CVAS Data Flow Methods

Arthur Barros

2024-05-10

Objectives

The aim of this document is to show how to run the 'data_cleaning.R', 'expansions.R', and 'summary.R' scripts to load and clean data from the Central Valley Angler Survey (CVAS) database and automatically prepare it for publication in a machine readable and open source format. The end results of this script is a csv file containing the effort and harvest estimates for each survey location for the annual survey period, as well as a series of intermediate csv files and scripts that are provided for transparency. This process aims to automate the data cleaning, QC, and analysis of CVAS data reporting in order to save labor time.

Establishing the environment

The first part of the script sets the environment for the R project by first clearing the environment with the `rm(list=ls())` call. We can then load all the relevant packages, which will just be the 'tidyverse' package for this project:

```r
rm(list=ls())
library(tidyverse)
```

Data Preparation

Before we calculate effort and harvest data using the survey data, we need to load, clean, format, and prepare the tables produced by the database. The data preparation is handled in it's own separate "data_prep.R" script, with outputs being saved as .rds files that can later be loaded into different scripts for estimate calculation.

Pull Data

Typically I would try to use the "bridgeAccess.R" script to pull the data straight from the access database, but because that is behind some security measures, that's currently impossible. Instead, we can export the necessary tables from the database as csv files, and then load them here with the `read.csv()` call. We'll be importing ten different tables into our project:

```r
tblcountSurv<-read.csv("data/tblcountSurv.csv")
tblGeoLoc<-read.csv("data/tblGeoLoc.csv")
tblcountDetail<-read.csv("data/tblCountDetail.csv")
tbl2ndcountSurv<-read.csv("data/tbl2ndCountSurv.csv")
tbl2ndcountDetail<-read.csv("data/tbl2ndCountDetail.csv")
tblIvSurv<-read.csv("data/tblIvSurv.csv")
tblIvDetail<-read.csv("data/tblIvDetail.csv")
```

```r
tblSpecies<-read.csv("data/tblSpecies.csv")
tblCatch<-read.csv("data/tblCatch.csv")
tblMethod<-read.csv("data/tblMethod.csv")
```

- tblcountSurv: contains relevant survey information, including SurveyDate, GeoLocID, Start, and Finish for survey first pass.
- tblGeoLoc: a look-up table linking GeoLocID numbers to relevant river sections.
- tblcountDetail: contains survey counts of boat and shore anglers for each surveys first pass.
- tbl2ndcountSurv: contains relevant survey information for second survey counts.
- tbl2ndcountDetail: contains survey counts of boat and shore anglers for each surveys second pass.
- tblIvSurv: contains relevant roving interview survey information.
- tblIvDetail: contains each recorded interview information for roving interview surveys.
- tblSpecies: species look-up table linking SpeciesID numbers to relevant taxa information.
- tblCatch: tracks information of species caught and kept recorded during roving interview surveys.
- tblMethod: methods look-up table linking MethodID to MethodName.

Prepare first pass survey counts

For our first step, we'll prep the survey count data for the first survey passes. Our work is going to be targeted at a specific year, so we can start by setting our opening and closing dates for our data set to make sure we are only pulling the data for the target year:

```r
#set open and close dates
open_date<-"2022-07-16"
close_date<-"2022-12-31"
```

Once we've set our range of dates that we want to pull the data for, our following piece of code does a lot of work for us:

1. Joins the tblcountSurv table to the tblcountDetail table by the countSurvId field.
2. Joins the above to the tblGeoLoc table to include river section information.
3. Formats all the SurveyDate and Start/Finish times into POSIXct format so they can be used as dates and times within R.
4. Filter the data to only include information from the target open and close date range.
5. Group the data by each location and SurveyDate combination, and calculating the sum of boat anglers and shore anglers.
6. Assign a day type to each survey, either weekday (WD) or weekend (WE).

```r
first_count<-tblcountSurv %>%
  left_join(tblcountDetail, by = "CountSurvId")%>%
  left_join(tblGeoLoc,by='GeoLocId')%>%
  mutate(SurveyDate = as.Date(SurveyDate, format = "%m/%d/%Y"))%>%
  mutate(Start=as.POSIXct(Start,format="%m/%d/%Y %H:%M:%S"))%>%
  mutate(Start=format(Start,format='%H:%M'))%>%
  mutate(Finish=as.POSIXct(Finish,format="%m/%d/%Y %H:%M:%S"))%>%
  mutate(Finish=format(Finish,format='%H:%M'))%>%
  # Filter the survey dates
  filter(as.Date(SurveyDate) >= as.Date(open_date) & as.Date(SurveyDate) <= as.Date(close_date)) %>%
  # Group by required columns and calculate sum of Anglers
  group_by(GeoLocDE, SurveyDate, Start, Finish, Day_Type = ifelse(weekdays(SurveyDate) %in% c("Saturday
  summarise(B_Anglers = sum(coalesce(B_Anglers, 0)), S_Anglers = sum(coalesce(S_Anglers, 0))) %>%
  # Order the result
  arrange(LocSort)
```

Next we can use the dplyr `select()` call to select and rename just the fields we want to utilize in our resulting first_count table. After that we can use the tidyverse `pivot_longer()` call to transform the table so that we end up with a separate record for each method type, either "B" or "S" for boaters and shore anglers respectively, and the total count for both types.

```r
first_count<-dplyr::select(first_count,section=GeoLocDE,SurveyDate,Start,Finish,Day_Type,B_Anglers,S_An

first_count<-first_count%>%
  pivot_longer(
    cols=B_Anglers:S_Anglers,
    names_to = "Method",
    values_to = "Anglers"
  )
first_count$Method<-ifelse(first_count$Method=='B_Anglers','B','S')
```

