Influence of instream wood and increases in flood magnitude on Pacific salmon spawning habitat

Tongass National Forest, Alaska

Jon Hart, GIS Spatial Analyst, Wild Salmon Center

3/21/2022

Setup

```
### Attach required packages
library(pacman)
p_load(janitor, tidyverse, readxl, here, data.table)
```

```
### Read in raw data & manipulate data frames

## Data Controls (USER INPUTS REQUIRED)
#---------------------------------------------
# Check home directory
getwd()
```

```
## [1] "C:/Projects/Tongass Logging Exposure/TongassNF Wood Recruitment - R/TongassNF Wood Exposure"
```

```
# Create text string of home directory (paste in WD path from above)
wd.text<-("/Projects/Tongass Logging Exposure/TongassNF Wood Recruitment - R/TongassNF Wood Exposure")

# Set new working directory (paste in path to folder where reach datasets are located)
setwd(paste(wd.text,"/Reaches_Raw", sep=""))
#---------------------------------------------

# Create a matrix of file names, removing the extension
filenames <- gsub(".xlsx","", list.files(pattern=".xlsx$"))

# ============================================================================================================
# Predicted reach average median streambed grain size (D50) = [(phS)^(1-n)]/[(ps-p)kg^n] [Buffington et al., 2004
] where h is bankfull depth, g # is gravitational acceleration, S is channel slope, ps and p are sediment (2650 k
g·m−3·s−1) and fluid densities (1000 # kg·m−3·s−1), respectively. The terms k and n are empirical values relating
bankfull Shields stress (Tbf*) and total # bankfull shear stress (T): Tbf* = kTn. Values for k and n vary by chan
nel type and are given for Pacific Northwest mountain drainages.
# ============================================================================================================

# Assign constant values

k_pb <- 0.0054 # plane-bed k value
k_pr <- 0.013 # pool-riffle k value
n_pb <- 0.6 # plane-bed n value
n_pr <- 0.5 # pool-riffle n value
k_wfpr <- 0.014 # wood-forced pool-riffle k value
n_wfpr <- 0.59 # wood-forced pool-riffle n value
g <- 9.8 # gravitational acceleration
dyn20 <-  0.3552*0.2^0.9649 # dynamic response parameter (+20% flood magnitude)
dyn30 <-  0.3552*0.3^0.9649 # dynamic response parameter (+30% flood magnitude)
regions <- c("Admiralty", "Craig", "Hoonah", "Juneau", "Ketchikan", "Petersburg", "Sitka", "Thorne", "Wrangell",
"Yakutat")

# Create function to calculate pool-riffle d50
PR_d50 = function(n_pr, k_pr, g, h, slope) {
result = 1000*((1000*h*slope)^(1-n_pr))/((2650-
1000)*k_pr*g^n_pr)
return(result)
}

# Create function to calculate plane-bed d50
PB_d50 = function(n_pb, k_pb, g, h, slope) {
result = 1000*((1000*h*slope)^(1-n_pb))/((2650-
1000)*k_pb*g^n_pb)
return(result)
}

# Create function to calculate wood-forced pool-riffle d50
PR_wfr_d50 = function(n_wfpr, k_wfpr, g, h, slope) {
result = 1000*((1000*h*slope)^(1-n_wfpr))/((2650-
1000)*k_wfpr*g^n_wfpr)
return(result)
}

# Run loop to read in each .xlsx region file and calculate area (square meters), flow depth (h), channel width (w
), and channel type for each reach in the region

for(i in filenames){
  assign(i, read_xlsx(paste(i, ".xlsx", sep="")) %>%
    clean_names() %>%
    rename(slope = gradient) %>%
    mutate(area_sqm = area_sqkm * 1000000,
           h = 0.0505*area_sqm^0.1702, # h = 0.0505A^0.1702
           w = 0.0027*area_sqm^0.5129, # w = 0.0027A^0.5129
           channel_type = ifelse(slope < 0.02, "PR","PB"), # If slope <2 degrees then assign reach type as pool-r
iffle. If >2 degrees then assign reach type as plane-bed.
           region = i,
           region_id = paste(region, "_", id, sep=""))) # region_id = region name + reach id
}
```

Intermediate Reach Data

```
### Create an "intermediate" folder and save intermediate data as .csv files

## -------------------------USER INPUT-----------------------------------
folder_name <- "intermediate" # specify the name of the new folder to create
## ----------------------------------------------------------------------

# Create new folder and set as working directory
dir.create(folder_name)
setwd(paste(wd.text,"/",folder_name, sep=""))

# Create lists of data frames
list.dfs<-list()
list.dfs <- lapply(regions, function(i){ # note: regions defined in first code chunk
data <- get(i)
})

# Name the lists
names(list.dfs) <- lapply(regions, function(i){
i
})

# Extract column names
columns <- c(colnames(get(regions[1]))) # All data frames have same column headers so anyone will do

# Run loop, saving each data frame as a .csv in the "intermediate" folder
for (i in 1:length(list.dfs)){
write.csv(as.data.frame(list.dfs[i], col.names = names(columns)), paste0(names(list.dfs[i]), ".csv"))
}
```

