Fishery definition for YFT in dolphin sets

Haikun Xu

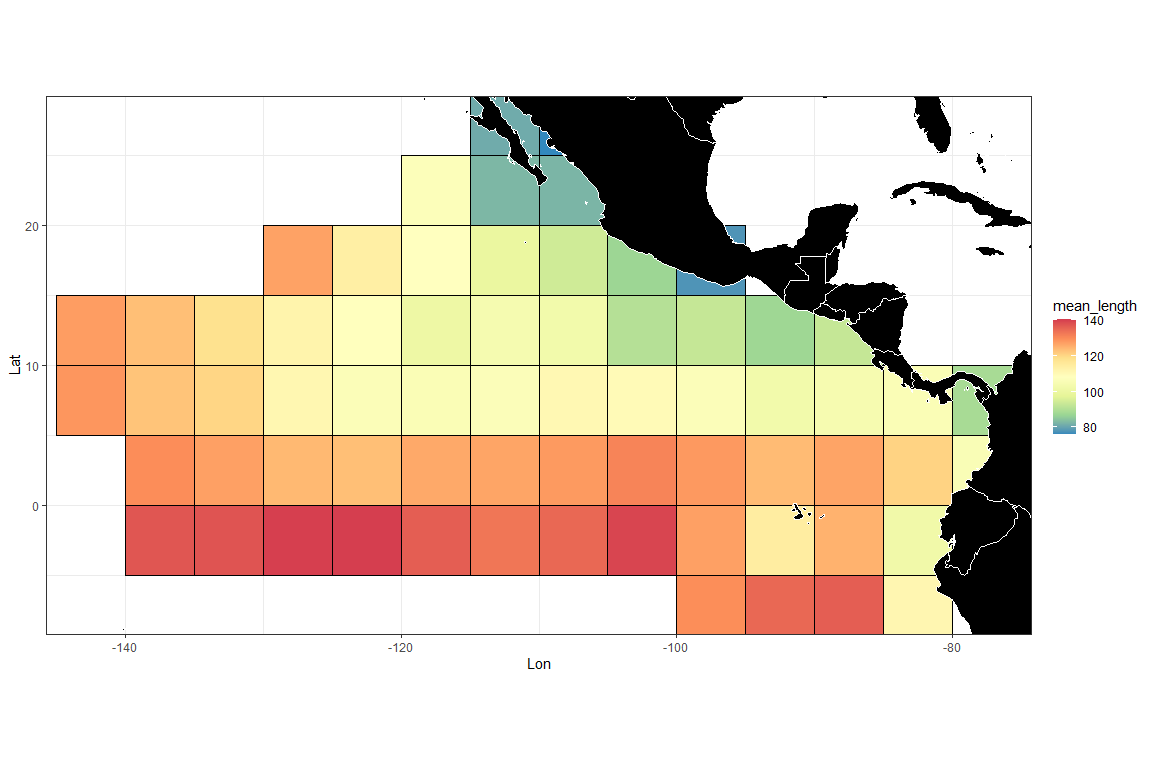
2024-08-15

library(FishFreqClustering)  
library(FishFreqTree)  
library(tidyverse)

