Post Activity Code Clean Up

Marina Chaji

February 16, 2022, 15:06

Project setup

Load packages. Install if necessary.

```
# Set Path
here::i_am("data_extraction_and_processing_code.Rmd")
```

```
## here() starts at /net/home2/mlee/Effort-Displacement---Scallop
```

```
PKG <- c("here","leaflet", "tidyverse", "sf", "RODBC", "RODM", "dbplyr","raster","rgdal","readxl", "data
for (p in PKG) {
  if(!require(p,character.only = TRUE)) {
    install.packages(p)
    require(p,character.only = TRUE)}
}
```

```
## Loading required package: here
```

```
## Loading required package: leaflet
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
## Loading required package: sf
```

```
## Linking to GEOS 3.7.2, GDAL 3.1.3, PROJ 6.3.2
```

```
## Loading required package: RODBC
```

```
## Loading required package: RODM
```

```
## Loading required package: dbplyr
```

```
##
## Attaching package: 'dbplyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     ident, sql
```

```
## Loading required package: raster

## Loading required package: sp

##
## Attaching package: 'raster'

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

## Loading required package: rgdal

## Please note that rgdal will be retired by the end of 2023,
## plan transition to sf/stars/terra functions using GDAL and PROJ
## at your earliest convenience.
##
## rgdal: version: 1.5-27, (SVN revision 1148)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 3.1.3, released 2020/09/01
## Path to GDAL shared files: /usr/local/share/gdal
## GDAL binary built with GEOS: TRUE
## Loaded PROJ runtime: Rel. 6.3.2, May 1st, 2020, [PJ_VERSION: 632]
## Path to PROJ shared files: /usr/share/proj
## Linking to sp version:1.4-6
## To mute warnings of possible GDAL/OSR exportToProj4() degradation,
## use options("rgdal_show_exportToProj4_warnings"="none") before loading sp or rgdal.

## Loading required package: readxl

## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following object is masked from 'package:raster':
##
##     shift

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose

## Loading required package: tmaptools

## Loading required package: tmap
vintage_string<-Sys.Date()
vintage_string<-gsub("-","_",vintage_string)
```

Organization

We will:

1. Try to avoid copying data; when we rely on data from other people, we will read it directly into memory from the network location or Oracle.

2. Sometimes this is unnecessary, so we will copy external data into the "data/external" folder. We will have a separate subfolder for shapefiles.
3. Store an intermediate data product in "data/intermediate".
4. Store final data products in "data/main."
5. Use a vintage "suffix" to denote when we have extracted data.

Read in oracle passwords and set network directory

This is a block of code where we set up the oracle passwords and make R aware of folders on the network.

```
source(here("credentials.R"))

# Set the network_location_desktop and network_location_remote variables somewhere OUTSIDE of this code

#Comment one of these out, depending on whether you are running this code on a server or locally (with
net<-network_location_desktop
net<-network_location_remote

# These are not part of the project path
offshoreWind_directory<-file.path(net,"home5", "dcorvi","OffshoreWind","offshoreWind4","data")
spacepanels_directory<-file.path(net,"home2", "mlee","dropoff","wind")
cost_directory<-file.path(net,"work5","socialsci","Trip_Costs","2007-2020")

# Set up paths.
East_Cst_crop_2020_path<- here("data","external","shapefiles","East_Cst_crop_2020_extended")
TMSQ_path<-here("data","external","shapefiles","Ten Minute Squares Cut North and Greater Atlantic")
All_Lease_Areas_Shapefile_path<-here("data","external","shapefiles","All_Lease_Areas_Shapefile")
```

Introduction

The main idea of the model is that the fishermen/decision-makers choose from a number of alternatives, where the choice occasion is a fishing trip and selects the one that yields the highest expected utility level on any given choice occasion. By observing and modeling how decision-makers change their preferred site option in response to the changes in the levels of the site attributes, it is possible to determine how decision-makers tradeoff between the different fishing ground characteristics.

Long Term Objectives

The project objective is to develop a site-choice model primarily, improve, maintain, and disseminate a standardized fisheries dependent data set and analytical summaries that provide a more precise, accurate, comprehensive, and timely evaluation of area-specific socioeconomic impacts associated with ecosystem fishery management initiatives, offshore energy development, and offshore aquaculture development. The site-choice model and underlying data set will help support fishery and ecosystem management decisions to achieve optimum yield in each fishery and the nation's most significant benefit.

Understanding the effects of wind energy areas that are early in the process may be more impactful from a policy perspective. So, not necessarily the current wind areas, but the next block that may be coming over the next 10-30 years. Also, cumulative effects may be important.

Empirical Setting

Scallop Fishery

We are modeling the location choices of fishing vessels in the Limited Access Days-at-Sea scallop fishery. There are approximately 300-330 of these fishing vessels. They are allocated "Open Area Days-at-Sea" and a quantity of trips and/or pounds into the "Access Areas." They catch approximately 95% of the scallops. The Limited Access DAS fleet can be further subdivided into Full-Time, Part-Time, and Occasional Fleets. Vessels primarily use the New Bedford scallop dredge, but a few use a smaller dredge or a bottom trawl. Over the 13 years in our dataset, there are approximately 40,000 trips taken by this fleet, split roughly evenly into "Open areas" and "Access Areas."

For Fishing Year 2016 and earlier, the fishing year Ran from March 1 to Feb 28/29. For fishing year 2017, the year ran from March 1 to March 31. For 2018 and later, the fishing year runs from April 1 to March 31.

Wind Energy

Here is a short description of the wind energy areas and how they will close (or not close) area to fishing. 18 wind areas currently under dev. But many more are likely.

How close will fishing be able to occur within Wind Lease Areas / Turbines?

The wind energy areas do not match the ten minute squares; we are currently planning on simulating the effects of closing a wind energy area by closing an entire ten minute square that is inside or touching a WEA.

The buried cable route from a WEA to shore is likely to be closed as well. Cable buried at shallow depths and marked with concrete.

Purpose

This code extracts and processes data. Our goal is to construct a dataset that can be used to estimate a location choice model at the trip level for the Limited Access Scallop Fishery, using data from 2007-2019 (calendar years). The main datasource is a frozen DMIS table.

Dependencies

This code depends on:

1. Network access to get the APSD_DMIS_2.rda and trip cost data from places on the NEFSC network.
2. The ability to connect to NEFSC oracle databases (VTR and the Live DMIS tables at APSD.t_ssb_trip_current@garfo_nefsc)

Data Overview

There are four main data sources (so far). None are perfect.

