

# Package: peptools (via r-universe)

August 10, 2024

**Title** Analysis Tools for Importing, Wrangling, and Summarizing Suffolk County Water Quality Data

**Version** 1.1.0

**Date** 2024-03-05

**Description** Analysis tools for importing, wrangling, and summarizing Suffolk County water quality data. Functions are used to create reporting materials.

**Depends** R (>= 3.5)

**Imports** dataRetrieval, DescTools, dplyr, ggplot2, htr, jsonlite, leaflet, lubridate, mapview, plotly, purrr, reactable, readxl, RColorBrewer, scales, sf, stringr, survival, tidyr

**License** CC0

**Encoding** UTF-8

**LazyData** true

**LazyDataCompression** xz

**RoxygenNote** 7.2.3

**Suggests** knitr, patchwork, rmarkdown, tibble

**VignetteBuilder** knitr

**URL** <https://tbep-tech.github.io/peptools>,  
<https://tbep-tech.github.io/peptools/>

**Repository** <https://fawda123.r-universe.dev>

**RemoteUrl** <https://github.com/tbep-tech/peptools>

**RemoteRef** HEAD

**RemoteSha** 2d14a4111b4642a08fa3ad52f31d9c4954be23fe

## Contents

anlz_attainpep . . . . .	2
anlz_dodlypep . . . . .	3

anlz_domopep . . . . .	4
anlz_entpep . . . . .	5
anlz_medpep . . . . .	5
beaches . . . . .	6
dodat . . . . .	7
entdat . . . . .	7
pepseg . . . . .	8
pepstations . . . . .	9
peptargets . . . . .	10
rawdat . . . . .	11
read_pepdo . . . . .	12
read_pepent . . . . .	13
read_pepwq . . . . .	13
show_allthrpep . . . . .	14
show_boxpep . . . . .	15
show_domatrix . . . . .	16
show_entmatrix . . . . .	17
show_matrixpep . . . . .	18
show_plotlypep . . . . .	19
show_reactablepep . . . . .	20
show_segmatrixpep . . . . .	20
show_sitemappep . . . . .	21
show_thrpep . . . . .	22
show_wqmatrixpep . . . . .	23

## Index 25

---

anlz_attainpep	<i>Get attainment categories</i>
----------------	----------------------------------

---

### Description

Get attainment categories for each year and bay segment using chlorophyll and secchi depth

### Usage

```
anlz_attainpep(meddat, magdurout = FALSE, trgs = NULL)
```

### Arguments

meddat	result returned from <a href="#">anlz_medpep</a>
magdurout	logical indicating if the separate magnitude and duration estimates are returned
trgs	optional data.frame for annual bay segment water quality targets, defaults to <a href="#">peptargets</a>

### Value

A data.frame for each year and bay segment showing the attainment category

**Examples**

```
meddat <- anlz_medpep(rawdat)
anlz_attainpep(meddat)
```

---

anlz_dodlypep	<i>Analyze daily DO values relative to threshold</i>
---------------	--

---

**Description**

Analyze daily DO values relative to threshold

**Usage**

```
anlz_dodlypep(dodat, thr = 3, impute = TRUE)
```

**Arguments**

dodat	result returned from <a href="#">read_pesto</a>
thr	numeric indicating appropriate dissolved oxygen thresholds, usually 3 mg/L for acute, 4.8 mg/L for chronic
impute	logical indicating if missing dissolved oxygen values are imputed with the year, month, site average

**Details**

The [dodat](#) data object can be used as input without downloading USGS data

If `impute = TRUE`, missing dissolved oxygen values in the complete daily time series are imputed to the average for the year, month, site combination. This is often necessary to create summary values that make sense. For example, if a month has incomplete data, the maximum `below_cumsum` value will not show 30 or 31 days even if every day in the observed record is below the threshold.