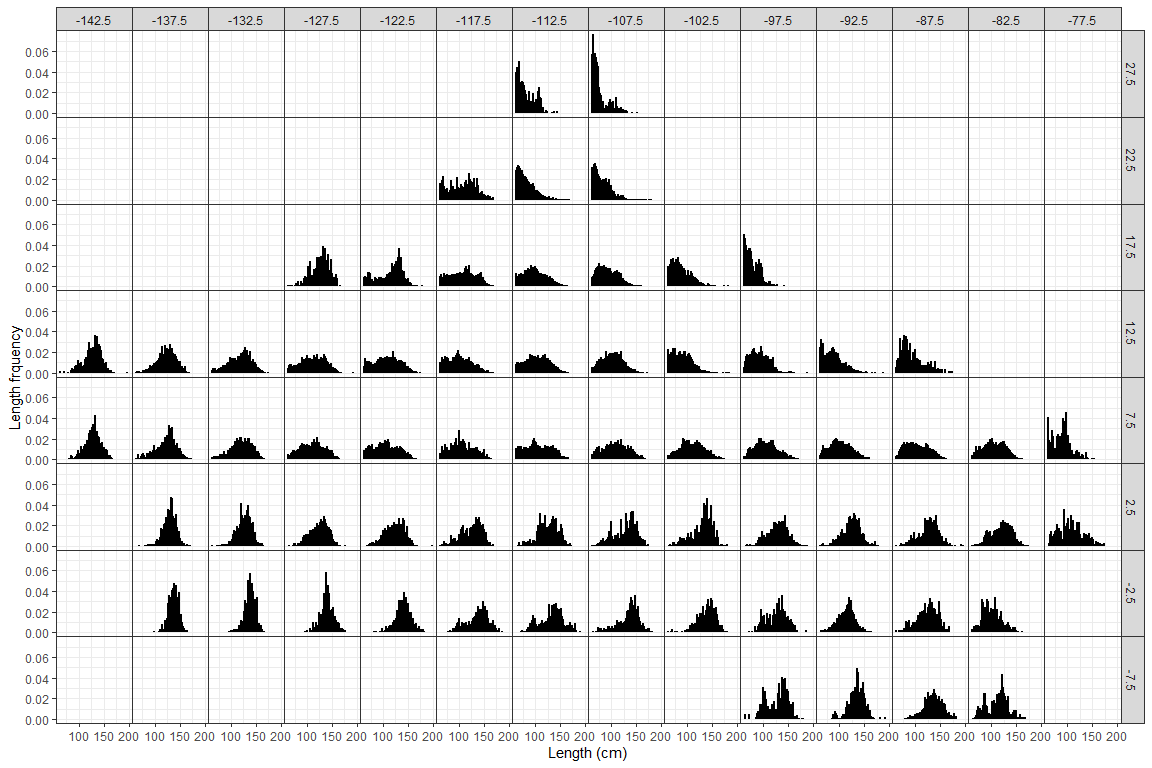
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.2 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.4.2 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.1   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

directory <- "D:/OneDrive - IATTC/IATTC/2024/Irregular clustering/YFT DEL/"  
setwd(directory)  
  
Raw <- read.csv("yft\_lf\_2000\_to\_2022.csv")  
Raw$quarter = ceiling(Raw$month / 3)  
Raw$lat = Raw$lat.5deg + 2.5  
Raw$lon = Raw$lon.5deg - 2.5

LF.DEL <- Raw %>% filter(class == 6, setype == 1) # 1=DEL; 4=NOA; 5=OBJ  
  
LF <- LF.DEL[, c("year", "quarter", "lat", "lon", paste0("X", 1:201))] %>%  
 group\_by(lat, lon) %>%  
 mutate(N = length(unique(paste0(year, "-", quarter)))) %>%  
 filter(N > 3, lat > -10) # remove the cells with less than 4 quarters of data since 2000  
  
bins <- seq(1, 201, 1) # data length bins  
new\_bins <- seq(61, 200, 1) # bins to be used in the clustering analysis  
  
# first aggregate the raw LF to the new bins by quarter  
LF1 <- lf.aggregate(LF, fcol = 5, lcol = 205, bins, new\_bins, LengthOnly = FALSE)  
  
# Chekcing the data by making two plots  
bins <- new\_bins # use the new bins  
nbins <- length(bins)  
fcol = 5  
lcol = 4 + length(bins)  
save\_dir=directory  
  
make.meanl.map(LF1, fcol, lcol, bins, save\_dir, width = 10, height = 10)



make.lf.map(LF1, fcol, lcol, bins, save\_dir)

