Matching benthic abundance product to benthic habitat data

Tom Webb

15/03/2021

Load packages

First, load required packages:

```
library(tidyverse)
library(worrms)
library(raster)
library(sf)
library(mapview)
library(viridis)
library(concaveman)
```

Processing the EMODnet Benthic Numerical Abundance product

This code reads in, processes, and exports the EMODnet benthic numerical abundance product, for use in the benthic habitat matching product. The numerical abundance product is described here https://mda.vliz.be/directlink.php?fid=VLIZ_00000727_60196601e2e81. Download the data using its direct download URL, add to the raw data folder, unzip, and read in to R. NB - this is quite a large download (~420MB zipped). The zipped file is also removed.

```
download.file(
  url = "https://mda.vliz.be/download.php?file=VLIZ_00000727_60196601e2e81",
              destfile = here::here(
                "data", "raw_data/EMODnet-Biology-Benthos_numeric_abundance.zip"))
unzip(here::here("data", "raw_data/EMODnet-Biology-Benthos_numeric_abundance.zip"),
      exdir = here::here("data", "raw_data"))
unzip(here::here("data", "raw_data/EMODnet-Biology-Benthos_numeric_abundance.zip"),
      exdir = here::here("data", "raw_data"))
invisible(file.remove(
  here::here("data", "raw_data/EMODnet-Biology-Benthos_numeric_abundance.zip")))

benth_events <- read_csv(
  here::here("data",
             "raw_data/EMODnet-Biology-Benthos_numeric_abundance/product/maps/spesh.csv"))
```

This is a very big, wide df, with a row for each unique sampling event and a column for each species. Here we turn this into a long dataset (with a 'species' column - actually WoRMS AphiaID, stripping zeros which = absence), and a separate dataset containing only information about the individual sampling events (every unique location in space and time). This takes a while and needs to be done in chunks or we hit memory problems:

```
benth_abundances_1 <- benth_events %>% dplyr::select(-c(datasetid:sampid)) %>%
  slice(1:10000) %>%
  mutate(across(starts_with("ab_")), na_if(., 0)) %>%
  pivot_longer(cols = names(benth_events)[str_detect(names(benth_events), "ab_")],
               names_to = "AphiaID", names_prefix = "ab_",
               values_to = "abundance", values_drop_na = TRUE)
benth_abundances_2 <- benth_events %>% dplyr::select(-c(datasetid:sampid)) %>%
  slice(10001:30000) %>%
  mutate(across(starts_with("ab_")), na_if(., 0)) %>%
  pivot_longer(cols = names(benth_events)[str_detect(names(benth_events), "ab_")],
               names_to = "AphiaID", names_prefix = "ab_",
               values_to = "abundance", values_drop_na = TRUE)
benth_abundances_3 <- benth_events %>% dplyr::select(-c(datasetid:sampid)) %>%
  slice(30001:50000) %>%
  mutate(across(starts_with("ab_")), na_if(., 0)) %>%
  pivot_longer(cols = names(benth_events)[str_detect(names(benth_events), "ab_")],
               names_to = "AphiaID", names_prefix = "ab_",
               values_to = "abundance", values_drop_na = TRUE)
benth_abundances_4 <- benth_events %>% dplyr::select(-c(datasetid:sampid)) %>%
  slice(50001:72092) %>%
  mutate(across(starts_with("ab_")), na_if(., 0)) %>%
  pivot_longer(cols = names(benth_events)[str_detect(names(benth_events), "ab_")],
               names_to = "AphiaID", names_prefix = "ab_",
               values_to = "abundance", values_drop_na = TRUE)

# compile into a single dataset of abundances
benth_abundances <- bind_rows(benth_abundances_1, benth_abundances_2,
                              benth_abundances_3, benth_abundances_4)
# remove the intermediate datasets
rm(benth_abundances_1, benth_abundances_2,
   benth_abundances_3, benth_abundances_4)

# create the sampling events dataset with no species info
benth_events <- benth_events %>% dplyr::select(datasetid:eventNummer)
```

This all results in a 3 column, ~1.2M row `benth_abundances` dataset which includes the abundance of each species (referenced by its AphiaID) in each of the sampling events (eventNummer) in which it occurs; and ~72K row dataset, `benth_events`, which contains a row for each unique sampling event, giving dataset ID, date, longitude, latitude, depth, sample ID, and event number (which links to eventNummer in `benth_abundances`). The other dataset to create is taxonomic classification for the species now identified in `benth_abundances` by AphiaID only. First, get a list of distinct Aphia IDs:

```
benth_taxa <- benth_abundances %>%
  dplyr::select(AphiaID) %>%
  distinct()
```