**Value**

data.frame

**Examples**

```
data(dodat)
dat <- anlz_dodlypep(dodat)
dat
```

---

`anlz_domopep`*Analyze monthly DO values relative to threshold*

---

### Description

Analyze monthly DO values relative to threshold

### Usage

```
anlz_domopep(dodat, thr = 3, impute = TRUE)
```

### Arguments

<code>dodat</code>	result returned from <a href="#">read_pestdo</a>
<code>thr</code>	numeric indicating appropriate dissolved oxygen thresholds, usually 3 mg/L for acute, 4.8 mg/L for chronic
<code>impute</code>	logical indicating of missing dissolved oxygen values are imputed with the year, month, site average

### Details

The `dodat` data object can be used as input without downloading USGS data

The data are summarized as three different values, where `do_mgl` is the average of all daily DO averages across the month, `below_ave` is the proportion of days in a month when concentrations in a given day fell below the threshold (1 would mean all days had an instance of DO below the threshold, 0 would mean none), and `below_maxrun` is the maximum number of sequential days in a month when concentrations in a given day fell below the threshold (30 or 31, depending on month, would indicate all days in a month had an instance of DO below the threshold).

If `impute = TRUE`, missing dissolved oxygen values in the complete daily time series are imputed to the average for the year, month, site combination. This is often necessary to create summary values that make sense. For example, if a month has incomplete data, the `below_ave` summary may indicate a value of one if all daily averages in the available data are below the threshold, whereas the `below_maxrun` summary may indicate a maximum run of days not equal to the number of days in the month.

### Value

`data.frame`

### Examples

```
data(dodat)
dat <- anlz_domopep(dodat)
dat
```

---

anlz_entpep	<i>Count beach exceedances for enterococcus</i>
-------------	---

---

**Description**

Count beach exceedances for enterococcus

**Usage**

```
anlz_entpep(entdat, thr = 104)
```

**Arguments**

entdat	result returned from <a href="#">read_pepent</a>
thr	numeric value defining threshold for exceedance

**Details**

The exceedance threshold is set by default as 104 cfu/100 ml criterion. This is simply based on counts in a year when any value at any station was above the threshold for each 24 hour period in the record.

The samples column shows how many days of the year were sampled at each beach and the exceedances column shows how many samples were above the threshold.

**Value**

A data.frame with counts of exceedances per year for each beach

**Examples**

```
anlz_entpep(entdat)
```

---

anlz_medpep	<i>Estimate annual medians</i>
-------------	--------------------------------

---

**Description**

Estimate annual medians by segment for chlorophyll, secchi, and total nitrogen data

**Usage**

```
anlz_medpep(dat)
```

**Arguments**

dat	data.frame formatted from <a href="#">read_pepwq</a>
-----	--

**Value**

Median estimates for chlorophyll, secchi, and total nitrogen

**Examples**

```
# view median estimates
anzl_medpep(rawdat)
```

---

beaches

*Bathing beaches*

---

**Description**

Bathing beaches

**Usage**

```
beaches
```

**Format**

A data.frame object

**Examples**

```
## Not run:

library(sf)
library(dplyr)

beaches <- read.csv('inst/extdata/beaches.csv') %>%
  st_as_sf(
    coords = c('Longitude', 'Latitude'),
    crs = '+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs'
  )

save(beaches, file = 'data/beaches.RData')

## End(Not run)
```

---

dodat	<i>Dissolved oxygen data for USGS stations</i>
-------	--

---

**Description**

Dissolved oxygen data for USGS stations

**Usage**

dodat

**Format**

A data.frame object

**Examples**

```
## Not run:  
  
# 01304562 is Peconic River, 01304200 is Orient Harbor, 01304650 is Shelter Island  
dodat <- read_pepdo(site = c('01304562', '01304200', '01304650'),  
  nms = c('Peconic River', 'Orient Harbor', 'Shelter Island'))  
  
save(dodat, file = 'data/dodat.RData', compress = 'xz')  
  
## End(Not run)
```