There are some caveats with how we report our angler survey data, in that some river sections report effort and harvest estimates separately for boat and shore anglers. These are sections 7.1, 12.1, and 12.0D. section 7.1 already get's tracked as "7.1S" and "7.1B" so we will change the other sections to match that format. Here we change sections 12.0D and 12.1 to reflect their specific method type, so that records for boaters on 12.0D will be tracked as "12.0DB", and those for shore anglers as "12.0DS".

```r
#boating and shore surveys that are tracked separately in expansion lookups for 12.1 and 12.0D
targets<-c('12.0D','12.1')
first_count$section<-ifelse(first_count$section%in%targets,paste(first_count$section,first_count$Method
```

Finally we can group all the information by each SurveyDate and section and sum the angler counts. We then add a column that tracks the month of each survey, and finally save the output as "first_counts.rds" in the outputs folder. Note that a '.rds' file is a common format for saving an R object, and allows us to ensure the formatting doesn't change.

```r
#next group and sum by section x month
first_count<-first_count%>%
  group_by(section,SurveyDate,Start,Finish,Day_Type)%>%
  summarise(Anglers=sum(Anglers))

first_count$month<-month(first_count$SurveyDate,label=T,abbr=F)
```

```
saveRDS(first_count,'outputs/first_counts.rds')
```

Prepare second pass survey counts

Next we can do everything we did above but now for the second survey pass. We'll run that all as one chunk below, since the steps are the same as what we did for the first count.

```
second_count<-tblGeoLoc %>%
  right_join(tbl2ndcountSurv %>%
               left_join(tbl2ndcountDetail, by = "CountSurvId"),
             by = "GeoLocId") %>%
  mutate(SurveyDate = as.Date(SurveyDate, format = "%m/%d/%Y"))%>%
  mutate(Start=as.POSIXct(Start,format="%m/%d/%Y %H:%M:%S"))%>%
  mutate(Start=format(Start,format='%H:%M'))%>%
  mutate(Finish=as.POSIXct(Finish,format="%m/%d/%Y %H:%M:%S"))%>%
  mutate(Finish=format(Finish,format='%H:%M'))%>%
  # Filter the survey dates
  filter(as.Date(SurveyDate) >= as.Date(open_date) & as.Date(SurveyDate) <= as.Date(close_date)) %>%
  # Group by required columns and calculate sum of Anglers
  group_by(GeoLocDE, SurveyDate, Start, Finish, Day_Type = ifelse(weekdays(SurveyDate) %in% c("Saturday
  summarise(B_Anglers = sum(coalesce(B_Anglers, 0)), S_Anglers = sum(coalesce(S_Anglers, 0))) %>%
  # Order the result
  arrange(LocSort)

second_count<-dplyr::select(second_count,section=GeoLocDE,SurveyDate,Start,Finish,Day_Type,B_Anglers,S_

second_count<-second_count%>%
  pivot_longer(
    cols=B_Anglers:S_Anglers,
    names_to = "Method",
    values_to = "Anglers"
  )
second_count$Method<-ifelse(second_count$Method=='B_Anglers','B','S')

#boating and shore surveys that are tracked separately in expansion lookups for 12.1 and 12.0D
targets<-c('12.0D','12.1')
second_count$section<-ifelse(second_count$section%in%targets,paste(second_count$section,second_count$Me

second_count<-second_count%>%
  group_by(section,SurveyDate,Start,Finish,Day_Type)%>%
  summarise(Anglers=sum(Anglers))

second_count$month<-month(second_count$SurveyDate,label=T,abbr=F)

saveRDS(second_count,'outputs/second_counts.rds')
```

Prepare roving data

The roving interview data also needs to be cleaned and formatted similar to how the first and second count tables were prepared. The next chunk of code does the following:

1. Joins the tblIvDetail table to the tblCatch table.

2. Joins the above to the tblGeoLoc table to include river section information, and to the tblSecies look-up table.
3. Formats all the SurveyDate and Start/Finish times into POSIXct format so they can be used as dates and times within R.
4. Filter the data to only include information from the target open and close date range.
5. Assign a day type to each survey, either weekday (WD) or weekend (WE).

```
roving<-select(tblIvDetail,IvDetId,IvSurvId,IvPage,IvLine,NumberAnglers,FST,HrsFished,MethodId,tgtSpeci
  left_join(select(tblCatch,CatchId,IvDetId,Kept,Released,SpeciesId))%>%
  left_join(select(tblSpecies,SpeciesId,SpeciesDE),by="SpeciesId")%>%
  left_join(select(tblIvSurv,IvSurvId,GeoLocId,SurveyDate),by="IvSurvId")%>%
  left_join(select(tblGeoLoc,GeoLocId,GeoLocDE))%>%
  left_join(select(tblMethod,MethodId,MethodName))%>%
  mutate(SurveyDate = as.Date(SurveyDate, format = "%m/%d/%Y"))%>%
  mutate(FST=as.POSIXct(FST,format="%m/%d/%Y %H:%M:%S"))%>%
  mutate(FST=format(FST,format='%H:%M'))%>%
  filter(as.Date(SurveyDate) >= as.Date(open_date) & as.Date(SurveyDate) <= as.Date(close_date))
```

Next we can assign week day types to our data, and calculate the total hours fished by anglers:

```
roving$Day_Type<-ifelse(weekdays(roving$SurveyDate)%in% c("Saturday", "Sunday"), "WE", "WD")
roving$TotalHours<-roving$NumberAnglers*roving$HrsFished
```

Next we can select the fields we want to use from our roving table, and then we want to change all the roving data from section "BF" to represent section "7.1B".

```
roving<-dplyr::select(roving,section=GeoLocDE,SurveyDate,Day_Type,Page=IvPage,IvLine,Anglers=NumberAngl

roving$section<-ifelse(roving$section=='BF','7.1B',roving$section)
```

