# Data Preparation Scripts

* Note that these scrips are fluid and sections were commented / uncommented for analysis and checking purposes; use with great care and attention

Table of Contents

[Data Preparation Scripts 1](#_Toc17443316)

[Download ECCC data and write to file 2](#_Toc17443317)

[Prepare the driving data from each source for comparison between sources 3](#_Toc17443318)

[Combine Air Pressure data 12](#_Toc17443319)

[Combine Shortwave and Longwave Data 15](#_Toc17443320)

[Combine Precipitation Data 20](#_Toc17443321)

[Combine Specific Humidity Data 24](#_Toc17443322)

[Combine Temperature Data 27](#_Toc17443323)

[Combine Wind Speed Data 33](#_Toc17443324)

[Soil Temperature and Moisture Observations 36](#_Toc17443325)

[Streamflow Data 42](#_Toc17443326)

[Model Output Processing – R Notebook 48](#_Toc17443327)

## Download ECCC data and write to file

library(weathercan)

library(CRHMr)

library(tidyverse)

#Download the weather station data from the ECCC site; can specify dates from and to if you want

#Yellowknife has 2 different station names which cover the period from 1953 to present

y1706\_hr <- weather\_dl(station\_id = 1706,interval="hour")

y1706\_day <- weather\_dl(station\_id = 1706,interval="day")

y51058\_hr <- weather\_dl(station\_id = 51058,interval="hour")

y51058\_day <- weather\_dl(station\_id = 51058,interval="day")

#Combine the hourly and daily data from the different station numbers into one

yknife\_all\_hr <- rbind(y1706\_hr,y51058\_hr)

yknife\_all\_day <- rbind(y1706\_day,y51058\_day)

#Change the yellowknife data to a data frame

yknife\_all\_hr <- as.data.frame(yknife\_all\_hr)

yknife\_all\_day <- as.data.frame(yknife\_all\_day)

#Reorder the columns so that the date and time column is first, and rename it to "datetime" for use with CRHMr

yknife\_all\_hr2 <- yknife\_all\_hr[,c(12,1,2,3,4,5,6,7,8,9,10,11,13:ncol(yknife\_all\_hr))]

yknife\_all\_hr2 <- rename(yknife\_all\_hr2, datetime=time)

#Save the weather data to a csv

setwd('C:/<your working directory here, if not already set where you want>')

write.csv(yknife\_all\_hr,file='./YellowknifeA\_Hourly (no precip).csv')

write.csv(yknife\_all\_day,file='./YellowknifeA\_Daily.csv')

#Save the Rda object to file for easy loading later

save(yknife\_all\_hr,file="yknife\_all\_hr.Rda")

save(yknife\_all\_day,file="yknife\_all\_day.Rda")

## Prepare the driving data from each source for comparison between sources

#The purpose of this script is to gather, inspect, clean, and compile the driving data for Haley and Sadiq's MWS Capstone Projects

#Load libraries

library(tidyverse)

library(dplyr)

library(lubridate)

library(zoo)

#---------------------------------------------------------------------------------------------------

#LOAD DATA

#Vital Tower

colnames\_vital=c('Year', 'DateTime', 'AirP', 'Kin', 'Kout', 'Lin', 'Lout', 'Qstar', 'T\_4.4m', 'RH\_4.4m', 'T\_2m', 'RH\_2m', 'u\_4.4m', 'Rain\_mm', 'Qe', 'Qh')

Vital\_load <- read\_csv(file="C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/Data/ESSD Baker Creek Data/HydrometeorologicalData/vital tower half hourly time series v1.csv", col\_names = colnames\_vital, skip=1)

#Units are: kPa, W/m2, degC, %, m/s, and mm

#Fix the format of the DateTime column

Vital\_load$DateTime <- dmy\_hm(Vital\_load$DateTime)

Vital <- Vital\_load

Vital$DateTime <- force\_tz(Vital$DateTime,tzone="America/Yellowknife")

head(Vital)

#Replace all "9999" values with "NA"

Vital[Vital==9999]<- NA

Vital[Vital==99999]<- NA

#Convert air pressure from kPa to Pa (replace)

#Convert RH to specific humidity (replace; use T at the same height as RH for conversion), and

#Add a precip rate (mm/s) column;

#Specific humidity in kg kg-1 is given as:

#qa = 0.622ea / (Pa - 0.378 ea)

# where:qa = Specific humidity in kg kg-1

# Pa = Surface pressure in Pa

# ea = Vapour pressure:

# ea = rh10^[(0.7859 + 0.03477Ta)/(1.0 + 0.00412Ta) + 2]

# where:

# Ta = Air temperature in °C

# rh = Relative humidity in %

Vital <- Vital %>%

mutate(AirP\_Pa=AirP\*1000) %>%

mutate(ea\_4.4m=RH\_4.4m/100\*10^((0.7859+0.03477\*T\_4.4m)/(1.0+0.00412\*T\_4.4m)+2))%>%

mutate(q\_4.4m=0.622\*ea\_4.4m/(AirP\_Pa-0.378\*ea\_4.4m)) %>%

mutate(ea\_2m=RH\_2m/100\*10^((0.7859+0.03477\*T\_2m)/(1.0+0.00412\*T\_2m)+2))%>%

mutate(q\_2m=0.622\*ea\_2m/(AirP\_Pa-0.378\*ea\_2m)) %>%

# mutate(qa\_4.4m=0.622\*RH\_4.4m/100\*611\*exp(17.27\*T\_4.4m/(T\_4.4m+237.3))/AirP\_Pa) %>%

# mutate(qa\_1.1m=0.622\*RH\_1.1m/100\*611\*exp(17.27\*T\_1.1m/(T\_1.1m+237.3))/AirP\_Pa) %>%

mutate(Rain\_rate=Rain\_mm/0.5/3600) #Convert mm/0.5 hr to mm/s

#Remove unneeded columns (AirP(kPa), RH, and ea(intermediate calculation))

Vital <- Vital %>% select(-c(AirP,ea\_4.4m, ea\_2m, Year))

#Check specific humidity calculation using Dingman eq'n 3.9a, 3.11 and 3.12

#write\_csv(Vital,"Vital\_Check.csv")

# RH = 51.975 #Percent

# Ta = 16.791 #degrees C

# Pa = 97641 #Air pressure, Pa

#

# e=611\*exp(17.27\*Ta/(Ta+237.3))

# q=0.622\*RH/100\*611\*exp(17.27\*Ta/(Ta+237.3))/Pa

#

# ea=RH/100\*10^((0.7859+0.03477\*Ta)/(1.0+0.00412\*Ta)+2)

# q\_wiki=0.622\*ea/(Pa-0.378\*ea)

#Add missing columns and re-order columns

full\_colnames <- read\_csv("met\_variable\_names.csv")

full\_colnames <- colnames(full\_colnames)

Vital\_cols <- colnames(Vital)

missing\_colnames <- setdiff(full\_colnames,Vital\_cols)

Vital[,missing\_colnames] <- NA

Vital$Station <- "Vital"

Vital$Precip\_rate <- Vital$Rain\_rate

Vital <- Vital[, full\_colnames]

#Remove rows that are all NA

Vital <- Vital[rowSums(is.na(Vital))!= ncol(Vital)-2,]

save(Vital, file="Vital.Rda")

#qplot(data=Vital,x=DateTime, y=T\_4.4m, geom='point')

#qplot(data=Vital,x=DateTime, y=AirP\_Pa, geom='point')

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#Landing Tower

colnames\_landing=c('DateTime', 'u\_1.1m', 'u\_dir', 'T\_1.1m', 'e\_1.1m', 'Qstar', 'Kin', 'Kout', 'Twater', 'Qe', 'Qh')

Landing\_load <- read\_csv(file="C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/Data/ESSD Baker Creek Data/HydrometeorologicalData/landing tower half hourly time series v1.csv",col\_names=colnames\_landing, skip=1)

#Units are: kPa, W/m2, degC, %, m/s, and mm; Actual meas. height of T and e is 1.4 m; labelled as 1.1m for simplification with other datasets

#Fix the format of the DateTime column

Landing\_load$DateTime <- dmy\_hm(Landing\_load$DateTime)

Landing <- Landing\_load

Landing$DateTime <- force\_tz(Landing$DateTime,tzone="America/Yellowknife")

#Replace all "9999" values with "NA"

Landing[Landing==9999]<- NA

#Convert e to specific humidity (replace; use T at the same height as e and air pressure from Vital for conversion)

Vital\_AirP <- tibble(DateTime=Vital$DateTime, Vital\_AirP\_Pa=Vital$AirP\_Pa)

Landing <- merge(x=Landing,y=Vital\_AirP, by="DateTime", all=TRUE)

Landing <- Landing %>%

mutate(e\_1.1m=e\_1.1m\*1000) %>% #Convert kPa to Pa

mutate(q\_1.1m=0.622\*e\_1.1m/(Vital\_AirP\_Pa-0.378\*e\_1.1m))

#Remove unneeded columns (used for intermediate calculation)

Landing <- Landing %>% select(-c(u\_dir,Twater,Vital\_AirP\_Pa, ea\_1.1m))

#Add missing columns and re-order columns

full\_colnames <- read\_csv("met\_variable\_names.csv") #This was done above

full\_colnames <- colnames(full\_colnames)

Landing\_cols <- colnames(Landing)

missing\_colnames <- setdiff(full\_colnames,Landing\_cols)

Landing[,missing\_colnames] <- NA

Landing$Station <- "Landing"

Landing <- Landing[, full\_colnames]

#Remove rows that are all NA

Landing <- Landing[rowSums(is.na(Landing))!= ncol(Landing)-2,]

save(Landing, file="Landing.Rda")

#qplot(data=Landing,x=DateTime, y=T\_1.1m, geom='point')

# ggplot(data=Landing) +

# facet\_grid(year(Landing$DateTime) ~ .) +

# geom\_line(mapping=aes(x=month(Landing$DateTime), y=T\_1.1m))

# qplot(data=Landing,x=DateTime, y=Kin, geom='point')

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#GEM

#Load the files

GEMFiles <- list.files(path="C:/Users/haley/Documents/1. MWS 2018-2019/T2/ECCC Project\_Cdrive/GEM\_CaPA\_Data", pattern=glob2rx("rdps\*.csv"))

GEMFiles

file\_names <- c("Lin", "AirP\_Pa", "Kin", "q\_2m", "q\_40m", "T\_2m\_degC","T\_2m\_degK","T\_40m\_degC", "T\_40mdegK", "u\_10m", "u\_40m")

#Units: W/m2, Pa, kg/kg, degrees C, degrees K, m/s; no need to do conversions

setwd("C:/Users/haley/Documents/1. MWS 2018-2019/T2/ECCC Project\_Cdrive/GEM\_CaPA\_Data")

i=1

colNames\_GEM=c("DateTime", "GEMYellowknifeA", "GEMLanding", "GEMVital")

for(x in GEMFiles) {

assign(file\_names[i],read\_csv(x,skip=3,col\_names=colNames\_GEM)) #Read the file and assign name

i=i+1

}

#Change the hourly GEM timeseries to halfhourly using the user-defined "oneToHalfHr" function

# If you want to assign values to the half hour by interpolating, pass interpolate=1, otherwise interpolate=0 will assign the hour value to the following half hour

#This block of code was for testing purposes

#Lin\_test <- Lin

#Lin\_test <- filter(Lin\_test, DateTime <= "2005-05-18 17:00:00")

### Note: this is returning the last value at 2005-05-18 23:00:00, which is UTC+6

oneToHalfHr <- function(myData, interpolate) {

p <- period(30, unit="minutes")

HalfHourly <- myData

if (interpolate == 1) {

HalfHourly[,1] <- pull(HalfHourly[,1]) - p

for (i in 2:ncol(myData)){

HalfHourly[,i] <- rollmean(myData[,i], 2, align="right", fill=NA)

}

HalfHourly <- HalfHourly[-1,]

} else {

HalfHourly[,1] <- pull(HalfHourly[,1]) + p

#HalfHourly <- HalfHourly[-nrow(HalfHourly),]

}

myData <- rbind(myData, HalfHourly)

myData <- arrange(myData,DateTime)

}

Lin <- oneToHalfHr(Lin,1)

Kin <- oneToHalfHr(Kin,1)

AirP\_Pa <- oneToHalfHr(AirP\_Pa,1)

q\_2m <- oneToHalfHr(q\_2m,1)

q\_40m <- oneToHalfHr(q\_40m,1)

T\_2m\_degC <- oneToHalfHr(T\_2m\_degC,1)

T\_2m\_degK <- oneToHalfHr(T\_2m\_degK,1)

T\_40m\_degC <- oneToHalfHr(T\_40m\_degC,1)

T\_40mdegK <- oneToHalfHr(T\_40mdegK,1)

u\_10m <- oneToHalfHr(u\_10m,1)

u\_40m <- oneToHalfHr(u\_40m,1)

#Test if the average is preserved

ColNamesu\_10m <- colnames(u\_10m)

Avg\_check <- summarise\_each(u\_10m, fun=mean, ColNamesu\_10m[-1])

Avg\_check[2,] <- summarise\_each(u\_10m2, fun=mean, ColNamesu\_10m[-1])

Avg\_check[3,] <- summarise\_each(Avg\_check, fun=diff, ColNamesu\_10m[-1])

##### Using the interpolation method, the mean is of by about 0.0002%; Using the stepwise method, there is no difference in the mean

#Gather and combine all the datasets; couldn't figure out how to do this in a loop

#First, create the Dataset

Lin <- gather(Lin,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='Lin')

GEM\_data <- Lin

#Then gather each value and add to the combined dataset, GEM\_data

AirP\_Pa <- gather(AirP\_Pa,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='AirP\_Pa')

GEM\_data <- merge(x=GEM\_data, y=AirP\_Pa, by=c("DateTime","Station"), all=TRUE)

Kin <- gather(Kin,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='Kin')

GEM\_data <- merge(x=GEM\_data, y=Kin, by=c("DateTime","Station"), all=TRUE)

q\_2m <- gather(q\_2m,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='q\_2m')

GEM\_data <- merge(x=GEM\_data, y=q\_2m, by=c("DateTime","Station"), all=TRUE)

q\_40m <- gather(q\_40m,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='q\_40m')

GEM\_data <- merge(x=GEM\_data, y=q\_40m, by=c("DateTime","Station"), all=TRUE)

T\_2m\_degC <- gather(T\_2m\_degC,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='T\_2m')

GEM\_data <- merge(x=GEM\_data, y=T\_2m\_degC, by=c("DateTime","Station"), all=TRUE)

T\_40m\_degC <- gather(T\_40m\_degC,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='T\_40m')

GEM\_data <- merge(x=GEM\_data, y=T\_40m\_degC, by=c("DateTime","Station"), all=TRUE)

u\_10m <- gather(u\_10m,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='u\_10m')

GEM\_data <- merge(x=GEM\_data, y=u\_10m, by=c("DateTime","Station"), all=TRUE)

u\_40m <- gather(u\_40m,'GEMYellowknifeA', 'GEMLanding', 'GEMVital', key="Station", value='u\_40m')

GEM\_data <- merge(x=GEM\_data, y=u\_40m, by=c("DateTime","Station"), all=TRUE)

#Not including T\_2m\_degK or T\_40m\_degK -> will convert all temps to K later

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

#Convert the GEM data to Local time in Yellowknife (UTC-7)

GEM\_data <- GEM\_data %>%

mutate(DateTime=update(DateTime,hour=hour(DateTime)-7))

#add\_column(Unit='W m-2', Height='', Observation='LWin')

#gather(x, 'YellowknifeA', 'Landing', 'Vital',key='Location', value='Observation')

#Add missing columns and rearrange to match other datasets

load("./GEM\_data.Rda")

full\_colnames <- read\_csv("met\_variable\_names.csv") #This was done above

full\_colnames <- colnames(full\_colnames)

GEM\_cols <- colnames(GEM\_data)

missing\_colnames <- setdiff(full\_colnames,GEM\_cols)

GEM\_data[,missing\_colnames] <- NA

GEM\_data <- GEM\_data[, full\_colnames]

GEM\_data$DateTime <- force\_tz(GEM\_data$DateTime, tzone="America/Yellowknife")

save(GEM\_data, file="GEM\_data.Rda")

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#CaPA

setwd("C:/Users/haley/Documents/1. MWS 2018-2019/T2/ECCC Project\_Cdrive/GEM\_CaPA\_Data")

colNames\_CAPA=c("DateTime", "CAPAYellowknifeA", "CAPALanding", "CAPAVital")

CAPA\_load <- read\_csv("rdpa\_rain\_nearest\_20020101\_20190101.csv",skip=3,col\_names=colNames\_CAPA)

#Convert the CAPA data to Local time in Yellowknife (UTC-7)

CAPA\_data <- CAPA\_load

CAPA\_data <- CAPA\_data %>%

mutate(DateTime=update(DateTime,hour=hour(DateTime)-7))

#Change the CAPA 6 hr timeseries into 1/2 hour timeseries and backfill the values of precip rate

# Created a spreadsheet to manually create the timeseries

# Found out later could do it this way:

# dates\_seq <- seq(as.POSIXct("2015-01-01"), as.POSIXct("2018-12-31"), by=(30))

# dates\_seq <- tibble(dates\_seq)

# write.csv(CAPA\_data\_0.5hr,"C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/CAPA\_timeseries.csv")

CAPA\_data <- CAPA\_data %>% filter(year(DateTime)>=2005)

CAPA\_data\_0.5hr <- read\_csv("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/CAPA\_timeseries.csv", skip=0, col\_names="DateTime")

#This is a half-hourly timeseries in the 1st column with no heading

CAPA\_data\_0.5hr$DateTime <- ymd\_hm(CAPA\_data\_0.5hr$DateTime)

head(CAPA\_data\_0.5hr)

#This takes a long time to run but it works

CAPA0.5\_row=1

CAPA\_row=1

numit=nrow(CAPA\_data\_0.5hr)/12

for (i in 1:numit){

for (j in 1:12){

CAPA\_data\_0.5hr[CAPA0.5\_row,2] <- CAPA\_data[CAPA\_row,2]

CAPA\_data\_0.5hr[CAPA0.5\_row,3] <- CAPA\_data[CAPA\_row,3]

CAPA\_data\_0.5hr[CAPA0.5\_row,4] <- CAPA\_data[CAPA\_row,4]

CAPA0.5\_row=CAPA0.5\_row+1

}

CAPA\_row=CAPA\_row+1

}

#write.csv(CAPA\_data\_0.5hr,"C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/CAPA\_timeseries.csv")

CAPA\_data\_0.5hr <- gather(CAPA\_data\_0.5hr,'CAPAYellowknifeA', 'CAPALanding', 'CAPAVital', key="Station", value='Precip\_rate')

setwd('C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/')

save(CAPA\_data\_0.5hr, file='CAPA\_data\_0.5hr.Rda')

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#Yellowknife Hourly Met Data

setwd('C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/')

load('yknife\_all\_hr.Rda')

Yknife <- yknife\_all\_hr

# glimpse(Yknife)

# head(Yknife$date)

# tail(Yknife$date)

#Select only the variables needed and rename the columns

Yknife\_cols <- colnames(Yknife)

Yknife <- select(Yknife,time,pressure, rel\_hum, temp, wind\_spd)

Yknife <- rename(Yknife, DateTime=time, AirP=pressure, RH\_2m=rel\_hum, T\_2m=temp, u\_10m=wind\_spd)

#Assumed temperature an humidity are at a height of 2m

Yknife\_cols <- colnames(Yknife)

#Convert AirPressure to Pa from kPa, RH to q, and u from km/h to m/s

Yknife <- Yknife %>%

mutate(AirP\_Pa=AirP\*1000) %>%

mutate(ea\_2m=RH\_2m/100\*10^((0.7859+0.03477\*T\_2m)/(1.0+0.00412\*T\_2m)+2))%>%

mutate(q\_2m=0.622\*ea\_2m/(AirP\_Pa-0.378\*ea\_2m)) %>%

mutate(u\_10m=u\_10m/3.6)

# mutate(qa\_4.4m=0.622\*RH\_4.4m/100\*611\*exp(17.27\*T\_4.4m/(T\_4.4m+237.3))/AirP\_Pa) #Dingman eq'ns

#Remove unneeded columns (AirP(kPa), RH, and ea(intermediate calculation))

Yknife <- Yknife %>% select(-c(AirP))

#Add missing columns and re-order columns

full\_colnames <- read\_csv("met\_variable\_names.csv") #This was done above

full\_colnames <- colnames(full\_colnames)

Yknife\_cols <- colnames(Yknife)

missing\_colnames <- setdiff(full\_colnames,Yknife\_cols)

Yknife[,missing\_colnames] <- NA

Yknife$Station <- "YellowknifeA"

Yknife <- Yknife[, full\_colnames]

Yknife\_1hr <- Yknife

save(Yknife\_1hr, file="Yknife\_1hr.Rda")

load("./Yknife\_1hr.Rda")

Yknife <- filter(Yknife, DateTime > "2005-01-01 00:00:00", DateTime <= "2019-01-01 00:00:00")

#Convert Yellowknife Data to Half Hourly using interpolation between the hourly points

p <- period(30, unit="minutes")

HalfHourly <- Yknife

HalfHourly[,1] <- pull(HalfHourly[,1]) - p

for (i in 3:ncol(Yknife)){

#To interpolate between hourly observations for the half-hourly, use these lines:

HalfHourly[,i] <- rollmean(Yknife[,i], 2, align="right", fill=NA)

}

HalfHourly <- HalfHourly[-1,]

#If don't want to interpolate, use these lines instead:

#HalfHourly[,1] <- pull(HalfHourly[,1]) + p

#HalfHourly <- HalfHourly[-nrow(HalfHourly),]

Yknife <- rbind(Yknife, HalfHourly)

Yknife <- arrange(Yknife,DateTime)

save(Yknife, file="Yknife\_HalfHr.Rda")

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#Yellowknife Daily Precip Data