---

entdat	<i>Raw beach pathogen data from Suffolk County</i>
--------	--

---

**Description**

Raw beach pathogen data from Suffolk County

**Usage**

entdat

**Format**

A data.frame object

**Examples**

```

## Not run:
library(dplyr)
entdat1 <- read_pepent() %>%
  filter(lubridate::year(Date) < 2022)

entdat2 <- read_pepent(path = '~/Desktop/Enterodata_2023.xlsx') %>%
  filter(lubridate::year(Date) >= 2022)

entdat <- bind_rows(entdat1, entdat2)

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

# or the file can be manually downloaded from here (format is not the same as path input from SCM)
# https://gis.suffolkcountyny.gov/portal/home/item.html?id=e3b344ff82b74762b625caciaa3e9621a
entdat <- read.csv('~/.Desktop/Beach_Water_Quality_Data.csv', header = T) |>
  dplyr::filter(Type %in% c('Enterococcus', 'Enterococci')) %>%
  dplyr::select(
    Name,
    FieldNum,
    Date = ColDate,
    value = Result,
    status = Character
  ) %>%
  dplyr::mutate(
    status = gsub('[:digit:]+|\\.', '', value),
    status = ifelse(status == '', '=', status),
    value = as.numeric(gsub('>|<', '', value)),
    Date = suppressWarnings({dplyr::case_when(
      grepl('\\/', Date) ~ lubridate::mdy(Date),
      grepl("^[[:digit:]]+$", Date) ~ as.Date(as.numeric(Date), origin = "1899-12-30"),
      T ~ NA
    ))},
    Name = dplyr::case_when(
      Name == 'Crescent Beach - Shelter Island' ~ 'Crescent Beach - Suffolk',
      T ~ Name
    )
  ) %>%
  dplyr::filter(Name %in% beaches$Name) |>
  dplyr::arrange(Name, FieldNum, Date)

save(entdat, file = 'data/entdat.RData')

## End(Not run)

```

---

 pepseg

*Polygon shapefile of segment boundaries*


---

**Description**

Polygon shapefile of segment boundaries



**Usage**

```
pepseg
```

**Format**

A sf object

**Examples**

```
## Not run:
library(sf)
library(dplyr)

pepseg <- st_read('~/Desktop/TBEP/Peconic/PEPSegments/PEP_Seg_Reeves.shp') %>%
  st_transform(crs = '+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs') %>%
  select(bay_segment = NewSegment) %>%
  st_buffer(dist = 0) %>%
  st_crop(xmin = -72.76, xmax = -71.8, ymin = 40.83, ymax = 41.2)

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

## End(Not run)
```

---

pepstations

*Bay stations by segment*

---

**Description**

Bay stations by segment

**Usage**

```
pepstations
```

**Format**

A data.frame object

**Examples**

```
## Not run:
library(tidyr)
library(sf)
library(dplyr)
library(readxl)
library(mapedit)
library(mapview)

prj <- 4326
```

```

locs <- read_xlsx('inst/extdata/stationmeta.xlsx') %>%
  select(BayStation = `Station Number`, StationName, Longitude = LON, Latitude = LAT) %>%
  mutate(
    BayStation = as.character(BayStation)
  ) %>%
  sf::st_as_sf(coords = c("Longitude", "Latitude"), crs = prj)

# # use this for spatial selection
# p <- mapview(pepseg) + mapview(locs)
# e1 <- editMap(p)
# locs[e1$finished, ] %>% pull(BayStation) %>% dput

pepstations <- list(
  `1a` = c("60280", "60275", "60270", "60265", "60266", "60260", "60250",
           "60240", "60230", "60210", "60220"),
  `1b` = c("60170", "60101", "60130", "60290", "60148"),
  `2` = c("60102", "60103", "60113", "60104", "60300", "60105", "60106",
           "60107", "60310", "60320", "60114", "60109", "60340", "60121",
           "60124", "60119", "60127", "60126", "60111", "60131", "60118"),
  `3` = c("60122", "60115", "60116", "60330", "60132", "60137", "60133",
           "60134", "60135")
) %>%
  tibble::enframe('bay_segment', 'BayStation') %>%
  unnest(BayStation) %>%
  left_join(locs, ., by = 'BayStation')
crds <- st_coordinates(pepstations)
pepstations <- pepstations %>%
  st_set_geometry(NULL) %>%
  mutate(
    Longitude = crds[, 1],
    Latitude = crds[, 2],
    bay_segment = factor(bay_segment, levels = c('1a', '1b', '2', '3'))
  )

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

## End(Not run)

