Yellowtail_Kingfish_SDM_migration

EcoCommons

04/10/2022

We recommend going to this google link and downloading the whole of the "Marine" folder and then putting it in your local directory. You may need to merge multiple folders from the download, the raw data files are large. https://drive.google.com/drive/folders/19YemtBtbZbdtwDYcorKOyzlBCNHvRv9h?usp=sharing

```
getwd()
```

```
## [1] "/Users/s2992269/Downloads/Marine"
```

You may need to reset your local directory, and keep in mind directories are treated differently in R, Rmarkdown, and Jupyter

setwd("~/Downloads/Marine") # set this to your local directory if needed

You can download records directly from the Atlas of Living Australia and this includes IMOS records. The exact data we used to generate the use case is available through the google line:41 below. Keep in mind with stochastic learning algorithms like Maxent, results may vary somewhat from run to run, even if using the same data.

Again, we recommend downloading directly from a large database each time you start a new model, but here we got some additional records from AODN which are not in the ALA data, so use the google link (line 41) to replicate our results. Note our ALA download includes WA near-coastal records, and does not include SA near-coastal data which were in the cleaned IMOS AODN data.

Below we provide an ALA download example

There are a few steps you need to take to download records from ALA First, you need to register with ALA you can register here: https://auth.ala.org.au/userdetails/registration/createAccount then run the config function in R galah_config(email = "your-email@email.com") This email needs to be registered with ALA

```
require(galah)
require(terra)
require(tidyverse)
require(googledrive)
require(purrr)
```

```
galah_config(email = "r.clemens@griffith.edu.au") # replace my email with the email you registered to A
```

```
## These configuration options will only be saved for this session.
##     Set ‘preserve = TRUE‘ to preserve them for future sessions.
```

```
YT <- galah_call() %>%
  galah_identify("Seriola lalandi")%>%
  galah_filter(basisOfRecord == "MACHINE_OBSERVATION" | basisOfRecord == "HUMAN_OBSERVATION")%>%
  galah_filter(coordinateUncertaintyInMeters < 501)%>%
  galah_filter(year > 2011)%>%
  galah_select("month")%>%
  atlas_occurrences()
```

```
## This query will return 4605 records
```

```
##   |                                                                              |
```

Use atlas_counts() instead of atlas_occurrences() if you want just a count of the number of records in your query

To use the same occurrence data as we did in the marine use case, download the data from Google, link below. We use direct links to get a google drive file url which will download https://sites.google.com/site/gdocs2direct/

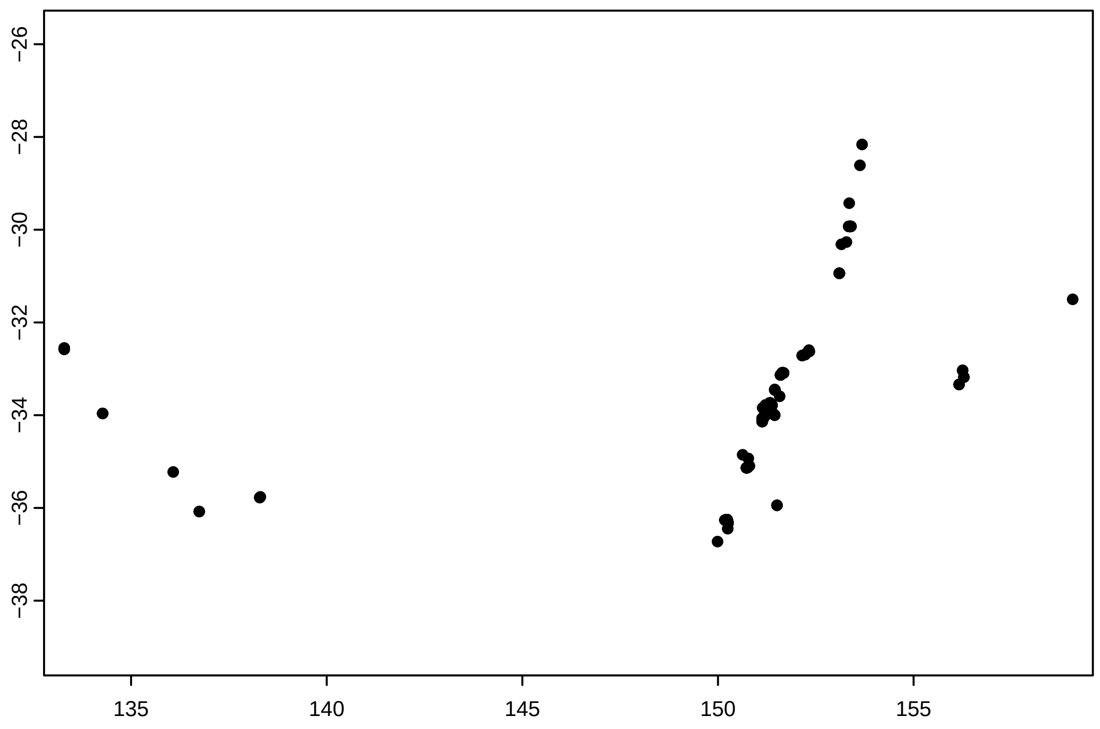
```
YT <- read.csv("https://drive.google.com/uc?export=download&id=1CRv5ZWIS-jWwnz9VH4kng1A_Ts053wA_", strin
# when checking the data, there is a negative latitude which plots to an odd place, so we will filter o
YT_2 <- YT[YT$lon >0,]

YT_pts <- terra::vect(YT_2, geom=c("lon", "lat")) # this turns the dataframe into a spatial points laye

# look at your data
YT_pts
```

```
##  class       : SpatVector
##  geometry    : points
##  dimensions  : 4445, 6  (geometries, attributes)
##  extent      : 133.291, 159.0657, -36.72587, -28.162  (xmin, xmax, ymin, ymax)
##  coord. ref. :
##  names       :   vernacularName         species coordinateUncertaintyInMeters
##  type        :            <chr>           <chr>                         <num>
##  values      : Yellowtail King~ Seriola lalandi                             2
##                Yellowtail King~ Seriola lalandi                             4
##                Yellowtail King~ Seriola lalandi                             4
##   year month     basisOfRecord
##  <int> <int>             <chr>
##   2019     2 HUMAN_OBSERVATION
##   2016     3 HUMAN_OBSERVATION
##   2020     3 HUMAN_OBSERVATION
```

```
plot(YT_pts)
```

