SURIMI WP2 - Data Disaggregation protocols

Sbrana, Sabatella

2025-08-07

Table of Contents

# Introduction

The EU Horizon SURIMI project aims to deliver a suite of ready-to-use socio-economic and ecological simulation models that are integrated into the EU Digital Twin Ocean. These models leverage a wide range of EU fishery activity data, including datasets from the Fisheries Data Collection Framework (DCF), to support evidence-based policy and sustainable management of marine resources. However, most data are collected in an aggregated form, which limits their spatial resolution and applicability to the fine-scale assessment and simulation of vessel behaviour. For these reasons, one of the objectives of the SURIMI project is to create a set of algorithms to disaggregate the socio-economic dataset at a finer spatial and technical level or even at the level of individual fishing vessels, using public data and online resources (DCF and Global Fishing Watch).

### Summary of the procedure

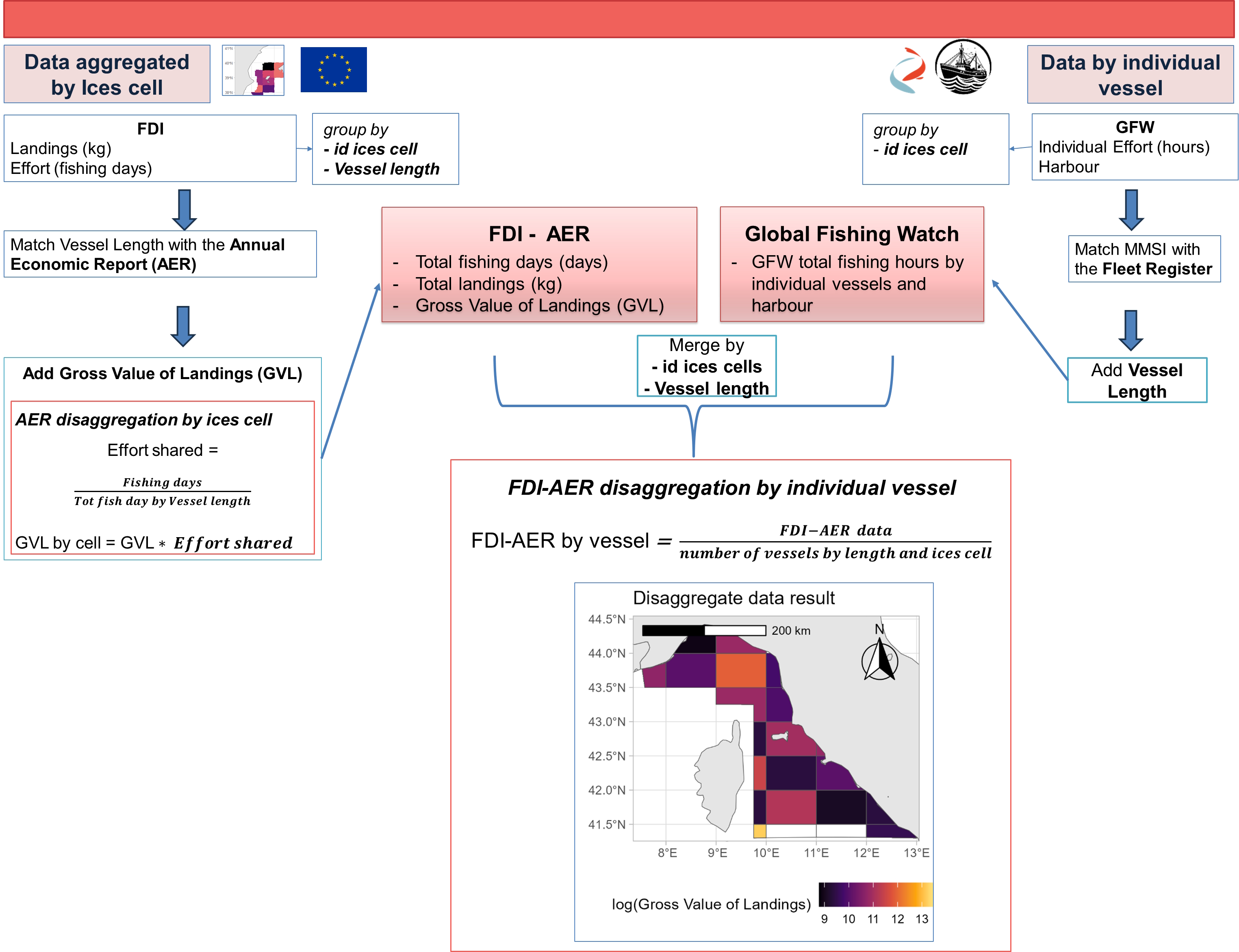
This procedure (Fig XX) outlines a method to estimate fisheries economic indicators at the level of individual vessels and ICES cells by integrating multiple data sources through a disaggregation methodology. Aggregated data from the EU’s FDI—including effort and landing data —is provided at the ICES cell level and by metier, while the AER database, including economic indicators, is grouped by supra-region, country and fleet segment. Global Fishing Watch (GFW) data is used to provide total fishing hours per vessel and harbour of landings.

Firstly, the FDI data were matched to the Annual Economic Report of Fisheries (AER) according to vessel length, in order to extract the relevant economic indicators. For this example, the focus was on the Gross Value of Landings (GVL); however, any information contained in the AER can be extracted.

In parallel, Global Fishing Watch (GFW) data was used to provide information on the effort (hours of fishing) of each individual vessel and their associated harbours for the selected case study. The MMSI identifier was used to match GFW individual vessels with the EU Fleet Register to assign vessel lengths. Finally, individual vessels were grouped by ICES cell and length to align with the FDI dataset.

**Note**: Here, we use public GFW data to demonstrate a procedure in cases where access to other types of data is not possible. However, the GFW dataset is restricted to a few fisheries as it only shows those with AIS coverage. However, if more detailed data are available—such as those obtained from the Vessel Monitoring System (VMS)—the procedure can be reliably replicated using that dataset as well.

The FDI-AER and GFW datasets were merged using common identifiers (ICES cell and vessel length). The resulting combined dataset allowed disaggregation of FDI-AER data by individual vessel, computed by dividing total values by the number of vessels in each vessel length and ICES cell. This enabled spatially explicit estimates of GVL at the vessel level.



Procedure scheme

## 1 Protocol - FDI effort by ices cell

In this session, the methodology for disaggregating data will be systematically explained through the application in a specific area that is the GSA09 (Northern Tyrrhenian Sea).

**Firstly, to ensure proper data management, it is necessary to download and save the data in a folder specifically dedicated to this purpose.**

The data to be downloaded includes:

Data Input

| Data | Description |
| --- | --- |
| [FDI effort and landing](https://jeodpp.jrc.ec.europa.eu/ftp/jrc-opendata/FAD/fdi/FDI_spatial_data_EU28.zip) | Effort and landing data divided by year, country, GSA, gear, vessel length and ices-cell. The geographical reference, expressed as ices-cell longitude-latitude coordinates, is provided in the shapefiles. |
| [AER](https://stecf.jrc.ec.europa.eu/documents/d/stecf/stecf_24-07_eu-fleet-economic-and-transversal-data) | Economic indicators by year, country, GSA, gear, and vessel length. |
| [Fleet register](https://webgate.ec.europa.eu/fleet-europa/search_en) | Descriptive information on individual vessels: vessel name, MMSI identifier, vessel length, port of registration, tonnage, power, gear, etc. |
| [EMODNET main ports for the European Seas](https://emodnet.ec.europa.eu/geonetwork/srv/eng/catalog.search#/metadata/379d0425-8924-4a41-a088-1a002d2ea748) | Main ports’ locations data from 1997 to 2024 |
| [FAO ASFIS List of Species for Fisheries](https://www.fao.org/fishery/static/ASFIS/ASFIS_sp.zip) | The ASFIS (Aquatic Sciences and Fisheries Information System) list for fishery statistics represents the standard taxonomic reference system for the FAO Statistics Team. |
| [FAO Geographical Sub-Areas](https://gfcmsitestorage.blob.core.windows.net/website/5.Data/ArcGIS/GSAs_simplified_updated_division%20(2).zip) | FAO GFCM area of application, comprised of the Mediterranean and the Black Sea, as Major Fishing Area 37. |

***Save the data to a folder and set the folder as the data location in the R environment:***

wd = "SET YOUR DATA FOLDER DIRECTORY"

library(curl)  
library(dplyr)  
library(doBy)  
library(ggplot2)  
library(ggrepel)  
library(ggridges)  
library(ggspatial)  
library(gfwr)  
library(gridExtra)  
library(gtsummary)  
library(leaflet)  
library(openxlsx)  
library(patchwork)  
library(RColorBrewer)  
library(reshape2)  
library(rnaturalearth)  
library(rnaturalearthdata)  
library(sf)  
library(tidyverse)  
library(tidytext)  
library(terra)  
library(VennDiagram)  
library(webr)  
library(webshot2)

## Data manipulation for a case study area

Users could establish parameters for their case study, which will subsequently inform the procedure.

Here we test Italian Bottom Otter Trawlers (ITA-OTB) in 2021 for GSA09.

CS\_name = "FAO GSA09 - Western Med"  
Gear\_CS = "OTB"  
Year\_CS = "2021"  
Country\_CS = "ITA"  
Country\_code = "IT"  
GSAs\_CS = "GSA09"  
GSAa\_CS = "GSA9"

### Step 1 - Open and subset FDI data Effort

Open and subset data from FDI by: *Gear type*, *Year*, and *Country*

effort = read.csv(paste0(wd,"FDI\_spatial\_data\_EU28/EU28/spatial\_effort\_tableau\_pts\_EU28.csv"))   
effort = effort %>%   
 filter(year %in% Year\_CS & gear\_type %in% Gear\_CS & cscode != "") %>%   
 mutate(totfishdays = as.numeric(totfishdays))

Subsequently, spatial ICES cells were used to map the total effort and landing data.

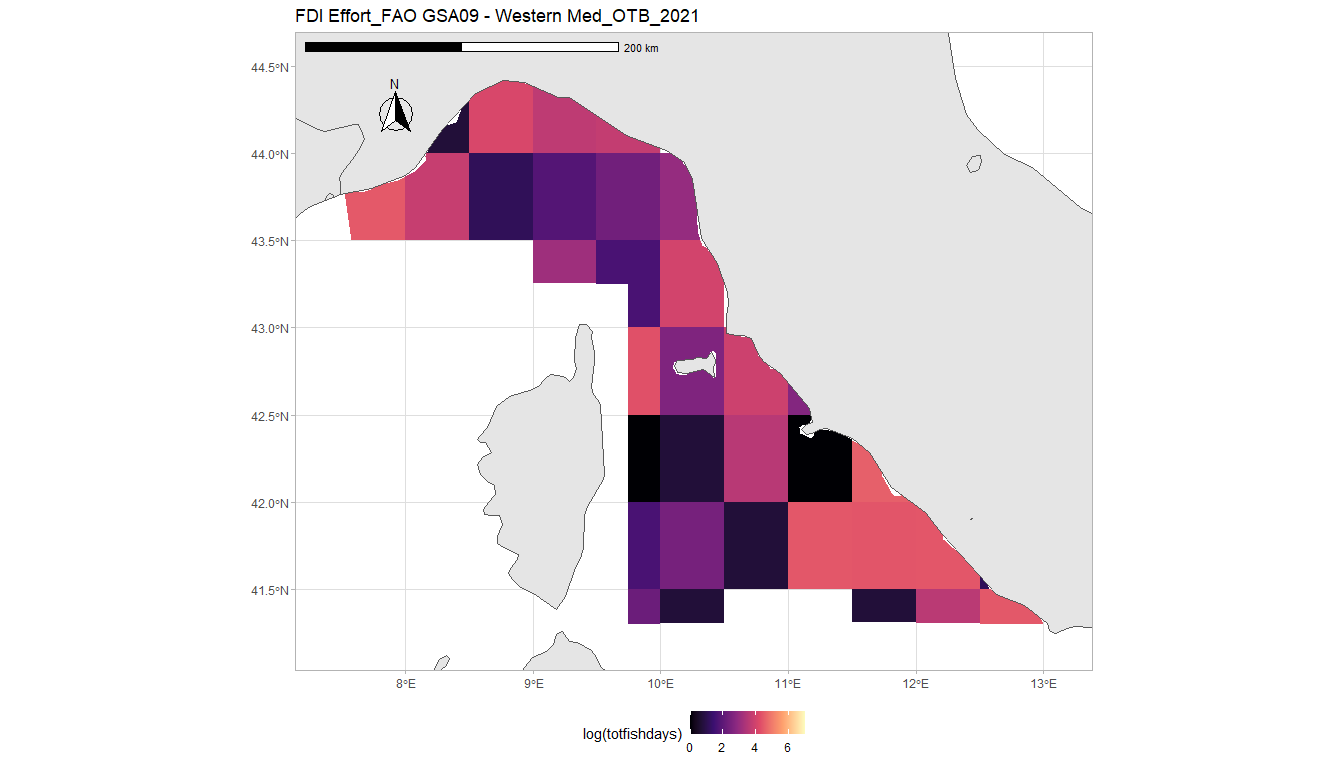
spatial\_effort = read\_sf(paste0(wd,"FDI\_spatial\_data\_EU28/EU28/effort\_csquares.shp"))  
spatial\_effort = spatial\_effort %>%   
 filter(cscode != is.na(cscode) & cscode != "")  
  
#Join data   
  
effort\_sf = st\_as\_sf(left\_join(effort, spatial\_effort, by = "cscode"))

Then the GSA polygon was used to subset and plot the landing and effort on the map.

# Effort and landing by GSA  
  
GSA = read\_sf(paste0(wd,"GSAs\_simplified.shp")) %>%  
 filter(SECT\_COD == GSAs\_CS)  
  
effort\_GSA = effort\_sf %>%   
 filter(sub\_region == GSAa\_CS)  
   
effort\_GSA = st\_intersection(effort\_GSA, GSA)  
  
  
effort\_sf = effort\_GSA  
  
CS = GSA  
  
#Set parameter for the map  
  
world <- ne\_countries(scale = "medium", returnclass = "sf", continent = "europe")  
world = st\_transform(world, crs = st\_crs(CS))  
  
xmin = as.numeric(st\_bbox(effort\_sf)[1])-0.1  
xmax = as.numeric(st\_bbox(effort\_sf)[3])+0.1  
ymin = as.numeric(st\_bbox(effort\_sf)[2])-0.1  
ymax = as.numeric(st\_bbox(effort\_sf)[4])+0.1

#### Total effort coverage for the case study area - resulting from FDI data

eff = ggplot()+  
 geom\_sf(data = effort\_sf, aes(fill = log(totfishdays)), color = NA)+  
 scale\_fill\_viridis\_c(option = "A", na.value = "white")+   
 geom\_sf(data = world)+  
 coord\_sf(xlim = c(xmin, xmax), ylim = c(ymin, ymax))+  
 annotation\_scale(location = "tl", width\_hint = 0.5) +  
 annotation\_north\_arrow(location = "tl", which\_north = "true",   
 pad\_x = unit(0.75, "in"), pad\_y = unit(0.5, "in"),  
 style = north\_arrow\_fancy\_orienteering) +  
 ggtitle(paste0("FDI Effort\_",CS\_name,"\_",Gear\_CS,"\_",Year\_CS))+  
 theme\_light()+  
 theme(legend.position = "bottom")  
  
print(eff)

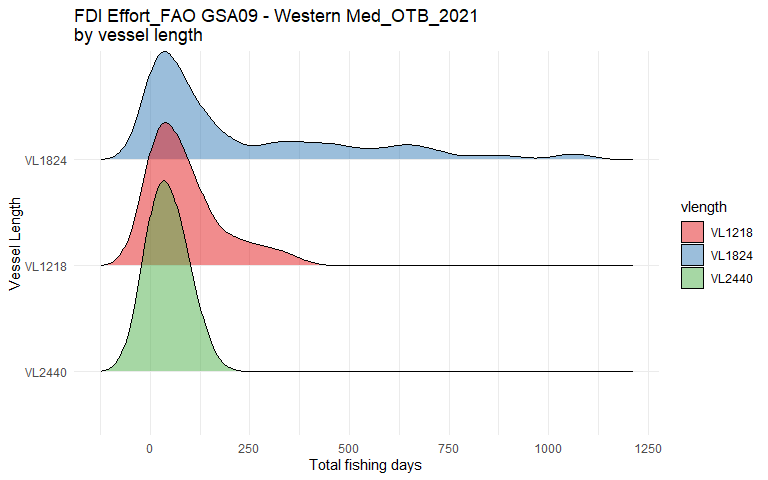


***Save all the filtered data in a specific folder fd = “CaseStudy/Data/”***

fd = "CaseStudy/Data/"  
  
write\_sf(effort\_sf, paste0(fd,"FDI\_effort\_CS.shp"))  
  
write.csv(  
 st\_drop\_geometry(effort\_sf) %>%  
 rename(id = cscode, gear = gear\_type, vlength = vessel\_length, tot\_fish\_day = totfishdays), paste0(fd,"FDI\_effort\_CS.csv"), row.names = F)  
  
remove(effort)

#### Descriptive analysis Effort

FDI\_effort\_CS = read.csv(paste0(fd,"FDI\_effort\_CS.csv")) %>% mutate(quarter = as.character(quarter))  
  
ggplot()+  
 geom\_density\_ridges(data = FDI\_effort\_CS, aes(y = fct\_reorder(vlength,tot\_fish\_day), x = tot\_fish\_day, fill = vlength),alpha = 0.5)+  
 theme\_minimal()+  
 scale\_fill\_brewer(palette = "Set1")+  
  
 ylab("Vessel Length")+  
 xlab("Total fishing days")+  
 ggtitle(paste0("FDI Effort\_",CS\_name,"\_",Gear\_CS,"\_",Year\_CS, "\nby vessel length"))



