

# FR-CGFS: French English Channel Bottom Trawl survey data processing summary

fishglob, Aurore A. Maureaud & Juliano Palacios Abrantes & P.D van Denderen

January, 2023

## Contents

General info . . . . .	1
Data cleaning in R . . . . .	1
1. Overview of the survey data table . . . . .	23
2. Summary of sampling intensity . . . . .	24
3. Summary of sampling variables from the survey . . . . .	25
4. Summary of biological variables . . . . .	26
5. Extreme values . . . . .	27
6. Summary of variables against swept area . . . . .	28
7. Abundance or Weight trends of the six most abundant species . . . . .	29
8. Distribution mapping . . . . .	30
9. Taxonomic flagging . . . . .	31
10. Spatio-temporal standardization . . . . .	32
a. Standardization method 1 . . . . .	32
b. Standardization method 2 . . . . .	35
c. Standardization summary . . . . .	36

## General info

This document presents the cleaning code and summary of the French English Channel bottom trawl survey provided by DATRAS, ICES. It contains data from **1998** and up to **2020**.

## Data cleaning in R

```
#####
#### R code to download and clean DATRAS data from ICES
#### URL: https://datras.ices.dk/
#### Coding: Aurore Maureaud + Juliano Palacios + Daniel van Denderen, December 2022
#####
rm(list=ls())

date <- "27DEC2022"

#####
#### LOAD LIBRARIES & options to decide in the code
#####
library(data.table)
library(tidyverse)
library(icesDatras)
```

```

library(worrms)
library(curl)
library(urltools)
library(here) # for easy work around on multiple computers
library(taxize) # for getting correct species names
library(googledrive)
library(readxl)

# load relevant functions
source("functions/write_clean_data.r")
source("functions/clean_taxa.R")
source("functions/write_clean_data.R")
source("functions/apply_trimming_method1.R")
source("functions/apply_trimming_method2.R")
source("functions/flag_spp.R")
fishglob_data_columns <- read_excel("standard_formats/fishglob_data_columns.xlsx")

# should the last version of DATRAS be downloaded?
download_last_version <- FALSE

# should the data be loaded from the last version saved?
load_stored_datras <- FALSE

# should we save a new version of hh and hl?
save_hh_and_nl <- FALSE

# should we get the length-weight relationships from fishbase?
need_get_lw_rel <- FALSE

#####
##### LOAD FILES #####
#####

if(download_last_version == TRUE){
  last.year <- 2020

  # Haul info from Datras
  hh.ns <- getDATRAS(record='HH', survey='NS-IBTS', years=c(1967:last.year),
    quarters=c(1,3))
  hh.baltic <- getDATRAS(record='HH', survey='BITS', years=c(1991:last.year),
    quarters=c(1,4))
  hh.evhoe <- getDATRAS(record='HH', survey='EVHOE', years=c(1997:last.year),
    quarters=4)
  hh.cgfs <- getDATRAS(record='HH', survey='FR-CGFS', years=c(1998:last.year),
    quarters=4)
  hh.igfs <- getDATRAS(record='HH', survey='IE-IGFS', years=c(2003:last.year),
    quarters=4)
  hh.nigfs <- getDATRAS(record='HH', survey='NIGFS', years=c(2005:last.year),
    quarters=c(1:4))
  hh.pt <- getDATRAS(record='HH', survey='PT-IBTS', years=c(2002:last.year),
    quarters=c(3:4))
  hh.rock <- getDATRAS(record='HH', survey='ROCKALL', years=c(1999:2009),

```

```

            quarters=3)
hh.scorock <- getDATRAS(record='HH', survey='SCOROC', years=c(2011:last.year),
                        quarters=3)
hh.swc <- getDATRAS(record='HH', survey='SWC-IBTS', years=c(1985:2010),
                     quarters=c(1:4))
hh.scowcgfs <- getDATRAS(record='HH', survey='SCOWCGFS', years=c(2011:last.year),
                           quarters=c(1:4))

write.csv(hh.ns, file = "E:/fishglob data/Publicly available/DATRAS/hh.ns.csv",
          row.names = F)
write.csv(hh.baltic, file = "E:/fishglob data/Publicly available/DATRAS/hh.baltic.csv",
          row.names = F)
write.csv(hh.evhoe, file = "E:/fishglob data/Publicly available/DATRAS/hh.evhoe.csv",
          row.names = F)
write.csv(hh.cgfs, file = "E:/fishglob data/Publicly available/DATRAS/hh.cgfs.csv",
          row.names = F)
write.csv(hh.igfs, file = "E:/fishglob data/Publicly available/DATRAS/hh.igfs.csv",
          row.names = F)
write.csv(hh.nigfs, file = "E:/fishglob data/Publicly available/DATRAS/hh.nigfs.csv",
          row.names = F)
write.csv(hh.pt, file = "E:/fishglob data/Publicly available/DATRAS/hh.pt.csv",
          row.names = F)
write.csv(hh.rock, file = "E:/fishglob data/Publicly available/DATRAS/hh.rock.csv",
          row.names = F)
write.csv(hh.scorock, file = "E:/fishglob data/Publicly available/DATRAS/hh.scorock.csv",
          row.names = F)
write.csv(hh.swc, file = "E:/fishglob data/Publicly available/DATRAS/hh.swc.csv",
          row.names = F)
write.csv(hh.scowcgfs, file = "E:/fishglob data/Publicly available/DATRAS/
                           hh.scowcgfs.csv", row.names = F)

# Length info from DATRAS
hl.ns <- getDATRAS(record='HL', survey='NS-IBTS', years=c(1967:last.year),
                     quarters=c(1,3))
hl.baltic <- getDATRAS(record='HL', survey='BITS', years=c(1991:last.year),
                        quarters=c(1,4))
hl.evhoe <- getDATRAS(record='HL', survey='EVHOE', years=c(1997:last.year),
                       quarters=4)
hl.cgfs <- getDATRAS(record='HL', survey='FR-CGFS', years=c(1998:last.year),
                      quarters=4)
hl.igfs <- getDATRAS(record='HL', survey='IE-IGFS', years=c(2003:last.year),
                      quarters=4)
hl.nigfs <- getDATRAS(record='HL', survey='NIGFS', years=c(2005:last.year),
                       quarters=c(1:4))
hl.pt <- getDATRAS(record='HL', survey='PT-IBTS', years=c(2002:last.year),
                    quarters=c(3:4))
hl.rock <- getDATRAS(record='HL', survey='ROCKALL', years=c(1999:2009),
                      quarters=3)
hl.scorock <- getDATRAS(record='HL', survey='SCOROC', years=c(2011:last.year),
                         quarters=3)
hl.swc <- getDATRAS(record='HL', survey='SWC-IBTS', years=c(1985:2010),
                     quarters=c(1:4))

```

```

hl.scowcgfs <- getDATRAS(record='HL', survey='SCOWCGFS', years=c(2011:last.year),
                           quarters=c(1:4))

write.csv(hl.ns, file = "E:/fishglob data/Publicly available/DATRAS/hl.ns.csv",
          row.names = F)
write.csv(hl.baltic, file = "E:/fishglob data/Publicly available/DATRAS/hl.baltic.csv",
           row.names = F)
write.csv(hl.evhoe, file = "E:/fishglob data/Publicly available/DATRAS/hl.evhoe.csv",
           row.names = F)
write.csv(hl.cgfs, file = "E:/fishglob data/Publicly available/DATRAS/hl.cgfs.csv",
           row.names = F)
write.csv(hl.igfs, file = "E:/fishglob data/Publicly available/DATRAS/hl.igfs.csv",
           row.names = F)
write.csv(hl.nigfs, file = "E:/fishglob data/Publicly available/DATRAS/hl.nigfs.csv",
           row.names = F)
#write.csv(hl.pt, file = "E:/fishglob data/Publicly available/DATRAS/hl.pt.csv",
#           row.names = F)
write.csv(hl.rock, file = "E:/fishglob data/Publicly available/DATRAS/hl.rock.csv",
           row.names = F)
write.csv(hl.scorock, file = "E:/fishglob data/Publicly available/DATRAS/hl.scorock.csv",
           row.names = F)
#write.csv(hl.swc, file = "E:/fishglob data/Publicly available/DATRAS/hl.swc.csv",
#           row.names = F)
write.csv(hl.scowcgfs, file = "E:/fishglob data/Publicly available/DATRAS/hl.scowcgfs.csv",
           row.names = F)

}

if(load_stored_datras == TRUE){

  hh.ns <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.ns.csv")
  hh.baltic <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.baltic.csv")
  hh.evhoe <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.evhoe.csv")
  hh.cgfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.cgfs.csv")
  hh.igfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.igfs.csv")
  hh.nigfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.nigfs.csv")
  hh.pt <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.pt.csv")
  hh.rock <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.rock.csv")
  hh.scorock <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.scorock.csv")
  hh.swc <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.swc.csv")
  hh.scowcgfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.scowcgfs.csv")

  hl.ns <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.ns.csv")
  hl.baltic <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.baltic.csv")
  hl.evhoe <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.evhoe.csv")
  hl.cgfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.cgfs.csv")
  hl.igfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.igfs.csv")
  hl.nigfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.nigfs.csv")
  hl.pt <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.pt.csv") %>%
    dplyr::rename(Valid_Aphia = ValidAphiaID) %>%
    select(RecordType, Survey, Quarter, Country, Ship, Gear, SweepLngt, GearEx,
           DoorType, StNo, HaulNo, Year, SpecCodeType, SpecCode, SpecVal, Sex,
           TotalNo, CatIdentifier, NoMeas, SubFactor, SubWgt, CatCatchWgt, LngtCode,

```

```

LngtClass, HLNoAtLngt, DevStage, LenMeasType, DateofCalculation,
Valid_Aphia)
hl.rock <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.rock.csv")
hl.scorock <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.scorock.csv")
hl.swc <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.swc.csv")%>%
  dplyr::rename(Valid_Aphia = ValidAphiaID) %>%
  select(RecordType, Survey, Quarter, Country, Ship, SweepLngt, GearEx,
         DoorType, StNo, HaulNo, Year, SpecCodeType, SpecCode, SpecVal, Sex,
         TotalNo, CatIdentifier, NoMeas, SubFactor, SubWgt, CatCatchWgt, LngtCode,
         LngtClass, HLNoAtLngt, DevStage, LenMeasType, DateofCalculation,
         Valid_Aphia)
hl.scowcgfs <- read.csv("E:/fishglob data/Publicly available/DATRAS/hl.scowcgfs.csv")

hh <- rbind(hh.ns, hh.baltic, hh.evhoe, hh.cgfs, hh.igfs, hh.nigfs, hh.pt, hh.rock,
            hh.scorock, hh.swc, hh.scowcgfs)

hl <- rbind(hl.ns, hl.baltic, hl.evhoe, hl.cgfs, hl.igfs, hl.nigfs, hl.pt, hl.rock,
            hl.scorock, hl.swc, hl.scowcgfs)

rm(hl.ns, hl.baltic, hl.evhoe, hl.cgfs, hl.igfs, hl.nigfs, hl.pt, hl.rock, hl.scorock,
    hl.swc, hl.scowcgfs,
    hh.ns, hh.baltic, hh.evhoe, hh.cgfs, hh.igfs, hh.nigfs, hh.pt, hh.rock, hh.scorock,
    hh.swc, hh.scowcgfs)
#rm(ca.ns, ca.baltic, ca.evhoe, ca.cgfs, ca.igfs, ca.nigfs, ca.pt, ca.rock,
# ca.scorock, ca.swc, ca.scowcgfs)

}

if(save_hh_and_hl == TRUE){
  save(hh, file = paste0("E:/fishglob data/Publicly available/DATRAS/hh.",date,".RData"))
  save(hl, file = paste0("E:/fishglob data/Publicly available/DATRAS/hl.",date,".RData"))
}

#####
# Alternative
#####

# Juliano
hl <- fread("/Volumes/Enterprise/Data/FishGlob/Data/Raw/ices_hl.csv")
unique(hl$Survey)
hh <- fread("/Volumes/Enterprise/Data/FishGlob/Data/Raw/ices_hh.csv")
unique(hh$Survey)

# Aurore
load("/Volumes/Elements/fishglob data/Publicly available/DATRAS/hl.29JUL2021.RData")
load("/Volumes/Elements/fishglob data/Publicly available/DATRAS/hh.29JUL2021.RData")

#####
##### CREATE A UNIQUE HAUL ID #####
#####

hl$HaulID <- paste(hl$Survey, hl$Year, hl$Quarter, hl$Country, hl$Ship, hl$Gear, hl$StNo,