#Load Yellowknife Precip Data

load('yknife\_all\_day.Rda')

Yknife\_precip <- yknife\_all\_day

Yknife\_precip <- Yknife\_precip %>%

select(c(date, total\_precip, total\_rain, total\_snow, snow\_grnd)) %>%

filter(year(Yknife\_precip$date)>=2005) %>%

rename(DateTime=date)

head(Yknife\_precip)

Yknife\_precip <- Yknife\_precip %>% add\_column("NewDate"="00:00") %>%

mutate(DateTime=paste(DateTime,NewDate,sep=" ")) %>%

select(-NewDate)

Yknife\_precip$DateTime <- ymd\_hm(Yknife\_precip$DateTime)

save(Yknife\_precip, file="Yknife\_precip.Rda")

## Combine Air Pressure data

#####The purpose of this script is to pull in the processed driving data for the various stations around Baker Creek, NWT, plot the various variables, adjust the variables, and combine the data into 1 continuous set

#####Load Libraries

library(tidyverse)

library(dplyr)

library(lubridate)

#####Load the individual driving data files

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

load("./Vital.Rda")

load("./Landing.Rda")

load("./Yknife\_HalfHr.Rda")

load("./GEM\_data.Rda")

GEM\_data <- filter(GEM\_data, year(DateTime)>= 2005) #Filter down GEM data since longer period

#####Shift the 2009 Vital data back by 6 days to match the Yellowknife and GEM data

p <- period(6, units="day")

VitalShift09 <- mutate(Vital, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

#####Combine all the data

##Use this first section to choose whether or not to use the shifted version of the Vital data, or the original

DrivingShift <- rbind(VitalShift09, Landing, Yknife, GEM\_data)

DrivingOrig <- rbind(Vital, Landing, Yknife, GEM\_data)

##### Explore the Air Pressure data

P1 <- select(DrivingShift, DateTime, Station, AirP\_Pa)

P <- P1

P <- P %>%

filter(!Station %in% c("GEMLanding", "GEMYellowknifeA", "Landing")) %>%

mutate(Date=date(DateTime))

# group\_by(Date, Station) %>%

# summarise(DailyAvgP=mean(AirP\_Pa))

#Plot and compare the data

P$CommonDate <- as.Date(paste0("2000-", format(P$Date, "%j")),"%Y-%j")

P\_05\_11 <- filter(P, year(Date)>=2005 & year(Date)<=2011)

ggplot() +

geom\_line(data=P\_05\_11, mapping=aes(x=CommonDate, y=DailyAvgP, color=Station), size=0.5) +

facet\_grid(year(P\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Air Pressure - 2005-2011(shift6)")

P\_12\_18 <- filter(P, year(Date)>=2012 & year(Date)<=2018)

ggplot() +

geom\_line(data=P\_12\_18, mapping=aes(x=CommonDate, y=DailyAvgP, color=Station), size=0.5) +

facet\_grid(year(P\_12\_18$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Air Pressure - 2012-2018")

P\_09 <- P %>% filter(year(Date)==2009)

ggplot() +

geom\_line(data=P\_09, mapping=aes(x=CommonDate, y=DailyAvgP, color=Station), size=0.5) +

facet\_grid(year(P\_09$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Air Pressure - 2009(shift6)")

##### Combine the AirP dataset and write to file

###In order to combine, need the columns to be DateTime, Station1, Station2, etc.; make sure you haven't calculated Daily average values above (comment out the "group\_by" and "summarise" lines above)

PGEMVital <- filter(P, Station=="GEMVital")

PGEMVital <- PGEMVital %>%

rename(GEMVital=AirP\_Pa) %>%

select(DateTime, GEMVital)

PVital <- filter(P, Station=="Vital")

PVital <- PVital %>%

rename(Vital=AirP\_Pa) %>%

select(DateTime, Vital)

PComb <- PGEMVital

PComb <- merge(PComb, PVital, by="DateTime", all=TRUE)

PComb$Combined <- NA

PComb <- filter(PComb, is.na(DateTime)==FALSE)

PComb$Combined[is.na(PComb$Combined)] <- paste0(PComb$Vital[is.na(PComb$Combined)])

PComb$Combined <- as.double(PComb$Combined)

PComb$Combined[is.na(PComb$Combined)] <- paste0(PComb$GEMVital[is.na(PComb$Combined)])

PComb$Combined <- as.double(PComb$Combined)

#Plot and check the combination

Check <- gather(PComb, GEMVital, Vital, Combined, key="Location", value="AirP\_Pa")

Check <- Check %>%

as\_tibble %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date, Location) %>%

summarise(DayAvgP=mean(AirP\_Pa))

Check$LineSize <- rep(0.5, nrow(Check))

Check$LineSize[Check$Location=="Combined"]<- 1.0

Check$CommonDate <- as.Date(paste0("2001-", format(Check$Date, "%j")),"%Y-%j")

P\_09 <- filter(Check, year(Date)==2009)

ggplot(data=P\_09, mapping=aes(x=CommonDate, y=DayAvgP, color=Location, size=LineSize)) +

geom\_line() +

scale\_size(range=c(0.5,1.0), guide="none") +

facet\_grid(year(P\_09$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Air Pressure - 2009")

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

PFinal <- select(PComb, DateTime, Combined)

PWrite <- select(PComb, Combined)

write\_excel\_csv(PFinal, "../MESH Model/Baker Creek Model Files/basin\_pres.xlsx.csv")

write\_tsv(PWrite, "../MESH Model/Baker Creek Model Files/basin\_pres.csv", col\_names=FALSE)

## Combine Shortwave and Longwave Data

#####The purpose of this script is to pull in the processed driving data for the various stations around Baker Creek, NWT, plot the various variables, adjust the variables, and combine the data into 1 continuous set

#####Load Libraries

library(tidyverse)

library(dplyr)

library(lubridate)

#####Load the individual driving data files

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

load("./Vital.Rda")

load("./Landing.Rda")

load("./GEM\_data.Rda")

GEM\_data <- filter(GEM\_data, year(DateTime)>= 2005) #Filter down GEM data since longer period

# p <- period(6, units="day")

# VitalShift09 <- mutate(Vital, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

##### Remove Vital data in 2009 from June 21 to the end of the year for Kin and Lin, and 2016 before (and including) April 16 for the Kin data

VitalKin <- filter(Vital, !(date(DateTime)>= "2008-06-20" & date(DateTime)<= "2008-12-31"))

VitalKin <- filter(VitalKin, !(date(DateTime)>="2016-04-01"&date(DateTime)<"2016-04-16"))

#####Shift the 2009 Vital data back by 3 days to match the Yellowknife and GEM Temperature data

p <- period(3, units="day")

VitalKin <- mutate(VitalKin, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

VitalLin <- filter(Vital, !(date(DateTime)>= "2008-06-20" & date(DateTime)<= "2008-12-31"))

VitalLin <- mutate(VitalLin, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

#####Combine all the data

##Use this first section to choose whether or not to use the shifted, filtered, or original version of the Vital data

DrivingShift <- rbind(VitalShift09, Landing, GEM\_data)

DrivingOrig <- rbind(Vital, Landing, GEM\_data)

DrivingKin <- rbind(VitalKin, Landing, GEM\_data)

DrivingLin <- rbind(VitalLin, Landing, GEM\_data)

##### Explore the Kin (incoming shortwave radiation) data

Kin1 <- select(DrivingKin, DateTime, Station, Kin)

Kin <- Kin1

Kin <- Kin %>%

filter(!Station %in% c("GEMLanding", "GEMYellowknifeA", "YellowknifeA")) %>%

mutate(Date=date(DateTime)) #%>%

# group\_by(Date, Station) %>%

# summarise(DayAvgK=mean(Kin))

Kin$CommonDate <- as.Date(paste0("2001-", format(Kin$Date, "%j")),"%Y-%j")

#Plot and compare the data

Kin\_05\_11 <- filter(Kin, year(Date)>=2005 & year(Date)<=2011)

ggplot() +

geom\_line(data=Kin\_05\_11, mapping=aes(x=CommonDate, y=DayAvgK, color=Station), size=0.5) +

facet\_grid(year(Kin\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Incoming Shortwave - 2005-2011(filtered,shifted)")

Kin\_12\_18 <- filter(Kin, year(Date)>=2012)

ggplot() +

geom\_line(data=Kin\_12\_18, mapping=aes(x=CommonDate, y=DayAvgK, color=Station), size=0.5) +

facet\_grid(year(Kin\_12\_18$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Incoming Shortwave - 2012-2018(filtered)")

#

# Kin08Inspect <- filter(Kin, DateTime>="2008-06-20 00:00:00")

# Kin08Inspect <- spread(Kin08Inspect, key="Station", value="Kin")

# Kin08Inspect <- mutate(Kin08Inspect, Diff=GEMVital-Vital)

##### Explore the Lin (incoming longwave radiation) data

Lin1 <- select(DrivingLin, DateTime, Station, Lin)

Lin <- Lin1

Lin <- Lin %>%

filter(!Station %in% c("GEMLanding", "GEMYellowknifeA", "Landing", "YellowknifeA")) %>%

mutate(Date=date(DateTime)) #%>%

# mutate(Year = year(DateTime), Month=month(DateTime), Day=day(DateTime), Time=paste(hour(DateTime),minute(DateTime),second(DateTime),sep=":"))

# group\_by(Date, Station) %>%

# summarise(DayAvgL=mean(Lin))

Lin$CommonDate <- as.Date(paste0("2001-", format(Lin$Date, "%j")),"%Y-%j")

#Plot and compare the data

# Only change the filter specs and the title, not the name of the var.; save as jpg for compare

# Also change the "y" variable as necessary

Lin\_plot <- filter(Lin, year(Date)<=2011)

Lin\_plot <- Lin\_plot %>%

group\_by(Date, Station) %>%

summarise(DayAvgL=mean(Lin))

Lin\_plot$CommonDate <- as.Date(paste0("2001-", format(Lin\_plot$Date, "%j")),"%Y-%j")

ggplot() +

geom\_line(data=Lin\_plot, mapping=aes(x=CommonDate, y=DayAvgL, color=Station), size=0.5) +

facet\_grid(year(Lin\_plot$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Incoming Longwave - June-July 2008")

##### Combine the Kin dataset and write to file

###In order to combine, need the columns to be DateTime, Station1, Station2, etc.; make sure you haven't calculated Daily average values above (comment out the "group\_by" and "summarise" lines above)

KGEMVital <- filter(Kin, Station=="GEMVital")

KGEMVital <- KGEMVital %>%

rename(GEMVital=Kin) %>%

select(DateTime, GEMVital)

KVital <- filter(Kin, Station=="Vital")

KVital <- KVital %>%

rename(Vital=Kin) %>%

select(DateTime, Vital)

KLanding <- filter(Kin, Station=="Landing")

KLanding <- KLanding %>%

rename(Landing=Kin) %>%

select(DateTime, Landing)

KinComb <- KGEMVital

KinComb <- merge(KinComb, KVital, by="DateTime", all=TRUE)

KinComb <- merge(KinComb, KLanding, by="DateTime", all=TRUE)

KinComb$Combined <- NA

KinComb <- filter(KinComb, is.na(DateTime)==FALSE)

KinComb$Combined[is.na(KinComb$Combined)] <- paste0(KinComb$Vital[is.na(KinComb$Combined)])

KinComb$Combined <- as.double(KinComb$Combined)

KinComb$Combined[is.na(KinComb$Combined)] <- paste0(KinComb$Landing[is.na(KinComb$Combined)])

KinComb$Combined <- as.double(KinComb$Combined)

KinComb$Combined[is.na(KinComb$Combined)] <- paste0(KinComb$GEMVital[is.na(KinComb$Combined)])

KinComb$Combined <- as.double(KinComb$Combined)

#Plot and check the combination

Check <- gather(KinComb, GEMVital, Vital, Landing, Combined, key="Location", value="Kin")

Check <- Check %>%

as\_tibble %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date, Location) %>%

summarise(DayAvgK=mean(Kin))

Check$LineSize <- rep(0.5, nrow(Check))

Check$LineSize[Check$Location=="Combined"]<- 1.0

Check$CommonDate <- as.Date(paste0("2001-", format(Check$Date, "%j")),"%Y-%j")

Kin\_05\_11 <- filter(Check, year(Date)<=2011)

ggplot(data=Kin\_05\_11, mapping=aes(x=CommonDate, y=DayAvgK, color=Location, size=LineSize)) +

geom\_line() +

scale\_size(range=c(0.5,1.0), guide="none") +

facet\_grid(year(Kin\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Incoming Shortwave - 2005-2011(check)")

Kin\_12\_18 <- filter(Check, year(Date)>=2012)

ggplot(data=Kin\_12\_18, mapping=aes(x=CommonDate, y=DayAvgK, color=Location, size=LineSize)) +

geom\_line() +

scale\_size(range=c(0.5,1.0), guide="none") +

facet\_grid(year(Kin\_12\_18$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Incoming Shortwave - 2012-2018(check)")

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

KFinal <- select(KinComb, DateTime, Combined)

KWrite <- select(KinComb, Combined)

write\_excel\_csv(KFinal, "../MESH Model/Baker Creek Model Files/basin\_shortwave.xlsx.csv")

write\_tsv(KWrite, "../MESH Model/Baker Creek Model Files/basin\_shortwave.csv", col\_names=FALSE)

##### Combine the Lin dataset and write to file

###In order to combine, need the columns to be DateTime, Station1, Station2, etc.; make sure you haven't calculated Daily average values above (comment out the "group\_by" and "summarise" lines above)

LGEMVital <- filter(Lin, Station=="GEMVital")

LGEMVital <- LGEMVital %>%

rename(GEMVital=Lin) %>%

select(DateTime, GEMVital)

LVital <- filter(Lin, Station=="Vital")

LVital <- LVital %>%

rename(Vital=Lin) %>%

select(DateTime, Vital)

LinComb <- LGEMVital

LinComb <- merge(LinComb, LVital, by="DateTime", all=TRUE)

LinComb$Combined <- NA

LinComb <- filter(LinComb, is.na(DateTime)==FALSE)

LinComb$Combined[is.na(LinComb$Combined)] <- paste0(LinComb$Vital[is.na(LinComb$Combined)])

LinComb$Combined <- as.double(LinComb$Combined)

LinComb$Combined[is.na(LinComb$Combined)] <- paste0(LinComb$GEMVital[is.na(LinComb$Combined)])

LinComb$Combined <- as.double(LinComb$Combined)

#Plot and check the combination

Check <- gather(LinComb, GEMVital, Vital, Combined, key="Location", value="Lin")

Check <- Check %>%

as\_tibble %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date, Location) %>%

summarise(DayAvgL=mean(Lin))

Check$LineSize <- rep(0.5, nrow(Check))

Check$LineSize[Check$Location=="Combined"]<- 1.0

Check$CommonDate <- as.Date(paste0("2001-", format(Check$Date, "%j")),"%Y-%j")

Lin\_05\_11 <- filter(Check, year(Date)<=2011)

ggplot(data=Lin\_05\_11, mapping=aes(x=CommonDate, y=DayAvgL, color=Location, size=LineSize)) +

geom\_line() +

scale\_size(range=c(0.5,1.0), guide="none") +

facet\_grid(year(Lin\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Incoming Longwave - 2005-2011(check)")

Lin\_12\_18 <- filter(Check, year(Date)>=2012)

ggplot(data=Lin\_12\_18, mapping=aes(x=CommonDate, y=DayAvgL, color=Location, size=LineSize)) +

geom\_line() +

scale\_size(range=c(0.5,1.0), guide="none") +

facet\_grid(year(Lin\_12\_18$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Incoming Longwave - 2012-2018(check)")

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

LFinal <- select(LinComb, DateTime, Combined)

LWrite <- select(LinComb, Combined)

write\_excel\_csv(LFinal, "../MESH Model/Baker Creek Model Files/basin\_longwave.xlsx.csv")

write\_tsv(LWrite, "../MESH Model/Baker Creek Model Files/basin\_longwave.csv", col\_names=FALSE)

## Combine Precipitation Data

#####The purpose of this script is to pull in the processed driving data for the various stations around Baker Creek, NWT, plot the various variables, adjust the variables, and combine the data into 1 continuous set

#####Load Libraries

library(tidyverse)

library(dplyr)

library(lubridate)

library(colorspace)

#####Load the individual driving data files

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

load("./Vital.Rda")

load("./CAPA\_data\_0.5hr.Rda")

load("./Yknife\_precip.Rda")

CAPA\_data\_0.5hr <- filter(CAPA\_data\_0.5hr, year(DateTime)>= 2005) #Filter down GEM data since longer period

#####Shift the 2009 Vital data back by 6 days to match the Yellowknife and GEM data

p <- period(6, units="day")

VitalShift09 <- mutate(Vital, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

#####Combine all the data

##First, calculate daily totals

CAPA <- CAPA\_data\_0.5hr

CAPADaily <- CAPA

CAPADaily <- CAPADaily %>%

mutate(Date=date(DateTime)) %>%

mutate(Amount=Precip\_rate\*3600\*0.5) %>%

group\_by(Date, Station) %>%

summarise(DailyPrecip=sum(Amount)) %>%

ungroup()

p <- period(6, units="day")

VitalPrecip <- select(Vital, DateTime, Station, Precip\_rate)

VitalPrecip <- VitalPrecip %>%

mutate(DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime)) %>%

mutate(Precip\_rate=if\_else(year(DateTime)==2008,9999,Precip\_rate))

VitalPrecip[VitalPrecip==9999] <- NA

VitalPrecipDaily <- VitalPrecip

VitalPrecipDaily <- VitalPrecipDaily %>%

mutate(Date=date(DateTime)) %>%

mutate(Amount=Precip\_rate\*3600\*0.5) %>%

group\_by(Date, Station) %>%

summarise(DailyPrecip=sum(Amount)) %>%

ungroup()

YknifePrecipDaily <- select(Yknife\_precip, DateTime, total\_precip)

YknifePrecipDaily <- YknifePrecipDaily %>%

filter(year(DateTime) %in% 2005:2018) %>%

rename(DailyPrecip=total\_precip) %>%

mutate(Date=date(DateTime)) %>%

select(-DateTime)

YknifePrecipDaily$Station <- "YellowknifeA"

YknifePrecipDaily <- YknifePrecipDaily[,c(2,3,1)] #reorder columns

PlotPrecip <- rbind(CAPADaily, VitalPrecipDaily, YknifePrecipDaily)

PlotPrecip <- filter(PlotPrecip, Station %in% c("Vital", "CAPAVital"))

PlotPrecip$CommonDate <- as.Date(paste0("2001-", format(PlotPrecip$Date, "%j")),"%Y-%j")

# Precip\_05\_11 <- filter(PlotPrecip, year(Date) %in% 2005:2011)

# ggplot() +

# geom\_col(data=Precip\_05\_11, position="dodge", mapping=aes(x=CommonDate, y=DailyPrecip, color=Station), size=0.5) +

# facet\_grid(year(Precip\_05\_11$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Precip - 2005-2011 (shifted)")

# Precip\_05\_11<- filter(PlotPrecip, year(Date) %in% 2005:2011)

# ggplot() +

# geom\_line(data=Precip\_05\_11, mapping=aes(x=CommonDate, y=DailyPrecip, color=Station), size=0.5) +

# facet\_grid(year(Precip\_05\_11$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Precip - 2005-2011 (shifted)")

#

# Precip\_12\_18<- filter(PlotPrecip, year(Date) %in% 2012:2018)

# ggplot() +

# geom\_line(data=Precip\_12\_18, mapping=aes(x=CommonDate, y=DailyPrecip, color=Station), size=0.5) +

# facet\_grid(year(Precip\_12\_18$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Precip - 2012-2018")

# ggplot() +

# geom\_line(data=Precip\_05\_11, mapping=aes(x=CommonDate, y=DailyPrecip, color=Station), size=0.5) +

# facet\_grid(year(Precip\_05\_11$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Precip - 2005-2011 (shifted)")

PrecipSums <- PlotPrecip

PrecipSums <- PrecipSums %>%

mutate(Year=year(Date)) %>%

group\_by(Year, Station) %>%

summarise(AnnualPrecip = sum(DailyPrecip, na.rm=TRUE))

# ggplot()+

# geom\_col(data=PrecipSums, position="dodge", mapping=aes(x=Year, y=AnnualPrecip, fill=Station)) +

# labs(title="Annual Precip")

#

##### Combine the Precip dataset and write to file

###In order to combine, need the columns to be DateTime, Station1, Station2, etc.; make sure you haven't calculated Daily average values above (comment out the "group\_by" and "summarise" lines above)

DrivingPrecip <- rbind(CAPA, VitalPrecip)

DrivingPrecip <- arrange(DrivingPrecip, DateTime)

# Not using Yellowknife precip for driving data since only daily precip values available

PCAPAVital <- filter(DrivingPrecip, Station=="CAPAVital")

PCAPAVital <- PCAPAVital %>%

rename(PCAPAVital=Precip\_rate) %>%

select(DateTime, PCAPAVital)

PVital <- filter(DrivingPrecip, Station=="Vital")

PVital <- PVital %>%

rename(PVital=Precip\_rate) %>%

select(DateTime, PVital)

PrecipComb <- PCAPAVital

PrecipComb <- merge(PrecipComb, PVital, by="DateTime", all=TRUE)

PrecipComb$Combined <- NA

PrecipComb <- filter(PrecipComb, is.na(DateTime)==FALSE)

PrecipComb$Combined[is.na(PrecipComb$Combined)] <- paste0(PrecipComb$PVital[is.na(PrecipComb$Combined)])

PrecipComb$Combined <- as.double(PrecipComb$Combined)

PrecipComb$Combined[is.na(PrecipComb$Combined)] <- paste0(PrecipComb$PCAPAVital[is.na(PrecipComb$Combined)])

PrecipComb$Combined <- as.double(PrecipComb$Combined)

#Plot and check the combination

# Calculate annual totals and compare the Combined with all CAPA values and with YellowknifeA

PrecipCombDaily <- select(PrecipComb, DateTime, Combined)