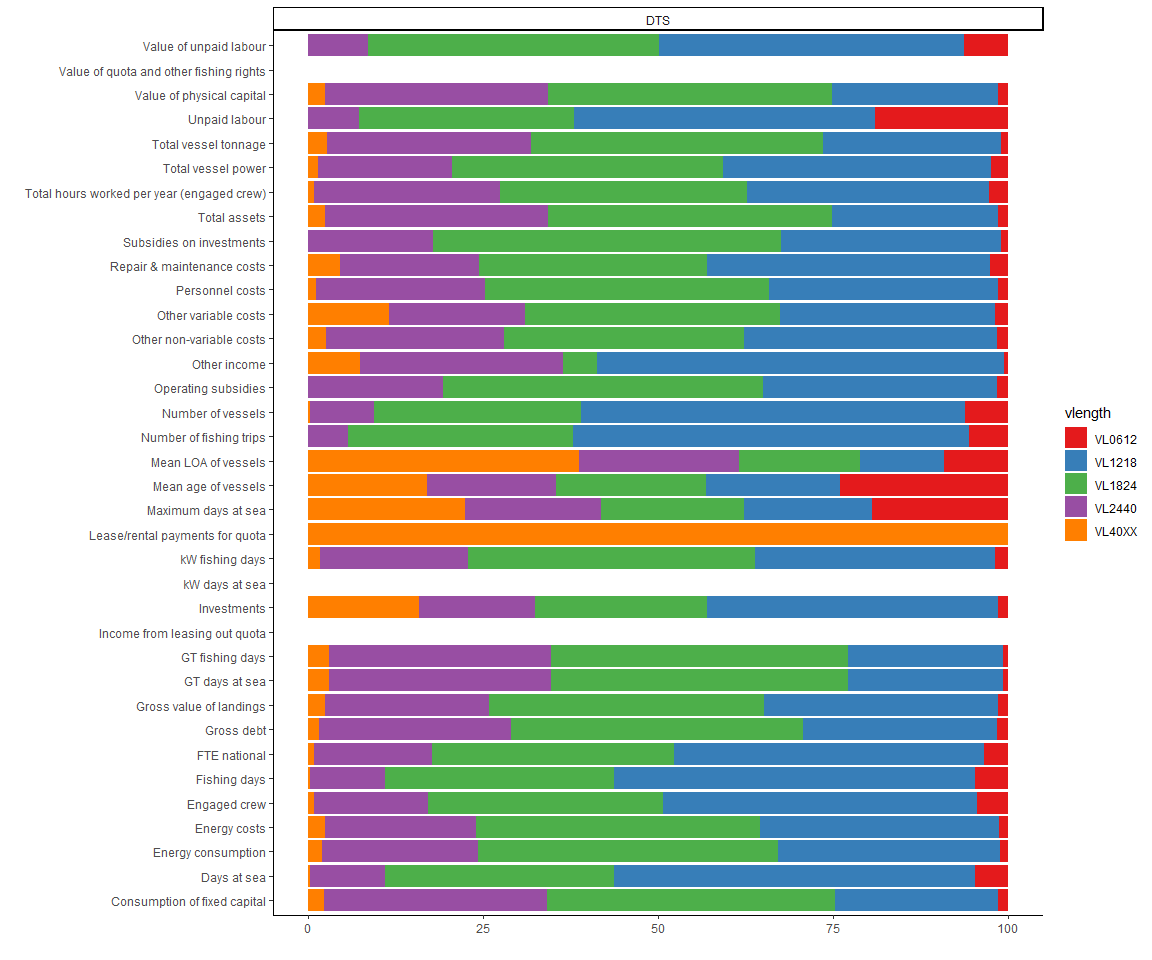
FDI\_CS\_data = FDI\_effort\_CS

### Step 2 - Open and subset AER data

AER\_FS = read.xlsx(paste0(wd,"STECF\_24-07\_EU Fleet Economic and Transversal data/STECF 24-07 - EU Fleet Economic and Transversal data\_fleet segment level.xlsx"), sheet = 2) %>%   
 filter(year %in% Year\_CS & fishing\_tech %in% c("DTS") & country\_code %in% Country\_CS)   
  
write.csv(AER\_FS, paste0(fd,"Economic\_data.csv"), row.names = F)

#### Descriptive analysis AER data

AER\_CS = read.csv(paste0(fd,"Economic\_data.csv"))  
  
AER\_CS = AER\_CS %>%   
 rename(vlength = vessel\_length) %>%   
 select(c(country\_code, year, fishing\_tech, vlength, variable\_group, variable\_name, variable\_code, value, unit ))  
  
data\_sum = AER\_CS %>%   
 group\_by(country\_code, fishing\_tech,vlength, variable\_group, variable\_name, unit) %>%   
 summarise(val = sum(value,na.rm = T)) %>%   
 rename(gear = fishing\_tech)  
  
 data\_sum %>%   
 group\_by(variable\_group, variable\_name) %>%   
 mutate(val\_prop = val/sum(val)\*100) %>%   
 ggplot()+  
 geom\_bar(aes(y = variable\_name, x= val\_prop, fill = vlength), stat = "identity")+  
 facet\_wrap(~ variable\_group, scales = "free")+   
 scale\_fill\_brewer(palette = "Set1")+  
 xlab("")+  
 ylab("")+  
 facet\_wrap(~ gear)+  
 theme\_classic()



AER\_wide = AER\_CS %>%   
 select(-c(variable\_code, unit, variable\_group)) %>%   
 dcast(...~ variable\_name, value.var = "value")  
  
 write.csv(AER\_wide, paste0(fd,"Economic\_data\_wide.csv"), row.names = F)

### Step 3 - Join AER with - FDI Effort-Landing

FDI\_CS\_data = read.csv(paste0(fd,"FDI\_effort\_CS.csv")) %>% mutate(quarter = as.character(quarter))  
AER\_CS = read.csv(paste0(fd,"Economic\_data.csv")) %>% rename(vlength = vessel\_length)  
  
  
FDI\_sub = unique(FDI\_CS\_data[,c("year","quarter", "vlength", "id", "tot\_fish\_day" )])  
AER\_sub = AER\_wide[,c("year", "vlength", "Fishing days", "Days at sea")]

### Step 4 - Find vessels track by Global Fishing Watch (GFW)

#### Extrapolate data

In this step, we will identify all vessels present in the CS area in a defined moment (here, we use the year 2021 as an example). The vessels were extrapolated from the GFW dataset, which uses AIS data to identify vessel tracks, fishing areas, and zones of navigation. Furthermore, it has the capacity to identify the ports visited by individual vessels. For more datails see <https://globalfishingwatch.org/our-apis/>

The use of gfwr requires a **GFW API token**, which users can request from the GFW API Portal. ***Save this token to your .Renviron file using usethis::edit\_r\_environ() and adding a variable named GFW\_TOKEN to the file (GFW\_TOKEN=“PASTE\_YOUR\_TOKEN\_HERE”). Save the .Renviron file and restart the R session to make the edit effective.***

gfwr functions are set to use key = gfw\_auth() by default so in general you shouldn’t need to refer to the key in your function calls.

key = gfw\_auth()

CS\_polygon <- sf::st\_bbox(c(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax),  
 crs = 4326) |>  
 sf::st\_as\_sfc() |>  
 sf::st\_as\_sf()  
  
GFW\_effort = get\_raster(spatial\_resolution = 'LOW',  
 temporal\_resolution = 'MONTHLY',  
 group\_by = 'VESSEL\_ID',  
 start\_date = "2021-01-01",  
 end\_date = "2021-12-31",  
 region = CS\_polygon,  
 region\_source = 'USER\_SHAPEFILE',  
 key = key)  
  
colnames(GFW\_effort) = make.names(colnames(GFW\_effort))  
  
GFW\_effort %>%   
 group\_by(Flag,Gear.Type) %>%   
 summarise(h = sum(Apparent.Fishing.Hours)) %>%   
 ggplot()+  
 geom\_bar(aes(x = h, y = reorder(Gear.Type, h), fill = Flag), stat = "identity")+  
 ggtitle("GFW data from CS polygon")+  
 xlab("Fishing hours")+  
 ylab("Gear type")+  
 theme\_light()  
  
write.csv(GFW\_effort, paste0(fd,"GFW\_effort\_tot\_CS.csv"))

#### Subset by Country, Gear, and CS Area

FDI\_effort\_CS\_sf = read\_sf(paste0(fd,"FDI\_effort\_CS.shp"))  
GFW\_effort = read.csv(paste0(fd,"GFW\_effort\_tot\_CS.csv"))  
  
GFW\_effort\_CS\_sf = GFW\_effort %>%  
 filter(Flag == "ITA" & Gear.Type %in% c("TRAWLERS")) %>%   
 st\_as\_sf(coords = c("Lon", "Lat"), crs = 4326)   
  
GFW\_effort\_CS\_sf$month = as.integer(substr(GFW\_effort\_CS\_sf$Time.Range, 6,7))  
GFW\_effort\_CS\_sf$quarter = as.character(c(1,2,3,4)[findInterval(GFW\_effort\_CS\_sf$month, c(1,3,6,9,13))])

Performs spatial joins quarter-by-quarter to ensure temporal alignment

FDI\_effort\_CS\_sf\_by\_quarter = FDI\_effort\_CS\_sf %>%  
 group\_by(quarter,cscode, ger\_typ) %>%  
 summarise(FDI\_tot\_fish\_day\_by\_ICES = sum(ttfshdy))  
  
quarter = c("1","2","3","4")  
  
GFW\_effort\_CS\_sf\_grid <- NULL   
for(i in 1:length(quarter)) {  
 a <- st\_join(  
 GFW\_effort\_CS\_sf[which(GFW\_effort\_CS\_sf$quarter %in% quarter[i]), ],  
 FDI\_effort\_CS\_sf\_by\_quarter[which(FDI\_effort\_CS\_sf\_by\_quarter$quarter %in% quarter[i]), "cscode"],  
 left = T  
 )  
  
 GFW\_effort\_CS\_sf\_grid <- rbind(GFW\_effort\_CS\_sf\_grid, a)  
}  
   
GFW\_effort\_CS\_sf\_grid = GFW\_effort\_CS\_sf\_grid %>%   
 filter(!is.na(cscode)) %>%   
 rename(id = cscode)   
  
  
write\_sf(GFW\_effort\_CS\_sf\_grid, paste0(fd,"GFW\_effort\_CS\_sf\_grid.shp"))  
write.csv(  
 st\_drop\_geometry(GFW\_effort\_CS\_sf\_grid),paste0(fd,"GFW\_effort\_CS\_sf\_grid.csv"), row.names = F)

### Step 5 - Find port visited by Global Fishing Watch (GFW)

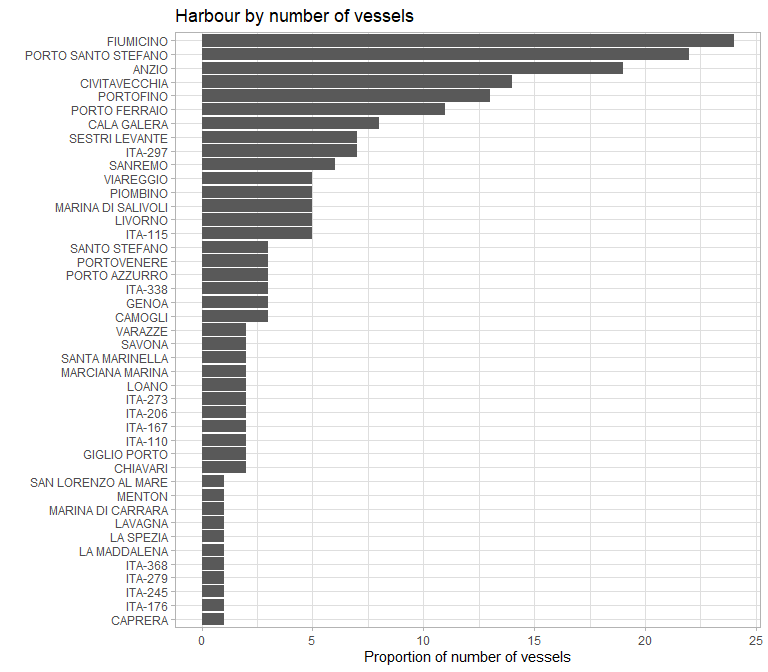
Download port visited by GFW database Only for CS vessels.

**Please note that this phase can take a long time.**

vID = unique(GFW\_effort\_CS\_sf\_grid$Vessel.ID)  
  
  
  
# Initialize port\_FV with correct column types  
port\_FV <- data.frame(  
 port = character(),   
 lat = numeric(),   
 lon = numeric(),   
 vessel\_name = character(),  
 MMSI = character(),  
 month = character(),  
 stringsAsFactors = FALSE  
)  
  
for (i in 1:length(vID)) {  
   
 port\_event <- get\_event(  
 event\_type = 'PORT\_VISIT',  
 start\_date = "2021-01-01",  
 end\_date = "2021-12-31",  
 region = CS\_polygon,  
 vessels = vID[i],  
 region\_source= 'USER\_SHAPEFILE',  
 key = key  
 )  
   
 if (is.null(port\_event)) next  
   
 for (j in 1:nrow(port\_event)) {  
   
 # Extract values safely, replacing NULL with NA  
 port\_name <- port\_event$event\_info[[j]]$startAnchorage$name  
 lat <- port\_event$event\_info[[j]]$startAnchorage$lat  
 lon <- port\_event$event\_info[[j]]$startAnchorage$lon  
 vessel\_name <- port\_event$vessel\_name  
 MMSI <- port\_event$vessel\_ssvid  
 month <- as.character(month(port\_event$start))  
   
  
 # Create the data frame with NULL-safe values  
 port\_event\_df <- data.frame(  
 port = ifelse(length(port\_name) == 0, NA, port\_name),  
 lat = ifelse(length(lat) == 0, NA, lat),  
 lon = ifelse(length(lon) == 0, NA, lon),  
 vessel\_name = ifelse(length(vessel\_name) == 0, NA, vessel\_name),  
 MMSI = ifelse(length(MMSI) == 0, NA, MMSI),  
 month = ifelse(length(month) == 0, NA, month),  
  
 stringsAsFactors = FALSE  
 )  
   
 # Append the row to the result dataframe  
 port\_FV <- bind\_rows(port\_FV, port\_event\_df)  
 }  
}  
  
# Remove duplicates and drop rows with NA values  
port\_CS\_OTB <- port\_FV %>%   
 unique() %>%   
 drop\_na() %>%   
 mutate(quarter = case\_when(  
 month %in% c("1", "2", "3") ~ "1",  
 month %in% c("4", "5", "6") ~ "2",  
 month %in% c("7", "8", "9") ~ "3",  
 month %in% c("10", "11", "12") ~ "4" )) %>%   
 group\_by(port, vessel\_name, MMSI, quarter) %>%   
 summarise(lat = mean(lat), lon = mean(lon))  
  
  
  
write.xlsx(port\_CS\_OTB, paste0(fd,"GFW\_port\_CS.xlsx"))

#### Descriptive analysis

GFW\_port\_CS = read.xlsx(paste0(fd,"GFW\_port\_CS.xlsx"))  
  
GFW\_port\_CS %>%   
 group\_by(port) %>%  
 summarise(  
 lon = mean(lon, na.rm = TRUE),  
 lat = mean(lat, na.rm = TRUE),  
 nvessel = n()) %>%   
   
 ggplot()+  
 geom\_bar(aes(y = reorder(port,nvessel) , x = nvessel), stat = "identity")+  
 theme\_light()+  
 ggtitle("Harbour by number of vessels")+  
 xlab("Proportion of number of vessels")+  
 ylab("")



#### Open Fleet Register and Add Vessel length (LOA) by MMSI - Vessel name for GFW data

Take in effort only MMSI of the fleet register: we take only vessels present also in the Fleet Register, and we add the vessel length.

fleetReg = read.csv(paste0(wd,"vesselRegistryListResults.csv"), sep = ";")  
  
fleetReg[fleetReg$Main.fishing.gear %in% c("TBN","OTS", "TBS", "OT"), "Main.fishing.gear"] <- "OTB"  
fleetReg[fleetReg$Main.fishing.gear %in% c("SV","SX"), "Main.fishing.gear"] <- "SDN"  
fleetReg[fleetReg$Main.fishing.gear %in% c("DRM", "DRH") , "Main.fishing.gear"] <- "DRB"  
fleetReg[fleetReg$Main.fishing.gear %in% c("GTN","GNC", "GN", "FIX"), "Main.fishing.gear"] <- "GNS"  
fleetReg[fleetReg$Main.fishing.gear %in% "SPR", "Main.fishing.gear"] <- "SSC"  
fleetReg[fleetReg$Main.fishing.gear %in% c("SB", "NK"), "Main.fishing.gear"] <- "MIS"  
fleetReg[fleetReg$Main.fishing.gear %in% c("LL", "LX"), "Main.fishing.gear"] <- "LLS"  
  
  
fleetReg\_info = fleetReg %>%   
 rename(vessel\_name = "Name.of.vessel", Gear = "Main.fishing.gear", Country ="Country.of.Registration") %>%   
 mutate(MMSI = as.character(MMSI)) %>%   
 filter(Country %in% Country\_CS) %>%   
 filter(Gear %in% Gear\_CS)   
  
  
fleetReg\_info$vlength = c("VL0006","VL0612","VL1218", "VL1824", "VL2440", "VL40XX" )[findInterval(fleetReg\_info$LOA, c(0,06,12,18,24,40, 100))]  
  
write.csv(fleetReg\_info, paste0(fd,"FleetReg\_info\_CS.csv"), row.names = F)

We lost ~ 50 vessels because they do not have MMSI associated to the fleet register

fleetReg\_info = read.csv(paste0(fd,"FleetReg\_info\_CS.csv")) %>%   
 select(vessel\_name, MMSI, Gear, vlength) %>%   
 mutate(MMSI = as.character(MMSI)) %>%   
 rename(name\_vreg = vessel\_name)  
  
GFW\_port\_CS\_fReg = GFW\_port\_CS %>%   
 left\_join(fleetReg\_info, by = "MMSI") %>%   
 filter(MMSI %in% fleetReg\_info$MMSI)  
   
  
GFW\_effort\_CS\_sf\_grid\_fReg = read.csv(paste0(fd,"GFW\_effort\_CS\_sf\_grid.csv")) %>%   
 mutate(MMSI = as.character(MMSI)) %>%   
 left\_join(fleetReg\_info, by = "MMSI") %>%   
 filter(MMSI %in% unique(fleetReg\_info$MMSI))  
  
  
write.csv(GFW\_effort\_CS\_sf\_grid\_fReg, paste0(fd,"GFW\_effort\_CS\_sf\_grid\_fReg.csv"))  
write.csv(GFW\_port\_CS\_fReg, paste0(fd,"GFW\_port\_CS\_fReg.csv"), row.names = F)

#### Check for main ports

We filtered for main ports resulted fro the: Main Ports (Vessels Traffic by Type 1997-2024)

Since the two datasets are not perfectly comparable, we first identify all the GFW ports that are also present in the EMODNET dataset by performing a join on the port name. Then, a buffer of 3 km is created around the EMODNET ports, and the GFW ports within that buffer are assigned the same name as the EMODNET ports.