```

```

            hl$HaulNo)
hh$HaulID <- paste(hh$Survey, hh$Year, hh$Quarter, hh$Country, hh$Ship, hh$Gear, hh$StNo,
                     hh$HaulNo)

# Is the HaulID unique?
hhn <- unique(hh$HaulID)
length(hhn)==nrow(hh)

# check which one is not
pb <- c()
for (i in 1:length(hhn)){
  j <- which(hh$HaulID==hhn[i])
  if(length(j)>1){pb <- hhn[i]}
}

# problem with one haul in NS-IBTS
hh <- hh %>%
  filter(HaulID!=pb)
hl <- hl %>%
  filter(HaulID!=pb)

# Only keep hauls where there is the length composition.
# 70273 hauls in hh and 70273 in hl
hh <- subset(hh, hh$HaulID %in% hl$HaulID)
hl <- subset(hl, hl$HaulID %in% hh$HaulID)

#####
##### MERGE HH and HL FILES
#####

haulidhl <- sort(unique(hl$HaulID))
haulidhh <- sort(unique(hh$HaulID))
identical(haulidhh, haulidhl)
rm(haulidhh, haulidhl)

# remove some columns in hl
hl$SweepLngt <- hl$SpecCodeType <- hl$SpecCode <- hl$Sex <- hl>DateofCalculation <- NULL
hl$RecordType <- hl$GearEx <- NULL

# remove some columns in hh
hh>DateofCalculation <- hh$ThClineDepth <- hh$ThermoCline <- hh$SwellHeight <- NULL
hh$SwellDir <- hh$WindSpeed <- hh$WindDir <- hh$BotCurSpeed <- NULL
hh$BotCurDir <- hh$SurCurSpeed <- hh$SurCurDir <- hh$SpeedWatter <- hh$TowDir <- NULL
hh$WgtGroundRope <- hh$KiteDim <- hh$Buoyancy <- hh$Tickler <- NULL
hh$DoorWgt <- hh$DoorSurface <- hh$WarpDen <- hh$Warpdia <- hh$Warplngt <- NULL
hh$Rigging <- hh$HydroStNo <- hh$HaulLat <- hh$HaulLong <- hh$DayNight <- NULL
hh$Stratum <- hh$TimeShot <- hh$Day <- hh$RecordType <- hh$GearExp <- hh$DoorType <- NULL

#survey <- merge(hh, hl, by='HaulID', all.x=FALSE, all.y=TRUE)
survey <- right_join(hh, hl, by=c('HaulID', 'Survey', 'Quarter', 'Country', 'Ship',
                                    'Gear', 'StNo', 'HaulNo', 'Year'))
nrow(survey)==nrow(hl)

```

```

survey <- survey %>%
  dplyr::rename(SBT = BotTemp,
    SST = SurTemp,
    Speed = GroundSpeed,
    AphiaID = Valid_Aphia)

### Check if the HaulID is unique
### Not the case for the baltic sea, a lot of duplicates!!!
#ids <- unique(hh$HaulID)
#pb <- vector()
#for(i in 1:length(ids)){
#  x <- which(hh$HaulID==ids[i])
#  if(length(x)>1){pb[length(pb)+1] <- ids[i]}
#}
#print(pb) # dim 0 ok!

#####
##### REMOVE INVALID DATA #####
#####
survey <- survey %>%
  filter(HaulVal %in% 'V', #Remove invalid hauls
    !is.na(AphiaID), # Remove invalid species records
    SpecVal %in% c(1,10,4,7),
    DataType %in% c('S','R','C'))

#####
##### RESCALE DATA INTO ABUNDANCE FOR THE HAUL DURATION AND ABUNDANCE AT LENGTH #####
#####
# If Data Type=='C', abundance at length already readjusted with time so get back the
# abundance for the actual duration of the haul.
# If data type=='R', abundance at length is multiplied by subfactor and adjusted to time
survey$CatCatchWgt = as.numeric(survey$CatCatchWgt)

survey <- survey %>%
  mutate(HLNoAtLngt = case_when(DataType=='C' ~ HLNoAtLngt*SubFactor*HaulDur/60,
    DataType %in% c('S','R') ~ HLNoAtLngt*SubFactor),
  TotalNo = case_when(DataType=='C' ~ TotalNo*HaulDur/60,
    DataType %in% c('S','R') ~ TotalNo),
  CatCatchWgt = case_when(DataType=='C' ~ CatCatchWgt*HaulDur/60,
    DataType %in% c('S','R') ~ CatCatchWgt)) %>%
  select(-HaulVal, -DataType, -StdSpecRecCode, -SpecVal, -SubWgt, -SubFactor) %>%
  mutate(Survey = if_else(Survey=='SCOWCGFS', 'SWC-IBTS', Survey)) %>%
  mutate(Survey = if_else(Survey=='SCOROC', 'ROCKALL', Survey)) %>%
  filter(!(Survey=="NS-IBTS" & BySpecRecCode %in% c(0,2,3,4,5)),
    # remove hauls where not all species are recorded
    !(Survey=="BITS" & BySpecRecCode==0))

#####
##### GET THE SWEPT AREA in km2 #####
#####

```

```

#####
source('cleaning_codes/source_DATRAS_wing_doorspread.R')

#####
#### GET CPUEs AND RIGHT COLUMNS NAMES
#####

# Remove data without length composition or negative values
xx <- subset(survey, HLNoAtLngt<0 | is.na(LngtClass))
no_length_hauls <- sort(unique(xx$HaulID)) # 10,178 hauls without length data

# Only keep abundances/weight
survey <- survey %>%
  #filter(!(HaulID %in% no_length_hauls)) %>% # remove hauls without length data
  mutate(numcpue = TotalNo/Area.swept, # abundance/km2
         wtcpue = CatCatchWgt/(Area.swept*1000), # weight in kg/km2
         numh = (TotalNo*60)/HaulDur, # abundance/hour
         wgth = CatCatchWgt*60/(HaulDur*1000), # weight in kg/h
         num = TotalNo, #raw number of individuals
         wgt = CatCatchWgt/1000, # raw weight in kg
         numlencpue = HLNoAtLngt/Area.swept, #abundance/km2 per length class
         numlenh = HLNoAtLngt*60/HaulDur, #abundance/h per length class
         Season = 'NA',
         Depth = replace(Depth, Depth<0, NA),
         SBT = replace(SBT, SBT<0, NA),
         SST = replace(SST, SST<0, NA),
         LngtClass = ifelse(LngtCode %in% c('.','0'), LngtClass*0.1, LngtClass)) %>%
  # fix unit of length class
  dplyr::rename(Length = LngtClass) %>%
  select(Survey, HaulID, StatRec, Year, Month, Quarter, Season, ShootLat, ShootLong,
         HaulDur, Area.swept, Gear, Depth, SBT, SST, AphiaID, CatIdentifier, numcpue,
         wtcpue, numh, wgth, num, wgt, Length, numlencpue, numlenh)
survey <- data.frame(survey)

#####
## fishglob taxa cleaning ##
#####

# Make AphiaID list per survey
aphia_datras <- survey %>%
  select(Survey, AphiaID) %>%
  dplyr::rename(survey = Survey,
                worms_id_datras = AphiaID) %>%
  distinct()

# Clean taxa north sea
ns_data <- aphia_datras %>% filter(survey=="NS-IBTS")
clean_ns <- clean_taxa(ns_data$worms_id_datras, input_survey = "NS-IBTS",
                       save=F, fishbase=TRUE)

```

```

# Clean taxa bay of biscay
evhoe_data <- aphia_datras %>% filter(survey=="EVHOE")
clean_evhoe <- clean_taxa(evhoe_data$worms_id_datras, input_survey = "EVHOE",
                           save=F, fishbase=TRUE)

# Clean taxa english channel
cgfs_data <- aphia_datras %>% filter(survey=="FR-CGFS")
clean_cgfs <- clean_taxa(cgfs_data$worms_id_datras, input_survey = "FR-CGFS",
                           save=F, fishbase=TRUE)

# Clean taxa baltic sea
bits_data <- aphia_datras %>% filter(survey=="BITS")
clean_bits <- clean_taxa(bits_data$worms_id_datras, input_survey = "BITS",
                           save=F, fishbase=TRUE)

# Clean taxa scottish sea
swc_data <- aphia_datras %>% filter(survey %in% c("SCOWCGFS", "SWC-IBTS"))
clean_swc <- clean_taxa(swc_data$worms_id_datras, input_survey = "SWC-IBTS",
                           save=F, fishbase=TRUE)

# Clean taxa rockall
rock_data <- aphia_datras %>% filter(survey %in% c("SCOROC", "ROCKALL"))
clean_rock <- clean_taxa(rock_data$worms_id_datras, input_survey = "ROCKALL",
                           save=F, fishbase=TRUE)

# Clean taxa irish sea
ir_data <- aphia_datras %>% filter(survey=="IE-IGFS")
clean_ir <- clean_taxa(ir_data$worms_id_datras, input_survey = "IE-IGFS",
                           save=F, fishbase=TRUE)

# Clean taxa northern ireland
nigfs_data <- aphia_datras %>% filter(survey=="NIGFS")
clean_nigfs <- clean_taxa(nigfs_data$worms_id_datras, input_survey = "NIGFS",
                           save=F, fishbase=TRUE)

# Clean taxa for portugal
pt_data <- aphia_datras %>% filter(survey=="PT-IBTS")
clean_pt <- clean_taxa(pt_data$worms_id_datras, input_survey = "PT-IBTS",
                           save=F, fishbase=TRUE)

clean_datras_taxa <- rbind(clean_bits, clean_cgfs, clean_evhoe, clean_ir, clean_nigfs,
                           clean_pt, clean_rock, clean_swc, clean_ns) %>%
  mutate(query = as.numeric(as.vector(query))) %>%
  distinct()

recoded_taxa <- c("Dipturus", "Liparis", "Chelon", "Mustelus", "Alosa", "Argentina",
                  "Callionymus", "Ciliata", "Gaidropsarus", "Sebastes", "Syngnathus",
                  "Pomatoschistus", "Gobius")

spp_to_recode <- c("Dipturus batis", "Dipturus flossada", "Dipturus batis-complex",
                  "Dipturus intermedia", "Liparis montagui", "Liparis liparis",
                  "Liparis liparis liparis", "Chelon aurata", "Chelon ramada",
                  "Mustelus mustelus/asterias", "Mustelus mustelus", "Mustelus asterias",