PrecipCombDaily$Station <- "Combined"

PrecipCombDaily <- PrecipCombDaily %>%

mutate(Amount=Combined\*3600\*0.5) %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date, Station) %>%

summarise(DailyPrecip=sum(Amount)) %>%

ungroup()

Check <- rbind(PrecipCombDaily, CAPADaily, YknifePrecipDaily, VitalPrecipDaily)

Check <- Check %>%

mutate(Year=year(Date)) %>%

group\_by(Year, Station) %>%

summarise(AnnualPrecip=sum(DailyPrecip, na.rm=TRUE))

ggplot()+

geom\_col(data=Check, position="dodge", mapping=aes(x=Year, y=AnnualPrecip, fill=Station), color="gray27") +

labs(title="Annual Precip Comparison with Combined")

# scale\_fill\_discrete\_diverging(pal(3))

save(PrecipComb, file="C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/PrecipComb.Rda")

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

PrecipFinal <- select(PrecipComb, DateTime, Combined)

PrecipWrite <- select(PrecipComb, Combined)

write\_excel\_csv(PrecipFinal, "../MESH Model/Baker Creek Model Files/basin\_rain.xlsx.csv")

write\_tsv(PrecipWrite, "../MESH Model/Baker Creek Model Files/basin\_rain.csv", col\_names=FALSE)

##### Decide start date for the model (no recent rain events)

RecentRain <- filter(PrecipComb, year(DateTime)==2005 & month(DateTime) %in% c(09, 10))

RecentRain <- RecentRain %>%

select(DateTime, Combined) %>%

mutate(Amount=Combined\*3600\*0.5) %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date) %>%

summarise(DailyPrecip=sum(Amount)) %>%

ungroup()

## Combine Specific Humidity Data

#####The purpose of this script is to pull in the processed driving data for the various stations around Baker Creek, NWT, plot the various variables, adjust the variables, and combine the data into 1 continuous set

#####Load Libraries

library(tidyverse)

library(dplyr)

library(lubridate)

#####Load the individual driving data files

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

load("./Vital.Rda")

load("./Landing.Rda")

load("./Yknife\_HalfHr.Rda")

load("./GEM\_data.Rda")

GEM\_data <- filter(GEM\_data, year(DateTime)>= 2005) #Filter down GEM data since longer period

#####Shift the 2009 Vital data back by 6 days to match the Yellowknife and GEM data

p <- period(6, units="day")

VitalShift09 <- mutate(Vital, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

##### Scale the specific humidity to 40m by converting RH to q using T40m rather than T2m for Vital and Yellowknife stations only (won't use Landing data in the final combined)

load(file="./TempScaledCombined.Rda")

VitalShiftq <- merge(VitalShift09, TComb, by="DateTime", all=TRUE)

VitalShiftq <- VitalShiftq %>%

mutate(T\_40m=Vital2.8) %>%

select(-c("GEMVital", "YellowknifeA", "Vital2.8", "Vital4.4", "Combined")) %>%

mutate(ea\_40m=RH\_2m/100\*10^((0.7859+0.03477\*T\_40m)/(1.0+0.00412\*T\_40m)+2))%>%

mutate(q\_40m=0.622\*ea\_40m/(AirP\_Pa-0.378\*ea\_40m)) %>%

filter(is.na(DateTime)==FALSE, is.na(Station)==FALSE) %>%

select(-ea\_40m)

### Didn't make much difference, so use the observed q in the model

#####Combine all the data

##Use this first section to choose whether or not to use the shifted version of the Vital data, or the original

DrivingShift <- rbind(VitalShift09, Landing, Yknife, GEM\_data)

DrivingOrig <- rbind(Vital, Landing, Yknife, GEM\_data)

DrivingShiftq <- rbind(VitalShiftq, Landing, Yknife, GEM\_data)

##### Explore the Specific Humidity data

q1 <- select(DrivingShift, DateTime, Station, q\_1.1m, q\_2m, q\_4.4m, q\_40m)

q <- q1

q <- q %>% gather(q\_1.1m, q\_2m, q\_4.4m, q\_40m, key="Height", value="q")%>%

arrange(DateTime) %>%

filter(is.na(q)==FALSE)

q$Height <- str\_sub(q$Height, start=3)

q <- q %>%

filter(!Station %in% c("GEMLanding", "GEMYellowknifeA", "Landing")) %>%

unite(Station, Height, col="Station", sep="\_") %>%

mutate(Date=date(DateTime))

qplot <- q

qplot <- qplot %>%

group\_by(Date, Station) %>%

summarise(DailyAvgq=mean(q))

#Plot and compare the data

qplot$CommonDate <- as.Date(paste0("2000-", format(qplot$Date, "%j")),"%Y-%j")

q\_05\_11 <- filter(qplot, year(Date)>=2005 & year(Date)<=2011)

ggplot() +

geom\_line(data=q\_05\_11, mapping=aes(x=CommonDate, y=DailyAvgq, color=Station), size=0.5) +

facet\_grid(year(q\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Specific Humidity - 2005-2011(shifted)")

#

q\_12\_18 <- filter(qplot, year(Date)>=2012 & year(Date)<=2018)

ggplot() +

geom\_line(data=q\_12\_18, mapping=aes(x=CommonDate, y=DailyAvgq, color=Station), size=0.5) +

facet\_grid(year(q\_12\_18$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Specific Humidity - 2012-2018(shifted)")

# Temp\_2009 <- Temp1 %>% filter(year(Date)==2009)

#

# ggplot() +

# geom\_line(data=Temp\_2009, mapping=aes(x=CommonDate, y=DailyAvgT, color=Station), size=0.5) +

# facet\_grid(year(Temp\_2009$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Average Temperature - 2009(shifted)")

##### Combine the AirP dataset and write to file

###In order to combine, need the columns to be DateTime, Station1, Station2, etc.; make sure you haven't calculated Daily average values above (comment out the "group\_by" and "summarise" lines above)

qVital <- filter(q, Station=="Vital\_4.4m")

qVital <- qVital %>%

rename(Vital=q) %>%

select(DateTime, Vital)

qGEMVital <- filter(q, Station=="GEMVital\_40m")

qGEMVital <- qGEMVital %>%

rename(GEMVital=q) %>%

select(DateTime, GEMVital)

qComb <- qGEMVital

qComb <- merge(qComb, qVital, by="DateTime", all=TRUE)

qComb$Combined <- NA

qComb <- filter(qComb, is.na(DateTime)==FALSE)

qComb$Combined[is.na(qComb$Combined)] <- paste0(qComb$Vital[is.na(qComb$Combined)])

qComb$Combined <- as.double(qComb$Combined)

qComb$Combined[is.na(qComb$Combined)] <- paste0(qComb$GEMVital[is.na(qComb$Combined)])

qComb$Combined <- as.double(qComb$Combined)

#Plot and check the combination

Check <- gather(qComb, GEMVital, Vital, Combined, key="Location", value="q")

Check <- Check %>%

as\_tibble %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date, Location) %>%

summarise(DayAvgq=mean(q))

Check$LineSize <- rep(0.5, nrow(Check))

Check$LineSize[Check$Location=="Combined"]<- 1.0

Check$CommonDate <- as.Date(paste0("2001-", format(Check$Date, "%j")),"%Y-%j")

q\_05\_11 <- filter(Check, year(Date) %in% 2005:2011)

ggplot(data=q\_05\_11, mapping=aes(x=CommonDate, y=DayAvgq, color=Location, size=LineSize)) +

geom\_line() +

scale\_size(range=c(0.5,1.0), guide="none") +

facet\_grid(year(q\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Specific Humidity - 2005-2011 (combined)")

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

qFinal <- select(qComb, DateTime, Combined)

qWrite <- select(qComb, Combined)

write\_excel\_csv(qFinal, "../MESH Model/Baker Creek Model Files/basin\_humidity.xlsx.csv")

write\_tsv(qWrite, "../MESH Model/Baker Creek Model Files/basin\_humidity.csv", col\_names=FALSE)

## Combine Temperature Data

#####The purpose of this script is to pull in the processed driving data for the various stations around Baker Creek, NWT, plot the various variables, adjust the variables, and combine the data into 1 continuous set

#####Load Libraries

library(tidyverse)

library(dplyr)

library(lubridate)

#####Load the individual driving data files

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

load("./Vital.Rda")

load("./Landing.Rda")

load("./Yknife\_HalfHr.Rda")

load("./GEM\_data.Rda")

GEM\_data <- filter(GEM\_data, year(DateTime)>= 2005) #Filter down GEM data since longer period

#####Check that the column names are consistent

# ColNames\_Vital <- colnames(Vital)

# ColNames\_Landing <- colnames(Landing)

# ColNames\_Yknife <- colnames(Yknife)

# ColNames\_GEM <- colnames(GEM\_data)

#

# Check\_Colnames <- data.frame(ColNames\_Vital, ColNames\_Landing, ColNames\_Yknife, ColNames\_GEM)

#####Shift the 2009 Vital data back by 6 days to match the Yellowknife and GEM data

p <- period(6, units="day")

VitalShift09 <- mutate(Vital, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

#####Check that "Vital" matches the original data where the first point: T\_2m=0.22843, T\_4.4m=-0.04837, u\_4.4m=5.247

# Vital09Only <- filter(Vital, year(DateTime)==2009)

# VitalShift09Only <- filter(VitalShift09, year(DateTime)==2009)

# head(Vital09Only)

# head(VitalShift09Only)

#VitalShift09 <- filter(VitalShift09, year(DateTime)==2009)

#VitalShift09$DateTimeShift <- ymd\_hms(VitalShift09$DateTimeShift)

#####Combine all the data

##Use this first section to choose whether or not to use the shifted version of the Vital data, or the original

DrivingShift <- rbind(VitalShift09, Landing, Yknife, GEM\_data)

DrivingOrig <- rbind(Vital, Landing, Yknife, GEM\_data)

#####Count the number of observations at each station for each variable between Sept. 1 2005 and Oct. 1 2018 (inclusive); I used this when trying to check what % of the entire dataset each station makes up for each variable; didn't actually complete the calculation

# Driving\_subset <- filter(Driving, DateTime >= "2005-09-01 00:00:00" & DateTime <= "2018-10-01 23:30:00")

# Driving\_subset <- Driving\_subset %>%

# select(Station, T\_40m) %>%

# group\_by(Station) %>%

# summarise\_each(~n())

# Driving\_subset <- Driving\_subset %>%

# gather(ColNames\_Vital[3:25], key="Obs", value="Value") %>%

# group\_by(Station, Obs) %>%

# filter(is.na(Value)==FALSE) %>%

# summarise\_at("Value", ~n(), na.rm=TRUE)

##### Explore the temperature data

# Temp <- select(Driving, DateTime, Station, T\_1.1m, T\_2m, T\_4.4m, T\_40m)

# Temp1 <- Temp

# Temp1 <- Temp1 %>% gather(T\_1.1m, T\_2m, T\_4.4m, T\_40m, key="Height", value="T\_degC")%>%

# arrange(DateTime) %>%

# filter(is.na(T\_degC)==FALSE) %>%

# filter(!Station %in% c("GEMLanding", "GEMYellowknifeA", "Landing")) %>%

# mutate(Year = year(DateTime), Date=date(DateTime), Month=month(DateTime), Day=day(DateTime), Time=paste(hour(DateTime),minute(DateTime),second(DateTime),sep=":")) %>%

# group\_by(Date, Station, Height) %>%

# summarise(DailyAvgT=mean(T\_degC))

# Temp1$Height <- str\_sub(Temp1$Height, start=3)

# # Temp <- group\_by("Station")

#

# Temp1$CommonDate <- as.Date(paste0("2000-", format(Temp1$Date, "%j")),"%Y-%j")

#

# Temp\_05\_11 <- filter(Temp1, year(Date)>=2005 & year(Date)<=2011)

# ggplot() +

# geom\_line(data=Temp\_05\_11, mapping=aes(x=CommonDate, y=DailyAvgT, color=Station), size=0.5) +

# facet\_grid(year(Temp\_05\_11$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Average Temperature - 2005-2011(shifted)")

#

# Temp\_12\_18 <- filter(Temp1, year(Date)>=2012 & year(Date)<=2018)

# ggplot() +

# geom\_line(data=Temp\_12\_18, mapping=aes(x=CommonDate, y=DailyAvgT, color=Station), size=0.5) +

# facet\_grid(year(Temp\_12\_18$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Average Temperature - 2012-2018")

# #Note: 2009 Vital data appears to be shifted forward by a few days compared to the other observations

#

# Temp\_2009 <- Temp1 %>% filter(year(Date)==2009)

#

# ggplot() +

# geom\_line(data=Temp\_2009, mapping=aes(x=CommonDate, y=DailyAvgT, color=Station), size=0.5) +

# facet\_grid(year(Temp\_2009$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Daily Average Temperature - 2009(shifted)")

#####Calculate the adiabatic lapse rate for the period where there is GEM temp. data at both 2m and 40m (from Oct. 1 2011 onward)

### Could look at this code: https://rdrr.io/github/ilyamaclean/microclima/man/lapserate.html

# View(GEM\_data)

GEMLapse <- GEM\_data

GEMLapse <- GEMLapse %>%

select(DateTime, Station, T\_2m, T\_40m) %>%

filter(Station=="GEMVital", is.na(T\_2m)==FALSE) %>%

mutate(LapseRate=(T\_40m-T\_2m)/-0.038) %>%

mutate(Date=date(DateTime))

GEMLapse$CommonDate <- as.Date(paste0("2001-", format(GEMLapse$Date, "%j")),"%Y-%j")

# ggplot() +

# geom\_point(data=GEMLapse, mapping=aes(x=CommonDate, y=LapseRate, color=Station), size=0.5) +

# facet\_grid(year(GEMLapse$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="GEM Lapse Rate - 2011-2018")

#Calculate min, mean, and max lapse rates

min(GEMLapse$LapseRate)

mean(GEMLapse$LapseRate)

max(GEMLapse$LapseRate)

#####Adjust the Vital and Yellowknife temperature data up to 40m

# DrivingLapsed <- rbind(VitalShift09, Landing, Yknife, GEM\_data)

# LR <- mean(GEMLapse$LapseRate)

LR <- 6.5

Temp2 <- select(DrivingShift, DateTime, Station, T\_1.1m, T\_2m, T\_4.4m, T\_40m)

Temp2 <- arrange(Temp2, DateTime)

#First, compare the T\_40m values between the stations and the GEM 40m data

Vital2.8 <- select(Temp2, DateTime, Station, T\_2m)

Vital2.8 <- Vital2.8 %>%

filter(Station=="Vital") %>%

mutate(Vital2.8 = -LR\*(40-2.8)/1000+T\_2m) %>%

select(-c(T\_2m, Station))

Vital4.4 <- select(Temp2, DateTime, Station, T\_4.4m)

Vital4.4 <- Vital4.4 %>%

filter(Station=="Vital") %>%

mutate(Vital4.4 = -LR\*(40-4.4)/1000+T\_4.4m) %>%

select(-c(T\_4.4m, Station))

YknifeLapsed <- select(Temp2, DateTime, Station, T\_2m)

YknifeLapsed <- YknifeLapsed %>%

filter(Station=="YellowknifeA") %>%

mutate(YellowknifeA = -LR\*(40-2)/1000+T\_2m) %>%

select(-c(T\_2m, Station))

GEM40 <- select(Temp2, DateTime, Station, T\_40m)

GEM40 <- GEM40 %>%

filter(Station=="GEMVital") %>%

filter(year(DateTime)>=2005) %>%

rename(GEMVital=T\_40m) %>%

select(-Station)

##### Explore and plot the lapsed temp values from various stations

# TempLapsed <- rbind(Vital1.1, Vital4.4, YknifeLapsed, GEM40)

#

# TempLapsed <- arrange(TempLapsed, DateTime)

#

# TempLapsed <- spread(TempLapsed, Station, T\_40m)

#

# TempLapsed$Date <- date(TempLapsed$DateTime)

# TempLapsed <- TempLapsed %>%

# group\_by(Date, Station) %>%

# summarise(DailyAvgT=mean(T\_40m))

#

# TempLapsed$CommonDate <- as.Date(paste0("2001-", format(TempLapsed$Date, "%j")),"%Y-%j")

# # TempLapsed <- filter(TempLapsed, is.na(CommonDate)==FALSE)

#

# TLapse0511 <- filter(TempLapsed, year(Date)>=2005 & year(Date) <= 2011)

# ggplot() +

# geom\_line(data=TLapse0511, mapping=aes(x=CommonDate, y=DailyAvgT, color=Station), size=0.5) +

# facet\_grid(year(TLapse0511$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Temp at 40m 2005-2011 (Lapse Rate = -24.44 degC/km))")

#

# TLapse05 <- filter(TempLapsed, year(Date)==2005)

#

# ggplot() +

# geom\_line(data=TLapse05, mapping=aes(x=CommonDate, y=DailyAvgT, color=Station), size=0.5) +

# facet\_grid(year(TLapse05$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

# labs(title="Temp at 40m (Lapse Rate = -24.44 degC/km)")

##### Combine the temperature data

TComb <- GEM40

TComb <- merge(TComb, YknifeLapsed, by="DateTime", all=TRUE)

TComb <- merge(TComb, Vital2.8, by="DateTime", all=TRUE)

TComb <- merge(TComb, Vital4.4, by="DateTime", all=TRUE)

TComb$Combined <- NA

TComb <- filter(TComb, is.na(DateTime)==FALSE)

TComb$Combined[is.na(TComb$Combined)] <- paste0(TComb$Vital4.4[is.na(TComb$Combined)])

TComb$Combined <- as.double(TComb$Combined)

TComb$Combined[is.na(TComb$Combined)] <- paste0(TComb$Vital2.8[is.na(TComb$Combined)])

TComb$Combined <- as.double(TComb$Combined)

TComb$Combined[is.na(TComb$Combined)] <- paste0(TComb$YellowknifeA[is.na(TComb$Combined)])

TComb$Combined <- as.double(TComb$Combined)

TComb$Combined[is.na(TComb$Combined)] <- paste0(TComb$GEMVital[is.na(TComb$Combined)])

TComb$Combined <- as.double(TComb$Combined)

#Convert to degrees kelvin

TComb <- TComb %>%

mutate(CombinedK=(Combined+273.15))

save(TComb, file="TempScaledCombined.Rda")

TFinal <- select(TComb, DateTime, CombinedK)

TWrite <- select(TComb, CombinedK)

write\_excel\_csv(TFinal, "../MESH Model/Baker Creek Model Files/basin\_temperature.xlsx.csv")

write\_tsv(TWrite, "../MESH Model/Baker Creek Model Files/basin\_temperature.csv", col\_names=FALSE)

### Plotting the results of the combined temperature data

load("./TempScaledCombined.Rda")

TCombPlot <- TComb

TCombPlot$Date <- date(TCombPlot$DateTime)

TCombPlot <- TCombPlot %>%

gather(Combined, GEMVital, key="Station", value="T\_40m") %>%

group\_by(Date, Station) %>%

summarise(DailyAvgT=mean(T\_40m))

TCombPlot$CommonDate <- as.Date(paste0("2001-", format(TCombPlot$Date, "%j")),"%Y-%j")

TCombPlot\_05\_11 <- filter(TCombPlot, year(Date) %in% 2005:2011)

ggplot(data=TCombPlot\_05\_11) +

geom\_line(mapping=aes(x=CommonDate, y=DailyAvgT, colour=Station), size=0.5) +

# geom\_line(mapping=aes(x=CommonDate, y=GEM40), size=0.5) +

facet\_grid(year(TCombPlot\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Temperature Data")

##### Look at the temperature data around where the model will start (2005)

TStart <- filter(TComb, year(DateTime)==2005 & month(DateTime) %in% c(09, 10))

TStart <- TStart %>%

select(DateTime, Combined) %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date) %>%

summarise(DailyAvgT=mean(Combined)) %>%

ungroup()

##### Get the daily average temperature on Sept. 14 for use as the starting TCAN in the model

Tstart <- filter(TComb, date(DateTime)=='2005-09-14')

TCAN <- summarise(Tstart, Tavg=mean(Combined))

## Combine Wind Speed Data

#####The purpose of this script is to pull in the processed driving data for the various stations around Baker Creek, NWT, plot the various variables, adjust the variables, and combine the data into 1 continuous set

#####Load Libraries

library(tidyverse)

library(dplyr)

library(lubridate)

#####Load the individual driving data files

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

load("./Vital.Rda")

load("./Landing.Rda")

load("./Yknife\_HalfHr.Rda")

load("./GEM\_data.Rda")

GEM\_data <- filter(GEM\_data, year(DateTime)>= 2005) #Filter down GEM data since longer period

#####Shift the 2009 Vital data back by 6 days to match the Yellowknife and GEM data

p <- period(6, units="day")

VitalShift09 <- mutate(Vital, DateTime=if\_else(year(DateTime)==2009,DateTime-p,DateTime))

#####Combine all the data

##Use this first section to choose whether or not to use the shifted version of the Vital data, or the original

DrivingShift <- rbind(VitalShift09, Landing, Yknife, GEM\_data)

DrivingOrig <- rbind(Vital, Landing, Yknife, GEM\_data)

##### Explore the Wind data

u1 <- select(DrivingShift, DateTime, Station, u\_4.4m, u\_10m, u\_40m)

u <- u1

u <- u %>%

gather(u\_4.4m, u\_10m, u\_40m, key="Height", value="Wind")%>%

arrange(DateTime) %>%

filter(is.na(Wind)==FALSE)

u$Height <- str\_sub(u$Height, start=3)

u <- u %>%

filter(!Station %in% c("GEMLanding", "GEMYellowknifeA", "Landing")) %>%

unite(Station, Height, col="Station", sep="\_") %>%

mutate(Date=date(DateTime))

##### Scale wind at Vital station (zm=4.4m) and YellowknifeA (10m) up to 40m height using equations 3.27 and 3.30a from Dingman