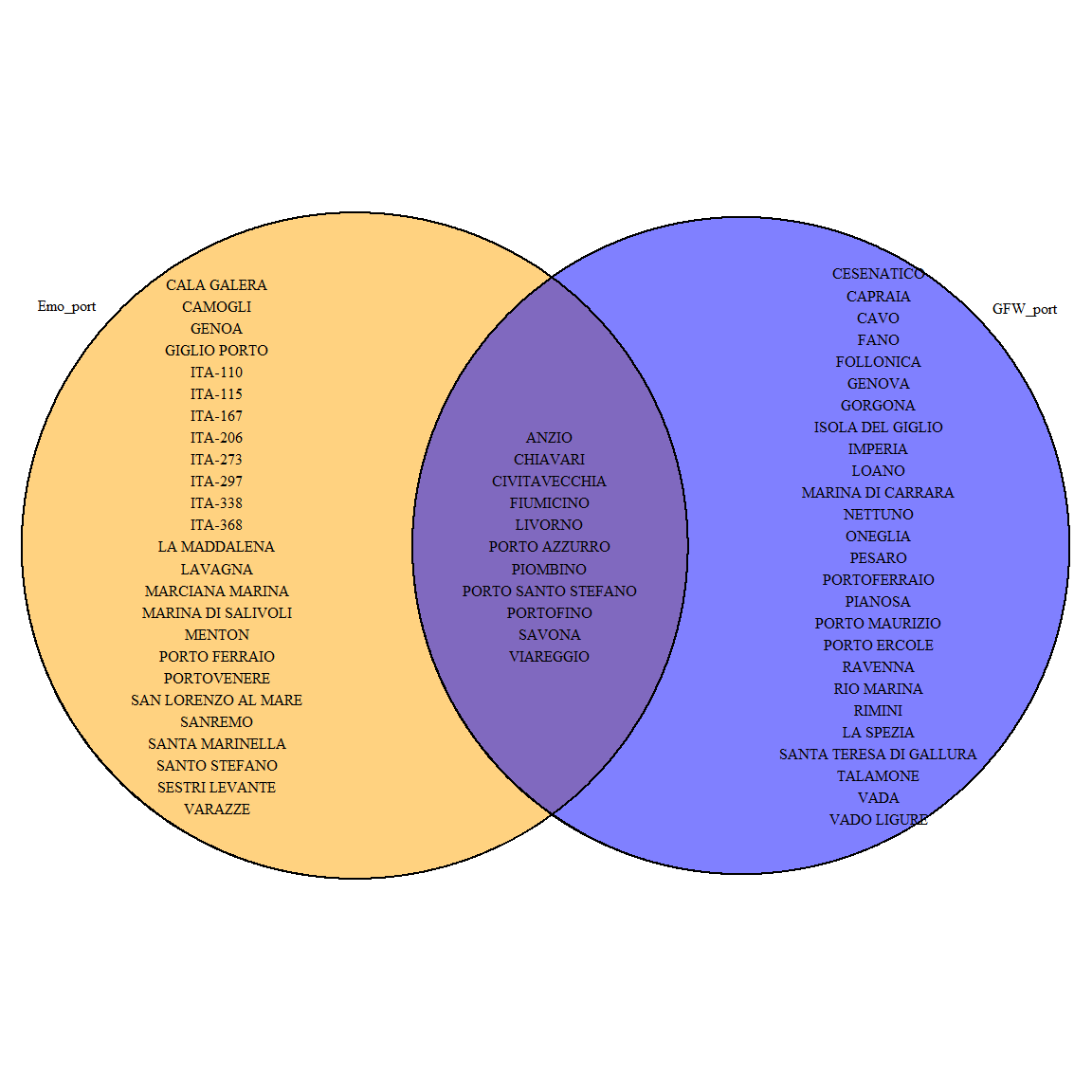


During this phase, it is essential to consult a Case Study expert who will be able to manually modify the port name in the GFW dataset, where feasible.

CS\_polygon <- sf::st\_bbox(c(xmin = xmin, xmax = xmax, ymin = ymin, ymax = ymax),  
 crs = 4326) |>  
 sf::st\_as\_sfc() |>  
 sf::st\_as\_sf()  
  
GFW\_port\_CS\_fReg = read.csv(paste0(fd,"GFW\_port\_CS\_fReg.csv"))  
  
EMODNET\_port\_sf = read\_sf(paste0(wd,"EMODnet\_HA\_MainPorts\_Traffic\_20241112/EMODnet\_HA\_MainPorts\_Ports2025\_20241112.shp")) %>%   
 filter(CNTR\_CODE %in% Country\_code) %>%   
 st\_intersection(CS\_polygon)  
  
  
GFW\_port\_CS = GFW\_port\_CS\_fReg   
   
  
GFW\_port\_sf = GFW\_port\_CS %>% st\_as\_sf(coords = c("lon","lat"), crs = st\_crs(EMODNET\_port\_sf))  
  
EMODNET\_port\_sf$PORT\_NAME = toupper(EMODNET\_port\_sf$PORT\_NAME)  
  
Emo\_port = EMODNET\_port\_sf$PORT\_NAME  
GFW\_port = unique(GFW\_port\_sf$port)

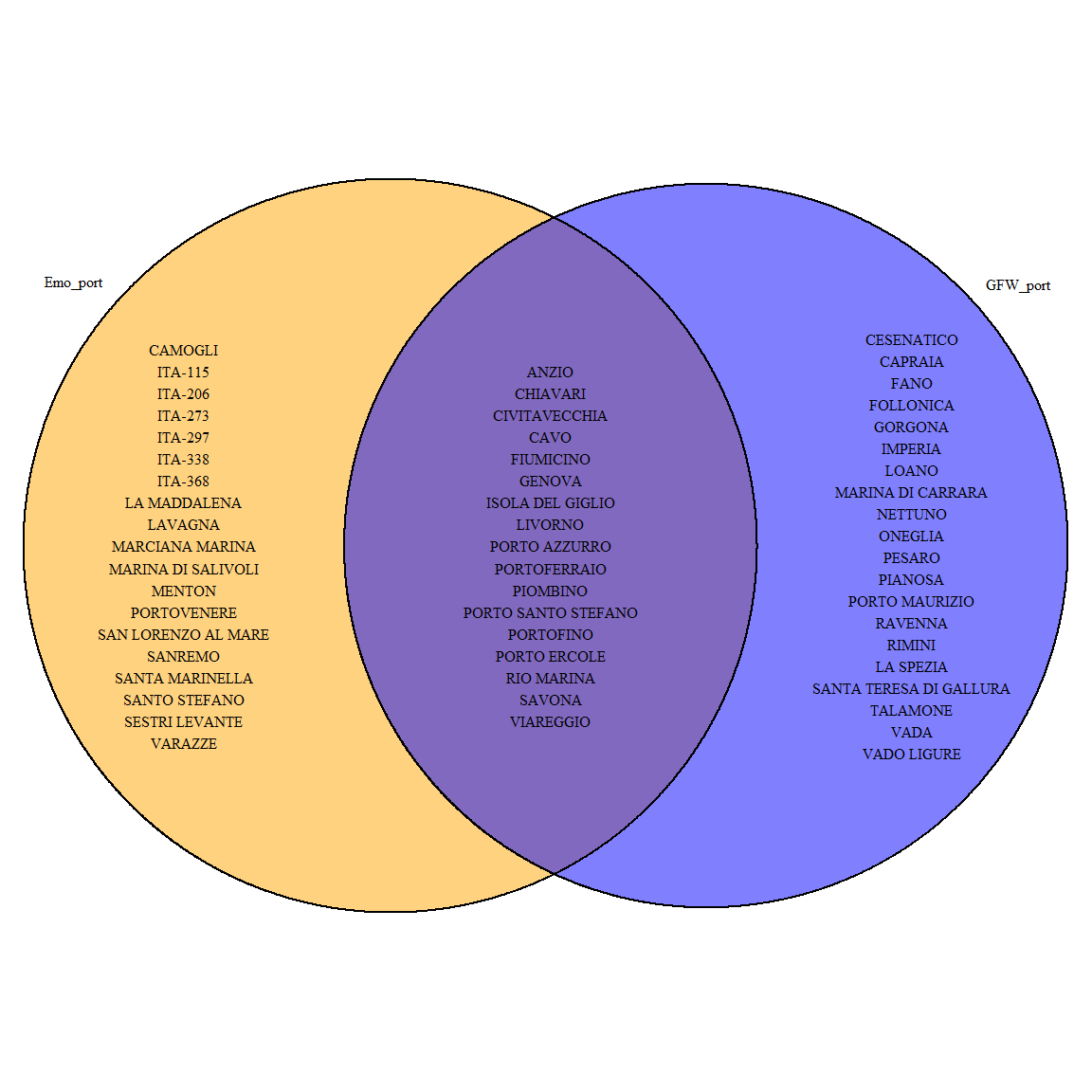
Analysed the common port names between the Emodnet and GFW datasets and change name when is possible.

# Generate initial Venn diagram  
v <- venn.diagram(  
 list(Emo\_port = Emo\_port, GFW\_port = GFW\_port),  
 fill = c("orange", "blue"),  
 alpha = c(0.5, 0.5),   
 cat.cex = 1.0,   
 cex = 1.0,  
 filename = NULL  
)  
  
# Inspect original labels (optional, can be commented out)  
# lapply(v, function(i) i$label)  
  
# Customize labels - these indices (5-7) may need adjustment based on your data  
v[[5]]$label <- paste(setdiff(GFW\_port, Emo\_port), collapse = "\n") # GFW\_port only  
v[[6]]$label <- paste(setdiff(Emo\_port, GFW\_port), collapse = "\n") # Emo\_port only  
v[[7]]$label <- paste(intersect(Emo\_port, GFW\_port), collapse = "\n") # Intersection  
  
# Render the plot  
grid.newpage()  
grid.draw(v)



GFW\_port\_sf = GFW\_port\_sf %>%   
 mutate(port = case\_when(  
 port == "CALA GALERA" ~ "PORTO ERCOLE",  
 port == "GENOA" ~ "GENOVA",  
 port == "GIGLIO PORTO" ~ "ISOLA DEL GIGLIO",  
 port == "PORTO FERRAIO" ~ "PORTOFERRAIO",  
 TRUE ~ port # Keep other values unchanged  
 ))  
  
GFW\_port = unique(GFW\_port\_sf$port)  
  
Portdiff = setdiff(Emo\_port, GFW\_port)  
  
  
  
EMODNET\_port\_buffer\_GFW = EMODNET\_port\_sf %>%   
 filter(PORT\_NAME %in% Portdiff) %>%   
 st\_buffer(dist = 3000) %>%   
 st\_intersection(GFW\_port\_sf)  
   
# EMODNET\_port\_buffer\_GFW   
###inspect manually   
  
  
GFW\_port\_sf = GFW\_port\_sf %>%   
 mutate(port = case\_when(  
 port == "ITA-110" ~ "RIO MARINA",  
 port == "ITA-167" ~ "CAVO",  
 TRUE ~ port ))  
  
  
Emo\_port = EMODNET\_port\_sf$PORT\_NAME  
GFW\_port = unique(GFW\_port\_sf$port)

# Generate plot  
v <- venn.diagram(list(Emo\_port = Emo\_port, GFW\_port = GFW\_port),  
 fill = c("orange", "blue"),  
 alpha = c(0.5, 0.5), cat.cex = 1.0, cex=1.0,  
 filename=NULL)  
  
# lapply(v, function(i) i$label)  
  
  
v[[5]]$label <- paste(setdiff(GFW\_port, Emo\_port), collapse="\n")   
# in baa only  
v[[6]]$label <- paste(setdiff(Emo\_port, GFW\_port) , collapse="\n")   
# intesection  
v[[7]]$label <- paste(intersect(Emo\_port, GFW\_port), collapse="\n")   
  
# plot   
grid.newpage()  
grid.draw(v)

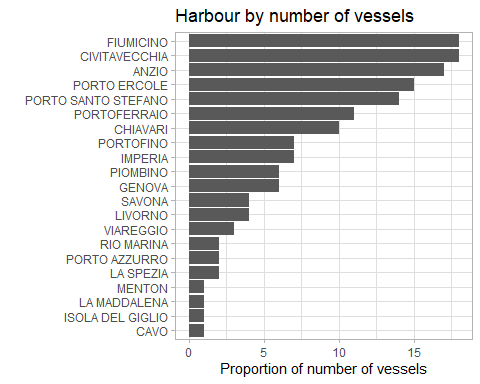


Finally, double-check the GFW ports without a unique identifier and assign the correct nearest port name to each. See the leaflet map as an example.

*GFW ports without ID are in red, while correct ports are in blue.*

final\_port\_name = intersect(Emo\_port, GFW\_port)  
  
xmin = as.numeric(st\_bbox(FDI\_effort\_CS\_sf\_by\_quarter)[1])-0.1  
xmax = as.numeric(st\_bbox(FDI\_effort\_CS\_sf\_by\_quarter)[3])+0.1  
ymin = as.numeric(st\_bbox(FDI\_effort\_CS\_sf\_by\_quarter)[2])-0.1  
ymax = as.numeric(st\_bbox(FDI\_effort\_CS\_sf\_by\_quarter)[4])+0.1  
  
 leaflet() %>%  
 addTiles() %>%  
 fitBounds(lng1 = xmin, lat1 = ymin, lng2 = xmax, lat2 = ymax) %>%  
   
 # All GFW ports (red)  
 addCircleMarkers(  
 data = GFW\_port\_sf,  
 radius = 5,  
 color = "red",  
 popup = ~ port  
 ) %>%  
   
 # Highlighted ports (blue)  
 addCircleMarkers(  
 data = GFW\_port\_sf[which(GFW\_port\_sf$port %in% final\_port\_name),],  
 radius = 5,  
 color = "blue",  
 popup = ~ port  
 ) %>%  
   
 # Final ports (green)  
 addMarkers(  
 data = EMODNET\_port\_sf,  
 popup = ~ PORT\_NAME  
 )

GFW\_port\_sf = GFW\_port\_sf %>%   
 mutate(port = case\_when(  
 port == "ITA-338" ~ "ANZIO",  
 port == "ITA-206" ~ "CIVITAVECCHIA",  
 port == "SANTA MARINELLA" ~ "CIVITAVECCHIA",  
 port == "ITA-368" ~ "CIVITAVECCHIA",  
 port == "ITA-297" ~ "PORTO ERCOLE",  
 port == "ITA-115" ~ "PORTOFERRAIO",  
 port == "MARCIANA MARINA" ~ "PORTOFERRAIO",  
 port == "MARINA DI SALIVOLI" ~ "PIOMBINO",  
 port == "PORTOVENERE" ~ "LA SPEZIA",  
 port == "LAVAGNA" ~ "CHIAVARI",  
 port == "SESTRI LEVANTE" ~ "CHIAVARI",  
 port == "CAMOGLI" ~ "GENOVA",  
 port == "ITA-273" ~ "GENOVA",  
 port == "VARAZZE" ~ "SAVONA",  
 port == "SAN LORENZO AL MARE" ~ "IMPERIA",  
 port == "SANTO STEFANO" ~ "IMPERIA",  
 port == "SANREMO" ~ "IMPERIA",  
 TRUE ~ port ))   
   
   
   
   
 GFW\_port = unique(GFW\_port\_sf$port)  
 final\_port\_name = intersect(Emo\_port, GFW\_port)  
   
 EMODNET\_final\_port = EMODNET\_port\_sf %>% select(PORT\_NAME) %>% filter(PORT\_NAME %in% final\_port\_name) %>% rename(port = PORT\_NAME)  
  
   
 GFW\_port\_sf = st\_drop\_geometry(GFW\_port\_sf) %>%   
 left\_join(EMODNET\_final\_port) %>%   
 st\_as\_sf()  
   
 GFW\_port\_sf %>%   
 mutate(n = 1) %>%   
 group\_by(port) %>%  
 summarise(nvessel = sum(n)) %>%  
 ggplot()+  
 geom\_bar(aes(y = reorder(port,nvessel) , x = nvessel), stat = "identity")+  
 # geom\_vline(xintercept = 10, color = "red")+  
 theme\_light()+  
 ggtitle("Harbour by number of vessels")+  
 xlab("Proportion of number of vessels")+  
 ylab("")



write\_sf(GFW\_port\_sf, paste0(fd,"GFW\_port\_CS\_fReg\_coords.shp"))   
write.csv(data.frame(st\_coordinates(GFW\_port\_sf), st\_drop\_geometry(GFW\_port\_sf)), row.names = F, paste0(fd,"GFW\_port\_CS\_fReg\_coords.csv"))