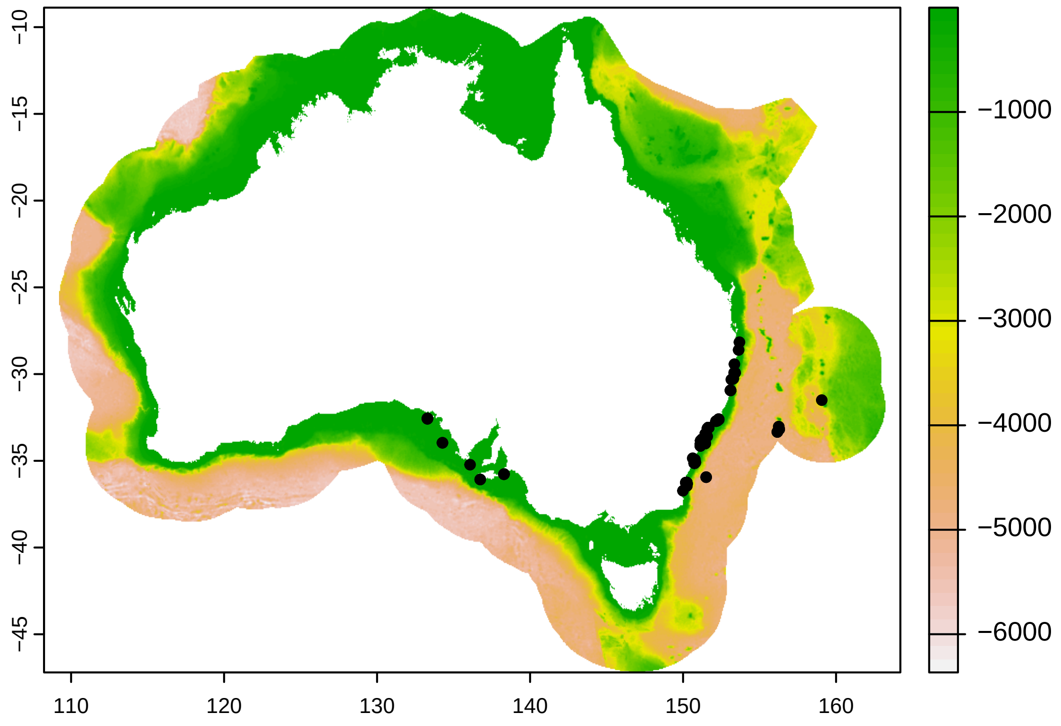


Now we are going to upload some bathemetry data

```r
bath <- rast("https://drive.google.com/uc?export=download&id=1uSOShdrsuz4OBmzGu0yvyP7XjRlrQnfS")
crs(bath)
```

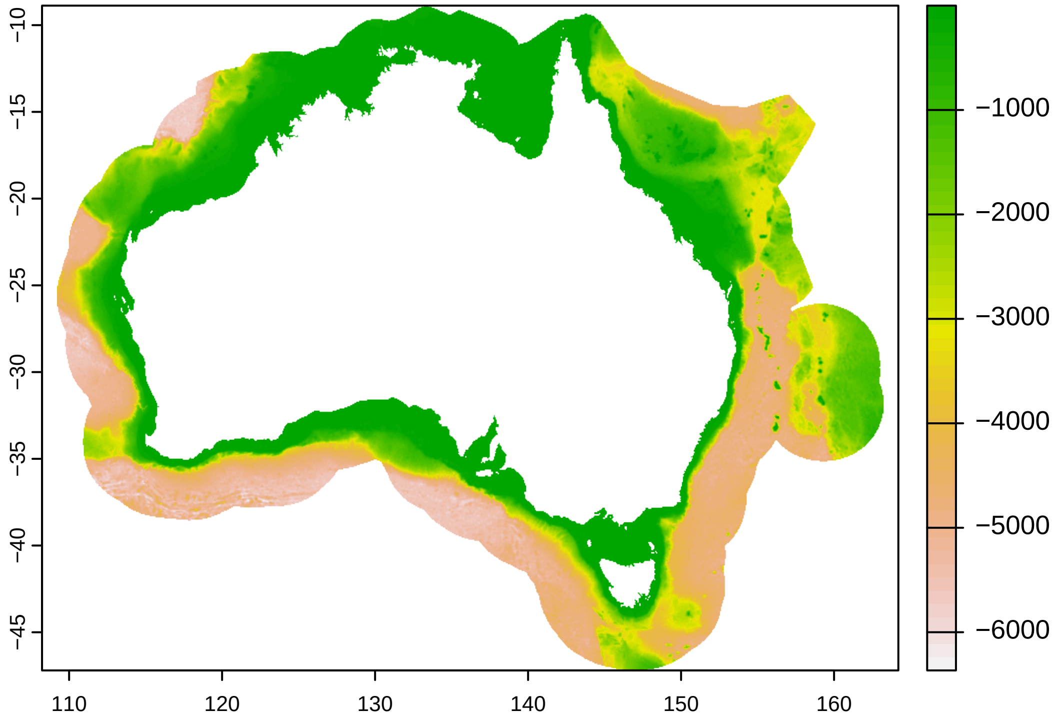
```
## [1] "GEOGCRS[\"WGS 84\",\n    DATUM[\"World Geodetic System 1984\",\n        ELLIPSOID[\"WGS 84\",63
```

```r
plot(bath)
points(YT_pts)
```



Next we use a focal function on the edge of the raster to estimate bathemetry values at near-coastal areas where bathymetry data is missing by specifying arguments (na.policy="only",na.rm=T), by averaging values within 7 grid cells

```
wt <- matrix (data = 1, nrow = 7, ncol = 7)
bath2 <- terra::focal(bath, w=wt, fun="mean", na.policy="only", na.rm=T)
plot(bath2)
```



Next we verify that we now have bathemetry data at all the sensor locations, including those in shallow water

```
YT_bath <- terra::extract(bath2,YT[,c(4,3)])
summary(YT_bath)
```

```
##        ID         focal_mean
##  Min.   :   1   Min.   :-4807.000
##  1st Qu.:1112   1st Qu.:  -12.000
##  Median :2224   Median :   -1.143
##  Mean   :2224   Mean   :  -12.219
##  3rd Qu.:3335   3rd Qu.:   -1.000
##  Max.   :4446   Max.   :   -1.000
##                 NA's   :1
```