Finally, we can pull out all of the by-catch count data that we don't want in our roving data, and then save our final roving table. The by-catch data will be used later in this script.

```
#Pull out by-catch to remove from roving and use later
bycatch_targets<-c(1,11)
by_catch<-roving%>%
  filter(!tgtSpecies%in%bycatch_targets & SpeciesCaught=='CS')

by_catch_filter<-unique(select(by_catch,section,SurveyDate,Day_Type,Page,IvLine,SpeciesCaught))

#roving<-roving%>%anti_join(by_catch_filter)
saveRDS(roving,'outputs/roving.rds')
```

Prepare ratio data

Next we want to prepare the ratio data, or the ratio of angler effort spent on Chinook. We do this by first pulling out the relevant fields from the roving table we just made,

```
ratio_prep<-unique(select(roving,section,SurveyDate,Day_Type,IvLine,Page,TotalHours,MethodName,MethodNa
```

Next we need to format the sections for 12.0D and 12.1 so that they are divided into “B” and “S” representing boating and shore anglers, so they can be reported separately.

```
target_sections<-c('12.0D','12.1')
target_method<-c('Boat','Guided boat party')

ratio_prep<-ratio_prep%>%
  mutate(MethodName=ifelse(MethodName%in%target_method,'B','S'))
ratio_prep$section<-ifelse(ratio_prep$section %in%target_sections,paste(ratio_prep$section,ratio_prep$M
ratio_prep$section<-ifelse(ratio_prep$section=='BF','7.1B',ratio_prep$section)
```

For the ratio data we next calculate the total effort of all anglers, then the proportion of that effort targeting Chinook. Finally we can save the ratio table for later use.

```
Total_effort<-ratio_prep%>%
  group_by(section,SurveyDate,Day_Type)%>%
  summarise(Total_Effort=sum(TotalHours))
CS_effort<-ratio_prep%>%
  filter(tgtSpecies%in%c(11))%>%
  group_by(section,SurveyDate,Day_Type)%>%
  summarise(CS_Effort=sum(TotalHours))
Ratio<-Total_effort%>%
  left_join(CS_effort)

Ratio$ratio_cs<-ifelse(is.na(Ratio$CS_Effort)==T,0,Ratio$CS_Effort/Ratio$Total_Effort)

saveRDS(Ratio,'outputs/ratio.rds')
```

Prepare by-catch data

The final data set we have to prepare before calculations is the by-catch, which we pulled out of the roving data earlier. This will be used later when we calculate effort and harvest estimates. First we select only the records that don’t have Chinook Salmon, listed as species code 11.

```
by_catch<-select(by_catch,section,SurveyDate,Day_Type,Page,IvLine,SpeciesCaught,Kept,Released,TotalHour
by_catch_surveys<-unique(select(by_catch,section,SurveyDate))
by_catch_effort<-roving%>%
  inner_join(by_catch_surveys)%>%
  filter(tgtSpecies!=11)
```

Next we plug in some empty ‘Kept’ and ‘Released’ zeros so that we can join the by-catch data to the by-catch effort, and save the output as another rds file.

```
by_catch_effort<-unique(select(by_catch_effort,SurveyDate,section,Day_Type,Page,IvLine,TotalHours,Metho
by_catch_effort$SpeciesCaught=''
by_catch_effort$Kept=0
by_catch_effort$Released=0

by_catch<-by_catch%>%rbind(by_catch_effort)
```

```r
saveRDS(by_catch,'outputs/by_catch.rds')
```

Before we move on, let's review the tables we created so far. We can use the `str()` call to see the names of each column in the table, as well as the first few values for each of column. First we can examine our two counts tables, first_counts and second_counts, which are structured the same:

```r
str(data.frame(first_count))
```

```
## 'data.frame':    1272 obs. of  7 variables:
##  $ section   : chr  "1" "1" "1" "1" ...
##  $ SurveyDate: Date, format: "2022-07-17" "2022-07-19" ...
##  $ Start     : chr  "08:25" "08:25" "10:31" "09:24" ...
##  $ Finish    : chr  "09:40" "09:46" "11:50" "10:44" ...
##  $ Day_Type  : chr  "WE" "WD" "WD" "WE" ...
##  $ Anglers   : num  60 8 34 54 124 69 99 48 43 153 ...
##  $ month     : Ord.factor w/ 12 levels "January"<"February"<..: 7 7 7 7 9 9 9 9 9 9 ...
```

Next we can examine the roving data we prepared:

```r
str(data.frame(roving))
```

```
## 'data.frame':    8700 obs. of  14 variables:
##  $ section      : chr  "1.2" "1.2" "1.2" "4" ...
##  $ SurveyDate   : Date, format: "2022-07-19" "2022-07-19" ...
##  $ Day_Type     : chr  "WD" "WD" "WD" "WE" ...
##  $ Page         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ IvLine       : int  1 2 3 1 2 2 3 4 5 6 ...
##  $ Anglers      : int  1 1 3 2 3 3 2 2 3 1 ...
##  $ FST          : chr  "09:30" "09:45" "07:30" "10:30" ...
##  $ HoursFished  : num  0.75 0.5 2.75 2 2.25 2.25 0.25 4.75 3.5 1 ...
##  $ tgtSpecies   : int  1 1 13 13 13 13 13 1 11 13 ...
##  $ SpeciesCaught: chr  "SHRK" NA "SB" "SB" ...
##  $ Kept         : int  0 NA 1 0 1 0 NA NA NA NA ...
##  $ Released     : int  1 NA 0 2 0 1 NA NA NA NA ...
##  $ TotalHours   : num  0.75 0.5 8.25 4 6.75 6.75 0.5 9.5 10.5 1 ...
##  $ MethodName   : chr  "Shore" "Shore" "Shore" "Shore" ...
```

The ratio data:

```r
str(data.frame(Ratio))
```

```
## 'data.frame':    982 obs. of  6 variables:
##  $ section     : chr  "1" "1" "1" "1" ...
##  $ SurveyDate  : Date, format: "2022-07-17" "2022-07-19" ...
##  $ Day_Type    : chr  "WE" "WD" "WD" "WE" ...
##  $ Total_Effort: num  108.5 25 87.8 105.2 219.2 ...
##  $ CS_Effort   : num  NA NA NA 1 NA ...
##  $ ratio_cs    : num  0 0 0 0.0095 0 ...
```