Species Rules

Habitat Quality = Slope, Bankfull Width(m), Bankfull depth(m), Confinement, D50(mm)

Coho
high = <2.0, >2.0, >0.5, unconfined (0), 10 - 50
moderate = <4.5, >2.0, >0.5, confined (1), 10 - 50
Chum
high = <2.0, >10.0, >0.5, unconfined (0), 5 - 50
high = <=1.0, >2.0, >0.5, unconfined (0), 5 - 50
moderate = <4.5, >10.0, >0.5, confined (1), 5 - 50
Pink
high = <2.0, >10.0, >0.5, unconfined (0), 5 - 25
high = <=1.0, >2.0, >0.5, unconfined (0), 5 - 25
moderate = <4.5, >10.0, >0.5, confined (1), 5 - 25

Current Scenarios

```
### Create a "current" folder, manipulate data frames, and save d50 calculations under current timeframe as .csv
files

## -------------------------USER INPUT-----------------------------------
timeframe <- "current"
## ----------------------------------------------------------------------

# Create new folder and set as working directory
dir.create(paste(timeframe))
setwd(paste(wd.text,"/", timeframe, sep=""))

# Specify new file/data frame names
new_names <- paste0(names(list.dfs), "_", timeframe, sep = "")

# Run loop, creating data frames with d50 calculations for current scenario and saving each data frame as a .csv
in the "current" folder

for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         mutate(
d50_mm =
ifelse(channel_type == "PR", round(PR_d50(n_pr=n_pr, k_pr
=k_pr, g=g, h=h, slope=slope), 0), round(PB_d50(n_pb=n_pb, k_pb=k_pb, g=g, h=h, slope=slope),0)),

d50_mm_wfr =
ifelse(d50_mm > 50, round (PR_wfr_d50(n_wfpr=n_wfpr, k_wfpr=k_wfpr, g=g, h=h, slope=slope),0), d50_mm)))

# No wood d50: If channel type is PR, calculate d50 using PR function. If not, calculate d50 using PB function.
# Wood d50: If d50 > 50, calculate d50 using wood-forced PR function. If not, write d50 from no wood scenario.

write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
```

Function

```
## create function with setup for additional code chunks
# Assign variables
setup = function(timeframe , condition ) {
assign("timeframe", timeframe, envir = globalenv())
assign("condition", condition, envir = globalenv())
assign("scenario",paste(timeframe, "_", condition, sep=""),envir = globalenv())
assign("list.dfs", list(), envir = globalenv())

# Create new folder and set as working directory
dir.create(paste(scenario))
setwd(paste(wd.text,"/", scenario, sep=""))

# Create lists of data frames
list.dfs <- lapply(regions, function(i){ # note: regions defined in first code chunk
df <- get(paste(i, "_", timeframe, sep=""))
})
.GlobalEnv$list.dfs <- list.dfs

# Name the lists
names(.GlobalEnv$list.dfs) <- lapply(regions, function(i){
paste(i, "_", timeframe, sep="")
})

# Extract column names
columns <- c(colnames(get(paste(regions[1],"_", timeframe, sep = ""))))
.GlobalEnv$columns <-columns

# Specify new file/data frame names
new_names <- paste0(names(.GlobalEnv$list.dfs), "_", condition, sep = "")
.GlobalEnv$new_names <-new_names
}
```

Current No Wood

```
### Create a "current_nowood" folder, manipulate data frames and save as .csv files

## -------------------------USER INPUT-----------------------------------
setup(timeframe = "current", condition = "nowood")
## ----------------------------------------------------------------------

# Run loop, creating data frames specifying habitat quality for each species under current (no wood) scenario and
saving each data frame as a .csv in the "current_nowood" folder. Combine all regions into one dataframe and save.
full.list<-list()
for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, slope, h, w, confinement, d50_mm, length_m) %>%
         mutate(

coho_habitat = ifelse(slope < 2.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm >= 10 & d50_mm <= 50, "high", i
felse(slope < 4.5 & w > 2.0 & h > 0.5 & confinement == 1 & d50_mm >= 10 & d50_mm <= 50, "moderate","NA")),

pink_habitat = ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm >= 5 & d50_mm <= 25, "high", i
felse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm >= 5 & d50_mm <= 25, "high", ifelse(slope < 4.
5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm >= 5 & d50_mm <= 25, "moderate", "NA"))),

chum_habitat = ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm >= 5 & d50_mm <= 50, "high", i
felse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm >= 5 & d50_mm <= 50, "high", ifelse(slope < 4.
5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm >= 5 & d50_mm <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
   }
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Current Wood

```
### Create a "current_wood" folder, manipulate data frames and save as .csv files

## -------------------------USER INPUT-----------------------------------
setup(timeframe = "current", condition = "wood")
## ----------------------------------------------------------------------

