfs)

See Also

util_nps_getusgsflow

Examples

```
usgsflow
```

---

util_aa_npsfactors *Create NPS disaggregation factors for allocation assessment*

Description

Create NPS disaggregation factors for allocation assessment

Usage

```
util_aa_npsfactors(tbbase, rcclucsid, emc)
```

Arguments

`tbbase`	Data frame returned from util_nps_tbbase containing land use, soils, and jurisdiction data. Must include columns `bay_seg`, `basin`, `drnfeat`, `entity`, `CLUCSID`, `hydgrp`, and `area_ha`.
`rcclucsid`	Data frame of runoff coefficients by land use class and hydrologic soil group. See rcclucsid.
`emc`	Data frame of event mean concentrations by land use class. Must include columns `clucsid` and `mean_tn`. See emc.

Details

These factors are used internally by anlz_aa to disaggregate basin-level NPS loads to individual MS4 jurisdictions (entities).

Two factors are required because the disaggregation is a two-step process matching the original SAS workflow:

TN factor (`factor_tn`) Distributes total basin TN load among land use classes. Based on area × event mean TN concentration per CLUCSID. Jurisdiction is not required, sums to basin total.

RC factor (`factor_rc`) Distributes each land use class's TN and water loads among MS4 entities based on each entity's share of that land use type's weighted runoff (area × runoff coefficient). Jurisdiction is required here, i.e., `tbbase` includes the entity overlay.

Annual runoff coefficient: `rc = (dry_rc * 8 + wet_rc * 4) / 12`.

Basin remapping matches the SAS preprocessing: nested basins 02303000 and 02303330 are assigned to 02304500; 02301000 and 02301300 to 02301500; 02299950 to LMANATEE; basin 206-5 is assigned to bay_seg 55. Basin 02307359 is excluded entirely.

Non-contributing drainage features (`drnfeat == "NONCON"`) and water/tidal CLUCSIDs (17, 21, 22) are excluded from both factors.

Compound hydrologic soil groups (e.g., `"A/D"`) are simplified to their primary group (`"A"`) before joining runoff coefficients.

These factors are specific to a land use layer and should be rebuilt whenever the underlying `tbbase` changes.

Value

A named list with two elements:

rc Data frame of RC factors: `bay_seg`, `basin`, `entity`, `category`, `clucsid`, `factor_rc`. `factor_rc` is each entity's fractional share of the weighted area × runoff coefficient within each basin × CLUCSID combination. Sums to 1 across all entities for each basin × CLUCSID.

tn Data frame of TN factors: `bay_seg`, `basin`, `clucsid`, `factor_tn`. `factor_tn` is each CLUCSID's fractional share of basin TN load, weighted by area × event mean TN concentration. Sums to 1 across all CLUCSIDs for each basin.

Examples

```
data(tbbase)
data(rcclucsid)
data(emc)
nps_factors <- util_aa_npsfactors(tbbase, rcclucsid, emc)
```

`util_getrain` *Get rainfall data at NOAA NCDC sites for atmospheric deposition and non-point source ungaged calculations*

Description

Get rainfall data at NOAA NCDC sites for atmospheric deposition and non-point source ungaged calculations

Usage

```
util_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 and NPS ungaged 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

`rain`

Examples

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

## End(Not run)
```

---

util_gw_getcontour — *Download and rasterize FDEP Upper Floridan Aquifer potentiometric surface*

Description

Download and rasterize FDEP Upper Floridan Aquifer potentiometric surface

Usage

```
util_gw_getcontour(
  season = c("dry", "wet"),
  yr,
  max_records = 1000,
  verbose = TRUE
)
```

Arguments

`season`	character, `"dry"` or `"wet"`.
`yr`	integer, year for which to retrieve data. Biannual (May/September) observations are available from approximately 2010 onward.
`max_records`	integer, maximum records per paginated API request. Default 1000.
`verbose`	logical, print download and interpolation progress. Default `TRUE`.

Details

Downloads Upper Floridan Aquifer potentiometric surface contour lines from the FDEP / Florida Geological Survey ArcGIS REST service (https://ca.dep.state.fl.us/arcgis/rest/services/OpenData/FGS_PUBLIC/MapServer/8) and interpolates them to a 1-mile `SpatRaster` using inverse distance weighting (IDW).

Spatial extent: The API query covers the Tampa Bay watershed (`tbfullshed`) buffered outward by 40 miles (211,200 US Survey Feet), converted to WGS84. This wider extent captures high potentiometric values outside of the surficial subwatersheds that drive groundwater flow.

Interpolation: Contour line vertices are used as elevation observations and interpolated to a 1-mile grid via IDW (5-mile radius, power = 2). Cells more than 5 miles from any contour vertex are left `NA` to avoid extrapolation into data-sparse regions. Five passes of a 3x3 focal mean then fill small gaps. The 5-mile radius was chosen to bridge typical contour spacing in the Tampa Bay region without extrapolating into the panhandle or coastal areas.

Season mapping:

- `"dry"` maps to May of `yr`
- `"wet"` maps to September of `yr`

Value

A `SpatRaster` of potentiometric head (ft above MSL) at 1-mile resolution in the CRS of `tbfullshed` (EPSG 6443). Returns `NULL` with a warning if no features are found for the requested season/year.

Examples

```
## Not run:
pot_dry <- util_gw_getcontour("dry", 2022)
pot_wet <- util_gw_getcontour("wet", 2022)

## End(Not run)
```

---

util_gw_getwq — *Get groundwater quality concentrations for Floridan aquifer segments*

Description

Get groundwater quality concentrations for Floridan aquifer segments

Usage

```
util_gw_getwq(sta_ids = NULL, yrrng = NULL, verbose = TRUE)
```

Arguments

`sta_ids`	character vector of SWFWMD station IDs to query. When `NULL` (default), uses stations 18340 (CR 581 North Fldn) and 18965 (SR 52 and CR 581 Deep), the two Pasco County Floridan aquifer monitoring wells used in the 2022-2024 Tampa Bay groundwater loading analysis.
`yrrng`	integer vector of length 2 specifying the start and end year for computing concentration means, e.g. `c(2020, 2024)`. When `NULL` (default), all available observations are used.
`verbose`	logical, if `TRUE` (default) a progress message is printed.

Details

Retrieves TN and TP concentrations (mg/L) from the Water Atlas API (`GET /api/samplingdata/stream`) for Upper Floridan Aquifer monitoring stations and computes grand-mean concentrations per station. Station means are then mapped to bay segments:

- **OTB (segment 1):** mean of `sta_ids[1]` only (default: CR 581 North Fldn, station 18340).
- **HB (segment 2):** arithmetic mean of the per-station means across all `sta_ids` (default: mean of stations 18340 and 18965, SR 52 and CR 581 Deep).
- **Segments 3-7:** fixed constants carried forward from `gwupdate95-98_final.xls` (the original 1995-1998 SWFWMD monitoring analysis). These values were used unchanged in every loading script from 2012 through 2021 and are not updated from the API.

History: Through the 2021 loading cycle, all seven segments used hardcoded Floridan concentrations sourced from the 1995-1998 spreadsheet (TN: 0.010-0.025 mg/L, TP: 0.097-0.137 mg/L). For the 2022-2024 update, new SWFWMD well data showed substantially higher TN in the Pasco County Floridan aquifer, so segments 1 and 2 were revised using stations 18340 and 18965. Segments 3-7 retained the original values.

TN is taken from the `TN_mgl` parameter and TP from `TP_mgl` in the Water Atlas API response.

Value

A data frame with one row per bay segment (1-7) and columns:

- `bay_seg`: integer, bay segment number
- `tn_mgl`: numeric, mean total nitrogen concentration (mg/L)
- `tp_mgl`: numeric, mean total phosphorus concentration (mg/L)

Examples

```
## Not run:
# default stations, all available data
conc <- util_gw_getwq()

# restrict to a specific period
```

```
conc <- util_gw_getwq(yrrng = c(2020, 2024))

## End(Not run)
```

`util_gw_grad` *Compute hydraulic gradient per bay segment from potentiometric surface raster*

Description

Compute hydraulic gradient per bay segment from potentiometric surface raster

Usage

```
util_gw_grad(
  pot_rast,
  season = c("dry", "wet"),
  segs = tbsubshed,
  shoreline = tbsegdetail,
  buf_segs = NULL
)
```

Arguments

`pot_rast`	`SpatRaster` (or `PackedSpatRaster`) of Upper Floridan Aquifer potentiometric head (ft above MSL) as returned by `util_gw_getcontour`, or the package datasets `contdry` / `contwet`.
`season`	character, `"dry"` or `"wet"`.
`segs`	`sf` object of sub-watershed polygons. Defaults to `tbsubshed`.
`shoreline`	`sf` object of bay segment polygons used to derive the bay water mask and measure distances. Defaults to `tbsegdetail`.
`buf_segs`	named numeric vector mapping bay segment IDs (as character strings) to omnidirectional buffer distances in US Survey Feet (CRS 6443). The subwatershed for each listed segment is buffered outward by the given distance and bay water is removed before the potentiometric high-point search. Listing a segment here also removes it from the default zero-gradient set so it is computed dynamically. When `NULL`, season-specific defaults are used (see Details).