And finally the by_catch table:

```
str(data.frame(by_catch))
```

```
## 'data.frame':    50 obs. of  10 variables:
##  $ section      : chr  "10.2" "10" "10" "10" ...
##  $ SurveyDate   : Date, format: "2022-07-17" "2022-10-30" ...
##  $ Day_Type     : chr  "WE" "WE" "WE" "WE" ...
##  $ Page         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ IvLine       : int  1 22 7 22 5 2 1 5 6 15 ...
##  $ SpeciesCaught: chr  "CS" "CS" "CS" "CS" ...
##  $ Kept         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Released     : num  1 1 1 1 2 1 0 0 0 0 ...
##  $ TotalHours   : num  4.5 0.5 6 1.5 4 8 4.5 0.75 0.75 0.5 ...
##  $ MethodName   : chr  "Shore" "Shore" "Boat" "Shore" ...
```

Now that all of our data is prepared. The next step involves utilizing the first and second count tables we created and expanding their efforts using the EDM effort expansion factors.

Effort Expansions

CVAS surveyors can't be everywhere to interview all the anglers, so we can rely instead on a series of expansion factors to take the survey results and extrapolate an estimate of effort for each river section and month.

Load Data

To do this we start by loading the first_count and second_count tables we created in the data preparation section, as well as the expansion look-up table and the edm look-up table. The 'edm_lookup' table provides an EDM code for each month and section combination, which can then be used, along with survey hours, to reference the 'exp_lookup' table to expand effort.

```
first_counts<-data.frame(readRDS('outputs/first_counts.rds'))
second_counts<-readRDS('outputs/second_counts.rds')

exp_lookup<-read.csv('data/expansion_lookup.csv')
exp_lookup$WE<-as.numeric(exp_lookup$WE)
exp_lookup$WD<-as.numeric(exp_lookup$WD)
exp_lookup$both<-as.numeric(exp_lookup$both)

edm_lookup<-read.csv('data/auto_edm_lookup.csv')
```

Create effort averaging function

The next step is to calculate the average effort for the target month, section, and hours combination using the exp_lookup table. This is slightly complicated, so we are going to first create a function that takes a row of data from our counts data, joins it to the exp_lookup table, and calculates the mean expansion factor for that hour range. We will call this `effort_average()`.

```
#create a function to calculate the average effort for the target surveys month, site-group, and hours
effort_average<-function(row){
  joined_data<-exp_lookup%>%
    filter(month==row$month & site_grouping==row$EDM_code & hours %in% c(row$start_hour:row$finish_hour
  exp_factor<-if(row$Day_Type=="WD"){
    mean(joined_data$WD)/100
  }else if(row$Day_Type=="WE"){
    mean(joined_data$WE)/100
  }else if(row$Day_Type=='both'){
    mean(joined_data$both)/100
  }
}
```

We will be utilizing this function with both the first_counts and second_counts data sets. Before we do that though, we need to format some of the data for use with the function.

First counts preparation for expansion

To prep for the expansion process, we will first join our first_counts data to the edm_lookup table, then create two new columns called 'start_hour' and 'finish_hour', which are just the numeric value of the hour from the 'Start' and 'Finish' survey times. We also add in a 'month' column pulled from the SurveyDate.

```
first_counts<-first_counts%>%left_join(edm_lookup)
first_counts$start_hour<-as.numeric(format(as.POSIXct(first_counts$Start,format="%H:%M"),format='%H'))
first_counts$finish_hour<-as.numeric(format(as.POSIXct(first_counts$Finish,format="%H:%M"),format='%H')
first_counts$month<-format(as.Date(first_counts$SurveyDate, format="%Y-%m-%d"),"%B")
```

Then we can create a new data frame 'first_counts_grouped' that groups all the first_counts records by month, SurveyDate, start_hour, finish_hour, section, Day_Type, and EDM_code. We then calculate the total number of Anglers for those groupings as 'Total_Anglers'.

```
first_counts_grouped<-first_counts%>%
  group_by(month,SurveyDate,start_hour,finish_hour,Day_Type,EDM_code,section)%>%
  dplyr::summarise(Total_Anglers=sum(Anglers))
```

First counts effort expansion

Now we can actually use the `sapply()` function from the tidyverse to apply the `effort_average()` function we just made to each row of the first_counts_grouped data frame, to give us the expansion factor for that row. With longer processes like this it's always good to measure how long it takes using the `Sys.time()` call. For the 2022 first_counts_grouped data this took just over four minutes to run. Note that in the markdown document, this chunk has been set to `eval=F` to save time in document preperation.

```
start<-Sys.time()
first_counts_grouped$expansion_factor<-sapply(1:nrow(first_counts_grouped),function(i){
  effort_average(first_counts_grouped[i,])
})
finish<-Sys.time()
finish-start
saveRDS(first_counts_grouped,'outputs/first_counts_grouped.rds')
```

Now we can use the total number of anglers and divide it by the expansion factors we just calculated for each day and section to produce an estimate of total angler effort for that section and day.

```r
#Calculate total angler effort
first_counts_grouped<-data.frame(readRDS('outputs/first_counts_grouped.rds'))
first_expansions<-first_counts_grouped%>%
  mutate(angler_effort=Total_Anglers/expansion_factor)
first_expansions$count<-'First'
```