Now, use the `worrms` package to get a full classification for each of these IDs (NB this takes several minutes to run over all ~4200 IDs), and do a little tidying of the output - here keeping only major taxonomic groupings:

```
benth_classification <- wm_classification_(id = pull(benth_taxa, AphiaID)) %>%
  dplyr::select(-AphiaID) %>%
  pivot_wider(names_from = rank, values_from = scientificname) %>%
  unnest(benth_classification,
         cols = c(Kingdom, Phylum, Subphylum, Superclass, Class, Subclass,
                  Superorder, Order, Suborder, Infraorder, Parvorder,
```

```
                  Superfamily, Family, Genus, Species, Infraclass, Subfamily,
                  Tribe, Subterclass, Subspecies, Subgenus, Section, Subsection,
                  Subtribe, Epifamily)) %>%
  dplyr::select(id, Species, Phylum, Class, Order, Family, Genus) %>%
  distinct(id, .keep_all = TRUE) %>%
  raname(AphiaID = id)
```

Reading in sediment property data

The first dataset we use is a gridded sediment property map at 0.125degree resolution for NW Europe from Wilson et al. (2018) A synthetic map of the north-west European Shelf sedimentary environment for applications in marine science Earth Syst. Sci. Data, 10, 109–130 https://doi.org/10.5194/essd-10-109-2018, with the data product available from https://doi.org/10.15129/1e27b806-1eae-494d-83b5-a5f4792c46fc which we download directly into the raw data folder, unzip (then remove the zipped version), and read in. Final step here is to create a new variable, log_D50, which is the log (base 10) of the median sediment grain size in each grid cell.

```
curl::curl_download(
  url = "https://pureportal.strath.ac.uk/files/69962174/data_files_csv.zip",
  destfile = "data/raw_data/strath_synthetic_nw_european_shelf_sed_map.zip")

# unzip
unzip("data/raw_data/strath_synthetic_nw_european_shelf_sed_map.zip", exdir = "data/raw_data/strath_sedm
# remove zip file
invisible(file.remove(
  here::here("data", "raw_data/strath_synthetic_nw_european_shelf_sed_map.zip")))

# read in sediment properties data and descriptors
sed_props <- read_csv(
  "data/raw_data/strath_sedmap/sediment_properties.csv")
sed_props_descriptors <- read_csv(
  "data/raw_data/strath_sedmap/sediment_properties_descriptors.csv")

# add a logged total D50
sed_props <- sed_props %>% mutate(log_D50 = log10(TotalD50))
```

This is a gridded data product, so we create a raster from it, and plot it for info:

```
sed_r <- rasterFromXYZ(dplyr::select(
  sed_props, Longitude, Latitude, MudPercent:Rock50cm, log_D50),
                         res = 0.125)
plot(sed_r, col = inferno(50))
```

Matching benthic sampling events to the sediment properties map

To match the benthic sampling events to the sediment properties data, we need to make `benth_events` spatial, setting the projection to WGS84 lon/lat (EPSG 4326), and setting `remove = FALSE` to retain the lon and lat coordinates in the data frame.

```
benth_events <- st_as_sf(benth_events,
                         coords = c("decimallongitude", "decimallatitude"),
                         crs = 4326, remove = FALSE)
```

To produce a quick plot of the benthic sampling events:

```
mapview(benth_events)
```

And of a layer of the sediment raster:

```
crs(sed_r) <- crs(benth_events)
mapview(sed_r[[1]])
```

So most of the sampling events should get matched to the sediment data, apart from Baltic Sea and more northerly ones. Do the matching:

```
sed_by_event <- raster::extract(sed_r,
                                cbind(benth_events$decimallongitude,
                                      benth_events$decimallatitude)) %>%
  as_tibble()
```

And bind back to benth_events:

```
benth_events <- benth_events %>% bind_cols(sed_by_event)
```

Check for total number of successful matches:

```
colSums(!is.na(sed_by_event))
```

This shows that around 40% of events matched to sediment values. Add these sediment values back to `benth_events`:

```
benth_events <- benth_events %>% bind_cols(sed_by_event)
```

If you want to check where the unmatched events are:

```
mapview(filter(benth_events, is.na(MudPercent)))
```

This shows they are either northern (i.e. Norwegian seas - as expected, as this region is not covered in the sediment dataset) or very close to shore (e.g. coastal, intertidal, estuarine) and not well covered by the sediment dataset.