Details

Computes the Floridan Aquifer hydraulic gradient I (ft/mile) per bay segment using the Darcy's Law framework of Zarbock et al. (1994):

$$I = \frac{\text{elevation (ft)}}{\text{distance to shoreline (miles)}}$$

The max potentiometric head within the search area is located in the interpolated raster and the distance is measured from the nearest shoreline crossing along the bay centroid-to-head transect (see below).

Zero-gradient segments (hardcoded): The following segments always receive a gradient of 0 based on the original SAS loading analysis (Zarbock et al., 1994; GWld2224_SASCode.txt):

- Boca Ciega Bay (segments 5 and 55), both seasons: the urbanized coastal watershed has no meaningful Floridan Aquifer recharge directed toward the bay.
- Lower Tampa Bay (4), Terra Ceia Bay (6), Manatee River (7), dry season only: the potentiometric gradient is negligible during the dry season. These segments are computed dynamically in the wet season via the default `buf_segs`.

Any segment listed in `buf_segs` is removed from the zero set and computed dynamically.

Default buf_segs (calibrated against 2021 SAS reference values):

- Dry season: `c("1" = 100000)` – Old Tampa Bay subwatershed buffered ~19 miles; captures the potentiometric high north/northeast of the standard watershed boundary.
- Wet season: `c("1" = 100000, "4" = 100000, "6" = 100000, "7" = 100000)` – adds LTB, TCB, and MR (each ~19 miles) to unlock wet-season computation for those segments.

Buffer distances were tuned to produce gradients within ~15\ FDEP potentiometric surface values used in the SAS analysis.

Distance calculation: Rather than measuring from the potentiometric high to the nearest shoreline point (which can hit an extreme geographic corner), the function draws a line from the bay segment centroid to the max-head cell. The portion of that line inside the bay polygon is subtracted from the total length, giving the distance from the shoreline crossing point to the high point along a representative transect.

Hillsborough Bay (segment 2): Uses a three-zone weighted gradient (Polk County 0.4, Pasco County 0.3, Alafia River 0.3) following the original flow net analysis. Sub-zones are constructed from `tbdbasin` drainage basins as in the original SAS code.

Benchmark warning: After computing gradients, each non-zero segment is compared against the 2021 SAS reference values (GWld2224_SASCode.txt). A warning is issued for any segment whose computed gradient deviates by more than 50\ reference, indicating a potentially anomalous potentiometric surface or a need to revisit the `buf_segs` configuration.

Value

A data frame with columns `bay_seg` (integer) and `grad` (numeric, ft/mile; 0 for zero-gradient segments).

References

Zarbock, H., A. Janicki, D. Wade, D. Heimbuch, and H. Wilson. 1994. Estimates of Total Nitrogen, Total Phosphorus, and Total Suspended Solids Loadings to Tampa Bay, Florida. Technical Publication #04-94. Prepared by Coastal Environmental, Inc. Prepared for Tampa Bay National Estuary Program. St. Petersburg, FL.

Examples

```
## Not run:
pot_dry <- util_gw_getcontour("dry", 2022)
util_gw_grad(pot_dry, season = "dry")

pot_wet <- util_gw_getcontour("wet", 2022)
util_gw_grad(pot_wet, season = "wet")

## End(Not run)
```

---

util_gw_showgrad *Visualise the hydraulic gradient for a bay segment*

Description

Visualise the hydraulic gradient for a bay segment

Usage

```
util_gw_showgrad(
  pot_rast,
  season = c("dry", "wet"),
  seg,
  segs = tbsubshed,
  shoreline = tbsegdetail,
  buf_segs = NULL
)
```

Arguments

pot_rast	SpatRaster (or PackedSpatRaster) of Upper Floridan Aquifer potentiometric head (ft above MSL) as returned by util_gw_getcontour, or the package datasets contdry / contwet.
season	character, "dry" or "wet".
seg	integer, bay segment number (1-7).
segs	sf object of sub-watershed polygons. Defaults to tbsubshed.
shoreline	sf object of bay segment polygons. Defaults to tbsegdetail.
buf_segs	named numeric vector of buffer distances (US Survey Feet) in the same format accepted by util_gw_grad. When NULL, season-specific defaults are used (see util_gw_grad for details).

Details

Returns a `ggplot2` map for the requested segment showing:

- The potentiometric surface (ft) within the search area, coloured by head value.
- The search area boundary (light yellow).
- All bay segments (grey background) and the target segment (blue).
- A dotted line from the bay centroid to the max-head land cell (showing the full transect used in the distance calculation).
- A dashed line for the land portion of that transect (the actual gradient distance).
- The max-head point (red dot).

The subtitle reports max head (ft), distance (miles), and gradient (ft/mi). See `util_gw_grad` for the distance calculation methodology.

Value

A `ggplot` object.

Examples

```
## Not run:
contdry <- util_gw_getcontour("dry", 2022)

## End(Not run)
util_gw_showgrad(contdry, season = "dry", seg = 1)
util_gw_showgrad(contdry, season = "dry", seg = 3)

## Not run:
contwet <- util_gw_getcontour("wet", 2022)

## End(Not run)
util_gw_showgrad(contwet, season = "wet", seg = 4)
util_gw_showgrad(contwet, season = "wet", seg = 7)
```

---

util_nps_fillmiswq — *Fill in missing water quality values for non-point source (NPS) data*

Description

Fill in missing water quality values for non-point source (NPS) data

Usage

```
util_nps_fillmiswq(wq, yrrng = c("2021-01-01", "2023-12-31"))
```

Arguments

`wq`	A data frame of water quality data returned by `util_nps_getwq`
`yrrng`	A vector of two dates in 'YYYY-MM-DD' format, specifying the date range to retrieve flow data. Default is from '2021-01-01' to '2023-12-31'.

Details

Missing end date monthly values are filled with prior 5 year averages. Then, missing monthly values are linearly interpolated using `na.approx`.

Value

Input data frame with missing data filled as described above.

Examples

```
## Not run:
data(allwq)
wq <- util_nps_fillmiswq(allwq)

## End(Not run)
```

`util_nps_getextflow` *Get external flow data not from USGS for NPS calculations*

Description

Get external flow data not from USGS for NPS calculations

Usage

```
util_nps_getextflow(pth, loc, yrrng = c(2021, 2023))
```

Arguments

`pth`	Path to the external Excel file
`loc`	Location of the external flow data. Options are 'LMANATEE', 'TBYPASS', or '02301500'.
`yrrng`	Numeric vector of length 2 indicating the year range to filter the data. Default is c(2021, 2023).

Details

This function retrieves and formats external flow data that cannot be obtained from the USGS API. The three required locations are Lake Manatee, Tampa Bypass Canal (s160), and the Alafia River Bell Shoals Bell Shoals.

External data can be obtained as follows:

- LMANATEE: Lake Manatee flow for the Manatee River dam, from Manatee County Utilities, input flow is cfs (Manatee County contact is Amanda ShawverKarnitz, amanda.shawverkarnitz@mym
- TBYPASS: Tampa Bypass Canal flow from Tampa Bay Water. Input flow is MGD and is converted to cfs (Tampa Bay Water contact is Cathleen Jonas, cjonas@tampabaywater.org, device ID 957). Requests for this data should be bundled with requests for the Lithia and Buckhorn spring discharge data (device IDs 3381, 4586, 3388, and 3649) used in the springs workflow.
- 02301500: Alafia River Bell Shoals flow data from SWFWMD WMIS Pumpage Reports for Permit 11794 (`https://www18.swfwmd.state.fl.us/search/search/searchwupsimple.aspx`) or optionally from Tampa Bay Water reported withdrawals for Site 4626 (Cathleen Jonas, cjonas@tampabaywater.org, device ID 4626, request should be bundled with device IDs 3381, 4586, 3388, and 3649 used in the springs workflow). Input flow from latter is daily average converted to cfs.

System files are included in the package which can be updated annually.

Value

A data frame of flow data for the location in `loc`

Examples

```
# lake manatee
pth <- system.file('extdata/nps_extflow_lakemanatee.xlsx', package = 'tbeploads')
extflo <- util_nps_getextflow(pth, loc = "LMANATEE")

# tampa bypass
pth <- system.file('extdata/nps_extflow_tampabypass.xlsx', package = 'tbeploads')
extflo <- util_nps_getextflow(pth, loc = "TBYPASS")

# bell shoals
pth <- system.file('extdata/nps_extflow_bellshoals.xls', package = 'tbeploads')
extflo <- util_nps_getextflow(pth, loc = "02301500")
```

---

util_nps_getflow *Get flow data from for NPS calculations at gaged sites*

Description

Get flow data from for NPS calculations at gaged sites

Usage

```
util_nps_getflow(
  lakemanpth = NULL,
  tampabypth = NULL,
  bellshlpth = NULL,
  yrrng = c(2021, 2023),
  usgsflow = NULL,
  verbose = TRUE
)
```

Arguments

`lakemanpth`	character, path to the file containing the Lake Manatee flow data
`tampabypth`	character, path to the file containing the Tampa Bypass flow data
`bellshlpth`	character, path to the file containing the Bell shoals data
`yrrng`	vector of two integers, the year range for which to retrieve flow data. Default is c(2021, 2023).
`usgsflow`	data frame of USGS flow data, if already available from `util_nps_getusgsflow`, otherwise NULL and the function will retrieve the data. Default is NULL.
`verbose`	logical indicating whether to print verbose output

Details

Missing flow values are linearly interpolated using `na.approx`. The function combines external and USGS API flow data using the `util_nps_getextflow` and `util_nps_getusgsflow` functions.