```

```

    "Alosa alosa", "Alosa fallax", "Argentina silus", "Argentina sphyraena",
    "Callionymus reticulatus", "Callionymus maculatus", "Ciliata mustela",
    "Ciliata septentrionalis", "Gaidropsaurus macrophthalmus",
    "Gaidropsaurus mediterraneus", "Gaidropsaurus vulgaris",
    "Sebastes norvegicus", "Sebastes mentella", "Sebastes marinus",
    "Syngnathus rostellatus", "Syngnathus acus", "Syngnathus typhle",
    "Nerophis ophidion", "Pomatoschistus microps", "Pomatoschistus minutus",
    "Pomatoschistus pictus", "Gobius cobitis", "Gobius niger",
    "Leusueurigobius friesii", "Neogobius melanostomus")

alphaid <- get_wormsid(recoded_taxa)
alphaid <- tibble(taxa = recoded_taxa,
                  worms_id = alphaid[1:length(recoded_taxa)])
clean_manual_recoded <- clean_taxa(alphaid$worms_id, input_survey = "recoded",
                                      save = F, fishbase=TRUE)

clean_datras_taxa <- clean_datras_taxa %>%
  select(-survey) %>%
  mutate(SpecCode = ifelse(taxa %in% spp_to_recode, NA, SpecCode),
         rank = ifelse(taxa %in% spp_to_recode, "Genus", rank),
         #dipturus
         worms_id = ifelse(taxa %in% c("Dipturus batis", "Dipturus flossada",
                                         "Dipturus batis-complex", "Dipturus intermedia"),
                            105762, worms_id),
         taxa = ifelse(taxa %in% c("Dipturus batis", "Dipturus flossada",
                                    "Dipturus batis-complex", "Dipturus intermedia"),
                       "Dipturus", taxa),
         # liparis
         worms_id = ifelse(taxa %in% c("Liparis montagui", "Liparis liparis",
                                         "Liparis liparis liparis"), 126160, worms_id),
         taxa = ifelse(taxa %in% c("Liparis montagui", "Liparis liparis",
                                   "Liparis liparis liparis"), "Liparis", taxa),
         # chelon
         worms_id = ifelse(taxa %in% c("Chelon aurata", "Chelon ramada"), 126030, worms_id),
         taxa = ifelse(taxa %in% c("Chelon aurata", "Chelon ramada"), "Chelon", taxa),
         #mustelus
         worms_id = ifelse(taxa %in% c("Mustelus mustelus/asterias", "Mustelus mustelus",
                                         "Mustelus asterias"), 105732, worms_id),
         taxa = ifelse(taxa %in% c("Mustelus mustelus/asterias", "Mustelus mustelus",
                                   "Mustelus asterias"), "Mustelus", taxa),
         #alosa
         worms_id = ifelse(taxa %in% c("Alosa alosa", "Alosa fallax"), 125715, worms_id),
         taxa = ifelse(taxa %in% c("Alosa alosa", "Alosa fallax"), "Alosa", taxa),
         #argentina
         worms_id = ifelse(taxa %in% c("Argentina silus", "Argentina sphyraena"),
                           125885, worms_id),
         taxa = ifelse(taxa %in% c("Argentina silus", "Argentina sphyraena"),
                       "Argentina", taxa),
         # callionymus
         worms_id = ifelse(taxa %in% c("Callionymus reticulatus", "Callionymus maculatus"),
                           125930, worms_id),
         taxa = ifelse(taxa %in% c("Callionymus reticulatus", "Callionymus maculatus"),
                       "Callionymus", taxa),

```

```

# ciliata
worms_id = ifelse(taxa %in% c("Ciliata mustela", "Ciliata septentrionalis"),
                   125741, worms_id),
taxa = ifelse(taxa %in% c("Ciliata mustela", "Ciliata septentrionalis"),
              "Ciliata", taxa),
# gaidropsarus
worms_id = ifelse(taxa %in% c("Gaidropsaurus macropthalmus",
                               "Gaidropsaurus mediterraneus",
                               "Gaidropsaurus vulgaris"),
                   125743, worms_id),
taxa = ifelse(taxa %in% c("Gaidropsaurus macropthalmus",
                           "Gaidropsaurus mediterraneus",
                           "Gaidropsaurus vulgaris"), "Gaidropsarus", taxa),
# sebastes
worms_id = ifelse(taxa %in% c("Sebastes norvegicus", "Sebastes mentella",
                               "Sebastes marinus"),
                   126175, worms_id),
taxa = ifelse(taxa %in% c("Sebastes norvegicus", "Sebastes mentella",
                           "Sebastes marinus"),
              "Sebastes", taxa),
# syngnathus
worms_id = ifelse(taxa %in% c("Syngnathus rostellatus", "Syngnathus acus",
                               "Syngnathus typhle", "Nerophis ophidion"),
                   126227, worms_id),
taxa = ifelse(taxa %in% c("Syngnathus rostellatus", "Syngnathus acus",
                           "Syngnathus typhle", "Nerophis ophidion"),
              "Syngnathus", taxa),
# pomatosc
worms_id = ifelse(taxa %in% c("Pomatoschistus microps", "Pomatoschistus minutus",
                               "Pomatoschistus pictus"), 125999, worms_id),
taxa = ifelse(taxa %in% c("Pomatoschistus microps", "Pomatoschistus minutus",
                           "Pomatoschistus pictus"), "Pomatoschistus", taxa),
# gobius
worms_id = ifelse(taxa %in% c("Gobius cobitis", "Gobius niger",
                               "Leusueurigobius friesii",
                               "Neogobius melanostomus"), 125988, worms_id),
taxa = ifelse(taxa %in% c("Gobius cobitis", "Gobius niger",
                           "Leusueurigobius friesii",
                           "Neogobius melanostomus"), "Gobius", taxa),
) %>%
distinct()

# add taxonomy to data
survey <- left_join(survey, clean_datras_taxa, by=c("AphiaID" = "query")) %>%
filter(!is.na(AphiaID))

#####
##### RE-CALCULATE WEIGHTS #####
#####
detach(package:worms)
detach(package:plyr)

```

```

# select only certain gears
# 1. summary of gears per survey
gears <- data.frame(survey) %>%
  group_by(Survey, Gear) %>%
  summarise(hauls = length(unique(HaulID)), years = length(unique(Year))) %>%
  select(Survey, Gear, hauls, years)

# 2. only select certain gears per survey (GOV and/or most dominant in cases without GOV)
survey <- survey %>%
  filter(!(Survey=="NS-IBTS" & Gear %in% c('ABD', 'BOT', 'DHT', 'FOT', 'GRT',
                                             'H18', 'HOB', 'HT', 'KAB', 'VIN'))),
  !(Survey=="BITS" & Gear %in% c('CAM', 'CHP', 'DT', 'EGY', 'ESB', 'EXP',
                                   'FOT', 'GRT', 'H20', 'HAK', 'LBT', 'SON')),
  !(Survey=="PT-IBTS" & Gear=='CAR'),
  !(Survey=="Can-Mar" & Gear=='Y36'))

# 3. associate an LME to each haul and make final list of species
### Prepare list for estimating length-weight parameters
list.taxa <- survey %>%
  mutate(species = str_split(taxa, pattern = " ", simplify = TRUE)[,2]) %>%
  select(HaulID, Survey, ShootLat, ShootLong, family, genus, species) %>%
  distinct()

# get LME
library(rgdal)
shape1 <- readOGR(dsn = "length_weight/LME66", layer="LMEs66")
coords <- list.taxa %>%
  dplyr::select(ShootLat, ShootLong, Survey) %>%
  distinct()
str(coords)

coordinates(coords) <- ~ ShootLong + ShootLat
proj4string(coords) <- proj4string(shape1)
lme <- over(coords, shape1)

coords <- list.taxa %>%
  dplyr::select(ShootLat, ShootLong, Survey) %>%
  distinct()
coords <- cbind(coords, lme$LME_NUMBER)
setnames(coords, old='lme$LME_NUMBER', new='lme')

coords$lme <- as.factor(coords$lme)
#Select from each LME 50 long and lat
ind <- c()
for (i in 1:nlevels(coords$lme)){
  ind <- c(ind, sample(which(coords$lme==levels(coords$lme)[i]), 50,
                        replace = FALSE))
}
long50 <- coords$ShootLong[ind]
lat50 <- coords$ShootLat[ind]
lme50 <- rep(levels(coords$lme), each=50)
#For each haul without LME find a close LME that has an LME number already

```

```

nlme <- subset(coords, is.na(lme)) # many hauls without LME 710
nlme$ShootLat <- as.numeric(as.vector(nlme$ShootLat))
nlme$ShootLong <- as.numeric(as.vector(nlme$ShootLong))
long50 <- as.numeric(as.vector(long50))
lat50 <- as.numeric(as.vector(lat50))
dilme <- c()
for (i in 1:length(lme50)){
  dilme <- cbind(dilme, (nlme$ShootLat-lat50[i])**2 + (nlme$ShootLong-long50[i])**2)
}
mindi <- apply(dilme, 1, which.min)
coords$lme[is.na(coords$lme)] <- lme50[mindi]
# assign the closest LME number to each haul without LME

#Check
coords$ShootLat <- as.numeric(as.vector(coords$ShootLat))
coords$ShootLong <- as.numeric(as.vector(coords$ShootLong))

# rockall not assigned to Faroe plateau but to celtic sea LME
coords$lme <- as.character(coords$lme)
coords <- coords %>%
  mutate(lme = replace(lme, Survey == 'ROCKALL', '60')) %>%
  as.data.frame()

survey <- left_join(survey, coords, by=c('ShootLat', 'ShootLong', 'Survey'))

if(need_get_lw_rel == TRUE){
  list.taxa <- survey %>%
    select(taxa, family, genus, taxa, lme, rank, Survey) %>%
    filter(!is.na(taxa),
           rank!="Suborder") %>%
    rename(survey = Survey) %>%
    distinct()

  write.csv(data.frame(list.taxa), file="length_weight/taxa.DATRAS.FB.tofill.csv",
            row.names=FALSE)
  save.image("length_weight/DATRAS_before_lw_4AUG2021.RData")

  ### run length-weight relationships open R 32bit
  list.taxa <- read.csv("length_weight/taxa.DATRAS.FB.tofill.csv")
  get_coeffs(list.taxa, survey="DATRAS", save=TRUE)
}

# 4. re-calculate weights with length-weight relationships
#load("length.weight/DATRAS_before_lw_4AUG2021.RData")
datalw <- read.csv('length_weight/length.weight_DATRAS.csv') %>%
  rename(Survey = survey) %>%
  select(-X) %>%
  mutate(lme = as.character(lme))

survey <- survey %>%
  filter(rank!="Suborder")

```

```

# summarize abundance/weight at the haul level
survey.num <- left_join(survey, datalw, by=c("taxa", "family", "genus", "lme",
                                             "rank", "Survey")) %>%
  select(Survey, HaulID, StatRec, Year, Month, Quarter, Season, ShootLat, ShootLong,
         HaulDur, Area.swept, Gear, Depth, SBT, SST, family, genus, taxa, AphiaID, worms_id,
         SpecCode, kingdom, class, order, phylum, rank,
         CatIdentifier, numcpue, numh, num) %>%
  distinct() %>%
  group_by(Survey, HaulID, StatRec, Year, Month, Quarter, Season, ShootLat, ShootLong,
            HaulDur, Area.swept, Gear, Depth, SBT, SST, family, genus, taxa, AphiaID,
            worms_id, SpecCode, kingdom, class, order, phylum, rank) %>%
  summarize_at(.vars=c('numcpue', 'numh', 'num'), .funs = function(x) sum(x)) %>%
  ungroup()

survey.wgt <- left_join(survey, datalw, by=c("taxa", "family", "genus", "lme",
                                              "rank", "Survey")) %>%
  select(Survey, HaulID, StatRec, Year, Month, Quarter, Season, ShootLat, ShootLong, HaulDur,
         Area.swept, Gear, Depth, SBT, SST, family, genus, taxa, AphiaID, worms_id, SpecCode,
         kingdom, class, order, phylum, rank,
         CatIdentifier, wtcpue, wgth, wgt) %>%
  distinct() %>%
  group_by(Survey, HaulID, StatRec, Year, Month, Quarter, Season, ShootLat, ShootLong,
            HaulDur, Area.swept, Gear, Depth, SBT, SST, family, genus, taxa, AphiaID, worms_id,
            SpecCode, kingdom, class, order, phylum, rank) %>%
  summarize_at(.vars=c('wtcpue', 'wgth', 'wgt'), .funs = function(x) sum(x)) %>%
  ungroup()

survey1 <- full_join(survey.num, survey.wgt,
                      by=c('Survey', 'HaulID', 'StatRec', 'Year', 'Month', 'Quarter',
                           'Season', 'ShootLat', 'ShootLong', 'HaulDur', 'Area.swept',
                           'Gear', 'Depth', 'SBT', 'SST', 'family', 'genus', 'taxa', 'AphiaID',
                           'worms_id', 'SpecCode',
                           'kingdom', 'phylum', 'class', 'order', 'rank'))

# summarize abundance/weight from length data
survey2 <- left_join(survey, datalw, by=c("taxa", "family", "genus", "lme",
                                             "rank", "Survey")) %>%
  mutate(wgtlencpue = numlencpue*a*Length^b/1000, # divide by 1000 to get kg/km2
        wgtlenth = numlenth*a*Length^b/1000) %>% # divide by 1000 to get kg/h
  group_by(Survey, HaulID, StatRec, Year, Month, Quarter, Season, ShootLat, ShootLong, HaulDur,
            Area.swept, Gear, Depth, SBT, SST, family, genus, taxa, AphiaID, worms_id, SpecCode, a, b,
            kingdom, class, order, phylum, rank) %>%
  summarize_at(.vars=c('numlencpue', 'numlenth', 'wgtlencpue', 'wgtlenth'),
               .funs=function(x) sum(x)) %>%
  ungroup()

# merge both and compare
nrow(survey1)==nrow(survey2)
survey3 <- full_join(survey1, survey2, by=c('Survey', 'HaulID', 'StatRec', 'Year', 'Month',
                                              'Quarter', 'Season', 'ShootLat', 'ShootLong',
                                              'HaulDur', 'Area.swept', 'Gear', 'Depth',
                                              'SBT', 'SST', 'family', 'genus', 'taxa',
                                              'AphiaID', 'worms_id', 'SpecCode',
                                              'kingdom', 'phylum', 'class', 'order', 'rank'))