# Run loop, creating data frames specifying habitat quality for each species under current (wood) scenario and sa
ving each data frame as a .csv in the "current_wood" folder. Combine all regions into one dataframe and save.
full.list<-list()
for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, slope, h, w, confinement, d50_mm_wfr,length_m) %>%
         mutate(

coho_habitat = ifelse(slope < 2.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr >= 10 & d50_mm_wfr <= 50, "
high", ifelse(slope < 4.5 & w > 2.0 & h > 0.5 & confinement == 1 &  d50_mm_wfr >= 10 & d50_mm_wfr <= 50, "moderat
e","NA")),

pink_habitat = ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_wfr >= 5 & d50_mm_wfr <=25, "h
igh", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr >=5 & d50_mm_wfr <= 25, "high", ife
lse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_wfr >= 5 & d50_mm_wfr <= 25, "moderate", "NA")))
,

chum_habitat = ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_wfr >= 5 & d50_mm_wfr <= 50, "
high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr >=5 & d50_mm_wfr <= 50, "high", if
else(slope < 4.5 & w > 10.0 & h > 0.5 & confinement ==1 & d50_mm_wfr >= 5 & d50_mm_wfr <= 50, "moderate", "NA")))
))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
   }
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Static Flood Scenarios

```
## -------------------------USER INPUT-----------------------------------
timeframe <- "static"
## ----------------------------------------------------------------------

# Create new folder and set as working directory
dir.create(paste(timeframe))
setwd(paste(wd.text,"/", timeframe, sep=""))

# Create lists of data frames
list.dfs<-list()
list.dfs <- lapply(regions, function(i){ # note: regions defined in first code chunk
data <- get(i)
})

# Name the lists
names(list.dfs) <- lapply(regions, function(i){
i
})

# Extract column names
columns <- c(colnames(get(regions[1]))) # All data frames have same column headers so anyone will do

# Specify new file/dataframe names
new_names <- paste0(names(list.dfs), "_", timeframe, sep = "")

# Run loop, creating data frames with d50 calculations for static flood scenarios and saving each data frame as a
.csv in the "static" folder

for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         mutate(
h_static20 = ifelse(confinement == 1, ((1+(0.0043+0.5196*0.2))*h), h),
h_static30 = ifelse(confinement == 1, ((1+(0.0043+0.5196*0.3))*h), h),

d50_mm_static20 =
ifelse(channel_type == "PR", round(PR_d50(n_pr=n_pr, k_pr=k_pr, g=g, h=h_static20, slope=slope), 0), round(PB_d50
(n_pb=n_pb, k_pb=k_pb, g=g, h=h_static20, slope=slope),0)),

d50_mm_static30 =
ifelse(channel_type == "PR", round(PR_d50(n_pr=n_pr, k_pr=k_pr, g=g, h=h_static30, slope=slope), 0), round(PB_d50
(n_pb=n_pb, k_pb=k_pb, g=g, h=h_static30, slope=slope),0)),

d50_mm_wfr_static20 =
ifelse(d50_mm_static20 > 50, round(PR_wfr_d50(n_wfpr=n_wfpr, k_wfpr=k_wfpr, g=g, h=h_static20, slope=slope),0), d
50_mm_static20),

d50_mm_wfr_static30 =
ifelse(d50_mm_static30 > 50, round(PR_wfr_d50(n_wfpr=n_wfpr, k_wfpr=k_wfpr,g=g, h=h_static30, slope=slope),0), d5
0_mm_static30)))

# If channel unconfined, channel flow depth (h) remains the same. If channel confined, flow depth (h) increases p
er equation for given flood magnitude increase

# No wood d50: If channel type is PR, calculate d50 using PR function. If not, calculate d50 using PB function.
# Wood d50: If d50 > 50, calculate d50 using wood-forced PR function. If not, write d50 from no wood scenario.

write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
```

Static20 No Wood

```
### Create a "static20_nowood" folder, manipulate data frames and save as .csv files

## -------------------------USER INPUT-----------------------------------
setup(timeframe = "static", condition = "20_nowood")
## ----------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under static20 (no wood) scenario a
nd saving each data frame as a .csv in the "static20_nowood" folder. Combine all regions into one dataframe and s
ave.
# Under static scenarios, current channel dimensions are used to identify suitable spawning reaches.

full.list<-list()
for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, length_m, slope, h, w, confinement, d50_mm_static20) %>%
         mutate(
coho_habitat =
ifelse(slope < 2.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_static20 >= 10 & d50_mm_static20 <= 50, "high"
, ifelse(slope < 4.5 & w > 2.0 & h > 0.5 & confinement == 1 & d50_mm_static20 >= 10 & d50_mm_static20 <= 50, "mod
erate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_static20 >= 5 &
d50_mm_static20 <= 25, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_static20 >= 5
& d50_mm_static20 <= 25, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_static20 >=
5 & d50_mm_static20 <= 25, "moderate", "NA"))),

chum_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_static20 >= 5 &
d50_mm_static20 <= 50, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_static20 >= 5
& d50_mm_static20 <= 50, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_static20 >=
5 & d50_mm_static20 <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Static20 Wood

```
### Create a "static20_wood" folder, manipulate data frames and save as .csv files