A preprocessed USGS flow data frame can be provided using the `usgsflow` argument to avoid re-downloading the data.

Value

A data frame of monthly mean flow for fifteen USGS stations and three external flow sites

See Also

`util_nps_getextflow`, `util_nps_getusgsflow`

Examples

```
## Not run:
usgsflow <- util_nps_getusgsflow(yrrng = as.Date(c('2021-01-01', '2023-12-31')))

## End(Not run)
lakemanpth <- system.file('extdata/nps_extflow_lakemanatee.xlsx', package = 'tbeploads')
tampabypth <- system.file('extdata/nps_extflow_tampabypass.xlsx', package = 'tbeploads')
bellshlpth <- system.file('extdata/nps_extflow_bellshoals.xls', package = 'tbeploads')
allflo <- util_nps_getflow(lakemanpth, tampabypth, bellshlpth, usgsflow = usgsflow)
```

`util_nps_getswfwmd` *Retrieve non-point source (NPS) supporting data from SWFWMD web services*

Description

Retrieve non-point source (NPS) supporting data from SWFWMD web services

Usage

```
util_nps_getswfwmd(dat, max_records = 1000, verbose = TRUE)
```

Arguments

`dat`	Character string indicating the type of data to retrieve. Options are 'soil', 'lulc2020', or 'lulc2023'.
`max_records`	Integer specifying the maximum number of records to retrieve in each request. Default is 1000.
`verbose`	Logical indicating whether to print verbose output. Default is TRUE.

Details

This function retrieves data from the SWFWMD web services for soils and land use/land cover (LULC) for the years 2020 and 2023. Soils data from https://www25.swfwmd.state.fl.us/arcgis12/rest/services/BaseVector and land use data from https://www25.swfwmd.state.fl.us/arcgis12/rest/services/OpenData.

Value

A simple features object for the relevant data, clipped by the Tampa Bay watershed boundary (`tbfullshed`).

Examples

```
## Not run:
# Retrieve soil data
soil_data <- util_nps_getswfwmd('soil')

# Retrieve LULC data for 2020
lulc2020_data <- util_nps_getswfwmd('lulc2020')

# Retrieve LULC data for 2023
lulc2023_data <- util_nps_getswfwmd('lulc2023')


## End(Not run)
```

util_nps_getusgsflow *Get flow data from USGS for NPS calculations*

Description

Get flow data from USGS for NPS calculations

Usage

```
util_nps_getusgsflow(
  site = NULL,
  yrrng = c("2021-01-01", "2023-12-31"),
  verbose = TRUE
)
```

Arguments

site	A character vector of USGS site numbers. If NULL, defaults to a predefined set of sites. Default is NULL, see details.
yrrng	A vector of two dates in 'YYYY-MM-DD' format, specifying the date range to retrieve flow data. Default is from '2021-01-01' to '2023-12-31'.
verbose	logical indicating whether to print verbose output

Details

Stations are from the USGS NWIS database and include 02299950, 02300042, 02300500, 02300700, 02301000, 02301300, 02301500, 02301750, 02303000, 02303330, 02304500, 02306647, 02307000, 02307359, and 02307498. Uses the `read_waterdata_daily` function from the `dataRetrieval` package.

Value

A data frame of daily flow values in cfs for fifteen stations

Examples

```
## Not run:
usgsflow <- util_nps_getusgsflow()

## End(Not run)
```

util_nps_getwq *Get water quality data for NPS gaged flows*

Description

Get water quality data for NPS gaged flows

Usage

```
util_nps_getwq(
  yrrng = c("2021-01-01", "2023-12-31"),
  mancopth = NULL,
  pincopth = NULL,
  verbose = TRUE
)
```

Arguments

yrrng	A vector of two dates in 'YYYY-MM-DD' format, specifying the date range to retrieve flow data. Default is from '2021-01-01' to '2023-12-31'.
mancopth	character, path to the Manatee County water quality data file, see details
pincopth	character, path to the Pinellas County water quality data file, see details
verbose	logical indicating whether to print verbose output

Details

If `mancopth` or `pincopth` are `NULL`, Manatee and Pinellas County data are retrieved from the FDEP WIN database using `read_importwqwin`. Hillsborough County data is retrieved using `read_importwq`. If `mancopth` or `pincopth` are not `NULL`, then data are imported from disk using the path specified. Data from the Environmental Protection Commission (EPC) of Hillsborough County are imported using `read_importepc` from the tbeptools R package.

Local data files can be downloaded from the FDEP WIN database at https://prodenv.dep.state.fl.us/DearWin/public/wavesSearchFilter?calledBy=menu, using filters for 21FLMANA and 21FLPDEM for Manatee and Pinellas County, respectively. Activity start and end dates are bounded by the values in `yrrng`. Chosen analytes are Nitrate-Nitrite (N), Nitrogen- Total Kjeldahl, Phosphorus- Total, and Residues- Nonfilterable (TSS). Chosen stations are ER2 and UM2 for Manatee County and station 06-06 for Pinellas County. EPC stations retained are 105, 113, 114, 132, 141, 138, 142, and 147.

The data are filtered to include only the following analytes: "Nitrate-Nitrite (N)", "Nitrogen-Total Kjeldahl", "Phosphorus- Total", and "Residues- Nonfilterable (TSS)". The units for all analytes are assumed to be mg/L.

Value

A data frame of water quality data for Manatee, Pinellas, and Hillsborough County

Examples

```
## Not run:
# import from WIN
wqdat <- util_nps_getwq(c('2021-01-01', '2023-12-31'))

# use system files
mancopth <- system.file('extdata/nps_wq_manco.txt', package = 'tbeploads')
pincopth <- system.file('extdata/nps_wq_pinco.txt', package = 'tbeploads')
wqdat <- util_nps_getwq(c('2021-01-01', '2023-12-31'), mancopth = mancopth, pincopth = pincopth)

## End(Not run)
```

`util_nps_landsoil` *Utility function for non-point source (NPS) ungaged workflow to create land use and soil data*

Description

Utility function for non-point source (NPS) ungaged workflow to create land use and soil data

Usage

```
util_nps_landsoil(tbbase)
```

Arguments

`tbbase`	Input data frame returned from `util_nps_tbbase`

Value

A data frame summarizing land use and soil by bay segment, sub-basin, drainage feature, CLUCSID, hydrologic group, and improved status.

Examples

```
data(tbbase)
util_nps_landsoil(tbbase)
```

util_nps_landsoilrc — *Utility function to create non-point source (NPS) ungaged land use and soil runoff coefficients*

Description

Utility function to create non-point source (NPS) ungaged land use and soil runoff coefficients

Usage

```
util_nps_landsoilrc(tbbase, yrexp = c(2021:2023))
```

Arguments

tbbase	Data frame returned from util_nps_tbbase containing land use and soil data.
yrexp	Years to expand the data frame to include all months for each year.

Value

A data frame with land use (CLUCSID) and soil runoff coefficients by year and month.

Examples

```
data(tbbase)

util_nps_landsoilrc(tbbase, yrexp = c(2021:2023))
```

util_nps_lusumm — *Summarize non-point source (NPS) ungaged loads by land use*

Description

Summarize non-point source (NPS) ungaged loads by land use

Usage

```
util_nps_lusumm(
  dat,
  summ = c("basin", "segment", "all"),
  summtime = c("month", "year")
)
```

Arguments

`dat`	Input data frame as an intermediate result from `anlz_nps`
`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, loading data in tons per month or year depending on the `summtime` argument, and hydrologic load in million cubic meters per month or year depending on the `summtime` argument.