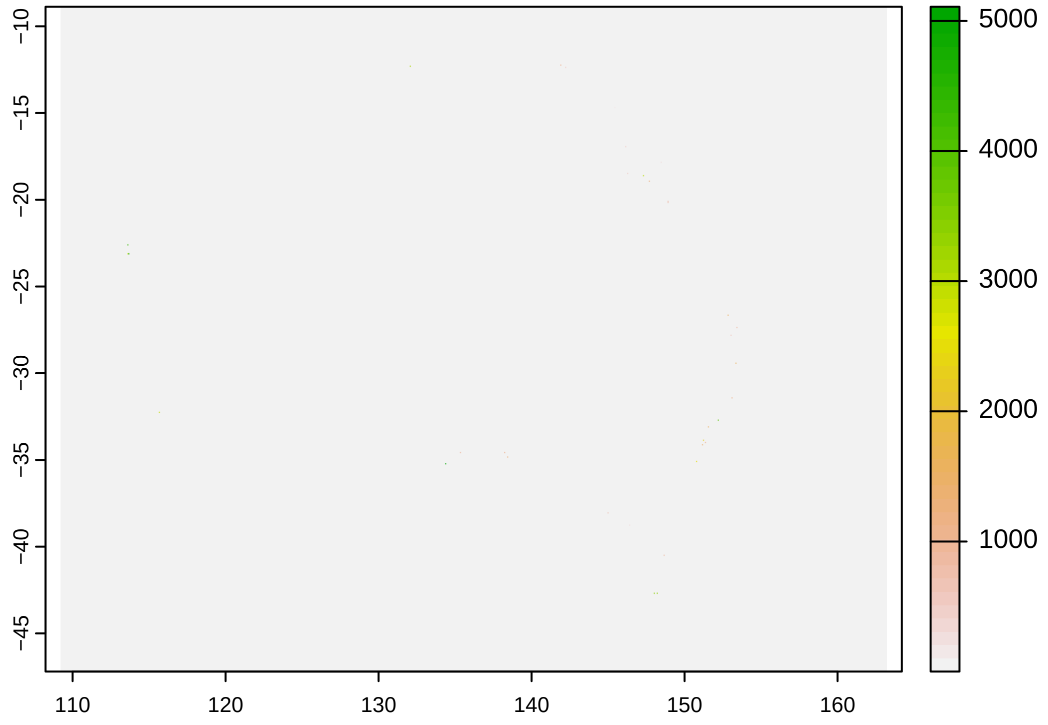
```
# then write the result as a GeoTif
dir.create("data1", showWarnings = FALSE)
writeRaster(bath2,paste0(getwd(),"/",("data1/bathemetry.tif")),overwrite=TRUE)
```

Note: we dropped the 'ucur' variable after initial analysis (it added little to the result). We also dropped the distance from the coast & bathymetry variables (both resulted in under-prediction far from the coast)

Below we show one example of how to geneerate a BIAS layer

AODN sensors are deployed at shallow areas around the Australian continent and the sensors detect any tagged fish that swims by a sensor. Evironmental conditions far from the locations where sensors have been deployed are unlikely to be represented in the occurrence data.

```
stations <- read.csv("IMOS_array_summarised.csv")
st_xy <- cbind(stations$lon,stations$lat)
colnames(st_xy)<- c("Long","Lat")
days <- stations$days_deployed
b1 <- terra::rasterize(st_xy,bath2,days,fun=sum,background=1)
plot(b1)
```



```
b1
```

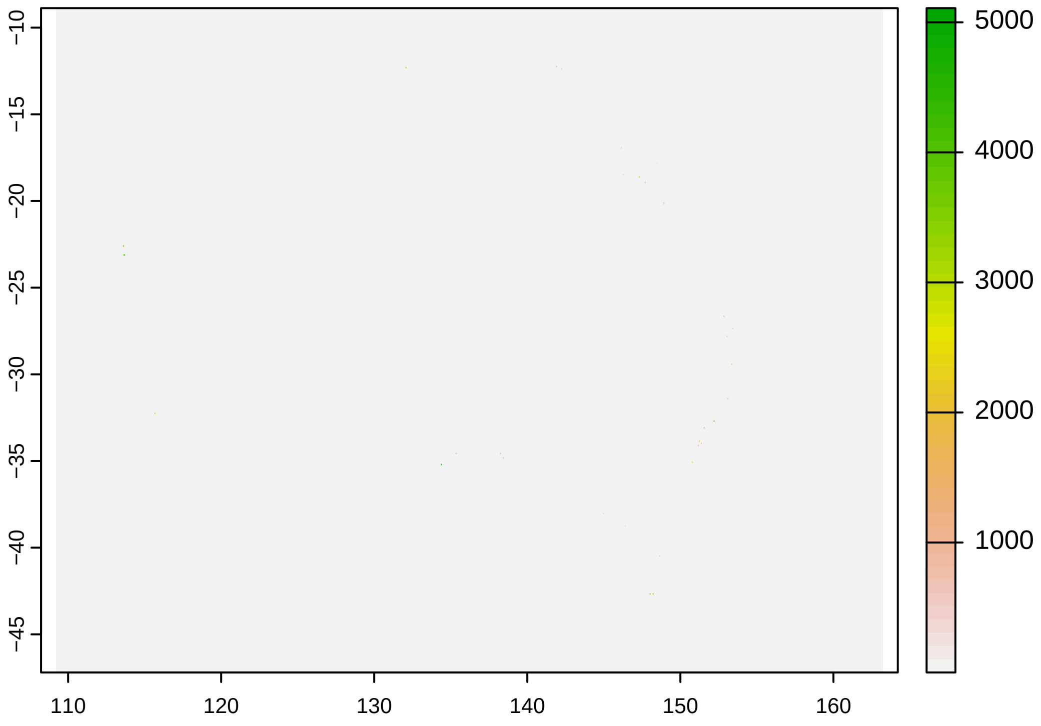
```
## class       : SpatRaster
## dimensions  : 4599, 6476, 1  (nrow, ncol, nlyr)
## resolution  : 0.008333333, 0.008333333  (x, y)
## extent      : 109.2333, 163.2, -47.2, -8.875  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat WGS 84 (EPSG:4326)
## source      : memory
## name        :   focal_mean
## min value   : 0.0006944444
## max value   :     29357.85
```

```
summary(b1)
```