## -------------------------USER INPUT-----------------------------------
setup(timeframe = "static", condition = "20_wood")
## ----------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under static20 (wood) scenario and
saving each data frame as a .csv in the "static20_wood" folder. Combine all regions into one dataframe and save.
# Under static scenarios, current channel dimensions are used to identify suitable spawning reaches.
full.list<-list()

for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, length_m, slope, h, w, confinement, d50_mm_wfr_static20) %>%
         mutate(

coho_habitat =
ifelse(slope < 2.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static20 >= 10 & d50_mm_wfr_static20 <= 50
, "high", ifelse(slope < 4.5 & w > 2.0 & h > 0.5 & confinement == 1 & d50_mm_wfr_static20 >= 10 & d50_mm_wfr_stat
ic20 <= 50, "moderate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static20 >= 5 & d50_mm_wfr_static20 <= 25
, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static20 >= 5 & d50_mm_wfr_stat
ic20 <= 25, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_wfr_static20 >= 5 & d50_m
m_wfr_static20 <= 25, "moderate", "NA"))),

chum_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static20 >= 5 &  d50_mm_wfr_static20 <= 5
0, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static20 >= 5 & d50_mm_wfr_sta
tic20 <= 50, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_wfr_static20 >= 5 & d50_
mm_wfr_static20 <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Static30 No Wood

```
### Create a "static30_nowood" folder, manipulate data frames and save as .csv files

## -------------------------USER INPUT-----------------------------------
setup(timeframe = "static", condition = "30_nowood")
## ----------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under static30 (no wood) scenario a
nd saving each data frame as a .csv in the "static30_nowood" folder. Combine all regions into one dataframe and s
ave.
# Under static scenarios, current channel dimensions are used to identify suitable spawning reaches.
full.list<-list()

for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, length_m, slope, h, w, confinement, d50_mm_static30) %>%
         mutate(
coho_habitat = ifelse(slope < 2.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_static30 >= 10 & d50_mm_static3
0 <= 50, "high", ifelse(slope < 4.5 & w > 2.0 & h > 0.5 & confinement == 1 & d50_mm_static30 >= 10 & d50_mm_stati
c30 <= 50, "moderate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_static30 >= 5 &
d50_mm_static30 <= 25, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_static30 >= 5
& d50_mm_static30 <= 25, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_static30 >=
5 & d50_mm_static30 <= 25, "moderate", "NA"))),

chum_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_static30 >= 5 &
d50_mm_static30 <= 50, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_static30 >= 5
& d50_mm_static30 <= 50, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_static30 >=
5 & d50_mm_static30 <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Static30 Wood

```
### Create a "static30_wood" folder, manipulate data frames and save as .csv files

## -------------------------USER INPUT-----------------------------------
setup(timeframe = "static", condition = "30_wood")
## ----------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under static20 (wood) scenario and
saving each data frame as a .csv in the "static30_wood" folder. Combine all regions into one dataframe and save.
# Under static scenarios, current channel dimensions are used to identify suitable spawning reaches.
full.list<-list()

for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
  select(id, region, region_id, length_m, slope, h, w, confinement, d50_mm_wfr_static30) %>%
  mutate(

coho_habitat =
ifelse(slope < 2.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static30 >= 10 & d50_mm_wfr_static30 <= 50
, "high", ifelse(slope < 4.5 & w > 2.0 & h > 0.5 & confinement == 1 & d50_mm_wfr_static30 >= 10 & d50_mm_wfr_stat
ic30 <= 50, "moderate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static30 >= 5 & d50_mm_wfr_static30 <= 25
, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static30 >= 5 & d50_mm_wfr_stat
ic30 <= 25, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_wfr_static30 >= 5 & d50_m
m_wfr_static30 <= 25, "moderate", "NA"))),

