Package 'oceanexplorer'

May 9, 2026

Title Explore Our Planet's Oceans with NOAA

Version 0.1.0

Description Provides tools for easy exploration of the world ocean atlas of the US agency National Oceanic and Atmospheric Administration (NOAA). It includes functions to extract NetCDF data from the repository and code to visualize several physical and chemical parameters of the ocean. A Shiny app further allows interactive exploration of the data. The methods for data collecting and quality checks are described in several papers, which can be found here: <https://www.ncei.noaa.gov/products/world-ocean-atlas>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Suggests globals (>= 0.14.0), knitr, rmarkdown, shinytest (>= 1.5.1), spelling, testthat (>= 3.1.2), tibble, vdiffr (>= 1.0.2)

Imports stars (>= 0.5.5), shiny (>= 1.7.1), ggplot2 (>= 3.3.5), sf (>= 1.0.5), waiter (>= 0.2.5), bslib (>= 0.3.1), thematic (>= 0.1.2.1), shinyFeedback (>= 0.4.0), purrr (>= 0.3.4), miniUI (>= 0.1.1.1), rstudioapi (>= 0.13), DT (>= 0.20), fs (>= 1.5.2), glue (>= 1.6.0), shinyjs (>= 2.1.0), rlang (>= 0.4.11), maps (>= 3.4.0), ncmeta (>= 0.3.0), RNetCDF (>= 2.6.1), dplyr

Config/testthat/edition 3

VignetteBuilder knitr

Depends R (>= 4.1.0)

URL https://martinschobben.github.io/oceanexplorer/, https://martinschobben.shinyapps.io/oceanexplorer/, https://www.ncei.noaa.gov/products/world-ocean-atlas, https://github.com/MartinSchobben/oceanexplorer

BugReports https://github.com/MartinSchobben/oceanexplorer/issues

Language en-US

LazyData true

NeedsCompilation no

Author Martin Schobben [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0001-8560-0037>),
Peter Bijl [ctb] (ORCID: <https://orcid.org/0000-0002-1710-4012>)

Maintainer Martin Schobben <schobbenmartin@gmail.com>

Repository CRAN

Date/Publication 2023-10-16 19:40:02 UTC

Contents

`env_parm_labeller` *Parsing expressions for plot labels*

Description

Conveniently converts NOAA world ocean atlas parameter names into full oceanographic variable names including units for parsing in plot labels.

Usage

```
env_parm_labeller(var, prefix = character(1), postfix = character(1))
```

Arguments

`var`	Environmental parameter.
`prefix`	Prefix.
`postfix`	Postfix.

Value

Expression

Examples

```
# expression
env_parm_labeller("t_an")

# plot with temperature axis label
library(ggplot2)

ggplot() +
 geom_blank() +
 ylab(env_parm_labeller("t_an"))
```

`filter_NOAA` *Filter NOAA*

Description

This function aids filtering of NOAA datasets.

Usage

```
filter_NOAA(NOAA, depth = 0, coord = NULL, epsg = NULL, fuzzy = 0)
```

Arguments

`NOAA`	Dataset of the NOAA World Ocean Atlas (with `get_NOAA()`).
`depth`	Depth in meters
`coord`	List with named elements, matrix with dimnames, or simple feature geometry list column: `lon` for longitude in degrees, and `lat` for latitude in degrees.
`epsg`	Coordinate reference number.
`fuzzy`	If no values are returned, fuzzy uses a buffer area around the point to extract values from adjacent grid cells. The fuzzy argument is supplied in units of kilometer (great circle distance).

Details

This function helps filtering relevant data from NOAA World Ocean Atlas 3D arrays (longitude, latitude, and depth) which have been stored with `get_NOAA()`. An 2D `stars` object is returned if only providing a depth. An `sf` object is returned, when further providing coordinates, as a list (e.g. `list(lon = -120, lat = 12)`), a matrix (e.g. `cbind(lon = -120, lat = 12)`), or an `sf` object with POINT geometries. In the latter case it is import to follow the GeoJSON conventions for the order in `sf` vectors with `x` (`lon` = longitude) followed by `y` (`lat` = latitude).

Value

Either a stars object or sf dataframe.

See Also

Simple Features for R.

Examples

```
if (interactive()) {

# get atlas
NOAAatlas <- get_NOAA("oxygen", 1, "annual")

# filter atlas for specific depth and coordinate location
filter_NOAA(NOAAatlas, 30, list(lon = c(-160, -120), lat =  c(11, 12)))

}
```

---

filter_ui *NOAA filter module*

Description

This shiny module (filter_ui() + filter_server()) allows filtering of the currently loaded NOAA data via shiny textInput() interfaces.

Usage

```
filter_ui(id, extended = TRUE)

filter_server(
  id,
  NOAA,
  external,
  ivars = c("depth", "lon", "lat"),
  variable,
  extended = TRUE
)
```

Arguments

id	Namespace id shiny module.
extended	Boolean whether to build the extended module (default = TRUE).
NOAA	Reactive value for the dataset containing the locations coordinates.
external	Reactive values for latitude, longitude and depth from plot module.
ivars	Character vector for the variables for filtering.
variable	Reactivevalues for selected variable information.

