Fishery effort indicators - Appendix 4 from ?

July 2021

Overview

Details regarding the fishery effort data extraction, formatting, and summarization for the Ecosystem and Climate Indicators Report Card presented to SC17. This work is intended to be ongoing, and therefore ensuring consistency and transparency for repeatability was paramount. In addition, we are hoping that this report card will be a living document. These details are provided to encourage discussion and engagement in an effort to craft indicators that are informative and valuable for the members.

Load R packages

```
library(knitr)

library(tidyverse)
library(magrittr)
library(maps)
library(zoo)
library(RODBC)
library(sp)
library(RGeostats)
```

Data Extraction

This section described the data extraction command for the in-house databases (S_BEST and L_BEST) at SPC, and is for internal transparency for future iterations of this ecosystem and climate indicators report card.

```
################################################################################
# Metrics of purse seine and longline effort
  # Temporal: Annual metrics & 5 year rolling average
  # Metrics:
    # 1. central tendency and inertia (All PS, LL; FAD sets; free-school sets)
    # 2. distributional spread (All PS, LL; FAD sets; free-school sets)
################################################################################

## S_BEST purse seine catch and effort
s_eff_query <- "
SELECT      s.S_BEST_ID, YY, MM, lat_short, lon_short, ez_aprx_code, days,
            sh.Schass_code, days_sch, sets_sch
FROM        best.S_BEST_AGG s
LEFT OUTER JOIN  best.S_BEST_AGG_SCHEFF sh on s.S_BEST_ID = sh.S_BEST_ID
WHERE       s.GEAR_CODE like 'S' and YY>1999 and YY<2020
```

```
"
s_eff <- sqlQuery(channel, s_eff_query, as.is=TRUE)

s_catch_query <- "
SELECT      s.S_BEST_ID, YY, MM, lat_short, lon_short, ez_aprx_code, days,
            Schass_code, sp_code, sp_mt
FROM        best.S_BEST_AGG s
LEFT OUTER JOIN  best.S_BEST_AGG_CATCH c on s.S_BEST_ID = c.S_BEST_ID
WHERE       s.GEAR_CODE like 'S' and YY>1999 and YY<2020
"
s_catch <- sqlQuery(channel, s_catch_query, as.is=TRUE)


## L_BEST longline catch and effort
l_eff_query <- "
SELECT      L_BEST_ID, YY, MM, lat_short, lon_short, hhooks, std_effort
FROM        best.L_BEST_AGG
WHERE       GEAR_CODE like 'L' and YY>1999 and YY<2020
"
l_eff <- sqlQuery(channel, l_eff_query, as.is=TRUE)

l_catch_query <- "
SELECT      l.L_BEST_ID, YY, MM, lat_short, lon_short, sp_code, sp_n, sp_mt
FROM        best.L_BEST_AGG l
LEFT OUTER JOIN  best.L_BEST_AGG_CATCH c on l.L_BEST_ID = c.L_BEST_ID
WHERE       l.GEAR_CODE like 'L' and YY>1999 and YY<2020
"
l_catch <- sqlQuery(channel, l_catch_query, as.is=TRUE)

# Close the database connection
odbcCloseAll()
```

Spatial domain

Filter data to include only information from the WCPFC.

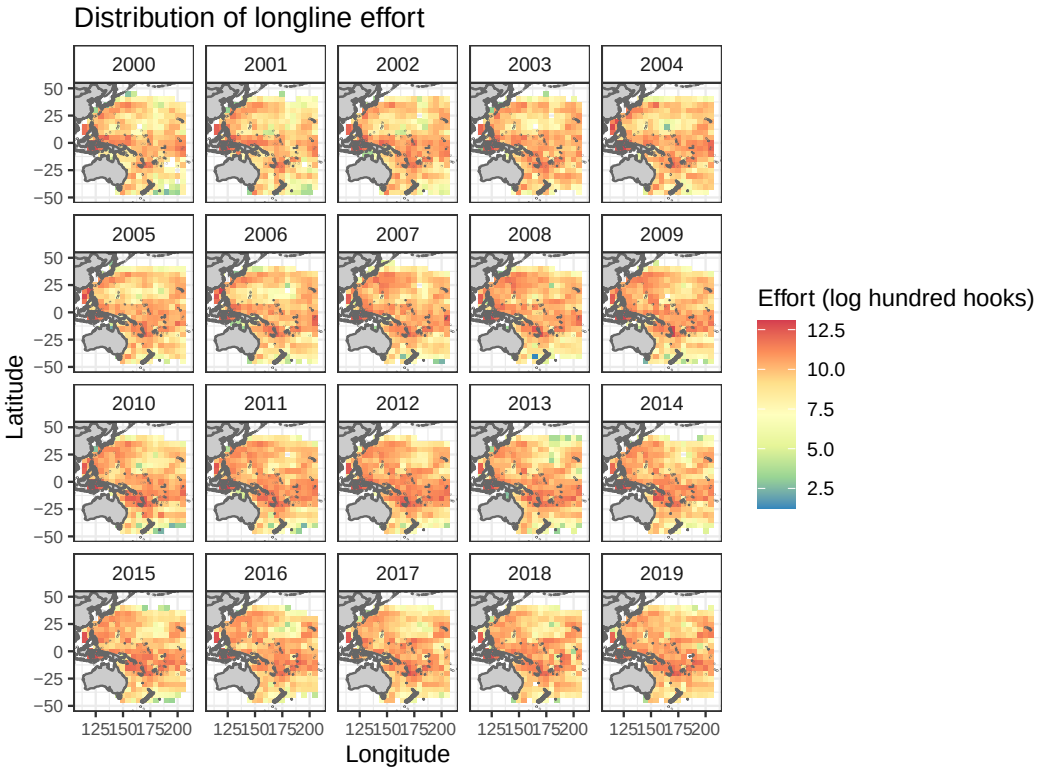
```
#########################################################
# Read in WCPO boundary info
#########################################################
wcp_ca = read.csv("wcpfc_ca_stat_area.csv")
# Convert to spatial polygon
wcp = Polygon(wcp_ca)
wcp = SpatialPolygons(list(Polygons(list(wcp), ID = "CA")),
       proj4string=CRS("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"))

# Check that points are in WCP-CA
coords = wcp@polygons[[1]]@Polygons[[1]]@coords
```