### u\_star=k\*u(zm)/ln((zm-zd)/z0) where k=0.4, zd=0.7\*zveg, z0=0.1\*zveg

### u(z) = 1/k\*u\_star\*ln((z-zd)/z0)

zveg <- 2

zd <- 0.7\*zveg

z0 <- 0.1\*zveg

u <- u %>%

mutate(u\_star=if\_else(grepl("Vital\_4.4m",u$Station),0.4\*Wind/log((4.4-zd)/z0),if\_else(grepl("GEMVital\_10m",u$Station),0.4\*Wind/log((10-zd)/z0),if\_else(grepl("YellowknifeA\_10m",u$Station),0.4\*Wind/log((10-zd)/z0),9999)))) %>%

mutate(u\_40m=if\_else(grepl("Vital\_4.4m",u$Station),1/0.4\*u\_star\*log((40-zd)/z0),if\_else(grepl("GEMVital\_10m",u$Station),1/0.4\*u\_star\*log((40-zd)/z0),if\_else(grepl("YellowknifeA\_10m",u$Station),1/0.4\*u\_star\*log((40-zd)/z0),Wind))))

# group\_by(Date, Station) %>%

# summarise(DailyAvgu=mean(u\_40m))

u$CommonDate <- as.Date(paste0("2001-", format(u$Date, "%j")),"%Y-%j")

u\_05\_11 <- filter(u, year(Date)>=2005 & year(Date)<=2011)

ggplot() +

geom\_line(data=u\_05\_11, mapping=aes(x=CommonDate, y=DailyAvgu, color=Station), size=0.5) +

facet\_grid(year(u\_05\_11$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Wind Speed - 2005-2011(scaled)")

u\_12\_18 <- filter(u, year(Date)>=2012 & year(Date)<=2018)

ggplot() +

geom\_line(data=u\_12\_18, mapping=aes(x=CommonDate, y=DailyAvgu, color=Station), size=0.5) +

facet\_grid(year(u\_12\_18$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Wind Speed - 2012-2018(scaled)")

u\_09 <- filter(u, year(Date)==2009)

ggplot() +

geom\_line(data=u\_09, mapping=aes(x=DateTime, y=u\_40m, color=Station), size=0.5) +

facet\_grid(year(u\_09$Date) ~ .) +

# scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Daily Average Wind Speed - 2009(scaled)")

##### Combine the Lin dataset and write to file

###In order to combine, need the columns to be DateTime, Station1, Station2, etc.; make sure you haven't calculated Daily average values above (comment out the "group\_by" and "summarise" lines above)

uVital <- filter(u, Station=="Vital\_4.4m")

uVital <- uVital %>%

rename(Vital=u\_40m) %>%

select(DateTime, Vital)

uGEMVital <- filter(u, Station=="GEMVital\_40m")

uGEMVital <- uGEMVital %>%

rename(GEMVital=u\_40m) %>%

select(DateTime, GEMVital)

uComb <- uGEMVital

uComb <- merge(uComb, uVital, by="DateTime", all=TRUE)

uComb$Combined <- NA

uComb <- filter(uComb, is.na(DateTime)==FALSE)

uComb$Combined[is.na(uComb$Combined)] <- paste0(uComb$Vital[is.na(uComb$Combined)])

uComb$Combined <- as.double(uComb$Combined)

uComb$Combined[is.na(uComb$Combined)] <- paste0(uComb$GEMVital[is.na(uComb$Combined)])

uComb$Combined <- as.double(uComb$Combined)

#Plot and check the combination

Check <- gather(uComb, GEMVital, Vital, Combined, key="Location", value="u\_40m")

Check <- Check %>%

as\_tibble %>%

mutate(Date=date(DateTime)) %>%

group\_by(Date, Location) %>%

summarise(DayAvgu=mean(u\_40m))

Check$LineSize <- rep(0.5, nrow(Check))

Check$LineSize[Check$Location=="Combined"]<- 1.0

Check$CommonDate <- as.Date(paste0("2001-", format(Check$Date, "%j")),"%Y-%j")

u\_09 <- filter(Check, year(Date)==2009)

ggplot(data=u\_09, mapping=aes(x=CommonDate, y=DayAvgu, color=Location, size=LineSize)) +

geom\_line() +

scale\_size(range=c(0.5,1.0), guide="none") +

facet\_grid(year(u\_09$Date) ~ .) +

scale\_x\_date(labels=function(x) format(x,"%d-%b")) +

labs(title="Wind Speed - 2009")

setwd("C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/")

uFinal <- select(uComb, DateTime, Combined)

uWrite <- select(uComb, Combined)

write\_excel\_csv(uFinal, "../MESH Model/Baker Creek Model Files/basin\_wind.xlsx.csv")

write\_tsv(uWrite, "../MESH Model/Baker Creek Model Files/basin\_wind.csv", col\_names=FALSE)

## Soil Temperature and Moisture Observations

#This scritp is used to produce tables of minimum, mean, and maximum soil temperature and moisture observations for the dates specified at surface and 25cm depth in the Baker Creek watershed, NWT

#Data source: Spence, C., & Hedstrom, N. (2018). Hydrometeorological data from Baker Creek Research Watershed, Northwest Territories, Canada. Earth System Science Data, 10(4), 1753-1767.

library(tidyverse)

library(lubridate)

library(dplyr)

#Load Soil Temperature and Soil Moisture Data

ColNames <- c("DateTime", "wb\_surf", "wb\_25cm", "av\_surf", "av\_25cm", "lp\_surf", "lp\_25cm", "tp\_surf", "tp\_25cm", "cv\_surf", "cv\_25cm")

ColNames\_Rock <- c("DateTime", "T\_surf", "T\_10cm", "T\_20cm", "T\_30cm", "T\_46cm")

SoilT\_load <- read\_csv('../Data/ESSD Baker Creek Data/GroundTemperatureData/soil temperature time series v1.csv',col\_names=ColNames, skip=1)

SoilM\_load <- read\_csv('../Data/ESSD Baker Creek Data/SoilMoistureData/soil moisture time series v1.csv',col\_names=ColNames, skip=1)

RockT\_load <- read\_csv('../Data/ESSD Baker Creek Data/GroundTemperatureData/exposed rock temperature time series v1.csv',col\_names=ColNames\_Rock, skip=1)

#Change date formate to dttm

SoilT\_load$DateTime <- dmy\_hm(SoilT\_load$DateTime)

SoilM\_load$DateTime <- dmy\_hm(SoilM\_load$DateTime)

RockT\_load$DateTime <- dmy(RockT\_load$DateTime)

head(SoilT\_load)

head(SoilM\_load)

#Remove the DateTime name from column names for use in the "gather" function", then gather the data, and add a Landuse column

ColNames <- ColNames <- c("wb\_surf", "wb\_25cm", "av\_surf", "av\_25cm", "lp\_surf", "lp\_25cm", "tp\_surf", "tp\_25cm", "cv\_surf", "cv\_25cm")

SoilT <- SoilT\_load #To preserve the original loaded values

SoilM <- SoilM\_load #To preserve the original loaded values

RockT <- RockT\_load

SoilT <- SoilT %>% #This block for Soil Temp

gather(ColNames, key="Station", value="SoilTemp") %>%

separate(Station, c("Location", "T\_Depth")) %>%

filter(!SoilTemp==9999) %>%

mutate(Landuse=ifelse(Location=="lp"|Location=="cv","Hillslope",ifelse(Location=="wb"|Location=="tp","Peatland","Wetland")))

# unique(SoilT$SoilTemp)

# unique(SoilT$T\_Depth)

# unique(SoilT$Location)

SoilM <- SoilM %>% #This block for Soil Moisture

gather(ColNames, key="Station", value="SoilMoist") %>%

separate(Station, c("Location", "M\_Depth")) %>%

filter(!SoilMoist==9999)%>%

mutate(Landuse=ifelse(Location=="lp"|Location=="cv","Hillslope",ifelse(Location=="wb"|Location=="tp","Peatland","Wetland")))

# unique(SoilM$SoilMoist)

# unique(SoilM$M\_Depth)

# unique(SoilM$Location)

#Calculate the min, max, and average SOIL TEMP on Sept 15 at 25cm and surface for all locations and overall

Sept15\_avgT <- filter(SoilT, month(DateTime)==9, day(DateTime)==15)

Sept15\_avgT <- Sept15\_avgT %>%

spread(key=T\_Depth, value=SoilTemp,sep='\_')

Total <- c("Overall",min(Sept15\_avgT$T\_Depth\_25cm,na.rm=TRUE),min(Sept15\_avgT$T\_Depth\_surf,na.rm=TRUE),mean(Sept15\_avgT$T\_Depth\_25cm,na.rm=TRUE),mean(Sept15\_avgT$T\_Depth\_surf,na.rm=TRUE),max(Sept15\_avgT$T\_Depth\_25cm,na.rm=TRUE),max(Sept15\_avgT$T\_Depth\_surf,na.rm=TRUE))

Sept15\_avgT <- Sept15\_avgT %>%

select(-DateTime) %>%

#group\_by(Location) %>%

group\_by(Landuse)%>%

summarise\_each(funs(min(.,na.rm=TRUE), mean(.,na.rm=TRUE), max(.,na.rm=TRUE)), T\_Depth\_25cm, T\_Depth\_surf)

Sept15\_avgT <- rbind(Sept15\_avgT,Total)

#Calculate the min, max, and average SOIL TEMP on Oct. 1 at 25cm and surface for all locations and overall

Oct1\_avgT <- filter(SoilT, month(DateTime)==10, day(DateTime)==1)

Oct1\_avgT <- Oct1\_avgT %>%

spread(key=T\_Depth, value=SoilTemp,sep='\_')

tp <- filter(Oct1\_avgT, Location=='tp')

Total <- c("Overall",min(Oct1\_avgT$T\_Depth\_25cm,na.rm=TRUE),min(Oct1\_avgT$T\_Depth\_surf,na.rm=TRUE),mean(Oct1\_avgT$T\_Depth\_25cm,na.rm=TRUE),mean(Oct1\_avgT$T\_Depth\_surf,na.rm=TRUE),max(Oct1\_avgT$T\_Depth\_25cm,na.rm=TRUE),max(Oct1\_avgT$T\_Depth\_surf,na.rm=TRUE))

Oct1\_avgT <- Oct1\_avgT %>%

# select(-DateTime) %>%

#group\_by(Location) %>%

group\_by(Landuse) %>%

summarise\_each(funs(min(.,na.rm=TRUE), mean(.,na.rm=TRUE), max(.,na.rm=TRUE)), T\_Depth\_25cm, T\_Depth\_surf)

Oct1\_avgT <- rbind(Oct1\_avgT,Total)

#Note: there are no observations for location 'tp' at 25cm, hence the Inf/NaN values

#Calculate the min, max, and average SOIL MOISTURE on Sept 15 at 25cm and surf for all locations and overall

Sept15\_avgM <- filter(SoilM, month(DateTime)==9, day(DateTime)==15)

Sept15\_avgM <- Sept15\_avgM %>%

spread(key=M\_Depth, value=SoilMoist,sep='\_')

Total <- c("Overall",min(Sept15\_avgM$M\_Depth\_25cm,na.rm=TRUE),min(Sept15\_avgM$M\_Depth\_surf,na.rm=TRUE),mean(Sept15\_avgM$M\_Depth\_25cm,na.rm=TRUE),mean(Sept15\_avgM$M\_Depth\_surf,na.rm=TRUE),max(Sept15\_avgM$M\_Depth\_25cm,na.rm=TRUE),max(Sept15\_avgM$M\_Depth\_surf,na.rm=TRUE))

Sept15\_avgM <- Sept15\_avgM %>%

select(-DateTime) %>%

#group\_by(Location) %>%

group\_by(Landuse) %>%

summarise\_each(funs(min(.,na.rm=TRUE), mean(.,na.rm=TRUE), max(.,na.rm=TRUE)), M\_Depth\_25cm, M\_Depth\_surf)

Sept15\_avgM <- rbind(Sept15\_avgM,Total)

#Calculate the min, max, and average SOIL MOISTURE on Oct. 1 at 25cm and surf for all locations and overall

Oct1\_avgM <- filter(SoilM, month(DateTime)==10, day(DateTime)==1)

Oct1\_avgM <- Oct1\_avgM %>%

spread(key=M\_Depth, value=SoilMoist,sep='\_')

tp <- filter(Oct1\_avgM, Location=='tp')

Total <- c("Overall",min(Oct1\_avgM$M\_Depth\_25cm,na.rm=TRUE),min(Oct1\_avgM$M\_Depth\_surf,na.rm=TRUE),mean(Oct1\_avgM$M\_Depth\_25cm,na.rm=TRUE),mean(Oct1\_avgM$M\_Depth\_surf,na.rm=TRUE),max(Oct1\_avgM$M\_Depth\_25cm,na.rm=TRUE),max(Oct1\_avgM$M\_Depth\_surf,na.rm=TRUE))

Oct1\_avgM <- Oct1\_avgM %>%

# select(-DateTime) %>%

#group\_by(Location) %>%

group\_by(Landuse) %>%

summarise\_each(funs(min(.,na.rm=TRUE), mean(.,na.rm=TRUE), max(.,na.rm=TRUE)), M\_Depth\_25cm, M\_Depth\_surf)

Oct1\_avgM <- rbind(Oct1\_avgM,Total)

#Note: there are no observations for location 'tp' at 25cm, hence the Inf/NaN values

#Exposed Rock Calculations

RockT <- RockT\_load

ColNames\_Rock <- c("T\_surf", "T\_10cm", "T\_20cm", "T\_30cm", "T\_46cm")

RockT <- RockT %>%

gather(ColNames\_Rock, key="Depth", value="RockTemp") %>%

filter(!RockTemp==9999) %>%

spread(key=Depth, value=RockTemp)

Sept15\_avgRockT <- filter(RockT, month(DateTime)==9, day(DateTime)==15)

Sept15\_avgRockT <- Sept15\_avgRockT %>%

summarise\_each(funs(min(.,na.rm=TRUE), mean(.,na.rm=TRUE), max(.,na.rm=TRUE)), T\_surf, T\_20cm, T\_30cm) #%>%

ColOrder <- c("T\_surf\_min", "T\_surf\_mean", "T\_surf\_max", "T\_20cm\_min", "T\_20cm\_mean","T\_20cm\_max", "T\_25cm\_min", "T\_25cm\_mean","T\_25cm\_max","T\_30cm\_min", "T\_30cm\_mean","T\_30cm\_max")

Sept15\_avgRockT <- Sept15\_avgRockT %>% mutate(T\_25cm\_min=mean(T\_20cm\_min, T\_30cm\_min), T\_25cm\_mean=mean(T\_20cm\_mean, T\_30cm\_mean), T\_25cm\_max=mean(T\_20cm\_max, T\_30cm\_max))

Sept15\_avgRockT <- Sept15\_avgRockT[,ColOrder]

#October 1 calc

Oct1\_avgRockT <- filter(RockT, month(DateTime)==10, day(DateTime)==1)

Oct1\_avgRockT <- Oct1\_avgRockT %>%

summarise\_each(funs(min(.,na.rm=TRUE), mean(.,na.rm=TRUE), max(.,na.rm=TRUE)), T\_surf, T\_20cm, T\_30cm) #%>%

ColOrder <- c("T\_surf\_min", "T\_surf\_mean", "T\_surf\_max", "T\_20cm\_min", "T\_20cm\_mean","T\_20cm\_max", "T\_25cm\_min", "T\_25cm\_mean","T\_25cm\_max","T\_30cm\_min", "T\_30cm\_mean","T\_30cm\_max")

Oct1\_avgRockT <- Oct1\_avgRockT %>% mutate(T\_25cm\_min=mean(T\_20cm\_min, T\_30cm\_min), T\_25cm\_mean=mean(T\_20cm\_mean, T\_30cm\_mean), T\_25cm\_max=mean(T\_20cm\_max, T\_30cm\_max))

Oct1\_avgRockT <- Oct1\_avgRockT[,ColOrder]

#Calculate Aggregate Soil Temp and Moisture weighted by landcover fraction

#First, for soil Temperature (include Bedrock)

Landcover\_frac <- tibble("Type"=c("Bedrock", "Hillslope", "Peatland", "Wetland"), "Fraction"=c(0.3992, 0.2078+0.0075, 0.1007, 0.0585))

sum\_frac <- sum(Landcover\_frac$Fraction)

sum\_frac

Landcover\_frac$Fraction <- Landcover\_frac$Fraction/sum\_frac

Landcover\_frac$Oct1\_surf\_mean <- NA

Landcover\_frac$Oct1\_25cm\_mean <- NA

Landcover\_frac[1,3] <- Oct1\_avgRockT$T\_surf\_mean

Landcover\_frac[1,4] <- Oct1\_avgRockT$T\_25cm\_mean

Oct1\_T <- filter(Oct1\_avgT, !Landuse=="Overall")

Landcover\_frac[2:4,3] <- (Oct1\_T$T\_Depth\_surf\_mean)

Landcover\_frac[2:4,4] <- Oct1\_T$T\_Depth\_25cm\_mean

Landcover\_frac <- mutate(Landcover\_frac, Frac\_T\_surf=Fraction\*Oct1\_surf\_mean, Frac\_T\_25cm=Fraction\*Oct1\_25cm\_mean)

Landcover\_frac$Oct1\_surf\_mean <- as.numeric(Landcover\_frac$Oct1\_surf\_mean)

Landcover\_frac$Oct1\_25cm\_mean <- as.numeric(Landcover\_frac$Oct1\_25cm\_mean)

head(Landcover\_frac)

Total\_surf\_T <- sum(Landcover\_frac$Fraction\*Landcover\_frac$Oct1\_surf\_mean)

Total\_25cm\_T <- sum(Landcover\_frac$Fraction\*Landcover\_frac$Oct1\_25cm\_mean)

Total\_surf\_T

Total\_25cm\_T

#Calculate the weighted average soil moisture on October 1 for the whole watershed

Landcover\_frac <- select(Landcover\_frac, Type, Fraction)

Landcover\_frac <- filter(Landcover\_frac, Type!="Bedrock")

sumfrac <- sum(Landcover\_frac$Fraction)

Landcover\_frac$Fraction <- Landcover\_frac$Fraction/sumfrac

Landcover\_frac$Oct1\_surf\_mean <- NA

Landcover\_frac$Oct1\_25cm\_mean <- NA

Oct1\_M <- filter(Oct1\_avgM, !Landuse=="Overall")

Oct1\_M

Landcover\_frac$Oct1\_surf\_mean <- Oct1\_M$M\_Depth\_surf\_mean

Landcover\_frac$Oct1\_25cm\_mean <- Oct1\_M$M\_Depth\_25cm\_mean

Landcover\_frac

Landcover\_frac$Oct1\_surf\_mean <- as.numeric(Landcover\_frac$Oct1\_surf\_mean)

Landcover\_frac$Oct1\_25cm\_mean <- as.numeric(Landcover\_frac$Oct1\_25cm\_mean)

Total\_surf\_M <- sum(Landcover\_frac$Fraction\*Landcover\_frac$Oct1\_surf\_mean)

Total\_25cm\_M <- sum(Landcover\_frac$Fraction\*Landcover\_frac$Oct1\_25cm\_mean)

Total\_surf\_M

Total\_25cm\_M

sum(Landcover\_frac$Fraction)

#Do some plotting of the soil temperatures to get a sense of the change with depth and time

ColNames\_Rock <- c("T\_surf", "T\_10cm", "T\_20cm", "T\_30cm", "T\_46cm")

RockT\_plot <- gather(RockT, ColNames\_Rock, key="Depth", value="RockTemp")

RockT\_plot <- mutate(RockT\_plot, Year=year(DateTime))

RockT\_plot <- mutate(RockT\_plot, JDay=yday(DateTime))

SoilT\_plot <- mutate(SoilT, Year=year(DateTime), JDay=yday(DateTime), MonthDay=month(DateTime))

SoilT\_plot <- filter(SoilT\_plot, month(DateTime)==9 | month(DateTime)==10)

ggplot() +

geom\_line(data=SoilT\_plot, mapping=aes(x=JDay, y=SoilTemp,color=T\_Depth)) +

facet\_grid(Year ~ .)

ggplot() +

geom\_line(data=SoilT\_plot, mapping=aes(x=JDay, y=SoilTemp,color=T\_Depth)) +

facet\_grid(Year ~ .)

#Obtaining the air temperature of the canopy on October 1, 2005 (assuming = air temp)

#Note: Vital has no data so use Yellowknife Data

load('C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/R Code/Yknife.Rda')

head(Yknife)

Oct1\_TCAN <- filter(Yknife, date(DateTime)=="2005-10-01")

Oct1\_TCAN <- mean(Oct1\_TCAN$T\_2m)

Oct1\_TCAN

#Obtaining the ponding temperature; use Twater from the Landing data

colnames\_landing=c('DateTime', 'u\_1.1m', 'u\_dir', 'T\_1.1m', 'e\_1.1m', 'Qstar', 'Kin', 'Kout', 'Twater', 'Qe', 'Qh')

Landing\_load <- read\_csv(file="C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/Data/ESSD Baker Creek Data/HydrometeorologicalData/landing tower half hourly time series v1.csv",col\_names=colnames\_landing, skip=1)

Landing\_load$DateTime <- dmy\_hm(Landing\_load$DateTime)

head(Landing\_load)

Oct1\_TPND <- filter(Landing\_load, !Twater==9999, month(DateTime)==10, day(DateTime)==1)

Oct1\_TPND <- mean(Oct1\_TPND$Twater)

head(Oct1\_TPND)

## Streamflow Data

---

title: "Baker Creek Watershed MESH Model - Data Preparation"

output: html\_notebook

---

This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

First, load the libraries that will be used in the code.

```{r}

library(tidyverse)

library(dplyr)

library(lubridate)

library(devtools)

library(CRHMr)

library(ggpubr)

```

## Prepare Streamflow Data for the MESH Model

Streamflow data was obtained from the Water Survey of Canada website for station 07SB013 Baker Creek at the Outlet of Lower Martin Lake [WSC - 07SB013](https://wateroffice.ec.gc.ca/search/historical\_results\_e.html?search\_type=station\_number&station\_number=07sb013&start\_year=1850&end\_year=2019&minimum\_years=&gross\_drainage\_operator=%3E&gross\_drainage\_area=&effective\_drainage\_operator=%3E&effective\_drainage\_area=)

This station includes both discharge (param=1) and water level (param=2) data for the years 1983-2016, so it was filtered for discharge data only from 2015 onward.