We have now obtained the number of vessels for the main ports, and we must link them to the effort. However, we are unable to retain individual boat information because some vessels are associated with many different ports. So we make last modification:

GFW\_port\_CS = read.csv(paste0(fd,"GFW\_port\_CS\_fReg\_coords.csv")) %>% mutate(MMSI = as.character(MMSI))  
  
fleetReg\_place\_reg = read.csv(paste0(fd,"FleetReg\_info\_CS.csv")) %>%   
 select(MMSI, Place.of.registration.name) %>%   
 mutate(MMSI = as.character(MMSI))  
  
GFW\_port\_fREG = GFW\_port\_CS %>% left\_join(fleetReg\_place\_reg)

GFW\_port\_fREG <- GFW\_port\_fREG %>%  
 group\_by(MMSI) %>%  
 mutate(  
 port\_count = n\_distinct(port),  
 match\_port = if\_else(port\_count > 1, Place.of.registration.name, port),  
 final\_port = if\_else(is.na(match\_port), port, match\_port )  
 ) %>%  
 mutate(final\_port = toupper(final\_port)) %>%   
 ungroup() %>%  
 select(-port\_count)   
  
setdiff(unique(GFW\_port\_fREG$final\_port), unique(GFW\_port\_fREG$port))

## [1] "SESTRI LEVANTE" "SANTA MARGHERITA LIGURE"  
## [3] "ROMA" "SAN REMO"

GFW\_port\_fREG <- GFW\_port\_fREG %>%  
 mutate(final\_port = case\_when(  
 final\_port == "ROMA" ~ "FIUMICINO",  
 final\_port == "SAN REMO" ~ "IMPERIA",  
 TRUE ~ final\_port ))  
   
 setdiff(unique(GFW\_port\_fREG$final\_port), unique(GFW\_port\_fREG$port))

## [1] "SESTRI LEVANTE" "SANTA MARGHERITA LIGURE"

GFW\_port\_fREG %>%   
 filter(final\_port %in% c("SESTRI LEVANTE" , "SANTA MARGHERITA LIGURE")) %>%   
 distinct(vessel\_name,final\_port)

## # A tibble: 4 × 2  
## vessel\_name final\_port   
## <chr> <chr>   
## 1 JAZZ SESTRI LEVANTE   
## 2 ARDITO SANTA MARGHERITA LIGURE  
## 3 SGIUSEPPE SESTRI LEVANTE   
## 4 TERESA MADRE SANTA MARGHERITA LIGURE

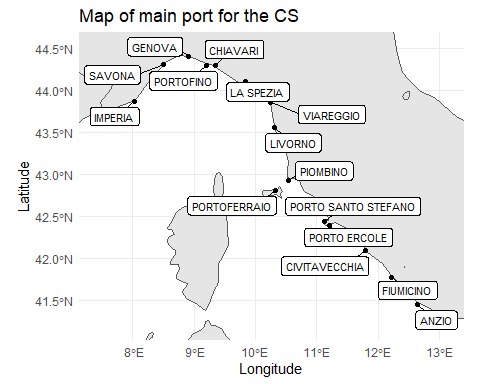
GFW\_port\_fREG <- GFW\_port\_fREG %>%  
 mutate(final\_port = case\_when(  
 final\_port == "SESTRI LEVANTE" & vessel\_name == "JAZZ" ~ "CHIAVARI",   
 final\_port == "SANTA MARGHERITA LIGURE" & vessel\_name == "ARDITO" ~ "CHIAVARI",  
 final\_port == "SANTA MARGHERITA LIGURE" & vessel\_name == "TERESA MADRE" ~ "CHIAVARI",  
 final\_port == "SESTRI LEVANTE" & vessel\_name == "SGIUSEPPE" ~ "CHIAVARI",  
 TRUE ~ final\_port ))  
   
 setdiff(unique(GFW\_port\_fREG$final\_port), unique(GFW\_port\_fREG$port))

## character(0)

unique(GFW\_port\_fREG$final\_port)

## [1] "ANZIO" "PORTO ERCOLE" "PORTO SANTO STEFANO"  
## [4] "CIVITAVECCHIA" "GENOVA" "CHIAVARI"   
## [7] "FIUMICINO" "PIOMBINO" "LIVORNO"   
## [10] "PORTOFERRAIO" "SAVONA" "IMPERIA"   
## [13] "PORTOFINO" "LA SPEZIA" "VIAREGGIO"

port\_coords = unique(GFW\_port\_fREG[c("X","Y","port")]) %>%   
 rename(lon\_port = X,  
 lat\_port = Y)  
  
GFW\_port\_fREG\_CS = GFW\_port\_fREG %>%   
 select(-c(X,Y,port,match\_port)) %>%   
 rename(port = final\_port) %>%   
 left\_join(port\_coords)  
   
  
GFW\_port\_fREG\_CS\_sf = GFW\_port\_fREG\_CS %>%  
 select(port,lon\_port, lat\_port) %>%   
 unique() %>%   
 st\_as\_sf(coords = c("lon\_port", "lat\_port"), crs = st\_crs(GSA))  
  
 ggplot() +  
 geom\_sf(data = world)+  
 geom\_sf(data = GFW\_port\_fREG\_CS\_sf)+  
 geom\_label\_repel(  
 data = GFW\_port\_fREG\_CS %>% select(port,lon\_port, lat\_port) %>%   
 unique() ,  
 aes(x = lon\_port, y = lat\_port, label = port),  
 size = 3,  
 min.segment.length = 0  
 ) +  
 coord\_sf(xlim = c(xmin, xmax), ylim = c(ymin, ymax))+  
 theme\_minimal()+  
 ggtitle("Map of main port for the CS")+  
 xlab("Longitude")+  
 ylab("Latitude")



write.csv(GFW\_port\_fREG\_CS, paste0(fd,"GFW\_port\_CS\_fReg\_coords.csv"), row.names = F)   
 GFW\_port\_fREG\_sf = st\_as\_sf(GFW\_port\_fREG\_CS, coords = c("lon\_port","lat\_port"), crs = 4326)  
   
 write\_sf(GFW\_port\_fREG\_sf, paste0(fd,"GFW\_port\_CS\_fReg\_coords.shp"))

#### Find the number of vessels by each port

Join vessels by port to find how many vessels are in each port

–> Remove quarter: The data is aggregated by year and the seasonal variation is removed because the AIS data in this case study does not have optimal resolution.

GFW\_port\_fREG = read.csv(paste0(fd,"GFW\_port\_CS\_fReg\_coords.csv")) %>%   
 select(MMSI, Gear, vlength, port, lon\_port, lat\_port) %>%   
 mutate(MMSI = as.character(MMSI)) %>%   
 unique()  
  
GFW\_effort\_CS\_sf = read\_sf(paste0(fd,"GFW\_effort\_CS\_sf\_grid.shp")) %>%   
 filter(MMSI %in% unique(GFW\_port\_fREG$MMSI)) %>%   
 dplyr::select(MMSI, App\_F\_H, id) %>%   
 mutate(MMSI = as.character(MMSI))  
  
  
GFW\_effort\_CS\_sf =GFW\_effort\_CS\_sf %>%   
 left\_join(GFW\_port\_fREG, by = "MMSI")  
  
write\_sf(GFW\_effort\_CS\_sf, paste0(fd,"GFW\_effort\_port\_by\_icell.shp"))

### Step 6 - Disaggregation process

#### AER disaggregation by ICES cell

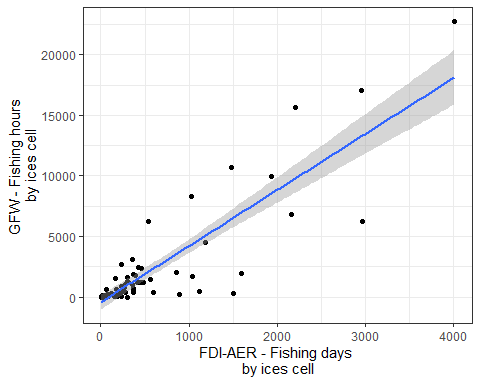
Starting from the data collected by NISEA, it is possible to fit a series of economic models.

Compute the disaggregated economic value for each ices cell:

AER = read.csv(paste0(fd,"Economic\_data\_wide.csv")) %>% rename(vssl\_ln = vlength) %>%   
 dplyr::select(c("vssl\_ln", "Fishing.days", "Energy.costs", "Gross.value.of.landings"))  
   
FDI\_effort\_CS\_sf = read\_sf(paste0(fd,"FDI\_effort\_CS.shp"))  
  
FDI\_id = FDI\_effort\_CS\_sf %>%   
 group\_by(cscode, ger\_typ, vssl\_ln ) %>%   
 summarise(day = sum(ttfshdy)) %>%   
 rename(id = cscode)  
  
FDI\_vl = FDI\_id %>%   
 group\_by(ger\_typ,vssl\_ln) %>%   
 mutate(tot\_day = sum(day)) %>%   
 st\_drop\_geometry() %>%   
 ungroup()  
  
FDI\_vl <- FDI\_vl %>%  
 group\_by(ger\_typ) %>%   
 mutate(effort\_share = day / tot\_day)  
  
FDI\_vl <- FDI\_vl %>%  
 left\_join(AER, by = c("vssl\_ln"))  
  
FDI\_vl <- FDI\_vl %>%  
 mutate(across(c(Fishing.days, Energy.costs, Gross.value.of.landings), ~ .x \* effort\_share, .names = "{.col}\_AER")) %>%   
 dplyr::select(-c(Fishing.days,Energy.costs,Gross.value.of.landings))  
  
  
write.csv(FDI\_vl, paste0(fd,"FDI\_AER\_by\_icell.csv"), row.names = F)

Add GFW data by gear and vessel length

GFW\_effort\_port\_by\_icell = read\_sf(paste0(fd,"GFW\_effort\_port\_by\_icell.shp")) %>%   
 rename(vssl\_ln = vlength)  
  
FDI\_AER\_by\_icell = read.csv(paste0(fd,"FDI\_AER\_by\_icell.csv"))  
  
  
  
GFW\_id = GFW\_effort\_port\_by\_icell %>%   
 group\_by(id, Gear, vssl\_ln) %>%   
 summarise(h = sum(App\_F\_H)) %>%   
 st\_drop\_geometry()  
  
FDI.AER\_id = FDI\_AER\_by\_icell %>%   
 group\_by(id, ger\_typ, vssl\_ln ) %>%   
 summarise(day = sum(Fishing.days\_AER))   
  
### check data effort of FDI and GFW  
  
FDI\_GFW = inner\_join(GFW\_id, FDI\_id)   
  
  
FDI\_GFW\_plot = ggplot(data = FDI\_GFW, aes(x = day, y = h))+  
 geom\_point()+  
 geom\_smooth(method = "lm")+  
 theme\_bw()+  
 xlab("FDI-AER - Fishing days \nby ices cell")+  
 ylab("GFW - Fishing hours \nby ices cell")  
  
FDI\_GFW\_plot



#### FDI effort - AER disaggregation by spatial ICES cells

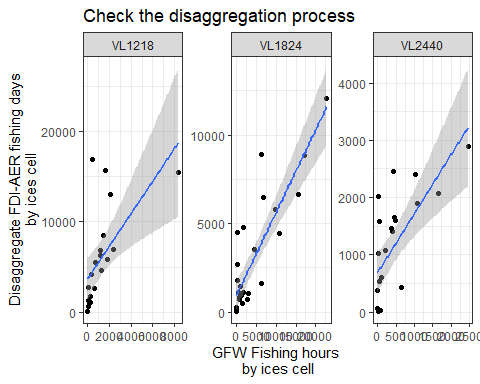
Since there is a linear relationship between GFW effort and FDI fishing days, we can simply divide the FDI data x cell x the number of vessels in a given cell

gfw\_df = GFW\_effort\_port\_by\_icell %>%   
 group\_by(id, vssl\_ln) %>%   
 mutate(n\_track = n()) %>%   
 mutate(n\_prop = n\_track \* sum(n\_track)) %>%   
 ungroup()  
  
gfw\_fdi\_merged\_df <- gfw\_df %>%  
 left\_join(FDI\_AER\_by\_icell, by = c("id", "vssl\_ln"))  
  
final\_df <- gfw\_fdi\_merged\_df %>%  
 mutate(across(c(Fishing.days\_AER, Energy.costs\_AER, Gross.value.of.landings\_AER), ~ .x / n\_track, .names = "{.col}\_by\_vessel")) %>%   
 dplyr::select(-c(Fishing.days\_AER,Energy.costs\_AER,Gross.value.of.landings\_AER))

#### Output

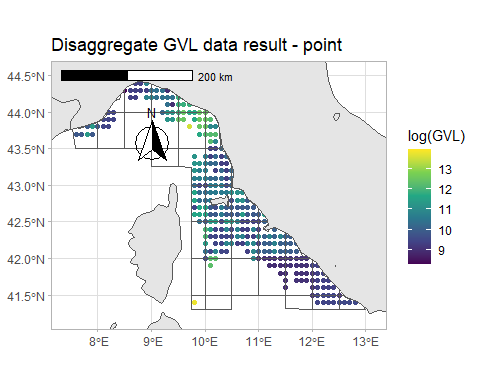
Check the disaggregation process

print(final\_df %>%   
 group\_by(id, vssl\_ln) %>%   
 summarise(GFW\_H = sum(App\_F\_H),FDI\_days = sum(Fishing.days\_AER\_by\_vessel)) %>%   
  
ggplot(aes(x = GFW\_H, y = FDI\_days))+  
 geom\_point()+  
 geom\_smooth(method = "lm")+  
 theme\_bw()+  
 facet\_wrap(~ vssl\_ln, scale = "free")+  
 xlab("GFW Fishing hours \nby ices cell")+  
 ylab("Disaggregate FDI-AER fishing days\nby ices cell")+  
 ggtitle("Check the disaggregation process")  
)

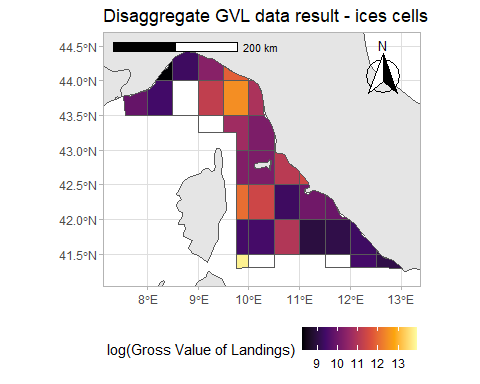


Map the Gross Value of Landings (GVL) resulting from the disaggregation process

map\_GVL = ggplot()+  
 geom\_sf(data = final\_df, aes(color = log(Gross.value.of.landings\_AER\_by\_vessel)))+  
 scale\_color\_viridis\_c("log(GVL)",option = "D", na.value = "white")+   
 geom\_sf(data = FDI\_id, fill = "NA")+  
 geom\_sf(data = world)+  
 coord\_sf(xlim = c(xmin, xmax), ylim = c(ymin, ymax))+  
 annotation\_scale(location = "tl", width\_hint = 0.5) +  
 annotation\_north\_arrow(location = "tl", which\_north = "true",   
 pad\_x = unit(0.75, "in"), pad\_y = unit(0.5, "in"),  
 style = north\_arrow\_fancy\_orienteering) +  
 ggtitle(paste0("Disaggregate GVL data result - point"))+  
 theme\_light()  
  
map\_GVL

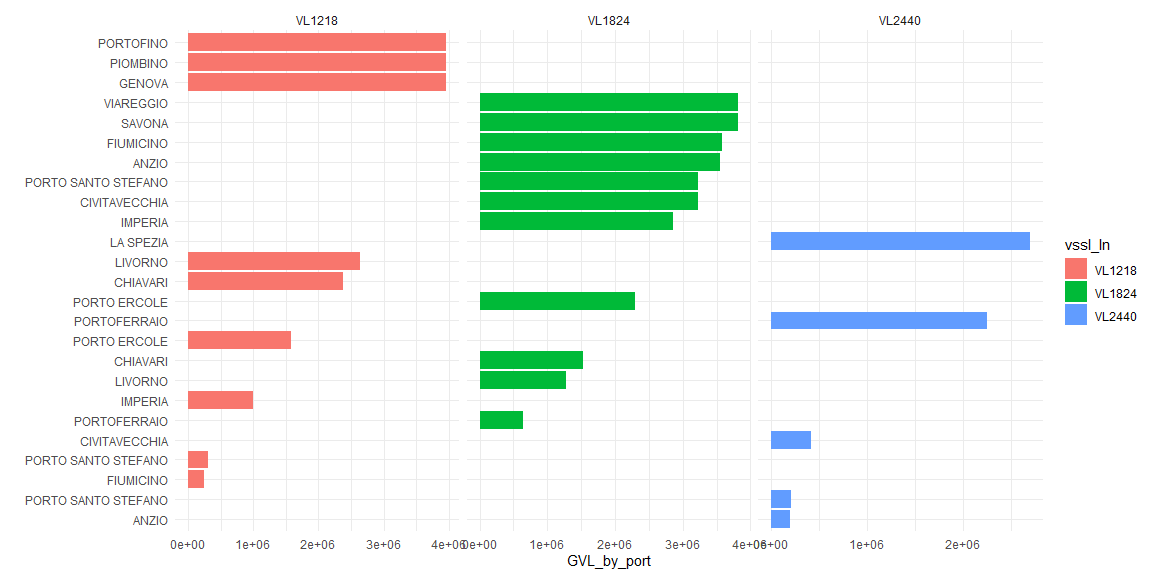


final\_df\_by\_ices = FDI\_id %>%  
 select(geometry) %>%   
 st\_join(final\_df) %>%   
 ggplot +   
 geom\_sf(aes(fill = log(Gross.value.of.landings\_AER\_by\_vessel)))+  
 geom\_sf(data = world)+  
 scale\_fill\_viridis\_c("log(Gross Value of Landings)",option = "inferno", na.value = "white")+   
 coord\_sf(xlim = c(xmin, xmax), ylim = c(ymin, ymax))+  
 annotation\_scale(location = "tl", width\_hint = 0.5) +  
 annotation\_north\_arrow(location = "tr", which\_north = "true",   
 pad\_x = unit(0.1, "in"), pad\_y = unit(0.1, "in"),  
 style = north\_arrow\_fancy\_orienteering) +  
 ggtitle(paste0("Disaggregate GVL data result - ices cells"))+  
 theme\_light()+  
 theme(legend.position = "bottom")  
  
final\_df\_by\_ices



Gross Value of landings by port

n\_port = final\_df %>%   
 st\_drop\_geometry() %>%   
 select(MMSI, vssl\_ln, port, Gear) %>%   
 unique() %>%  
 group\_by(Gear, vssl\_ln, port) %>%  
 summarise(n\_vessel\_port = n())  
  
  
nvessel\_weighted <- n\_port %>%  
 group\_by(Gear, port) %>%   
 mutate(weight = n\_vessel\_port / sum(n\_vessel\_port))   
  