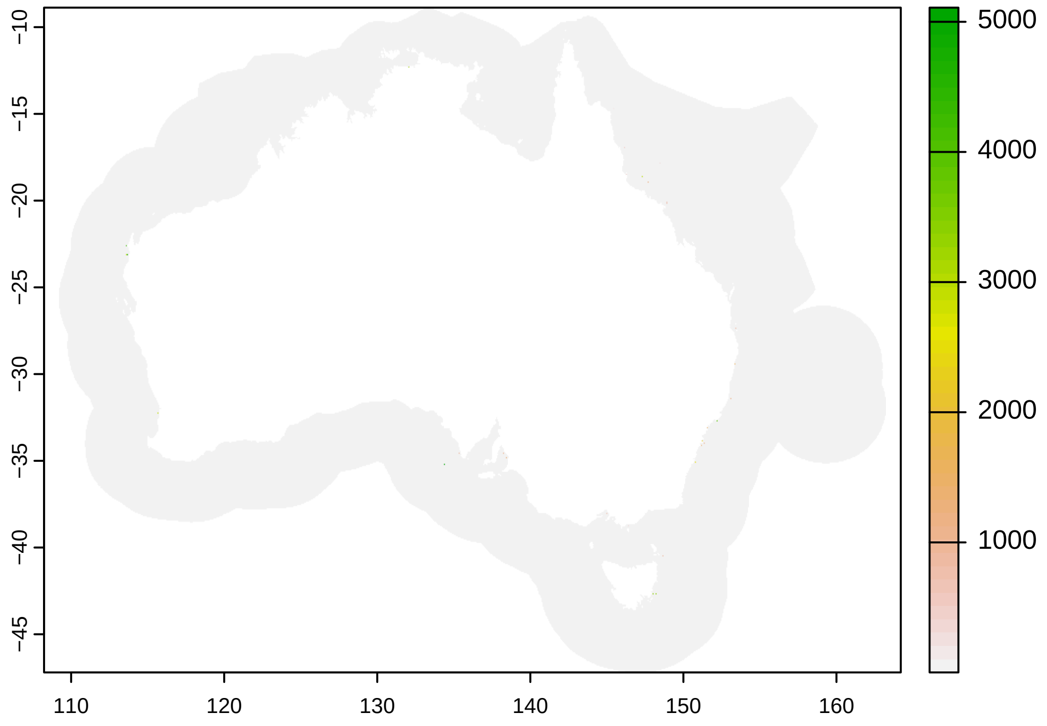
```
## Warning: [summary] used a sample
```

```
##    focal_mean
##  Min.   :   1.00
##  1st Qu.:   1.00
##  Median :   1.00
##  Mean   :   1.18
##  3rd Qu.:   1.00
##  Max.   :5597.61
```

```
b2 <- terra::project(b1,bath2)
plot(b2)
```



```
b3<-mask(b2,bath2)
plot(b3)  # if there had been enough sampling in enouch cells this could be the bias layer
```



```
terra::ext(b3)
```

```
## SpatExtent : 109.233333333, 163.2, -47.2, -8.875 (xmin, xmax, ymin, ymax)
```

```
lg<-c(109.2333,163.2)
lt<-c(-47.2,-8.875)
coordss<-cbind(lg,lt)
st_xy2<-rbind(st_xy,coordss) # this step ensures we have coordinates
#  needed to calculate a kernal density for the entire extent
```

Now we use a two dimensional density kernal to capture spatial bias in sampling of environmental space (geographic distance is used as proxy of environmental space)

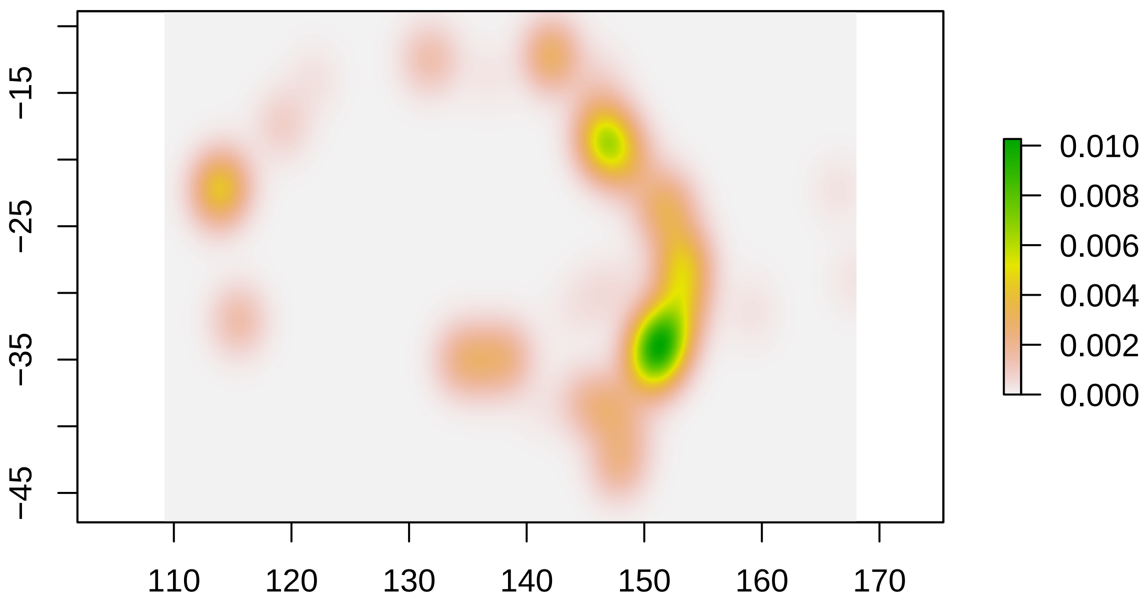
```
require(MASS)
```

```
## Loading required package: MASS
```

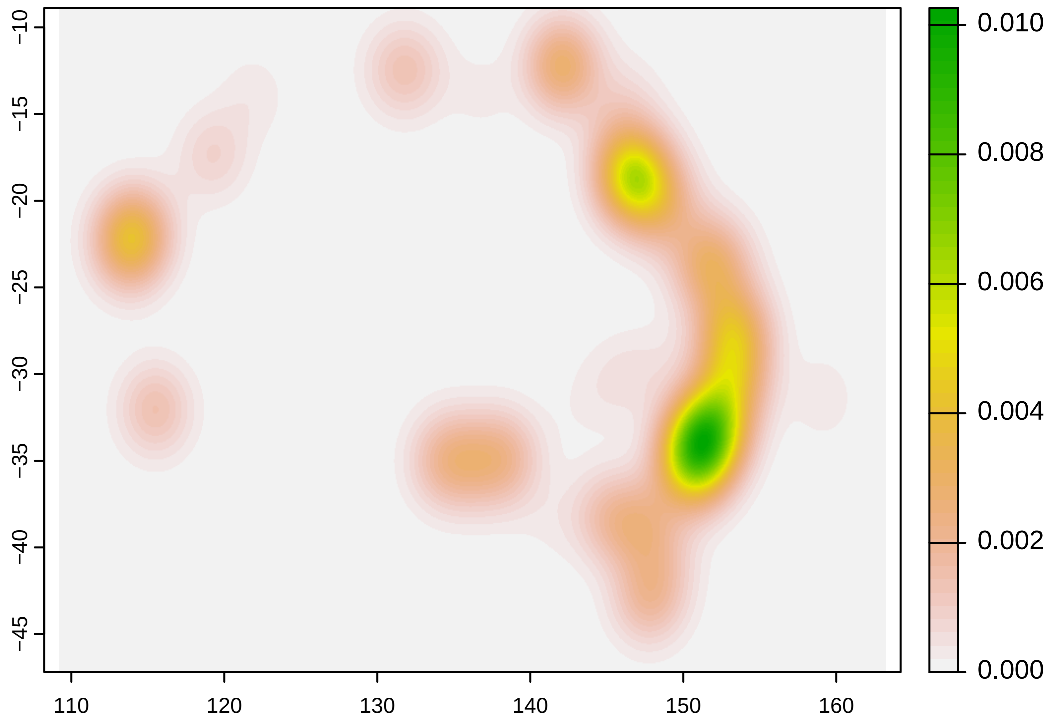
```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
## The following object is masked from 'package:terra':
##
##     area
```