```

---

 peptargets

*Bay segment targets*


---

## Description

Bay segment targets

## Usage

peptargets

**Format**

A data.frame object

**Examples**

```
## Not run:
peptargets <- tibble::tibble(
  bay_segment = factor(c('1a', '1b', '2', '3'),
    levels = c('1a', '1b', '2', '3')),
  name = factor(c('1a', '1b', '2', '3'),
    levels = c('1a', '1b', '2', '3')),
  sd_thresh = c(6.5, 6.5, 6.5, 6.5),
  chla_thresh = c(5.5, 5.5, 5.5, 5.5),
  tn_thresh = c(0.4, 0.4, 0.4, 0.4)
)
save(peptargets, file = 'data/peptargets.RData', compress = 'xz')

## End(Not run)
```

---

rawdat

*Raw data from Suffolk County*

---

**Description**

Raw data from Suffolk County

**Usage**

rawdat

**Format**

A data.frame object

**Examples**

```
## Not run:
path <- system.file("extdata", "currentdata.xlsx", package = "peptools")
rawdat <- read_pepwq(path)
save(rawdat, file = 'data/rawdat.RData', compress = 'xz')

## End(Not run)
```

---

read_pepdo	<i>Import dissolved oxygen data</i>
------------	-------------------------------------

---

**Description**

Import dissolved oxygen data

**Usage**

```
read_pepdo(
  site = c("01304562", "01304200", "01304650"),
  nms = c("Peconic River", "Orient Harbor", "Shelter Island"),
  startDate = "2013-01-01",
  endDate = "2023-12-31"
)
```

**Arguments**

site	chr string of site numbers for USGS stations
nms	chr string vector of names to reassign to site numbers
startDate	chr string of start date in YYYY-MM-DD
endDate	chr string of end date in YYYY-MM-DD

**Details**

Raw data are downloaded using the USGS dataRetrieval R package, this function is a simple wrapper to the [readNWISuv](#) function.

Note that downloading the station data with the default arguments may take a few minutes. Sites are 01304562 for Peconic River, 01304200 for Orient Harbor, 01304650 for Shelter Island.

**Value**

data.frame

**Examples**

```
## Not run:
dodat <- read_pepdo(site = c('01304562', '01304200', '01304650'),
  nms = c('Peconic River', 'Orient Harbor', 'Shelter Island'),
  StartDate = '2020-06-01', endDate = '2021-06-30')
dodat

## End(Not run)
```

---

read_pepent	<i>Import raw enterococcus data</i>
-------------	-------------------------------------

---

**Description**

Import raw enterococcus data

**Usage**

```
read_pepent(path = NULL)
```

**Arguments**

path                    chr string of path for excel file (optional)

**Details**

Data are from the ArcGIS REST Services here [https://gis.suffolkcountyny.gov/hosted/rest/services/Hosted/Beach\\_Water\\_Quality\\_Data/FeatureServer/](https://gis.suffolkcountyny.gov/hosted/rest/services/Hosted/Beach_Water_Quality_Data/FeatureServer/).

The API is queried by beach names in the [beaches](#) data object. The queries are done individually for each beach to not exceed the 2000 record limit.

Data can also be imported from an Excel if the path argument is used for the location to the file.

**Value**

data.frame

**Examples**

```
## Not run:  
entdat <- read_pepent()  
  
## End(Not run)  
head(entdat)
```

---

read_pepwq	<i>Import raw water quality data</i>
------------	--------------------------------------

---

**Description**

Import raw water quality data

**Usage**

```
read_pepwq(path)
```

**Arguments**

path                    chr string of path for excel file

**Details**

Raw data from here <https://gis.suffolkcountyny.gov/portal/home/item.html?id=5d4b53ec44204219a8da685f18>

All data prior to 1990 are removed - some exist but the data are scarce.