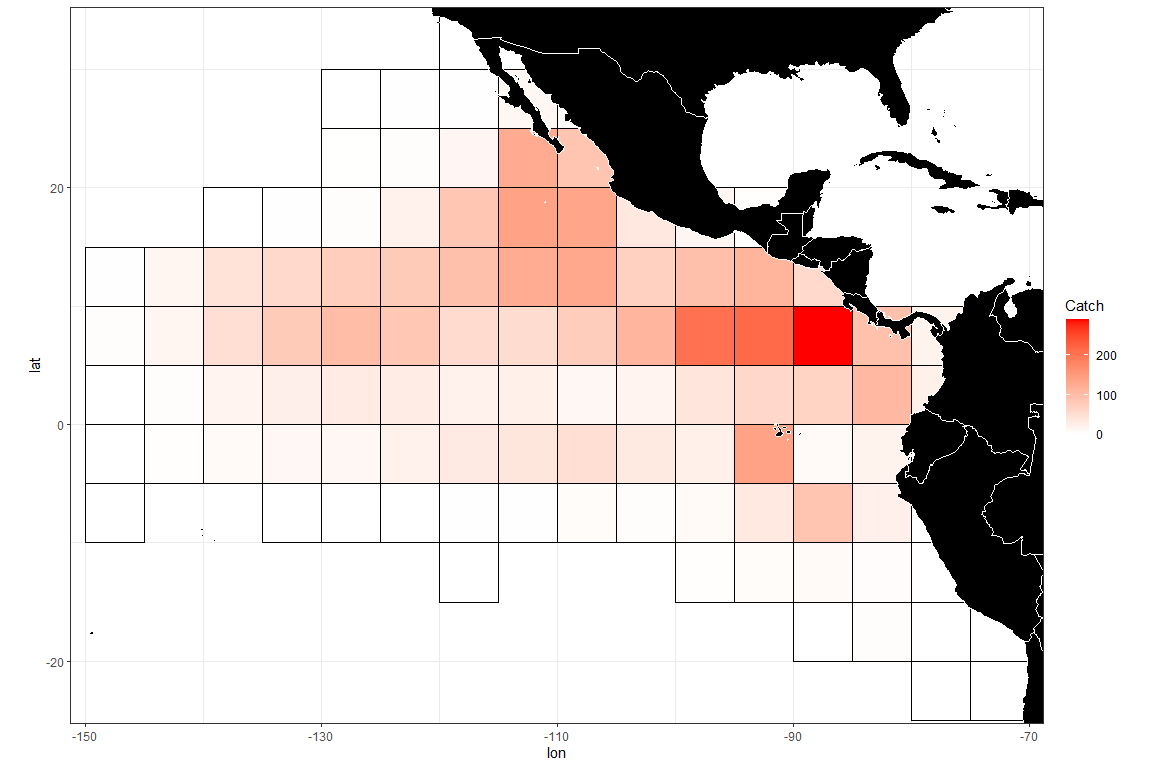


# divide the LF by the mean LF for the year-quarter  
LF2 <- lf.demean(LF1, fcol, lcol, bins)

cae <- BSE::read.cae.f("D:/OneDrive - IATTC/IATTC/2024/SAC15/PS Database/", "CAE-LatLon2000-2023.txt",2000,2023)  
  
YFT\_DEL\_catch <- cae %>%  
 filter(setype == 1, class == 6) %>%  
 rename(lat = latc5, lon = lonc5) %>%  
 group\_by(lat, lon) %>%  
 summarise(Catch = sum(sum.trop.tunas)/1000)