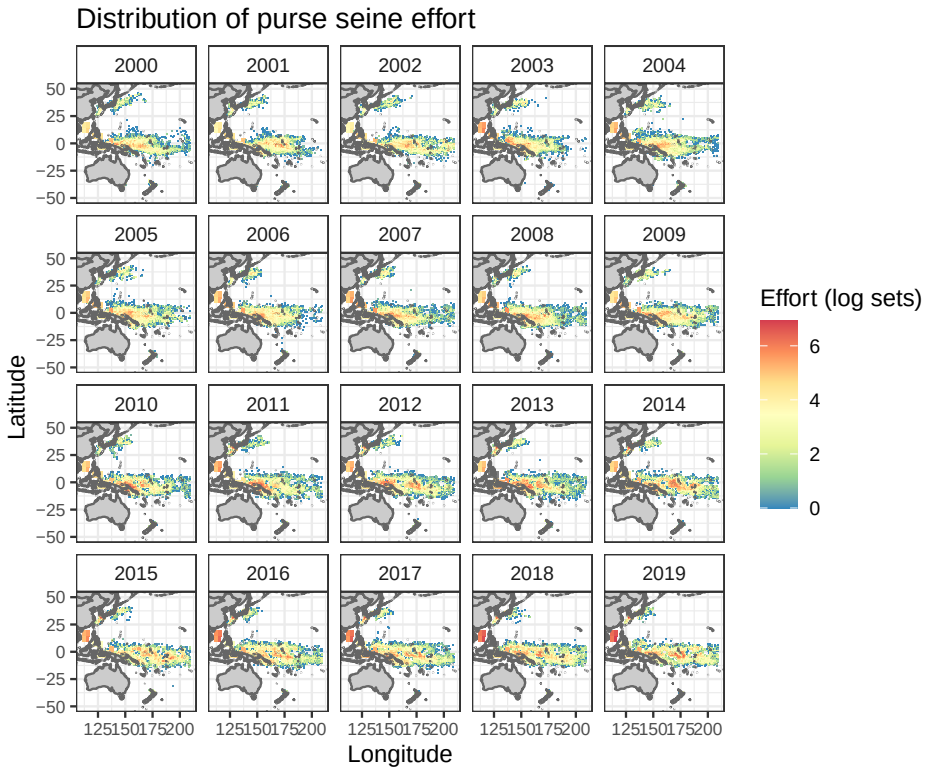
Data manipulation and formatting

Here we are formatting the latitude and longitude positions, and ensuring numeric fields are stored as such. In addition, we are grouping the free-school sets and FAD sets. It should be noted that sets made in associated with whales (dead or alive) or whale sharks are considered free-school sets for this analysis.

```
# Format data and filter out observations outside WCPFC-CA
s_eff %<>% mutate(lat = as.numeric(substr(lat_short,1,2)),
                  lat.dir = substr(lat_short,3,3),
                  lon = as.numeric(substr(lon_short,1,3)),
                  lon.dir = substr(lon_short,4,4),
                  lat = ifelse(lat.dir=='S', lat*-1,lat),
                  lon = ifelse(lon.dir=='W', 360-lon, lon)) %>%
  select(-c(lon_short, lon.dir, lat_short, lat.dir)) %>%
  mutate(days_sch = as.numeric(days_sch), sets_sch = as.numeric(sets_sch))

s_catch %<>% mutate(lat = as.numeric(substr(lat_short,1,2)),
                    lat.dir = substr(lat_short,3,3),
                    lon = as.numeric(substr(lon_short,1,3)),
                    lon.dir = substr(lon_short,4,4),
                    lat = ifelse(lat.dir=='S', lat*-1,lat),
                    lon = ifelse(lon.dir=='W', 360-lon, lon)) %>%
  select(-c(lon_short, lon.dir, lat_short, lat.dir))

l_eff %<>% mutate(lat = as.numeric(substr(lat_short,1,2)),
                  lat.dir = substr(lat_short,3,3),
                  lon = as.numeric(substr(lon_short,1,3)),
                  lon.dir = substr(lon_short,4,4),
                  lat = ifelse(lat.dir=='S', lat*-1,lat),
                  lon = ifelse(lon.dir=='W', 360-lon, lon)) %>%
  select(-c(lon_short, lon.dir, lat_short, lat.dir)) %>%
  mutate(hhoks = as.numeric(hhooks), std_effort = as.numeric(std_effort))

l_catch %<>% mutate(lat = as.numeric(substr(lat_short,1,2)),
                    lat.dir = substr(lat_short,3,3),
                    lon = as.numeric(substr(lon_short,1,3)),
                    lon.dir = substr(lon_short,4,4),
                    lat = ifelse(lat.dir=='S', lat*-1,lat),
                    lon = ifelse(lon.dir=='W', 360-lon, lon)) %>%
  select(-c(lon_short, lon.dir, lat_short, lat.dir))


s_eff$WCP_CA = point.in.polygon(s_eff$lon, s_eff$lat, coords[,1], coords[,2])
  s_eff %<>% filter(WCP_CA %in% c(1,2)) %>%  select(-WCP_CA)
```