chum_habitat =
ifelse(slope < 2.0 & w > 10.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static30 >= 5 & d50_mm_wfr_static30 <= 50
, "high", ifelse(slope <= 1.0 & w > 2.0 & h > 0.5 & confinement == 0 & d50_mm_wfr_static30 >= 5 & d50_mm_wfr_stat
ic30 <= 50, "high", ifelse(slope < 4.5 & w > 10.0 & h > 0.5 & confinement == 1 & d50_mm_wfr_static30 >= 5 & d50_m
m_wfr_static30 <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Dynamic Scenarios

```
## -------------------------USER INPUT-----------------------------------
timeframe <- "dynamic"
## ----------------------------------------------------------------------

# Create new folder and set as working directory
dir.create(paste(timeframe))
setwd(paste(wd.text,"/", timeframe, sep=""))

# create lists of dataframes
list.dfs<-list()
list.dfs <- lapply(regions, function(i){ # note: regions defined in first code chunk
data <- get(i)
})

# Name the lists
names(list.dfs) <- lapply(regions, function(i){
i
})

# Extract column names
columns <- c(colnames(get(regions[1]))) # All data frames have same column headers so anyone will do

# Specify new file/dataframe names
new_names <- paste0(names(list.dfs), "_", timeframe, sep = "")

## Run loop, creating data frames with d50 calculations for dynamic flood scenarios and saving each data frame as
a .csv in the "dynamic" folder

for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         mutate(h_dyn20 = ((1+dyn20)*h),
                h_dyn30 = ((1+dyn30)*h),
d50_mm_dyn20 =
ifelse(channel_type == "PR", round(PR_d50(n_pr=n_pr, k_pr=k_pr, g=g, h=h_dyn20, slope=slope), 0), round(PB_d50(n_
pb=n_pb, k_pb=k_pb, g=g, h=h_dyn20, slope=slope),0)),

d50_mm_dyn30 =
ifelse(channel_type == "PR", round(PR_d50(n_pr=n_pr, k_pr=k_pr, g=g, h=h_dyn30, slope=slope), 0), round(PB_d50(n_
pb=n_pb, k_pb=k_pb, g=g, h=h_dyn30, slope=slope),0)),

d50_mm_wfr_dyn20 =
ifelse(d50_mm_dyn20 > 50, round(PR_wfr_d50(n_wfpr=n_wfpr, k_wfpr=k_wfpr, g=g, h=h_dyn20, slope=slope),0), d50_mm_
dyn20),

d50_mm_wfr_dyn30 =
ifelse(d50_mm_dyn30 > 50, round(PR_wfr_d50(n_wfpr=n_wfpr, k_wfpr=k_wfpr, g=g, h=h_dyn30, slope=slope),0), d50_mm_
dyn30)))

# flow depth (h) increases in both confined and unconfined channels by dynamic response factor (see constants)

# No wood d50: If channel type is PR, calculate d50 using PR function and new h. If not, calculate d50 using PB f
unction and new h.
# Wood d50: If d50 > 50, calculate d50 using wood-forced PR function and new h. If not, write d50 from no wood sc
enario.

write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
```

Dynamic20 No Wood

```
### Create a "dynamic20_nowood" folder, manipulate data frames and save as .csv files

## -------------------------USER INPUT-----------------------------------
setup(timeframe = "dynamic", condition = "20_nowood")
## ----------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under dynamic20 (no wood) scenario
and saving each data frame as a .csv in the "dynamic20_nowood" folder. Combine all regions into one data frame an
d save.
# Under dynamic scenarios, new channel dimensions are used to identify suitable spawning reaches.

full.list<-list()
for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, length_m, slope, h_dyn20, w, confinement, d50_mm_dyn20) %>%
         mutate(

coho_habitat =
ifelse(slope < 2.0 & w > 2.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_dyn20 >= 10 & d50_mm_dyn20 <= 50, "high"
, ifelse(slope < 4.5 & w > 2.0 & h_dyn20 > 0.5 & confinement == 1 & d50_mm_dyn20 >= 10 & d50_mm_dyn20 <= 50, "mod
erate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_dyn20 >= 5 & d50_mm_dyn20 <= 25, "high"
, ifelse(slope <= 1.0 & w > 2.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_dyn20 >= 5 & d50_mm_dyn20 <= 25, "hig
h", ifelse(slope < 4.5 & w > 10.0 & h_dyn20 > 0.5 & confinement == 1 & d50_mm_dyn20 >= 5 & d50_mm_dyn20 <= 25, "m
oderate", "NA"))),

chum_habitat =
ifelse(slope < 2.0 & w > 10.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_dyn20 >= 5 & d50_mm_dyn20 <= 50, "high"
, ifelse(slope <= 1.0 & w > 2.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_dyn20 >= 5 & d50_mm_dyn20 <= 50, "hig
h", ifelse(slope < 4.5 & w > 10.0 & h_dyn20 > 0.5 & confinement == 1 & d50_mm_dyn20 >= 5 & d50_mm_dyn20 <= 50, "m
oderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Dynamic20 Wood

```
## -------------------------USER INPUT-----------------------------------
setup(timeframe = "dynamic", condition = "20_wood")
## --------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under dynamic20 (wood) scenario and
saving each data frame as a .csv in the "dynamic20_wood" folder. Combine all regions into one data frame and save
.
# Under dynamic scenarios, new channel dimensions are used to identify suitable spawning reaches.

full.list<-list()
for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, length_m, slope, h_dyn20, w, confinement, d50_mm_wfr_dyn20) %>%
  mutate(

coho_habitat =
ifelse(slope < 2.0 & w > 2.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_wfr_dyn20 >= 10 & d50_mm_wfr_dyn20 <= 50
, "high", ifelse(slope < 4.5 & w > 2.0 & h_dyn20 > 0.5 & confinement == 1 & d50_mm_wfr_dyn20 >= 10 & d50_mm_wfr_d
yn20 <= 50, "moderate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_wfr_dyn20 >= 5 & d50_mm_wfr_dyn20 <= 25
, "high", ifelse(slope <= 1.0 & w > 2.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_wfr_dyn20 >= 5 & d50_mm_wfr_d
yn20 <= 25, "high", ifelse(slope < 4.5 & w > 10.0 & h_dyn20 > 0.5 & confinement == 1 & d50_mm_wfr_dyn20 >= 5 & d5
0_mm_wfr_dyn20 <= 25, "moderate", "NA"))),