Source	Name
DMIS_APSD_2	DMIS is a Northeast Regional Office data record matching system. Primary data sources include Allocation Management System (AMS) Database, Vessel Trip Reports (VTRs), Dealer Reports, Vessel Monitoring System (VMS) Catch Reports, Observer Reports, Vessel Permit Database, and the MQRS database, which tracks limited access fishing eligibilities
VTR (Vessel Trip Reports)	A vessel trip report (VTR) must be received by NMFS or postmarked within 15 days after the reporting month's end. For vessels that also hold a NE multispecies permit, VTRs must be submitted weekly by Tuesday of the week after the fishing trip ends. Copies of VTRs must be retained on board the vessel for 1 year after the last entry on the log and otherwise retained for 3 years after the date of the last entry on the log. If no fishing activity occurred during a reporting period (week or month), then a VTR must be submitted stating that no fishing trips were taken.
Vessel Monitoring System (VMS)	All vessels issued a Federal scallop permit are required to have an active VMS unit and must use their VMS unit to declare all vessel activity, including fishing trips and transiting.
Cost data	Werner et al predict estimate a model of trip costs. Predictions (in and out of sample) are used

We have decided to use the DMIS as our primary dataset. DMIS primarily uses Vessel Trip Reports (VTRs) for "trip" and "effort" data and dealer databases for landings. A drawback of using these data are that there is a single point (latitude and longitude) for each time a vessel deploys a particular type of gear into a statistical area. In the LADAS scallop fleet, vessels rarely, if ever, will switch gears at sea. So, a trip is most likely to have multiple VTRs if it switches statistical areas.

There aren't any big incentives (yet) to misreport statistical areas in Scallop. Unlike groundfish, scallop open areas are all managed with one control (Days-at-Sea). And fishing in the Access Areas, but reporting open areas could occur. But vessels need to declare in, so this is a very risky proposition if you are caught fishing in an Access Area, but declared into an open area.

By choosing to represent the trip as a single point, or as inside a homogeneous ten minute square, we may not have the ability to answer our research question. Alternatively, do we have the ability to model at the sub-trip level?

Other possibilities were considered for our primary dataset:

1. Observer cover a subset of the fishery. According to the 2021 SBRM report, it was approximately 8-10% of effort for the Limited access fleet. This would provide haul level lat-lon and estimates of catch for the sampled subset. We viewed the subset as too limited - it would provide us with observations of approximately 200 Access area and 100 open area trips per year.Observer data contains the sailing and landing port.
2. VMS - VMS data would provide lat-lon at a high frequency. Other researchers have used this; however we uncomfortable with figuring out how to allocate catch along the VMS track. VMS data contains the sailing and landing port.
3. Rasters. The raster data are an intermediate data product that combines trip report with a statistical model describes the distance between observed hauls and the vtr point location. This allows for a smoothing of effort catch across a non-arbitrary grid (like a 10 minute square, statistical area, or just a lat-lon point).

DMIS

We are using the DMIS_APSD_2 table.

Column	Description
DOCID	VTR DOCUMENT table record identifier; Primary key, internally generated at scanning based on vessel id and date/time sailed. Each DOCID represents one trip; equivalent to TRIPID in VESLOG tables.
Dates	VTR land date, AMS Land Date, Dealer Sold Date Trip date is broken down into fields Calendar_Year, Month_of_Year, Week_of_Year, and Day_of_Year
DDLAT	Latitude in decimal degrees
DDLON	Longitude in decimal degrees
PERMIT	Six-digit vessel fishing permit number assigned by the NE Regional Office permit system
DOLLAR	This is the value of fish sold.An imputed price is used in cases where the value was not reported.
POUNDS	POUNDS is live weight, (in the shell)
LANDED	LANDED can be meat weights or shell weights, but is usually meats
TRIP_LENGTH	Trip length is in days; It is calculated from the elapsed time between the date-time sailed and date-time landed. This is a measure of days absent.

Ports – currently assuming that a trip departs from the same place it lands. An alternate assumption is that people just leave from the place where they made their last landing.

The DDLAT and DDLOn are self reported lat-lons from logbooks(VTRs). We have supplemented this with some extra information.

Column	Description
TRIP_ID	
DOCID	VTR DOCUMENT table record identifier; Primary key, internally generated at scanning based on vessel id and date/time sailed. Each DOCID represents one trip; equivalent to TRIPID in VESLOG tables.
ACTIVITY_CODE	Complicated set of letters and numbers. See below
PLAN_CAT	LGC_A LGC_B, LGC_C, SC_2, SC_3, SC_4, SC_5, SC_6, SC_7, SC_8, SC_9, SG_1A, SG_1B are a collection of true and false variables the indicate if the vessel had a particular permit when the trip was taken.

ACTIVITY_CODE and a set of PLAN_CAT categorical variables are used from the live DMIS tables. We join using TRIP_ID, DOCID. See here

Description of the VTR data.

Column	Description
TRIPID	VESLOG Trip record identifier, which is generated internally and used for linking
tripcatg	(only commercial categories are selected), recreational and RSA/EFP are not.
operator	Name of the captain
opernum	Captains Identification number
permit	Six-digit vessel fishing permit number assigned by the NE Regional Office permit system
nsubtrip	Number of subtrips (see description of subtrips
crew	number of crew, including captain
port	6 digit numeric code for the port, renamed to VTR_PORTNUM to make clear is a companion to VTR_PORT and VTR_STATE

Description of the SPACEPANELS data.

Column	Description
TRIPID	VESLOG Trip record identifier, which is generated internally and used for linking
geoid	10 digit county subdivision from US Census.
state_fips	2 digit state fips code
portlnd1	string of the name of the port that the vessel operator writes on the VTR.
state1	2 letter abbreviation of the state
port	6 digit port code. Should match vtr_portnum from VTR
namelsad	Name and Legal/Statistical description
port_lat	Latitude of the geoid
port_lon	longitude of the geoid
previous_geoid	Geoid of landing port for previous trip
previous_state_fips	2 digit state fips code for previous trip
previous_namelsad	Name and Legal/Statistical description for previous trip
previous_port_lat	Latitude of the geoid for previous trip
previous_port_lon	longitude of the geoid for previous trip

The spacepanels data tidies up vtr ports and aggregates them to the US Census county subdivision. You should *not* expect an exact match between portlnd1, state1, and port in the spacepanels dataset compared to the same columns in the raw vtr because some data clean was done on these fields. The code is in the spacepanels repo, "just_ports.do."