GVL\_by\_vlength = FDI\_AER\_by\_icell %>%   
 group\_by(ger\_typ, vssl\_ln) %>%   
 summarise(GVL = mean(Gross.value.of.landings\_AER)) %>% rename(Gear = ger\_typ)   
   
FDI\_AER\_nvessel <- GVL\_by\_vlength %>%  
 inner\_join(nvessel\_weighted, by = c("Gear", "vssl\_ln"))  
  
FDI\_AER\_nvessel <- FDI\_AER\_nvessel %>%  
 mutate(  
 GVL\_by\_port = GVL \* weight)  
  
  
print(  
 FDI\_AER\_nvessel %>%   
 ggplot()+  
 geom\_bar(aes(y = reorder\_within(port,GVL\_by\_port, vssl\_ln), x = GVL\_by\_port, fill = vssl\_ln), stat = "identity")+  
 facet\_wrap(~ vssl\_ln, scales = "free\_x")+  
 scale\_y\_reordered() +  
 theme\_minimal()+  
 ylab("")  
)



Save data results

final\_df = final\_df %>%   
 select(MMSI,App\_F\_H, id,Gear,vssl\_ln,port, lon\_port, lat\_port, geometry) %>%   
 rename(GFW\_hours = App\_F\_H,  
 cscode = id,  
 vlength = vssl\_ln)  
   
write\_sf(final\_df, paste0(fd, "output\_df\_",Year\_CS,".shp"))

## 2 Protocol - FDI landings (species kg and price) by port

The protocol disaggregate FDI landing data for a time series.

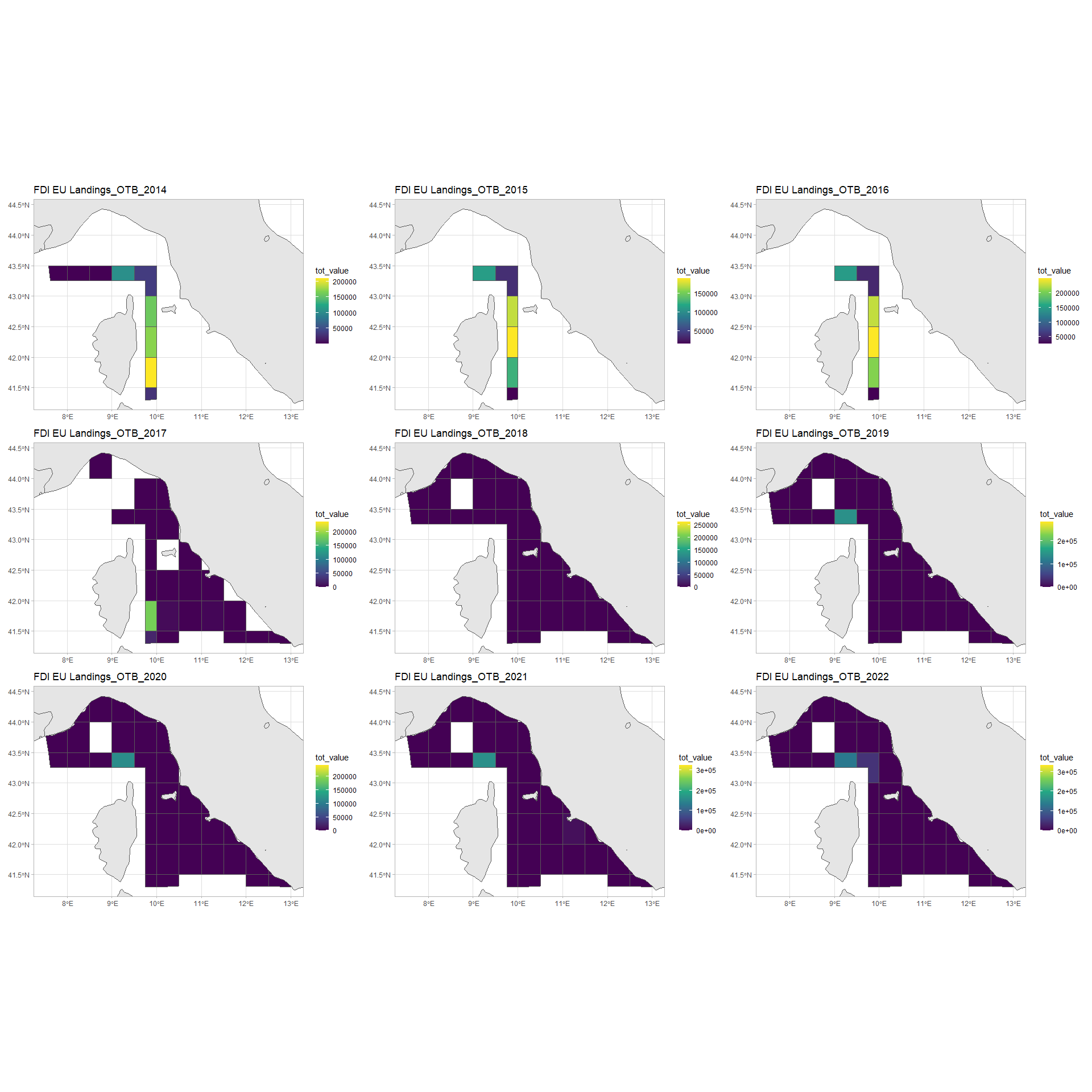
### Step 1 - Open and subset FDI Landing data

Open FDI landing data

Year\_CS = c(2014:2022)

Total landings coverage for the case study area - resulting from FDI data

#Set parameter for the map  
  
world <- ne\_countries(scale = "medium", returnclass = "sf", continent = "europe")  
world = st\_transform(world, crs = st\_crs(GSA))  
  
xmin = as.numeric(st\_bbox(GSA)[1])-0.0001  
xmax = as.numeric(st\_bbox(GSA)[3])+0.0001  
ymin = as.numeric(st\_bbox(GSA)[2])-0.0001  
ymax = as.numeric(st\_bbox(GSA)[4])+0.0001  
  
  
#landing  
  
for(g in 1 : length(Gear\_CS)){  
  
l\_data = purrr::map(landing\_sf, ~ .x %>%  
 filter(gear\_type == Gear\_CS[g]))  
   
plot\_list <- list()   
   
 for(y in 1: length(Year\_CS)){  
l\_plot = l\_data[[y]] %>%   
 group\_by(year,gear\_type,sub\_region,cscode) %>%   
 summarise(tot\_kg = sum(totwghtlandg), tot\_value = sum(totvallandg)) %>%   
 ggplot()+  
 geom\_sf(aes(fill = tot\_value))+  
 geom\_sf(data = world)+  
 coord\_sf(xlim = c(xmin, xmax), ylim = c(ymin, ymax))+  
 scale\_fill\_viridis\_c(option = "D")+   
 ggtitle(paste0("FDI EU Landings\_",Gear\_CS[g],"\_",Year\_CS[y]))+  
 theme\_light()  
  
plot\_list[[y]] <- l\_plot  
 }  
 combined\_plot <- wrap\_plots(plotlist = plot\_list, ncol = 3)  
 print(combined\_plot)  
   
 }



–> Remove first 3 years (2014-2015-2016) because there are few data

Save all the filtered data in a specific folder fd = “CaseStudy/Data/”

landing\_sf = landing\_sf[!(names(landing\_sf) %in% c("2014", "2015", "2016"))]  
  
fd\_price = "CaseStudy/Data\_price/"  
  
saveRDS(landing\_sf, paste0(fd\_price,"Landing\_sf.RData"))  
  
saveRDS(  
 purrr::map(landing\_sf, ~ .x %>%   
 st\_drop\_geometry() %>%   
 rename(id = cscode, gear = gear\_type, vlength = vessel\_length, tot\_fish\_weight = totwghtlandg, tot\_fish\_value = totvallandg)),   
 paste0(fd,"landing\_CS.rData"))