chum_habitat =
ifelse(slope < 2.0 & w > 10.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_wfr_dyn20 >= 5 & d50_mm_wfr_dyn20 <= 50
, "high", ifelse(slope <= 1.0 & w > 2.0 & h_dyn20 > 0.5 & confinement == 0 & d50_mm_wfr_dyn20 >= 5 & d50_mm_wfr_d
yn20 <= 50, "high", ifelse(slope < 4.5 & w > 10.0 & h_dyn20 > 0.5 & confinement == 1 & d50_mm_wfr_dyn20 >= 5 & d5
0_mm_wfr_dyn20 <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Dynamic30 No Wood

```
## -------------------------USER INPUT-----------------------------------
setup(timeframe = "dynamic", condition = "30_nowood")
## --------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under dynamic30 (no wood) scenario
and saving each data frame as a .csv in the "dynamic30_nowood" folder. Combine all regions into one data frame an
d save.
# Under dynamic scenarios, new channel dimensions are used to identify suitable spawning reaches.

full.list<-list()
for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, length_m, slope, h_dyn30, w, confinement, d50_mm_dyn30) %>% mutate(

coho_habitat =
ifelse(slope < 2.0 & w > 2.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_dyn30 >= 10 &  d50_mm_dyn30 <= 50, "high
", ifelse(slope < 4.5 & w > 2.0 & h_dyn30 > 0.5 & confinement == 1 & d50_mm_dyn30 >= 10 & d50_mm_dyn30 <= 50, "mo
derate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_dyn30 >= 5 & d50_mm_dyn30 <= 25, "high"
, ifelse(slope <= 1.0 & w > 2.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_dyn30 >= 5 & d50_mm_dyn30 <= 25, "hig
h", ifelse(slope < 4.5 & w > 10.0 & h_dyn30 > 0.5 & confinement == 1 & d50_mm_dyn30 >= 5 & d50_mm_dyn30 <= 25, "m
oderate", "NA"))),

chum_habitat = ifelse(slope < 2.0 & w > 10.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_dyn30 >= 5 & d50_mm_dyn3
0 <= 50, "high", ifelse(slope <= 1.0 & w > 2.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_dyn30 >= 5 & d50_mm_dy
n30 <= 50, "high", ifelse(slope < 4.5 & w > 10.0 & h_dyn30 > 0.5 & confinement == 1 & d50_mm_dyn30 >= 5 & d50_mm_
dyn30 <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Dynamic30 Wood

```
## -------------------------USER INPUT-----------------------------------
setup(timeframe = "dynamic", condition = "30_wood")
## --------------------------------------------------------------------

## Run loop, creating data frames specifying habitat quality for each species under dynamic30 (wood) scenario and
saving each data frame as a .csv in the "dynamic30_wood" folder. Combine all regions into one datanframe and save
.
# Under dynamic scenarios, new channel dimensions are used to identify suitable spawning reaches.

full.list<-list()
for (i in 1:length(list.dfs)){
assign(new_names[i], as.data.frame(list.dfs[i], col.names = names(columns)) %>%
         select(id, region, region_id, length_m, slope, h_dyn30, w, confinement, d50_mm_wfr_dyn30) %>%
  mutate(

coho_habitat = ifelse(slope < 2.0 & w > 2.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_wfr_dyn30 >= 10 & d50_mm_
wfr_dyn30 <= 50, "high", ifelse(slope < 4.5 & w > 2.0 & h_dyn30 > 0.5 & confinement == 1 & d50_mm_wfr_dyn30 >= 10
& d50_mm_wfr_dyn30 <= 50, "moderate","NA")),

pink_habitat =
ifelse(slope < 2.0 & w > 10.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_wfr_dyn30 >= 5 & d50_mm_wfr_dyn30 <= 25
, "high", ifelse(slope <= 1.0 & w > 2.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_wfr_dyn30 >= 5 & d50_mm_wfr_d
yn30 <= 25, "high", ifelse(slope < 4.5 & w > 10.0 & h_dyn30 > 0.5 & confinement == 1 & d50_mm_wfr_dyn30 >= 5 & d5
0_mm_wfr_dyn30 <= 25, "moderate", "NA"))),

chum_habitat =
ifelse(slope < 2.0 & w > 10.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_wfr_dyn30 >= 5 & d50_mm_wfr_dyn30 <= 50
, "high", ifelse(slope <= 1.0 & w > 2.0 & h_dyn30 > 0.5 & confinement == 0 & d50_mm_wfr_dyn30 >= 5 & d50_mm_wfr_d
yn30 <= 50, "high", ifelse(slope < 4.5 & w > 10.0 & h_dyn30 > 0.5 & confinement == 1 & d50_mm_wfr_dyn30 >= 5 & d5
0_mm_wfr_dyn30 <= 50, "moderate", "NA")))))