### Loading in the data

```{r}

Qload <- read\_csv(file="C:/Users/haley/OneDrive/Documents/1.MWS2018-2019/T2/Project/ECCC\_Project/Data/WSC Streamflow/07SB013 - Daily\_\_May-13-2019\_Date-Data.csv")

# Param=1: Daily Discharge, Param=2: Daily Water Level

# Symbols: E=Estimate, A=PartialDay, B=Ice Conditions, D=Dry, R=Revised

```

### Using CRHMr to explore the missing values in the data

```{r}

Q <- Qload

Q <- Q %>%

filter(PARAM==1 & Date>="2005-01-01") %>%

select(-ID, -PARAM) %>%

rename(datetime=Date)

#Convert date of Q to POSIXct for use with CRHMr package

Q\_df <- as.data.frame(Q)

Q\_df$datetime <- as.POSIXct(Q\_df$datetime, tz="MST")

head(Q\_df)

Qgaps <- findGaps(Q\_df, minlength=1, quiet=FALSE)

```

### Generate plots of the full, observed streamflow to view it

```{r}

QPlot <- Q

ggplot(data=QPlot, mapping=aes(x=datetime))+

geom\_line(aes(y=Value)) +

scale\_x\_date(date\_labels=("%Y"), date\_breaks=("years"))

QPlot2 <- filter(QPlot, datetime>=as.Date("2006-09-15"))

QPlot2 <- filter(QPlot2, datetime<=as.Date("2016-09-14"))

QPlot2 <- select(QPlot2, -SYM)

Q\_Report <- ggplot(data=QPlot2, mapping=aes(x=datetime))+

geom\_line(aes(y=Value)) +

scale\_x\_date(date\_labels=("%Y"), date\_breaks=("years"))+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date")+

theme(axis.title.y=element\_text(size=7))

```

### Load and plot precipitation used in the model to compare with streamflow

```{r}

#Load in the precipitation data used in the model

Pload <- read.csv("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Driving Data/Original/basin\_rain.xlsx.csv")

#Since the units of P used in the model are mm/s, convert to mm by multiplying by 60\*30

P <- Pload

P <- mutate(P, P\_mm=Combined\*60\*30)

colnames(P) <- c("Datetime", "P\_mm\_s", "P\_mm")

#Convert to daily rainfall

P <- mutate(P, Date=date(Datetime))

P\_daily <- P

P\_daily <- P\_daily %>%

group\_by(Date) %>%

summarise(DailySum=sum(P\_mm))%>%

filter(Date>= as.Date("2006-09-15"))%>%

filter(Date <= as.Date("2016-09-14"))

write\_excel\_csv(P\_daily, "F:/ECCC\_Project/R Code/DailyPModel.csv")

P\_Report <- ggplot(P\_daily) +

geom\_col(mapping=aes(x=Date, y=DailySum))+

scale\_x\_date(date\_labels=("%Y"), date\_breaks=("years"))+

ylab("Daily Precipitation (mm)")+

theme(axis.title.x = element\_blank(), axis.text.x=element\_text(size=0), axis.title.y=element\_text(size=7))+

ylim(30,0)

P\_Report

PandQPlot <- ggarrange(P\_Report, Q\_Report, ncol=1, nrow=2, heights=c(0.75, 1.5), align="v")

PandQPlot

ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/PandQPlot.jpg", plot=PandQPlot, width=17.75, height=9, units="cm")

```

### Separate the streamflow into calibration and validation periods and save them as .csv files

The model starts on September 15, 2005 (day 258), so will choose the calibration periods to also start on Sept. 15 and end on Sept. 14

(Note: chose spin-up period of 2006-258 through 2007-257. Afterward realized this has to be consistent for both cal and val periods; therefore, not running the model for 2005-2006. Incorporated this into the streamflow values.)

Calibration period: 2007-258 through 2010-257, and 2013-258 to 2015-257 (inclusive; start Sept. 15 and end Sept. 14)

Validation period: remainder of the modelled period, i.e. 2010-258 to 2013-257, and 2015-258 to 2016-258

```{r}

# Qfull is the complete, original streamflow dataset, filtered to start at 2005-09-15 and to replace missing values with -9999

Qfull <- Qload

#Fill missing values with a negative number (for the model input)

Qfull <- Qfull %>%

filter(PARAM==1 & Date>="2005-01-01") %>%

select(-ID, -PARAM)

Qfull$Value[is.na(Qfull$Value)==TRUE] <- -9999

# Create Qcal which contains only the measured flow during the calibration period

Qcal1 <- Qfull

Qcal1 <- filter(Qcal1, Date>="2007-09-15" & Date<="2010-09-14")

Qcal2 <- Qfull

Qcal2 <- filter(Qcal2, Date >= "2013-09-15" & Date <= "2015-09-14")

Qcal <- rbind(Qcal1, Qcal2)

Qcal <- rename(Qcal, CalFlow=Value)

# Create Qval which contains only the measured flow during the validation period

Qval1 <- Qfull

Qval1 <- filter(Qval1, Date>="2010-09-15" & Date<="2013-09-14")

Qval2 <- Qfull

Qval2 <- filter(Qval2, Date >= "2015-09-15" & Date <= "2016-09-14")

Qval <- rbind(Qval1, Qval2)

Qval <- rename(Qval, ValFlow=Value)

# Create the "negative" flow, which changes the sign of flow>0, and represents missing and zero flows with -9999

Qneg <- mutate(Qfull, Negative=ifelse(Value==0|Value==-9999,-9999,-1\*Value))

Qneg <- select(Qneg, Date, Negative)

# Qneg\_check <- filter(Qneg, Value==-9999)

# Qneg\_check <- filter(Qneg, Value==0)

# Put the negative, calibration, and validation flows together

Qboth <- merge(Qneg, Qcal,by="Date", all=TRUE )

Qboth <- select(Qboth, -SYM)

Qboth <- merge(Qboth, Qval, by="Date", all=TRUE)

Qboth <- select(Qboth, -SYM)

Qboth$CalAll <- NA

Qboth$ValAll <- NA

# This section first pastes Cal.Period into Cal anywhere that Cal has an NA value (probaly wouldn't have needed the first step -> could have straight up started with Cal.Period). From that remaining, it pastes the negative streamflow (or missing=0=-9999) into the Cal NA points (which represent all the times outside the cal period)

Qboth$CalAll <- Qboth$CalFlow

Qboth$CalAll[is.na(Qboth$CalAll)] <- paste0(Qboth$Negative[is.na(Qboth$CalAll)])

Qboth$CalAll <- as.double(Qboth$CalAll)

Qboth$ValAll <- Qboth$ValFlow

Qboth$ValAll[is.na(Qboth$ValAll)] <- paste0(Qboth$Negative[is.na(Qboth$ValAll)])

Qboth$ValAll <- as.double(Qboth$ValAll)

QPlot <- Qboth

QPlot <- mutate(QPlot, CalAll=ifelse(CalAll==-9999,NA,CalAll), ValAll=ifelse(ValAll==-9999,NA,ValAll))

ggplot(data=QPlot, mapping=aes(x=Date))+

geom\_line(aes(y=CalAll), color="blue", size=1) +

scale\_x\_date(date\_labels=("%Y"), date\_breaks=("years")) +

geom\_line(aes(y=ValAll), color="red", size=0.5)+

labs(x="Date", y="Discharge")

#Check that the morphed dataset is the same as the original Q dataset

Qcheck <- Qfull

Qcheck$Check <- NA

Qcheck$Check <- abs(QPlot$CalAll)-Qcheck$Value

CalCheck <- force(unique(Qcheck$Check))

CalCheck

Qcheck$Check <- abs(QPlot$ValAll)-Qcheck$Value

ValCheck <- force(unique(Qcheck$Check))

ValCheck

```

Note one last thing: the model needs a positive value on the start date of the model. Therefore, change the streamflow value on 2006-09-15 to a positive value

```{r}

ModelStart <- which(Qboth$Date==as.Date("2006-09-15")) #Returns the line where date=2006=09-15

Qboth$CalAll[ModelStart] <- -1\*Qboth$CalAll[ModelStart]

Qboth$ValAll[ModelStart] <- -1\*Qboth$ValAll[ModelStart]

Qboth[ModelStart,]

```

### Write the streamflow values to file - both an excel .csv file including the date as well, and a .csv file with only the flow values

```{r}

setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files")

QFinal <- select(Qfull, Date, Value)

QWrite <- select(Qfull, Value)

write\_excel\_csv(QFinal, "Streamflow\_full.xlsx.csv")

write\_tsv(QWrite, "Streamflow\_full.csv", col\_names=FALSE)

QFinal <- select(Qboth, Date, CalAll)

QWrite <- select(Qboth, CalAll)

write\_excel\_csv(QFinal, "Streamflow\_cal.xlsx.csv")

write\_tsv(QWrite, "Streamflow\_cal.csv", col\_names=FALSE)

QFinal <- select(Qboth, Date, ValAll)

QWrite <- select(Qboth, ValAll)

write\_excel\_csv(QFinal, "Streamflow\_val.xlsx.csv")

write\_tsv(QWrite, "Streamflow\_val.csv", col\_names=FALSE)

```

# Model Output Processing – R Notebook

---

title: "Baker Creek - MESH Output Visualization and Analysis"

author: "Haley Brauner"

output: html\_notebook

---

# Introduction

The purpose of the R notebook is to have a consistent means to visualize, analyze, evaluate, and compare the results of each scenario of the MESH modelling being conducted in the Baker Creek, NWT watershed during the course of this Masters of Water Security Capstone Project.

The general order to this document is:

- Load in and process the results

- Plot the desired outputs

# MESH Output Processing

## The Code

### Load Libraries

```{r}

library(tidyverse)

library(dplyr)

library(lubridate)

library(CRHMr)

library(MESHr)

library(splitstackshape)

library(plotly)

```

## Evaluate the results of the calibration scenarios

### Load and check the MESH calibration results

This section gathers the results from each of the 100 trials, creates a table of the NS results for each trial, and plots of the NSE value (vs calibration trial). There is one code chunk per scenario

```{r}

### This section works with results from the MESH calibration trials when all the results, saved in ostOutputXXX folders for each trial, are located within a common folder. The only sub-folders in the common folder should be the ostOutputXXX folders

### Currently, the code is copied for each scearnio. May want to update this to make it a function to be run for each scenario with an option within the fuction to rename the variables according to the scenario

#### Scenario 1 Load Calibration Results

folder\_S1 <- list.dirs("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Scenario1\_Calibrated\_1")

folder\_S1 <- folder\_S1[-1]

MetricsFile <- "Metrics\_Out.txt"

CalResultsFile <- "OstModel0.txt"

NSResults\_S1 <- data.frame(Trial=1:100, NS=NA, SubFolder=NA, NObs=NA)

NSEvolve <- data.frame(Run=1:1000)

for (y in 1:length(folder\_S1)){

OutputDir <- folder\_S1[y]

setwd(OutputDir)

Metrics <- read.table(MetricsFile, header=TRUE)

Trial <- str\_sub(folder\_S1[y], start=-3)

Trial <- as.numeric(Trial)

NSResults\_S1[Trial,2] <- Metrics$NSD[1]

NSResults\_S1[Trial,3] <- folder\_S1[y]

# NSE\_read <- read.table(CalResultsFile, header=TRUE)

# NSE <- NSE\_read[,1:2]

# NSE <- mutate(NSE, obj.function=obj.function\*-1)

# NSEvolve <- merge(NSEvolve, NSE, by="Run")

# ColNames <- colnames(NSEvolve)

# ColNames[y+1] <- str\_sub(folder\_S1[y], start=-12)

# colnames(NSEvolve) <- ColNames

}

# ColNames <- ColNames[-1]

# NSEvolve\_gathered <- gather(NSEvolve, ColNames, key="Trial", value="NSE")

#

# ggplot(NSEvolve\_gathered) +

# geom\_line(mapping=aes(x=Run, y=NSE, colour=Trial))+

# theme(legend.position="none")

# write.csv(NSEvolve, file="Scenario1NSEvolve.csv")

#Add a column for the number of streamflow observations and create a sub-set of the NS Results showing only the runs that ran for the full calibration period

#Number of days between 2006-09-15 and 2016-09-14: 3652

#Number of days between 2006-09-15 and 2015-09-14: 3286

for (z in 1:nrow(NSResults\_S1)){

if (is.na(NSResults\_S1$SubFolder[z])==TRUE){

next

}

setwd(NSResults\_S1$SubFolder[z])

Q<- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold = 1e-6)

NSResults\_S1$NObs[z] <- nrow(Q)

}

Full\_Trials\_S1 <- filter(NSResults\_S1, NObs==3652)

NFull\_S1 <- nrow(Full\_Trials\_S1)

NFull\_S1

#These next 2 variables store the ranked NS results (from best NS to worst NS) and the ranked (ordered) parameter values. The NS\_Full\_Ranked dataframe is used to obtain the top 10 best parameter sets for use in validation runs

NSRanked\_S1 <- NSResults\_S1[order( as.numeric(as.character(NSResults\_S1$NS)), decreasing=TRUE ), ]

NSFullRanked\_S1 <- Full\_Trials\_S1[order( as.numeric(as.character(Full\_Trials\_S1$NS)), decreasing=TRUE ), ]

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 1 Full Re-Run Load Calibration Results

####

folder\_S1\_2 <- list.dirs("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Scenario1\_Calibrated\_2")

folder\_S1\_2 <- folder\_S1\_2[-1]

MetricsFile <- "Metrics\_Out.txt"

NSResults\_S1\_2 <- data.frame(Trial=1:100, NS=NA, SubFolder=NA, NObs=NA)

for (y in 1:length(folder\_S1\_2)){

OutputDir <- folder\_S1\_2[y]

setwd(OutputDir)

Metrics <- read.table(MetricsFile, header=TRUE)

Trial <- str\_sub(folder\_S1\_2[y], start=-3)

Trial <- as.numeric(Trial)

NSResults\_S1\_2[Trial,2] <- Metrics$NSD[1]

NSResults\_S1\_2[Trial,3] <- folder\_S1\_2[y]

}

#Add a column for the number of streamflow observations

for (z in 1:nrow(NSResults\_S1\_2)){

if (is.na(NSResults\_S1\_2$SubFolder[z])==TRUE){

next

}

setwd(NSResults\_S1\_2$SubFolder[z])

Q<- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold = 1e-6)

NSResults\_S1\_2$NObs[z] <- nrow(Q)

}

#Add a column for the number of streamflow observations and create a sub-set of the NS Results showing only the runs that ran for the full calibration period

#Number of days between 2006-09-15 and 2016-09-14: 3652

#Number of days between 2006-09-15 and 2015-09-14: 3286

Full\_Trials\_S1\_2 <- filter(NSResults\_S1\_2, NObs>=3286)

NFull\_S1\_2 <- nrow(Full\_Trials\_S1\_2)

NFull\_S1\_2

#These next 2 variables store the ranked NS results (from best NS to worst NS) and the ranked (ordered) parameter values. The NS\_Full\_Ranked dataframe is used to obtain the top 10 best parameter sets for use in validation runs

NSRanked\_S1\_2 <- NSResults\_S1\_2[order( as.numeric(as.character(NSResults\_S1\_2$NS)), decreasing=TRUE ), ]

NSFullRanked\_S1\_2 <- Full\_Trials\_S1\_2[order( as.numeric(as.character(Full\_Trials\_S1\_2$NS)), decreasing=TRUE ), ]

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 1 Full Re-Run #2 (S1\_3) Load Calibration Results

####

folder\_S1\_3 <- list.dirs("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Scenario1\_Calibrated\_3")

folder\_S1\_3 <- folder\_S1\_3[-1]

MetricsFile <- "Metrics\_Out.txt"

NSResults\_S1\_3 <- data.frame(Trial=1:100, NS=NA, SubFolder=NA, NObs=NA)

for (y in 1:length(folder\_S1\_3)){

OutputDir <- folder\_S1\_3[y]

setwd(OutputDir)

Metrics <- read.table(MetricsFile, header=TRUE)

Trial <- str\_sub(folder\_S1\_3[y], start=-3)

Trial <- as.numeric(Trial)

NSResults\_S1\_3[Trial,2] <- Metrics$NSD[1]

NSResults\_S1\_3[Trial,3] <- folder\_S1\_3[y]

}

#Add a column for the number of streamflow observations

for (z in 1:nrow(NSResults\_S1\_3)){

if (is.na(NSResults\_S1\_3$SubFolder[z])==TRUE){

next

}

setwd(NSResults\_S1\_3$SubFolder[z])

Q<- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold = 1e-6)

NSResults\_S1\_3$NObs[z] <- nrow(Q)

}

#Add a column for the number of streamflow observations and create a sub-set of the NS Results showing only the runs that ran for the full calibration period

#Number of days between 2006-09-15 and 2016-09-14: 3652

#Number of days between 2006-09-15 and 2015-09-14: 3286

Full\_Trials\_S1\_3 <- filter(NSResults\_S1\_3, NObs>=3286)

NFull\_S1\_3 <- nrow(Full\_Trials\_S1\_3)

NFull\_S1\_3

#These next 2 variables store the ranked NS results (from best NS to worst NS) and the ranked (ordered) parameter values. The NS\_Full\_Ranked dataframe is used to obtain the top 10 best parameter sets for use in validation runs

NSRanked\_S1\_3 <- NSResults\_S1\_3[order( as.numeric(as.character(NSResults\_S1\_3$NS)), decreasing=TRUE ), ]

NSFullRanked\_S1\_3 <- Full\_Trials\_S1\_3[order( as.numeric(as.character(Full\_Trials\_S1\_3$NS)), decreasing=TRUE ), ]

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 2 Load Calibration Results

####

folder\_S2 <- list.dirs("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Scenario2\_Calibrated")

folder\_S2 <- folder\_S2[-1]

MetricsFile <- "Metrics\_Out.txt"

NSResults\_S2 <- data.frame(Trial=1:100, NS=NA, SubFolder=NA, NObs=NA)

for (y in 1:length(folder\_S2)){

OutputDir <- folder\_S2[y]

setwd(OutputDir)

Metrics <- read.table(MetricsFile, header=TRUE)

Trial <- str\_sub(folder\_S2[y], start=-3)

Trial <- as.numeric(Trial)

NSResults\_S2[Trial,2] <- Metrics$NSD[1]

NSResults\_S2[Trial,3] <- folder\_S2[y]

}

#Add a column for the number of streamflow observations

for (z in 1:nrow(NSResults\_S2)){

if (is.na(NSResults\_S2$SubFolder[z])==TRUE){

next

}

setwd(NSResults\_S2$SubFolder[z])

Q<- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold = 1e-6)

NSResults\_S2$NObs[z] <- nrow(Q)

}

#Add a column for the number of streamflow observations and create a sub-set of the NS Results showing only the runs that ran for the full calibration period

#Number of days between 2006-09-15 and 2016-09-14: 3652

#Number of days between 2006-09-15 and 2015-09-14: 3286

Full\_Trials\_S2 <- filter(NSResults\_S2, NObs>=3286)

NFull\_S2 <- nrow(Full\_Trials\_S2)

NFull\_S2

#These next 2 variables store the ranked NS results (from best NS to worst NS) and the ranked (ordered) parameter values. The NS\_Full\_Ranked dataframe is used to obtain the top 10 best parameter sets for use in validation runs

NSRanked\_S2 <- NSResults\_S2[order( as.numeric(as.character(NSResults\_S2$NS)), decreasing=TRUE ), ]

NSFullRanked\_S2 <- Full\_Trials\_S2[order( as.numeric(as.character(Full\_Trials\_S2$NS)), decreasing=TRUE ), ]

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 3 Load Calibration Results

####

folder\_S3 <- list.dirs("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Scenario3\_Calibrated")

folder\_S3 <- folder\_S3[-1]

MetricsFile <- "Metrics\_Out.txt"

NSResults\_S3 <- data.frame(Trial=1:100, NS=NA, SubFolder=NA, NObs=NA)

for (y in 1:length(folder\_S3)){

OutputDir <- folder\_S3[y]

setwd(OutputDir)

Metrics <- read.table(MetricsFile, header=TRUE)

Trial <- str\_sub(folder\_S3[y], start=-3)

Trial <- as.numeric(Trial)

NSResults\_S3[Trial,2] <- Metrics$NSD[1]

NSResults\_S3[Trial,3] <- folder\_S3[y]

}

#Add a column for the number of streamflow observations

for (z in 1:nrow(NSResults\_S3)){

if (is.na(NSResults\_S3$SubFolder[z])==TRUE){

next

}

setwd(NSResults\_S3$SubFolder[z])

Q<- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold = 1e-6)

NSResults\_S3$NObs[z] <- nrow(Q)

}

#Add a column for the number of streamflow observations and create a sub-set of the NS Results showing only the runs that ran for the full calibration period

#Number of days between 2006-09-15 and 2016-09-14: 3652

#Number of days between 2006-09-15 and 2015-09-14: 3286

Full\_Trials\_S3 <- filter(NSResults\_S3, NObs>=3286)

NFull\_S3 <- nrow(Full\_Trials\_S3)

NFull\_S3

#These next 2 variables store the ranked NS results (from best NS to worst NS) and the ranked (ordered) parameter values. The NS\_Full\_Ranked dataframe is used to obtain the top 10 best parameter sets for use in validation runs

NSRanked\_S3 <- NSResults\_S3[order( as.numeric(as.character(NSResults\_S3$NS)), decreasing=TRUE ), ]

NSFullRanked\_S3 <- Full\_Trials\_S3[order( as.numeric(as.character(Full\_Trials\_S3$NS)), decreasing=TRUE ), ]

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 1P Load Calibration Results

####

folder\_S1P <- list.dirs("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Scenario1\_PDM\_Calibrated")

folder\_S1P <- folder\_S1P[-1]

MetricsFile <- "Metrics\_Out.txt"

NSResults\_S1P <- data.frame(Trial=1:100, NS=NA, SubFolder=NA, NObs=NA)

for (y in 1:length(folder\_S1P)){

OutputDir <- folder\_S1P[y]

setwd(OutputDir)

Metrics <- read.table(MetricsFile, header=TRUE)

Trial <- str\_sub(folder\_S1P[y], start=-3)

Trial <- as.numeric(Trial)

NSResults\_S1P[Trial,2] <- Metrics$NSD[1]

NSResults\_S1P[Trial,3] <- folder\_S1P[y]

}

#These next 2 variables store the ranked NS results (from best NS to worst NS) and the ranked (ordered) parameter values. The NS\_Full\_Ranked dataframe is used to obtain the top 10 best parameter sets for use in validation runs

NSRanked\_S1P <- NSResults\_S1P[order( as.numeric(as.character(NSResults\_S1P$NS)), decreasing=TRUE ), ]

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 2P Load Calibration Results

####

folder\_S2P <- list.dirs("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Scenario2\_PDM\_Calibrated")

folder\_S2P <- folder\_S2P[-1]

MetricsFile <- "Metrics\_Out.txt"

NSResults\_S2P <- data.frame(Trial=1:100, NS=NA, SubFolder=NA, NObs=NA)

for (y in 1:length(folder\_S2P)){

OutputDir <- folder\_S2P[y]

setwd(OutputDir)

Metrics <- read.table(MetricsFile, header=TRUE)

Trial <- str\_sub(folder\_S2P[y], start=-3)

Trial <- as.numeric(Trial)

NSResults\_S2P[Trial,2] <- Metrics$NSD[1]

NSResults\_S2P[Trial,3] <- folder\_S2P[y]

}

#These next 2 variables store the ranked NS results (from best NS to worst NS) and the ranked (ordered) parameter values. The NS\_Full\_Ranked dataframe is used to obtain the top 10 best parameter sets for use in validation runs

NSRanked\_S2P <- NSResults\_S2P[order( as.numeric(as.character(NSResults\_S2P$NS)), decreasing=TRUE ), ]