Value

Shiny module.

Examples

```
# run filter module stand-alone
if (interactive()) {

library(oceanexplorer)
library(shiny)

# data
NOAA <- get_NOAA("oxygen", 1, "annual")

# gui
ui <- fluidPage(filter_ui("filter"), plot_ui("worldmap"))

# server
server <-function(input, output, session) {
 # table
 filter <- filter_server(
  "filter",
  reactive(NOAA),
  external = reactiveValues(lon = 190, lat = 33, depth = 20),
  variable = reactiveValues(variable = "temperature")
 )

 # plot data
 output_plot <- plot_server("worldmap", reactive(NOAA), filter$coord)
 }

 # run app
 shinyApp(ui, server)
 }
```

---

get_NOAA *Obtain NOAA World Ocean Atlas dataset*

Description

Retrieves data from the NOAA World Ocean Atlas.

Usage

```
get_NOAA(var, spat_res, av_period, cache = FALSE)

url_parser(var, spat_res, av_period, cache = FALSE)
```

Arguments

var	The chemical or physical variable of interest (possible choices: "temperature", "phosphate", "nitrate", "silicate", "oxygen", "salinity", "density").
spat_res	Spatial resolution, either 1 or 5 degree grid-cells (numeric) .
av_period	Temporal resolution, either "annual", specific seasons (e.g. "winter"), or month (e.g. "August").
cache	Caching the extracted NOAA file in the package's extdata directory (default = FALSE). Size of individual files is around 12 Mb. Use list_NOAA() to list cached data resources.

Details

Functions to retrieve data from the NOAA World Ocean Atlas . Data is an 3D array (longitude, latitude, and depth) and is loaded as a stars object. Check NOAA_data for available variables, respective units and their citations. The function can automatically cache the extracted files (default: cache = FALSE). The cached file will then reside in the package's extdata directory.

Value

stars object or path.

See Also

Introduction to the stars package

Examples

```
# path to NOAA server or local data source
url_parser("oxygen", 1, "annual")

if (interactive()) {

# retrieve NOAA data
get_NOAA("oxygen", 1, "annual")

}
```

---

input_ui *NOAA data module*

Description

These shiny modules control loading of data from the NOAA world ocean atlas (input_ui() + input_server()). In addition, the output_ui() + output_server() can be used to export the filtered data in csv format. The citation_ui() provides the associated references of the dataset currently loaded.

Usage

```
input_ui(id, citation = NULL, extended = TRUE)

citation_ui(id)

output_ui(id)

input_server(id, cache = FALSE)

output_server(id, NOAA, variable)
```

Arguments

id	Namespace id shiny module.
citation	Additional space for citation element.
extended	Boolean whether to build the extended module (default = TRUE).
cache	Caching the extracted NOAA file in the package's extdata directory (default = FALSE). Size of individual files is around 12 Mb. Use list_NOAA() to list cached data resources.
NOAA	Reactive value for the dataset containing the locations coordinates.
variable	Reactivevalues for selected variable information.

Value

Shiny module.

Examples

```
# run data module stand-alone
if (interactive()) {

library(oceanexplorer)
library(shiny)

# data
NOAA <- get_NOAA("oxygen", 1, "annual")

# gui
ui <- fluidPage(input_ui("NOAA"), plot_ui("worldmap"))

# server

server <-function(input, output, session) {
 # table
 NOAA <- input_server("NOAA")
 # plot data
 output_plot <- plot_server("worldmap", NOAA$data, reactive(NULL))
}
```

```
# run app
shinyApp(ui, server)
}
```

`list_NOAA` *List cached NOAA data files*

Description

List all cached NOAA data files from package's `extdata` directory.

Usage

```
list_NOAA()
```

Value

A character vector containing the names of the files in the specified directories (empty if there were no files). If a path does not exist or is not a directory or is unreadable it is skipped.

Examples

```
# show cached NOAA files
list_NOAA()
```

`NOAA_addin` *Ocean explorer addin*

Description

Wrapper function that launches the NOAA RStudio addin

Usage

```
NOAA_addin(cache = FALSE)
```

Arguments

`cache`	Caching the extracted NOAA file in the package's `extdata` directory (default = `FALSE`). Size of individual files is around 12 Mb. Use `list_NOAA()` to list cached data resources.

Value

Rstudio gadget

Examples

```
if (interactive()) {

# run RStudio addin (can also be launched from `Addins` dropdown menu)
NOAA_addin()

}
```

NOAA_app *Ocean explorer app*

Description

Wrapper function that launches the NOAA app.

Usage

```
NOAA_app(cache = FALSE)

NOAA_server(extended = TRUE, cache)
```

Arguments

cache	Caching the extracted NOAA file in the package's extdata directory (default = FALSE). Size of individual files is around 12 Mb. Use list_NOAA() to list cached data resources.
extended	Boolean whether to build the extended module (default = TRUE).