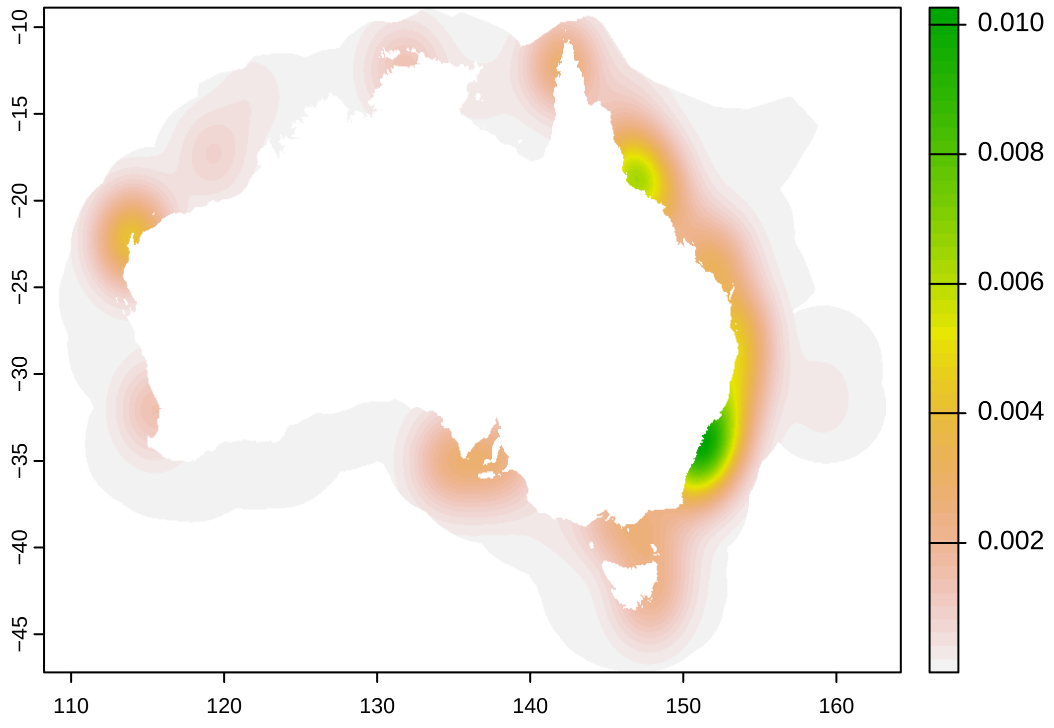
```
dens <- kde2d(st_xy2[,1], st_xy2[,2], n = c(nrow(b2), ncol(b2)))
b4<-raster::raster(dens)
plot(b4)
```



```
b4a<-terra::rast(b4)
b5 <- terra::project(b4a,bath2)
plot(b5)
```



```
b6 <- mask(b5,bath2)
plot(b6)
```



```
dir.create("bias",showWarnings = FALSE)
writeRaster(b6,paste0(getwd(),"/",("bias/Bias_AODN.tif")),overwrite=TRUE)
```

Next we are going to summarise the available raw data into monthly averages.

First, we need to create some directories. Note the formatting of the paste0 directory can be fiddly depending on your local environment.

```
dir.create("monthly_predictors",showWarnings = FALSE)

sub_dirs <- c("01","02","03","04","05","06","07","08","09","10","11","12")

for(i in 1:length(sub_dirs)){
  dir.create(paste0("~/Downloads/Marine/monthly_predictors/",sub_dirs[i]))
}
```

```
## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/01' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/02' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/03' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/04' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/05' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/06' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/07' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/08' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/09' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/10' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/11' already
## exists

## Warning in dir.create(paste0("~/Downloads/Marine/monthly_predictors/",
## sub_dirs[i])): '/Users/s2992269/Downloads/Marine/monthly_predictors/12' already
## exists
```

Now we will generate monthly data for chl or chlorophyll.

```r
setwd("~/Downloads/Marine/raw_data/chl")

t1<-list.files(recursive=TRUE)
#get only unique charater strings near end of string in new list
t2<-unique(str_sub(t1,-6,-5))

for (i in 1:length(unique(t2))){
  txt_files=list.files(pattern=paste("*\\-",unique(t2)[i],'.*\\.tif',sep=""),recursive=TRUE)
  stack<-terra::rast(txt_files)
  filename<-"chl"
  filen<-mean(stack)
  mean_chl_1 <-project(filen,bath2)
  wt2 <- matrix (data = 1, nrow = 33, ncol = 33)
  mean_chl_2 <- focal(mean_chl_1, wt2, fun = mean, na.rm = TRUE, NAonly = TRUE)
  mean_chl_3 <- raster::mask(mean_chl_2, bath2)
  writeRaster(mean_chl_3,paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/chl.tif"),overw
}
```

Next we generate monthly summaries of the gsla variable

```
setwd("~/Downloads/Marine/raw_data/gsla")

t1<-list.files(recursive=TRUE)
#get only unique charater strings near end of string in new list
t2<-unique(str_sub(t1,-6,-5))

for (i in 1:length(unique(t2))){
  txt_files=list.files(pattern=paste("*\\-",unique(t2)[i],'.*\\.tif',sep=""),recursive=TRUE)
  stack<-terra::rast(txt_files)
  filename<-"gsla"
  filen<-mean(stack)
  mean_gsla_1 <-project(filen,bath2)
  wt2 <- matrix (data = 1, nrow = 15, ncol = 15)
  mean_gsla_2 <- focal(mean_gsla_1, wt2, fun = mean, na.rm = TRUE, NAonly = TRUE)
  mean_gsla_3 <- raster::mask(mean_gsla_2, bath2)
  writeRaster(mean_gsla_3,paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/gsla.tif"),ove
}
```