**Value**

data.frame

**Examples**

```
path <- system.file("extdata", "currentdata.xlsx", package="peptools")
dat <- read_pepwq(path)
dat
```

---

show\_allthrpep

*Plot annual water quality value and thresholds for all segments*

---

**Description**

Plot annual water quality values and thresholds for all bay segments

**Usage**

```
show_allthrpep(
  dat,
  param = c("chla", "sd", "tn"),
  trgs = NULL,
  yrrng = NULL,
  family = NA,
  labelexp = TRUE,
  txtlab = TRUE
)
```

**Arguments**

dat                    data frame of data returned by [read\\_pepwq](#)

param                chr string indicating which water quality value and appropriate threshold to plot, one of "chla" for chlorophyll, "sd" for secchi depth, or "tn" for total nitrogen

trgs                 optional data.frame for annual bay segment water quality thresholds, defaults to [peptargets](#)

yrrng                numeric vector indicating min, max years to include

family               optional chr string indicating font family for text labels

labelexp	logical indicating if y axis and target labels are plotted as expressions, default TRUE
txtlab	logical indicating if a text label for the target value is shown in the plot

### Details

This function is conceptually similar to [show\\_thrpep](#), but results are shown as annual medians across all bay segments for the selected parameter.

### Value

A [ggplot](#) object

### Examples

```
show_allthrpep(rawdat, param = 'chl')
```

---

show_boxpep	<i>Plot monthly chlorophyll, secchi, or tn values values for a segment</i>
-------------	--

---

### Description

Plot monthly chlorophyll, secchi, or tn values values for a bay segment

### Usage

```
show_boxpep(
  dat,
  param = c("chla", "sd", "tn"),
  yrsel = NULL,
  yrrng = NULL,
  ptsz = 0.5,
  bay_segment = c("1a", "1b", "2", "3"),
  trgs = NULL,
  family = NA,
  labelexp = TRUE,
  txtlab = TRUE
)
```

### Arguments

dat	data frame of data returned by <a href="#">read_pepwq</a>
param	chr string indicating which water quality value and appropriate threshold to plot, one of "chla" for chlorophyll, "sd" for secchi depth, or "tn" for total nitrogen
yrsel	numeric for year to emphasize, shown as separate red points on the plot
yrrng	numeric vector indicating min, max years to include

ptsz	numeric indicating point size of observations not in yrse1
bay_segment	chr string for the bay segment, one of "1a", "1b", "2", or "3"
trgs	optional data.frame for annual bay segment water quality targets, defaults to <a href="#">peptargets</a>
family	optional chr string indicating font family for text labels
labelexp	logical indicating if y axis and target labels are plotted as expressions, default TRUE
txtlab	logical indicating if a text label for the target value is shown in the plot

### Details

Points not included in yrse1 are plotted over the box plots using [position\\_jitter](#). Use ptsz = -1 to suppress.

### Value

A [ggplot](#) object

### Examples

```
show_boxpep(rawdat, bay_segment = '1a')
```

---

show_domatrix	<i>Create a colorized table for reporting dissolved oxygen data by site</i>
---------------	---

---

### Description

Create a colorized table for reporting dissolved oxygen data by site

### Usage

```
show_domatrix(
  dodat,
  site,
  show = c("below_ave", "below_maxrun"),
  txtsz = 3,
  thr = 4.8,
  impute = TRUE,
  yrrng = NULL,
  family = NA
)
```



**Arguments**

dodat	data frame of dissolved oxygen data returned by <a href="#">read_pegdo</a>
site	character string of the site to plot taken from the nms argument in <a href="#">read_pegdo</a> , usually one of "Peconic River", "Orient Harbor", or "Shelter Island"
show	chr string indicating which summary value to plot from <a href="#">anlz_domopep</a> , one of 'below_ave' or 'below_maxrun'
txtsz	numeric for size of text in the plot, applies only if asreact = FALSE
thr	numeric indicating appropriate dissolved oxygen thresholds, usually 3 mg/L for acute, 4.8 mg/L for chronic
impute	logical indicating of missing dissolved oxygen values are imputed with the year, month, site average
yrrng	numeric vector indicating min, max years to include
family	optional chr string indicating font family for text labels

**Value**

A static [ggplot](#) object is returned.