- `Geoid` is geoid10. The lat-lons are either the centroid of the geoid and/or adjusted to the coast. There is probably some error here, but if the goal is to use these points to help construct distances or costs to go fishing, they are probably accurate enough.
- `namelsad` is a convenient name (Like "Boston city"). However, be aware that there are some places with the same name, so use either the `geoid` or the `namelsad` plus `state` or `state_fips`. This could also be solved by using the `namelsad` as factor levels This dataset includes all trips from 1996-2021. Use the vintage date appended to the end of the file to assess whether the final year of data is "complete" enough for your purposes. The following corresponds to all trips, not just scallop trips:
- Missing `tripids`. there are missing tripids prior to 2003. These correspond to SCOQ.
- Missing `vtr_portnum` There are about 183 obs. These are 2019-2021 and probably reflect changes in the underlying data that haven't been picked up and cleaned in the code. I am not going to deal with these.
- Missing `geoid`. There are about 3,000 obs. Many of these are because the vtr_port is "Other State". A few are missing because of a new port.
- There is a set of `previous_` variables. These were constructed using this code:

```
bysort permit (datelnd1 tripid): gen previous_geoid=geoid[_n-1]
bysort permit (datelnd1 tripid): gen previous_namelsad=namelsad[_n-1]
bysort permit (datelnd1 tripid): gen previous_state_fips=state_fips[_n-1]
```

- It's possible that the previous trip was the same day. It's also possible that the previous trip was from years before.
  *The date that I am using here is the `datelnd1` field from VESLOG_T, if those are somehow missing, I have used datesold from VESLOG_S. I am using something from the Clam logbooks, but I'm not positive as to what. I normally used the datesold in VESLOG_S for spacepanels.

- More of the `previous_` variables are missing. This is expected, because the first observation of a permit will be missing something. I don't think this will cause too much of a problem. Some will be missing if the prior trip was a "other state."

Description of the VMS data.

We are currently not using the VMS data directly.

Description of the Cost data.

Column	Description
Place	holder
Place	holder

Load Offshore Wind Tool Data sets

The frozen DMIS table from the offshoreWind project (APSD_DMIS_2) is the base dataset for the analysis. The DMIS data are formed by combining many datasets, including VTR and Dealer. In brief, the APSD_DMIS_2 dataset contains a mix of trip attributes (port, date), sub-trip attributes (gear, location) , and catch outcomes (species, pounds, landed, dollar). You can read more about DMIS here.

```
load(file.path(offshoreWind_directory, "APSD_DMIS_2.rda"))
#load("~/offshoreWind-master/data-raw/REVENUEFILE.Rdata")
#load("~/offshoreWind-master/data/APSD_DMIS_2.Rdata")
```

Read in Port lats and lons

Load in the lats and lons of all ports from the spacepanels directory. Change the column names to clarify that the lat-lons are the ports. Join this to the APSD_DMIS_2 dataset, keeping all rows of the APSD_DMIS_2 dataset and dropping any rows from tripids_geoids that do not match.

```
#Import Data
tripid_geoids<- haven::read_dta(file.path(spacepanels_directory,"just_ports_2022_01_26.dta"))
#Could do all this in one step, but ...
# Pick the cols that start with previous
prev<-tripid_geoids[grepl("^previous_", colnames(tripid_geoids))]
#pick the rest of the cols
tripid_geoids<-tripid_geoids[c("tripid","geoid", "namelsad", "state_fips","port_lat","port_lon")]
#cbind the two together
tripid_geoids<-cbind(tripid_geoids,prev)

APSD_DMIS_2<-merge(APSD_DMIS_2,tripid_geoids, by.x="DOCID", by.y="tripid", all.x=TRUE, all.y=FALSE)
```

This match should be pretty good for source=DMIS. But it will not work for source=SFCLAM.

Loading Data fom Oracle

The APSD_DMIS_2 table must be supplemented with additional data. This section queries the Oracle databases to extract additional information.

VTR Data

Some Trip-level data in the VTR schema is needed. See table at the top. We extract them here.

```
oracle_server = "sole"
ODBC.CONNECTION <- RODBC::odbcConnect(dsn=oracle_server, uid=oracle_username, pwd=oracle_password, beli
START.YEAR = 2007
END.YEAR = 2019
for(i in START.YEAR:END.YEAR) {
  print(i)
  CURRENT.QUERY = paste("SELECT VTR.veslog",i,"t.TRIPID,tripcatg, operator, opernum, permit, nsubtrip, 
                FROM VTR.veslog",i,"t", sep="")
  YEAR.RESULT = sqlQuery(ODBC.CONNECTION, CURRENT.QUERY)

  # Now, the loop compiles the results; the first year must be treated slightly differently###
  if (i==START.YEAR) {
    RESULT.COMPILED = YEAR.RESULT
  } else {
    RESULT.COMPILED = rbind(RESULT.COMPILED, YEAR.RESULT) }
}    # End Main Loop


##Subtrip Data

## Note: This data is pulled in order to fill in a large number of blanks in reporting

  CURRENT.QUERY = paste ("SELECT VTR.veslog",i,"t.TRIPID,tripcatg, operator, opernum, permit, nsubtrip,
                FROM VTR.veslog",i,"t", sep="")
  VTR.veslog2019t = sqlQuery(ODBC.CONNECTION, CURRENT.QUERY)
```

Scallop LA IFQ Linking variables

We also extract the activity code from DMIS. This will describe the type of trip that the vessel has declared into. The most important types of trips will be Scallop Trips; however, fishing vessels with the proper permits are allowed to retain scallops while declared into other fisheries. When this happens, the volume of scallops will be much lower.