Examples

```
dat <- data.frame(
  bay_seg = rep(1:2, each = 6),
  basin = rep(c("02304500", "02306647"), each = 6),
  yr = rep(2021:2022, each = 3, times = 2),
  mo = rep(1:3, times = 4),
  clucsid = rep(1:3, times = 4),
  tnload = c(150, 250, 50, 180, 300, 40, 160, 270, 45, 170, 280, 35),
  tpload = c(15, 35, 8, 18, 42, 6, 16, 38, 7, 17, 40, 5),
  tssload = c(1200, 3500, 400, 1400, 4000, 350, 1300, 3800, 380, 1350, 3900, 320),
  bodload = c(800, 1500, 200, 900, 1800, 180, 850, 1600, 190, 870, 1650, 170),
  h2oload = c(50000, 80000, 25000, 55000, 85000, 22000, 52000, 82000, 23000, 53000, 83000, 21000)
)

util_nps_lusumm(dat, summ = 'basin', summtime = 'month')
```

---

util_nps_preplog — *Utility function for non-point source (NPS) ungaged workflow to prepare land use and soil data for logistic regression*

Description

Utility function for non-point source (NPS) ungaged workflow to prepare land use and soil data for logistic regression

Usage

```
util_nps_preplog(tbbase)
```

Arguments

tbbase	Input data frame returned from util_nps_tbbase.

Value

A data frame with land use and soil areas in long format with bay segment and basins in the rows.

Examples

```
util_nps_preplog(tbbase)
```

---

util_nps_preprain — *Prep rain data for non-point source (NPS) ungaged load estimates*

Description

Prep rain data for non-point source (NPS) ungaged load estimates

Usage

```
util_nps_preprain(rain, yrrng = NULL)
```

Arguments

rain	data frame of rainfall data, see rain
yrrng	numeric vector of a single year or a two year range to filter the data, defaults to NULL (no filtering)

Details

The inverse distance weighting scheme of the drainage basin to the rain gauge is used (with nps_distance) to estimate a cumulative monthly total rainfall, including lags of one and two months.

Value

A data frame of monthly total rainfall (inches) by drainage basin

Examples

```
util_nps_preprain(rain)

util_nps_preprain(rain, yrrng = c(2021))
```

util_nps_segment — *Create bay segment column for non-point source (NPS) load data*

Description

Create bay segment column for non-point source (NPS) load data

Usage

```
util_nps_segment(dat)
```

Arguments

dat	data frame with a basin and bay_seg columns

Details

This is a simple helper function used internally with anlz_nps and util_nps_lusumm to create a segment column based on the basin and bay_seg columns.

Value

The same data frame with an additional segment column indicating the major bay segment associated with each row

Examples

```
dat <- data.frame(
  basin = c("LTARPON", "TBYPASS", "02300500", "206-4", "EVERSRES", "UNKNOWN"),
  bay_seg = c(1, 2, 3, 4, 7, NA)
)
util_nps_segment(dat)
```

util_nps_tbbase — *Create unioned base layer for non-point source (NPS) ungaged load estimation in the Tampa Bay watershed*

Description

Create unioned base layer for non-point source (NPS) ungaged load estimation in the Tampa Bay watershed

Usage

```
util_nps_tbbase(
  tblu,
  tbsoil,
  gdal_path = NULL,
  chunk_size = NULL,
  cast = FALSE,
  verbose = TRUE
)
```

Arguments

`tblu`	sf object of land use/land cover in the Tampa Bay watershed, currently `link{tblu2023}`
`tbsoil`	sf object `link{tbsoil}` of soil data in the Tampa Bay watershed
`gdal_path`	Character string specifying the path to GDAL binaries (e.g., "C:/OSGeo4W/bin"). If NULL (default), assumes GDAL is in system PATH.
`chunk_size`	Integer. For large datasets, process in chunks of this many features. Set to NULL (default) to process all at once. This applies only to the final union with the soils data.
`cast`	Logical. If TRUE, will cast multipolygon geometries to polygons before processing. Default is FALSE, which keeps multipolygons as is (usually faster).
`verbose`	Logical. If TRUE, will print progress messages. Default is TRUE.

Details

Relies heavily on `util_nps_union` to perform the union operations efficiently using GDAL/OGR. All input must have the CRS of NAD83(2011) / Florida West (ftUS), EPSG:6443.

Value

A summarized data frame containing the union of all inputs showing major bay segment, sub-basin (basin), drainage feature (drnfeat), jurisdiction (entity), land use/land cover (FLUCCSCODE), CLUCSID, IMPROVED, hydrologic group (hydgrp), and area in hectares. These represent all relevant spatial combinations in the Tampa Bay watershed. Land falling under an active NPDES discharge permit (see `tbmines`) is reclassified to `CLUCSID = 22`, overriding any FLUCCS-based category, so it can be excluded from NPS load estimation in `util_aa_npsfactors`. Areas with no soil classification in `tbsoil` (`hydgrp` is `NA`, typically open water) are assigned `hydgrp = "D"` and retained rather than dropped. Basin/jurisdiction area not covered by `tblu` (also typically open bay/estuarine water) is likewise retained, assigned FLUCCS code 5400 (`CLUCSID = 17`, Saltwater) rather than dropped.

Examples

```
## Not run:
# Load required data
```

```
data(tblu2023)
data(tbsoil)
result <- util_nps_tbbase(tblu2023, tbsoil, gdal_path = "C:/OSGeo4W/bin", chunk_size = 1000)

## End(Not run)
```

`util_nps_union` *Fast Spatial Intersection and Union Using GDAL*

Description

Performs a spatial intersection and union of two sf objects using GDAL's optimized spatial operations. This function is significantly faster than native sf operations for large datasets.

Usage

```
util_nps_union(
  sf1,
  sf2,
  gdal_path = NULL,
  chunk_size = NULL,
  cast = FALSE,
  verbose = TRUE
)
```

Arguments

`sf1`	An sf object containing polygons. All non-geometry columns will be preserved in the output
`sf2`	An sf object containing polygons. All non-geometry columns will be preserved in the output
`gdal_path`	Character string specifying the path to GDAL binaries (e.g., "C:/OSGeo4W/bin"). If NULL (default), assumes GDAL is in system PATH
`chunk_size`	Integer. For large datasets, process in chunks of this many features from sf1. Set to NULL (default) to process all at once
`cast`	Logical. If TRUE, will cast multipolygon geometries to polygons before processing. Default is FALSE, which keeps multipolygons as is (usually faster).
`verbose`	Logical. If TRUE, will print progress messages. Default is TRUE.

Value

An sf object containing the spatial intersection of sf1 and sf2, with geometries unioned by unique combinations of all attributes from both input objects

This function uses GDAL's ogr2ogr utility to perform spatial intersection operations, which can be much faster than sf's native functions for large datasets. The process:

1. Exports both sf objects to temporary GeoPackage files
2. Combines them into a single file
3. Dynamically builds SQL query based on actual column names
4. Uses SQL with spatial functions to find intersections
5. Groups and unions results by all attribute combinations

For very large datasets that cause memory issues, the function can process data in chunks.

The function automatically detects all non-geometry columns from both input objects and includes them in the intersection operation.

Note

Requires GDAL/OGR to be installed and accessible. On Windows, this is typically provided by OSGeo4W or QGIS installations, downloadable at https://trac.osgeo.org/osgeo4w/.

Examples

```
## Not run:
data(tbsubshed)
data(tbjuris)
result <- util_nps_union(
  sf1 = tbsubshed,
  sf2 = tbjuris,
  "C:/OSGeo4W/bin"
  )

## End(Not run)
```

---

util_nps_unionchunk *Helper function for union operation*

Description

Helper function for union operation

Usage

```
util_nps_unionchunk(sf1, sf2, chunk_size, verbose = TRUE)
```

Arguments

sf1	First sf object
sf2	Second sf object
chunk_size	Integer. For large datasets, process in chunks of this many features from sf1.
verbose	Logical. If TRUE, will print progress messages. Default is TRUE.

Details

Used internally by util_nps_union. See the help file for more details.

Value

An sf object containing the spatial intersection of sf1 and sf2, with geometries unioned by unique combinations of all attributes from both input objects.

Examples

```
## Not run:
data(tbjuris)
data(tbsubshed)
result <- util_nps_unionchunk(tbsubshed, tbjuris)

## End(Not run)
```

---

util_nps_unionnochunk

Helper function for union operation

Description

Helper function for union operation

Usage

```
util_nps_unionnochunk(sf1, sf2)
```

Arguments

sf1	First sf object
sf2	Second sf object

Details

Used internally by util_nps_union. See the help file for more details.

Value

An sf object containing the spatial intersection of sf1 and sf2, with geometries unioned by unique combinations of all attributes from both input objects.

Examples

```
## Not run:
data(tbjuris)
data(tbsubshed)
result <- util_nps_unionnochunk(tbsubshed, tbjuris)

## End(Not run)
```

---

util_prepverna — *Prep Verna Wellfield data for use in AD and NPS calculations*

Description

Prep Verna Wellfield data for use in AD and NPS calculations

Usage

```
util_prepverna(fl, typ, fillmis = T, mincrit = 75)
```

Arguments

`fl`	text string for the file path to the Verna Wellfield data
`typ`	character string for the type of data to prepare, either 'AD' for atmospheric deposition or 'NPS' for nonpoint source. Uses different TP calculation for each type.
`fillmis`	logical indicating whether to fill missing data with monthly means, see details
`mincrit`	numeric from 0 to 100, minimum value of the NADP `Criteria1` (percent valid sample days) and `Criteria3` (percent of precipitation captured by valid samples) completeness flags for a month to be treated as valid, see details. Ignored if either column is absent from `fl`.