**See Also**

[anlz\\_domopep](#), [anlz\\_dodlypep](#)

**Examples**

```
show_domatrix(dodat, site = 'Peconic River')
```

---

show_entmatrix	<i>Create a colored table for beach pathogen exceedances</i>
----------------	--

---

**Description**

Create a colored table for beach pathogen exceedances

**Usage**

```
show_entmatrix(entdat, txtsz = 2, thr = 104, yrng = NULL, family = NA)
```

**Arguments**

entdat	data frame of enterococcus data returned by <a href="#">read_pegent</a>
txtsz	numeric for size of text in the plot, applies only if asreact = FALSE
thr	numeric value defining threshold for exceedance
yrng	numeric vector indicating min, max years to include
family	optional chr string indicating font family for text labels

**Value**

A static [ggplot](#) object is returned if `asreact = FALSE`, otherwise a [reactable](#) table is returned

**See Also**

[anz\\_entpep](#)

**Examples**

```
show_entmatrix(entdat)
```

---

<code>show_matrixpep</code>	<i>Create a colorized table for indicator reporting</i>
-----------------------------	---

---

**Description**

Create a colorized table for indicator reporting

**Usage**

```
show_matrixpep(
  dat,
  txtsz = 3,
  trgs = NULL,
  yrrng = NULL,
  bay_segment = c("1a", "1b", "2", "3"),
  asreact = FALSE,
  nrows = 10,
  abbrev = FALSE,
  family = NA
)
```

**Arguments**

<code>dat</code>	data frame of water quality data returned by <a href="#">read_pepwq</a>
<code>txtsz</code>	numeric for size of text in the plot, applies only if <code>asreact = FALSE</code>
<code>trgs</code>	optional <code>data.frame</code> for annual bay segment water quality targets, defaults to <a href="#">peptargets</a>
<code>yrrng</code>	numeric vector indicating min, max years to include
<code>bay_segment</code>	chr string for bay segments to include, one to all of "1a", "1b", "2", or "3"
<code>asreact</code>	logical indicating if a <a href="#">reactable</a> object is returned
<code>nrows</code>	if <code>asreact = TRUE</code> , a numeric specifying number of rows in the table
<code>abbrev</code>	logical indicating if text labels in the plot are abbreviated as the first letter
<code>family</code>	optional chr string indicating font family for text labels

**Value**

A static `ggplot` object is returned if `asreact = FALSE`, otherwise a `reactable` table is returned

**Examples**

```
show_matrixpep(rawdat)
```

---

show_plotlypep	<i>Plot chlorophyll and secchi data together with matrix outcomes</i>
----------------	---

---

**Description**

Plot chlorophyll and secchi data together with matrix outcomes

**Usage**

```
show_plotlypep(  
  dat,  
  bay_segment = c("1a", "1b", "2", "3"),  
  yrrng = NULL,  
  chllim = NULL,  
  seclim = NULL  
)
```

**Arguments**

<code>dat</code>	data.frame formatted from <code>read_pepwq</code>
<code>bay_segment</code>	chr string for the bay segment, one of "1a", "1b", "2", or "3"
<code>yrrng</code>	numeric for year range to plot
<code>chllim</code>	numeric vector of length two indicating range for the chlorophyll y-axis
<code>seclim</code>	numeric vector of length two indicating range for the secchi y-axis

**Value**

An interactive plotly object

**Examples**

```
show_plotlypep(rawdat)
```

---

show_reactablepep	<i>Create reactable table from matrix data</i>
-------------------	--

---

**Description**

Create reactable table from matrix data

**Usage**

```
show_reactablepep(totab, colfun, nrows = 10)
```

**Arguments**

totab	A data frame in wide format of summarized results
colfun	Function specifying how colors are treated in cell background
nrows	numeric specifying number of rows in the table

**Details**

This function is used internally within [show\\_matrixpep](#)