We also extract the T/F variables corresponding the the PLAN_CAT in DMIS.

```
oracle_server = "sole"
CURRENT.QUERY = paste ("SELECT TRIP_ID, DOCID, ACTIVITY_CODE, LGC_A, LGC_B, LGC_C, SC_2, SC_3, SC_4, SC
 FROM APSD.t_ssb_trip_current@garfo_nefsc")
Scallop_Linkingorg = sqlQuery(ODBC.CONNECTION, CURRENT.QUERY)

odbcCloseAll()
```

We classify the trips as FullTime, PartTime based on these PLAN_CAT variables. We also generate categorical variables corresponding to LA and GC columns. Note that a vessel can hold both an LA and

a GC permit at the same time. The summary tables below will have lots of observations corresponding to Scallop_Linkingorg[ftpt]=0, LA=0, and GC=0. This is expected. because it has everything from DMIS.

```
# Bin the LA vessels into full time or part time.
Scallop_Linkingorg$ftpt<-"None"
Scallop_Linkingorg$ftpt[Scallop_Linkingorg$SC_2=="TRUE"]<-"FullTime"
Scallop_Linkingorg$ftpt[Scallop_Linkingorg$SC_5=="TRUE"]<-"FullTime"
Scallop_Linkingorg$ftpt[Scallop_Linkingorg$SC_7=="TRUE"]<-"FullTime"

Scallop_Linkingorg$ftpt[Scallop_Linkingorg$SC_6=="TRUE"]<-"PartTime"
Scallop_Linkingorg$ftpt[Scallop_Linkingorg$SC_3=="TRUE"]<-"PartTime"

# Construct a logical variable for GC
Scallop_Linkingorg$GC<-(Scallop_Linkingorg$LGC_A=="TRUE" | Scallop_Linkingorg$LGC_B=="TRUE" | Scallop_Li

# Construct a logical variable for LA
Scallop_Linkingorg$LA<-(Scallop_Linkingorg$ftpt=="PartTime" | Scallop_Linkingorg$ftpt=="FullTime")


#Make some tables
table(Scallop_Linkingorg$ftpt)
```

```
##
## FullTime     None PartTime
##    83353  4372387    16849
```

```
table(Scallop_Linkingorg$GC)
```

```
##
##   FALSE    TRUE
## 3929661  541190
```

```
table(Scallop_Linkingorg$LA,Scallop_Linkingorg$GC)
```

```
##
##           FALSE    TRUE
##   FALSE 3881029  489620
##   TRUE    48632   51570
```

```
#Select certain columns
Scallop_Linkingorg_bak<-Scallop_Linkingorg
Scallop_Linkingorg<-dplyr::select(Scallop_Linkingorg, c(TRIP_ID,DOCID, ACTIVITY_CODE, ftpt, GC,LA))

is.logical(Scallop_Linkingorg$GC)
```

```
## [1] TRUE
```

```
is.logical(Scallop_Linkingorg$LA)
```

```
## [1] TRUE
```

We don't want to create a single plan column, because a vessel could have multiple kinds of scallop permits. Instead, if we want just the Fulltime LA vessels, we can do something like:

```
Limited_Access <-Scallop_Linkingorg %>%filter(LA=="TRUE")
Limited_Access_ft<-Limited_Access %>%filter(ftpt=="FullTime")
```

Data Cleaning

1. Filter down to only Scallop Species
2. Seperate Dates & Times and Delete Old Dates Column
3. Delete Columns that are not need
4. NESPP3 & SOURCE Values do not vary across the observations, so these two columns can be deleted

```
Scallops <- APSD_DMIS_2 %>% filter (SPPNAME == "SCALLOPS/BUSHEL")

#Separate Dates & Times
Scallops$Date <- as.Date(Scallops$DATE_TRIP)
Scallops$Time <- format(Scallops$DATE_TRIP,"%H:%M:%S")

#Drop columns that are not needed
Scallops$DATE_TRIP<- NULL
Scallops$NESPP3<- NULL
Scallops$SOURCE<- NULL
```

Merging

1. Merge Scallops & VTR Data Sets (RESULT.COMPILED). We keep all columns from both the APSD_DMIS_2 and RESULT.COMPILED datasets. We also:
    1. Filter out 2020 values
    2. Delete Extra PERMIT Column because there were a few missing values.
    3. Delete all TRIPCATG that are not 1. This isolates all commercial trips
    4. Drop rows corresponding to a "Not Fished" VTR.
2. Join the output of (1) with Activity Codes
3. Verify that we get what we think we should get.

```
##1. Merge Scallops & VTR Data Sets (RESULT.COMPILED). We keep all columns from both the APSD_DMIS_2 an

# all.x = TRUE & all.y = FALSE means I am keeping data with no match from DMIS table but dropping data

# DOCID is used because of the following found in the data dictionary "VESLOG Trip record identifier, w
VTR_DMIS_merge <- merge(RESULT.COMPILED,Scallops, by.x = "TRIPID", by.y = "DOCID", all.x = FALSE, all.y

## Filter out 2020 values
VTR_DMIS_merge <- VTR_DMIS_merge %>% filter(YEAR <= END.YEAR)

# Delete Extra PERMIT Column
## Note: X was deleted because PERMIT.y had zero NAs and PERMIT.x had 25
VTR_DMIS_merge$PERMIT.x <- NULL

# Delete all TRIPCATG that are not 1. This isolates all commercial trips
## Type of trip: 1=Commercial; 2=Party; 3=Charter; 4=RSA/EFP. Note: RSA/EFP landings represent a small

VTR_DMIS_merge <- VTR_DMIS_merge %>% filter(TRIPCATG == "1")
VTR_DMIS_merge$TRIPCATG <- NULL

# Delete all NOT_FISHED that are not 0. This indicates whether the 'Did not fish' box was checked on th
VTR_DMIS_merge <- VTR_DMIS_merge %>% filter(NOT_FISHED == "0")
VTR_DMIS_merge$NOT_FISHED <- NULL
```

```
## 2.
###Join VTR & DMIS Data with Activity Codes

# Delete duplicate rows; These are rows that share the same TRIPID, DOLLAR,LANDED, & TRIP_LENGTH
## Note: VTRs are self-reported and there is a potential for records to be submitted to regional office
VTR_DMIS_AC <- merge(VTR_DMIS_merge,Scallop_Linkingorg, by.x = "TRIPID", by.y = "DOCID", all.x = TRUE, a
VTR_DMIS_AC <- VTR_DMIS_AC %>% distinct(TRIPID,DOLLAR,TRIP_LENGTH,LANDED, .keep_all = TRUE)

#Split Activity codes to allow for easier data management. VMS Declaration code book is broken down by
VTR_DMIS_AC$ACTIVITY_CODE <- as.character(VTR_DMIS_AC$ACTIVITY_CODE)
VTR_DMIS_AC <- VTR_DMIS_AC %>% separate(ACTIVITY_CODE, into = c('Plan Code','Program Code','Area Identi
```

```
## Warning: Expected 4 pieces. Missing pieces filled with `NA` in 158536 rows [1,
## 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
VTR_DMIS_AC$`-` <- NULL



## 3.
### Testing Reported NAs in new data set (that they are relatively even across all years)
#### Note: The variable used in this command can be substituted for whatever needs to be tested. In thi

testing <- VTR_DMIS_AC %>%
group_by("YEAR") %>% filter(is.na(OPERNUM))
```