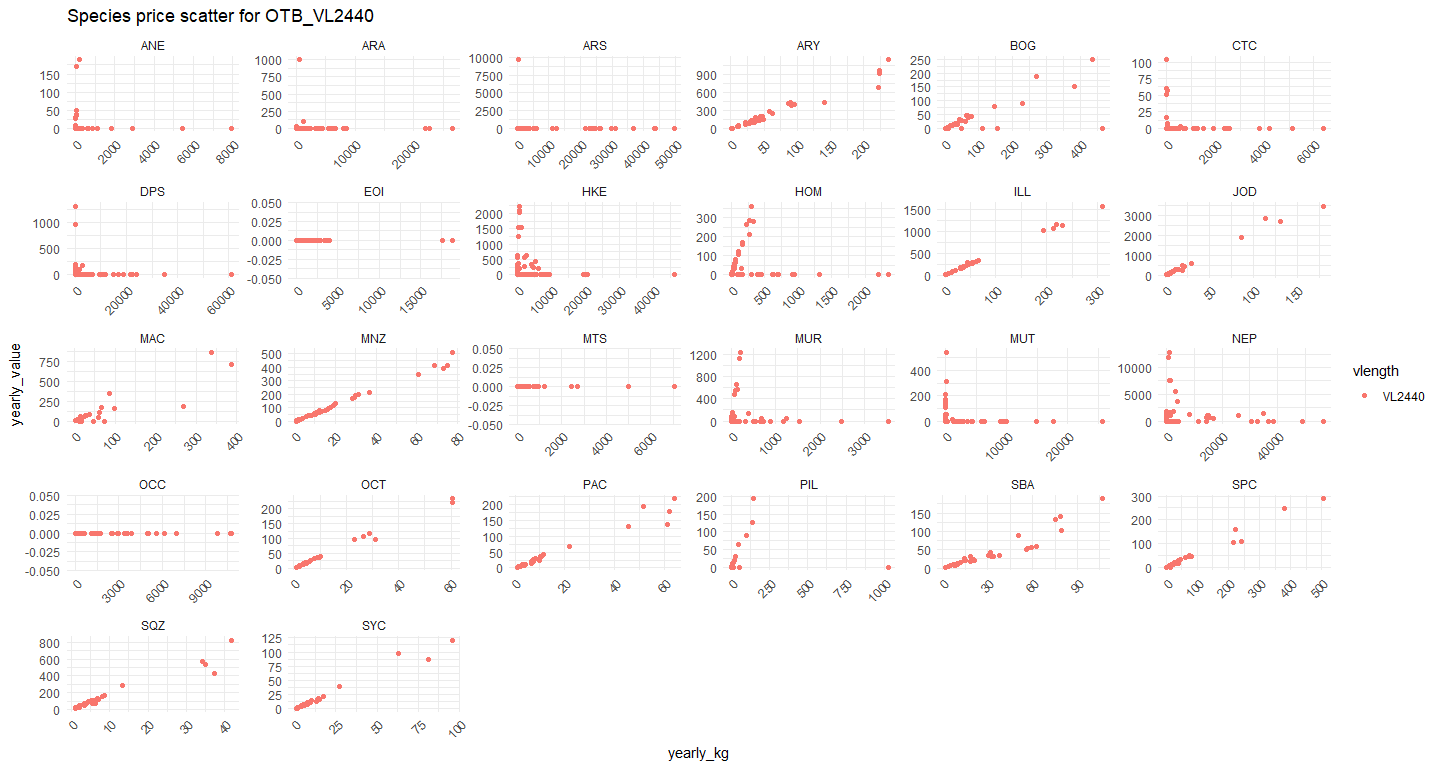
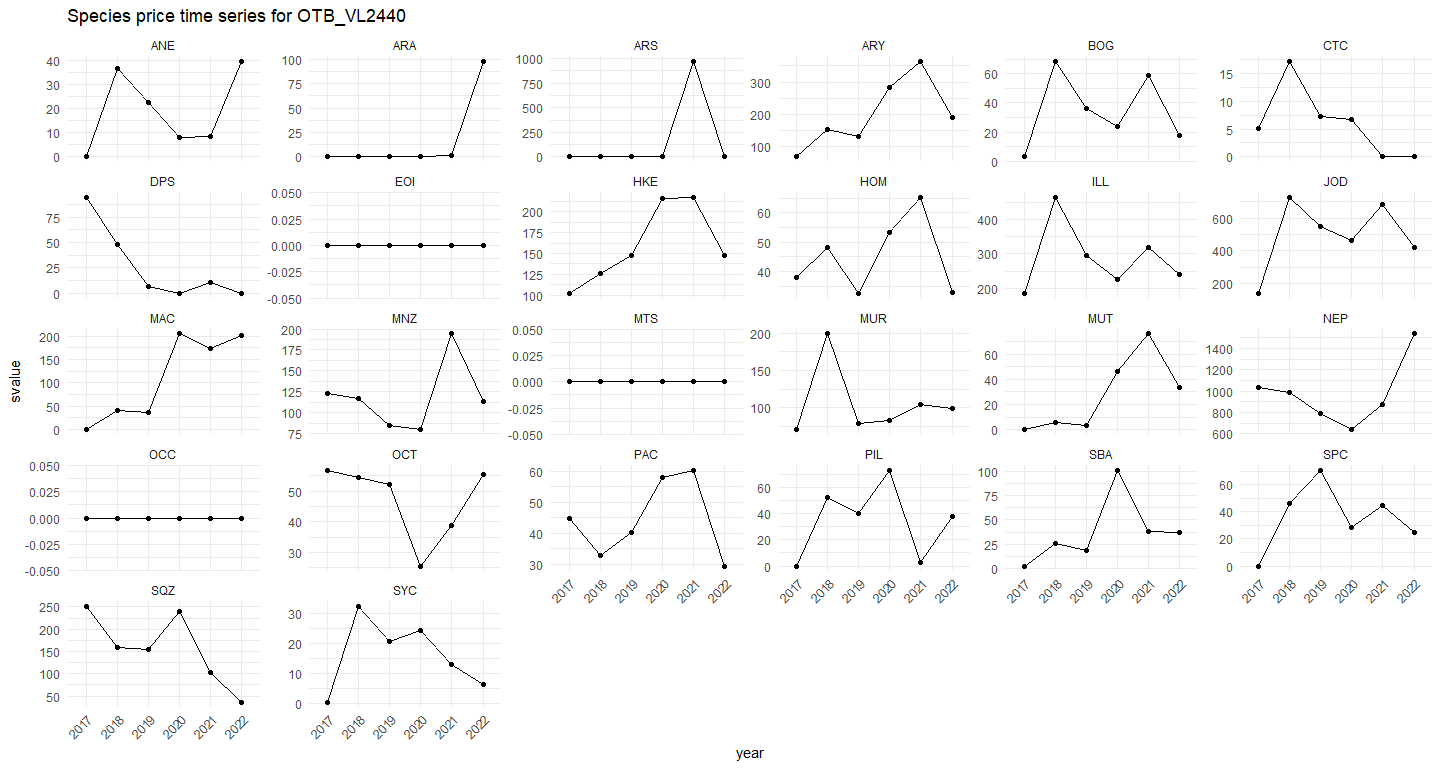
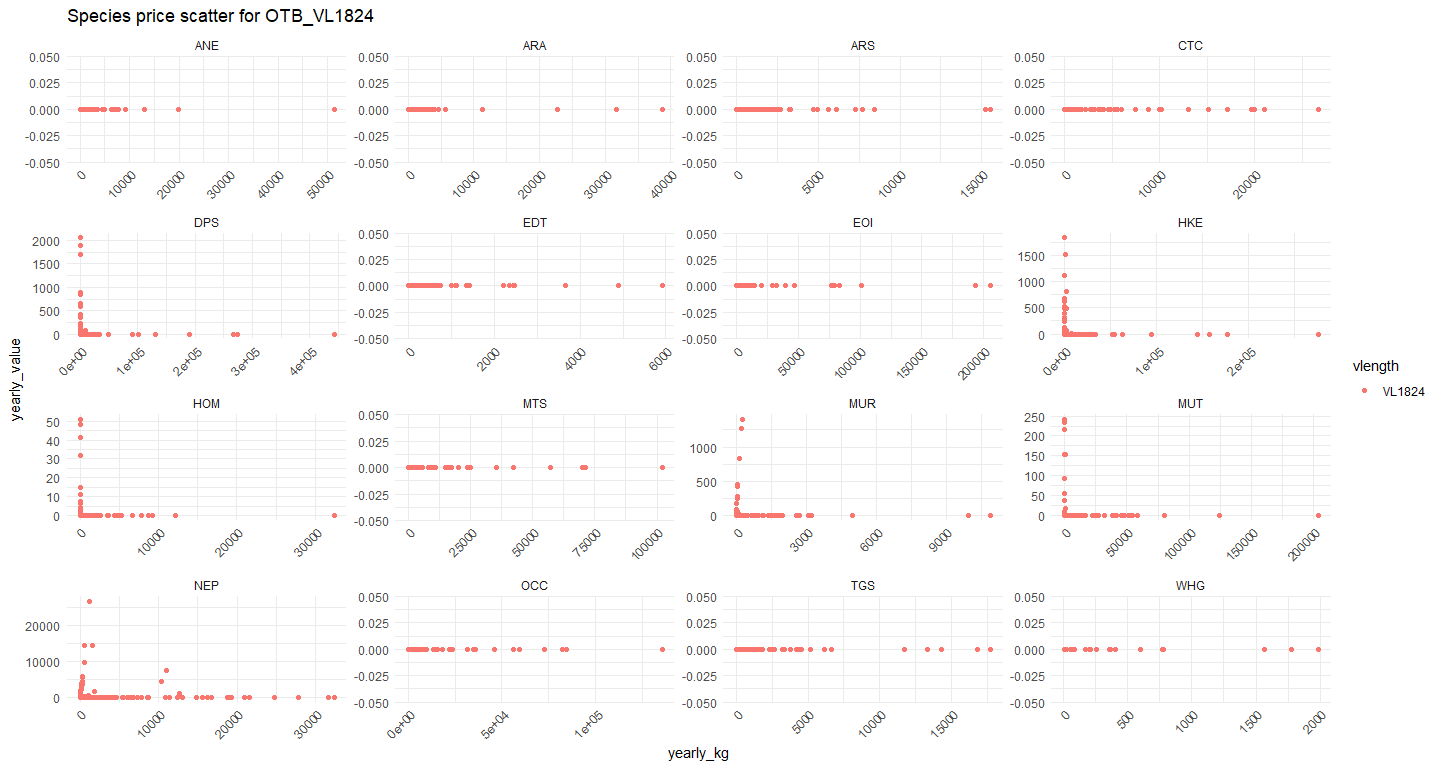
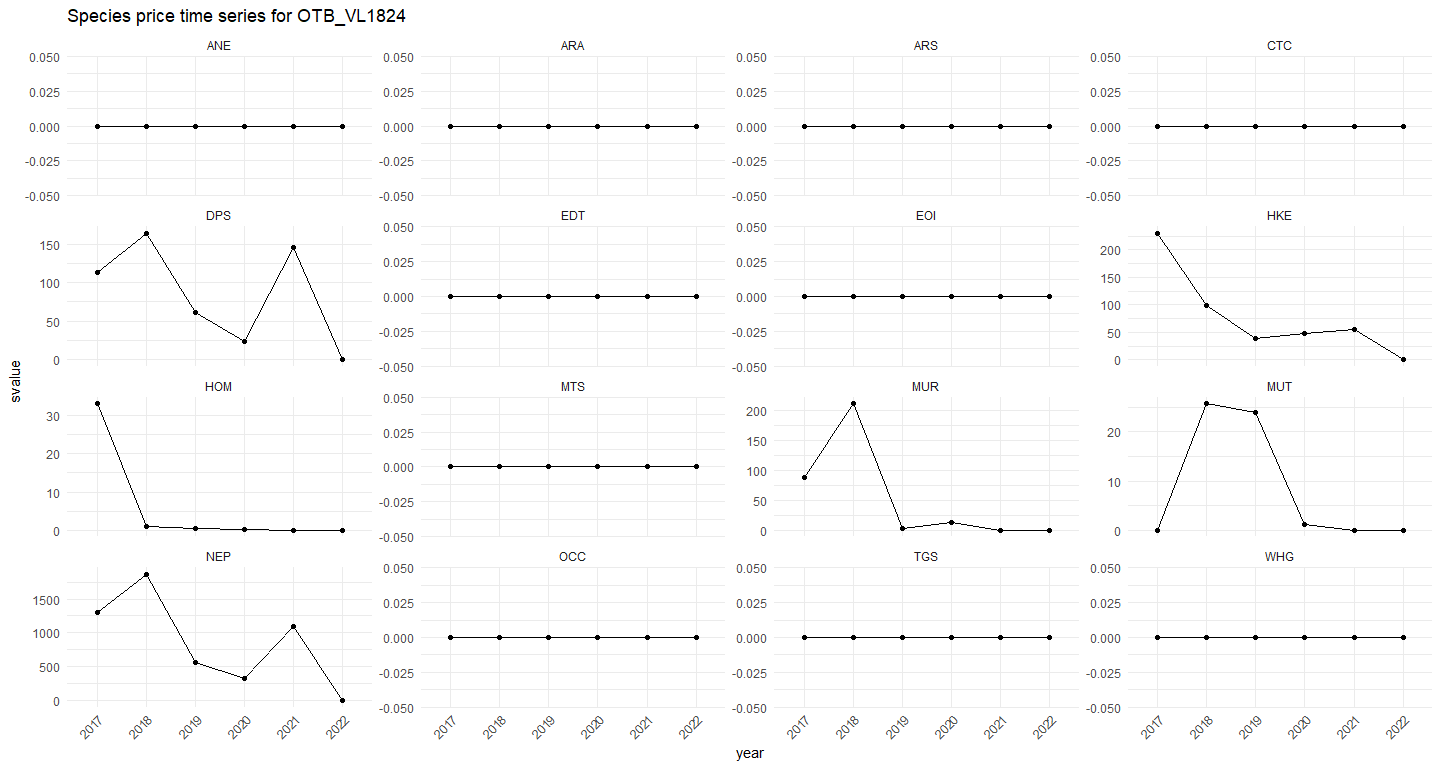
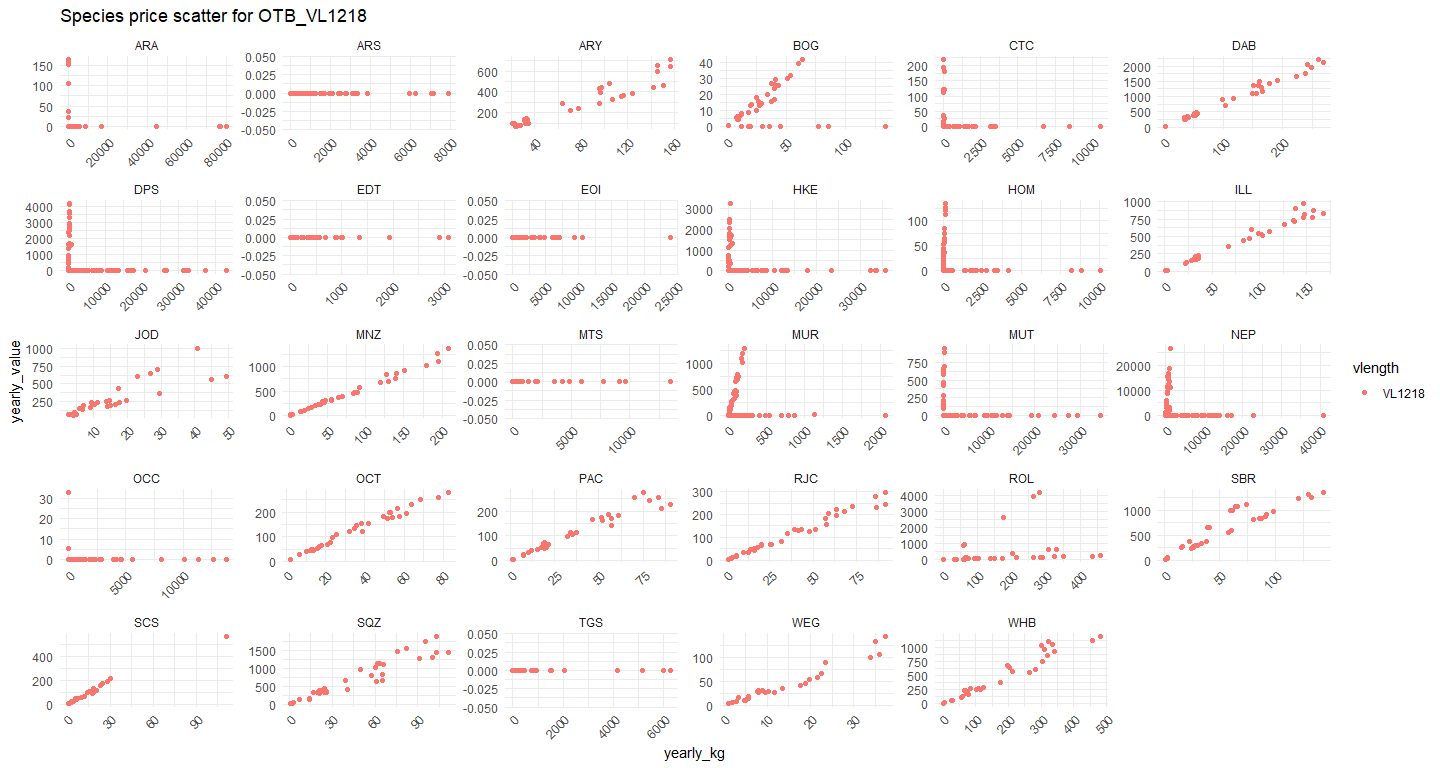
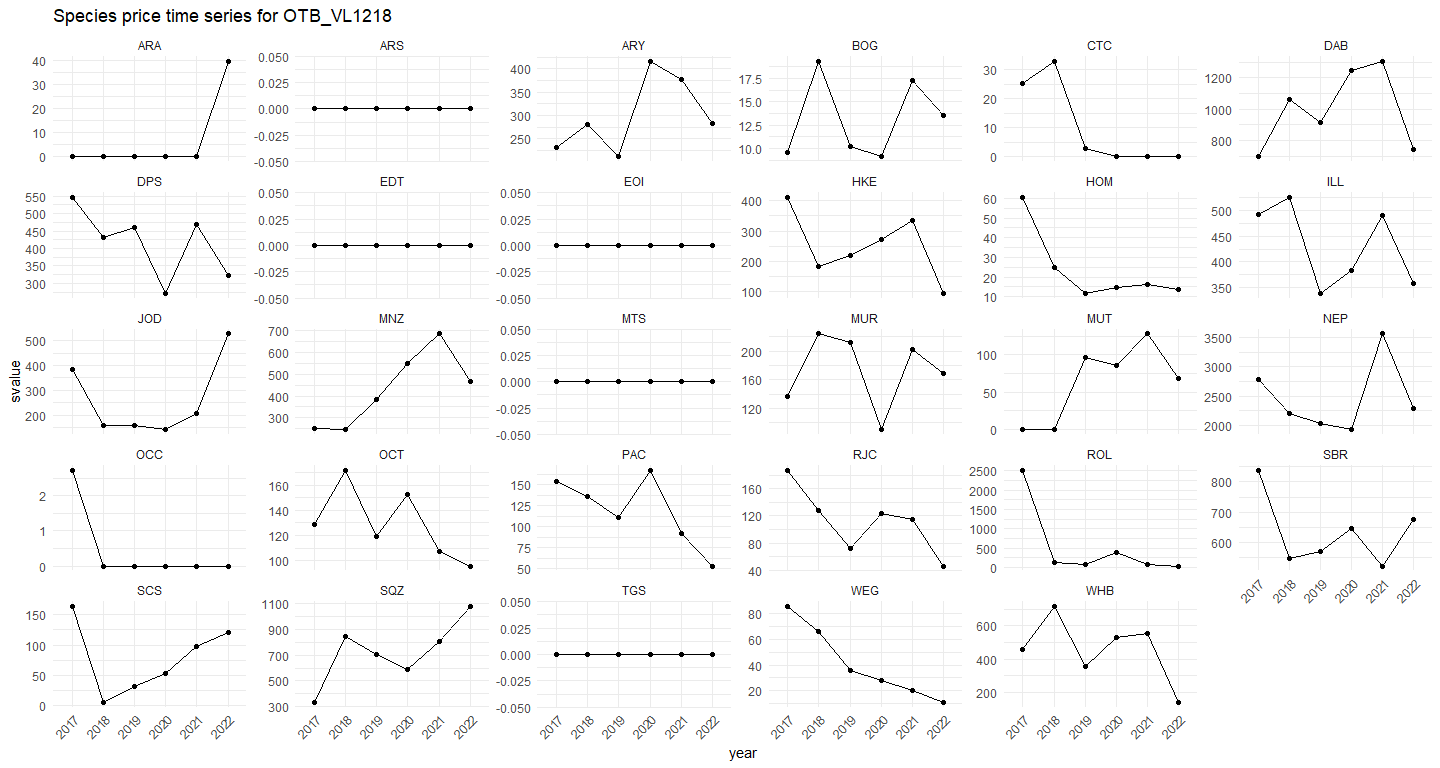
### Step 2 - Filter and clean landing data

Take only species that have values in all years of the time series

landing\_CS = readRDS(paste0(fd\_price,"Landing\_CS.RData"))  
  
euro\_species = purrr::map(  
 landing\_CS, ~ .x %>%   
 filter(tot\_fish\_weight != 0) )  
  
euro\_species\_df = do.call("rbind", euro\_species)  
  
species\_all\_years <- euro\_species\_df %>%  
 group\_by(gear,vlength,species) %>%   
 distinct(year, species) %>%  
 summarise(n\_years = n\_distinct(year), .groups = "drop") %>%  
 filter(n\_years == length(unique(euro\_species\_df$year)))  
  
euro\_species\_filtered <- euro\_species\_df %>%  
 inner\_join(species\_all\_years, by = c("species", "vlength", "gear"))  
  
# remove quarter  
  
euro\_species\_filtered <- euro\_species\_filtered %>%   
 group\_by(year,vlength, gear, id, species) %>%   
 summarise(yearly\_kg = mean(tot\_fish\_weight)\*1000, yearly\_value = mean(tot\_fish\_value))

In the time series check if there are some outlier.

df <- euro\_species\_filtered  
df$year <- as.character(df$year)  
  
vlent <- unique(df$vlength)  
  
empty\_cases <- list()  
  
  
for (g in seq\_along(Gear\_CS)) {  
 for (vl in seq\_along(vlent)) {  
   
 gear\_filter <- Gear\_CS[g]  
 vlength\_filter <- vlent[vl]  
   
 q\_data <- df %>%  
 filter(gear == gear\_filter, vlength == vlength\_filter)  
   
 if (nrow(q\_data) == 0) next  
   
 p\_data <- q\_data %>%  
 group\_by(year, species) %>%  
 summarise(svalue = mean(yearly\_value), .groups = "drop")  
   
 zero\_rows <- p\_data %>% filter(svalue == 0)  
   
 if (nrow(zero\_rows) > 0) {  
 for (i in seq\_len(nrow(zero\_rows))) {  
 empty\_cases[[length(empty\_cases) + 1]] <- list(  
 gear = gear\_filter,  
 vlength = vlength\_filter,  
 species = zero\_rows$species[i],  
 year = zero\_rows$year[i]  
 )  
 }  
 }  
  
 p <- ggplot(p\_data, aes(x = year, y = svalue, group = interaction(species))) +  
 geom\_point() +  
 geom\_line() +  
 facet\_wrap(~ species, scales = "free\_y") +  
 theme\_minimal() +  
 ggtitle(paste0("Species price time series for ", gear\_filter, "\_", vlength\_filter)) +  
 theme(axis.text.x = element\_text(angle = 45, vjust = 1, hjust = 1))  
   
 print(p)  
   
 q <- ggplot(q\_data) +  
 geom\_point(aes(x = yearly\_kg, y = yearly\_value, color = vlength)) +  
 facet\_wrap(~ species, scales = "free") +  
 theme\_minimal() +  
 ggtitle(paste0("Species price scatter for ", gear\_filter, "\_", vlength\_filter)) +  
 theme(axis.text.x = element\_text(angle = 45, vjust = 1, hjust = 1))  
   
 print(q)  
 }  
}



empty\_df <- do.call(rbind, lapply(empty\_cases, as.data.frame))  
print(empty\_df)