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:

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

$$TP = \begin{cases} 0.01262 \cdot TN + 0.00110 & \text{if } typ = \text{“}AD\text{”} \\ 0.195 & \text{if } typ = \text{“}NPS\text{”} \end{cases}$$

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 for atmospheric deposition estimates. A constant is used for non-point source estimates.

NADP flags each month's completeness with `Criteria1` (percent of days with a valid sample) and `Criteria3` (percent of precipitation captured by valid samples), both 0-100. A month with either value below `mincrit` (default 75) is treated as missing, in addition to the raw `-9` value, before any gap-filling occurs.

Missing data (-9 values, or values failing the `mincrit` completeness screen) can be filled using monthly means from the previous five years where data exist for that month. If there are less than five previous years of data for that month, the missing value is not filled.

Years with incomplete seasonal data will be filled with NA values if `fillmis = FALSE` or filled with monthly means if `fillmis = TRUE`.

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_prepverna(fl, typ = 'AD')
```

---

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_getaoc *Get AOC LLC discharge monitoring report data from FDEP OCULUS*

Description

Downloads and parses monthly Part A Discharge Monitoring Report (DMR) PDFs for the AOC LLC industrial wastewater facility (NPDES permit FL0029653, Polk County, FL) from the Florida Department of Environmental Protection (FDEP) OCULUS public document management system. Returns Average Daily Flow and Total Nitrogen concentration for each available monitoring month.

Usage

```
util_ps_getaoc(yr, search_xlsx, pdf_dir = NULL, out_file = NULL, quiet = FALSE)
```

Arguments

`yr`	numeric (length 1), the monitoring year to retrieve (e.g., 2025).
`search_xlsx`	character, path to an OCULUS search-results spreadsheet for facility FL0029653. See Details for instructions on generating this file.
`pdf_dir`	character or NULL. Directory in which to save the downloaded PDFs. If `NULL` (default), a temporary directory is used and all PDFs are deleted when the function exits. If a path is supplied, PDFs are retained there.
`out_file`	character or NULL. If provided, the results data frame is written to this path as an `.xlsx` workbook.
`quiet`	logical. Suppress progress messages (default `FALSE`).

Details

Generating the OCULUS search spreadsheet:

The `search_xlsx` file is an Excel export from the FDEP OCULUS public document portal. To generate it:

1. Navigate to https://depedms.dep.state.fl.us in a web browser.
2. Click **Public Oculus Login** (no account required).
3. In the search form, set:
   - **Catalog**: Wastewater
   - **Profile**: Sampling
   - **Facility-Site ID**: FL0029653 (this does not go in the Permit Number field)
   - **Document Date**: From MM-DD-YYYY to MM-DD-YYYY (covering the desired monitoring year)
   - **Document Type**: Discharge Monitoring Report (DMR)
4. Run the search and export the results to Excel (use the **Export to Excel** button).
5. Save the exported `.xlsx` file and pass its path as `search_xlsx`.

The file must contain `HYPERLINK()` formulas in **column A** pointing to the individual DMR PDFs and document subject lines in **column K**. Both are present in any standard OCULUS search export.

Document selection:

The function keeps only monthly (`MO`) Part A documents for the requested year. Annual summary (`YR`), Part B daily tables, and other document types are excluded automatically. If a month has multiple submissions (e.g., a revision), the most recently filed document is used.

Reporting period vs. OCULUS label:

For some facilities the OCULUS cycle label (e.g., "JAN MO") may not align with the calendar month of the monitoring period. The month returned in the output is always derived from the **monitoring period dates inside the PDF**, not from the OCULUS label.

No Discharge (NOD):

When the facility reports No Observable Discharge, `adf_mgd` is set to `0` and `tn_mgl` is set to `NA`. A zero flow value therefore implies no discharge for that month.

Parameters not currently monitored:

Total phosphorus (TP), biochemical oxygen demand (BOD), and total suspended solids (TSS) have not been recorded at this facility in recent years and are not included in the output.

Verifying results:

On the initial run, supply a `pdf_dir` path so the downloaded PDFs are retained for inspection. Verify that the monitoring months, flow values, and TN concentrations in the output data frame match those in the PDFs.

Value

A data frame with one row per available monitoring month, sorted by month. Calendar months for which no Part A document was found are omitted (a message is printed when `quiet = FALSE`). Columns:

Column	Type	Description
`yr`	integer	Monitoring year (from the PDF monitoring period).
`mo`	integer	Calendar month (1–12, from the PDF monitoring period).
`adf_mgd`	numeric	Average Daily Flow (MGD). `0` indicates no discharge (NOD).
`tn_mgl`	numeric	Total nitrogen grab-sample concentration (mg/L). `NA` when no discharge or not reported.

Examples

```
## Not run:
# Retrieve 2025 AOC DMR data
# (requires an OCULUS search spreadsheet generated as described in Details)
df <- util_ps_getaoc(
  yr          = 2025,
  search_xlsx = "AOC_OCULUSSearchData_2025.xlsx"
)

# Keep PDFs and save results to Excel
df <- util_ps_getaoc(
  yr          = 2025,
  search_xlsx = "AOC_OCULUSSearchData_2025.xlsx",
  pdf_dir     = "~/Desktop/AOC_DMR_2025",
  out_file    = "~/Desktop/AOC_DMR_2025_results.xlsx"
)

## End(Not run)
```

`util_ps_getkerry` *Get Kerry I&F discharge monitoring report data from FDEP OCULUS*

Description

Downloads and parses monthly Part A Discharge Monitoring Report (DMR) PDFs for the Kerry I&F Contracting Company industrial wastewater facility (NPDES permit FL0037389, Hillsborough County, FL) from the Florida Department of Environmental Protection (FDEP) OCULUS public document management system. Returns Average Daily Flow, BOD, Total Nitrogen, and Total Phosphorus concentration for each available monitoring month.

Usage

```
util_ps_getkerry(
  yr,
  search_xlsx,
  pdf_dir = NULL,
  out_file = NULL,
  quiet = FALSE
)
```

Arguments

`yr`	numeric (length 1), the monitoring year to retrieve (e.g., 2025).
`search_xlsx`	character, path to an OCULUS search-results spreadsheet for facility FL0037389. See Details for instructions on generating this file.
`pdf_dir`	character or NULL. Directory in which to save the downloaded PDFs. If `NULL` (default), a temporary directory is used and all PDFs are deleted when the function exits. If a path is supplied, PDFs are retained there.
`out_file`	character or NULL. If provided, the results data frame is written to this path as an `.xlsx` workbook.
`quiet`	logical. Suppress progress messages (default `FALSE`).

Details

Generating the OCULUS search spreadsheet:

The `search_xlsx` file is an Excel export from the FDEP OCULUS public document portal. To generate it:

1. Navigate to https://depedms.dep.state.fl.us in a web browser.
2. Click **Public Oculus Login** (no account required).
3. In the search form, set:
   - **Catalog**: Wastewater
   - **Profile**: Sampling
   - **Facility-Site ID**: FL0037389 (this does not go in the Permit Number field)
   - **Document Date**: From MM-DD-YYYY to MM-DD-YYYY (covering the desired monitoring year)
   - **Document Type**: Discharge Monitoring Report (DMR)
4. Run the search and export the results to Excel (use the **Export to Excel** button).
5. Save the exported `.xlsx` file and pass its path as `search_xlsx`.

The file must contain `HYPERLINK()` formulas in **column A** pointing to the individual DMR PDFs and document subject lines in **column K**. Both are present in any standard OCULUS search export. Also note that the search may return other reports (e.g., toxicity results), which can be safely removed from the Excel file.

Document selection:

The function keeps only monthly (`MO`) Part A documents for the requested year. Annual summary (`YR`) documents are excluded automatically. If a month has multiple submissions (e.g., a revision such as `"DMR (R) ..."`), the most recently filed document is used.

Reporting outfall:

Kerry I&F reports flow separately for three internal monitoring sites (`FLW-1`, `FLW-2`, and `FLW-3`). `FLW-3` is the calculated combined flow (`FLW-1` + `FLW-2`) and is the value returned here, consistent with the outfall label used in prior years' data for this facility.

No Discharge (NOD) and Acceptable Non-Collection (ANC):

`NOD` (No Observable Discharge) is treated as zero flow. `ANC` (Acceptable Not Collected) is treated as `NA`. Below-detection-limit values (e.g. `"<2"`) are returned as the reported numeric detection limit (`2`), not halved.

Parameters not currently monitored:

Total suspended solids (TSS) is not recorded at this facility and is returned as `NA` for all months.

Verifying results:

On the initial run, supply a `pdf_dir` path so the downloaded PDFs are retained for inspection. Verify that the monitoring months, flow values, and concentrations in the output data frame match those in the PDFs.

Value

A data frame with one row per available monitoring month, sorted by month. Calendar months for which no Part A document was found are omitted (a message is printed when `quiet = FALSE`). Columns:

Column	Type	Description
`yr`	integer	Monitoring year (from the PDF monitoring period).
`mo`	integer	Calendar month (1-12, from the PDF monitoring period).
`outfall`	character	Always `"FLW-3"` (the combined flow).
`flow_mgd`	numeric	Average Daily Flow (MGD), monthly average. `0` indicates no discharge (NOD).
`bod_mgl`	numeric	BOD, Carbonaceous (mg/L), monthly maximum. `NA` when ANC or not reported.
`tss_mgl`	numeric	Always `NA` (not monitored at this facility).
`tn_mgl`	numeric	Total nitrogen grab-sample concentration (mg/L). `NA` when ANC or not reported.
`tp_mgl`	numeric	Total phosphorus grab-sample concentration (mg/L). `NA` when ANC or not reported.

Examples