```

```

'kingdom', 'phylum', 'class', 'order', 'rank'))
```

library(ggplot2)

# correlation between abundances to check calculations are right

```

cor(x = survey3$numh, y = survey3$numlenh, method = 'pearson')
xx <- subset(survey3, !is.na(numcpue))
cor(x = xx$numcpue, y = xx$numlenh, method = 'pearson')
```

# check weights

```

xx <- subset(survey3, wtcpue >0 & wgtlencpue>0)
cor(x = xx$wtcpue , y = xx$wgtlencpue, method = 'pearson')
```

```

xx <- subset(survey3, wgth>0 & wgtlenh>0)
cor(x = xx$wgth, y = xx$wgtlenh, method = 'pearson')
```

### cor = 0.92 and 0.90 so something does not work.

```

#####
# CHECK PER SURVEY
#####

# no zeros
xx <- subset(survey3, wgth>0 & wgtlenh>0)

# rockall looks OK
ggplot(subset(xx, Survey=='ROCKALL'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

# IE-IGFS looks OK
ggplot(subset(xx, Survey=='IE-IGFS'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

# NIGFS looks OK
ggplot(subset(xx, Survey=='NIGFS'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

# PT-IBTS looks OK
ggplot(subset(xx, Survey=='PT-IBTS'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

# FR-CGFS looks OK
ggplot(subset(xx, Survey=='FR-CGFS'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

# SWC-IBTS issue
ggplot(subset(xx, Survey=='SWC-IBTS'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",

```

```

    linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp <- subset(xx, Survey=='SWC-IBTS') %>%
  select(HaulID,wgtlenh,wgth) %>%
  distinct() %>%
  group_by(HaulID) %>%
  summarize_at(.vars=c('wgtlenh', 'wgth'), .funs = function(x) sum(x)) %>%
  ungroup() %>%
  as.data.frame()

ggplot(comp, aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp$factor <- comp$wgtlenh / comp$wgth
plot(comp$factor)
resc <- comp$HaulID[comp$factor > 40]

# after check with original haul length data (HL) for some resc haulid, weight
# is clearly wrong factor 100
survey3 <- survey3 %>%
  mutate(wtcpue = if_else(HaulID %in% resc, wtcpue*100,wtcpue),
         wgth = if_else(HaulID %in% resc , wgth*100,wgth),
         wgt = if_else(HaulID %in% resc , wgt*100,wgt))

# BITS issue
ggplot(subset(xx, Survey=='BITS'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp <- subset(xx, Survey=='BITS') %>%
  select(HaulID,wgtlenh,wgth) %>%
  distinct() %>%
  group_by(HaulID) %>%
  summarize_at(.vars=c('wgtlenh', 'wgth'), .funs = function(x) sum(x)) %>%
  ungroup() %>%
  as.data.frame()

ggplot(comp, aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp$factor <- comp$wgtlenh / comp$wgth
plot(comp$factor)
resc <- comp$HaulID[comp$factor > 40]

# after check with original haul length data (HL) for some resc haulid, weight
# is clearly wrong factor 100
survey3 <- survey3 %>%
  mutate(wtcpue = if_else(HaulID %in% resc, wtcpue*100,wtcpue),
         wgth = if_else(HaulID %in% resc , wgth*100,wgth),
         wgt = if_else(HaulID %in% resc , wgt*100,wgt))

```

```

# EVHOE may have an issue, no changes as not very clear
ggplot(subset(xx, Survey=='EVHOE'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp <- subset(xx, Survey=='EVHOE') %>%
  select(HaulID,wgtlenh,wgth) %>%
  distinct() %>%
  group_by(HaulID) %>%
  summarize_at(.vars=c('wgtlenh', 'wgth'), .funs = function(x) sum(x)) %>%
  ungroup() %>%
  as.data.frame()

ggplot(comp, aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp$factor <- comp$wgtlenh / comp$wgth
plot(comp$factor)

# NS - IBTS issue
ggplot(subset(xx, Survey=='NS-IBTS'), aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp <- subset(xx, Survey=='NS-IBTS') %>%
  select(HaulID,wgtlenh,wgth) %>%
  distinct() %>%
  group_by(HaulID) %>%
  summarize_at(.vars=c('wgtlenh', 'wgth'), .funs = function(x) sum(x)) %>%
  ungroup() %>%
  as.data.frame()

ggplot(comp, aes(x=wgth, y=wgtlenh)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

comp$factor <- comp$wgtlenh / comp$wgth
comp$uni <- c(1:nrow(comp))
plot(comp$factor-comp$uni,ylim=c(0,120))
points(comp$factor[comp$factor > 20]~comp$uni[comp$factor > 20],col="red")
points(comp$factor[comp$factor > 8 & comp$factor <20]~
       comp$uni[comp$factor > 8 & comp$factor <20],col="blue")

# two issues - one estimate 100 times higher based on length, the other 10 times
resc <- comp$HaulID[comp$factor > 20]
resc2 <- comp$HaulID[comp$factor > 8 & comp$factor <20]

# after check with original haul length data (HL) for some resc haulid, weight
# is clearly wrong factor 100
# and also a cluster of factor 10
survey3 <- survey3 %>%
  mutate(wtcpue = if_else(HaulID %in% resc, wtcpue*100,wtcpue),

```

```

    wgth = if_else(HaulID %in% resc , wgth*100,wgth),
    wgt = if_else(HaulID %in% resc , wgt*100,wgt))

survey3 <- survey3 %>%
  mutate(wtcpue = if_else(HaulID %in% resc2, wtcpue*10,wtcpue),
         wgth = if_else(HaulID %in% resc2 , wgth*10,wgth),
         wgt = if_else(HaulID %in% resc2 , wgt*10,wgt))

# check again correlations
xx <- subset(survey3, wtcpue> 0 & wgtlencpue>0)
cor(x = xx$wtcpue , y = xx$wgtlencpue, method = 'pearson') # looks better

xx <- subset(survey3, wgth>0 & wgtlenth>0)
cor(x = xx$wgth, y = xx$wgtlenth, method = 'pearson') # looks better

# now check per haul without zeros, NAs
xx <- subset(survey3, wtcpue>0 & wgtlencpue>0)

comp <- xx %>%
  select(HaulID,wgtlencpue,wtcpue) %>%
  distinct() %>%
  group_by(HaulID) %>%
  summarize_at(.vars=c('wgtlencpue', 'wtcpue'), .funs = function(x) sum(x)) %>%
  ungroup() %>%
  as.data.frame()

ggplot(comp, aes(x=wtcpue, y=wgtlencpue)) + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="red",
              linetype="dashed", size=0.5) + scale_x_log10() + scale_y_log10()

cor(x = xx$wtcpue , y = xx$wgtlencpue, method = 'pearson')
# [1] 0.9635742

#####
##### Fishglob format #####
#####

survey4 <- survey3 %>%
  rename(survey = Survey,
         haul_id = HaulID,
         stat_rec = StatRec,
         year = Year,
         month = Month,
         quarter = Quarter,
         season = Season,
         latitude = ShootLat,
         longitude = ShootLong,
         haul_dur = HaulDur,
         area_swept = Area.swept,
         gear = Gear,
         depth = Depth,
         sbt = SBT,

```

```

    sst = SST,
    verbatim_aphia_id = AphiaID,
    aphia_id = worms_id,
    accepted_name = taxa,
) %>%
mutate(day = NA_integer_,
       verbatim_name = NA_character_,
       station = NA_character_,
       stratum = NA_character_,
       sub_area = NA_character_,
       continent = "europe",
       country = case_when(survey=="PT-IBTS" ~ "portugal",
                            survey=="EVHOE" ~ "france",
                            survey=="IE-IGFS" ~ "ireland",
                            survey %in% c("ROCKALL","SWC-IBTS","NIGFS") ~ "uk",
                            survey=="FR-CGFS" ~ "france",
                            survey %in% c("NS-IBTS","BITS") ~ "multi-countries"),
       num = numlencpue*area_swept,
       num_cpue = numlenh,
       num_cpua = numlencpue,
       wgt = wgtlencpue*area_swept,
       wgt_cpue = wgtlenh,
       wgt_cpua = wgtlencpue,
       haul_dur = haul_dur/60,
       source = "DATRAS ICES",
       timestamp = "2021-07",
       survey_unit = ifelse(survey %in% c("BITS","NS-IBTS","SWC-IBTS"),
                            paste0(survey,"-",quarter),survey),
       survey_unit = ifelse(survey %in% c("NEUS","SEUS","SCS","GMEX"),
                            paste0(survey,"-",season),survey_unit)) %>%
# Final format
select(fishglob_data_columns$`Column name fishglob`)

#####
# Save database
#####

# Just run this routine should be good for all
surveys <- sort(unique(survey4$survey))
for(i in 1:length(surveys)){
  xx <- survey4 %>%
    filter(survey == surveys[i])
  write_clean_data(data = xx, survey = surveys[i], overwrite = T,
                  rdata = TRUE)
}

# -----#
#### FAGS ####
# -----#

```

```

#install required packages that are not already installed
required_packages <- c("data.table",
                      "devtools",
                      "dgridR",
                      "dplyr",
                      "fields",
                      "forcats",
                      "ggplot2",
                      "here",
                      "magrittr",
                      "maps",
                      "maptools",
                      "raster",
                      "rcompendium",
                      "readr",
                      "remotes",
                      "rrtools",
                      "sf",
                      "sp",
                      "tidyR",
                      "usethis")

not_installed <- required_packages[!(required_packages %in% installed.packages()[, "Package"])]
if(length(not_installed)) install.packages(not_installed)

#load pipe operator
library(magrittr)

##### Apply taxonomic flagging per region
#get vector of regions (here the survey column)
regions <- levels(as.factor(survey4$survey))

#run flag_spp function in a loop
for (r in regions) {
  flag_spp(survey4, r)
}

##### Apply trimming per survey_unit method 1
#apply trimming for hex size 7
dat_new_method1_hex7 <- apply_trimming_per_survey_unit_method1(survey4, 7)

#apply trimming for hex size 8
dat_new_method1_hex8 <- apply_trimming_per_survey_unit_method1(survey4, 8)

##### Apply trimming per survey_unit method 2
dat_new_method2 <- apply_trimming_per_survey_unit_method2(survey4)

#-----#
#### ADD STANDARDIZATION FLAGS ####
#-----#
surveys <- sort(unique(survey4$survey))