## gear vlength species year  
## 1 OTB VL1218 ARA 2017  
## 2 OTB VL1218 ARS 2017  
## 3 OTB VL1218 EDT 2017  
## 4 OTB VL1218 EOI 2017  
## 5 OTB VL1218 MTS 2017  
## 6 OTB VL1218 MUT 2017  
## 7 OTB VL1218 TGS 2017  
## 8 OTB VL1218 ARA 2018  
## 9 OTB VL1218 ARS 2018  
## 10 OTB VL1218 EDT 2018  
## 11 OTB VL1218 EOI 2018  
## 12 OTB VL1218 MTS 2018  
## 13 OTB VL1218 MUT 2018  
## 14 OTB VL1218 OCC 2018  
## 15 OTB VL1218 TGS 2018  
## 16 OTB VL1218 ARA 2019  
## 17 OTB VL1218 ARS 2019  
## 18 OTB VL1218 EDT 2019  
## 19 OTB VL1218 EOI 2019  
## 20 OTB VL1218 MTS 2019  
## 21 OTB VL1218 OCC 2019  
## 22 OTB VL1218 TGS 2019  
## 23 OTB VL1218 ARA 2020  
## 24 OTB VL1218 ARS 2020  
## 25 OTB VL1218 CTC 2020  
## 26 OTB VL1218 EDT 2020  
## 27 OTB VL1218 EOI 2020  
## 28 OTB VL1218 MTS 2020  
## 29 OTB VL1218 OCC 2020  
## 30 OTB VL1218 TGS 2020  
## 31 OTB VL1218 ARA 2021  
## 32 OTB VL1218 ARS 2021  
## 33 OTB VL1218 CTC 2021  
## 34 OTB VL1218 EDT 2021  
## 35 OTB VL1218 EOI 2021  
## 36 OTB VL1218 MTS 2021  
## 37 OTB VL1218 OCC 2021  
## 38 OTB VL1218 TGS 2021  
## 39 OTB VL1218 ARS 2022  
## 40 OTB VL1218 CTC 2022  
## 41 OTB VL1218 EDT 2022  
## 42 OTB VL1218 EOI 2022  
## 43 OTB VL1218 MTS 2022  
## 44 OTB VL1218 OCC 2022  
## 45 OTB VL1218 TGS 2022  
## 46 OTB VL1824 ANE 2017  
## 47 OTB VL1824 ARA 2017  
## 48 OTB VL1824 ARS 2017  
## 49 OTB VL1824 CTC 2017  
## 50 OTB VL1824 EDT 2017  
## 51 OTB VL1824 EOI 2017  
## 52 OTB VL1824 MTS 2017  
## 53 OTB VL1824 MUT 2017  
## 54 OTB VL1824 OCC 2017  
## 55 OTB VL1824 TGS 2017  
## 56 OTB VL1824 WHG 2017  
## 57 OTB VL1824 ANE 2018  
## 58 OTB VL1824 ARA 2018  
## 59 OTB VL1824 ARS 2018  
## 60 OTB VL1824 CTC 2018  
## 61 OTB VL1824 EDT 2018  
## 62 OTB VL1824 EOI 2018  
## 63 OTB VL1824 MTS 2018  
## 64 OTB VL1824 OCC 2018  
## 65 OTB VL1824 TGS 2018  
## 66 OTB VL1824 WHG 2018  
## 67 OTB VL1824 ANE 2019  
## 68 OTB VL1824 ARA 2019  
## 69 OTB VL1824 ARS 2019  
## 70 OTB VL1824 CTC 2019  
## 71 OTB VL1824 EDT 2019  
## 72 OTB VL1824 EOI 2019  
## 73 OTB VL1824 MTS 2019  
## 74 OTB VL1824 OCC 2019  
## 75 OTB VL1824 TGS 2019  
## 76 OTB VL1824 WHG 2019  
## 77 OTB VL1824 ANE 2020  
## 78 OTB VL1824 ARA 2020  
## 79 OTB VL1824 ARS 2020  
## 80 OTB VL1824 CTC 2020  
## 81 OTB VL1824 EDT 2020  
## 82 OTB VL1824 EOI 2020  
## 83 OTB VL1824 MTS 2020  
## 84 OTB VL1824 OCC 2020  
## 85 OTB VL1824 TGS 2020  
## 86 OTB VL1824 WHG 2020  
## 87 OTB VL1824 ANE 2021  
## 88 OTB VL1824 ARA 2021  
## 89 OTB VL1824 ARS 2021  
## 90 OTB VL1824 CTC 2021  
## 91 OTB VL1824 EDT 2021  
## 92 OTB VL1824 EOI 2021  
## 93 OTB VL1824 HOM 2021  
## 94 OTB VL1824 MTS 2021  
## 95 OTB VL1824 MUR 2021  
## 96 OTB VL1824 MUT 2021  
## 97 OTB VL1824 OCC 2021  
## 98 OTB VL1824 TGS 2021  
## 99 OTB VL1824 WHG 2021  
## 100 OTB VL1824 ANE 2022  
## 101 OTB VL1824 ARA 2022  
## 102 OTB VL1824 ARS 2022  
## 103 OTB VL1824 CTC 2022  
## 104 OTB VL1824 DPS 2022  
## 105 OTB VL1824 EDT 2022  
## 106 OTB VL1824 EOI 2022  
## 107 OTB VL1824 HKE 2022  
## 108 OTB VL1824 HOM 2022  
## 109 OTB VL1824 MTS 2022  
## 110 OTB VL1824 MUR 2022  
## 111 OTB VL1824 MUT 2022  
## 112 OTB VL1824 NEP 2022  
## 113 OTB VL1824 OCC 2022  
## 114 OTB VL1824 TGS 2022  
## 115 OTB VL1824 WHG 2022  
## 116 OTB VL2440 ANE 2017  
## 117 OTB VL2440 ARA 2017  
## 118 OTB VL2440 ARS 2017  
## 119 OTB VL2440 EOI 2017  
## 120 OTB VL2440 MAC 2017  
## 121 OTB VL2440 MTS 2017  
## 122 OTB VL2440 MUT 2017  
## 123 OTB VL2440 OCC 2017  
## 124 OTB VL2440 PIL 2017  
## 125 OTB VL2440 SPC 2017  
## 126 OTB VL2440 ARA 2018  
## 127 OTB VL2440 ARS 2018  
## 128 OTB VL2440 EOI 2018  
## 129 OTB VL2440 MTS 2018  
## 130 OTB VL2440 OCC 2018  
## 131 OTB VL2440 ARA 2019  
## 132 OTB VL2440 ARS 2019  
## 133 OTB VL2440 EOI 2019  
## 134 OTB VL2440 MTS 2019  
## 135 OTB VL2440 OCC 2019  
## 136 OTB VL2440 ARA 2020  
## 137 OTB VL2440 ARS 2020  
## 138 OTB VL2440 DPS 2020  
## 139 OTB VL2440 EOI 2020  
## 140 OTB VL2440 MTS 2020  
## 141 OTB VL2440 OCC 2020  
## 142 OTB VL2440 CTC 2021  
## 143 OTB VL2440 EOI 2021  
## 144 OTB VL2440 MTS 2021  
## 145 OTB VL2440 OCC 2021  
## 146 OTB VL2440 ARS 2022  
## 147 OTB VL2440 CTC 2022  
## 148 OTB VL2440 DPS 2022  
## 149 OTB VL2440 EOI 2022  
## 150 OTB VL2440 MTS 2022  
## 151 OTB VL2440 OCC 2022

Some species have zero value for all time series for specific gear and length combinations, so we have decided to remove them.

empty\_df = empty\_df %>%  
 distinct(gear, vlength, species, year) %>%  
 group\_by(gear, vlength, species) %>%  
 summarise(n\_zero\_years = n(), .groups = "drop") %>%   
 filter(n\_zero\_years == length(unique(euro\_species\_filtered$year)))  
  
remove\_keys <- paste(empty\_df$gear, empty\_df$vlength, empty\_df$species, sep = "\_")  
  
euro\_species\_clean <- euro\_species\_filtered %>%  
 filter(!paste(gear, vlength, species, sep = "\_") %in% remove\_keys)

For other species, it is evident that not all years had values. However, it is notable that there is a positive correlation with kilograms. The issue under discussion is resolved by calculating the linear relationship between value and kilograms by species. Subsequently, the value of the relationship is associated with the missing points in the time series.

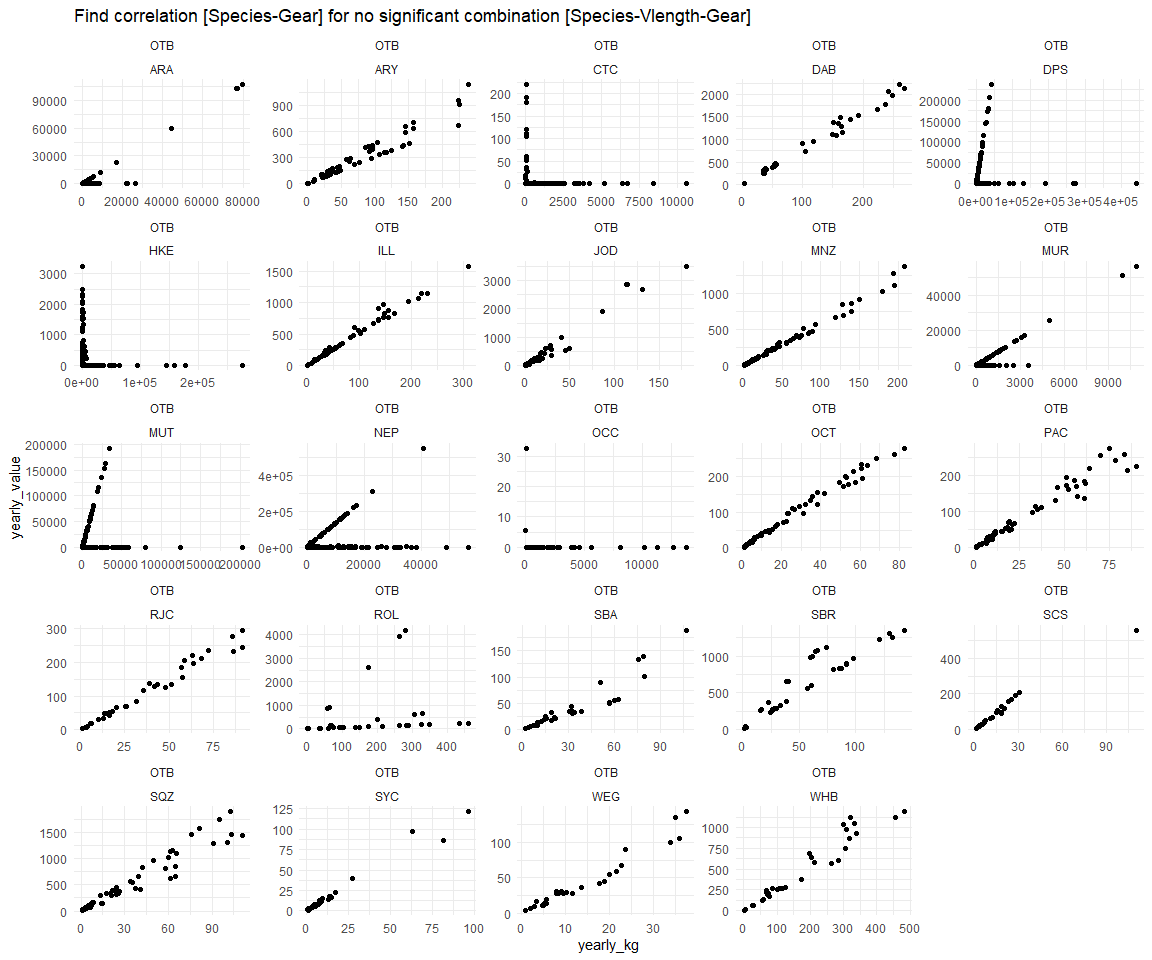
In the case that the relationship between value and kg is non-linear, the missing value was replaced with linear regression without vessel length.

df\_full <- euro\_species\_clean  
df\_model <- df\_full[df\_full$yearly\_value != 0, ]  
  
non\_significant\_models <- list()  
  
gear\_list <- unique(df\_model$gear)  
vlength\_list <- unique(df\_model$vlength)  
  
for (g in seq\_along(gear\_list)) {  
 for (vl in seq\_along(vlength\_list)) {  
   
 gear\_value <- gear\_list[g]  
 vlen\_value <- vlength\_list[vl]  
   
 subset\_model <- df\_model[df\_model$gear == gear\_value &   
 df\_model$vlength == vlen\_value, ]  
   
 species\_list <- unique(subset\_model$species)  
   
 for (s in seq\_along(species\_list)) {  
   
 species\_value <- species\_list[s]  
   
 subset\_species\_model <- subset\_model[subset\_model$species == species\_value, ]  
   
 subset\_zero <- df\_full$gear == gear\_value &   
 df\_full$vlength == vlen\_value &   
 df\_full$species == species\_value &   
 df\_full$yearly\_value == 0  
   
 if (nrow(subset\_species\_model) >= 2) {  
   
 model <- lm(yearly\_value ~ yearly\_kg, data = subset\_species\_model)  
 model\_summary <- summary(model)  
 fstat <- model\_summary$fstatistic  
 r\_squared <- model\_summary$r.squared  
 pval <- pf(fstat[1], fstat[2], fstat[3], lower.tail = FALSE)  
   
 if (!is.na(pval) && pval < 0.05 && r\_squared > 0.4 && any(subset\_zero)) {  
 # Previsione e sostituzione  
 df\_full[subset\_zero, "yearly\_value"] <- predict(model, newdata = df\_full[subset\_zero, ])  
 } else {  
 # Modello non significativo: salva  
 non\_significant\_models[[length(non\_significant\_models) + 1]] <- data.frame(  
 species = species\_value,  
 gear = gear\_value,  
 vlength = vlen\_value,  
 p\_value = ifelse(is.na(pval), NA, round(pval, 5)),  
 r\_squared = round(r\_squared, 3)  
 )  
 }  
 }  
 }  
 }  
}  
  
if (length(non\_significant\_models) > 0) {  
 non\_significant\_df <- do.call(rbind, non\_significant\_models)  
 print(non\_significant\_df)  
} else {  
 message("All models were significant.")  
}

## species gear vlength p\_value r\_squared  
## value HKE OTB VL1218 0.00054 0.301  
## value1 ILL OTB VL1218 0.00000 0.969  
## value2 MNZ OTB VL1218 0.00000 0.986  
## value3 OCC OTB VL1218 NA 1.000  
## value4 OCT OTB VL1218 0.00000 0.980  
## value5 RJC OTB VL1218 0.00000 0.974  
## value6 SCS OTB VL1218 0.00000 0.976  
## value7 SQZ OTB VL1218 0.00000 0.905  
## value8 WHB OTB VL1218 0.00000 0.940  
## value9 ARY OTB VL1218 0.00000 0.903  
## value10 CTC OTB VL1218 0.13643 0.190  
## value11 DAB OTB VL1218 0.00000 0.982  
## value12 JOD OTB VL1218 0.00000 0.763  
## value13 MUR OTB VL1218 0.59223 0.009  
## value14 PAC OTB VL1218 0.00000 0.946  
## value15 ROL OTB VL1218 0.17240 0.059  
## value16 SBR OTB VL1218 0.00000 0.858  
## value17 WEG OTB VL1218 0.00000 0.942  
## value18 DPS OTB VL1824 0.28945 0.045  
## value19 HKE OTB VL1824 0.24448 0.052  
## value20 NEP OTB VL1824 0.50660 0.017  
## value21 MUT OTB VL1824 0.52300 0.032  
## value22 ARY OTB VL2440 0.00000 0.956  
## value23 CTC OTB VL2440 0.29481 0.180  
## value24 DPS OTB VL2440 0.43113 0.052  
## value25 HKE OTB VL2440 0.78938 0.002  
## value26 ILL OTB VL2440 0.00000 0.999  
## value27 JOD OTB VL2440 0.00000 0.982  
## value28 MNZ OTB VL2440 0.00000 0.990  
## value29 MUR OTB VL2440 0.71690 0.004  
## value30 NEP OTB VL2440 0.75014 0.002  
## value31 OCT OTB VL2440 0.00000 0.993  
## value32 PAC OTB VL2440 0.00000 0.953  
## value33 SQZ OTB VL2440 0.00000 0.942  
## value34 SBA OTB VL2440 0.00000 0.877  
## value35 SYC OTB VL2440 0.00000 0.975  
## value36 MUT OTB VL2440 0.74592 0.005  
## value37 ARA OTB VL2440 0.68407 0.022

In the case that the relationship between value and kg for [gear-vessel length-species] is non-linear, the missing value was replaced with linear regression for [gear-species].

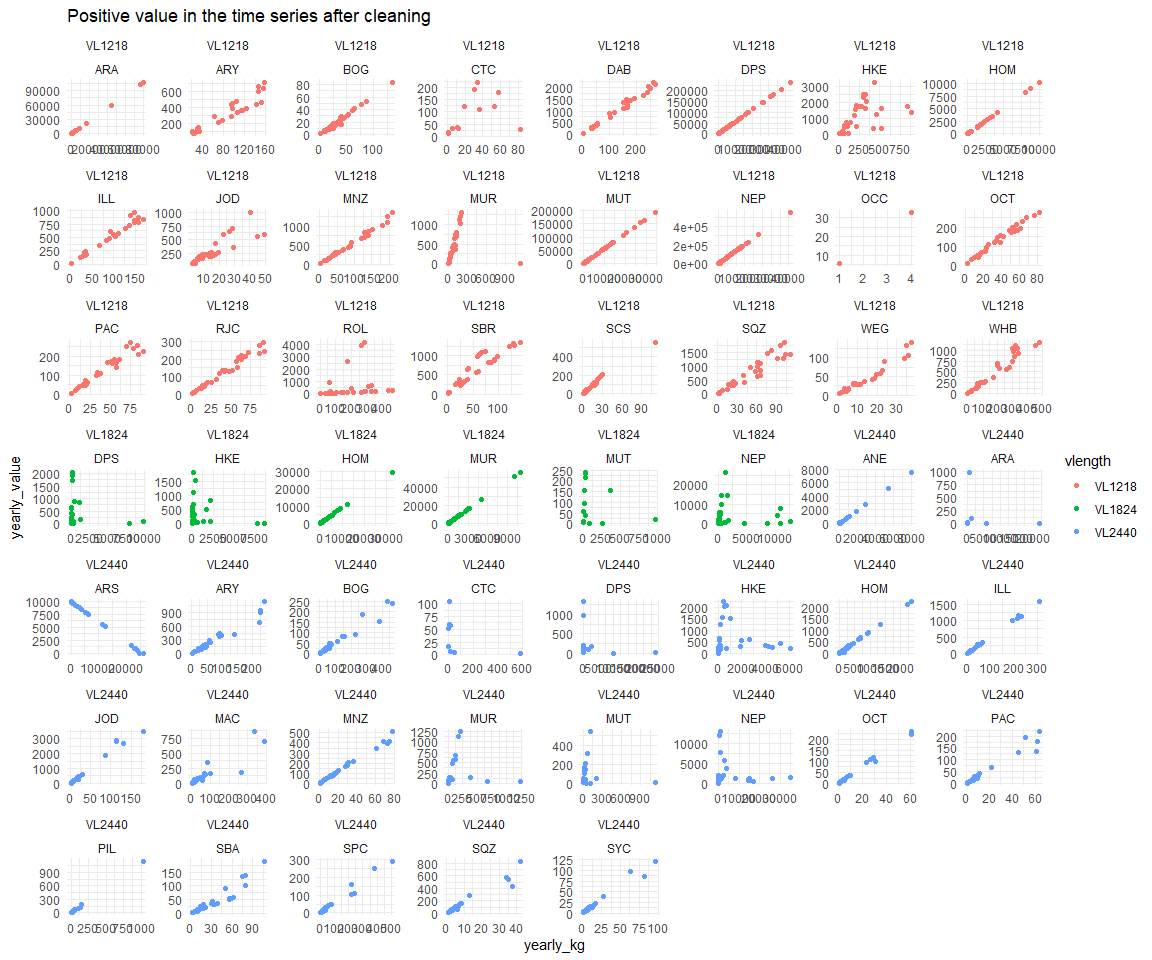
keys\_nosig <- paste(non\_significant\_df$gear, non\_significant\_df$species, sep = "\_")  
  