```

### This chunk contains a function to obtain the parameter sets of the best calibration results

```{r include=FALSE}

#Define a function to create a data frame of the optimal parameter values for each trail, calculate the min, max, and 10th and 90th percentile statistics, calculate the normalized values, and plot the parameter identifiability

ParamIdent <- function(FolderList, ScenarioNumber, XLabSize) {

for (j in 1:length(FolderList)){

Dir <- FolderList[j]

setwd(Dir)

OstOut <- read\_lines("OstOutput0.txt")

OstOut <- str\_replace\_all(OstOut,c("best fitness","trials remaining"),c("best\_fitness","trials\_remaining"))

OstOut <- data.frame(OstOut)

#Extract information from the "Optimal Parameter Set" section (OstBest) and put them all together in one dataframe (OstBestAll)

StartRow <- which(OstOut[,1]%in%"Optimal Parameter Set")

EndRow <- which(OstOut[,1]%in%"Summary of Constraints")-2

OstBest <- slice(OstOut,StartRow:EndRow)

OstBest <- OstBest[-c(1,2),]

OstBest <- data.frame(OstBest)

OstBest <- separate(OstBest,1,into=c("Parameter","Value"), sep=":")

# OstTop10 <- OstBest[-1,]

colnames(OstBest) <- c("Parameter", paste("Trial",j, sep=""))

# colnames(OstTop10) <- c("Parameter", paste("Trial",j, sep=""))

if (j==1){

OstBestAll <- OstBest

} else {

OstBestAll <- merge(OstBestAll, OstBest, by="Parameter")

}

}

OstBestAll[,1]<- gsub("\_","",OstBestAll[,1])

OstBest <- OstBest

OstBestAll <- OstBestAll

assign(paste("OstBestAll", "S", ScenarioNumber,sep=""),OstBestAll, envir=.GlobalEnv)

OstBestAllTrans <- t(OstBestAll)

names <- rownames(OstBestAllTrans)

names <- names[-1]

colnames(OstBestAllTrans) <- OstBestAllTrans[1,]

OstBestAllTrans <- OstBestAllTrans[-1,]

ColNames <- colnames(OstBestAllTrans)

OstBestAllTrans <- data.frame(apply(OstBestAllTrans,2,function(x) as.numeric(as.character(x))))

colnames(OstBestAllTrans) <- ColNames

rownames(OstBestAllTrans) <- names

#Calculate the min and max, as well as the 10th and 90th percentile values for each parameter.

OstBestSummary <- summarise\_all(OstBestAllTrans, min)

OstBestSummary[2,] <- summarise\_all(OstBestAllTrans, max)

OstBestSummary[3,] <- sapply(OstBestAllTrans, quantile, probs=0.10)

OstBestSummary[4,] <- sapply(OstBestAllTrans, quantile, probs=0.90)

rownames(OstBestSummary) <- c("Min", "Max", "Tenth", "Ninetieth")

Param\_Names <- colnames(OstBestSummary)

OstBestNormalized <- t(OstBestSummary)

OstBestNormalized <- data.frame(apply(OstBestNormalized,2,function(x) as.numeric(as.character(x))))

OstBestNormalized <- OstBestNormalized %>%

mutate(Norm\_10th=0+(Tenth-Min)\*(1-0)/(Max-Min))%>%

mutate(Norm\_90th=0+(Ninetieth-Min)\*(1-0)/(Max-Min)) %>%

mutate(Diff=Norm\_90th-Norm\_10th)

OstBestNormalized <- cbind(Param\_Names, OstBestNormalized)

assign(paste("OstBestNorm\_S",ScenarioNumber, sep=""), OstBestNormalized, envir=.GlobalEnv)

}

```

### Calculate Validation NSE

```{r}

# Load the measured streamflow and filter down to 2006-09-15 through 2016-09-13

Q\_val\_load <- read.csv("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Streamflow\_val.xlsx.csv")

Q\_val <- Q\_val\_load

Q\_val[,1] <- as.Date.factor(Q\_val[,1])

colnames(Q\_val) <- c("Date", "Q\_meas\_val")

Q\_val <- filter(Q\_val, Date>=as.Date("2006-09-15"), Date<=as.Date("2016-09-13"))

Q\_val <- mutate(Q\_val, Q\_meas\_val=ifelse(Q\_meas\_val==-9999,NA,Q\_meas\_val))

ggplot()+

geom\_line(data=Q\_val, mapping=aes(x=Date, y=Q\_meas\_val))

```

```{r}

# Scenario 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Load the simulated streamflow from the top 10 calibration runs

for (i in 1:10){

setwd(NSRanked\_S1$SubFolder[i])

Qsim\_load <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold=-100)

Qsim\_load <- select(Qsim\_load, -QOMEAS1)

colnames(Qsim\_load) <- c("Date", paste("Qsim\_Top",i, sep=""))

if (i==1){

Q\_Top10\_S1 <- Qsim\_load

} else {

Q\_Top10\_S1 <- merge(Q\_Top10\_S1, Qsim\_load, by="Date")

}

}

# Combine the measured and simulated streamflow and filter out all the missing value dates (which correspond to the spin-up and calibration periods)

Q\_val\_S1 <- merge(Q\_val, Q\_Top10\_S1, by="Date")

Q\_val\_S1 <- filter(Q\_val\_S1, Q\_meas\_val>=0 & is.na(Q\_meas\_val)==FALSE)

# Write Q\_val\_S1 to .csv to check calcs below

# setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/")

# write.csv(Q\_val\_S1, "NSE\_Calc\_Check.csv")

# Calculate the nash-sutcliffe for each of the Top 10 simulated streamflows

ValNSES1 <- data.frame(Top10=c(1:10), NSE=NA)

j=1

for (i in 3:12){

QObsAvg <- mean(Q\_val\_S1$Q\_meas\_val)

QDiffSq <- data.frame((Q\_val\_S1[,i] - Q\_val\_S1$Q\_meas\_val)^2)

Numerator <- sum(QDiffSq[,1])

QMeanSq <- data.frame((Q\_val\_S1$Q\_meas\_val-QObsAvg)^2)

Denom <- sum(QMeanSq[,1])

ValNSES1[j,2] <- 1-(Numerator/Denom)

j=j+1

}

```

```{r}

# Scenario 1\_2 (Re-Run) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Load the simulated streamflow from the top 10 calibration runs

for (i in 1:10){

setwd(NSRanked\_S1\_2$SubFolder[i])

Qsim\_load <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold=-100)

Qsim\_load <- select(Qsim\_load, -QOMEAS1)

colnames(Qsim\_load) <- c("Date", paste("Qsim\_Top",i, sep=""))

if (i==1){

Q\_Top10\_S1\_2 <- Qsim\_load

} else {

Q\_Top10\_S1\_2 <- merge(Q\_Top10\_S1\_2, Qsim\_load, by="Date")

}

}

# Combine the measured and simulated streamflow and filter out all the missing value dates (which correspond to the spin-up and calibration periods)

Q\_val\_S1\_2 <- merge(Q\_val, Q\_Top10\_S1\_2, by="Date")

Q\_val\_S1\_2 <- filter(Q\_val\_S1\_2, Q\_meas\_val>=0 & is.na(Q\_meas\_val)==FALSE)

# Write Q\_val\_S1 to .csv to check calcs below

# setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/")

# write.csv(Q\_val\_S1, "NSE\_Calc\_Check.csv")

# Calculate the nash-sutcliffe for each of the Top 10 simulated streamflows

ValNSES1\_2 <- data.frame(Top10=c(1:10), NSE=NA)

j=1

for (i in 3:12){

QObsAvg <- mean(Q\_val\_S1\_2$Q\_meas\_val)

QDiffSq <- data.frame((Q\_val\_S1\_2[,i] - Q\_val\_S1\_2$Q\_meas\_val)^2)

Numerator <- sum(QDiffSq[,1])

QMeanSq <- data.frame((Q\_val\_S1\_2$Q\_meas\_val-QObsAvg)^2)

Denom <- sum(QMeanSq[,1])

ValNSES1\_2[j,2] <- 1-(Numerator/Denom)

j=j+1

}

```

```{r}

# Scenario 1\_3 (Re-Run #2) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Load the simulated streamflow from the top 10 calibration runs

for (i in 1:10){

setwd(NSRanked\_S1\_3$SubFolder[i])

Qsim\_load <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold=-100)

Qsim\_load <- select(Qsim\_load, -QOMEAS1)

colnames(Qsim\_load) <- c("Date", paste("Qsim\_Top",i, sep=""))

if (i==1){

Q\_Top10\_S1\_3 <- Qsim\_load

} else {

Q\_Top10\_S1\_3 <- merge(Q\_Top10\_S1\_3, Qsim\_load, by="Date")

}

}

# Combine the measured and simulated streamflow and filter out all the missing value dates (which correspond to the spin-up and calibration periods)

Q\_val\_S1\_3 <- merge(Q\_val, Q\_Top10\_S1\_3, by="Date")

Q\_val\_S1\_3 <- filter(Q\_val\_S1\_3, Q\_meas\_val>=0 & is.na(Q\_meas\_val)==FALSE)

# Write Q\_val\_S1 to .csv to check calcs below

# setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/")

# write.csv(Q\_val\_S1, "NSE\_Calc\_Check.csv")

# Calculate the nash-sutcliffe for each of the Top 10 simulated streamflows

ValNSES1\_3 <- data.frame(Top10=c(1:10), NSE=NA)

j=1

for (i in 3:12){

QObsAvg <- mean(Q\_val\_S1\_3$Q\_meas\_val)

QDiffSq <- data.frame((Q\_val\_S1\_3[,i] - Q\_val\_S1\_3$Q\_meas\_val)^2)

Numerator <- sum(QDiffSq[,1])

QMeanSq <- data.frame((Q\_val\_S1\_3$Q\_meas\_val-QObsAvg)^2)

Denom <- sum(QMeanSq[,1])

ValNSES1\_3[j,2] <- 1-(Numerator/Denom)

j=j+1

}

```

```{r}

# Scenario 2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Load the simulated streamflow from the top 10 calibration runs

for (i in 1:10){

setwd(NSRanked\_S2$SubFolder[i])

Qsim\_load <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold=-100)

Qsim\_load <- select(Qsim\_load, -QOMEAS1)

colnames(Qsim\_load) <- c("Date", paste("Qsim\_Top",i, sep=""))

if (i==1){

Q\_Top10\_S2 <- Qsim\_load

} else {

Q\_Top10\_S2 <- merge(Q\_Top10\_S2, Qsim\_load, by="Date")

}

}

# Combine the measured and simulated streamflow and filter out all the missing value dates (which correspond to the spin-up and calibration periods)

Q\_val\_S2 <- merge(Q\_val, Q\_Top10\_S2, by="Date")

Q\_val\_S2 <- filter(Q\_val\_S2, Q\_meas\_val>=0 & is.na(Q\_meas\_val)==FALSE)

# Write Q\_val\_S2 to .csv to check calcs below

# setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/")

# write.csv(Q\_val\_S2, "NSE\_Calc\_Check.csv")

# Calculate the nash-sutcliffe for each of the Top 10 simulated streamflows

ValNSES2 <- data.frame(Top10=c(1:10), NSE=NA)

j=1

for (i in 3:12){

QObsAvg <- mean(Q\_val\_S2$Q\_meas\_val)

QDiffSq <- data.frame((Q\_val\_S2[,i] - Q\_val\_S2$Q\_meas\_val)^2)

Numerator <- sum(QDiffSq[,1])

QMeanSq <- data.frame((Q\_val\_S2$Q\_meas\_val-QObsAvg)^2)

Denom <- sum(QMeanSq[,1])

ValNSES2[j,2] <- 1-(Numerator/Denom)

j=j+1

}

```

```{r}

# Scenario 3 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Load the simulated streamflow from the top 10 calibration runs

for (i in 1:10){

setwd(NSRanked\_S3$SubFolder[i])

Qsim\_load <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold=-100)

Qsim\_load <- select(Qsim\_load, -QOMEAS1)

colnames(Qsim\_load) <- c("Date", paste("Qsim\_Top",i, sep=""))

if (i==1){

Q\_Top10\_S3 <- Qsim\_load

} else {

Q\_Top10\_S3 <- merge(Q\_Top10\_S3, Qsim\_load, by="Date")

}

}

# Combine the measured and simulated streamflow and filter out all the missing value dates (which correspond to the spin-up and calibration periods)

Q\_val\_S3 <- merge(Q\_val, Q\_Top10\_S3, by="Date")

Q\_val\_S3 <- filter(Q\_val\_S3, Q\_meas\_val>=0 & is.na(Q\_meas\_val)==FALSE)

# Write Q\_val\_S3 to .csv to check calcs below

# setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/")

# write.csv(Q\_val\_S3, "NSE\_Calc\_Check.csv")

# Calculate the nash-sutcliffe for each of the Top 10 simulated streamflows

ValNSES3 <- data.frame(Top10=c(1:10), NSE=NA)

j=1

for (i in 3:12){

QObsAvg <- mean(Q\_val\_S3$Q\_meas\_val)

QDiffSq <- data.frame((Q\_val\_S3[,i] - Q\_val\_S3$Q\_meas\_val)^2)

Numerator <- sum(QDiffSq[,1])

QMeanSq <- data.frame((Q\_val\_S3$Q\_meas\_val-QObsAvg)^2)

Denom <- sum(QMeanSq[,1])

ValNSES3[j,2] <- 1-(Numerator/Denom)

j=j+1

}

```

```{r}

# Scenario 1-P \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Load the simulated streamflow from the top 10 calibration runs

for (i in 1:10){

setwd(NSRanked\_S1P$SubFolder[i])

Qsim\_load <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold=-100)

Qsim\_load <- select(Qsim\_load, -QOMEAS1)

colnames(Qsim\_load) <- c("Date", paste("Qsim\_Top",i, sep=""))

if (i==1){

Q\_Top10\_S1P <- Qsim\_load

} else {

Q\_Top10\_S1P <- merge(Q\_Top10\_S1P, Qsim\_load, by="Date")

}

}

# Combine the measured and simulated streamflow and filter out all the missing value dates (which correspond to the spin-up and calibration periods)

Q\_val\_S1P <- merge(Q\_val, Q\_Top10\_S1P, by="Date")

Q\_val\_S1P <- filter(Q\_val\_S1P, Q\_meas\_val>=0 & is.na(Q\_meas\_val)==FALSE)

# Write Q\_val\_S1P to .csv to check calcs below

# setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/")

# write.csv(Q\_val\_S1P, "NSE\_Calc\_Check.csv")

# Calculate the nash-sutcliffe for each of the Top 10 simulated streamflows

ValNSES1P <- data.frame(Top10=c(1:10), NSE=NA)

j=1

for (i in 3:12){

QObsAvg <- mean(Q\_val\_S1P$Q\_meas\_val)

QDiffSq <- data.frame((Q\_val\_S1P[,i] - Q\_val\_S1P$Q\_meas\_val)^2)

Numerator <- sum(QDiffSq[,1])

QMeanSq <- data.frame((Q\_val\_S1P$Q\_meas\_val-QObsAvg)^2)

Denom <- sum(QMeanSq[,1])

ValNSES1P[j,2] <- 1-(Numerator/Denom)

j=j+1

}

```

```{r}

# Scenario 2-P \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Load the simulated streamflow from the top 10 calibration runs

for (i in 1:10){

setwd(NSRanked\_S2P$SubFolder[i])

Qsim\_load <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv", missingValueThreshold=-100)

Qsim\_load <- select(Qsim\_load, -QOMEAS1)

colnames(Qsim\_load) <- c("Date", paste("Qsim\_Top",i, sep=""))

if (i==1){

Q\_Top10\_S2P <- Qsim\_load

} else {

Q\_Top10\_S2P <- merge(Q\_Top10\_S2P, Qsim\_load, by="Date")

}

}

# Combine the measured and simulated streamflow and filter out all the missing value dates (which correspond to the spin-up and calibration periods)

Q\_val\_S2P <- merge(Q\_val, Q\_Top10\_S2P, by="Date")

Q\_val\_S2P <- filter(Q\_val\_S2P, Q\_meas\_val>=0 & is.na(Q\_meas\_val)==FALSE)

# Write Q\_val\_S2P to .csv to check calcs below

# setwd("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/")

# write.csv(Q\_val\_S2P, "NSE\_Calc\_Check.csv")

# Calculate the nash-sutcliffe for each of the Top 10 simulated streamflows

ValNSES2P <- data.frame(Top10=c(1:10), NSE=NA)

j=1

for (i in 3:12){

QObsAvg <- mean(Q\_val\_S2P$Q\_meas\_val)

QDiffSq <- data.frame((Q\_val\_S2P[,i] - Q\_val\_S2P$Q\_meas\_val)^2)

Numerator <- sum(QDiffSq[,1])

QMeanSq <- data.frame((Q\_val\_S2P$Q\_meas\_val-QObsAvg)^2)

Denom <- sum(QMeanSq[,1])

ValNSES2P[j,2] <- 1-(Numerator/Denom)

j=j+1

}

```

# Plots

## Table summary of Trial, NSE for Cal, and NSE for Val

```{r}

Perf\_Summary <- data.frame(Scenario=c("Scenario 1", "Scenario1\_2", "Scenario1\_3", "Scenario 2", "Scenario 3", "Scenario 1-P", "Scenario 2-P"), Trial=c(NSRanked\_S1$Trial[1],NSRanked\_S1\_2$Trial[1],NSRanked\_S1\_3$Trial[1], NSRanked\_S2$Trial[1],NSRanked\_S3$Trial[1], NSRanked\_S1P$Trial[1], NSRanked\_S2P$Trial[1]), Cal\_NSE=c(NSRanked\_S1$NS[1],NSRanked\_S1\_2$NS[1],NSRanked\_S1\_3$NS[1], NSRanked\_S2$NS[1],NSRanked\_S3$NS[1], NSRanked\_S1P$NS[1], NSRanked\_S2P$NS[1]),Val\_NSE=c(ValNSES1$NSE[1],ValNSES1\_2$NSE[1],ValNSES1\_3$NSE[1], ValNSES2$NSE[1],ValNSES3$NSE[1], ValNSES1P$NSE[1], ValNSES2P$NSE[1]))

```

## Model Performance for 100 Calibration Runs; All scenarios on the same plot

```{r}

#Box plot of the Objective parameter results of the 100 calibration trials

BoxPlot100 <- ggplot()+

# geom\_boxplot(data=NSResults\_S1, mapping=aes(x="Scenario 1", y=NS))+

# geom\_boxplot(data=NSResults\_S1\_2, mapping=aes(x="Scenario 1\_2", y=NS))+

geom\_boxplot(data=NSResults\_S1\_3, mapping=aes(x="Scenario 1", y=NS))+

geom\_boxplot(data=NSResults\_S1P, mapping=aes(x="Scenario 1-P", y=NS))+

ylab("Nash-Sutcliffe")+

xlab("") +

geom\_boxplot(data=NSResults\_S2, mapping=aes(x="Scenario 2", y=NS))+

geom\_boxplot(data=NSResults\_S2P, mapping=aes(x="Scenario 2-P", y=NS))+

geom\_boxplot(data=NSResults\_S3, mapping=aes(x="Scenario 3", y=NS))+

labs(title="Model Performance in 100 Calibration Trials", subtitle="MESH Model, Baker Creek Watershed")

BoxPlot100 <- BoxPlot100 + theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

BoxPlot100

ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/BoxPlot100.jpg", plot=BoxPlot100, height=9, width=17.75, unit="cm")

# Violin and box plot of the objective parameter results of the 100 calibration trials

# ggplot()+

# geom\_violin(data=NSResults\_S1\_2, mapping=aes(x="Scenario 1\_2", y=NS))+

# ylab("Nash-Sutcliffe")+

# xlab("") +

# geom\_violin(data=NSResults\_S2, mapping=aes(x="Scenario 2", y=NS))+

# geom\_violin(data=NSResults\_S3, mapping=aes(x="Scenario 3", y=NS))+

# labs(title="Model Performance in 100 Calibration Trials", subtitle="MESH Model, Baker Creek Watershed")+

# geom\_boxplot(data=NSResults\_S1, mapping=aes(x="Scenario 1", y=NS), width=0.1)+

# geom\_boxplot(data=NSResults\_S1\_2, mapping=aes(x="Scenario 1\_2", y=NS), width=0.1)+

# geom\_boxplot(data=NSResults\_S2, mapping=aes(x="Scenario 2", y=NS), width=0.1)+

# geom\_boxplot(data=NSResults\_S3, mapping=aes(x="Scenario 3", y=NS), width=0.1)

# Note: The lower and upper hinges of the boxplot correspond to the first and third quartiles (25th and 75th percentiles). The upper whisker extends from the hinge to a maximum of 1.5\* IQR (IQR=distance between the 1st and 3rd quartiles)