```

```

survey_units <- sort(unique(survey4$survey_unit))
survey_std <- survey4 %>%
  mutate(flag_taxa = NA_character_,
         flag_trimming_hex7_0 = NA_character_,
         flag_trimming_hex7_2 = NA_character_,
         flag_trimming_hex8_0 = NA_character_,
         flag_trimming_hex8_2 = NA_character_,
         flag_trimming_2 = NA_character_)

# integrate taxonomic flags
for(i in 1:length(surveys)){
  if(!surveys[i] %in% c("FALK", "GSL-N", "MRT", "NZ-CHAT", "SCS", "SWC-IBTS")){
    xx <- data.frame(read_delim(paste0("outputs/Flags/taxonomic_flagging/",
                                         surveys[i], "_flagsp.txt"),
                                 delim=";", escape_double = FALSE, col_names = FALSE,
                                 trim_ws = TRUE))
    xx <- as.vector(unlist(xx[1,]))
    survey_std <- survey_std %>%
      mutate(flag_taxa = ifelse(survey == surveys[i] & accepted_name %in% xx,
                                "TRUE", flag_taxa))

    rm(xx)
  }
}

# integrate spatio-temporal flags
for(i in 1:length(survey_units)){

  if(!survey_units[i] %in% c("DFO-SOG", "IS-TAU", "SCS-FALL", "WBLS")){

    hex_res7_0 <- read.csv(paste0("outputs/Flags/trimming_method1/hex_res7/",
                                   survey_units[i], "_hex_res_7_trimming_0_hauls_removed.csv"),
                            sep = ";")
    hex_res7_0 <- as.vector(hex_res7_0[,1])

    hex_res7_2 <- read.csv(paste0("outputs/Flags/trimming_method1/hex_res7/",
                                   survey_units[i], "_hex_res_7_trimming_02_hauls_removed.csv"),
                            sep = ";")
    hex_res7_2 <- as.vector(hex_res7_2[,1])

    hex_res8_0 <- read.csv(paste0("outputs/Flags/trimming_method1/hex_res8/",
                                   survey_units[i], "_hex_res_8_trimming_0_hauls_removed.csv"),
                            sep= ";")
    hex_res8_0 <- as.vector(hex_res8_0[,1])

    hex_res8_2 <- read.csv(paste0("outputs/Flags/trimming_method1/hex_res8/",
                                   survey_units[i], "_hex_res_8_trimming_02_hauls_removed.csv"),
                            sep = ";")
    hex_res8_2 <- as.vector(hex_res8_2[,1])

    trim_2 <- read.csv(paste0("outputs/Flags/trimming_method2/",
                               survey_units[i], "_hauls_removed.csv"))
  }
}

```

```

trim_2 <- as.vector(trim_2[,1])

survey_std <- survey_std %>%
  mutate(flag_trimming_hex7_0 = ifelse(survey_unit == survey_units[i] & haul_id %in% hex_res7_0,
                                         "TRUE",flag_trimming_hex7_0),
         flag_trimming_hex7_2 = ifelse(survey_unit == survey_units[i] & haul_id %in% hex_res7_2,
                                         "TRUE",flag_trimming_hex7_2),
         flag_trimming_hex8_0 = ifelse(survey_unit == survey_units[i] & haul_id %in% hex_res8_0,
                                         "TRUE",flag_trimming_hex8_0),
         flag_trimming_hex8_2 = ifelse(survey_unit == survey_units[i] & haul_id %in% hex_res8_2,
                                         "TRUE",flag_trimming_hex8_2),
         flag_trimming_2 = ifelse(survey_unit == survey_units[i] & haul_id %in% trim_2,
                                         "TRUE", flag_trimming_2)
  )
  rm(hex_res7_0, hex_res7_2, hex_res8_0, hex_res8_2, trim_2)
}

# Just run this routine should be good for all
for(i in 1:length(surveys)){
  xx <- survey_std %>%
    filter(survey == surveys[i])
  write_clean_data(data = xx, survey = paste0(surveys[i],"_std"), overwrite = T,
                  rdata = TRUE)
}

```

## 1. Overview of the survey data table

survey	source	timestamp	haul_id		country	sub_area
FR-CGFS	DATRAS ICES	2021-07	FR-CGFS 1998 4 FR 35GD GOV C98001 1		france	NA
FR-CGFS	DATRAS ICES	2021-07	FR-CGFS 1998 4 FR 35GD GOV C98001 1		france	NA
FR-CGFS	DATRAS ICES	2021-07	FR-CGFS 1998 4 FR 35GD GOV C98001 1		france	NA
FR-CGFS	DATRAS ICES	2021-07	FR-CGFS 1998 4 FR 35GD GOV C98001 1		france	NA
FR-CGFS	DATRAS ICES	2021-07	FR-CGFS 1998 4 FR 35GD GOV C98001 1		france	NA

continent	stat_rec	station	stratum	year	month	day	quarter	season
europe	30F1	NA	NA	1998	10	NA	4	NA
europe	30F1	NA	NA	1998	10	NA	4	NA
europe	30F1	NA	NA	1998	10	NA	4	NA
europe	30F1	NA	NA	1998	10	NA	4	NA
europe	30F1	NA	NA	1998	10	NA	4	NA

latitude	longitude	haul_dur	area_swept	gear	depth	sbt	sst
50.7717	1.558	0.3	0.0273602	GOV	18	17	17
50.7717	1.558	0.3	0.0273602	GOV	18	17	17
50.7717	1.558	0.3	0.0273602	GOV	18	17	17
50.7717	1.558	0.3	0.0273602	GOV	18	17	17
50.7717	1.558	0.3	0.0273602	GOV	18	17	17

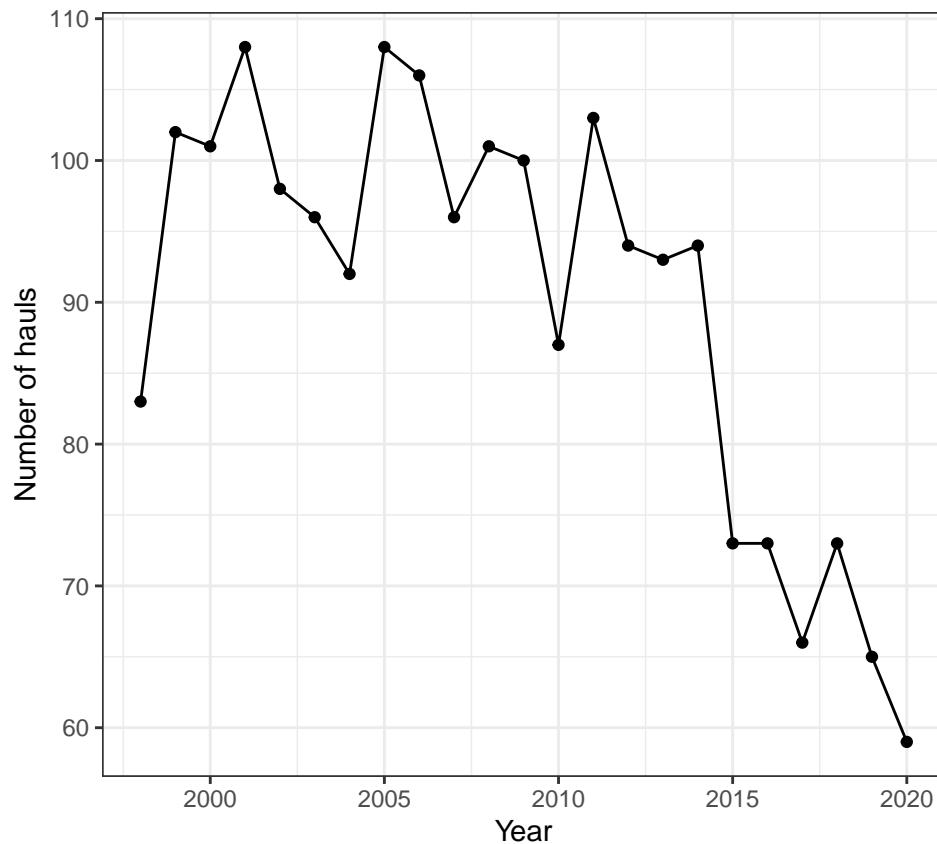
num	num_cpue	num_cpua	wgt	wgt_cpue	wgt_cpua	verbatim_name
34.998	116.66	1279.15811	4.7851339	15.9504462	174.894073	NA
3.999	13.33	146.16130	0.0301996	0.1006655	1.103781	NA
18.996	63.32	694.29360	0.0901120	0.3003735	3.293547	NA
1.998	6.66	73.02583	1.0316816	3.4389385	37.707407	NA
1.998	6.66	73.02583	3.0011300	10.0037668	109.689692	NA

verbatim_aphia_id	accepted_name	aphia_id	SpecCode	kingdom
126822	Trachurus trachurus	126822	1365	Animalia
126417	Clupea harengus	126417	24	Animalia
126425	Sprattus sprattus	126425	1357	Animalia
126975	Dicentrarchus labrax	126975	63	Animalia
1044127	Chelon auratus	1044127	1735	Animalia

phylum	class	order	family	genus	rank	survey_unit
Chordata	Teleostei	Carangiformes	Carangidae	Trachurus	Species	FR-CGFS
Chordata	Teleostei	Clupeiformes	Clupeidae	Clupea	Species	FR-CGFS
Chordata	Teleostei	Clupeiformes	Clupeidae	Sprattus	Species	FR-CGFS
Chordata	Teleostei	Eupercaria incertae sedis	Moronidae	Dicentrarchus	Species	FR-CGFS
Chordata	Teleostei	Mugiliformes	Mugilidae	Chelon	Species	FR-CGFS

## 2. Summary of sampling intensity

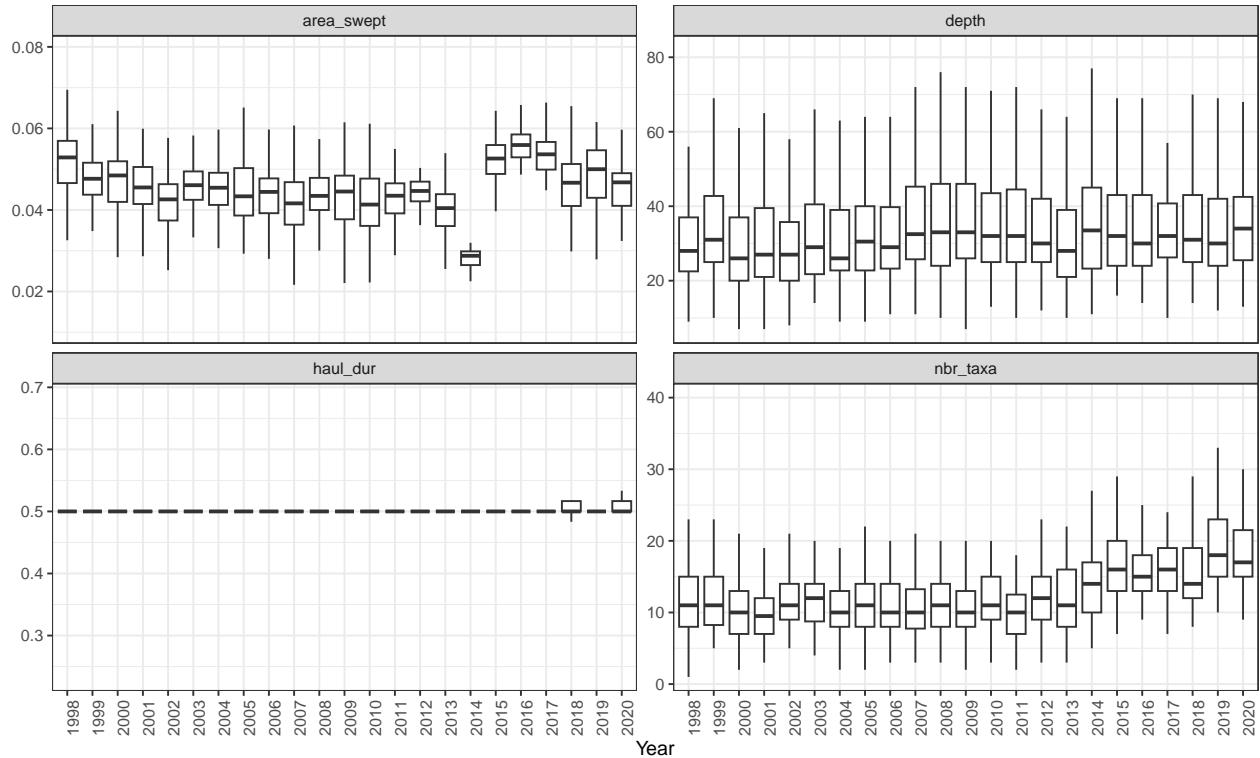
Number of hauls per year performed during the survey after data processing.