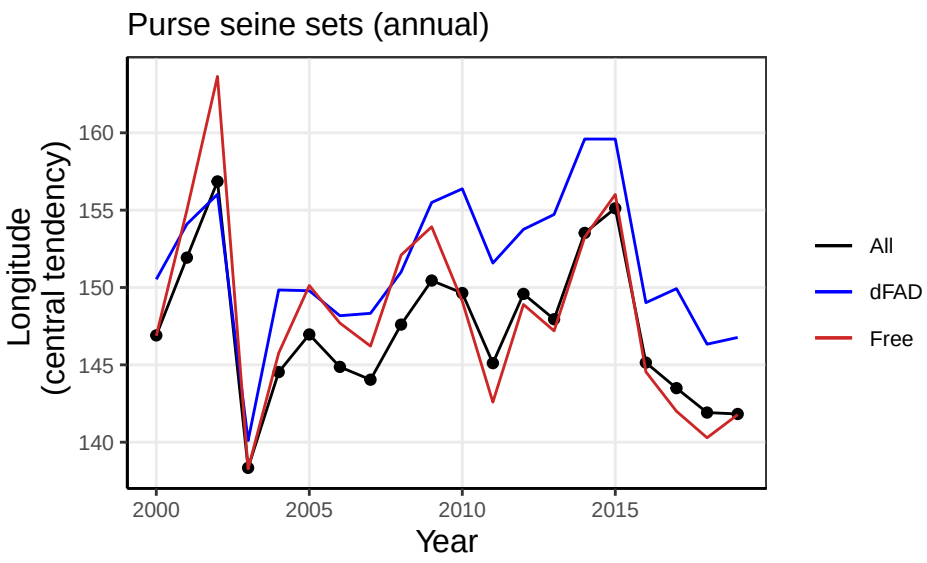
```r
# School association codes
ASS <- c(4,3) # drifting FAD only
AFAD = c(5) # anchored FAD
ASSWH = c(6,7) # whale and whale shark sets
UNA <- c(1,2) # free school


# Aggregate all set types together
s_eff_all = s_eff %>% filter(Schass_code %in% c(1,2,3,4,5,6,7)) %>%
            group_by(YY,MM, lat, lon) %>%
            mutate(sets_sch = ifelse(sets_sch<0,0,sets_sch)) %>%
            summarise(days = sum(days_sch), sets = sum(sets_sch) ) %>% ungroup()


s_eff_fad = s_eff %>% filter(Schass_code %in% ASS) %>%
            mutate(sets_sch = ifelse(sets_sch<0,0,sets_sch)) %>%
            group_by(YY,MM, lat, lon) %>%
            summarise(days = sum(days_sch), sets = sum(sets_sch) ) %>% ungroup()
s_eff_free = s_eff %>%  filter(Schass_code %in% UNA) %>%
            mutate(sets_sch = ifelse(sets_sch<0,0,sets_sch)) %>%
            group_by(YY,MM, lat, lon) %>%
            summarise(days = sum(days_sch), sets = sum(sets_sch) ) %>% ungroup()


s_catch$WCP_CA = point.in.polygon(s_catch$lon, s_catch$lat, coords[,1], coords[,2])
  s_catch %<>% filter(WCP_CA %in% c(1,2)) %>%  select(-WCP_CA)

l_eff$WCP_CA = point.in.polygon(l_eff$lon, l_eff$lat, coords[,1], coords[,2])

l_eff %<>% filter(WCP_CA %in% c(1,2)) %>%  select(-WCP_CA)
l_eff %<>% mutate(hhooks = as.numeric(hhooks))

l_catch$WCP_CA = point.in.polygon(l_catch$lon, l_catch$lat, coords[,1], coords[,2])

l_catch %<>% filter(WCP_CA %in% c(1,2)) %>%  select(-WCP_CA)
```

