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

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=TRU
E),mean(Sept15_avgT$T_Depth_25cm,na.rm=TRUE),mean(Sept15_avgT$T_Depth_surf,na.rm=TRUE),ma
x(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),m
ean(Oct1_avgT$T_Depth_25cm,na.rm=TRUE),mean(Oct1_avgT$T_Depth_surf,na.rm=TRUE),max(Oct1_a
vgT$T_Depth_25cm,na.rm=TRUE),max(Oct1_avgT$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=T
RUE),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&s
tation_number=07sb013&start_year=1850&end_year=2019&minimum_years=&gross_drainage_operat
or=%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")
```
~~~