### 3. Summary of sampling variables from the survey

Here we show the yearly total and average of the following variables reported in the survey data:

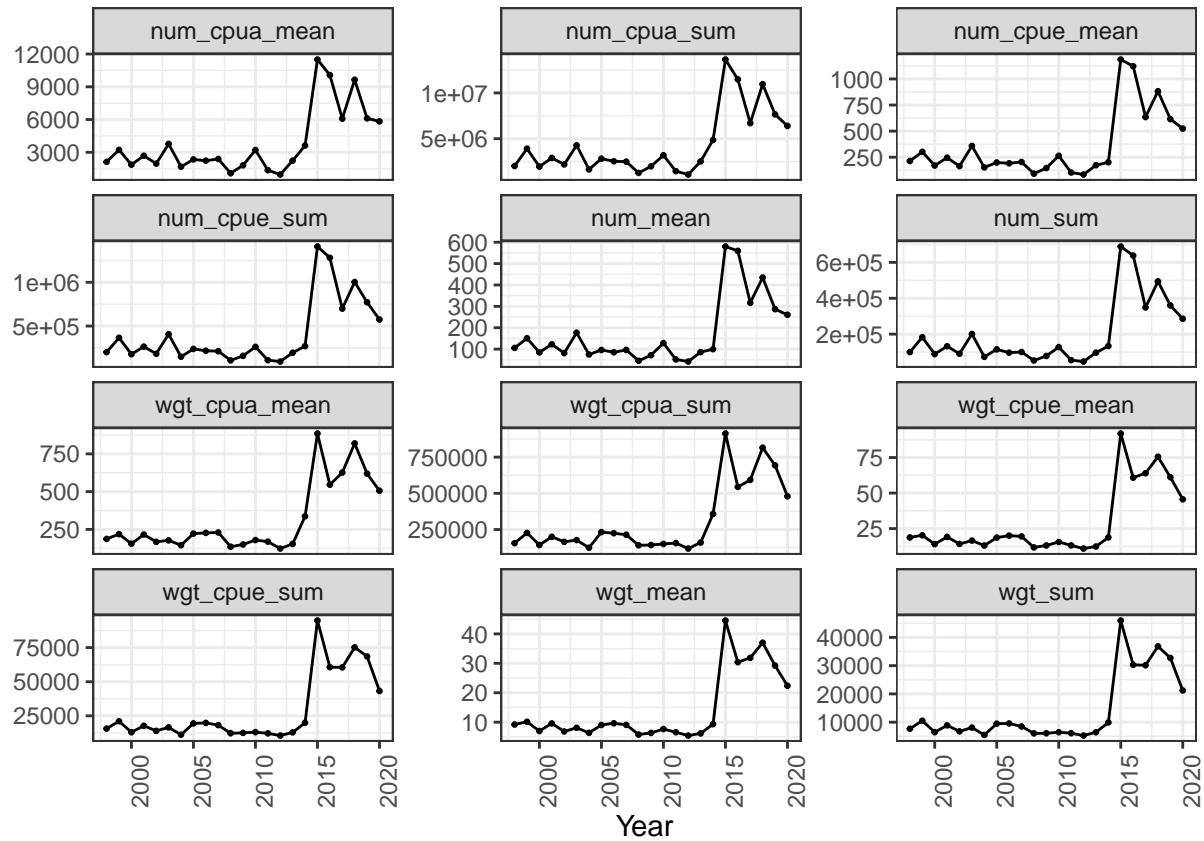
- *area\_swept*, swept area by the bottom trawl gear  $km^2$
- *depth*, sampling depth in  $m$
- *haul\_dur*, haul sampling duration *hours*
- *number of marine fish taxa*, taxa were cleaned following the last version of taxonomy from the World Register of Marine Species (<https://www.marinespecies.org/>, October 2021)



#### 4. Summary of biological variables

Here we display the yearly total and average across hauls of the following variables recorded in the data:

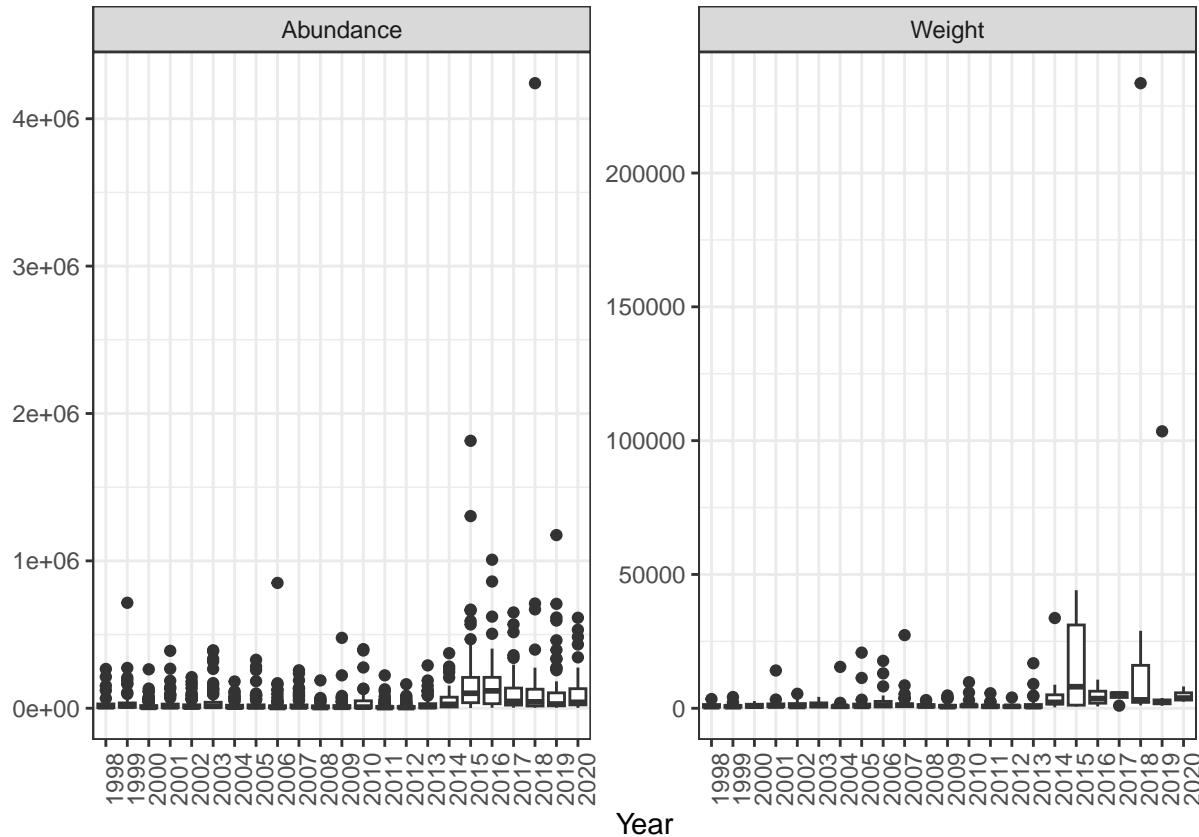
- $num\_cpua$ , number of individuals (abundance) in  $\frac{individuals}{km^2}$
- $num\_cpue$ , number of individuals (abundance) in  $\frac{individuals}{h}$
- $num$ , number of individuals (abundance)
- $wgt\_cpua$ , weight in  $\frac{kg}{km^2}$
- $wgt\_cpue$ , weight in  $\frac{kg}{h}$
- $wgt$ , weight in  $kg$



## 5. Extreme values

Here we show a yearly total distribution of the biomass data to visualize outliers:

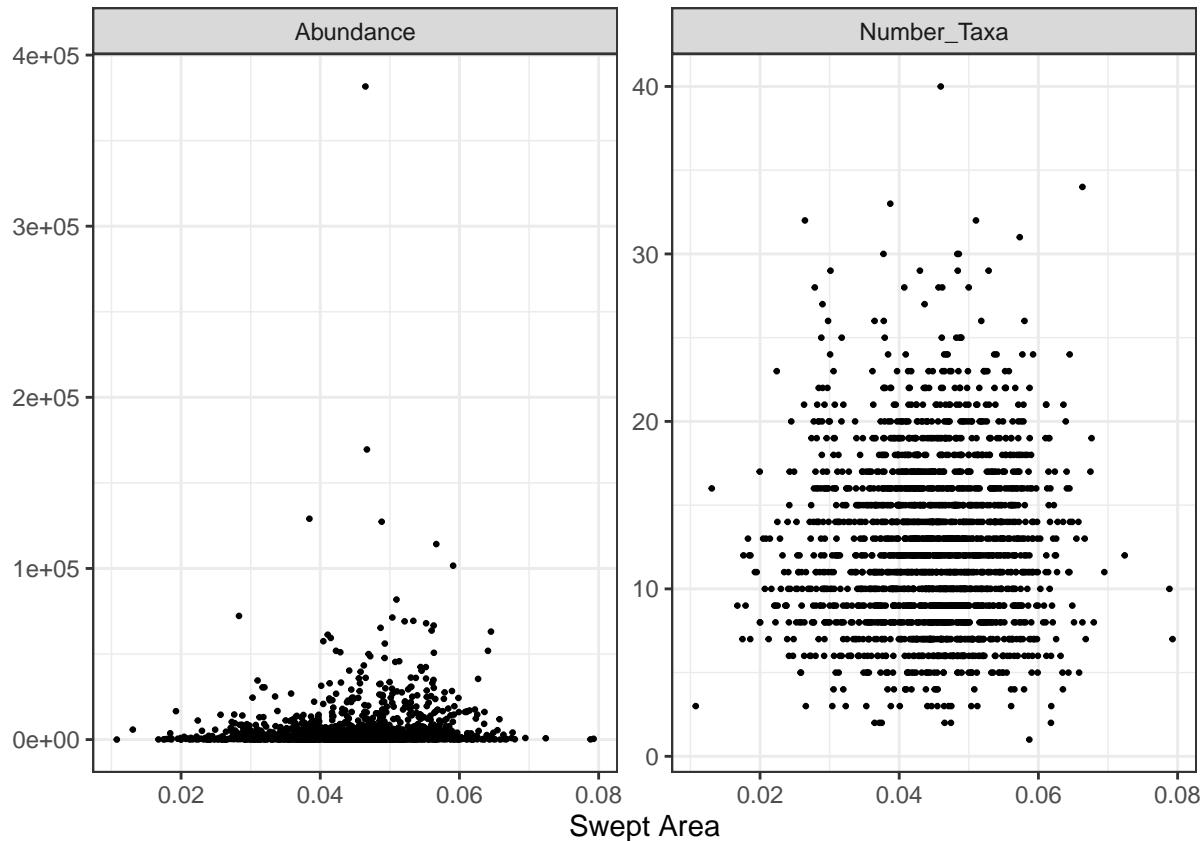
- $num\_cpue$ , number of individuals (abundance) in  $\frac{individuals}{km^2}$
- $wgt\_cpue$ , weight in  $\frac{kg}{km^2}$