```
## Not run:
# Retrieve 2025 Kerry I&F DMR data
df <- util_ps_getkerry(
  yr          = 2025,
  search_xlsx = "Kerry_OCULUSSearchData_2025.xlsx"
)

# Keep PDFs and save results to Excel
df <- util_ps_getkerry(
  yr          = 2025,
  search_xlsx = "Kerry_OCULUSSearchData_2025.xlsx",
  pdf_dir     = "~/Desktop/Kerry_DMR_2025",
  out_file    = "~/Desktop/Kerry_DMR_2025_results.xlsx"
)

## End(Not run)
```

---

util_ps_getmacdill — *Get MacDill AFB WWTP discharge monitoring report data from FDEP OCULUS*

Description

Downloads and parses monthly Discharge Monitoring Report (DMR) PDFs for the MacDill Air Force Base wastewater treatment plant (NPDES permit FLA012124, Hillsborough County, FL) from the FDEP OCULUS public document system. Returns monthly effluent parameters for three discharge outfalls.

Usage

```
util_ps_getmacdill(
  yr,
  search_xlsx,
  pdf_dir = NULL,
  out_file = NULL,
  quiet = FALSE
)
```

Arguments

`yr`	numeric (length 1), the monitoring year to retrieve (e.g., 2025).
`search_xlsx`	character, path to an OCULUS search-results spreadsheet for facility FLA012124. See Details for instructions on generating this file.
`pdf_dir`	character or NULL. Directory in which to save the downloaded PDFs. If `NULL` (default), a temporary directory is used and all PDFs are deleted when the function exits. If a path is supplied, PDFs are retained there under human-readable names (`macdill_{yr}_{mo}_{type}.pdf`).

`out_file`	character or NULL. If provided, the results data frame is written to this path as an `.xlsx` workbook.
`quiet`	logical. Suppress progress messages (default `FALSE`).

Details

Generating the OCULUS search spreadsheet:

The `search_xlsx` file is an Excel export from the FDEP OCULUS public document portal. To generate it:

1. Navigate to https://depedms.dep.state.fl.us in a web browser.
2. Click **Public Oculus Login** (no account required).
3. In the search form, set:
   - **Catalog**: Wastewater
   - **Profile**: Sampling
   - **Facility-Site ID**: FLA012124 (not the Permit Number field)
   - **Document Date**: From MM-DD-YYYY to MM-DD-YYYY (covering the desired year)
   - **Document Type**: Discharge Monitoring Report (DMR)
4. Run the search and export the results to Excel (**Export to Excel** button).
5. Save the exported `.xlsx` and pass its path as `search_xlsx`.

The file must contain `HYPERLINK()` formulas in **column A** and document subject lines in **column K**, both of which are present in any standard OCULUS search export.

Document selection and classification:

All monthly (`MO`) documents for the requested year are downloaded and inspected. Each PDF is then classified by its actual content:

- **Part A** (monthly summary) – contains the official permit-limit table with pre-computed monthly averages and permit compliance results.
- **Part B** (daily sample results) – contains a day-by-day table of flow and effluent quality measurements for a given month.

The OCULUS document labels ("Part A", "Part B") are not always reliable for this facility, so the function detects content type from the PDF text. Annual summary (`YR`) documents are excluded automatically.

Some older submissions are scanned image PDFs with no embedded text layer. These cannot be parsed and are saved as `macdill_unclassified_{document_subject}.pdf` in `pdf_dir` (when `pdf_dir` is supplied) so you can identify them from their OCULUS subject line and enter the values manually if needed.

Hybrid extraction methodology:

Where a Part B daily table is available for a given calendar month, BOD and TSS are computed as the mean of all observed daily values using the substitution rules `<1 \u{2192} 0.5` and `<2 \u{2192} 1.0` (consistent with the 2022–2024 reporting methodology). Monthly average flow is also derived from Part B daily readings when available. For months with no Part B, Part A monthly-summary values are used for BOD, TSS, and flow.

Total nitrogen (`tn_mgl`) is sourced from Part B page 2 (column 00620, biweekly readings averaged) when available, falling back to the Part A monthly maximum when no Part B is present.

Monitoring period vs. OCULUS label:

The OCULUS document cycle labels (e.g., "JAN MO") do not always align with the calendar month of the monitoring period. The `mo` column in the output is always derived from the monitoring period dates inside the PDF, not from the OCULUS label.

No Discharge (NOD) and acceptable non-collection (ANC):

`NOD` (No Observable Discharge) is treated as zero flow and `NA` concentration. `ANC` (Acceptable Not Collected) is treated as `NA`.

Value

A data frame with three rows per available monitoring month (one per outfall), sorted by month then outfall. Calendar months for which no usable document was found are omitted (a message is printed when `quiet = FALSE`). Columns:

Column	Type	Description
`yr`	integer	Monitoring year (from the PDF monitoring period).
`mo`	integer	Calendar month (1–12).
`outfall`	character	Outfall ID: `"R-001"`, `"R-002"`, or `"R-003"`.
`flow_mgd`	numeric	Average Daily Flow (MGD). `0` when NOD.
`bod_mgl`	numeric	BOD (mg/L). `NA` when ANC or not collected.
`tss_mgl`	numeric	TSS (mg/L). `NA` when ANC or not collected.
`tn_mgl`	numeric	Total nitrogen as NO3-N (mg/L), R-003 only; mean of biweekly Part B readings when avai
`verify`	logical	`TRUE` when one or more concentration values (TSS for R-001/R-003, TN for R-003) are sing

See Also

`util_ps_checkfls()`, `anlz_ips()`

Examples

```
## Not run:
# Retrieve 2025 MacDill DMR data
df <- util_ps_getmacdill(
  yr          = 2025,
  search_xlsx = "MacDill_OCULUSSearchData_2025.xlsx"
)

# Keep PDFs and save results to Excel
df <- util_ps_getmacdill(
  yr          = 2025,
  search_xlsx = "MacDill_OCULUSSearchData_2025.xlsx",
  pdf_dir     = "~/Desktop/MacDill_DMR_2025",
  out_file    = "~/Desktop/MacDill_DMR_2025_results.xlsx"
)

## End(Not run)
```

`util_ps_misc` *Fill missing concentration data for miscellaneous industrial point source facilities*

Description

Fill total phosphorus, total suspended solids, and biological oxygen demand for miscellaneous industrial point source records where one or more of these parameters are unmeasured and must be estimated from historical averages.

Usage

```
util_ps_misc(dat)
```

Arguments

`dat`	data frame for a single facility with columns matching the standard clean IPS format: `Permit.Number`, `Facility.Name`, `Outfall.ID`, `Year`, `Month`, `Average.Daily.Flow..ADF...mgd.`, `Total.N`, `Total.N.Unit`, `Total.P`, `Total.P.Unit`, `TSS`, `TSS.Unit`, `BOD`, `BOD.Unit`

Details

Unlike Mosaic facilities (see `util_ps_mosaic`), the facilities handled here already report some concentration parameters from DMR data. This function fills only those parameters that are chronically unmeasured, using historical averages or standard assumptions. Measured values are preserved unless the fill rule explicitly overrides them (e.g., Trademark Nitrogen TP and BOD are always set to historical means regardless of any reported value).

The general fill rule is: when flow is zero or missing, all concentrations are set to `NA`; when flow is positive, fill values are applied to unmeasured parameters and measured values are retained for everything else. Unit strings are standardised to `"mg/L"` when a value is present and `""` otherwise.

As for the `util_ps_mosaic` function, #' Note that this function may need to be updated if new data become available or if there are changes in the fill rules. The current fill values and rules are based on historical permit compliance data and may not reflect future conditions.

Value

The input data frame with `Total.P`, `TSS`, `BOD`, and their unit columns updated. All other columns are returned unchanged except that unit strings are standardised and concentrations are set to `NA` for zero- or missing-flow months.

Fill values (mg/L)

Alpha/ Owens Corning (D-001) TP = 1, TSS = 2, BOD = 6; last recorded BOD and TSS from Dec 2017 minimal discharge; TP from Grizzle-Figg limits

Big Bend Station (I-130) TP = 1.73, TSS = 12.11, BOD = 9.6; TP and TSS from 95-98 averages; BOD from Harper 1994

Brewster Phosphogypsum Stack System (D-001) BOD = 9.6 (Harper 1994); TP and TSS from actual DMR measurements

Busch Gardens (D-002) TSS = 5, BOD = 9.6 (Harper 1994); TP from actual DMR measurements

Coronet Industries (D-002, D-004, D-005) BOD = 9.6 (Harper 1994); TP and TSS from actual DMR measurements

CSX - ROCKPORT NEWPORT (D-001, D-002, D-008, D-010) TP = 0, TSS = 0, BOD = 9.6; no TP or TSS info; BOD same as Winston Yard previously

CSX Winston Yard (D-002) TP = 0, TSS = 0, BOD = 9.6 (Harper 1994); no TP or TSS info

Duke Energy-Bartow Plant (I-002) TP = 0, BOD = 9.6 (Harper 1994); TSS from actual DMR measurements; no TP measurements

DRS Piney Point (D-001, D-002, D-003) BOD = 9.6 (Harper 1994); TP and TSS from actual DMR measurements

Estech Agricola (D-001) BOD = 9.6 (Harper 1994); TP and TSS from actual DMR measurements

H.L. Culbreath Bayside Power Station (I-038) TP = 1, TSS = 12.96, BOD = 9.6; TP from Grizzle-Figg limits; TSS avg 2012-2014; BOD Harper 1994

Lowry Park Zoo (D-001) TSS = 5, BOD = 9.6 (Harper 1994); TP from actual DMR measurements; TN and TP for September 2023 filled with adjacent-month means (TN = 0.967, TP = 0.17)

Trademark Nitrogen Corporation (D-001) TP = 0.13333, BOD = 1.09833; both filled with means from 1995-1998 loadings regardless of measured values; TSS from actual DMR measurements

See Also

util_ps_mosaic for filling missing data for Mosaic facilities

Examples