**Value**

A [reactable](#) table

---

show_segmatrixpep	<i>Create a colored table for water quality outcomes and exceedances by segment</i>
-------------------	---

---

**Description**

Create a colored table for water quality outcomes by segment that includes the management action and chlorophyll, and secchi exceedances

**Usage**

```
show_segmatrixpep(  
  dat,  
  txtsz = 3,  
  trgs = NULL,  
  yrrng = NULL,  
  bay_segment = c("1a", "1b", "2", "3"),  
  abbrev = FALSE,  
  family = NA  
)
```

**Arguments**

dat	data.frame formatted from <a href="#">read_pegwq</a>
txtsz	numeric for size of text in the plot, applies only if tab = FALSE
trgs	optional data.frame for annual bay segment water quality targets, defaults to <a href="#">peptargets</a>
yrrng	numeric vector indicating min, max years to include
bay_segment	chr string for bay segments to include, one to all of "1a", "1b", "2", or "3"
abbrev	logical indicating if text labels in the plot are abbreviated as the first letter, applies only to center column
family	optional chr string indicating font family for text labels

**Details**

This function provides a combined output for the [show\\_wqmatrixpep](#) and [show\\_matrixpep](#) functions. Only one bay segment can be plotted for each function call.

**Value**

A static [ggplot](#) object is returned

**See Also**

[show\\_wqmatrixpep](#), [show\\_matrixpep](#)

**Examples**

```
show_segmatrixpep(rawdat, bay_segment = '1a')
```

---

show_sitemappep	<i>Map water quality data for a selected year</i>
-----------------	---

---

**Description**

Map water quality data for a selected year

**Usage**

```
show_sitemappep(
  dat,
  yrsel = NULL,
  mosel = NULL,
  param = c("chla", "sd", "tn"),
  bay_segment = c("1a", "1b", "2", "3"),
  maxrel = 0.99,
  relative = FALSE
)
```

**Arguments**

dat	data frame of data returned by <code>read_pepwq</code>
yrsel	numeric for years to plot, see details
mosel	numeric for months to plot, see details
param	chr string indicating which water quality value to plot, one of "chla" for chlorophyll, "sd" for secchi depth, or "tn" for total nitrogen
bay_segment	chr string for the bay segment, one or all of "1a", "1b", "2", or "3"
maxrel	numeric for the maximum quantile value for scaling if <code>relative = T</code> , this prevents outliers from skewing the scale
relative	logical indicating if sizes and colors are relative to the entire water quality data base, otherwise scaling is relative only for the points on the map

**Details**

Year estimates for the selected parameter are based on median observations across months. All twelve months are used if `mosel = NULL` (default). Monthly estimates for the selected parameter are based on median observations across years. All years are used if `yrsel = NULL` (default).

The color ramp and size scaling of points are reversed for Secchi observations.

**Value**

A `leaflet` object

**Examples**

```
# 2018, all months
show_sitemappep(rawdat, yrsel = 2018)

# 2018, July only
show_sitemappep(rawdat, yrsel = 2018, mosel = 7)

# July only, all years
show_sitemappep(rawdat, mosel = 7)
```

---

show\_thrpep

*Plot annual water quality value and thresholds for a segment*

---

**Description**

Plot annual water quality values and thresholds for a bay segment

**Usage**

```
show_thrpep(
  dat,
  bay_segment = c("1a", "1b", "2", "3"),
  param = c("chla", "sd", "tn"),
  ylim = NULL,
  trgs = NULL,
  yrrng = NULL,
  family = NA,
  labelexp = TRUE,
  txtlab = TRUE
)
```

**Arguments**

dat	data frame of data returned by <a href="#">read_pepwq</a>
bay_segment	chr string for the bay segment, one of "1a", "1b", "2", or "3"
param	chr string indicating which water quality value and appropriate threshold to plot, one of "chla" for chlorophyll, "sd" for secchi depth, or "tn" for total nitrogen
ylim	numeric vector of length two indicating range for the y-axis
trgs	optional data.frame for annual bay segment water quality thresholds, defaults to <a href="#">peptargets</a>
yrrng	numeric vector indicating min, max years to include
family	optional chr string indicating font family for text labels
labelexp	logical indicating if y axis and target labels are plotted as expressions, default TRUE
txtlab	logical indicating if a text label for the target value is shown in the plot