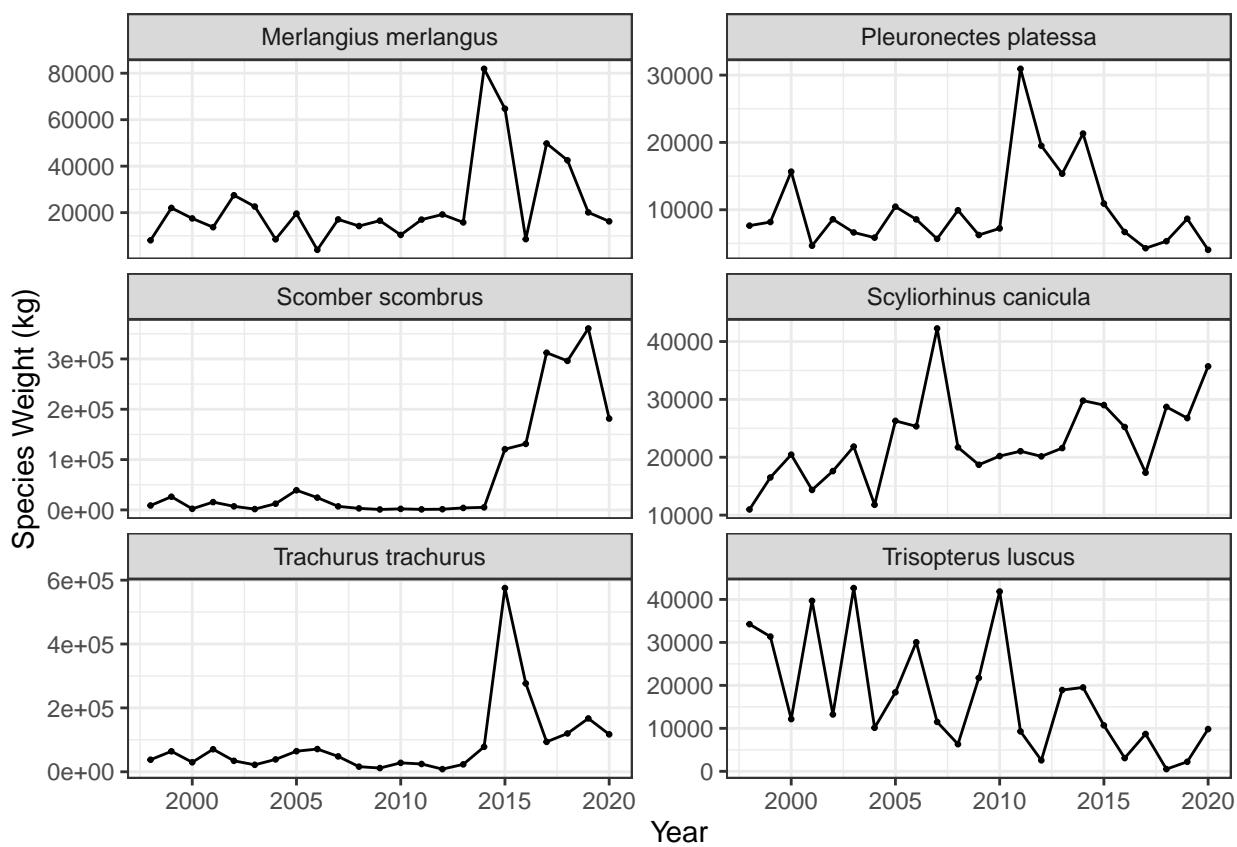
## 6. Summary of variables against swept area

Here we show the total abundance and number of taxa relationships with the area swept:

- $nbr\_taxa$ , number of marine fish taxa after taxonomic data cleaning
- $num\_cpua$ , number of individuals (abundance) in  $\frac{individuals}{km^2}$
- $wgt\_cpua$ , weight in  $\frac{kg}{km^2}$

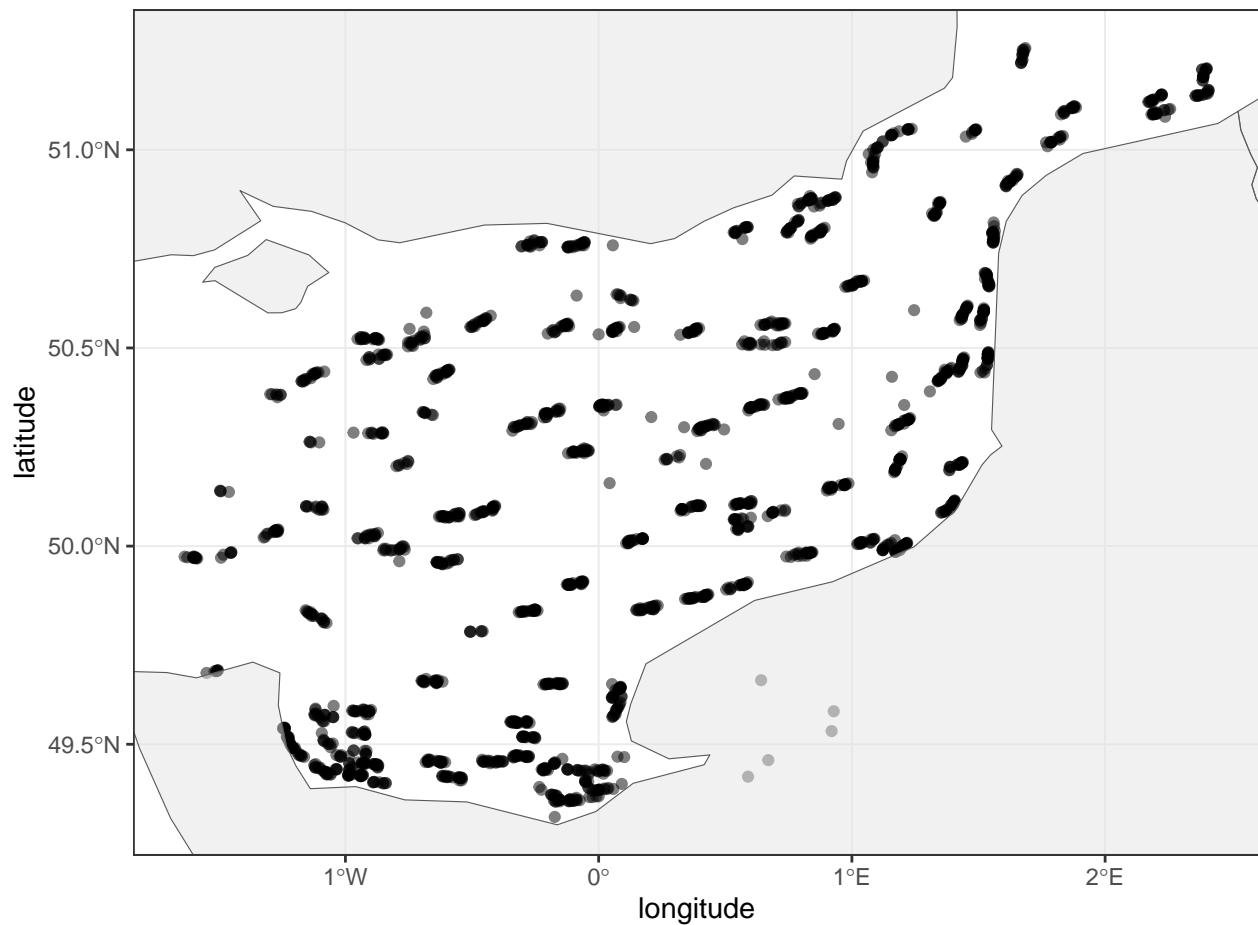


## 7. Abundance or Weight trends of the six most abundant species



## 8. Distribution mapping

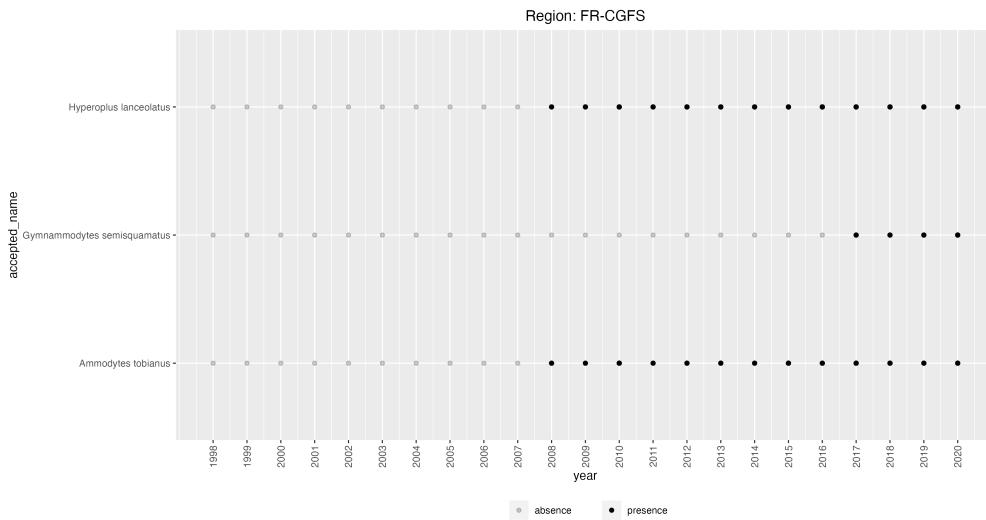
Map of the sampling distribution in space. Note that we only show one year per coordinate.



## 9. Taxonomic flagging

This species flagging method was adapted from <https://github.com/pinskylab/OceanAdapt/blob/master/R/add-spp-to-taxonomy.Rmd#L33>

Visualization of flagged taxa



Statistics related to the taxonomic flagging outputs

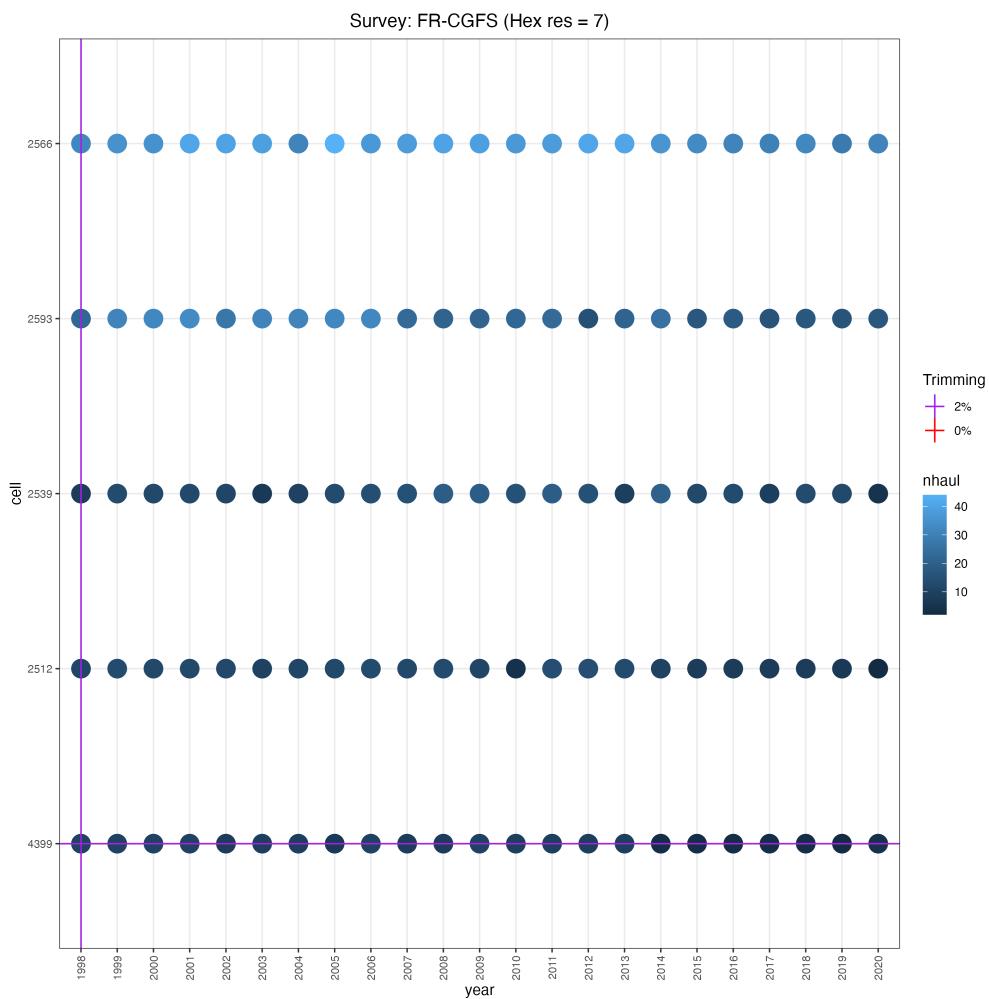
Total number of species	113.0
Percentage of species flagged	2.7