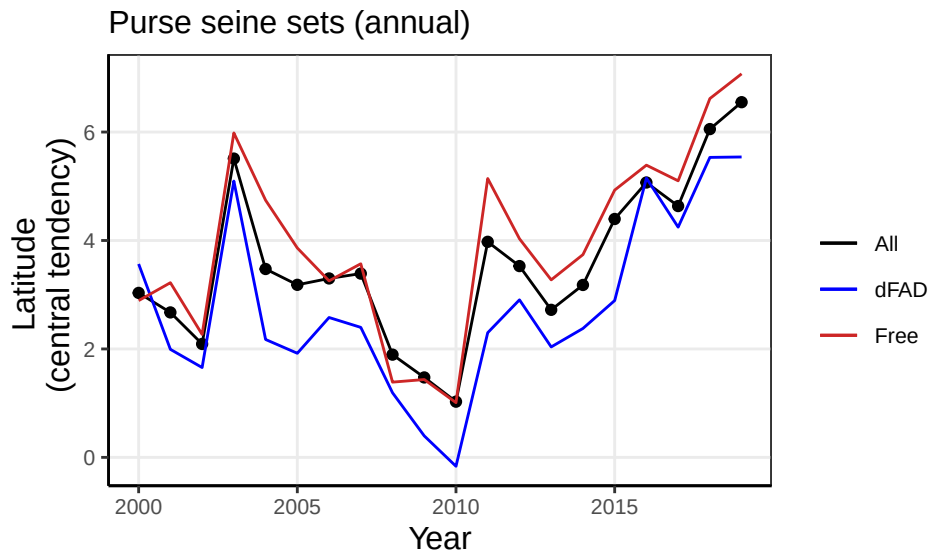


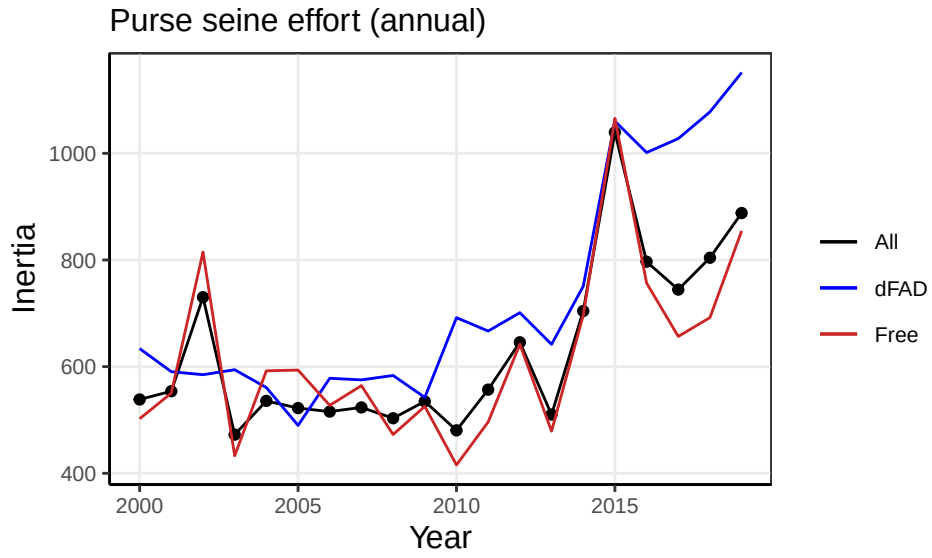


Centre of gravity and inertia

```r
##############################################################################
# Calculate centre of gravity and inertia - annual
##############################################################################

# Calculate annual values
ann.cg.inert = function(dat, effort, lab){
cgi = c()

for(i in 1:length(unique(dat$YY))){
    CG = dat %>% filter(YY==unique(YY)[i]) %>% select(lon,lat,effort,YY)
    names(CG)[3] = 'effort'
    CG %<>%  group_by(lon,lat) %>% summarize(z=sum(as.numeric(effort),na.rm=T))
    db = db.create(x1=CG$lon, x2=CG$lat, z1=CG$z)
    projec.define(projection="mean",db=db)

    projec.toggle(0)
    CG = SI.cgi(db, flag.plot=F)
    cgi  = rbind(cgi, c("yy"=unique(dat$YY)[i], "CG"=CG$center,
                        "inertia"=CG$inertia, "iso"=CG$iso))
}
cgi %<>% data.frame() %>% rename(lon=CG1, lat=CG2)
return(cgi)
}

ps.cg = ann.cg.inert(s_eff_all, effort='sets', lab='ps_annual')
fad.cg = ann.cg.inert(s_eff_fad, effort='sets', lab='fad_annual')
free.cg = ann.cg.inert(s_eff_free, effort='sets', lab='free_annual')
```