Second counts expansion

Next we can replicate the expansion calculations we just did above for the first_count data, but now for the second_count data. The methods are the same, so I'll just throw it all into the following code chunk:

```r
second_counts<-second_counts%>%left_join(edm_lookup)
second_counts$start_hour<-as.numeric(format(as.POSIXct(second_counts$Start,format="%H:%M"),format='%H')
second_counts$finish_hour<-as.numeric(format(as.POSIXct(second_counts$Finish,format="%H:%M"),format='%H
second_counts$month<-format(as.Date(second_counts$SurveyDate, format="%Y-%m-%d"),"%B")

#group_by edm site grouping, month and date and sum anglers
second_counts_grouped<-second_counts%>%
  group_by(month,SurveyDate,start_hour,finish_hour,Day_Type,EDM_code,section)%>%
  dplyr::summarise(Total_Anglers=sum(Anglers))

start<-Sys.time()
second_counts_grouped$expansion_factor<-sapply(1:nrow(second_counts_grouped),function(i){
  effort_average(second_counts_grouped[i,])
})
finish<-Sys.time()
finish-start

#Calculate total angler effort
second_expansions<-second_counts_grouped%>%
  mutate(angler_effort=Total_Anglers/expansion_factor)
second_expansions$count<-'Second'

saveRDS(second_expansions,'outputs/second_expansions.rds')
```

Now that our expansions are done for the second count, we can bind the expansions together into one data set and see what our final output looks like.

```r
second_expansions<-readRDS('outputs/second_expansions.rds')
count_expansions<-first_expansions%>%
  rbind(second_expansions)
str(count_expansions)
```

```
## 'data.frame':    2224 obs. of  11 variables:
##  $ month          : chr  "August" "August" "August" "August" ...
##  $ SurveyDate     : Date, format: "2022-08-01" "2022-08-01" ...
##  $ start_hour     : num  8 9 9 9 14 18 8 11 13 8 ...
##  $ finish_hour    : num  9 9 9 9 15 18 10 11 14 8 ...
##  $ Day_Type       : chr  "WD" "WD" "WD" "WD" ...
##  $ EDM_code       : chr  "2EDM" "10EDM" "2EDM" "2EDM" ...
##  $ section        : chr  "7" "7.1S" "7.1B" "8.5" ...
##  $ Total_Anglers  : num  110 12 117 7 50 25 5 1 1 2 ...
##  $ expansion_factor: num  0.1181 0.0762 0.1149 0.1149 0.059 ...
##  $ angler_effort   : num  931 157.4 1018.5 60.9 848 ...
##  $ count           : chr  "First" "First" "First" "First" ...
```

Finally we'll save out output as 'count_expansions.rds' for use in the next step, summarizing effort and harvest data.

```
saveRDS(count_expansions,'outputs/count_expansions.rds')
```

Calculating Monthly Summaries

This next session has several parts, but the end result is a data set 'summary_data', which contains effort and harvest estimats for each survey section, month, and survey count type ('First' or 'Multiple'). This is done by running the 'summary.R' script, which is walked through below.

We begin by loading the count_expansions, ratio, roving, and by_catch data we've already prepared, and then set the tgt_species value to a list of the numbers 11 and 15, to denote our target species codes.

```
count_expansions<-readRDS('outputs/count_expansions.rds')
ratio<-readRDS('outputs/ratio.rds')
roving<-readRDS('outputs/roving.rds')
by_catch<-readRDS('outputs/by_catch.rds')

tgt_species<-c(11) #here
```

Calculating number of each day type for each month

For use throughout our summary data calculation we will need a data set that contains the number of each day type (weekend vs weekdays) in each month of our survey period. The following chunk of code does that calculation for us based on our open and close dates, set in the first two lines, and gives us the output 'survey_daytypes'.

```
open_date<-as.Date("2022-07-16")
close_date<-as.Date("2022-12-31")

sum(!weekdays(seq(open_date, close_date, "days")) %in% c("Saturday", "Sunday"))
```

```
## [1] 120
```

```
survey_days<-data.frame(month=months(seq(open_date, close_date, "days")),day=seq(open_date, close_date,
survey_days$Day_Type<-ifelse(!weekdays(survey_days$day) %in% c("Saturday", "Sunday"),'WD','WE')
survey_daytypes<-survey_days%>%
  group_by(month,Day_Type)%>%
  dplyr::summarise(day_tally=n())

head(survey_daytypes)
```

```
## # A tibble: 6 x 3
## # Groups:   month [3]
##   month    Day_Type day_tally
##   <chr>    <chr>        <int>
## 1 August   WD               23
## 2 August   WE                8
## 3 December WD               22
## 4 December WE                9
## 5 July     WD               10
## 6 July     WE                6
```

Estimating Chinook Salmon effort

The next step is to use our expansions factors and estimate the total amount of angler effort targeting Chinook Salmon for each month, section, and day type, and count type.

First we join our count_expansions table to the ratio data to create the 'avg_dt_effort' table.

```
avg_dt_effort<-count_expansions%>%
  left_join(select(ratio,-Day_Type))
```

Next we need to deal with sections that have opening days, which are tracked separately because of the high disproportionate effort hey experience. Sections with opening days are 5, 6, and 7.1B and 7.1S. Here the code runner will also have to set the opening dates manually each year in the 'opening_dates' value. For sections with opening dates, we change their section code by pasting 'opener' to the end of it. We then rejoin everything back to the avg_dt_efforts, and now have separate records marked for sections on their opening dates.

```
#deal with opening days
opener_sections<-c('5','6','7.1B','7.1S')
opening_dates<-c(as.Date('2022-07-16'),as.Date('2022-07-16'),as.Date('2022-08-01'),as.Date('2022-08-01'
opening_df<-data.frame(section=opener_sections,SurveyDate=opening_dates)

openers_efforts<-avg_dt_effort%>%
  inner_join(opening_df)
openers_efforts$section<-paste(openers_efforts$section,'opener')

avg_dt_effort<-avg_dt_effort%>%
  anti_join(opening_df)
avg_dt_effort<-avg_dt_effort%>%rbind(openers_efforts)

avg_dt_effort<-avg_dt_effort%>%
  left_join(survey_daytypes)
```