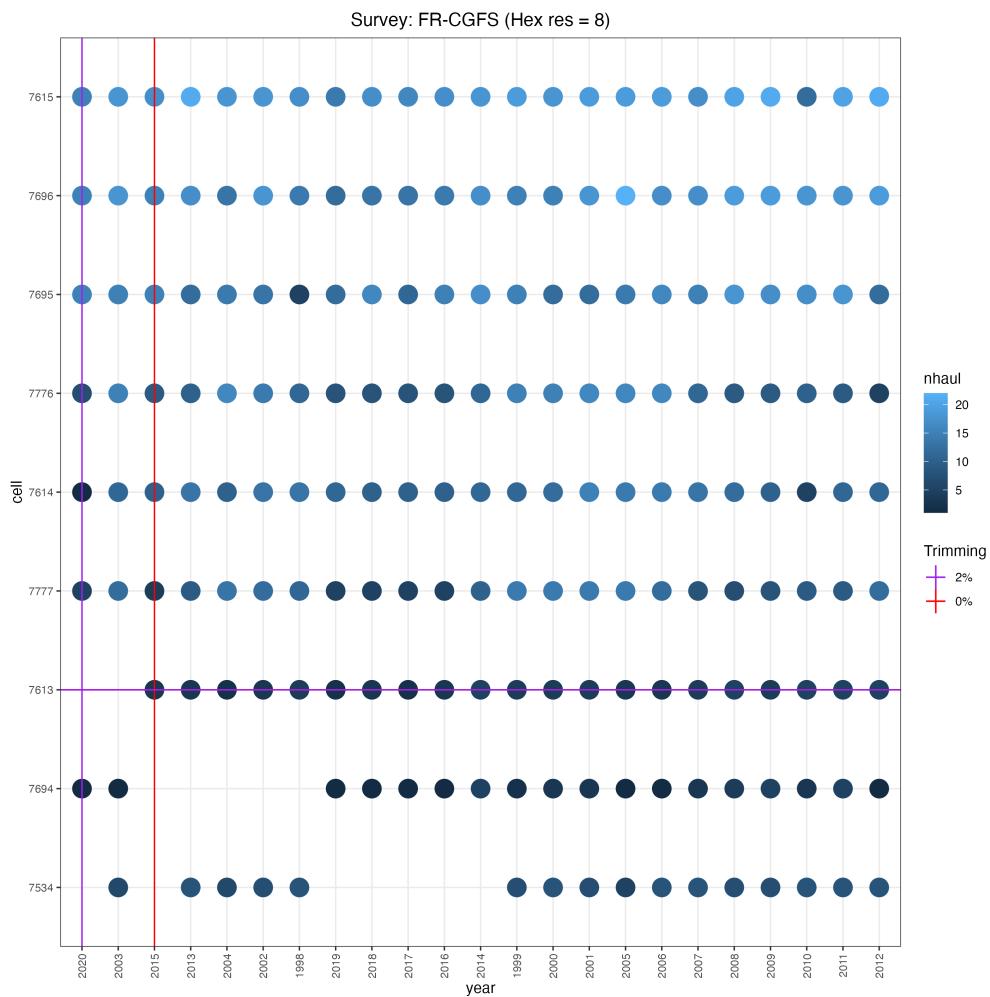
## 10. Spatio-temporal standardization

### a. Standardization method 1

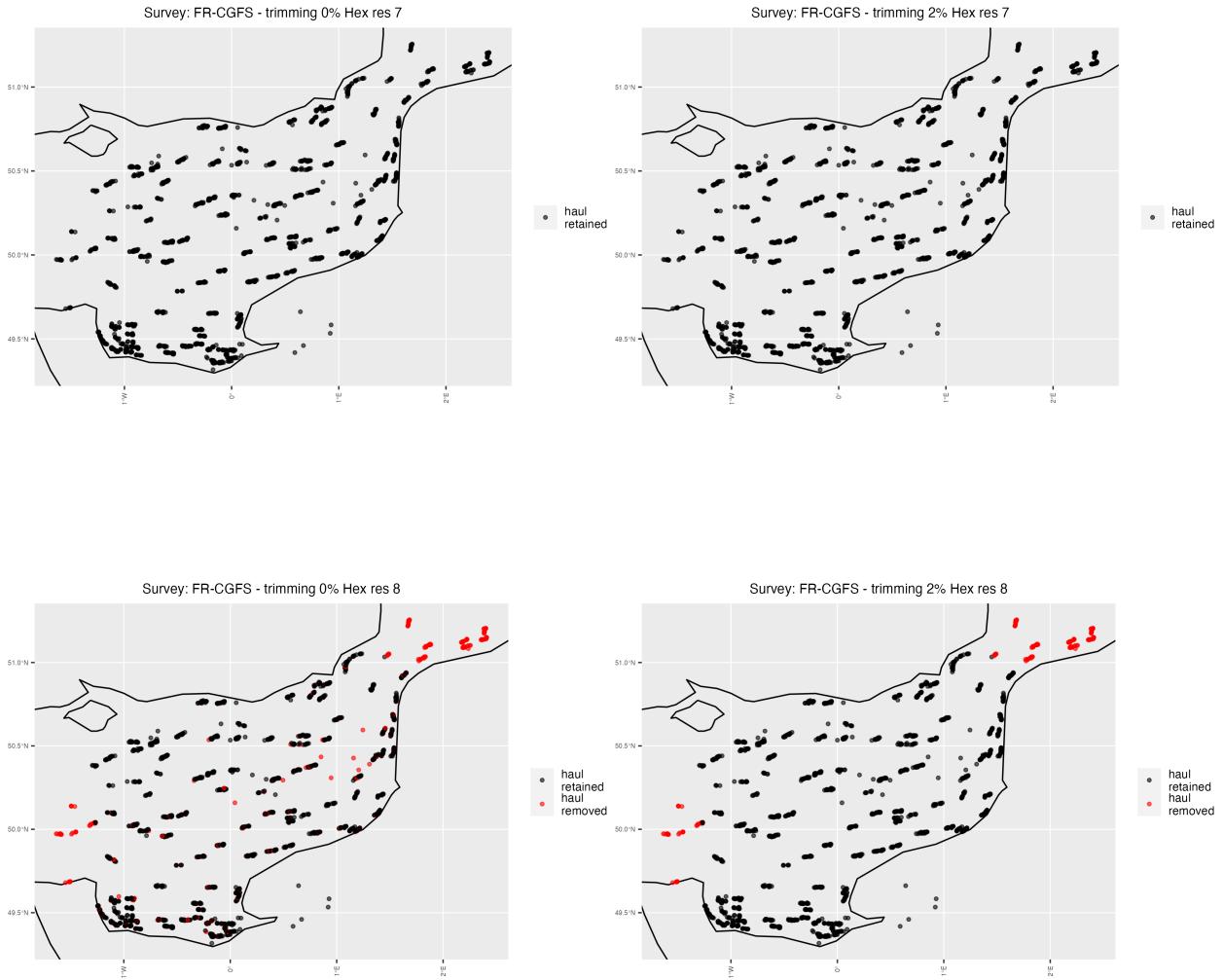
This standardization method was adapted from [https://github.com/zoekitchel/trawl\\_spatial\\_turnover/blob/master/data\\_prep\\_code/species/explore\\_NorthSea\\_trimming.Rmd](https://github.com/zoekitchel/trawl_spatial_turnover/blob/master/data_prep_code/species/explore_NorthSea_trimming.Rmd)  
It was run for hex resolution 7 and 8.

Plot of number of cells x years with overlaid flagging options

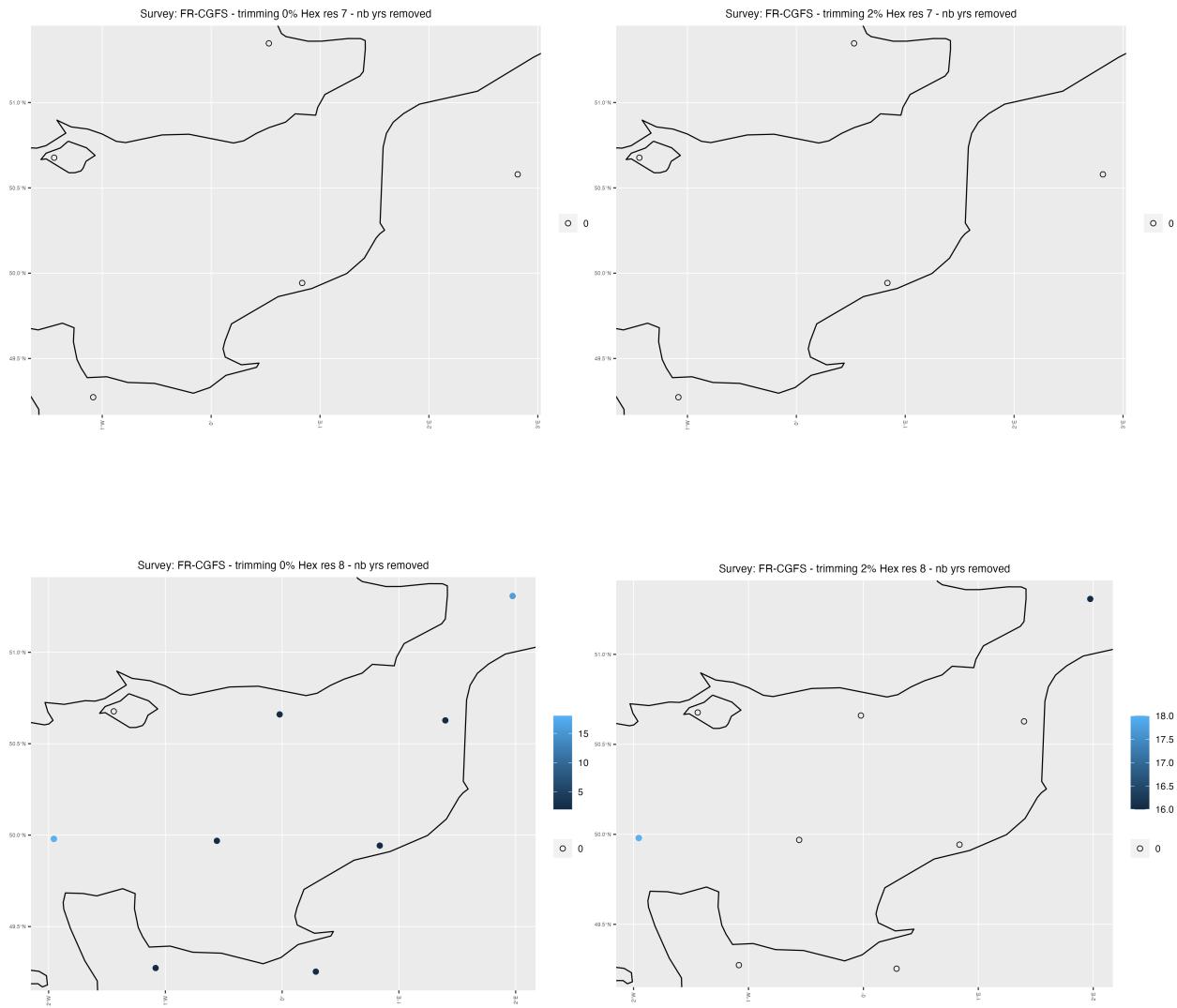




Map of hauls retained and removed per flagging method and threshold



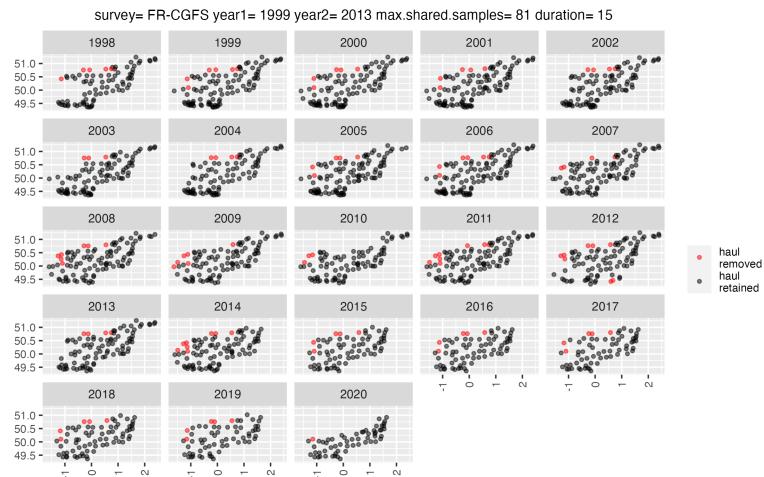
Map of numbers of years removed per grid cell and flagging method/threshold



## b. Standardization method 2

This standardization method was adapted from BioTIME code from [https://github.com/Wubing-Xu/Range\\_size\\_winners\\_losers](https://github.com/Wubing-Xu/Range_size_winners_losers)

Map of hauls retained and removed



### c. Standardization summary

Statistics of hauls removed for each standardization method

summary	grid cell 7, 0% threshold	grid cell 7, 2% threshold	grid cell 8, 0% threshold	grid cell 8, 2% threshold	method 2 (biotime)
number of hauls removed	0	0	305.0	158.0	1639.0
percentage of hauls removed	0	0	14.7	7.6	6.4