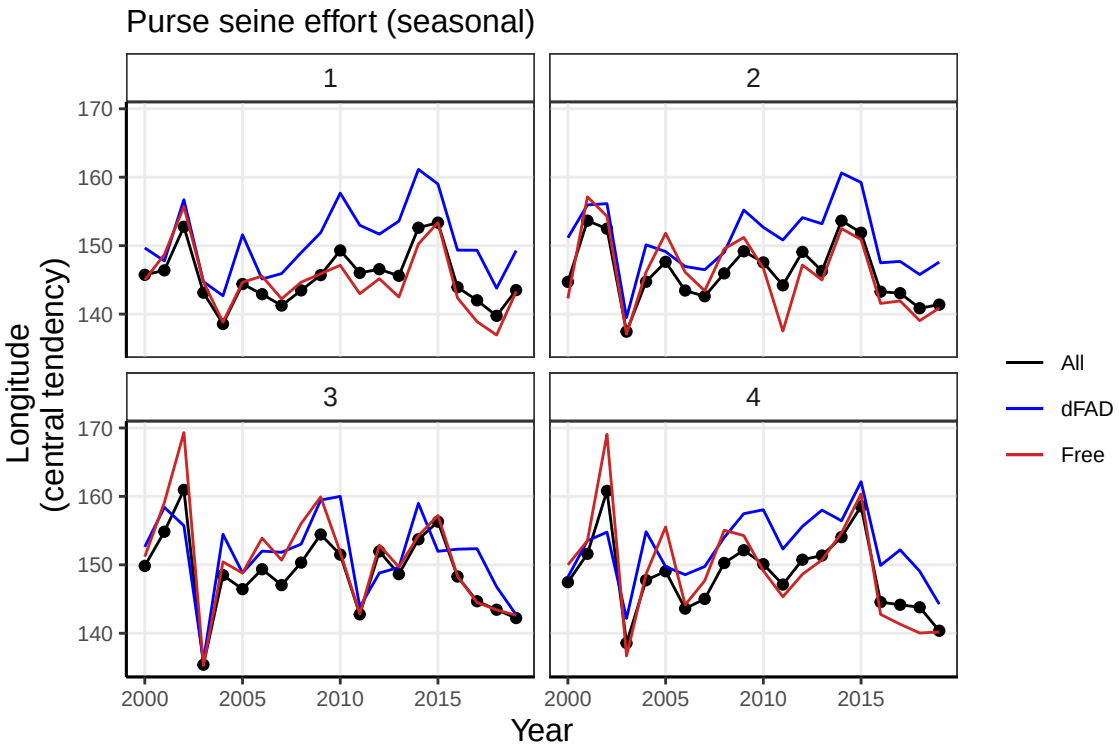


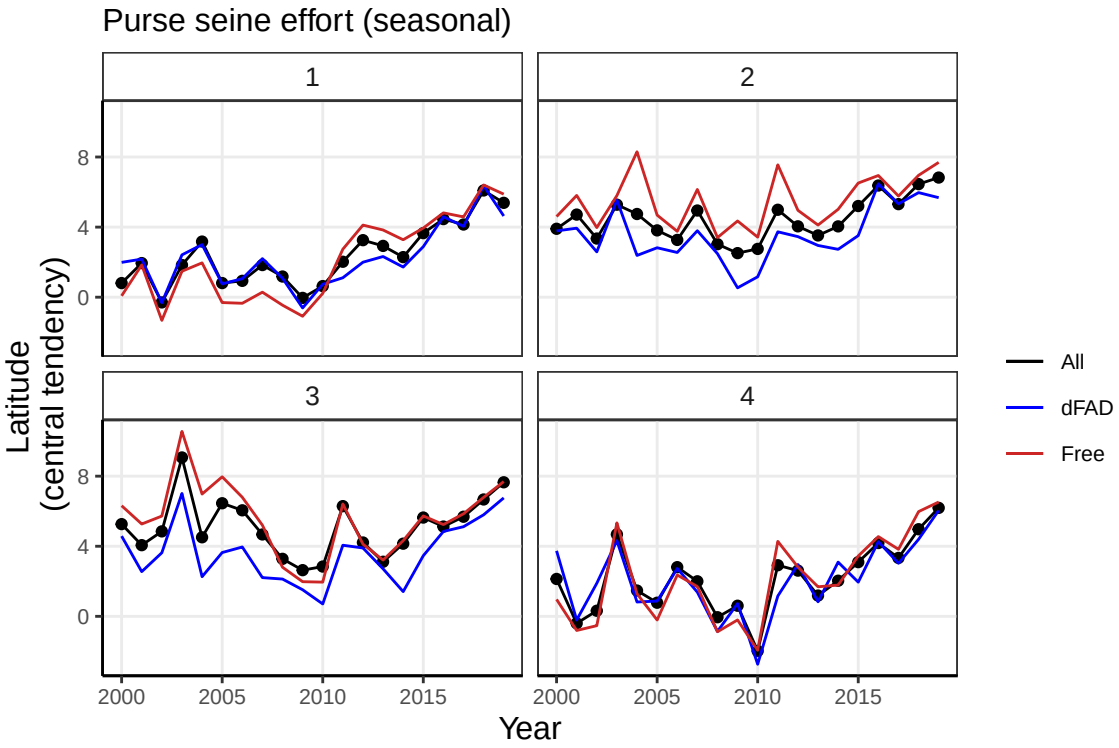


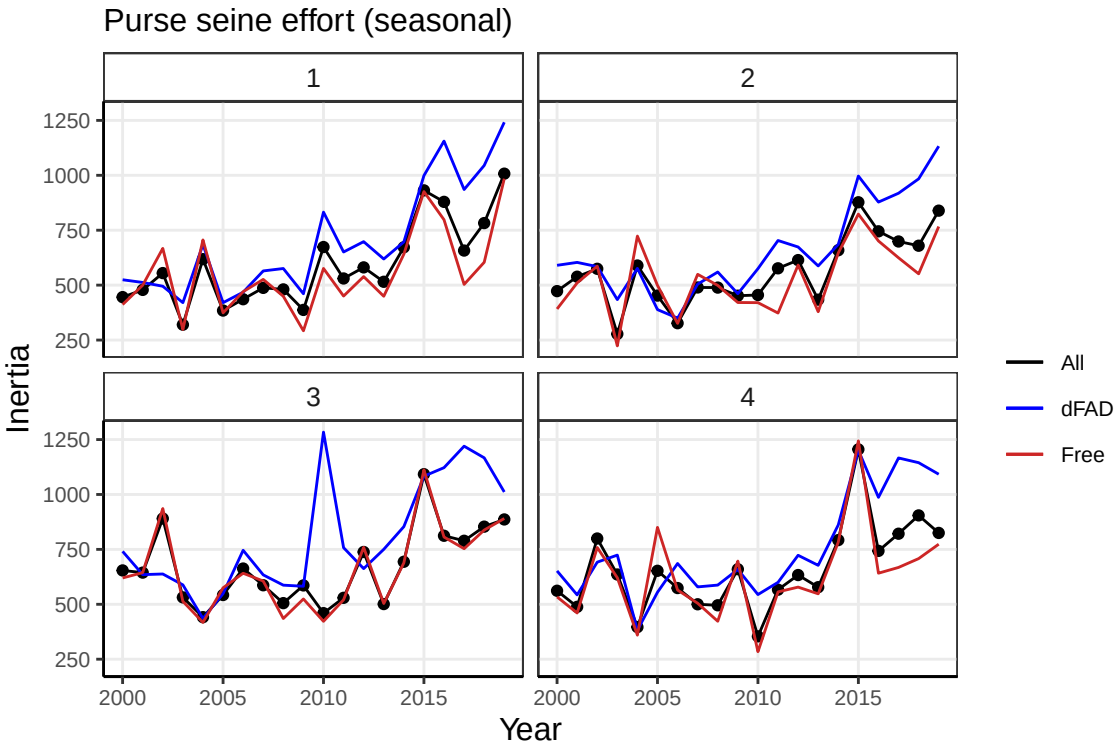
Calculate centre of gravity on a seasonal basis (year-quarter).

```
# Calculate seasonal centre of gravity and inertia
seas.cg.inert = function(dat, effort){
cgi = c()

for(i in 1:length(unique(dat$YY))){
  for(s in unique(dat$qtr)){
    CG = dat %>% filter(YY==unique(YY)[i], qtr==s) %>%
                 select(lon,lat,effort,YY,qtr)
    names(CG)[3] = 'effort'
    CG = db.create(x1=CG$lon, x2=CG$lat, z1=CG$effort)
    projec.toggle(0)
    CG = SI.cgi(CG, flag.plot=F)
    cgi  = rbind(cgi, c("yy"=unique(dat$YY)[i], "qtr" = s, "CG"=CG$center,
```

```
                          "inertia"=CG$inertia, "iso"=CG$iso))
  }
}
cgi %<>% data.frame() %>% rename(lon=CG1, lat=CG2)
return(cgi)
}

ps.seas.cg = seas.cg.inert(s_eff_all %>%
                             mutate(qtr = ceiling(MM/3)), effort='sets')
fad.seas.cg = seas.cg.inert(s_eff_fad %>%
                              mutate(qtr = ceiling(MM/3)), effort='sets')
free.seas.cg = seas.cg.inert(s_eff_free %>%
                               mutate(qtr = ceiling(MM/3)), effort='sets')
```