Again, having separate opening days that need to be reported ads some complications, so next we need to change the 'day_tally' field we calculated earlier for sections with opening days, which is done below.

```
#fix opening day tallys
avg_dt_effort$day_tally<-ifelse(avg_dt_effort$section%in%c('5','6') & avg_dt_effort$month=='July' & avg_

avg_dt_effort$day_tally<-ifelse(avg_dt_effort$section%in%c('7.1S','7.1B') & avg_dt_effort$month=='Augus

avg_dt_effort$day_tally<-ifelse(avg_dt_effort$section%in%c('7.1S opener','7.1B opener','5 opener','6 op
```

Our ending data frame here is again 'avg_dt_effort'. Each record here represents each survey, and includes the total number of anglers, the expansions factor, angler effort, total effort, total effort targeting Chinook, the ratio of effort targeting Chinook, and the tally of day types in that monthly survey period for that section.

```r
str(avg_dt_effort)
```

```
## 'data.frame':    2224 obs. of  15 variables:
##  $ month           : chr  "August" "August" "August" "August" ...
##  $ SurveyDate      : Date, format: "2022-08-01" "2022-08-01" ...
##  $ start_hour      : num  8 9 14 18 8 11 13 8 8 8 ...
##  $ finish_hour     : num  9 9 15 18 10 11 14 8 8 8 ...
##  $ Day_Type        : chr  "WD" "WD" "WD" "WD" ...
##  $ EDM_code        : chr  "2EDM" "2EDM" "1EDM" "1EDM" ...
##  $ section         : chr  "7" "8.5" "3" "9" ...
##  $ Total_Anglers   : num  110 7 50 25 5 1 1 2 11 1 ...
##  $ expansion_factor: num  0.1181 0.1149 0.059 0.0597 0.1144 ...
##  $ angler_effort   : num  931 60.9 848 419 43.7 ...
##  $ count           : chr  "First" "First" "First" "First" ...
##  $ Total_Effort    : num  355.5 22.2 98.5 43.8 28.2 ...
##  $ CS_Effort       : num  355.5 22.25 71.25 33.75 0.25 ...
##  $ ratio_cs        : num  1 1 0.72335 0.77143 0.00885 ...
##  $ day_tally       : num  23 23 23 23 23 23 23 23 23 23 ...
```

Next we can take the avg_dt_effort data frame and calculate the total amount of effort targeting Chinook for each section, month, and survey count type ('first' vs 'second'). We need to do it twice, once for the first counts, and then once for multiple counts, which needs to include all records for both 'first' and 'second' type counts.

We'll start by filtering the avg_dt_effort data frame for 'first' counts, then calculate the effort targeting Chinook. Then we group by month, section, and day type to calculate the average Chinook effort.

```r
#first_count
first_effort<-avg_dt_effort%>%
  filter(count=='First')
first_effort$cs_effort<-first_effort$ratio_cs*first_effort$angler_effort
first_effort$cs_effort[is.na(first_effort$cs_effort)]<-0

first_effort<-first_effort%>%
  group_by(month,section,Day_Type,count,day_tally)%>%
  summarise(avg_cs_effort=mean(cs_effort))

first_effort$effort_expanded<-first_effort$avg_cs_effort*first_effort$day_tally
```

Next we calculate the 'multiple' count effort targeting Chinook, by including survey data from both the first and second counts.

```r
#multi_count
multi_effort<-avg_dt_effort
multi_effort$cs_effort<-multi_effort$ratio_cs*multi_effort$angler_effort
multi_effort$cs_effort[is.na(multi_effort$cs_effort)]<-0

multi_effort<-multi_effort%>%
  group_by(month,SurveyDate,section,Day_Type,day_tally)%>%
  summarise(avg_cs_effort=mean(cs_effort))

multi_effort<-multi_effort%>%
  group_by(month,section,Day_Type,day_tally)%>%
  summarise(avg_cs_effort=mean(avg_cs_effort))

multi_effort$effort_expanded<-multi_effort$avg_cs_effort*multi_effort$day_tally
multi_effort$count<-'Multiple'
```

Finally we join the multi_effort and first_effort data sets together as an 'expanded efforts' data set containing the average effort targeting Chinook for each section, count type, day type, and month.

```
expanded_efforts<-first_effort%>%rbind(multi_effort)
str(data.frame(expanded_efforts))
```

```
## 'data.frame':    732 obs. of  7 variables:
##  $ month          : chr  "August" "August" "August" "August" ...
##  $ section        : chr  "1.2" "1.2" "10" "10" ...
##  $ Day_Type       : chr  "WD" "WE" "WD" "WE" ...
##  $ count          : chr  "First" "First" "First" "First" ...
##  $ day_tally      : num  23 8 23 8 23 8 23 8 23 8 ...
##  $ avg_cs_effort  : num  26.8 108.81 20.62 11.69 9.69 ...
##  $ effort_expanded: num  616.3 870.5 474.2 93.5 222.9 ...
```

Before we stop looking at effort, we also need two more data sets for our estimates, the average effort targeting Chinook for both count types, averaged for daily and monthly summaries.

```
daily_effort<-avg_dt_effort
daily_effort$CS_angler_effort<-daily_effort$angler_effort*daily_effort$ratio_cs
daily_effort<-daily_effort%>%
  group_by(SurveyDate,month,section,Day_Type)%>%
  summarise(multi_effort_average=mean(CS_angler_effort))
monthly_effort<-daily_effort%>%
  group_by(month,section)%>%
  summarise(sum_hours=sum(multi_effort_average))
```

Now that we have estimates of the amount of angler effort targeting Chinook Salmon for each reach and day, our next step is to utilize the roving data we created to estimate the harvest of Chinook.