Then we create monthly summaries again for sea surface temperature

```
setwd("~/Downloads/Marine/raw_data/sst")

t1<-list.files(recursive=TRUE)
#get only unique charater strings near end of string in new list
t2<-unique(str_sub(t1,-6,-5))

for (i in 1:length(unique(t2))){
  txt_files=list.files(pattern=paste("*\\-",unique(t2)[i],'.*\\.tif',sep=""),recursive=TRUE)
  stack<-rast(txt_files)
  all_sst_2 <-project(stack,bath2)
  all_sst_3 <- raster::mask(all_sst_2, bath2)
  filen<- mean(all_sst_3)
  wt2 <- matrix (data = 1, nrow = 35, ncol = 35)
  mean_sst_2 <- focal(filen, wt2, fun = mean, na.rm = TRUE, NAonly = TRUE)
  mean_sst_3 <- raster::mask(mean_sst_2, bath2)
  writeRaster(mean_sst_3,paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/sst.tif"),overw
}
```

And again for the vcur variable.

```
setwd("~/Downloads/Marine/raw_data/vcur")
```

```
t1<-list.files(recursive=TRUE)
#get only unique charater strings near end of string in new list
t2<-unique(str_sub(t1,-6,-5))

for (i in 1:length(unique(t2))){
  txt_files=list.files(pattern=paste("*\\-",unique(t2)[i],'.*\\.tif',sep=""),recursive=TRUE)
  stack<-rast(txt_files)
  filename<-"vcur"
  filen<-mean(stack)
  mean_vcur_1 <-project(filen,bath2)
  wt2 <- matrix (data = 1, nrow = 15, ncol = 15)
  mean_vcur_2 <- focal(mean_vcur_1, wt2, fun = mean, na.rm = TRUE, NAonly = TRUE)
  mean_vcur_3 <- raster::mask(mean_vcur_2, bath2)
  writeRaster(mean_vcur_3,paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/vcur.tif"),ove
}
```

Next we generate a unique raster for each month of the year with all grid-cell values corresponding to the numeric month value.

Then we create a base layer where each value is equal to one, then create a loop to generate monthly values.

```
base <- bath2/bath2

setwd("~/Downloads/Marine/raw_data/vcur")
t1<-list.files(recursive=TRUE)
t2<-unique(str_sub(t1,-6,-5))

for (i in 1:length(unique(t2))){
  month<- as.numeric(unique(t2)[i])
  mo<- base*month
  writeRaster(mo,paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/mo.tif"),overwrite=TRUE
}
```

The next steps will require some additional packages.

```
library(dismo)
library(raster)
library(sf)
library(rJava)
library(jpeg)
require(stringr)
```

Next we add a couple of columns to our occurrence dataset.

```
YT_2$month <- sprintf("%02d", as.numeric(YT_2$month))
YT_2$mo <- as.factor(as.numeric(YT_2$month))
head(YT_2)
```

```
##        vernacularName         species       lat      lon
## 1 Yellowtail Kingfish Seriola lalandi -32.71410 152.1508
## 2 Yellowtail Kingfish Seriola lalandi -30.93955 153.1014
## 3 Yellowtail Kingfish Seriola lalandi -34.06928 151.1298
## 4 Yellowtail Kingfish Seriola lalandi -33.08591 151.6397
## 5 Yellowtail Kingfish Seriola lalandi -34.06945 151.1296
## 6 Yellowtail Kingfish Seriola lalandi -34.85332 150.6299
##   coordinateUncertaintyInMeters year month     basisOfRecord mo
## 1                             2 2019    02 HUMAN_OBSERVATION  2
## 2                             4 2016    03 HUMAN_OBSERVATION  3
## 3                             4 2020    03 HUMAN_OBSERVATION  3
## 4                             4 2020    06 HUMAN_OBSERVATION  6
## 5                             4 2020    03 HUMAN_OBSERVATION  3
## 6                             5 2018    10 HUMAN_OBSERVATION 10
```

We then check if maxent.jar and associated files are available. Download the Maxent software to your working directory if needed. https://biodiversityinformatics.amnh.org/open_source/maxent/

```
jar <- paste(system.file(package = "dismo"), "/java/maxent.jar", sep = '')
if (file.exists(jar)) {
  cat("can continue, maxent is available")
} else {
  cat('cannot run this because maxent is not available')
}
```

```
## can continue, maxent is available
```

Now we are going to check for correlations in our variables using January data. First we need to create a stack of January rasters.

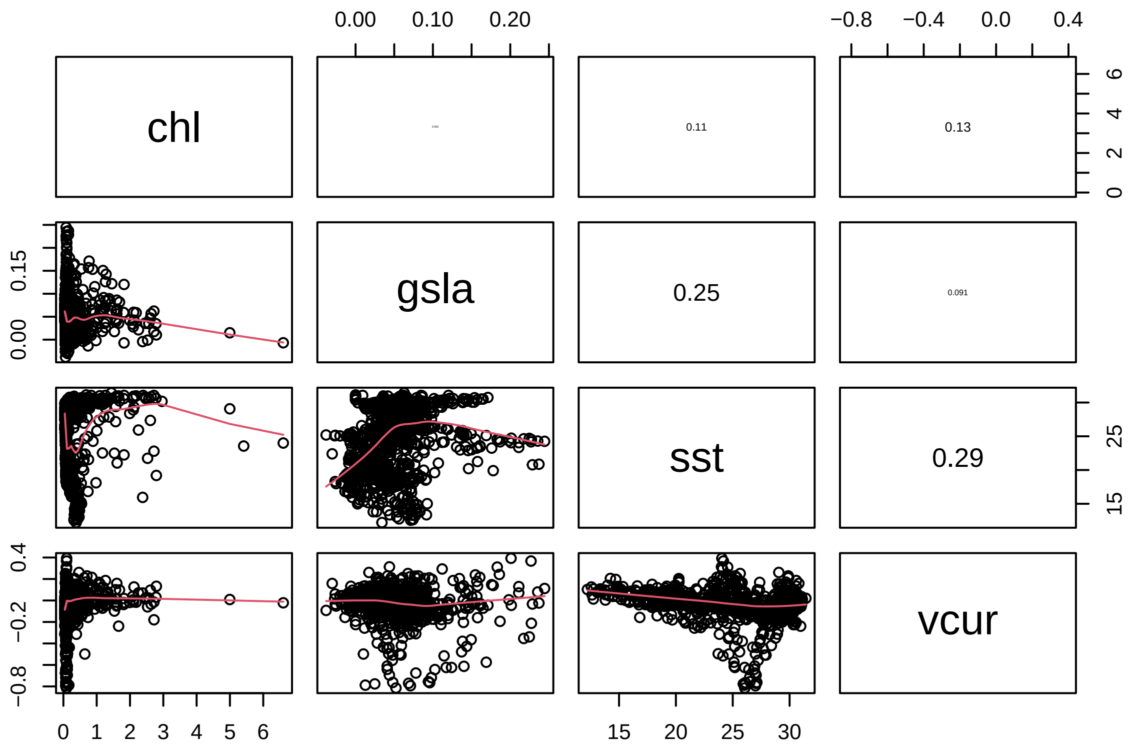
```
setwd("~/Downloads/Marine/monthly_predictors/01")
rast_lst <- list.files(pattern='.tif$', all.files=TRUE)
rast_lst<-rast_lst[-3]
YT_env_vars <- raster::stack(rast_lst)
crs(YT_env_vars)<-crs(bath)
YT_env_vars
```

```
## class      : RasterStack
## dimensions : 4599, 6476, 29783124, 4  (nrow, ncol, ncell, nlayers)
## resolution : 0.008333333, 0.008333333  (x, y)
## extent     : 109.2333, 163.2, -47.2, -8.875  (xmin, xmax, ymin, ymax)
## crs        : +proj=longlat +datum=WGS84 +no_defs
## names      :         chl,        gsla,          sst,        vcur
## min values :  0.03821435, -0.05287718, 12.07232491, -1.09386380
## max values :  37.1665578,   0.2903080,  31.7113419,   0.4307501
```