**Value**

A [ggplot](#) object

**Examples**

```
show_thrpep(rawdat, bay_segment = '1a', param = 'chl')
```

---

show\_wqmatrixpep

*Create a colorized table for chlorophyll or secchi exceedances*

---

**Description**

Create a colorized table for chlorophyll or secchi exceedances

**Usage**

```
show_wqmatrixpep(
  dat,
  param = c("chla", "sd"),
  txtsz = 3,
  trgs = NULL,
  yrrng = NULL,
  alpha = 1,
  bay_segment = c("1a", "1b", "2", "3"),
  asreact = FALSE,
  nrows = 10,
  family = NA
)
```

**Arguments**

dat	data.frame formatted from <a href="#">read_pepwq</a>
param	chr string for which parameter to plot, one of "chla" for chlorophyll or "sd" for secchi
txtsz	numeric for size of text in the plot, applies only if tab = FALSE
trgs	optional data.frame for annual bay segment water quality targets, defaults to <a href="#">peptargets</a>
yrrng	numeric vector indicating min, max years to include
alpha	numeric indicating color transparency
bay_segment	chr string for bay segments to include, one to all of "1a", "1b", "2", or "3"
asreact	logical indicating if a <a href="#">reactable</a> object is returned
nrows	if asreact = TRUE, a numeric specifying number of rows in the table
family	optional chr string indicating font family for text labels

**Value**

A static [ggplot](#) object is returned if asreact = FALSE, otherwise a [reactable](#) table is returned

**See Also**

[show\\_matrixpep](#), [show\\_segmatrixpep](#)

**Examples**

```
show_wqmatrixpep(rawdat)
```



# Index

## \* analyze

- anlz\_attainpep, 2
- anlz\_dodlypep, 3
- anlz\_domopep, 4
- anlz\_entpep, 5
- anlz\_medpep, 5

## \* datasets

- beaches, 6
- dodat, 7
- entdat, 7
- pepseg, 8
- pepstations, 9
- peptargets, 10
- rawdats, 11

## \* data

- beaches, 6
- dodat, 7
- entdat, 7
- pepseg, 8
- pepstations, 9
- peptargets, 10
- rawdats, 11

## \* read

- read\_pepdo, 12
- read\_pepent, 13
- read\_pepwq, 13

## \* visualize

- show\_allthrpep, 14
- show\_boxpep, 15
- show\_domatrix, 16
- show\_entmatrix, 17
- show\_matrixpep, 18
- show\_plotlypep, 19
- show\_reactablepep, 20
- show\_segmatrixpep, 20
- show\_sitemappep, 21
- show\_thrpep, 22
- show\_wqmatrixpep, 23

anlz\_attainpep, 2

anlz\_dodlypep, 3, 17

anlz\_domopep, 4, 17

anlz\_entpep, 5, 18

anlz\_medpep, 2, 5

beaches, 6, 13

dodat, 3, 4, 7

entdat, 7

ggplot, 15–19, 21, 23, 24

leaflet, 22

pepseg, 8

pepstations, 9

peptargets, 2, 10, 14, 16, 18, 21, 23, 24

position\_jitter, 16

rawdats, 11

reactable, 18–20, 24

read\_pepdo, 3, 4, 12, 17

read\_pepent, 5, 13, 17

read\_pepwq, 5, 13, 14, 15, 18, 19, 21–24

readNWISuv, 12

show\_allthrpep, 14

show\_boxpep, 15

show\_domatrix, 16

show\_entmatrix, 17

show\_matrixpep, 18, 20, 21, 24

show\_plotlypep, 19

show\_reactablepep, 20

show\_segmatrixpep, 20, 24

show\_sitemappep, 21

show\_thrpep, 15, 22

show\_wqmatrixpep, 21, 23