df\_full %>%   
 filter(paste(gear, species, sep = "\_") %in% keys\_nosig) %>%   
 ggplot()+  
 geom\_point(aes(x = yearly\_kg, y = yearly\_value))+  
 facet\_wrap(~gear+species, scales = "free")+  
 theme\_minimal()+  
 ggtitle("Find correlation [Species-Gear] for no significant combination [Species-Vlength-Gear]")



keys\_nosig <- paste(non\_significant\_df$gear, non\_significant\_df$vlength, non\_significant\_df$species, sep = "\_")  
  
df\_model\_gs <- df\_full %>%   
 filter(paste(gear, vlength, species, sep = "\_") %in% keys\_nosig) %>%   
 filter(yearly\_value != 0)  
  
non\_significant\_models <- list()  
  
gear\_list <- unique(df\_model\_gs$gear)  
  
for (g in seq\_along(gear\_list)) {  
   
 gear\_value <- gear\_list[g]  
 subset\_model <- df\_model\_gs[df\_model\_gs$gear == gear\_value, ]  
   
 species\_list <- unique(subset\_model$species)  
   
 for (s in seq\_along(species\_list)) {  
   
 species\_value <- species\_list[s]  
 subset\_species\_model <- subset\_model[subset\_model$species == species\_value, ]  
   
 if (nrow(subset\_species\_model) >= 2) {  
   
 model <- lm(yearly\_value ~ yearly\_kg, data = subset\_species\_model)  
 fstat <- summary(model)$fstatistic  
 pval <- pf(fstat[1], fstat[2], fstat[3], lower.tail = FALSE)  
   
 if (!is.na(pval) && pval <= 0.05) {  
   
 rows\_to\_update <- which(  
 paste(df\_full$gear, df\_full$vlength, df\_full$species, sep = "\_") %in% keys\_nosig &  
 df\_full$gear == gear\_value &  
 df\_full$species == species\_value &  
 df\_full$yearly\_value == 0  
 )  
   
 if (length(rows\_to\_update) > 0) {  
 predicted <- predict(model, newdata = df\_full[rows\_to\_update, ])  
 df\_full$yearly\_value[rows\_to\_update] <- predicted  
 }  
   
 } else {  
 non\_significant\_models[[length(non\_significant\_models) + 1]] <- data.frame(  
 gear = gear\_value,  
 species = species\_value,  
 p\_value = ifelse(is.na(pval), NA, round(pval, 5))  
 )  
 }  
 }  
 }  
 }  
  
  
if (length(non\_significant\_models) > 0) {  
 non\_significant\_df <- do.call(rbind, non\_significant\_models)  
 print(non\_significant\_df)  
} else {  
 message("All models were significant.")  
}

## gear species p\_value  
## value OTB HKE 0.37703  
## value1 OTB OCC NA  
## value2 OTB CTC 0.50123  
## value3 OTB MUR 0.55016  
## value4 OTB ROL 0.17240  
## value5 OTB DPS 0.17565  
## value6 OTB NEP 0.45257  
## value7 OTB MUT 0.51800  
## value8 OTB ARA 0.68407

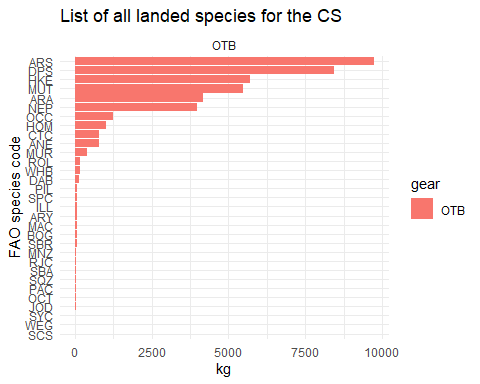
for (g in seq\_along(Gear\_CS)) {  
  
 p\_m = df\_full %>%   
 filter(gear == Gear\_CS[g]) %>%   
 filter(yearly\_value > 0) %>%   
 ggplot()+  
 geom\_point(aes(x = yearly\_kg, y = yearly\_value, color = vlength))+  
 facet\_wrap(~vlength+species, scales = "free")+  
 theme\_minimal()+  
 ggtitle("Positive value in the time series after cleaning")  
   
 print(p\_m)  
}



Now we are ready for disaggregate data by landing port. First we remove remaining zero value and we calculate price as: value/kg

df\_full %>%   
 group\_by(species, gear) %>%  
 summarise(mvalue = mean(yearly\_value), kg = (mean(yearly\_kg))) %>%  
 ggplot() +  
 geom\_bar(aes(x = kg, y = reorder\_within(species, kg, gear), fill = gear), stat = "identity")+  
 ylab("FAO species code")+  
 facet\_wrap(~ gear, scales = "free")+  
 xlab("kg")+  
 ggtitle("List of all landed species for the CS")+  
 scale\_y\_reordered()+  
 theme\_minimal()

## `summarise()` has grouped output by 'species'. You can override using the  
## `.groups` argument.



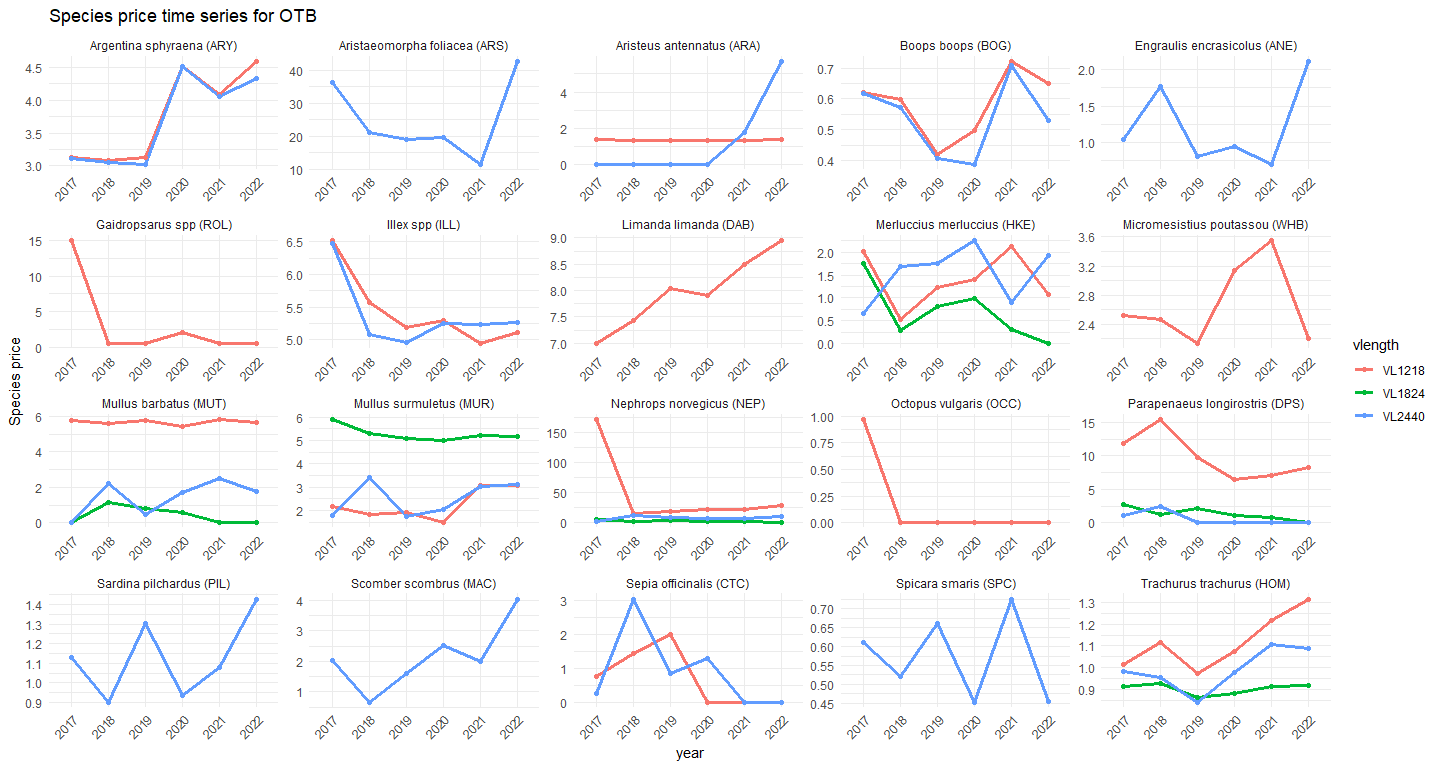
species\_sel <- df\_full %>%  
 group\_by(species, gear) %>%  
 summarise(m\_kg = mean(yearly\_kg), .groups = "drop") %>%  
 group\_by(gear) %>%  
 slice\_max(m\_kg, n = 20) %>%  
 ungroup()   
  
 FDI\_land\_spe\_filter <- df\_full %>%  
 inner\_join(species\_sel, by = c("species", "gear")) %>%   
 select(-m\_kg) %>%   
 mutate(price = yearly\_value/yearly\_kg)

Some results

FAO\_sp = read.csv(paste0(wd,"ASFIS\_sp\_2025.csv")) %>% select("Alpha3\_Code","Scientific\_Name") %>% rename("species" = "Alpha3\_Code")  
  
FDI\_land\_spe\_filter$year = as.character(FDI\_land\_spe\_filter$year)  
  
gear\_filter = unique(FDI\_land\_spe\_filter$gear)  
  
for (g in seq\_along(gear\_filter)) {  
  
 print(FDI\_land\_spe\_filter %>%   
 filter(gear == gear\_filter[g]) %>%   
 group\_by(year, vlength, species) %>%   
 summarise(m\_price = mean(price)) %>%   
 left\_join(FAO\_sp) %>%  
 mutate(sp\_id = paste0(Scientific\_Name," (",species,")")) %>%   
   
 ggplot(aes(x = year, y = m\_price, color = vlength,  
 group = interaction(species, vlength)  
   
 )) +  
 geom\_line(size = 1.1) +  
 geom\_point() +  
 facet\_wrap(~ sp\_id, scales = "free", ncol = 5) +  
 theme\_minimal() +  
 ggtitle(paste0("Species price time series for ", gear\_filter[g])) +  
 ylab("Species price")+  
 theme(axis.text.x = element\_text(angle = 45, vjust = 1, hjust = 1))  
)  
 }

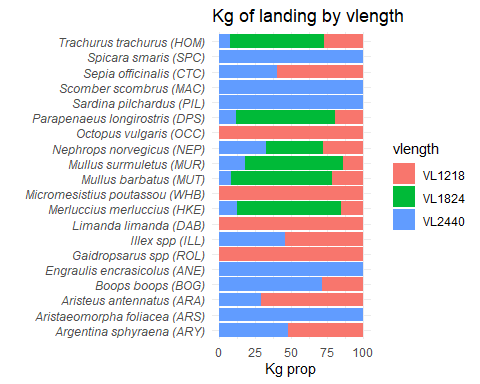
## `summarise()` has grouped output by 'year', 'vlength'. You can override using  
## the `.groups` argument.  
## Joining with `by = join\_by(species)`

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## ℹ Please use `linewidth` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



for (g in seq\_along(gear\_filter)) {  
print(  
 FDI\_land\_spe\_filter %>%  
 filter(gear == gear\_filter[g]) %>%   
 group\_by(vlength,species) %>%  
 summarise(kg\_gear = sum(yearly\_kg)) %>%  
 group\_by(species) %>%   
 mutate(kg\_tot = sum(kg\_gear)) %>%   
 mutate(kg\_prop = (kg\_gear/kg\_tot)\*100) %>%  
 left\_join(FAO\_sp) %>%  
 mutate(sp\_id = paste0(Scientific\_Name," (",species,")")) %>%   
 ggplot() +  
 geom\_bar(aes(y = sp\_id, x = kg\_prop, fill = vlength),stat = "identity") +  
 labs(title = "Kg of landing by vlength", y = "", x = "Kg prop") +  
 theme\_minimal()+  
 theme(axis.text.y = element\_text(face = "italic")))  
}

## `summarise()` has grouped output by 'vlength'. You can override using the  
## `.groups` argument.  
## Joining with `by = join\_by(species)`



Save data filtered

saveRDS(FDI\_land\_spe\_filter, paste0(fd\_price,"FDI\_land\_spe\_filter.RData"))

### Step 3 - Calculate and add price by species

Load dataset containing the number of vessels by port, type of fishing gear used, and vessel length. Load port coordinates.

Here we first chech gear code used and change as FDI code.

nvessel\_port = read\_sf(paste0(fd,"output\_df\_2021.shp")) %>%   
 st\_drop\_geometry() %>%   
 select(MMSI, vlength, port, Gear) %>%   
 mutate(year = "2021") %>%   
 unique()

nvessel\_port <- nvessel\_port %>%  
 group\_by(year, Gear, vlength, port) %>%  
 summarise(n\_vessel\_port = n())

## `summarise()` has grouped output by 'year', 'Gear', 'vlength'. You can override  
## using the `.groups` argument.

nvessel\_weighted <- nvessel\_port %>%  
 group\_by(year, Gear, port) %>%   
 mutate(weight = n\_vessel\_port / sum(n\_vessel\_port)) %>%   
 rename(gear = Gear)  
  
FDI\_land\_spe\_filter = readRDS(paste0(fd\_price,"FDI\_land\_spe\_filter.RData")) %>%   
 group\_by(year, gear, vlength, species) %>%   
 summarise(mval = mean(yearly\_value), mkg = mean(yearly\_kg), mprice = mean(price))

## `summarise()` has grouped output by 'year', 'gear', 'vlength'. You can override  
## using the `.groups` argument.

FDI\_land\_spe\_nvessel <- FDI\_land\_spe\_filter %>%  
 inner\_join(nvessel\_weighted, by = c("year", "gear", "vlength"))

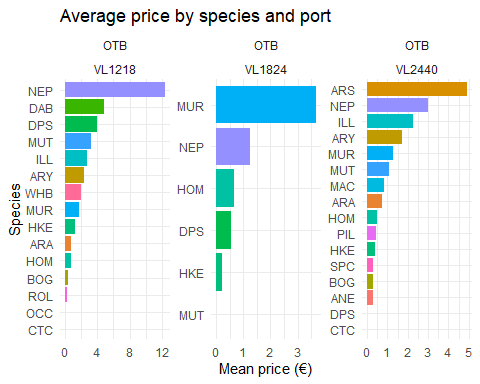
## Warning in inner\_join(., nvessel\_weighted, by = c("year", "gear", "vlength")): Detected an unexpected many-to-many relationship between `x` and `y`.  
## ℹ Row 149 of `x` matches multiple rows in `y`.  
## ℹ Row 1 of `y` matches multiple rows in `x`.  
## ℹ If a many-to-many relationship is expected, set `relationship =  
## "many-to-many"` to silence this warning.

FDI\_land\_spe\_nvessel <- FDI\_land\_spe\_nvessel %>%  
 mutate(  
 mval\_by\_port = mval \* weight,  
 mkg\_by\_port = mkg \* weight,  
 mprice\_by\_port = mprice \* weight)  
  
df\_port\_species <- FDI\_land\_spe\_nvessel %>%  
 group\_by(year, gear, vlength, port, species) %>%  
 summarise(  
 mval = sum(mval\_by\_port, na.rm = TRUE),  
 mkg = sum(mkg\_by\_port, na.rm = TRUE),  
 mprice = sum(mprice\_by\_port))

## `summarise()` has grouped output by 'year', 'gear', 'vlength', 'port'. You can  
## override using the `.groups` argument.

df\_port\_species %>%   
 group\_by(species, gear, vlength) %>%   
 summarise(mean\_price = mean(mprice)) %>%   
   
ggplot( aes(y = reorder\_within(species, mean\_price, interaction(gear, vlength)), x = mean\_price, fill = species)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 facet\_wrap(~ gear + vlength, scales = "free", ncol = 4) +  
 theme\_minimal() +  
 scale\_y\_reordered() +  
 theme(legend.position = "none")+  
 labs(title = "Average price by species and port", x = "Mean price (€)", y = "Species")

## `summarise()` has grouped output by 'species', 'gear'. You can override using  
## the `.groups` argument.



## Conclusion

Overall, the disaggregation protocol developed under WP2 provides a replicable and adaptable framework to enhance the spatial granularity and analytical potential of fisheries socio-economic data. While subject to certain limitations—chiefly related to data availability and resolution—the approach demonstrates the feasibility of integrating heterogeneous data sources (FDI, AER, GFW, and registries) to derive vessel-level and port-level indicators. This not only improves the precision of fisheries monitoring and assessment, but also strengthens the foundational data layer required for advanced socio-ecological modeling within the Digital Twin of the Ocean. Future work will focus on validating this methodology across multiple regions and fleet segments, with the goal of operationalizing it within broader simulation workflows and policy support tools.