# Export full data table for joining in GIS
full.list[[i]]<-get(new_names[i])
write.csv(get(new_names[i]), paste0(new_names[i], ".csv", sep= ""))
}
assign(paste("full", "_", timeframe,"_", condition, sep = ""), rbindlist(full.list, fill = TRUE))
write.csv(get(paste("full", "_", timeframe,"_", condition, sep = "")), paste("full", "_", timeframe,"_", conditio
n,".csv", sep= ""))
```

Scenario Summaries

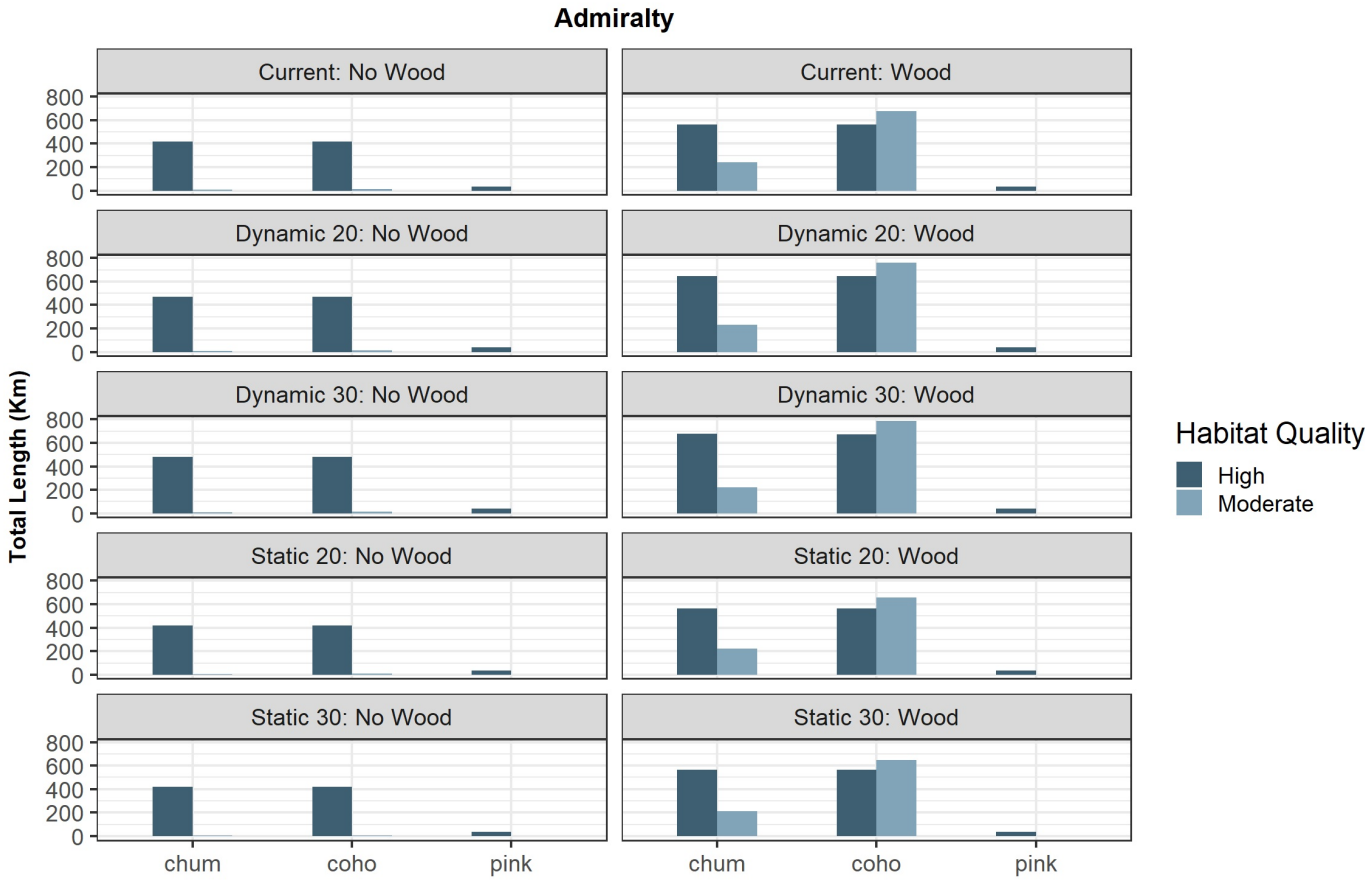
```
### Create scenario summaries indicating total channel length of each habitat quality type for each species and r
egion

all_scenarios <- c("current_nowood", "current_wood", "dynamic_20_nowood", "dynamic_20_wood", "dynamic_30_nowood",
"dynamic_30_wood", "static_20_nowood", "static_20_wood", "static_30_nowood", "static_30_wood")


for (i in 1:length(all_scenarios)){
  df <- get(paste("full", "_", paste(all_scenarios[i]), sep= ""))%>%
  pivot_longer(cols= coho_habitat:chum_habitat,
               names_to = "species",
               values_to = "habitat_quality") %>%
  group_by(region, species, habitat_quality ) %>%
  summarize(total_length = sum(length_m)) %>%
  filter(habitat_quality %in% c("high", "moderate"))
df$scenario <- paste(all_scenarios[i])
df$species[df$species == "chum_habitat"] <-"chum"
df$species[df$species == "coho_habitat"] <-"coho"
df$species[df$species == "pink_habitat"] <-"pink"
df<- df %>%
  rename(Region = region, Species = species, "Habitat Quality" = habitat_quality , "Total Length" = total_length
, Scenario = scenario) %>%
  select(Region, Scenario, Species, "Habitat Quality", "Total Length")
assign(paste("summary", "_", paste(all_scenarios[i]), sep = ""), df)
}
```

Initial Exploratory Graphs