Process EMODnet Seabed Habitat map

First, download the EMODnet broadscale habitat map - NB big download - 589MB zipped. Unzip and remove the zipped version.

```
download.file(
  url = "https://www.emodnet-seabedhabitats.eu/files/eusm2019_model_and_confidence.zip",
  destfile = here::here(
    "data", "raw_data/eusm2019_model_and_confidence.zip"))
unzip(here::here("data", "raw_data/eusm2019_model_and_confidence.zip"),
      exdir = here::here("data", "raw_data/eusm2019_model_and_confidence"))
# remove the zipped file
invisible(file.remove(here::here(
  "data", "raw_data/eusm2019_model_and_confidence.zip")))
```

To examine the layers:

```
st_layers(
  here::here("data",
             "raw_data/eusm2019_model_and_confidence/EUSM2019_EUNIS_BroadscaleModel.gdb"))
```

The ones of interest that match our benthic abundance product are EUSM_Arctic_Atlantic, EUSM_BalticSea, and EUSM_Macronesia_BayofBiscay. The following processes read in the required layers in turn, ensure

the geometries are valid, and crop them to the overall extent of the benthic dataset. This is quite a lengthy process as the layers are large.

First job is to create an appropriate boundary around the events in `benth_events` to set the region of interest for the habitat maps. We can do this using `concaveman`, then adding a 50km buffer, after transforming `benth_events` to the CRS used in the habitat map (EPSG = 3857):

```
benth_events <- benth_events %>% st_transform(crs = 3857)
benth_mch <- benth_events %>%
  concaveman(concavity = 1000) %>%
  st_buffer(dist = 50000)
```

Now read in and process the habitat maps:

```
arc_atl_substrate <- st_read(
  here::here("data",
             "raw_data/eusm2019_model_and_confidence/EUSM2019_EUNIS_BroadscaleModel.gdb"),
  layer = "EUSM_Arctic_Atlantic") %>%
  st_make_valid() %>%
  st_crop(y = benth_mch)

biscay_substrate <- st_read(
  here::here("data",
             "raw_data/eusm2019_model_and_confidence/EUSM2019_EUNIS_BroadscaleModel.gdb"),
  layer = "EUSM_Macronesia_BayofBiscay") %>%
  st_make_valid() %>%
  st_crop(y = benth_mch)

baltic_substrate <- st_read(
  here::here("data",
             "raw_data/eusm2019_model_and_confidence/EUSM2019_EUNIS_BroadscaleModel.gdb"),
  layer = "EUSM_BalticSea") %>%
  st_make_valid() %>%
  st_crop(y = benth_mch)
```

Stitch them together, select most relevant variables, and remove the individual datasets:

```
emodnet_habs <- bind_rows(arc_atl_substrate, baltic_substrate)
emodnet_habs <- bind_rows(emodnet_habs, biscay_substrate)
emodnet_habs <- emodnet_habs %>%
  dplyr::select(Energy, Biozone, Substrate, Salinity, Oxygen,
                EUNIScomb, EUNIScombD, Allcomb, AllcombD,
                SalcombD, MSFD_BBHT)
rm(arc_atl_substrate, biscay_substrate, baltic_substrate)
```

Matching benthic sampling events to the EMODnet broadscale habitats map

This is a simple spatial join to get a value for each variable in `emodnet_habs` for each point in `benth_events` (NB this introduces some duplicate rows, which are removed using `distinct`):

```
benth_events <- benth_events %>% st_join(emodnet_habs)
```

Check how this has worked - e.g. looking at the Substrate variable:

```
benth_events %>% st_drop_geometry() %>% count(Substrate)
```

Check which events have not been matched to the habitat map:

```
mapview(filter(benth_events, is.na(Substrate)))
```

These are all coastal, likely infralittoral, while the habitat map is only sublittoral.