Estimating Chinook Salmon harvest

To start with our roving data, we have to make sure we deal with separating the shore and boat angler data for sections 12.0D and 12.1. We do this in the following code chunk by changing the 'MethodName' field to either 'B' or 'S' and then pasting it to the section name for the target sections.

```
target_sections<-c('12.0D','12.1')
target_method<-c('Boat','Guided boat party')

roving_prep<-roving%>%
  mutate(MethodName=ifelse(MethodName%in%target_method,'B','S'))
roving_prep$section<-ifelse(roving_prep$section %in%target_sections,paste(roving_prep$section,roving_pr
```

Next we can create two new data sets: 'hours_targeting_CS' which sums the angler effort targeting Chinook, and CS_captured, which provides the sum of Chinook captured for each survey.

```
hours_targeting_CS<-unique(select(roving_prep,section,SurveyDate,Day_Type,tgtSpecies,Page,IvLine,TotalHo
  filter(tgtSpecies%in%tgt_species)%>%
  group_by(section,SurveyDate,Day_Type)%>%
  summarise(TotalHours=sum(TotalHours,na.rm=T))

CS_captured<-unique(select(roving_prep,section,SurveyDate,Day_Type,tgtSpecies,SpeciesCaught,IvLine,Kept
  filter(SpeciesCaught=='CS', tgtSpecies==11)%>%
  group_by(section,SurveyDate,Day_Type)%>%
  summarise(Kept=sum(Kept),Released=sum(Released))
```

We can now estimate the daily Catch-per-Unit Effort (CPUE) for each survey that was conducted. We estimate catch CPUE, harvest CPUE, and released CPUE in the 'daily_CPUE' data set.

```
daily_cpue<-hours_targeting_CS%>%left_join(CS_captured)
daily_cpue$catch_CPUE<-(daily_cpue$Kept+daily_cpue$Released)/daily_cpue$TotalHours
daily_cpue$harvest_CPUE<-(daily_cpue$Kept)/daily_cpue$TotalHours
daily_cpue$released_CPUE<-(daily_cpue$Released)/daily_cpue$TotalHours

daily_cpue[is.na(daily_cpue)] <- 0

daily_cpue<-unique(select(ungroup(count_expansions),SurveyDate,section,Day_Type))%>%
  left_join(daily_cpue)
daily_cpue[is.na(daily_cpue)] <- 0
```

Before we continue, we also have to fix estimates for sections with opening days. This is done by creating the opening_df data set, which lists the sections and their opening days. Note that this must be updated each year when applicable. This will allow us to have different opener sections for those specific surveys, incorporating their specific effort distributions.

```
opener_sections<-c('5','6','7.1B','7.1S')
opening_dates<-c(as.Date('2022-07-16'),as.Date('2022-07-16'),as.Date('2022-08-01'),as.Date('2022-08-01'
opening_df<-data.frame(section=opener_sections,SurveyDate=opening_dates)

openers_cpue<-daily_cpue%>%
  inner_join(opening_df)
openers_cpue$section<-paste(openers_cpue$section,'opener')

daily_cpue<-daily_cpue%>%
  anti_join(opening_df)
daily_cpue<-daily_cpue%>%rbind(openers_cpue)
```

Finally, we can use our daily_cpue data set to calculate the average cpue for each month, section, and day type. We can then join the 'avg_cpue' data set to the 'expanded_efforts', and estimate the total catch, harvest, and release for each day type.

```
avg_cpue<-daily_cpue%>%
  group_by(month=month(SurveyDate,label=T,abbr=F),section,Day_Type)%>%
  summarise(avg_catch_cpue=mean(catch_CPUE),avg_harvest_cpue=mean(harvest_CPUE),avg_released_cpue=mean(

avg_cpue<-avg_cpue%>%
  left_join(expanded_efforts)
```

```
total_catch<-avg_cpue
total_catch$Total_DT_Catch<-total_catch$avg_cs_effort*total_catch$avg_catch_cpue*total_catch$day_tally

total_catch$Total_DT_Harvest<-total_catch$avg_cs_effort*total_catch$avg_harvest_cpue*total_catch$day_ta

total_catch$Total_DT_Released<-total_catch$avg_cs_effort*total_catch$avg_released_cpue*total_catch$day_
```

Our final data set, 'total_catch', contains the average cpue for catch, harvest, and release, as well as the total estimates of catch, harvest, and release for each month, section, count type, and day type.

```
str(data.frame(total_catch))
```

```
## 'data.frame':    732 obs. of  13 variables:
##  $ month            : chr  "July" "July" "July" "July" ...
##  $ section          : chr  "1" "1" "1" "1" ...
##  $ Day_Type         : chr  "WD" "WD" "WE" "WE" ...
##  $ avg_catch_cpue   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ avg_harvest_cpue : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ avg_released_cpue: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ count            : chr  "First" "Multiple" "First" "Multiple" ...
##  $ day_tally        : num  10 10 6 6 10 10 6 6 10 10 ...
##  $ avg_cs_effort    : num  0 0 2.21 3.11 0 ...
##  $ effort_expanded  : num  0 0 13.3 18.7 0 ...
##  $ Total_DT_Catch   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Total_DT_Harvest : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Total_DT_Released: num  0 0 0 0 0 0 0 0 0 0 ...
```

Expanding by catch

For this next section, we can focus on the by-catch data we prepared earlier, to estimate the number of Chinook captured, harvested, and released for anglers not targeting them, also known as 'by-catch'.