```
dat <- data.frame(
  Permit.Number                  = rep('FL0185833', 3),
  Facility.Name                  = rep('Busch Gardens', 3),
  Outfall.ID                     = rep('D-002', 3),
  Year                           = rep(2022L, 3),
  Month                          = 1:3,
  Average.Daily.Flow..ADF...mgd. = c(0.78, 0.50, 0),
  Total.N                        = c(0.45, 0.06, NA),
  Total.N.Unit                   = c('mg/L', 'mg/L', ''),
  Total.P                        = c(0.08, 0.08, NA),
  Total.P.Unit                   = c('mg/L', 'mg/L', ''),
  TSS                            = c(NA, NA, NA),
  TSS.Unit                       = c('', '', ''),
  BOD                            = c(NA, NA, NA),
```

```
  BOD.Unit                        = c('', '', '')
)
util_ps_misc(dat)
```

`util_ps_mosaic` *Fill missing concentration data for Mosaic industrial point source facilities*

Description

Fill total phosphorus, total suspended solids, and biological oxygen demand for Mosaic industrial point source records, which contain only flow and total nitrogen.

Usage

```
util_ps_mosaic(dat)
```

Arguments

`dat`	data frame for a single Mosaic facility with columns `Facility.Name`, `Outfall.ID`, `Year`, `Month`, `Average.Daily.Flow..ADF...mgd.`, and `Total.N`

Details

Mosaic data contain only average daily flow (MGD) and total nitrogen (TN, mg/L). Total phosphorus (TP), total suspended solids (TSS), and biological oxygen demand (BOD) are not measured and are filled with historical averages derived from earlier permit compliance data (2008-2011 averages unless otherwise noted).

The general fill rule is: if average daily flow is zero or missing, TP, TSS, and BOD are set to `NA`; if flow is positive they are assigned the per-facility (or per-outfall) historical average. A few facilities always receive the historical fill regardless of flow and these include Mosaic Bartow, Green Bay, New Wales, South Pierce, and Bonnie outfall D-003. Missing flow values are replaced with zero. A permit number is added for each facility from FDEP permit records.

Value

A data frame with columns: `Permit.Number`, `Facility.Name`, `Outfall.ID`, `Year`, `Month`, `Average.Daily.Flow..ADF...mgd.`, `Total.N`, `Total.N.Unit`, `Total.P`, `Total.P.Unit`, `TSS`, `TSS.Unit`, `BOD`, `BOD.Unit`.

Fill values (mg/L)

Mosaic Bartow (all outfalls) TP = 1.61, TSS = 8.38, BOD = 9.6, always filled

Mosaic Black Point (fka Yara) I-002 TP = 0.56, TSS = 8.2, BOD = 2.45

Mosaic Bonnie D-005 TP = 0.18, TSS = 3.40, BOD = 9.6

Mosaic Bonnie D-006 TP = 0.85, TSS = 1.63, BOD = 9.6

Mosaic Bonnie D-04A TP = 0.50, TSS = 2.26, BOD = 9.6

Mosaic Bonnie D-07A TP = 0.23, TSS = 1.70, BOD = 9.6

Mosaic Bonnie I-001 TP = 0.73, TSS = 26.46, BOD = 9.6

Mosaic Bonnie I-003 TP = 2.30, TSS = 6.58, BOD = 9.6, always filled

Mosaic Four Corners D-001, D-003 TP = 1.12, TSS = 12.7, BOD = 9.6

Mosaic Green Bay (all outfalls) TP = 4.23, TSS = 7.90, BOD = 9.6, always filled

Mosaic Hookers Point (Ammonia Terminal) (all outfalls) TP = 25.3, TSS = 9.35, BOD = 9.6

Mosaic Lonesome (all outfalls) TP = 0.016, TSS = 2.4, BOD = 9.6

Mosaic Mulberry D-002 TP = 2.21, TSS = 5.07, BOD = 9.6

Mosaic Mulberry Phospho Stack D-01F TP = 6.67, TSS = 6.78, BOD = 9.6

Mosaic New Wales (all outfalls) TP = 0.27, TSS = 4.9, BOD = 9.6, always filled

Mosaic Nichols (all outfalls) TP = 0.21, TSS = 1.95, BOD = 1.85

Mosaic Plant City (all outfalls) TP = 0.65, TSS = 12.0, BOD = 9.6

Mosaic Port Sutton (all outfalls) TP = 0.66, TSS = 14.4, BOD = 9.6

Mosaic Riverview D-001, D-021, D-022, D-05A, D-05B TP = 10.65, TSS = 11.49, BOD = 1.8

Mosaic Riverview D-025 TP = 10.65, TSS = 8.70, BOD = 1.8

Mosaic South Pierce (all outfalls) TP = 1.50, TSS = 3.58, BOD = 9.6, always filled

Mosaic Tampa Marine Terminal D-SW1 TP = 22.0, TSS = 49.6, BOD = 9.6

Mosaic Tampa Marine Terminal D-SW3 TP = 25.3, TSS = 9.33, BOD = 9.6

Mosaic Hookers Prairie and Mosaic Riverview Stack Closure have no established fill values; TP, TSS, and BOD will be `NA` for those facilities.

Facilities with named outfall rules (Bonnie, Four Corners, Mulberry, Mulberry Phospho Stack, Riverview, and Tampa Marine Terminal) require that every outfall present in `dat` matches a known entry. An error is returned for unrecognised outfalls.

Note that this function may need to be updated if new data become available or if there are changes in the fill rules. The current fill values and rules are based on historical permit compliance data and may not reflect future conditions.

See Also

`util_ps_misc` for filling missing data for miscellaneous industrial point source facilities

Examples

```
dat <- data.frame(
  Facility.Name = rep('Mosaic Bartow', 3),
  Outfall.ID = rep('D-001', 3),
  Year = rep(2022L, 3),
  Month = 1:3,
  Average.Daily.Flow..ADF...mgd. = c(0.57, 0, 0.43),
  Total.N = c(3.94, NA, 2.11)
)
util_ps_mosaic(dat)
```

util_ps_pascoreuse *Create Pasco Reuse point source input data*

Description

Create Pasco Reuse point source input data from external hydrologic volume inputs and a constant TN concentration

Usage

```
util_ps_pascoreuse(
  yr,
  res,
  golf = rep(0, length(yr)),
  ribs = rep(0, length(yr)),
  ag = rep(0, length(yr)),
  tn_conc = 9,
  n_coastal = 2
)
```

Arguments

`yr`	integer vector of years
`res`	numeric vector of residential reuse volumes (million gallons per year x 1000), one value per year
`golf`	numeric vector of golf course reuse volumes (million gallons per year x 1000), one value per year
`ribs`	numeric vector of rapid infiltration basin volumes (million gallons per year x 1000), one value per year
`ag`	numeric vector of agricultural reuse volumes (million gallons per year x 1000), one value per year
`tn_conc`	numeric, constant TN concentration in mg/L applied to all records (default `9`)
`n_coastal`	integer, number of coastal bay segment codes over which flow is divided equally (default `2`)

Details

Pasco County reuse hydrologic inputs are provided externally as annual volumes (in million gallons per year x 1000) broken into land use categories: residential, golf courses, rapid infiltration basins (RIBs), and agriculture. These are summed and converted to million gallons (MG) per year, then distributed evenly across 12 months and divided equally across `n_coastal` coastal bay segment codes to produce average daily flow in MGD. A constant TN concentration of `tn_conc` mg/L is assumed. TP, TSS, and BOD are set to zero.

The output format matches the standard point source input data frame used by `anlz_dps_facility`.

Value

A `data.frame` with one row per year-month combination and columns:

Permit.Number character, `"PascoReuse"`
Facility.Name character, `"Pasco Reuse"`
Outfall.ID character, `"R-001"`
Year integer
Month integer, 1–12
Average.Daily.Flow..ADF...mgd. numeric, average daily flow in MGD
Total.N numeric, TN concentration in mg/L
TN.Unit character, `"mg/l"`
Total.P numeric, 0
TP.Unit character, `"mg/l"`
TSS numeric, 0
TSS.Unit character, `"mg/l"`
BOD numeric, 0
BOD.Unit character, `"mg/l"`

Examples