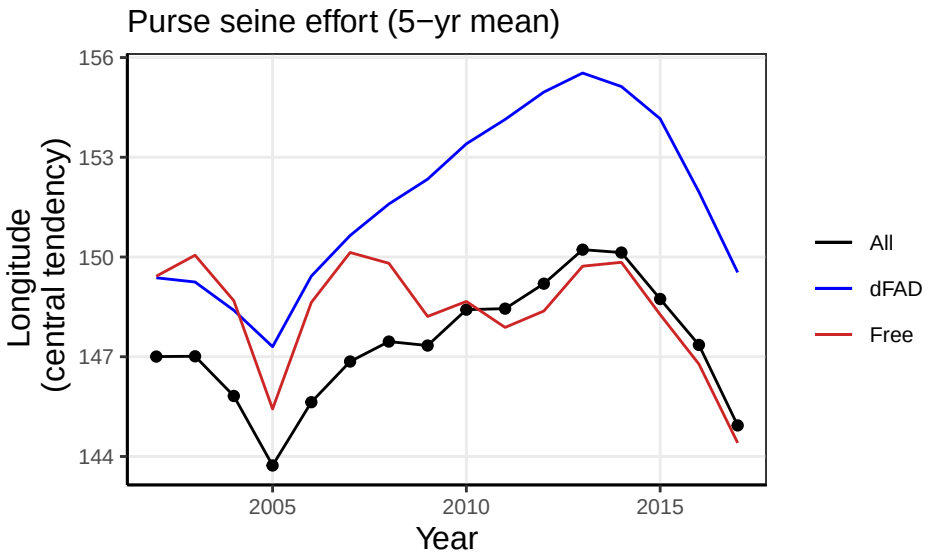


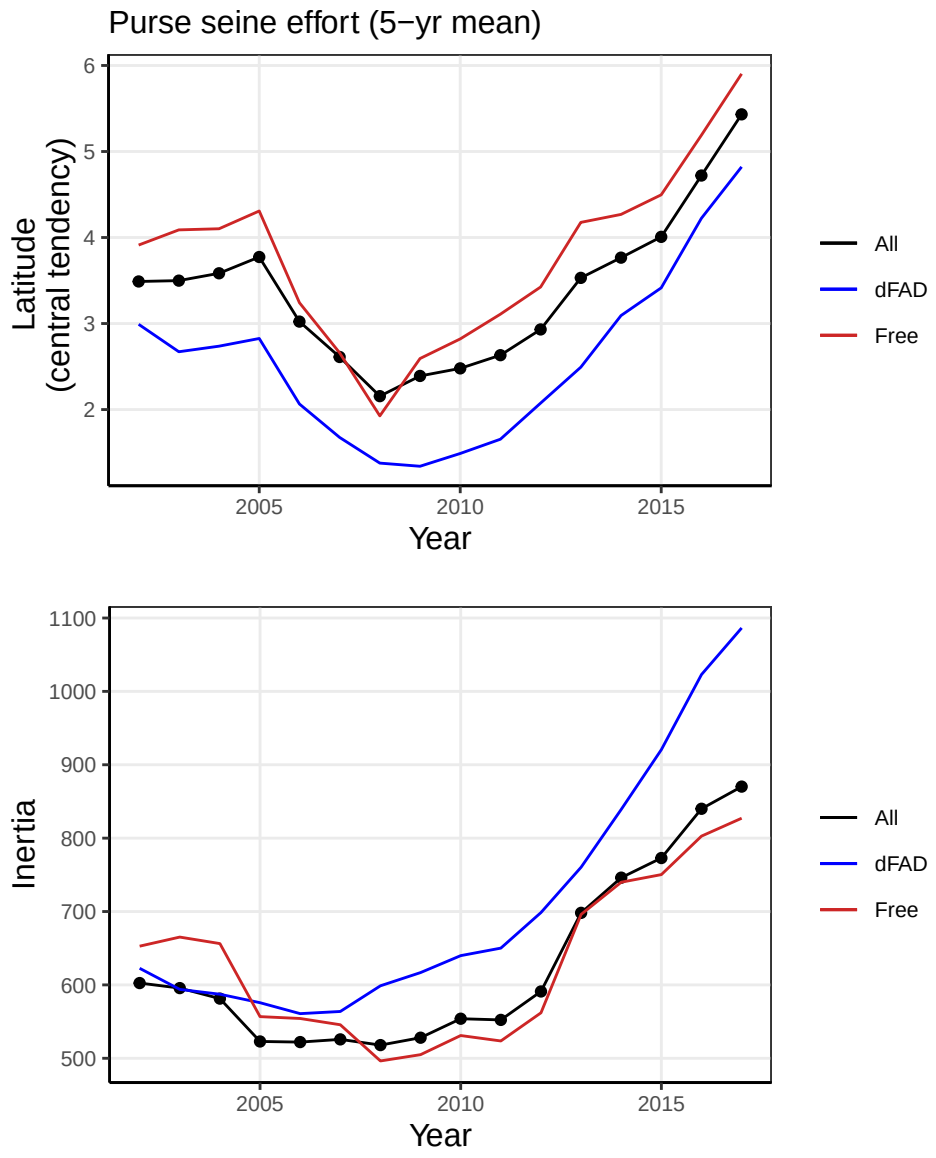


Calculate centre of gravity on a 5-year rolling average.

```
##############################################################
# 5-yr rolling average - with mid-point year as reference
```

```
#############################################################

roll5.cg.inert = function(dat, effort){
  cgi = c()

  for(i in 1:(length(unique(dat$YY))-4)){
    CG = dat %>% filter(YY %in% c(unique(YY)[c(i:(i+4))])) %>%
                 select(lon,lat,effort,YY)
    names(CG)[3] = 'effort'
   CG = db.create(x1=CG$lon, x2=CG$lat, z1=CG$effort)
   projec.toggle(0)
   CG = SI.cgi(CG, flag.plot=F)

    cgi  = rbind(cgi, c("yy"=unique(dat$YY)[i+2], "CG"=CG$center,
                        "inertia"=CG$inertia, "iso"=CG$iso))
  }

  cgi %<>% data.frame() %>% rename(lon=CG1, lat=CG2)
  return(cgi)
}

ps.cg5 = roll5.cg.inert(s_eff_all, effort='sets')
fad.cg5 = roll5.cg.inert(s_eff_fad, effort='sets')
free.cg5 = roll5.cg.inert(s_eff_free, effort='sets')
```