Now we will check for correlations in our environmental variables, assuming January is representative of other months.

```
panel.cor <- function(x, y, digits=2, prefix="", cex.cor, ...)
{
  usr <- par("usr"); on.exit(par(usr))
  par(usr = c(0, 1, 0, 1))
  r <- abs(cor(x, y, use="pairwise.complete.obs"))
  txt <- format(c(r, 0.123456789), digits=digits)[1]
  txt <- paste(prefix, txt, sep="")
```

```r
  if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)
  text(0.5, 0.5, txt, cex = cex.cor * r)
}
rpoints<-randomPoints(YT_env_vars,1000)
samp<-raster::extract(YT_env_vars,rpoints)
pairs(samp,lower.panel=panel.smooth,upper.panel=panel.cor)
```

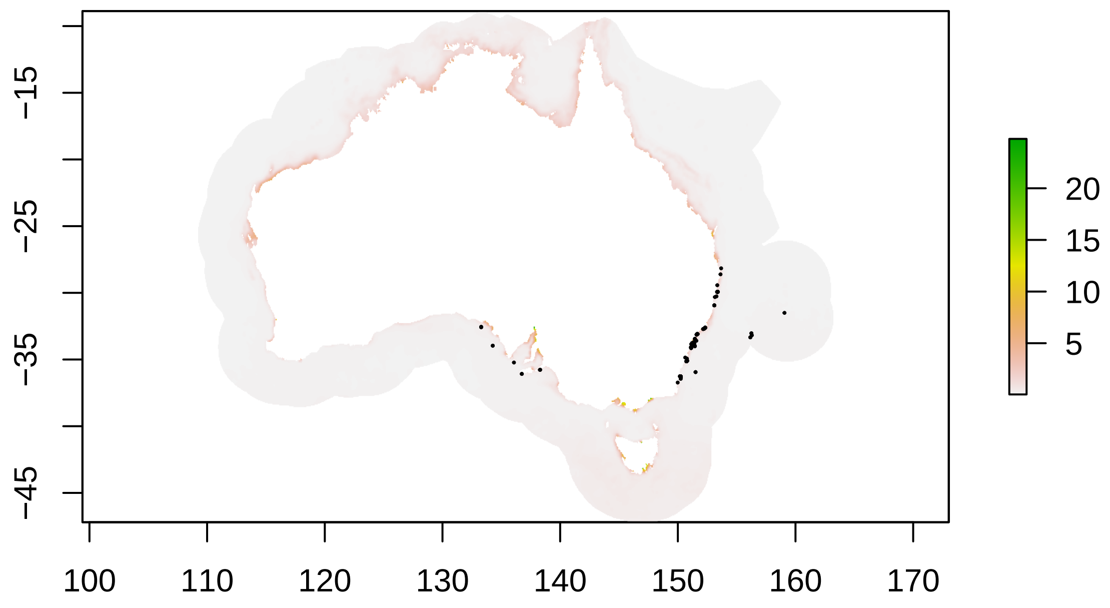


Next do some data prep

```r
x_YTKF<- YT_2$lon
y_YTKF<-YT_2$lat
xy_YTKF<-cbind(x_YTKF,y_YTKF)
xy.sp_YTKF<-SpatialPoints(xy_YTKF)
crs(xy.sp_YTKF) <- crs(YT_env_vars)
crs(xy.sp_YTKF)
```

```
## Coordinate Reference System:
## Deprecated Proj.4 representation: +proj=longlat +datum=WGS84 +no_defs
## WKT2 2019 representation:
## GEOGCRS["WGS 84 (with axis order normalized for visualization)",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["geodetic longitude (Lon)",east,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433,
##                 ID["EPSG",9122]]],
##         AXIS["geodetic latitude (Lat)",north,
##             ORDER[2],
##             ANGLEUNIT["degree",0.0174532925199433,
##                 ID["EPSG",9122]]],
##     REMARK["Axis order reversed compared to EPSG:4326"]]
```

```
plot(YT_env_vars[[1]])
points(xy.sp_YTKF,pch=20,cex=0.2)
```



```
mask_r <- YT_env_vars[[2]]
ext <- raster::extent(YT_env_vars)
```

Make some more directories

```
dir.create("data",showWarnings = FALSE)
dir.create("~/Downloads/Marine/data/env_dframe/",showWarnings = FALSE)
```

Next extract the data for each month in our dataframe from each month's environmental variables.

```r
t1<-unique(YT_2$month)

t1<-sort(t1)
bias <- raster("~/Downloads/Marine/bias/Bias_AODN.tif")

for (i in 1:length(unique(t1))){
  subsetYT<-YT_2[which(YT_2$month==unique(t1)[i]), ]
  longitude<-subsetYT$lon
  latitude<-subsetYT$lat
  xy<-cbind(longitude,latitude)
  xy.sp<-SpatialPoints(xy)
  raster::crs(xy.sp) <- raster::crs(YT_env_vars)

  chl<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/chl.tif"))

  gsla<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/gsla.tif"))

  mo<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/mo.tif"))

  sst<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/sst.tif"))

  vcur<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/vcur.tif"))

  EVstack<-raster::stack(chl,gsla,mo,sst,vcur)
  raster::crs(EVstack) <- raster::crs(YT_env_vars)

  presEV <- as.data.frame(raster::extract(EVstack,xy.sp))
  pres_p <- cbind(longitude,latitude,presEV)
  pres_p$pres <- 1


  rpoints<-as.data.frame(raster::xyFromCell(bias, sample(which(!is.na(values(bias))), 2000, prob=values
  colnames(rpoints)<-c("longitude","latitude")
  xy.bk<-SpatialPoints(rpoints)
  raster::crs(xy.bk) <- raster::crs(EVstack)
  bkgEV <- as.data.frame(raster::extract(EVstack,xy.bk))
  bkg_p <- cbind(rpoints,bkgEV)
  bkg_p$pres <- 0

  dat1 <- rbind(pres_p,bkg_p)
  write.csv(dat1,paste0("~/Downloads/Marine/data/env_dframe/env_",unique(t2)[i],".csv"))
}
```