```
util_ps_pascoreuse(
  yr   = 2022:2024,
  res  = c(744120, 522273, 344189),
  golf = c(0, 0, 0),
  ribs = c(0, 0, 0),
  ag   = c(169, 269, 153)
)
```

---

util_spr_getwq *Retrieve spring water quality data from APIs*

Description

Retrieve spring water quality data from APIs

Usage

```
util_spr_getwq(yrrng, raw = FALSE, verbose = TRUE)
```

Arguments

`yrrng`	integer vector of length 2, start and end year, e.g. `c(2022, 2024)`.
`raw`	logical, if `FALSE` (default) annual mean concentrations are returned (one row per spring per year). If `TRUE`, the raw, unaggregated observations are returned instead (one row per spring per sample date), with an additional `date` column in place of `yr`.
`verbose`	logical, if `TRUE` progress messages are printed.

Details

Fetches TN, TP, and TSS concentrations (mg/L) for Lithia, Buckhorn, and Sulphur springs from two external sources, returned either as annual means or as raw observations (see `raw`).

Lithia and Buckhorn springs are retrieved from the Water Atlas API (`GET /api/samplingdata/stream`), using Southwest Florida Water Management District (SWFWMD) monitoring stations 17805 (Lithia Main Spring) and 18276 (Buckhorn Main Spring) from the `WIN_21FLSWFD` data source. This is the same underlying dataset as FDEP's Impaired Waters Rule file but accessed directly via API. TSS is not routinely measured at these stations and will typically be `NA`, in which case `anlz_spr` substitutes fixed historical values.

Sulphur Spring data are retrieved via `read_importepc`, which downloads the Environmental Protection Commission of Hillsborough County (EPC) monitoring spreadsheet. Station 174 corresponds to the Sulphur Spring sampling location and provides monthly TN, TP, and TSS observations.

When `raw = FALSE`, means are computed across all observations within each calendar year, and TSS values that are `NaN` (i.e., no valid observations in a year) are converted to `NA` so that `anlz_spr` can apply the fixed fallback concentrations. When `raw = TRUE`, no aggregation is performed; each row reflects the actual sample date, and since TN, TP, and TSS are not always sampled on the same date, some rows may have `NA` for parameters not measured that day.

Value

A data frame with columns `spring`, `tn_mgl`, `tp_mgl`, and `tss_mgl`, plus `yr` (one row per spring per year) when `raw = FALSE`, or `date` (one row per spring per sample date) when `raw = TRUE`.

See Also

`anlz_spr`

Examples

```
## Not run:
wqdat <- util_spr_getwq(c(2022, 2024))

# raw, unaggregated observations instead of annual means
wqdat <- util_spr_getwq(c(2022, 2024), raw = TRUE)

## End(Not run)
```

---

util_summ — *Summarize load estimates*

Description

Summarize load estimates

Usage

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

Arguments

`dat`	Pre-processed data frame of 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 Hillsborough, 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_summ(ipsld, summ = 'entity', summtime = 'year')
```

Index

∗ **datasets**
- aa_corrections, 4
- ad_distance, 5
- allflo, 6
- allwq, 7
- clucsid, 34
- conserv_correction, 35
- contdry, 36
- contwet, 37
- dbasing, 38
- dps_allocations, 39
- emc, 40
- facilities, 40
- hydro_baseline, 41
- ml_allocations, 41
- nps_allocations, 42
- nps_distance, 43
- ps_allocations, 44
- rain, 45
- rain_historic, 46
- rcclucsid, 46
- tbbase, 50
- tbdbasin, 51
- tbfullshed, 52
- tbjuris, 53
- tblu2023, 54
- tbmines, 55
- tbsegdetail, 56
- tbsoil, 57
- tbsubshed, 58
- usgsflow, 59

∗ **show**
- show_aaassess, 47
- show_aaloads, 48

aa_corrections, 4, *11*
ad_distance, 5, *13*
allflo, 6
allwq, 7
anlz_aa, *4*, 8, *8*, *36*, *44*, *47*, *49*, *60*
anlz_ad, 12, *49*
anlz_dps, 14, *16*, *20*, *26*, *27*
anlz_dps_facility, *8*, *14*, 15, *15*, *39*, *102*
anlz_gw, 16, *49*
anlz_ips, 18, *26*, *27*
anlz_ips(), *97*
anlz_ips_facility, *8*, *18*, 19, *19*
anlz_ml, 20, *21*
anlz_ml_facility, *8*, *20*, 21, *21*
anlz_nps, *8*, 22, *26*, *27*, *78*, *80*
anlz_nps_gaged, *6*, *7*, *23*, 24
anlz_nps_psremove, *14*, *19*, 26
anlz_nps_ungaged, *6*, *23*, 28
anlz_spr, 31, *49*, *104*

clucsid, 34
conserv_correction, *9*, *10*, *12*, 35
contdry, *16*, 36, *37*, *66*, *68*
contwet, *16*, 37, *37*, *66*, *68*

dbasing, *9*, *27*, 38
dps_allocations, *9*, 39

emc, *10*, 40, *60*

facilities, *39*, 40, *42*, *44*
flextable, *48*, *50*

ggplot, *69*

hydro_baseline, *11*, 41

ml_allocations, *10*, *12*, 41

na.approx, *32*, *70*, *72*
nps_allocations, *11*, 42
nps_distance, 43, *79*

ps_allocations, *9*–*12*, 44

rain, *13*, *30*, 45, *63*, *79*

rain_historic, 46
rcclucsid, *10*, 46, *60*
read_importwq, *75*
read_importwqwin, *75*
read_waterdata_daily, *25*, *74*

sf, *51–58*, *66*, *68*
show_aaassess, 47, *49*
show_aaloads, *9*, 48
SpatRaster, *16*, *36*, *37*, *64*, *66*, *68*
system.file, *13*

tbbase, *9–11*, *30*, *35*, *36*, 50
tbdbasin, 51, *67*
tbfullshed, *37*, 52, *64*, *73*
tbjuris, 53
tblu2023, 54
tbmines, 55, *81*
tbsegdetail, 56, *66*, *68*
tbsoil, 57, *81*
tbsubshed, 58, *66*, *68*

usgsflow, 59
util_aa_npsfactors, *10*, *36*, *55*, 60, *81*
util_getrain, *13*, *14*, *45*, 61
util_gw_getcontour, *16*, *17*, *37*, 63, *66*, *68*
util_gw_getwq, *16*, 64
util_gw_grad, *17*, 66, *68*, *69*
util_gw_showgrad, 68
util_nps_fillmiswq, *23*, *25*, 69
util_nps_getextflow, *6*, *23*, *25*, *30*, 70, *72*
util_nps_getflow, *6*, *23*, *25*, *28*, *30*, 71
util_nps_getswfwmd, 73
util_nps_getusgsflow, *6*, *23*, *25*, *29*, *30*, *32*, *59*, *60*, *72*, 74
util_nps_getwq, *7*, *23*, *25*, 75
util_nps_landsoil, 76
util_nps_landsoilrc, *47*, 77
util_nps_lusumm, 77, *80*
util_nps_preplog, *23*, 78
util_nps_preprain, *23*, 79
util_nps_segment, *23*, 80
util_nps_tbbase, *30*, *35*, *55*, *57*, *60*, *76*, *77*, *79*, 80
util_nps_union, *81*, 82, *84*
util_nps_unionchunk, 83
util_nps_unionnochunk, 84
util_prepverna, *13*, *14*, *23*, 85
util_ps_addcol, 86
util_ps_checkfls, 87
util_ps_checkfls(), *97*
util_ps_checkuni, *87*, 88
util_ps_facinfo, 88
util_ps_fillmis, 89
util_ps_fixoutfall, 90
util_ps_getaoc, 90
util_ps_getkerry, 92
util_ps_getmacdill, 95
util_ps_misc, 98, *101*
util_ps_mosaic, *98*, *99*, 100
util_ps_pascoreuse, 102
util_spr_getwq, *31–33*, 103
util_summ, 104

wrap, *36*, *37*