Add in cost data

These data should reference the "Estimation of Commercial Fishing Trip Costs Using Sea Sampling Data" paper by Samantha Werner & Geret DePiper. We will likely use the Winsorized trip cost estimates.

```
#Import Data
X2007_2012 <- read_excel(file.path(cost_directory,"2007_2012.xlsx"))
X2013_2020 <- read_excel(file.path(cost_directory,"2013_2020.xlsx"))

#Merge 2007-2012 costs with 2013-2020 costs
all_yrs_costs <- merge(X2007_2012,X2013_2020, all = TRUE)
## Created two sets of cost joins.
### 1. Before LA Estimation
### 2. After LA Estimation
VTR_DMIS_AC <- merge(VTR_DMIS_AC,all_yrs_costs, by.x = "TRIPID", by.y = "VTR_TRIPID", all.x = TRUE,all.
```

Data Aggregating Trip Revenues & Delete duplicate TRIPIDs

Subtrips are generated when a vessel switches gear or statistical areas. Subtrips have identical TRIPID/DOCID. A trip may have many (8+) subtrips, but the majority of trips observed only have one subtrip (95.7% using original VTR & DMIS merged data set). If a trip has just 1 subtrip, the trip took place in a single statistical area. If a trip crosses four different statistical areas, the NSUBTRIP is then equal to 4, and the landings, value, latitude, and longitude are reported separately for each area.

```
table(VTR_DMIS_AC$NSUBTRIP)
```

```
##
##      1      2      3      4      5      6      7      8
## 165486   2890    631    229     79      2      7      1
```

Since our goal is to estimate a choice model at the trip level, we need to construct trip level variables. We retained the subtrip attributes (GEARCODE, DDLAT, DDLON) corresponding to the subtrip with the highest DOLLAR. We constructed trip-level values for revenue, pounds, and landed (DOLLAR, POUNDS, LANDED). The trip level variables are prefixed with "Agg_".

1. Aggregate DOLLAR, POUNDS, LANDED
2. Add back into original data set
3. Check / Test Maximum DOLLAR values by grouping by TRIPID
4. Drop duplicate TRIPIDs by keeping maximum DOLLAR values

This may not be realistic. There are anecdotes of vessels fishing in one spot on the way to another, further offshore spot. Subtrips may be a bigger issue when we extend to other fisheries.If we have the ability to model fishing choices at a finer scale than at a trip, this can be modified fairly easily.

DEPRECATED - Code to aggregate subtrip landings to subtrips.

We are now pulling subtrips along to the end instead of aggregating. If you want to contract multi-area or multi-gear trips down to a single observation, this is how you would do it.

```
### 1. Aggregate DOLLAR, POUNDS, LANDED
Agg_DOL_POUN_LAND <- VTR_DMIS_AC %>%
  group_by(TRIPID) %>%
 summarise(Agg_DOLLAR = sum(DOLLAR), Agg_POUNDS = sum(POUNDS), Agg_LANDED = sum(LANDED))

#### Testing to make sure there are no duplicates in TRIPID groups; this should equal 0
sum(duplicated(Agg_DOL_POUN_LAND$TRIPID))
stopifnot(sum(duplicated(Agg_DOL_POUN_LAND$TRIPID))==0)

###  2. Add back into original data set
#### all = FALSE is used to keep only rows that match from the data frames
VTR_DMIS_AC_Agg <- merge(VTR_DMIS_AC,Agg_DOL_POUN_LAND, by.x = "TRIPID", by.y = "TRIPID", all.x = TRUE,




### 3.Parse out Maximum Dollar amounts in order to drop lesser subtrips
VTR_DMIS_AC_Agg <- VTR_DMIS_AC_Agg %>% group_by(TRIPID) %>% filter(DOLLAR == max(DOLLAR))
### Another way to check this is by running the following code: VTR_DMIS_AC_Agg %>% group_by(TRIPID) %>

## Test out
sum(duplicated(VTR_DMIS_AC_Agg$TRIPID))

stopifnot(sum(duplicated(VTR_DMIS_AC_Agg$TRIPID))==0)
```