Value

Shiny app

Examples

```
if (interactive()) {

# run app
NOAA_app()

}
```

NOAA_data *NOAA variable names and units.*

Description

A dataset containing the variable names and units of data from NOAA made available through this package.

Usage

```
NOAA_data
```

Format

A tibble with 7 rows and 3 variables:

variable oceanographic variable

unit variable unit

citation citation of the dataset

Source

https://www.ncei.noaa.gov/products/world-ocean-atlas

plot_NOAA *Plotting the global NOAA World Ocean Atlas*

Description

Plots the NOAA World Ocean Atlas on worldmap including optional filtered locations.

Usage

```
plot_NOAA(NOAA, depth = 0, points = NULL, epsg = NULL, rng = NULL)
```

Arguments

NOAA	Dataset of the NOAA World Ocean Atlas (with get_NOAA()).
depth	Depth in meters.
points	Add locations of extracted point geometry (sf object).
epsg	The epsg used to project the data (currently supported 4326, 3031and 3995).
rng	A vector of two numeric values for the range of the oceanographic variable.

Details

A worldmap is plotted as an ggplot object which by default will plot the surface layer of the selected oceanographic variable. One can plot different depth slices by selecting the appropriate depth in meters (e.g., depth = 100). It is, furthermore possible to visualize the locations of data extractions with filter_NOAA(). See the examples below for a more detailed overview of this workflow. Different projections of the worldmap can be selected by supplying an epsg. Currently only three projections are allowed: 4326, 3031, and 3995, besides the original. It is possible to fix the range of the color scale (for the oceanographic variable) to a custom range. For example, one can fix the color scale to the total range of the ocean (instead of the current depth slice).

Value

ggplot2::ggplot()

Examples

```
if (interactive()) {

# data
NOAA <- get_NOAA("oxygen", 1, "annual")

# plot
plot_NOAA(NOAA)

# coordinates
pts <- filter_NOAA(NOAA, 1, list(lon = c(-160, -120), lat =  c(11,12)))

# plot
plot_NOAA(NOAA, points = pts)

}
```

---

plot_ui *NOAA plot module*

Description

This shiny module (plot_ui() + plot_server()) visualizes the loaded data according to the selected epsg projection ("original", "4326", "3031", or "3995"). In addition it provides an interactive plot interface to select location for data extraction based on a single-click.

Usage

```
plot_ui(id)

plot_server(id, NOAA, points)
```

Arguments

id	Namespace id shiny module.
NOAA	Reactive value for the dataset containing the locations coordinates.
points	Add locations of extracted point geometry.

Value

Shiny module.

Examples

```
# run plot module stand-alone
if (interactive()) {

library(oceanexplorer)
library(shiny)

# data
NOAA <- get_NOAA("oxygen", 1, "annual")

# coordinates
points <- filter_NOAA(NOAA, 1, list(lon = c(-160, -120), lat =  c(11, 12)))

# gui
ui <- fluidPage(plot_ui("plot"))

# server
server <-function(input, output, session) {
 plot_server("plot", reactive(NOAA), reactive(points))
}

# run app
shinyApp(ui, server)

}
```

---

reproject *Re-projecting spatial objects to new epsg*

Description

Easy re-projecting of the epsg of sf and stars objects.

Usage

```
reproject(obj, epsg, ...)

## S3 method for class 'sf'
```

```
reproject(obj, epsg, ...)

## S3 method for class 'stars'
reproject(obj, epsg, ...)
```

Arguments

obj	The sf or stars object to be re-projected.
epsg	The projection (currently only: "3031", or "3995").
...	Currently not supported.

Value

sf or stars object

Examples

```
if (interactive()) {
# get data
NOAA <- get_NOAA("temperature", 1, "annual")

# reproject data with new epsg
reproject(NOAA, 3031)
}
```

---

table_ui *NOAA table module*

Description

This shiny module (table_ui() + table_server()) visualizes the loaded and filtered data in a table format.

Usage

```
table_ui(id, download = NULL)

table_server(id, NOAA, variable)
```

Arguments

id	Namespace id shiny module.
download	Add download button.
NOAA	Reactive value for the dataset containing the locations coordinates.
variable	Reactivevalues for selected variable information.

Value

Shiny module.

Examples

```
if (interactive()) {
# run table module stand-alone

library(oceanexplorer)
library(shiny)

# data
NOAA <- get_NOAA("oxygen", 1, "annual")

# coordinates
points <- filter_NOAA(NOAA, 1, list(lon = c(-160, -120), lat =  c(11, 12)))

# gui
ui <- fluidPage(table_ui("table"))

# server
server <-function(input, output, session) {
 # table
 output_table <- table_server(
  "table",
  reactive(points),
  reactiveValues(parm = "temperature", spat = 1, temp = "annual")
  )
}

# run app
shinyApp(ui, server)

}
```

Index