Next we combine all the files from the monthly data extractions

```r
files <- list.files("~/Downloads/Marine/data/env_dframe/")
setwd("~/Downloads/Marine/data/env_dframe/")

data1 <- do.call("rbind",lapply(files,read.csv,header=TRUE))
data1$mo <- as.factor(data1$mo)
pa<- data1$pres

data1$pres<-NULL
```

```
data1$X<-NULL
data1$longitude<-NULL
data1$latitude<-NULL

write.csv(data1,"~/Downloads/Marine/data/maxent_env_dat.csv")
write.csv(pa,"~/Downloads/Marine/data/pa_vector.csv")
```

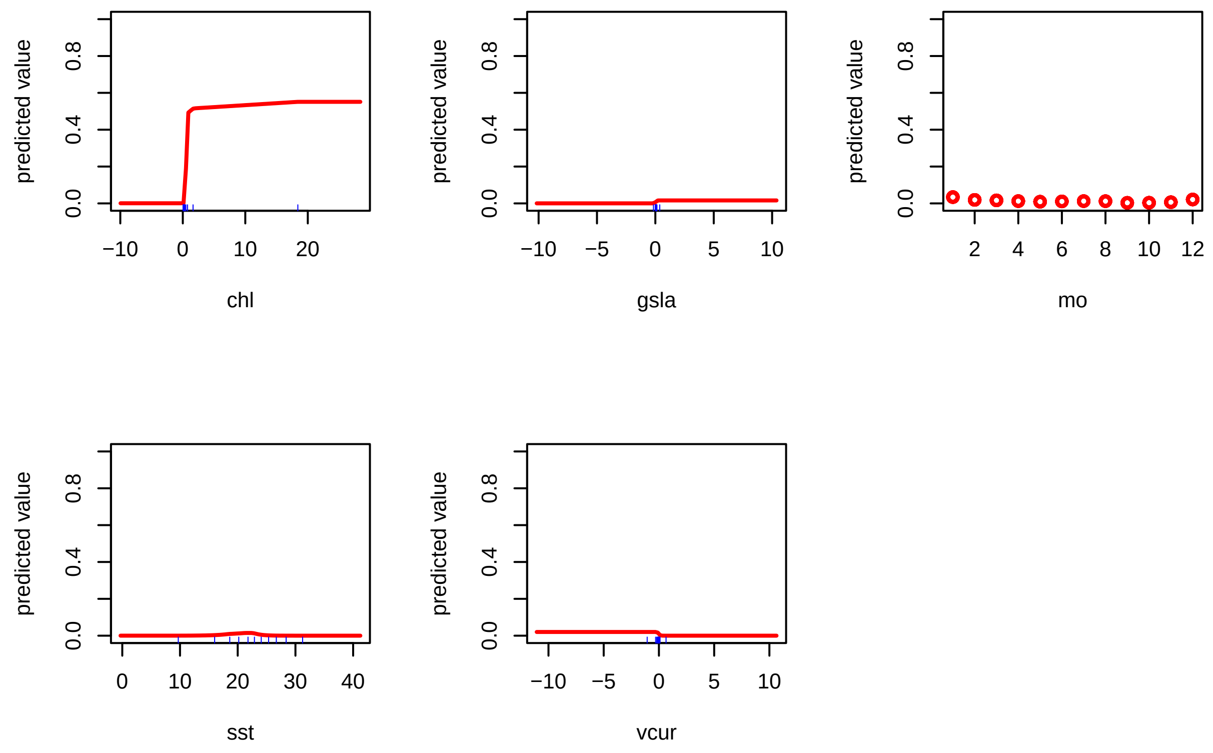
Now we create one maxent model from all the months of data combined.

```
maxent_args <- c('removeduplicates=TRUE','jackknife=TRUE','responsecurves=TRUE','plots=TRUE','betamulti
Maxent_model1 <- dismo::maxent(x=data1,p=pa,path = "results_April22",args= maxent_args)
```

```
## This is MaxEnt version 3.4.3
```

Plot response curves for the Maxent model, see the results_April22 folder for more results

```
response(Maxent_model1)
```



Now lets generate some monthly predictions based on this global model

```
for (i in 1:length(unique(t1))){
  subsetYT<-YT_2[which(YT_2$month==unique(t1)[i]), ]
  longitude<-subsetYT$lon
  latitude<-subsetYT$lat
  xy<-cbind(longitude,latitude)
  xy.sp<-SpatialPoints(xy)
  raster::crs(xy.sp) <- raster::crs(YT_env_vars)

  chl<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/chl.tif"))

  gsla<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/gsla.tif"))

  mo<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/mo.tif"))

  sst<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/sst.tif"))

  vcur<-raster(paste0("~/Downloads/Marine/monthly_predictors/",unique(t2)[i],"/vcur.tif"))

  Vstack<-stack(chl,gsla,mo,sst,vcur)
  raster::crs(EVstack) <- raster::crs(YT_env_vars)

  filenameM <- paste0("/results_April22/map_pred_",unique(t1)[i],".tif")
  map_predictions <- predict(Maxent_model1, Vstack,filename=paste0(getwd(),filenameM), args= c('outputf

  jpeg(paste0(getwd(),"/results_April22/YTKF_predicted",unique(t1)[i],".jpeg"))
  plot(map_predictions, main=paste0("Yellowtail Kingfish distribution month =",unique(t1)[i]))
  points(xy.sp,pch=20,cex=0.2)
  dev.off()

  results<-read.csv(paste0(getwd(),"/results_April22/maxentResults.csv"))

  thresh<-results$Maximum.training.sensitivity.plus.specificity.area
  m <- c(0, thresh, 0,  thresh, 1, 1)
  reclass <- matrix(m, ncol= 3, byrow= TRUE)
  rc <- reclassify(map_predictions, reclass)

  jpeg(paste0(getwd(),"/results_April22/YTKF_thresholded",unique(t1)[i],".jpeg"))
  plot(rc, main=paste0("Yellowtail Kingfish distribution month =",unique(t1)[i]))
  points(xy.sp,pch=20,cex=0.2)
  dev.off()
}
```