Trips reported on land will be dropped from observations

```
#################################################################################################
# change these variables to read in the veslogDMISmerge and what the network path to the shared drive i
coordinate_table_input <-VTR_DMIS_AC
lat_column = "DDLAT"
lon_column = "DDLON"
shapefile_path<-East_Cst_crop_2020_path
#################################################################################################
shapefile_path_to_spatialpolygons <- function(shapefile_path,
                                              projection = CRS("+proj=longlat +datum=NAD83 +no_defs +el

  # shapefile_path = "C:/Users/dennis.corvi/Documents/R/Projects/OffshoreWindDev/offshoreWind/areas_min
  # projection = CRS("+proj=longlat +datum=NAD83 +no_defs +ellps=GRS80 +towgs84=0,0,0")
  layer_name = unique(gsub(pattern="(.+)(.shp$)","\\1", ignore.case = TRUE , list.files(path=shapefile_
  if (length(layer_name)==0) {
    stop("Shapefile path does not contain a shapefile")
  }
  if (length(layer_name) > 1) {
    file_list <- list.files(shapefile_path, pattern = "*shp$", full.names = TRUE)
    shapefile_list <- lapply(file_list, sf::read_sf)
    all_shapes <- sf::st_as_sf(data.table::rbindlist(shapefile_list))
    all_shapes <- all_shapes[,(names(all_shapes) %in% c("Name"))]
    all_shapes <- sf::as_Spatial(all_shapes, cast = TRUE, IDs = paste0("ID", seq_along(from)))
    all_shapes@data$NAME <- all_shapes@data$Name
    all_shapes@data$Name <- NULL
  } else { # if only one shape
    all_shapes <- rgdal::readOGR(dsn=shapefile_path, layer=layer_name, verbose=F)
  }
  all_shapes <- spTransform(all_shapes, CRS=projection)
  return(all_shapes)
}
#################################################################################################

crs = CRS("+proj=longlat +datum=NAD83 +no_defs +ellps=GRS80 +towgs84=0,0,0")
shapefile_area <- SpatialPolygonsDataFrame(aggregate(shapefile_path_to_spatialpolygons(shapefile_path,
```

```
## Loading required namespace: rgeos
```

```
coordinate_table <- as_tibble(coordinate_table_input %>%
                                rename("LAT" = .data[[lat_column]], "LON" = .data[[lon_column]]) %>%
                                drop_na(LON, LAT) %>%
                                mutate(LON = if_else(LON>1, LON*-1, LON )) %>%
                                relocate(LON, LAT)) # drop LAT LON NAs, correct LON, change column orde

xy <- coordinate_table[,c(1,2)]
coordinate_table <- SpatialPointsDataFrame(coords = xy, data = coordinate_table, proj4string = crs)
coordinate_table <- spTransform(coordinate_table, CRSobj = crs)

vtridx <- over(coordinate_table, shapefile_area)

colnames(vtridx)[1] <- "NAME"

coordinate_table$Area <- vtridx$NAME
```

```
coordinate_table <- coordinate_table@data

VTR_DMIS_AC <- coordinate_table %>%
  mutate_if(is.factor, as.character) %>%
  mutate(Area = if_else(is.na(Area), "Non-land", Area)) %>% # change NAs to read "Non-land"
  rename("{lat_column}" := LAT, "{lon_column}" := LON) %>%  # change lat lon columns back to original n
  filter(Area == "Non-land")
#Delete Area Variable; Served its purpose as a filter
VTR_DMIS_AC$Area <- NULL
```

```
# Spatial join with ten minute squares

## Read in your shapefile
### Note: Viewing the table after this is done is helpful to ensure that the shapefile looks how you ex

## Import the data set you want to combine with your imported shape file


TMSQ_sp <- st_read(TMSQ_path)
```

```
## Reading layer 'Ten_Minute_Squares_Clip6' from data source
##   '/net/home2/mlee/Effort-Displacement---Scallop/data/external/shapefiles/Ten Minute Squares Cut Nor
##   using driver 'ESRI Shapefile'
## Simple feature collection with 2410 features and 15 fields
## Geometry type: POLYGON
## Dimension:     XYZ
## Bounding box:  xmin: -77.33333 ymin: 35.33333 xmax: -65 ymax: 45.33333
## z_range:       zmin: 0 zmax: 0
## Geodetic CRS:  NAD83
```

```
## Run the below chunk to see your shapefile plotted out
#qtm(TMSQ_sp) + tm_legend(show = FALSE)


point_geo <- st_as_sf(VTR_DMIS_AC,
                      coords = c(x  = "DDLON", y = "DDLAT"), crs = crs )

final_product <- st_join(point_geo, TMSQ_sp, join = st_within)
```

```
## st_as_s2(): dropping Z and/or M coordinate
```

```
#This chunk uses a "within" join, but other options are available using the sf package 1.0-6.
## st_intersects,st_disjoint,st_touches,st_crosses,st_within,st_contains,st_contains_properly,st_overla

#Delete unnecessary variables from join: keep the geometry, MN30SQID, and MN10SQID columns
final_product[,c('MN10SQROW','MN10SQCOL','POINT_Y','POINT_X','XTXT','YTXT','DG1SQLAT','DG1SQLON','DG1SQ

#final_product$geometry_old<-final_product$geometry
#Lease Area Joins

lease_sp <- st_read(All_Lease_Areas_Shapefile_path)
```

```
## Reading layer 'All_Lease_Areas' from data source
##   '/net/home2/mlee/Effort-Displacement---Scallop/data/external/shapefiles/All_Lease_Areas_Shapefile'
##   using driver 'ESRI Shapefile'
## Simple feature collection with 27 features and 1 field
```

```
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -75.49862 ymin: 36.14111 xmax: -70.02155 ymax: 41.29879
## Geodetic CRS:  NAD83
```

```
## Run the below chunk to see your shapefile plotted out
#qtm(lease_sp) + tm_legend(show = FALSE)

## This chunk uses the current data set, converts it into a sf geospatial object, and bins it into the

point_geo_lease <- st_as_sf(final_product,
                      coords = c(x  = "DDLON", y = "DDLAT"), crs = crs )

final_product_lease <- st_join(point_geo_lease, lease_sp, join = st_within)

#geometry carries over all the way from the initial read in.
identical(final_product_lease$geometry, point_geo$geometry)
```

```
## [1] TRUE
```

```
#however, we don't need the geometry colum, it's just a combination of the DDLAT and DDLON
final_product_lease[,c('geometry')]<-NULL

#This chunk uses a "within" join, but other options are available using the sf package 1.0-6.
## st_intersects,st_disjoint,st_touches,st_crosses,st_within,st_contains,st_contains_properly,st_overla









#save to RDS
final_product_savename<-paste0("final_product_lease",vintage_string,".Rds")
saveRDS(final_product_lease, file=here("data","main",final_product_savename))

# to read this in, you will want to do the here::i_am dance and then read in
# final_product_savename<-paste0("final_product_lease",vintage_string,".Rds")
# final_product_lease<-readRDS(here("data","main",final_product_savename))
```