We start by calculating the total hours of effort and the number of kept and released Chinook for each day and section. Then, similar to how we did above, we calculate CPUE values for captured, kept, and released Chinook.

```
bycatch_expanded<-by_catch%>%
  group_by(SurveyDate,section,Day_Type)%>%
  summarise(TotalHours=sum(TotalHours),Kept=sum(Kept),Released=sum(Released))

bycatch_expanded$catch_CPUE<-(bycatch_expanded$Kept+bycatch_expanded$Released)/bycatch_expanded$TotalHo
bycatch_expanded$harvest_CPUE<-(bycatch_expanded$Kept)/bycatch_expanded$TotalHours
bycatch_expanded$released_CPUE<-(bycatch_expanded$Released)/bycatch_expanded$TotalHours

bycatch_expanded$month<-month(bycatch_expanded$SurveyDate,label = T,abbr = F)
```

Next we want to create an 'empty_bycatch' data set, which takes the roving data and inputs zeros for catch, released, and kept values. This data set will represent the effort and lack of Chinook catch for by-catch data, and is then joined to the expanded by-catch data we just created.

```r
#create zeros for all other date sections
empty_bycatch<-unique(select(roving_prep,section,SurveyDate,Day_Type,TotalHours))%>%
  group_by(SurveyDate,section,Day_Type)%>%
  summarise(TotalHours=sum(TotalHours))
empty_bycatch$Kept=0
empty_bycatch$Released=0
empty_bycatch$catch_CPUE=0
empty_bycatch$month<-month(empty_bycatch$SurveyDate,label = T,abbr = F)
empty_bycatch$harvest_CPUE<-0
empty_bycatch$released_CPUE<-0

empty_bycatch<-empty_bycatch%>%
  anti_join(bycatch_expanded,by=c('SurveyDate','section'))

bycatch_expanded<-bycatch_expanded%>%
  rbind(empty_bycatch)
```

We now want to take the above and calculate the average Chinook by-catch for each section, month, and day type.

```r
avg_bycatch<-bycatch_expanded%>%
  group_by(month,section,Day_Type)%>%
  summarise(avg_DT_catchCPUE=mean(catch_CPUE),
            avg_DT_harvestCPUE=mean(harvest_CPUE),
            avg_DT_releasedCPUE=mean(released_CPUE))
```

Now we can bring in our average day type effort for Chinook, and estimate the total by-catch effort as the effort of angling not targeting Chinook salmon.

```r
#bring in effort data
avg_bycatch<-avg_bycatch%>%
  left_join(select(avg_dt_effort,section,SurveyDate,Day_Type,angler_effort,ratio_cs))

avg_bycatch$ratio_bycatch<-1-avg_bycatch$ratio_cs
avg_bycatch$bycatch_effort<-avg_bycatch$angler_effort*avg_bycatch$ratio_bycatch
```

We can then use the average by-catch data to estimate the total amount of Chinook captured, released, and kept/harvested as by-catch.

```r
#by-catch totals
total_bycatch<-avg_bycatch%>%
  group_by(month,section,Day_Type)%>%
  summarise(avg_DT_catchCPUE=mean(avg_DT_catchCPUE,rm.na=T),
            avg_DT_effort=mean(bycatch_effort,rm.na=T),
            avg_DT_harvestCPUE=mean(avg_DT_harvestCPUE,na.rm=T),
            avg_DT_releasedCPUE=mean(avg_DT_releasedCPUE,na.rm=T))
total_bycatch<-total_bycatch%>%left_join(survey_daytypes)
total_bycatch$total_bycatch_effort<-total_bycatch$avg_DT_effort*total_bycatch$day_tally
```

To wrap up our by-catch data-set, we can estimate the total amount of Chinook caught, harvested, and released for each month, section, and day type. This is then grouped by each month and section to estimate the total Chinook caught for each month and section. Note that the final 'summary_bycatch' data set has a lot of empty rows, as by-catch effort isn't super common.

```r
total_bycatch$avg_DT_catch<-total_bycatch$avg_DT_catchCPUE*total_bycatch$avg_DT_effort
total_bycatch$total_DT_catch<-total_bycatch$avg_DT_catch*total_bycatch$day_tally

total_bycatch$total_harvest<-(total_bycatch$avg_DT_effort*total_bycatch$avg_DT_harvestCPUE)*total_bycat

total_bycatch$total_released<-(total_bycatch$avg_DT_effort*total_bycatch$avg_DT_releasedCPUE)*total_byca

summary_bycatch<-total_bycatch%>%
  group_by(month,section)%>%
  summarise(bycatch_total_catch=sum(total_DT_catch),
            bycatch_total_harvest=sum(total_harvest),
            bycatch_total_released=sum(total_released))
```

Final monthly and section summary

To wrap up this long section, we will take all of the by-catch and total-catch data to create our monthly section summaries showing 'Total_angler_effort', 'TotalCatch_est', 'TotalHarvest_est', 'Total_Released_est', and the 'TotalCatch_recorded'.

```r
summary_data<-total_catch%>%
  group_by(month,section,count)%>%
  summarise(total_angler_effort=sum(effort_expanded),
            total_catch=sum(Total_DT_Catch),
            total_harvest=sum(Total_DT_Harvest),
            total_released=sum(Total_DT_Released))

summary_data<-summary_data%>%left_join(summary_bycatch) #join in by-catch summary

#colnames(summary_data)<-c('month','section','count','total_angler_effort','total_catch','total_harvest
                         # 'total_catch_recorded') #set all column names

write.csv(summary_data,'outputs/summary_data.csv',row.names = F)
```

Our final summary data frame:

```r
str(data.frame(summary_data))
```

```
## 'data.frame':    376 obs. of  10 variables:
##  $ month                 : chr  "August" "August" "August" "August" ...
##  $ section               : chr  "1.2" "1.2" "10" "10" ...
##  $ count                 : chr  "First" "Multiple" "First" "Multiple" ...
##  $ total_angler_effort   : num  1487 1627 568 568 298 ...
##  $ total_catch           : num  0 0 10.4 10.4 0 ...
##  $ total_harvest         : num  0 0 0 0 0 ...
##  $ total_released        : num  0 0 10.4 10.4 0 ...
##  $ bycatch_total_catch   : num  0 0 NA NA NA NA NA NA NA NA ...
##  $ bycatch_total_harvest : num  0 0 NA NA NA NA NA NA NA NA ...
##  $ bycatch_total_released: num  0 0 NA NA NA NA NA NA NA NA ...
```