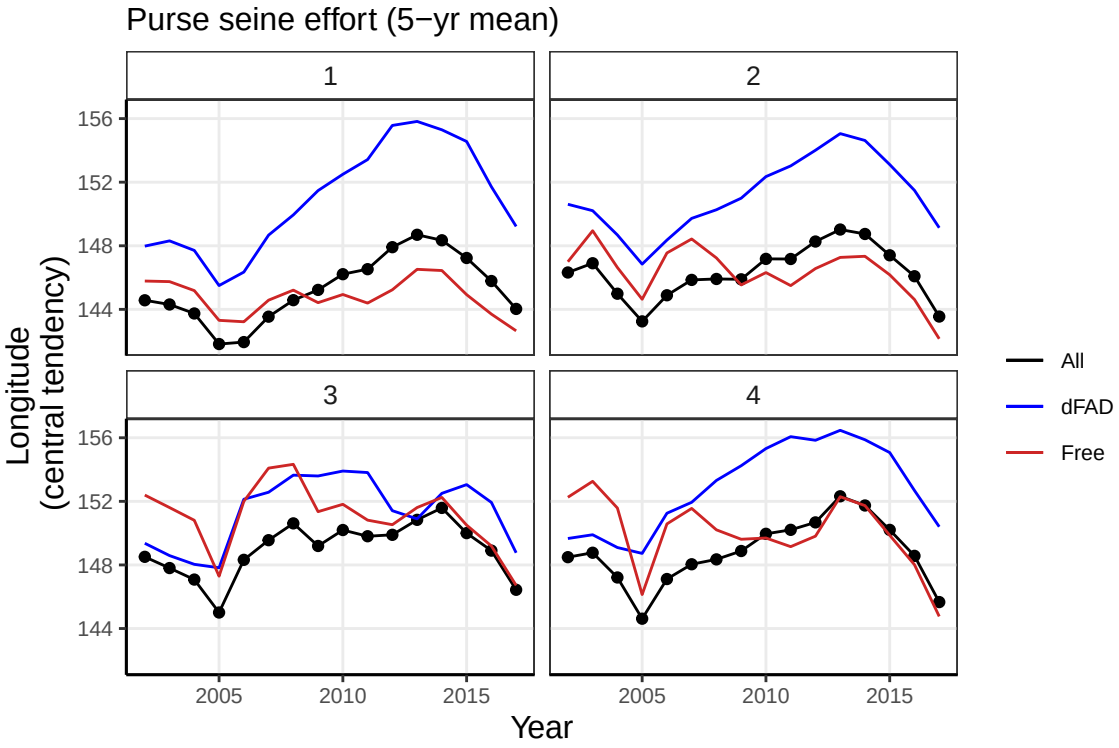


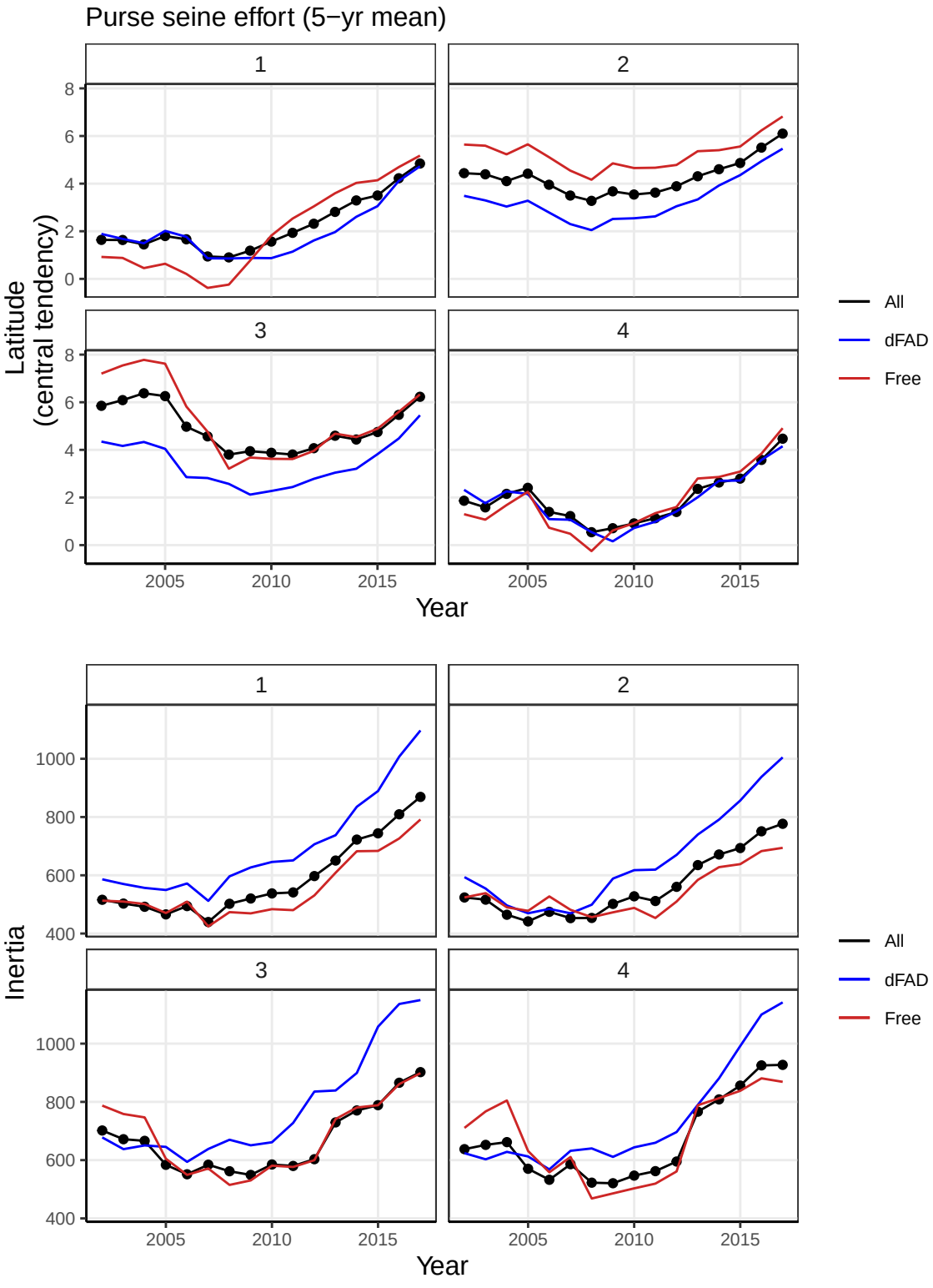


Centre of gravity calculated as a 5-year rolling average, seasonally.

```
roll5.seas.cg.inert = function(dat, effort){
  cgi = c()

  for(i in 1:(length(unique(dat$YY))-4)){
    for(s in unique(dat$qtr)){
    CG = dat %>% filter(YY %in% c(unique(YY)[c(i:(i+4))]), qtr==s) %>%
                 select(lon,lat,effort,YY, qtr)
    names(CG)[3] = 'effort'
    CG = db.create(x1=CG$lon, x2=CG$lat, z1=CG$effort)
    projec.toggle(0)
    CG = SI.cgi(CG, flag.plot=F)
    cgi  = rbind(cgi, c("yy"=unique(dat$YY)[i+2],'qtr'=s, "CG"=CG$center,
                        "inertia"=CG$inertia, "iso"=CG$iso))
```

```
    }
  }

  cgi %<>% data.frame() %>% rename(lon=CG1, lat=CG2)
  return(cgi)
}


ps.seas.cg5 = roll5.seas.cg.inert(s_eff_all %>%
                                    mutate(qtr = ceiling(MM/3)), effort='sets')
fad.seas.cg5 = roll5.seas.cg.inert(s_eff_fad %>%
                                     mutate(qtr = ceiling(MM/3)), effort='sets')
free.seas.cg5 = roll5.seas.cg.inert(s_eff_free %>%
                                      mutate(qtr = ceiling(MM/3)), effort='sets')
```