Some summary statistcs

```
# Here is a placeholder where we will make a few summary statistics tables.  Nothing too fancy. This ma
# This has not been tested. Yet.

summary(final_product_lease)
```

```
##      TRIPID              OPERATOR            OPERNUM             NSUBTRIP
##  Min.   :2.679e+06   Length:165868      Min.   :  410392   Min.   :1.000
##  1st Qu.:3.157e+06   Class :character   1st Qu.:10002645   1st Qu.:1.000
##  Median :4.020e+06   Mode  :character   Median :10009375   Median :1.000
##  Mean   :6.238e+11                      Mean   :10008984   Mean   :1.031
##  3rd Qu.:4.891e+06                      3rd Qu.:10014818   3rd Qu.:1.000
##  Max.   :4.105e+13                      Max.   :10024074   Max.   :8.000
##                                         NA's   :4370
```

```
##       CREW          VTR_PORTNUM          IMGID                YEAR
##  Min.   : 1.000   Min.   : 71011   Min.   :2.468e+06   Length:165868
##  1st Qu.: 3.000   1st Qu.:240403   1st Qu.:2.874e+06   Class :character
##  Median : 3.000   Median :330127   Median :3.755e+06   Mode  :character
##  Mean   : 3.843   Mean   :299650   Mean   :6.228e+13
##  3rd Qu.: 5.000   3rd Qu.:330309   3rd Qu.:4.657e+06
##  Max.   :33.000   Max.   :499101   Max.   :4.105e+15
##  NA's   :138      NA's   :3        NA's   :4
##    VTR_PORT          VTR_STATE          TRIP_LENGTH         PERMIT.y
##  Length:165868      Length:165868      Min.   : 0.0000   Min.   :110681
##  Class :character   Class :character   1st Qu.: 0.5938   1st Qu.:231428
##  Mode  :character   Mode  :character   Median : 0.9167   Median :310979
##                                        Mean   : 2.6138   Mean   :285436
##                                        3rd Qu.: 2.7083   3rd Qu.:330784
##                                        Max.   :24.7500   Max.   :550026
##
##    DEALNUM              DOLLAR              POUNDS              LANDED
##  Length:165868      Min.   :      0.5   Min.   :      0.5   Min.   :     0.29
##  Class :character   1st Qu.:   2141.0   1st Qu.:   2069.9   1st Qu.:   250.00
##  Mode  :character   Median :   3928.0   Median :   3332.0   Median :   400.00
##                     Mean   :  35550.8   Mean   :  31911.1   Mean   :  3834.71
##                     3rd Qu.:   8697.7   3rd Qu.:   6147.0   3rd Qu.:   750.00
##                     Max.   :1413380.0   Max.   :1186125.0   Max.   :142392.00
##
##    GEARCODE         SECGEARFISH          SPPNAME              geoid
##  Length:165868      Length:165868      Length:165868      Min.   :9.008e+08
##  Class :character   Class :character   Class :character   1st Qu.:2.501e+09
##  Mode  :character   Mode  :character   Mode  :character   Median :3.401e+09
##                                                           Mean   :3.106e+09
##                                                           3rd Qu.:3.403e+09
##                                                           Max.   :5.170e+09
##                                                           NA's   :1774
##    namelsad           state_fips       port_lat        port_lon
##  Length:165868      Min.   : 7.00   Min.   :34.71   Min.   :-76.86
##  Class :character   1st Qu.:24.00   1st Qu.:39.57   1st Qu.:-74.23
##  Mode  :character   Median :33.00   Median :40.87   Median :-72.52
##                     Mean   :29.92   Mean   :40.60   Mean   :-72.64
##                     3rd Qu.:33.00   3rd Qu.:41.64   3rd Qu.:-70.93
##                     Max.   :49.00   Max.   :44.95   Max.   :-66.98
##                     NA's   :1643    NA's   :1774    NA's   :1774
##  previous_geoid      previous_namelsad  previous_state_fips previous_port_lat
##  Min.   :9.008e+08   Length:165868      Min.   : 7.00       Min.   :40.68
##  1st Qu.:2.501e+09   Class :character   1st Qu.:24.00       1st Qu.:40.68
##  Median :3.401e+09   Mode  :character   Median :33.00       Median :40.68
##  Mean   :3.110e+09                      Mean   :29.96       Mean   :40.92
##  3rd Qu.:3.403e+09                      3rd Qu.:33.00       3rd Qu.:40.92
##  Max.   :5.181e+09                      Max.   :49.00       Max.   :41.63
##  NA's   :1882                           NA's   :1749        NA's   :165864
##  previous_port_lon      Date                Time             TRIP_ID
##  Min.   :-73.32    Min.   :2007-05-01   Length:165868      Length:165868
##  1st Qu.:-73.32    1st Qu.:2009-05-27   Class :character   Class :character
##  Median :-73.32    Median :2012-08-17   Mode  :character   Mode  :character
##  Mean   :-72.79    Mean   :2012-12-30
##  3rd Qu.:-72.79    3rd Qu.:2016-06-17
```

```
##  Max.   :-71.21    Max.   :2019-12-31
##  NA's   :165864
##   Plan Code          Program Code       Area Identifier        ftpt
##  Length:165868      Length:165868      Length:165868      Length:165868
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      GC              LA              hours          DB_LANDING_YEAR
##  Mode :logical   Mode :logical   Min.   :  0.0333   Min.   :2007
##  FALSE:34428     FALSE:117141    1st Qu.: 14.2500   1st Qu.:2009
##  TRUE :131381    TRUE :48727     Median : 22.0000   Median :2012
##  NA's :59                        Mean   : 62.7184   Mean   :2013
##                                  3rd Qu.: 65.0000   3rd Qu.:2016
##                                  Max.   :594.0000   Max.   :2019
##                                  NA's   :1          NA's   :1
##  TRIP_COST_2020_DOL TRIP_COST_WINSOR_2020_DOL OBSERVED_COST_DUMMY
##  Min.   :   16.78   Min.   :   29.47          Min.   :0.00000
##  1st Qu.:  637.13   1st Qu.:  637.13          1st Qu.:0.00000
##  Median : 1223.95   Median : 1223.95          Median :0.00000
##  Mean   : 4742.65   Mean   : 4699.89          Mean   :0.04871
##  3rd Qu.: 5377.22   3rd Qu.: 5376.89          3rd Qu.:0.00000
##  Max.   :52122.12   Max.   :30595.61          Max.   :1.00000
##  NA's   :1          NA's   :1                 NA's   :1
##     MN30SQID        MN10SQID          NAME
##  Min.   :35734   Min.   :357311   Length:165868
##  1st Qu.:39731   1st Qu.:397331   Class :character
##  Median :40714   Median :407121   Mode  :character
##  Mean   :40560   Mean   :405618
##  3rd Qu.:41691   3rd Qu.:416922
##  Max.   :44691   Max.   :446966
##
```

```
table(final_product_lease$YEAR)
```

```
##
##  2007  2008  2009  2010  2011  2012  2013  2014  2015  2016  2017  2018  2019
## 15905 18396 16483 11308 12546 12082 10970 10022 10728 12750 10922 12152 11604
```

```
table(final_product_lease$GEARCODE)
```

```
##
##     DREDGE-CLAM    DREDGE-OTHER  DREDGE-SCALLOP   GILLNET-OTHER    GILLNET-SINK
##            7996              35          130745               1              98
##        HANDLINE LONGLINE-BOTTOM           OTHER       POT-OTHER     SEINE-OTHER
##              99               3            4308              94               1
##    TRAWL-BOTTOM
##           22488
```

```
table(final_product_lease$ftpt)
```

```
##
## FullTime     None PartTime
##    42465   117141     6262
```

```
table(final_product_lease$VTR_STATE)
```

```
##
##    CT    DE    MA    MD    ME    NC    NH    NJ    NY    RI    VA
##  2371   196 58486  6162  5319   453  1831 63180 13779  7514  6536
```

```
table(final_product_lease$`Plan Code`)
```

```
##
##    DOF    HER    MID    MNK    NMS    SCO    SES    SMB
##   3459      2     12    431   5644   7280 139898    222
```

```
table(final_product_lease$`Program Code`)
```

```
##
##   BDP   CML   COM   DOF   HER   MMQ   MNK   MUL   NAC   NAF   NAS   NMA   OQU
##     6  1221   324  2190     2    54   161  1944     1     1    15     6  4462
##   PWD   REC   RSA   SAA   SAC   SAM   SAS   SCA   SCF   SCG   SCI   SEC   SFC
##   264     5   557 22496     1    13    67 18086     1 98498     4  2509  2764
##   SLM   SMA   SQI   SQL   SQM   SWE   TSP   TST   USC
##    25   327     4   134    59    11     3    35   698
```