```

## Observed vs Simulated Streamflow for the Best Calibration Runs

Plot observed vs. simulated streamflow for the best calibration run for each scenario

- Show the NSE on the graph

- Be sure to label axes and add a title and legend

```{r, eval=FALSE}

### Scenario 1 Streamflow Plot

# Obtain the full streamflow record (no negative values)

Q\_Full <- read.csv("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Streamflow\_full.xlsx.csv", col.names=c("DATE","QMeasFull"))

Q\_Full$DATE <- as.Date(Q\_Full$DATE, format="%Y-%m-%d")

Q\_Full <- filter(Q\_Full, QMeasFull!=-9999)

# Obtain the calibration streamflow

BestNS\_S1 <- max(NSResults\_S1$NS,na.rm=TRUE)

BestTrial\_S1 <- which(NSResults\_S1$NS==BestNS\_S1)

BestFolder\_S1 <- NSResults\_S1[BestTrial\_S1,3]

Timezone <- 'etc/GMT-7'

setwd(BestFolder\_S1)

QCal\_S1 <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv",Timezone, missingValueThreshold = 1e-6)

QCal\_S1 <- rename(QCal\_S1, Meas=QOMEAS1, SimCal=QOSIM1)

QCal\_S1 <- select(QCal\_S1, "DATE", "SimCal")

# Combine the streamflows into one "tidy" dataframe

Q\_S1 <- merge(QCal\_S1, Q\_Full, by="DATE", all.x=TRUE)

Q\_S1\_Plot <- gather(Q\_S1, SimCal, QMeasFull, key="ObsOrSim", value="Streamflow")

Q\_S1\_Plot <- filter(Q\_S1\_Plot, !is.na(Streamflow))

Q\_S1\_Plot <- mutate(Q\_S1\_Plot,FakeDate=fakeDate(DATE,fakeYear=2000))

Q\_S1\_Plot <- mutate(Q\_S1\_Plot, LineWt=ifelse(Q\_S1\_Plot$ObsOrSim=="QMeasFull",0.7,0.8), Legend=ifelse(ObsOrSim=="QMeasFull","Measured","Simulated"))

S1Hgraph <- ggplot(data=Q\_S1\_Plot) +

geom\_line(mapping=aes(x=DATE, y=Streamflow, color=Legend), size=Q\_S1\_Plot$LineWt)+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date") +

labs(title="Scenario 1 Best Calibrated Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

scale\_fill\_discrete(name="Legend", labels=c("Measured", "Simulated"))+

geom\_rect(aes(xmin=as.Date("2006-09-15"), xmax=as.Date("2007-09-14"), ymin=-1, ymax=10),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-1, ymax=10),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-1, ymax=10),colour="grey50", linetype=2, fill=NA) +

annotate("text", label="Spin-up", x=as.Date("2007-03-15"), y=11, size=3.5) +

annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=11, size=3.5) +

annotate("text", label="Validation", x=as.Date("2012-03-15"), y=11, size=3.5) +

annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=11, size=3.5)+

annotate("text", label="Validation", x=as.Date("2016-07-15"), y=11, size=3.5) +

ylim(-1,30)+

annotate("text", label="Calibration Period", x=as.Date("2015-03-15"), y=29, size=3.5)+

annotate("text", label=paste("NSE= ",round(NSRanked\_S1$NS[1],digits=2), sep=""), x=as.Date("2015-03-15"), y=27, size=3.5)+

annotate("text", label="Validation Period", x=as.Date("2015-03-15"), y=25, size=3.5)+

annotate("text", label=paste("NSE= ",round(ValNSES1$NSE[1],digits=2), sep=""), x=as.Date("2015-03-15"), y=23, size=3.5)+

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2016-09-14"), ymin=21, ymax=30),colour="black", linetype=1, fill=NA)+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))+

scale\_x\_date(date\_breaks = "1 year", date\_labels = "%Y")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

# xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(0,5)

S1Hgraph

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("S1Hydrograph.jpg", plot=S1Hgraph, width=17.75, height=9, units="cm")

# LogQPlot\_S1 <- ggplot(data=Q\_S1\_Plot) +

# geom\_line(mapping=aes(x=DATE, y=Streamflow, color=ObsOrSim))+

# ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

# xlab("Date") +

# labs(title="Scenario 1 Calibration Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

# geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-1, ymax=10),colour="black", fill=NA) +

# annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=12, size=3.5) +

# geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-1, ymax=10),colour="black", fill=NA, ) +

# annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=12, size=3.5)+

# scale\_y\_log10()

#

# LogQPlot\_S1

```

```{r, eval=FALSE}

### Scenario 1\_2 Streamflow Plot

# Obtain the full streamflow record (no negative values)

Q\_Full <- read.csv("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Streamflow\_full.xlsx.csv", col.names=c("DATE","QMeasFull"))

Q\_Full$DATE <- as.Date(Q\_Full$DATE, format="%Y-%m-%d")

Q\_Full <- filter(Q\_Full, QMeasFull!=-9999)

# Obtain the calibration streamflow

BestNS\_S1\_2 <- max(NSResults\_S1\_2$NS,na.rm=TRUE)

BestTrial\_S1\_2 <- which(NSResults\_S1\_2$NS==BestNS\_S1\_2)

BestFolder\_S1\_2 <- NSResults\_S1\_2[BestTrial\_S1\_2,3]

Timezone <- 'etc/GMT-7'

setwd(BestFolder\_S1\_2)

QCal\_S1\_2 <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv",Timezone, missingValueThreshold = 1e-6)

QCal\_S1\_2 <- rename(QCal\_S1\_2, Meas=QOMEAS1, SimCal=QOSIM1)

QCal\_S1\_2 <- select(QCal\_S1\_2, "DATE", "SimCal")

# Combine the streamflows into one "tidy" dataframe

Q\_S1\_2 <- merge(QCal\_S1\_2, Q\_Full, by="DATE", all.x=TRUE)

Q\_S1\_2\_Plot <- gather(Q\_S1\_2, SimCal, QMeasFull, key="ObsOrSim", value="Streamflow")

Q\_S1\_2\_Plot <- filter(Q\_S1\_2\_Plot, !is.na(Streamflow))

Q\_S1\_2\_Plot <- mutate(Q\_S1\_2\_Plot,FakeDate=fakeDate(DATE,fakeYear=2000))

Q\_S1\_2\_Plot <- mutate(Q\_S1\_2\_Plot, LineWt=ifelse(Q\_S1\_2\_Plot$ObsOrSim=="QMeasFull",0.7,0.8), Legend=ifelse(ObsOrSim=="QMeasFull","Measured","Simulated"))

S1\_2Hgraph <- ggplot(data=Q\_S1\_2\_Plot) +

geom\_line(mapping=aes(x=DATE, y=Streamflow, color=Legend), size=Q\_S1\_2\_Plot$LineWt)+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date") +

labs(title="Scenario 1\_2 Best Calibrated Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

scale\_fill\_discrete(name="Legend", labels=c("Measured", "Simulated"))+

geom\_rect(aes(xmin=as.Date("2006-09-15"), xmax=as.Date("2007-09-14"), ymin=-1, ymax=10),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-1, ymax=10),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-1, ymax=10),colour="grey50", linetype=2, fill=NA) +

annotate("text", label="Spin-up", x=as.Date("2007-03-15"), y=11, size=3.5) +

annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=11, size=3.5) +

annotate("text", label="Validation", x=as.Date("2012-03-15"), y=11, size=3.5) +

annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=11, size=3.5)+

annotate("text", label="Validation", x=as.Date("2016-07-15"), y=11, size=3.5) +

ylim(-1,30)+

annotate("text", label="Calibration Period", x=as.Date("2015-03-15"), y=29, size=3.5)+

annotate("text", label=paste("NSE= ",round(NSRanked\_S1\_2$NS[1],digits=2), sep=""), x=as.Date("2015-03-15"), y=27, size=3.5)+

annotate("text", label="Validation Period", x=as.Date("2015-03-15"), y=25, size=3.5)+

annotate("text", label=paste("NSE= ",round(ValNSES1\_2$NSE[1],digits=2), sep=""), x=as.Date("2015-03-15"), y=23, size=3.5)+

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2016-09-14"), ymin=21, ymax=30),colour="black", linetype=1, fill=NA)+

scale\_x\_date(date\_breaks = "1 year", date\_labels = "%Y")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

S1\_2Hgraph

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("S1\_2Hydrograph.jpg", plot=S1\_2Hgraph, width=17.75, height=9, units="cm")

# xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(0,5)+

```

```{r}

### Scenario 1\_3 Streamflow Plot

# Obtain the full streamflow record (no negative values)

Q\_Full <- read.csv("F:/ECCC\_Project/MESH Model/Baker Creek Model Files/Streamflow\_full.xlsx.csv", col.names=c("DATE","QMeasFull"))

Q\_Full$DATE <- as.Date(Q\_Full$DATE, format="%Y-%m-%d")

Q\_Full <- filter(Q\_Full, QMeasFull!=-9999)

# Obtain the calibration streamflow

BestNS\_S1\_3 <- max(NSResults\_S1\_3$NS,na.rm=TRUE)

BestTrial\_S1\_3 <- which(NSResults\_S1\_3$NS==BestNS\_S1\_3)

BestFolder\_S1\_3 <- NSResults\_S1\_3[BestTrial\_S1\_3,3]

Timezone <- 'etc/GMT-7'

setwd(BestFolder\_S1\_3)

QCal\_S1\_3 <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv",Timezone, missingValueThreshold = 1e-6)

QCal\_S1\_3 <- rename(QCal\_S1\_3, Meas=QOMEAS1, SimCal=QOSIM1)

QCal\_S1\_3 <- select(QCal\_S1\_3, "DATE", "SimCal")

# Combine the streamflows into one "tidy" dataframe

Q\_S1\_3 <- merge(QCal\_S1\_3, Q\_Full, by="DATE", all.x=TRUE)

Q\_S1\_3\_Plot <- gather(Q\_S1\_3, SimCal, QMeasFull, key="ObsOrSim", value="Streamflow")

Q\_S1\_3\_Plot <- filter(Q\_S1\_3\_Plot, !is.na(Streamflow))

Q\_S1\_3\_Plot <- mutate(Q\_S1\_3\_Plot,FakeDate=fakeDate(DATE,fakeYear=2000))

Q\_S1\_3\_Plot <- mutate(Q\_S1\_3\_Plot, LineWt=ifelse(Q\_S1\_3\_Plot$ObsOrSim=="QMeasFull",0.7,0.8), Legend=ifelse(ObsOrSim=="QMeasFull","Measured","Simulated"))

S1\_3Hgraph <- ggplot(data=Q\_S1\_3\_Plot) +

geom\_line(mapping=aes(x=DATE, y=Streamflow, color=Legend), size=Q\_S1\_3\_Plot$LineWt)+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date") +

labs(title="Scenario 1 Best Calibrated Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

scale\_fill\_discrete(name="Legend", labels=c("Measured", "Simulated"))+

geom\_rect(aes(xmin=as.Date("2006-09-15"), xmax=as.Date("2007-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

annotate("text", label="Spin-up", x=as.Date("2007-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=5.5, size=3.2) +

annotate("text", label="Validation", x=as.Date("2012-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=5.5, size=3.2)+

annotate("text", label="Validation", x=as.Date("2016-07-15"), y=5.5, size=3.2) +

ylim(-0.5,10)+

annotate("text", label=paste("Calibration NSE = ",round(NSRanked\_S1\_3$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7.75, size=3.2)+

annotate("text", label=paste("Validation NSE = ",round(ValNSES1\_3$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7, size=3.2)+

geom\_rect(aes(xmin=as.Date("2013-01-01"), xmax=as.Date("2016-01-01"), ymin=6.5, ymax=8.25),colour="black", linetype=1, fill=NA)+

scale\_x\_date(date\_breaks = "1 year", date\_labels = "%Y")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

# annotate("text", label="Calibration Period", x=as.Date("2015-03-15"), y=16, size=3.2)+

# annotate("text", label=paste("NSE = ",round(NSRanked\_S1\_3$NS[1],digits=2), sep=""), x=as.Date("2015-03-15"), y=15, size=3.2)+

# annotate("text", label="Validation Period", x=as.Date("2015-03-15"), y=14, size=3.2)+

# annotate("text", label=paste("NSE = ",round(ValNSES1\_3$NSE[1],digits=2), sep=""), x=as.Date("2015-03-15"), y=13, size=3.2)+

S1\_3Hgraph

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("S1\_3Hydrograph.jpg", plot=S1\_3Hgraph, width=17.75, height=9, units="cm")

# xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(0,5)+

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 2 Streamflow Plot

####

# Obtain the full streamflow record (no negative values)

# See above

# Obtain the calibration streamflow

BestNS\_S2 <- max(NSResults\_S2$NS,na.rm=TRUE)

BestTrial\_S2 <- which(NSResults\_S2$NS==BestNS\_S2)

BestFolder\_S2 <- NSResults\_S2[BestTrial\_S2,3]

Timezone <- 'etc/GMT-7'

setwd(BestFolder\_S2)

QCal\_S2 <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv",Timezone, missingValueThreshold = 1e-6)

QCal\_S2 <- rename(QCal\_S2, Meas=QOMEAS1, SimCal=QOSIM1)

QCal\_S2 <- select(QCal\_S2, "DATE", "SimCal")

# Combine the streamflows into one "tidy" dataframe

Q\_S2 <- merge(QCal\_S2, Q\_Full, by="DATE", all.x=TRUE)

Q\_S2\_Plot <- gather(Q\_S2, SimCal, QMeasFull, key="ObsOrSim", value="Streamflow")

Q\_S2\_Plot <- filter(Q\_S2\_Plot, !is.na(Streamflow))

Q\_S2\_Plot <- mutate(Q\_S2\_Plot,FakeDate=fakeDate(DATE,fakeYear=2000))

Q\_S2\_Plot <- mutate(Q\_S2\_Plot, LineWt=ifelse(Q\_S2\_Plot$ObsOrSim=="QMeasFull",0.7,0.8), Legend=ifelse(ObsOrSim=="QMeasFull","Measured","Simulated"))

S2Hgraph <- ggplot(data=Q\_S2\_Plot) +

geom\_line(mapping=aes(x=DATE, y=Streamflow, color=Legend), size=Q\_S2\_Plot$LineWt)+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date") +

labs(title="Scenario 2 Best Calibrated Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

scale\_fill\_discrete(name="Legend", labels=c("Measured", "Simulated"))+

geom\_rect(aes(xmin=as.Date("2006-09-15"), xmax=as.Date("2007-09-14"), ymin=-0.5, ymax=8),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

annotate("text", label="Spin-up", x=as.Date("2007-03-15"), y=8.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=5.5, size=3.2) +

annotate("text", label="Validation", x=as.Date("2012-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=5.5, size=3.2)+

annotate("text", label="Validation", x=as.Date("2016-07-15"), y=5.5, size=3.2) +

ylim(-0.5,10)+

annotate("text", label=paste("Calibration NSE = ",round(NSRanked\_S2$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7.75, size=3.2)+

annotate("text", label=paste("Validation NSE = ",round(ValNSES2$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7, size=3.2)+

geom\_rect(aes(xmin=as.Date("2013-01-01"), xmax=as.Date("2016-01-01"), ymin=6.5, ymax=8.25),colour="black", linetype=1, fill=NA)+

scale\_x\_date(date\_breaks = "1 year", date\_labels = "%Y")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

S2Hgraph

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("S2Hydrograph.jpg", plot=S2Hgraph, width=17.75, height=9, units="cm")

# xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(0,5)

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 3 Streamflow Plot

####

# Obtain the full streamflow record (no negative values)

# Q\_Full

# Obtain the calibration streamflow

BestNS\_S3 <- max(NSResults\_S3$NS,na.rm=TRUE)

BestTrial\_S3 <- which(NSResults\_S3$NS==BestNS\_S3)

BestFolder\_S3 <- NSResults\_S3[BestTrial\_S3,3]

Timezone <- 'etc/GMT-7'

setwd(BestFolder\_S3)

QCal\_S3 <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv",Timezone, missingValueThreshold = 1e-6)

QCal\_S3 <- rename(QCal\_S3, Meas=QOMEAS1, SimCal=QOSIM1)

QCal\_S3 <- select(QCal\_S3, "DATE", "SimCal")

# Combine the streamflows into one "tidy" dataframe

Q\_S3 <- merge(QCal\_S3, Q\_Full, by="DATE", all.x=TRUE)

Q\_S3\_Plot <- gather(Q\_S3, SimCal, QMeasFull, key="ObsOrSim", value="Streamflow")

Q\_S3\_Plot <- filter(Q\_S3\_Plot, !is.na(Streamflow))

Q\_S3\_Plot <- mutate(Q\_S3\_Plot,FakeDate=fakeDate(DATE,fakeYear=2000))

Q\_S3\_Plot <- mutate(Q\_S3\_Plot, LineWt=ifelse(Q\_S3\_Plot$ObsOrSim=="QMeasFull",0.7,0.8), Legend=ifelse(ObsOrSim=="QMeasFull","Measured","Simulated"))

S3Hgraph <- ggplot(data=Q\_S3\_Plot) +

geom\_line(mapping=aes(x=DATE, y=Streamflow, color=Legend), size=Q\_S3\_Plot$LineWt)+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date") +

labs(title="Scenario 3 Best Calibrated Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

scale\_fill\_discrete(name="Legend", labels=c("Measured", "Simulated"))+

geom\_rect(aes(xmin=as.Date("2006-09-15"), xmax=as.Date("2007-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

annotate("text", label="Spin-up", x=as.Date("2007-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=5.5, size=3.2) +

annotate("text", label="Validation", x=as.Date("2012-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=5.5, size=3.2)+

annotate("text", label="Validation", x=as.Date("2016-07-15"), y=5.5, size=3.2) +

ylim(-0.5,10)+

annotate("text", label=paste("Calibration NSE = ",round(NSRanked\_S3$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7.75, size=3.2)+

annotate("text", label=paste("Validation NSE = ",round(ValNSES3$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7, size=3.2)+

geom\_rect(aes(xmin=as.Date("2013-01-01"), xmax=as.Date("2016-01-01"), ymin=6.5, ymax=8.25),colour="black", linetype=1, fill=NA)+

scale\_x\_date(date\_breaks = "1 year", date\_labels = "%Y")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

S3Hgraph

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("S3Hydrograph.jpg", plot=S3Hgraph, width=17.75, height=9, units="cm")

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 1P Streamflow Plot

####

# Obtain the full streamflow record (no negative values)

# Q\_Full

# Obtain the calibration streamflow

BestNS\_S1P <- max(NSResults\_S1P$NS,na.rm=TRUE)

BestTrial\_S1P <- which(NSResults\_S1P$NS==BestNS\_S1P)

BestFolder\_S1P <- NSResults\_S1P[BestTrial\_S1P,3]

Timezone <- 'etc/GMT-7'

setwd(BestFolder\_S1P)

QCal\_S1P <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv",Timezone, missingValueThreshold = 1e-6)

QCal\_S1P <- rename(QCal\_S1P, Meas=QOMEAS1, SimCal=QOSIM1)

QCal\_S1P <- select(QCal\_S1P, "DATE", "SimCal")

# Combine the streamflows into one "tidy" dataframe

Q\_S1P <- merge(QCal\_S1P, Q\_Full, by="DATE", all.x=TRUE)

Q\_S1P\_Plot <- gather(Q\_S1P, SimCal, QMeasFull, key="ObsOrSim", value="Streamflow")

Q\_S1P\_Plot <- filter(Q\_S1P\_Plot, !is.na(Streamflow))

Q\_S1P\_Plot <- mutate(Q\_S1P\_Plot,FakeDate=fakeDate(DATE,fakeYear=2000))

Q\_S1P\_Plot <- mutate(Q\_S1P\_Plot, LineWt=ifelse(Q\_S1P\_Plot$ObsOrSim=="QMeasFull",0.7,0.8), Legend=ifelse(ObsOrSim=="QMeasFull","Measured","Simulated"))

S1PHgraph <- ggplot(data=Q\_S1P\_Plot) +

geom\_line(mapping=aes(x=DATE, y=Streamflow, color=Legend), size=Q\_S1P\_Plot$LineWt)+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date") +

labs(title="Scenario 1-P Best Calibrated Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

scale\_fill\_discrete(name="Legend", labels=c("Measured", "Simulated"))+

geom\_rect(aes(xmin=as.Date("2006-09-15"), xmax=as.Date("2007-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

annotate("text", label="Spin-up", x=as.Date("2007-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=5.5, size=3.2) +

annotate("text", label="Validation", x=as.Date("2012-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=5.5, size=3.2)+

annotate("text", label="Validation", x=as.Date("2016-07-15"), y=5.5, size=3.2) +

ylim(-0.5,10)+

annotate("text", label=paste("Calibration NSE = ",round(NSRanked\_S1P$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7.75, size=3.2)+

annotate("text", label=paste("Validation NSE = ",round(ValNSES1P$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7, size=3.2)+

geom\_rect(aes(xmin=as.Date("2013-01-01"), xmax=as.Date("2016-01-01"), ymin=6.5, ymax=8.25),colour="black", linetype=1, fill=NA)+

scale\_x\_date(date\_breaks = "1 year", date\_labels = "%Y")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

S1PHgraph

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("S1PHydrograph.jpg", plot=S1PHgraph, width=17.75, height=9, units="cm")