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

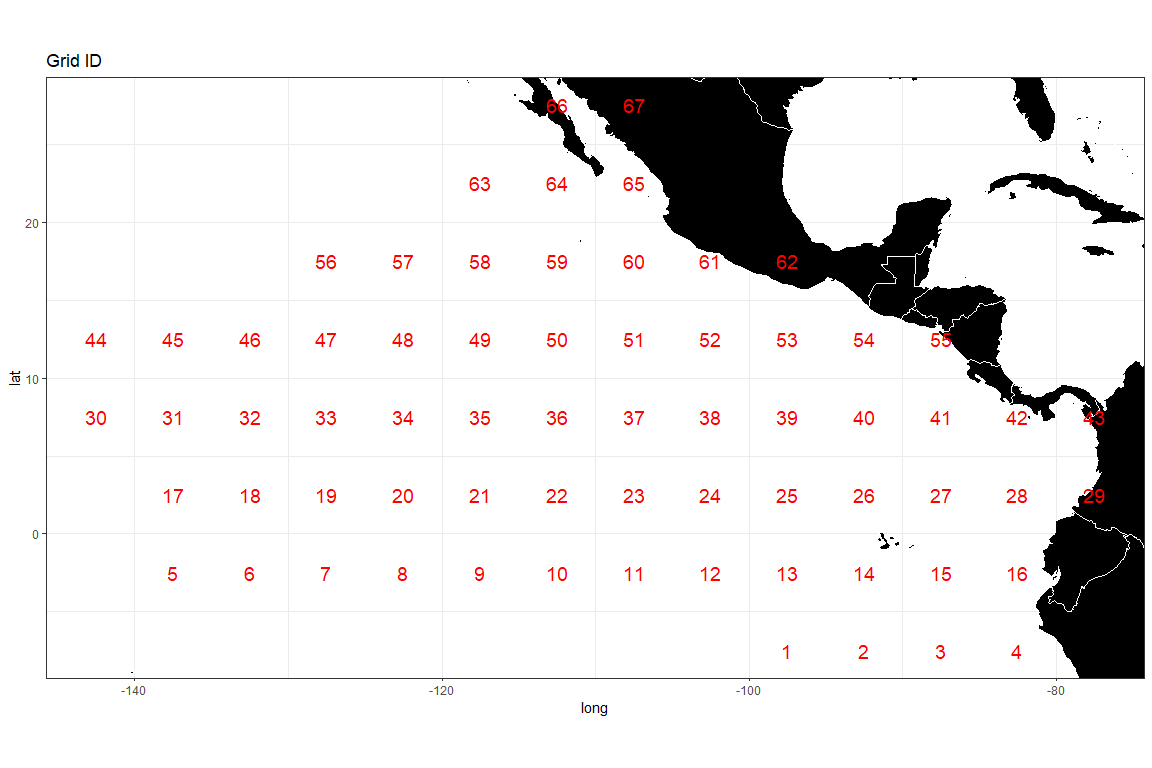
wmap <- map\_data("world")  
ggplot(data = YFT\_DEL\_catch) +  
 geom\_tile(aes(x = lon, y = lat, fill = Catch), color = "black") +  
 scale\_fill\_gradient(  
 low = "white",  
 high = "red",  
 na.value = "white",  
 limits = c(0, max(YFT\_DEL\_catch$Catch))  
 ) +  
 geom\_polygon(  
 data = wmap,  
 aes(long, lat, group = group),  
 fill = "black",  
 colour = "white",  
 lwd = 0.5  
 ) +  
 coord\_quickmap(ylim = c(min(YFT\_DEL\_catch$lat), max(YFT\_DEL\_catch$lat)), xlim = c(min(YFT\_DEL\_catch$lon), max(YFT\_DEL\_catch$lon))) +  
 theme\_bw()