```
## -------------------------USER INPUT-----------------------------------
graph_region <- "Admiralty"
## --------------------------------------------------------------------

all_scenarios <- c("current_nowood", "current_wood", "dynamic_20_nowood", "dynamic_20_wood", "dynamic_30_nowood",
"dynamic_30_wood", "static_20_nowood", "static_20_wood", "static_30_nowood", "static_30_wood") ## code from above
code chunk
```

```
graph_scenarios <- c(paste(graph_region, "_", all_scenarios[1], sep = ""),
paste(graph_region,"_",all_scenarios[2], sep = ""),
paste(graph_region,"_",all_scenarios[3], sep = ""),
paste(graph_region,"_",all_scenarios[4], sep = ""),
paste(graph_region,"_",all_scenarios[5], sep = ""),
paste(graph_region,"_",all_scenarios[6], sep = ""),
paste(graph_region,"_",all_scenarios[7], sep = ""),
paste(graph_region,"_",all_scenarios[8], sep = ""),
paste(graph_region,"_",all_scenarios[9], sep = ""),
paste(graph_region,"_",all_scenarios[10], sep = ""))

for (i in 1:length(graph_scenarios)){
   df <- get(graph_scenarios[i]) %>%
pivot_longer(cols= coho_habitat:chum_habitat,
               names_to = "species",
               values_to = "habitat_quality") %>%
  mutate(scenario = all_scenarios[i])%>%
  select(region, species, habitat_quality,scenario, length_m ) %>%
  group_by(region,species, habitat_quality, scenario ) %>%
  summarize(length_km= (sum(length_m))/1000) %>%
  filter(habitat_quality %in% c("high", "moderate"))
df$species[df$species == "chum_habitat"] <-"chum"
df$species[df$species == "coho_habitat"] <-"coho"
df$species[df$species == "pink_habitat"] <-"pink"
assign(paste("graph_sum", "_", i, sep = ""), df)
}

all_summaries <- c(paste("graph_sum_", "1", sep = ""),
paste("graph_sum_","2", sep = ""),
paste("graph_sum_",  "3", sep = ""),
paste("graph_sum_",  "4", sep = ""),
paste("graph_sum_",  "5", sep = ""),
paste("graph_sum_",  "6", sep = ""),
paste("graph_sum_",  "7", sep = ""),
paste("graph_sum_",  "8", sep = ""),
paste("graph_sum_",  "9", sep = ""),
paste("graph_sum_", "10", sep = ""))


graph_df<-data.frame()
for (i in 1:length(all_summaries)){
  graph_df <-graph_df %>%
    rbind(get(paste(all_summaries[i])))
}

graph_df$habitat_quality[graph_df$habitat_quality == "high"]<- "High"
graph_df$habitat_quality[graph_df$habitat_quality == "moderate"]<- "Moderate"


new_names <- c("current_nowood" = "Current: No Wood",
"current_wood" = "Current: Wood",
"dynamic_20_nowood"= "Dynamic 20: No Wood",
"dynamic_20_wood"= "Dynamic 20: Wood",
"dynamic_30_nowood"= "Dynamic 30: No Wood",
"dynamic_30_wood"= "Dynamic 30: Wood",
"static_20_nowood"= "Static 20: No Wood",
"static_20_wood"= "Static 20: Wood",
"static_30_nowood"= "Static 30: No Wood",
"static_30_wood"= "Static 30: Wood")

legend_title <- "Habitat Quality"

theme_set(theme_bw(18))

 ggplot(graph_df, aes(x = species, y = length_km, fill = habitat_quality))+
  geom_bar(stat= "identity", position = position_dodge(), width=0.5) +
  #geom_bar(aes(color = habitat_quality), show.legend = TRUE)+
  facet_wrap(~scenario, nrow=5, labeller = as_labeller(new_names))+
  #geom_text(aes(label = length_km), fontface = "bold", vjust = -1,
             #position = position_dodge(.9), size = 3)
  labs(title = graph_region, x = "", y = "Total Length (Km)") +
   theme(plot.title = element_text(hjust= 0.5,
                                   face = "bold",
                                   size = 16,
                                   margin=margin(0,0,10,0)))+
  theme(axis.title.x = element_text (face = "bold",
                                     size = 14,
                                     margin=margin(10,0,0,0))) +
```

```
theme(axis.title.y = element_text (face = "bold",
                                   size = 14,
                                   margin=margin(0,10,0,0)))+
scale_fill_manual(legend_title, values = c("#3E5E71", "#82A4B8"))
```




```
#scale_y_continuous(limits= c(0,700), expand = c(0, 0))
```