Note that some duplicates have been introduced in the `st_join` - these need to be removed (we can also remove the geometry column too at this stage):

```
benth_events <- benth_events %>% st_drop_geometry() %>% distinct()
```

Note that some events now have more than one entry (typically two):

```
benth_events %>% count(eventNummer) %>% arrange(desc(n))
```

This is because they match more than one classification in the EMODnet habitat map, e.g.

```
benth_events %>% filter(eventNummer == 2656) %>% dplyr::select(Substrate, EUNIScombD)
```

There is not an obvious way of choosing between the two, so we leave in both classifications for these events. Note it only concerns a tiny fraction of all events:

```
benth_events %>% count(eventNummer) %>%
  rename(n_instances = n) %>% count(n_instances, name = "n_events")
```

We now have fully matched the benthic sampling events to two benthic habitat data products, and have created a condensed version of the benthic abundances data, and an additional table of taxonomic classifications. These three derived datasets can now be written to file, for use in generating further products.

```
write_csv(benth_events,
          here::here(
            "data", "derived_data/benthic_abundance_sampling_events_seabed_habs.csv"))
write_csv(benth_abundances,
          here::here("data", "derived_data/benthic_abundances_long.csv"))
write_csv(benth_classification,
          here::here("data", "derived_data/benthic_taxa.csv"))
```

Reproducibility

Reproducibility receipt

```
## datetime
Sys.time()
```

```
## [1] "2021-04-01 14:02:11 BST"
```

```
## repository
git2r::repository()
```

```
## Local:    master /Users/tom/Google Drive/emodnet habitats/EMODnet_occs_habs
## Remote:   master @ origin (https://github.com/EMODnet/EMODnet-Biology-Benthic-Habitats-Occurrences-T
## Head:     [f9dd58f] 2021-04-01: Trying to deal with conflicts with rendered documents
```

```
## session info
sessionInfo()
```

```
## R version 3.6.2 (2019-12-12)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Catalina 10.15.7
##
## Matrix products: default
```

```
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] concaveman_1.1.0  viridis_0.5.1     viridisLite_0.3.0 mapview_2.7.7
##  [5] sf_0.9-7          raster_3.4-5      sp_1.4-5          worrms_0.4.0
##  [9] forcats_0.4.0     stringr_1.4.0     dplyr_1.0.4       purrr_0.3.4
## [13] readr_1.3.1       tidyr_1.0.0       tibble_3.0.6      ggplot2_3.3.3
## [17] tidyverse_1.3.0
##
## loaded via a namespace (and not attached):
##  [1] fs_1.3.1           satellite_1.0.2    lubridate_1.7.4    webshot_0.5.2
##  [5] httr_1.4.2         tools_3.6.2        backports_1.1.5    utf8_1.1.4
##  [9] R6_2.5.0           KernSmooth_2.23-16 DBI_1.1.1          colorspace_1.4-1
## [13] withr_2.1.2        tidyselect_1.1.0   gridExtra_2.3      leaflet_2.0.3
## [17] git2r_0.26.1       compiler_3.6.2     leafem_0.1.0       cli_2.3.1
## [21] rvest_0.3.5        xml2_1.3.2         scales_1.1.0       classInt_0.4-3
## [25] digest_0.6.27      rmarkdown_2.7      base64enc_0.1-3    pkgconfig_2.0.3
## [29] htmltools_0.5.1.1  dbplyr_1.4.2       fastmap_1.0.1      htmlwidgets_1.5.1
## [33] rlang_0.4.10       readxl_1.3.1       rstudioapi_0.13    shiny_1.4.0
## [37] generics_0.1.0     jsonlite_1.7.2     crosstalk_1.0.0    magrittr_2.0.1
## [41] Rcpp_1.0.6         munsell_0.5.0      fansi_0.4.2        lifecycle_1.0.0
## [45] stringi_1.5.3      yaml_2.2.1         grid_3.6.2         promises_1.1.0
## [49] crayon_1.4.1       lattice_0.20-38    haven_2.2.0        hms_0.5.3
## [53] knitr_1.31         pillar_1.5.0       codetools_0.2-16   stats4_3.6.2
## [57] reprex_0.3.0       glue_1.4.2         evaluate_0.14      modelr_0.1.5
## [61] vctrs_0.3.6        png_0.1-7          httpuv_1.5.2       cellranger_1.1.0
## [65] gtable_0.3.0       assertthat_0.2.1   xfun_0.21          mime_0.10
## [69] xtable_1.8-4       broom_0.7.2        e1071_1.7-4        later_1.0.0
## [73] class_7.3-15       units_0.6-7        ellipsis_0.3.1
```