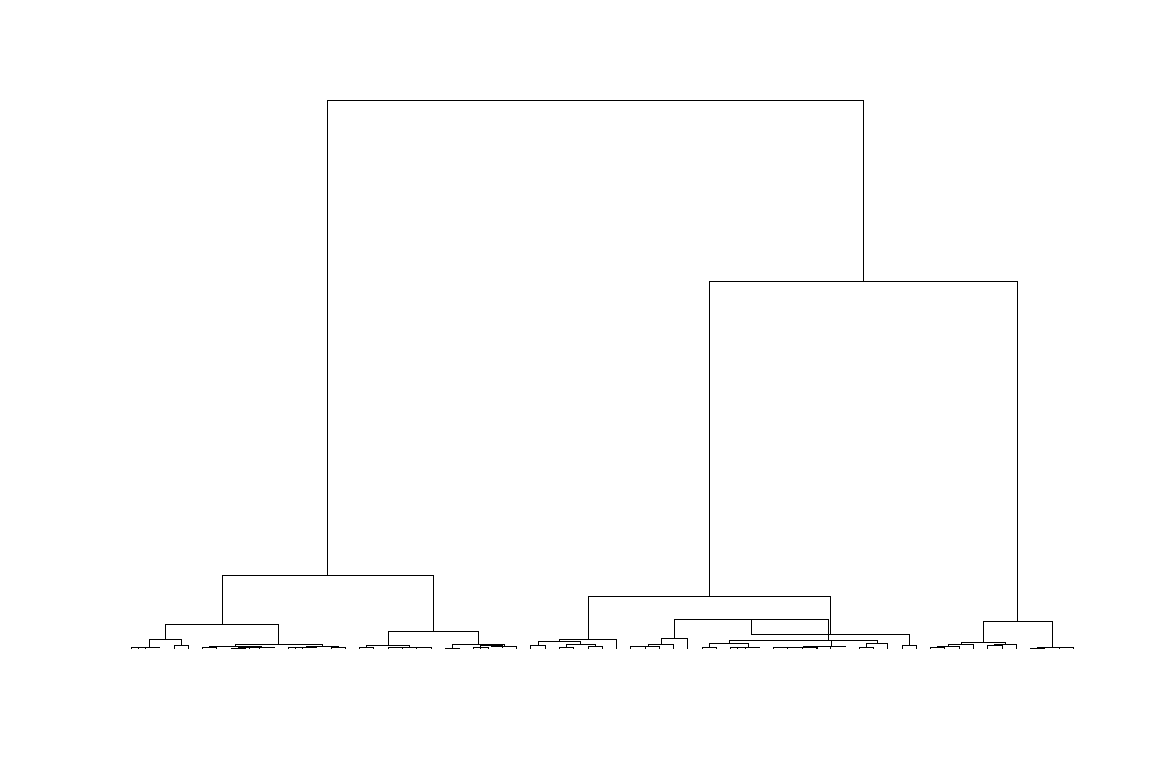
mmd <- LF2[,c(2,4:(lcol+1))] # mmd is the input data for the clustering analysis - it should have year, lat, lon, and bin numbers  
  
# setting up input data frames for clustering algorithm  
temp = packbylatlon(mmd, 5, 5, nbins) # aggregate the input LF across time for each grid cell  
  
packedmmd3 = temp$table1  
packedpdf3 = topdf(packedmmd3, 4, 3 + nbins)  
# packedcdf3 = tocdf(packedpdf3, 4, 3 + nbins)  
mmdt = packedmmd3[packedmmd3[, 4 + nbins] > 0,]  
rrs = mmdt[, 4 + nbins] # sample size  
mmdtpdf = packedpdf3[packedmmd3[, 4 + nbins] > 0,] # PDF sums to 1 for each grid  
mmdtpdf[, 4 + nbins] = mmdt[, 4 + nbins]  
# mmdtcdf = packedcdf3[packedmmd3[, 4 + nbins] > 0,]  
# mmdtcdf[, 4 + nbins] = mmdt[, 4 + nbins]  
  
names(mmdtpdf)[2:3] <- c("lat", "lon")  
mmdtpdf$Number <- 1:nrow(mmdtpdf)  
mmdfpdf\_catch <- left\_join(mmdtpdf, YFT\_DEL\_catch)

## Joining with `by = join\_by(lat, lon)`

# check grid ID  
ggplot(data = mmdfpdf\_catch) +  
 geom\_polygon(  
 data = wmap,  
 aes(long, lat, group = group),  
 fill = "black",  
 colour = "white",  
 lwd = 0.5  
 ) +  
 geom\_text(aes(x = lon, y = lat, label = Number), color = "red", size = 5) +  
 coord\_quickmap(ylim = c(min(mmdfpdf\_catch$lat), max(mmdfpdf\_catch$lat)), xlim = c(min(mmdfpdf\_catch$lon), max(mmdfpdf\_catch$lon))) +  
 ggtitle("Grid ID") +  
 theme\_bw()



densmatx = matrix(0, nrow(mmdt), 512)  
densmaty = matrix(0, nrow(mmdt), 512)  
for(i in 1:nrow(mmdt)) {  
 weightvec = t(mmdt[i, 4:(3 + nbins)])  
 weightvec = weightvec / sum(weightvec)  
 tempmmd = density(seq(0.61, 2.00, 0.01), weights = weightvec, bw = 0.10)  
 densmatx[i, ] = tempmmd$x  
 densmaty[i, ] = tempmmd$y  
}  
  