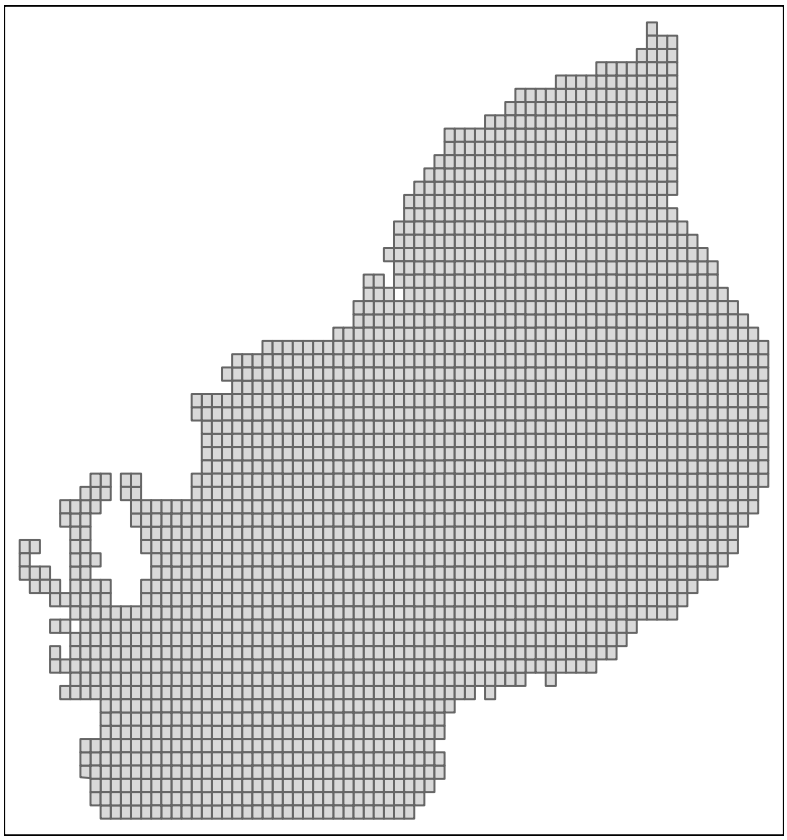

Figure 1: 10 minutes squares

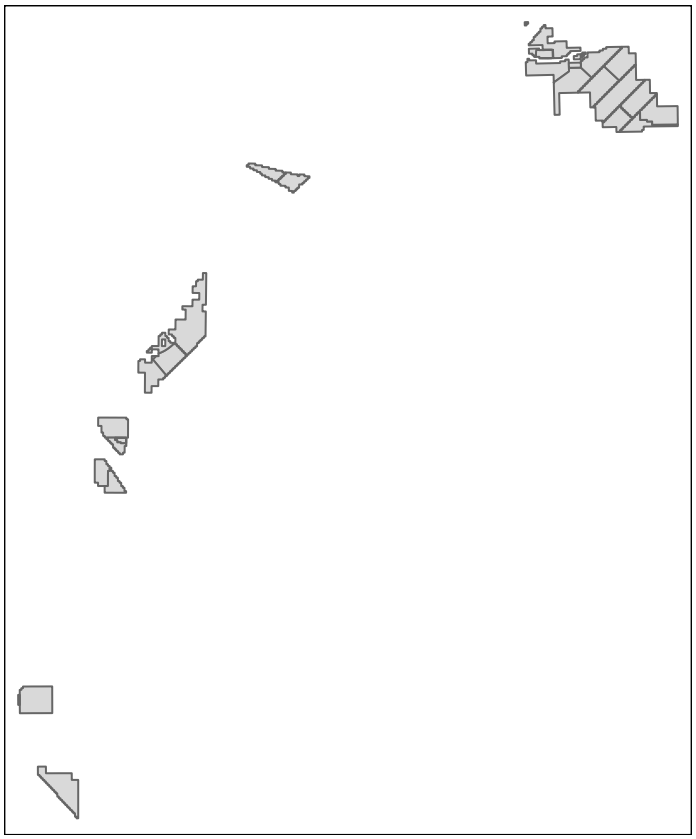

Figure 2: Wind Energy Areas

Here is some code that we are no longer using.

Code to filter on the Limited Access (LA) Fleet using landings and crew size

We considered filtering out the LADAS scallop fleet by using landings greater than or equal to 850 pounds and Crew less than or equal to 8. These are based on crew limits. We are using the activities codes instead. In summary:

FY2007-2014: No limit on crew (except for 7 in DMV starting in FY2014) FY2015-2019: 8

> Initially, vessels had the same crew limits in access areas as they did on DAS. However, Framework 18(fishing year 2006) eliminated the seven-person crew limit (five-person limit for small dredge category vessels) for scallop access area trips. The purpose of this was to eliminate inefficiencies caused by the crew limit for fishing activity that is limited by a possession limit. The crew limit was established to control vessels' shucking capacity when fishing under DAS.
>
> Eight years later, Framework 25 (fishing year 2014) imposed a crew limit of seven individuals (the same as DAS) per limited access vessel (five-person limit for small dredge category vessels) in DMV. The purpose of this was to protect small scallops and discourage vessels from highgrading.
>
> Framework 26 (fishing year 2015) implemented crew limits for all access areas. In an effort to protect small scallops and discourage vessels from high-grading. Framework 26 imposed a crew limit of eight individuals (one extra from DAS) per limited access vessel, including the captain, when fishing in any scallop access area. If a vessel is participating in the small dredge program, it may not have more than six people (one extra from DAS) on board, including the captain, on an access area trip.
>
> Finally, because the scallops in the NLS–S–D were expected to have lower yield than similar sized scallops in other areas, Framework 32 (fishing year 2020) allowed two additional crew members aboard both limited access full-time (10 in total) and limited access full-time small dredge vessels (8 in total). This allowed vessels to add additional crew members to increase the shucking capacity of the vessel and reach the possession limit in a time more consistent with other access areas. (Travis Ford @ GARFO - Nov 17,2021)

FY2007-2014: No limit (except for 7 in DMV starting in FY2014) FY2015-2019: 8

```
LA_Estimate <- VTR_DMIS_AC_Agg %>% filter(Agg_LANDED >= 850 & CREW <= 8)
```