Retrieving and visualizing satellite sea water temperature data for marine analyses

Virginia A. García Alonso, Pablo S. Milla Carmona

A case study using the *rerddap* R package

Motivation

Environmental variables are key determinants of many biological processes in marine ecosystems. Temperature variability is especially important in high latitudes where organisms are subject to marked seasonal variations which influence their life cycles. Since collection of *in situ* data entails different logistic challenges and may provide inadequate spatiotemporal resolutions, **satellite data emerges as a powerful tool to boost marine analyses**.

What is this poster about?

A **workflow** including the necessary steps to i) retrieve **satellite sea surface temperature (sst) data** from the ERDDAP server using the *rerddap* package [@R-rerddap] and ii) **visualize** the results.

Where and when?

The chosen study area is located in the southern border of the Southwest Atlantic Ocean, a region displaying both a **marked seasonality** and a **longitudinal gradient** in water temperature across seasons. We will visualize sst data from austral spring and autumn of 2015-2016.

Which data set?

The 'jplMURSST41' gridded data set, a **Multiscale Ultrahigh Resolution (MUR) L4 analysis** of sst. It includes a **global 0.01 degree grid** with interpolated sst expressed as Celsius degrees (ºC). More information can be found at the podaac dataset website.

To visualize other ERDDAP data sets check this page.

Steps

1. Load needed packages

Besides the *rerdapp* package, we will need the following packages to manipulate data and visualize it nicely :)

- *tidyverse* (Wickham 2019)
- *sf* (Pebesma 2022)
- *ggspatial* (Dunnington 2021)
- *rnaturalearth* (South 2017a)
- *rnaturalearthdata* (South 2017b)
- *rnaturalearthhires* (South 2021)
- *marmap* (Pante, Simon-Bouhet, and Irisson 2020)

2. Download sst data

We will retrieve sst data with the `info()` function and choose the desired data according to the spatiotemporal resolution needed with the `griddap()` function.

```
#retrieve the sst data set
sstInfo <- info('jplMURSST41')

#choose the spatiotemporal resolution
sst_spring <-

  griddap(sstInfo,
          latitude = c(-56, -52),
          longitude = c(-70, -55),
          time = c('2015-09-21',
                   '2015-12-21'),
          fields = 'analysed_sst',
          fmt ="csv")
```

Downloading this data takes a while... you can save it as a `.csv` file to use it in the future with the following command:

```
write.csv(sst_spring,"sst_spring.csv",row.names=FALSE)
```

3. Read and organize data

We have multiple sst for each latitude/longitude pair (one per day), so we will estimate **mean sst**.

```
#load the data as following
sst_spring <-

  read.csv(file = "sst_spring.csv",
           header = TRUE)

#transform and organize the data
mean_sst_spring <-

  #transform to a data frame
  as.data.frame(sst_spring) %>%

  #select the needed variables
  dplyr::select(longitude, latitude,
                analysed_sst) %>%

  #estimate mean sst for each lat/long
  group_by(longitude,latitude) %>%
  summarise(mean_sst =
              mean(analysed_sst)) %>%
  ungroup() %>%

  #set a new variable to identify
  #the season
  mutate(season="Spring")
```

We can repeat the same procedure for **autumn** creating a new object named `mean_sst_autumn` and finally join both datasets in a single object.

```
#join data from both seasons
mean_sst <- bind_rows(mean_sst_spring,
                      mean_sst_autumn)
```

Finally, the plot!

We will plot sst data with *ggplot2*, employing the `geom_raster()` function.

```
ggplot()+

  #add sst as a raster and interpolate
  geom_raster(data = mean_sst,
              aes(x = longitude,
                  y = latitude,
                  fill = mean_sst),
                  interpolate = T)+

  #set the fill
  scale_fill_viridis_c(alpha = 0.85)+

  #separate plots for each season
  facet_grid(season~.)
```

A few more tweaks allow including the continental territory and relevant bathymetric contours to the map and getting this!

Thank you very much for the attention!

Hope you find it useful :)

Materials employed for the poster and a spanish version are openly shared in this GitHub repository

References

Chamberlain, Scott. 2021. Rerddap: General Purpose Client for ERDDAP Servers. https://CRAN.R-project.org/package=rerddap. Dunnington, Dewey. 2021. Ggspatial: Spatial Data Framework for Ggplot2. https://CRAN.R-project.org/package=ggspatial. Pante, Eric, Benoit Simon-Bouhet, and Jean-Olivier Irisson. 2020. Marmap: Import, Plot and Analyze Bathymetric and Topographic Data. https://github.com/ericpante/marmap. Pebesma, Edzer. 2022. Sf: Simple Features for r. https://CRAN.R-project.org/package=sf. South, Andy. 2017a. Rnaturalearth: World Map Data from Natural Earth. https://github.com/ropenscilabs/rnaturalearth. ———. 2017b. Rnaturalearthdata: World Vector Map Data from Natural Earth Used in Rnaturalearth. https://github.com/ropenscilabs/rnaturalearthdata. ———. 2021. Rnaturalearthhires: High Resolution World Vector Map Data from Natural Earth

Used in Rnaturalearth. Wickham, Hadley. 2019. Tidyverse: Easily Install and Load the Tidyverse. https://CRAN.R-project.org/package=tidyverse.