# run distributional clustering with adjacency criterion  
adjmat <- adjinf(mmdtpdf[, 2], mmdtpdf[, 3], mindist = 5 \* sqrt(2)) # the matrix specifying adjacency   
adjmat[16, 29] <- 0  
adjmat[29, 16] <- 0  
  
alydens.spatial23 <-  
 hclust.regionsmm(  
 as.matrix(densmaty),  
 adj = TRUE, # adjacent areas  
 adjmat = adjmat,  
 rr = mmdfpdf\_catch$Catch # rr is the weighting factor; equal weighting is used in this case  
 )  
  
# Look at the tree structure  
cplotu(alydens.spatial23$merges, alydens.spatial23$distseq, hopt = 'dist')



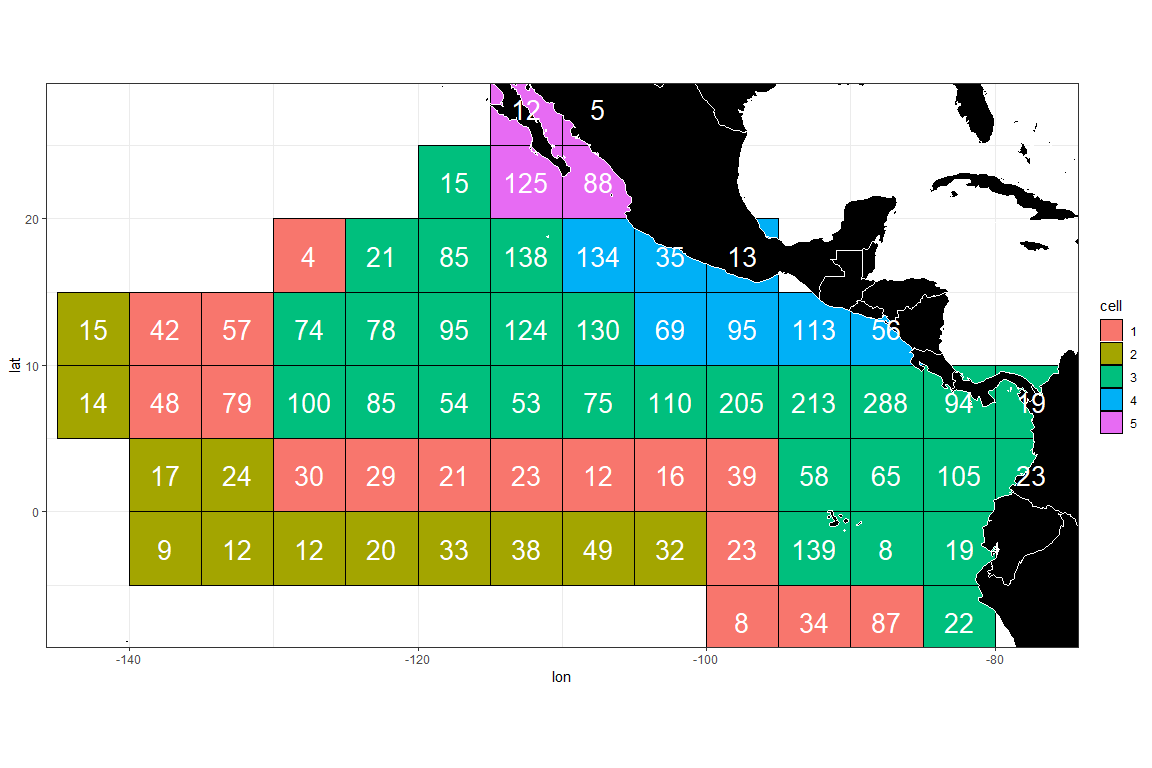
## NULL

teststat <- heterodist(  
 alydens.spatial23$merges,  
 alydens.spatial23$distseq,  
 tempmmd$x,  
 densmaty,  
 mmdfpdf\_catch$Catch,  
 doko = c(1, 50),  
 BB = 100  
)