```

```{r}

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Scenario 2P Streamflow Plot

####

# Obtain the full streamflow record (no negative values)

# Q\_Full

# Obtain the calibration streamflow

BestNS\_S2P <- max(NSResults\_S2P$NS,na.rm=TRUE)

BestTrial\_S2P <- which(NSResults\_S2P$NS==BestNS\_S2P)

BestFolder\_S2P <- NSResults\_S2P[BestTrial\_S2P,3]

Timezone <- 'etc/GMT-7'

setwd(BestFolder\_S2P)

QCal\_S2P <- read\_MESH\_OutputTimeseries\_csv("MESH\_output\_streamflow.csv",Timezone, missingValueThreshold = 1e-6)

QCal\_S2P <- rename(QCal\_S2P, Meas=QOMEAS1, SimCal=QOSIM1)

QCal\_S2P <- select(QCal\_S2P, "DATE", "SimCal")

# Combine the streamflows into one "tidy" dataframe

Q\_S2P <- merge(QCal\_S2P, Q\_Full, by="DATE", all.x=TRUE)

Q\_S2P\_Plot <- gather(Q\_S2P, SimCal, QMeasFull, key="ObsOrSim", value="Streamflow")

Q\_S2P\_Plot <- filter(Q\_S2P\_Plot, !is.na(Streamflow))

Q\_S2P\_Plot <- mutate(Q\_S2P\_Plot,FakeDate=fakeDate(DATE,fakeYear=2000))

Q\_S2P\_Plot <- mutate(Q\_S2P\_Plot, LineWt=ifelse(Q\_S2P\_Plot$ObsOrSim=="QMeasFull",0.7,0.8), Legend=ifelse(ObsOrSim=="QMeasFull","Measured","Simulated"))

S2PHgraph <- ggplot(data=Q\_S2P\_Plot) +

geom\_line(mapping=aes(x=DATE, y=Streamflow, color=Legend), size=Q\_S2P\_Plot$LineWt)+

ylab(expression(paste("Streamflow (m", ""^{3}, "/s)", sep = ""))) +

xlab("Date") +

labs(title="Scenario 2-P Best Calibrated Streamflow", subtitle="MESH Model, Baker Creek Watershed") +

scale\_fill\_discrete(name="Legend", labels=c("Measured", "Simulated"))+

geom\_rect(aes(xmin=as.Date("2006-09-15"), xmax=as.Date("2007-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2007-09-15"), xmax=as.Date("2010-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

geom\_rect(aes(xmin=as.Date("2013-09-15"), xmax=as.Date("2015-09-14"), ymin=-0.5, ymax=5),colour="grey50", linetype=2, fill=NA) +

annotate("text", label="Spin-up", x=as.Date("2007-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2009-03-15"), y=5.5, size=3.2) +

annotate("text", label="Validation", x=as.Date("2012-03-15"), y=5.5, size=3.2) +

annotate("text", label="Calibration", x=as.Date("2014-09-15"), y=5.5, size=3.2)+

annotate("text", label="Validation", x=as.Date("2016-07-15"), y=5.5, size=3.2) +

ylim(-0.5,10)+

annotate("text", label=paste("Calibration NSE = ",round(NSRanked\_S2P$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7.75, size=3.2)+

annotate("text", label=paste("Validation NSE = ",round(ValNSES2P$NS[1],digits=2), sep=""), x=as.Date("2014-06-15"), y=7, size=3.2)+

geom\_rect(aes(xmin=as.Date("2013-01-01"), xmax=as.Date("2016-01-01"), ymin=6.5, ymax=8.25),colour="black", linetype=1, fill=NA)+

scale\_x\_date(date\_breaks = "1 year", date\_labels = "%Y")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

S2PHgraph

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("S2PHydrograph.jpg", plot=S2PHgraph, width=17.75, height=9, units="cm")

```

## Water Balance Plots for the Best Calibration Runs

```{r}

WBFile <- "Basin\_average\_water\_balance.csv"

# BestFolder\_S1 <- NSRanked\_S1$SubFolder[1]

# setwd(BestFolder\_S1)

# WBOut\_S1 <- read\_MESH\_OutputTimeseries\_csv(WBFile, missingValueThreshold = -1e6)

# WB\_S1 <- MESHr::basinWaterBalancePlot(WBOut\_S1)

# SNO\_S1 <- MESHr::basinSnowPlot(WBOut\_S1)

# WB\_S1 + labs(title="Basin Water Balance for the Best Calibration Run - Scenario 1", subtitle="MESH Model, Baker Creek Watershed")+

# # xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(-1000,1500)

#

# SNO\_S1

#

# BestFolder\_S1\_2 <- NSRanked\_S1\_2$SubFolder[1]

# setwd(BestFolder\_S1\_2)

# WBOut\_S1\_2 <- read\_MESH\_OutputTimeseries\_csv(WBFile, missingValueThreshold = -1e6)

# WB\_S1\_2 <- MESHr::basinWaterBalancePlot(WBOut\_S1\_2)

# WB\_S1\_2 + labs(title="Basin Water Balance for the Best Calibration Run - Scenario 1\_2", subtitle="MESH Model, Baker Creek Watershed")+

# xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(-500,1500)

BestFolder\_S1\_3 <- NSRanked\_S1\_3$SubFolder[1]

setwd(BestFolder\_S1\_3)

WBOut\_S1\_3 <- read\_MESH\_OutputTimeseries\_csv(WBFile, missingValueThreshold = -1e6)

WB\_S1\_3 <- MESHr::basinWaterBalancePlot(WBOut\_S1\_3)

WB\_S1\_3 + labs(title="Basin Water Balance for the Best Calibration Run - Scenario 1", subtitle="MESH Model, Baker Creek Watershed")

# xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(-500,1500)

BestFolder\_S2 <- NSRanked\_S2$SubFolder[1]

setwd(BestFolder\_S2)

WBOut\_S2 <- read\_MESH\_OutputTimeseries\_csv(WBFile, missingValueThreshold = -1e6)

WB\_S2 <- MESHr::basinWaterBalancePlot(WBOut\_S2)

WB\_S2 + labs(title="Basin Water Balance for the Best Calibration Run - Scenario 2", subtitle="MESH Model, Baker Creek Watershed")

# xlim(as.Date("2007-09-15"), as.Date("2009-09-15"))+

# ylim(-500,1500)

BestFolder\_S3 <- NSRanked\_S3$SubFolder[1]

setwd(BestFolder\_S3)

WBOut\_S3 <- read\_MESH\_OutputTimeseries\_csv(WBFile, missingValueThreshold = -1e6)

WB\_S3 <- MESHr::basinWaterBalancePlot(WBOut\_S3)

WB\_S3 + labs(title="Basin Water Balance for the Best Calibration Run - Scenario 3", subtitle="MESH Model, Baker Creek Watershed")

BestFolder\_S1P <- NSRanked\_S1P$SubFolder[1]

setwd(BestFolder\_S1P)

WBOut\_S1P <- read\_MESH\_OutputTimeseries\_csv(WBFile, missingValueThreshold = -1e6)

WB\_S1P <- MESHr::basinWaterBalancePlot(WBOut\_S1P)

WB\_S1P + labs(title="Basin Water Balance for the Best Calibration Run - Scenario 1-P", subtitle="MESH Model, Baker Creek Watershed")

BestFolder\_S2P <- NSRanked\_S2P$SubFolder[1]

setwd(BestFolder\_S2P)

WBOut\_S2P <- read\_MESH\_OutputTimeseries\_csv(WBFile, missingValueThreshold = -1e6)

WB\_S2P <- MESHr::basinWaterBalancePlot(WBOut\_S2P)

WB\_S2P + labs(title="Basin Water Balance for the Best Calibration Run - Scenario 2-P", subtitle="MESH Model, Baker Creek Watershed")

```

### Q.How did the top 10% of calibration runs compare to the validation runs?

Create a box-whisker plot of the NSE values vs each calibration and validation period for the top 10% (10/100) calibration runs

```{r}

NSTop10 <- NSRanked\_S1\_3[c(1:10),2]

# NSTop10 <- cbind(NSTop10, ValNSES1[c(1:10),2])

# NSTop10 <- cbind(NSTop10, NSRanked\_S1\_2[c(1:10),2])

# NSTop10 <- cbind(NSTop10, ValNSES1\_2[c(1:10),2])

# NSTop10 <- cbind(NSTop10, NSRanked\_S1\_3[c(1:10),2])

NSTop10 <- cbind(NSTop10, ValNSES1\_3[c(1:10),2])

NSTop10 <- cbind(NSTop10, NSRanked\_S2[c(1:10),2])

NSTop10 <- cbind(NSTop10, ValNSES2[c(1:10),2])

NSTop10 <- cbind(NSTop10, NSRanked\_S3[c(1:10),2])

NSTop10 <- cbind(NSTop10, ValNSES3[c(1:10),2])

NSTop10 <- cbind(NSTop10, NSRanked\_S1P[c(1:10),2])

NSTop10 <- cbind(NSTop10, ValNSES1P[c(1:10),2])

NSTop10 <- cbind(NSTop10, NSRanked\_S2P[c(1:10),2])

NSTop10 <- cbind(NSTop10, ValNSES2P[c(1:10),2])

NSTop10 <- as.data.frame(NSTop10)

colnames(NSTop10) <- c("S1\_Cal", "S1\_Val","S2\_Cal", "S2\_Val","S3\_Cal", "S3\_Val", "S1P\_Cal", "S1P\_Val", "S2P\_Cal", "S2P\_Val")

Top10Box <- ggplot(stack(NSTop10), aes(x = ind, y = values)) +

geom\_boxplot()+

labs(title="Performance of the Top 10% of Calibration Runs", subtitle="MESH Model, Baker Creek Watershed")+

ylab("Nash-Sutcliffe Efficiency")+

xlab(NULL)+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

Top10Box

setwd("F:/ECCC\_Project/Report/MWSCapstoneReport/figures")

ggsave("Top10Box.jpg", plot=Top10Box, width=17.75, height=9, units="cm")

```

### Ranked model performance for 100 calibration runs

```{r}

AllRanked <- data.frame(Runs=c(100:1))

# AllRanked <- cbind(AllRanked, NSRanked\_S1$NS)

# AllRanked <- cbind(AllRanked, NSRanked\_S1\_2$NS)

AllRanked <- cbind(AllRanked, NSRanked\_S1\_3$NS)

AllRanked <- cbind(AllRanked, NSRanked\_S2$NS)

AllRanked <- cbind(AllRanked, NSRanked\_S3$NS)

AllRanked <- cbind(AllRanked, NSRanked\_S1P$NS)

AllRanked <- cbind(AllRanked, NSRanked\_S2P$NS)

colnames(AllRanked) <- c("Runs", "S1", "S2", "S3", "S1P", "S2P")

AllRanked <- gather(AllRanked, S1, S2, S3, S1P, S2P, key="Scenario", value="NS")

AllRanked$LnTyp <- ifelse(AllRanked$Scenario=="S1","a",ifelse(AllRanked$Scenario=="S2","b","d"))

AllRanked$Scenario <- factor(AllRanked$Scenario, levels=c("S1P", "S2P", "S3", "S2", "S1"))

RankedPlot <- ggplot()+

geom\_line(data=AllRanked, mapping=aes(x=Runs, y=NS, color=Scenario, linetype=Scenario), size=1)+

labs(title="Ranked Model Calibration Performance", subtitle="MESH Model, Baker Creek Watershed", x="Ranked Model Calibration Runs", y="Nash-Sutcliffe Value")+

scale\_x\_continuous(breaks=seq(0,100, by=10))+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

RankedPlot

ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/RankedPerf.jpg", plot=RankedPlot, width=17.75, height=9, unit="cm")

```

### Q. What was the tradeoff between model run time and model performance?

Create a plot showing the NSE and the run time for each configuration. Does it compare with Herbert's results?

```{r}

#Plot calibration NSE, Validation NSE, and run time (2 y-axis)

####Need to update with which Scenario 1 Calibration was actually used, as well as the PDMROF results

library(ggpubr)

Perf <- data.frame(Scenario=c(1,2,3,"1P", "2P"), Calibration = c(NSRanked\_S1\_3$NS[1],NSRanked\_S2$NS[1],NSRanked\_S3$NS[1], NSRanked\_S1P$NS[1], NSRanked\_S2P$NS[1]), Validation=c(ValNSES1\_3$NSE[1], ValNSES2$NSE[1],ValNSES3$NSE[1], ValNSES1P$NSE[1], ValNSES2P$NSE[1]), AvgTime=c(4.714,9.862,35.711, "7.036", "12.402"))

#Times taken from the start time at the beginning of one of the slurm files to the time of the last-finished trial divided by the number of trials per folder

Perf\_Plot <- gather(Perf, "Calibration", "Validation", key="Performance", value="NS")

### CAN'T GET THE PLOTS TO WORK OUT; PLOTTED IN EXCEL INSTEAD

# brp <- ggplot()+

# geom\_bar(data=Perf\_Plot, mapping=aes(x=Scenario, y=NS, fill=Performance), stat="identity", position=position\_dodge())+

# ylim(-0.5,0.5)+

# scale\_fill\_discrete(name=NULL)+

# ylab("Best Nash-Sutcliffe Value")+

# labs(title="Model Performance and Run Time", subtitle="MESH Model, Baker Creek Watershed")+

# theme(legend.position="top")

#

# lnp <- ggplot()+

# # geom\_line(data=Perf, mapping=aes(x=Scenario, y=AvgTime))+

# geom\_point(data=Perf, mapping=aes(x=Scenario, y=AvgTime))+

# # scale\_x\_discrete(name="Scenario")+

# # xlim("1", "1-P", "2", "2-P", "3")+

# ylab("Average Model Run Time (1 Calibration)")+

# theme(legend.position="bottom")

#

# xform <- list(categoryorder = "array", categoryarray=c(1,2,3,"1P", "2P"))

#

# plot\_ly(data=Perf, x=~Scenario) %>%

# add\_trace(y = ~Calibration, type = 'bar', name = "Calibration")%>%

# add\_trace(y = ~Validation, type = 'bar', name = "Validation")%>%

# add\_lines(y = ~AvgTime, name = 'Simulation Time', yaxis = 'y2', line = list(color = '#45171D')) %>%

# layout(title = 'Model Performance versus Simulation Time',

# xaxis = xform,

# yaxis = list(side = 'left', title = 'Nash-Sutcliffe Value', showgrid = TRUE, zeroline = TRUE),

# yaxis2 = list(side = 'right', overlaying = "y", title = 'Average Simulation Time (minutes)', showgrid = FALSE, zeroline = FALSE))

#

# type='scatter', mode = 'lines+markers',

#

# brpAll

#

# ggplot()+

# geom\_bar(data=Perf\_Plot, mapping=aes(x=Scenario, y=NS, fill=Performance), stat="identity", position=position\_dodge())+

# ylim(-0.5,0.5)+

# scale\_fill\_discrete(name=NULL)+

# geom\_point(data=Perf, mapping=aes(x=Scenario, y=AvgTime/100))+

# scale\_y\_continuous(sec.axis = sec\_axis(~./100, name = "Average Simulation Time (minutes)"))+

# # ylab("Best Nash-Sutcliffe Value")+

# labs(title="Model Performance and Run Time", subtitle="MESH Model, Baker Creek Watershed")+

# theme(legend.position="top")

#

# brp

# lnp

#

# ggarrange(brp, lnp, ncol=1, nrow=2, heights=c(2, 0.7), align="v")

```

### Plot the identifiability for each calibrated parameter for each scenario

Plot normalized parameter range vs parameter (see notes as well as Herbert's report) for each configuration; be sure to add a straight line for the "acceptable identifiability" cutoff.

```{r}

ParamIdent(folder\_S1\_3, 1, 8) #Resulting OstBest files are called "S1"

ParamIdent(folder\_S2, 2, 6)

ParamIdent(folder\_S3, 3, 6)

ParamIdent(folder\_S1P, "1P", 8)

ParamIdent(folder\_S2P, "2P", 6)

```

```{r}

OstBestNorm\_S1$Param\_Names <- gsub(" ", "", OstBestNorm\_S1$Param\_Names)

OstBestNorm\_S2$Param\_Names <- gsub(" ", "", OstBestNorm\_S2$Param\_Names)

OstBestNorm\_S3$Param\_Names <- gsub(" ", "", OstBestNorm\_S3$Param\_Names)

OstBestNorm\_S1P$Param\_Names <- gsub(" ", "", OstBestNorm\_S1P$Param\_Names)

OstBestNorm\_S2P$Param\_Names <- gsub(" ", "", OstBestNorm\_S2P$Param\_Names)

# Scenario 1 ----------------------------------------------------------------------------

IdenS1 <- ggplot(OstBestNorm\_S1,aes(x=Param\_Names, y=Diff, group=1))+

geom\_point(stat='summary', fun.y=sum)+

stat\_summary(fun.y=sum, geom="line")+

coord\_cartesian(ylim=c(0,1))+

scale\_y\_continuous(breaks=seq(0,1,0.1))+

geom\_line(mapping=aes(y=0.3), linetype=2)+

theme(axis.text.x=element\_text(angle=90, size=8, hjust=0.95, vjust=0.2), axis.title.y=element\_text(size=10))+

labs(y="Parameter Identifiability Range (10th-90th Percentile)", x="Parameter", title="Parameter Identifiability - Scenario 1", subtitle="MESH Model, Baker Creek Watershed")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

# Scenario 2 ----------------------------------------------------------------------------

IdenS2 <- ggplot(OstBestNorm\_S2,aes(x=Param\_Names, y=Diff, group=1))+

geom\_point(stat='summary', fun.y=sum)+

stat\_summary(fun.y=sum, geom="line")+

coord\_cartesian(ylim=c(0,1))+

scale\_y\_continuous(breaks=seq(0,1,0.1))+

geom\_line(mapping=aes(y=0.3), linetype=2)+

theme(axis.text.x=element\_text(angle=90, size=6, hjust=0.95, vjust=0.2), axis.title.y=element\_text(size=10))+

labs(y="Parameter Identifiability Range (10th-90th Percentile)", x="Parameter", title="Parameter Identifiability - Scenario 2", subtitle="MESH Model, Baker Creek Watershed")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

# Scenario 3 ----------------------------------------------------------------------------

IdenS3 <- ggplot(OstBestNorm\_S3,aes(x=Param\_Names, y=Diff, group=1))+

geom\_point(stat='summary', fun.y=sum)+

stat\_summary(fun.y=sum, geom="line")+

coord\_cartesian(ylim=c(0,1))+

scale\_y\_continuous(breaks=seq(0,1,0.1))+

geom\_line(mapping=aes(y=0.3), linetype=2)+

theme(axis.text.x=element\_text(angle=90, size=6, hjust=0.95, vjust=0.2), axis.title.y=element\_text(size=10))+

labs(y="Parameter Identifiability Range (10th-90th Percentile)", x="Parameter", title="Parameter Identifiability - Scenario 3", subtitle="MESH Model, Baker Creek Watershed")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

# Scenario 1P ----------------------------------------------------------------------------

IdenS1P <- ggplot(OstBestNorm\_S1P,aes(x=Param\_Names, y=Diff, group=1))+

geom\_point(stat='summary', fun.y=sum)+

stat\_summary(fun.y=sum, geom="line")+

coord\_cartesian(ylim=c(0,1))+

scale\_y\_continuous(breaks=seq(0,1,0.1))+

geom\_line(mapping=aes(y=0.3), linetype=2)+

theme(axis.text.x=element\_text(angle=90, size=6, hjust=0.95, vjust=0.2), axis.title.y=element\_text(size=10))+

labs(y="Parameter Identifiability Range (10th-90th Percentile)", x="Parameter", title="Parameter Identifiability - Scenario 1-P", subtitle="MESH Model, Baker Creek Watershed")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

# Scenario 2P ----------------------------------------------------------------------------

IdenS2P <- ggplot(OstBestNorm\_S2P,aes(x=Param\_Names, y=Diff, group=1))+

geom\_point(stat='summary', fun.y=sum)+

stat\_summary(fun.y=sum, geom="line")+

coord\_cartesian(ylim=c(0,1))+

scale\_y\_continuous(breaks=seq(0,1,0.1))+

geom\_line(mapping=aes(y=0.3), linetype=2)+

theme(axis.text.x=element\_text(angle=90, size=6, hjust=0.95, vjust=0.2), axis.title.y=element\_text(size=10))+

labs(y="Parameter Identifiability Range (10th-90th Percentile)", x="Parameter", title="Parameter Identifiability - Scenario 2-P", subtitle="MESH Model, Baker Creek Watershed")+

theme(plot.background = element\_rect(colour="black", linetype=1, size=1))

# Save plots

# ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/IdenS1.jpg", plot=IdenS1, width=17.75, height=9, unit="cm")

#

# ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/IdenS2.jpg", plot=IdenS2, width=17.75, height=9, unit="cm")

#

# ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/IdenS3.jpg", plot=IdenS3, width=17.75, height=9, unit="cm")

#

# ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/IdenS1P.jpg", plot=IdenS1P, width=17.75, height=9, unit="cm")

#

# ggsave("F:/ECCC\_Project/Report/MWSCapstoneReport/figures/IdenS2P.jpg", plot=IdenS2P, width=17.75, height=9, unit="cm")

```