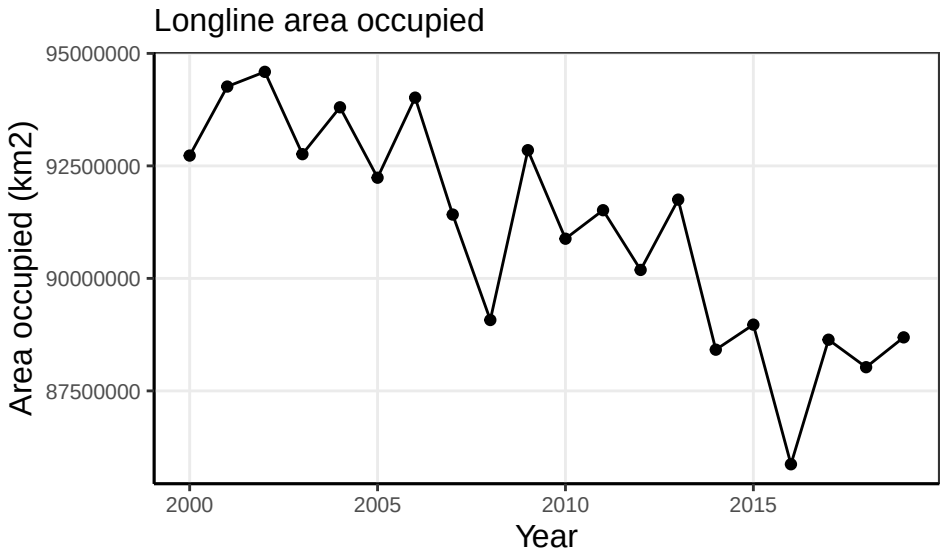


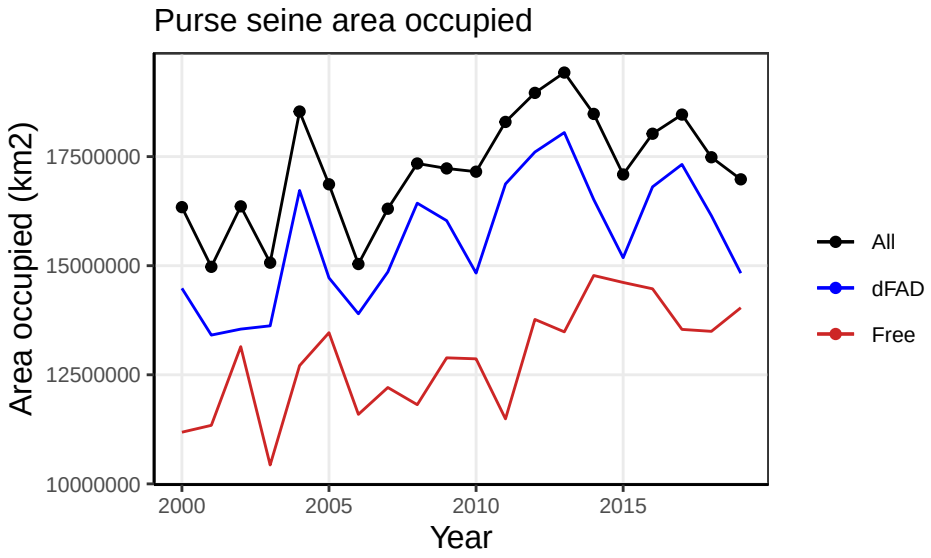

Purse seine effort (5−yr mean)
1
2
3
4
Latitude (central tendency)
Inertia
Year
All
dFAD
Free

Area occupied

Here we calculate the area occupied by each of the fisheries, on an annual basis.

```
#########################################################################
# Area occupied - unique 1x1 cells fished
#########################################################################

area.occ = function(dat, effort, res){
    library(raster)
    # get area of each cell
    grid.df <- expand.grid('Lat'=seq(min(dat$lat),max(dat$lat),res),
                           'Lon'=seq(min(dat$lon),max(dat$lon),res))
    r <- raster(ext = extent(min(dat$lon), max(dat$lon)+res, min(dat$lat),
                             max(dat$lat)+res), res=c(res,res))
    grid.df$Area_km2 =  area(r)@data@values
    detach('package:raster')

    ao = dat %>% filter(effort>0) %>%
          left_join(grid.df %>%  rename(lat=Lat, lon=Lon)) %>%
          mutate(cell = paste(lon, lat, sep=':')) %>%
          select(YY, cell, Area_km2) %>% distinct() %>%
          group_by(YY) %>%
          summarise(cells = n_distinct(cell), area = sum(Area_km2)) %>%
          mutate(cells5 = rollapply(cells,5,mean, align='center',fill=NA),
                 area5 = rollapply(area,5,mean, align='center',fill=NA))
    return(ao)

}

ll.ao = area.occ(l_eff, effort='hhooks', res=5) # res = spatial resolution 5x5
ps.ao = area.occ(s_eff_all, effort='sets', res=1) # spatial resolution of 1x1
free.ao = area.occ(s_eff_free, effort='sets', res=1) # spatial resolution of 1x1
fad.ao = area.occ(s_eff_fad, effort='sets', res=1) # spatial resolution of 1x1
```


Longline area occupied
Area occupied (km2)
95000000
92500000
90000000
87500000
2000
2005
2010
2015
Year


Purse seine area occupied
Area occupied (km2)
17500000
15000000
12500000
10000000
2000
2005
2010
2015
Year
All
dFAD
Free

References