## [1] "node = 66"  
## [1] "node = 65"  
## [1] "node = 64"  
## [1] "node = 63"  
## [1] "node = 62"  
## [1] "node = 61"  
## [1] "node = 60"  
## [1] "node = 59"  
## [1] "node = 58"  
## [1] "node = 57"  
## [1] "node = 56"  
## [1] "node = 55"  
## [1] "node = 54"  
## [1] "node = 53"  
## [1] "node = 52"  
## [1] "node = 51"  
## [1] "node = 50"  
## [1] "node = 49"  
## [1] "node = 48"  
## [1] "node = 47"  
## [1] "node = 46"  
## [1] "node = 45"  
## [1] "node = 44"  
## [1] "node = 43"  
## [1] "node = 42"  
## [1] "node = 41"  
## [1] "node = 40"  
## [1] "node = 39"  
## [1] "node = 38"  
## [1] "node = 37"  
## [1] "node = 36"  
## [1] "node = 35"  
## [1] "node = 34"  
## [1] "node = 33"  
## [1] "node = 32"  
## [1] "node = 31"  
## [1] "node = 30"  
## [1] "node = 29"  
## [1] "node = 28"  
## [1] "node = 27"  
## [1] "node = 26"  
## [1] "node = 25"  
## [1] "node = 24"  
## [1] "node = 23"  
## [1] "node = 22"  
## [1] "node = 21"  
## [1] "node = 20"  
## [1] "node = 19"  
## [1] "node = 18"  
## [1] "node = 17"

catch\_prop <- rep(0, nrow(teststat$statmat))  
for (i in 1:nrow(teststat$statmat)) {  
 Cells <- teststat$childnodes[i][[1]]  
 catch\_prop[i] <- sum(mmdfpdf\_catch$Catch[Cells]) / sum(mmdfpdf\_catch$Catch)  
}  
  
MJS\_statistics <- data.frame(cbind(1:nrow(teststat$statmat), teststat$statmat[,4], round(catch\_prop,2)))  
names(MJS\_statistics) <- c("Node\_Number", "STD\_distance", "Catch\_Proportion")  
write.csv(MJS\_statistics, file = "MJS distance (Fishery Structure).csv", row.names = FALSE)

# draw density curves by cluster  
final\_nodes <- find\_clusters(MJS\_statistics, distance\_threshold = 80, catch\_threshold = 0.05)  
  
temp2 <- rep(1, nrow(mmdt))  
for (i in 2:length(final\_nodes)) {  
 temp2[teststat$childnodes[final\_nodes[i]][[1]]] <- i  
}  
  
cluster <- cbind(mmdt[,2:3], factor(temp2), round(mmdfpdf\_catch$Catch,0))  
names(cluster) <- c("lat", "lon", "cell", "Weight")  
# write.csv(cluster, file = paste0(save\_dir, "cluster\_YFT", kk, ".csv"), row.names = FALSE)  
  
# # map of clusters  
ggplot(data = cluster) +  
 geom\_tile(aes(x = lon, y = lat, fill = cell), color = "black") +  
 geom\_polygon(  
 data = wmap,  
 aes(long, lat, group = group),  
 fill = "black",  
 colour = "white",  
 lwd = 0.5  
 ) +  
 geom\_text(aes(x = lon, y = lat, label = Weight),  
 color = "white",  
 size = 7,  
 ) +  
 coord\_quickmap(ylim = c(min(cluster$lat), max(cluster$lat)), xlim = c(min(cluster$lon), max(cluster$lon))) +  
 theme\_bw()



ggsave(file = paste0(save\_dir, "Clustering\_map.png"), h = 5, w= 8)  
  
colcol = rep(c(2, 3, 4, 5, 6, 7, 8), 3)  
densy\_df <- clusthistd3(kk = 5, colseq = temp2 + 1, colcol, plot = FALSE)  
  
ggplot(data = densy\_df) +  
 geom\_line(aes(x = Length, y = Density, color = Cell), linewidth = 2) +  
 theme\_bw()

