# Global and regional temperature model comparison

Eddie Wu

2024-07-24

## Introduction

This R markdown file is used to

1. Import and clean global water temperature model

2. Compare the weekly performance of a non-linear model and the global futureStreams river temperature model on a different seasonal scales (spring, summer, fall, annual).

```
library(dplyr)
library(tidyverse)
library(lme4)
library(nlme)
library(xts)
library(ModelMetrics)
library(zoo)
library(lubridate)
library(nls2)
```

```
get.traintest <- function(master.df, year.list, fold) {

  # create output
  combined_training_list <- vector("list",fold)
  combined_testing_list <- vector("list",fold)

  ## Loop 10 folds
  for (f in 1:fold) {

    # dataframe to store the sampled years for each location
    dfind = data.frame(location=character(),
                       year=integer(), stringsAsFactors = FALSE)
    dfindt = data.frame(location=character(),
                        year=integer(), stringsAsFactors = FALSE)

    for (i in 1:length(year.list)){ # i loops through each location
      smp = year.list[[i]] #train
      if (length(smp)>1){
        indx.train=(sample(smp, 1))
        ytrain=indx.train
      } else if (length(smp)==1){
        indx.train=smp
        ytrain=indx.train
      }
```

```r
      smpt = year.list[[i]][year.list[[i]] != indx.train] #test
      if (length(smpt)>1){
        indx.test=(sample(smpt, 1))
        ytest=indx.test
      } else if (length(smpt)==1){
        indx.test=smpt
        ytest=indx.test
      }

      dfind[i,] = c(names(year.list[i]), ytrain)
      dfindt[i,] = c(names(year.list[i]), ytest)
    }

    # subset the dataframe
    awf_train <- merge(master.df, dfind, by=c("location", "year"))
    awf_test <- merge(master.df, dfindt, by=c("location", "year"))

    combined_training_list[[f]] <- awf_train
    combined_testing_list[[f]] <- awf_test
  }
  return(list(combined_training_list, combined_testing_list))
}
good_year <- function(master.df, dayln) {
  # get table of sample years by location
  yr_out=table(master.df$location, master.df$year)

  # select sample years with more than X days (x=250)
  full_year=list()
  sind=vector()
  cnt=0
  ind=0

  for (i in seq_along(loc_seq)){
   rm(ind)
   if (ncol(yr_out)>=1) {
     ind=which(yr_out[row.names(yr_out)==loc_seq[i],]>dayln)

     if (length(ind)>0) {
       sind=c(sind,i)
       cnt=cnt+1
       full_year[[cnt]]=colnames(yr_out)[ind]
     }
   }
  }
  full_year=setNames(full_year,loc_seq[sind])
  print(full_year)
}
```

## Data importing and cleaning

```r
## Data import
air <- read.csv("tributary air temperatures clean.csv", stringsAsFactors=F)
water <- read.csv("tributary water temperature/water_temperature_d.csv",
```

```r
                  stringsAsFactors=F)
flow <- read.csv("tributary discharge.csv", stringsAsFactors=F)

# Convert to date format
air$date <- as.Date(air$date, "%m/%d/%Y")
water$date <- as.Date(water$date, "%m/%d/%Y")
flow$date <- as.Date(flow$date, "%m/%d/%Y")


## Calculate the MEAN airT for US locations
for (i in which(is.na(air$mean_temp))) {
  air$mean_temp <- (air$max_temp + air$min_temp) / 2
}
```

### Data combining

```r
## Combine water temperature and discharge
aw <- merge(air, water, by = c("station_name","date"))
awf <- merge(aw, flow, by = c("location","date"))
awf <- awf %>% arrange(location, date)

## Change into factor
awf$location <- as.factor(awf$location)
awf$station_name <- as.factor(awf$station_name)

# Get location sequence
loc_seq=levels(awf$location)


## Check if there are any duplicates
duplicates <- awf %>%
  group_by(location, date) %>%
  filter(n() > 1) # Duplicates should be NA...


## Print the result
table(awf$location)
```

```
##
##   bigcreek   bigotter        fox    genesee     humber mississagi    nipigon
##       4554        919       1374       1095       4446       1434        848
##    portage    saginaw      still    stlouis  vermilion
##        730       1095       1246       1349       1095
```

### Check for imputed values

Use a 7-days rolling mean to check for possibly imputed temperatures. If the variance of a certain day's temperature is very close to zero, then it is likely that this particular data is imputed.

```r
# make sure that no initial NA values in awf water temperature
which(is.na(awf$temp))
```

```
## integer(0)
```

```r
# Need to calculate the rolling mean for each location separately...
for(loc in loc_seq) {
  sub <- awf[awf$location == loc,]
  sub$rolling_mean <- rollmean(sub$temp, k = 7, fill = NA, align = "right")
  awf[awf$location == loc, "rolling_mean"] <- sub$rolling_mean
}

# Calculate variance
awf$variance <- (awf$temp - awf$rolling_mean)^2

# Assign NA when variance is very small
awf$temp[which(awf$variance < 1e-10)] <- NA

# Check results - how many imputed values are in each location
awf %>%
  group_by(location) %>%
  summarise(na_count = sum(is.na(temp)))
```

```
## # A tibble: 12 x 2
##    location   na_count
##    <fct>         <int>
##  1 bigcreek        101
##  2 bigotter          0
##  3 fox               6
##  4 genesee           6
##  5 humber           12
##  6 mississagi        6
##  7 nipigon           0
##  8 portage           2
##  9 saginaw          17
## 10 still           226
## 11 stlouis         144
## 12 vermilion       137
```

### Get lagged days

```r
## Get the time lag day variables
awf <- awf %>%
  group_by(location) %>%
  mutate(dmean_1 = lag(mean_temp, 1),
         dmean_2 = lag(mean_temp, 2),
         dmean_3 = lag(mean_temp, 3),
         dmean_4 = lag(mean_temp, 4),
         dmean_5 = lag(mean_temp, 5),
         dflow_1 = lag(flow, 1))


## Get cumulative air temp for past five days
awf <- awf %>%
  rowwise() %>%
  mutate(cair = (mean_temp+dmean_1+dmean_2+dmean_3+dmean_4+dmean_5)/6)

## Get relative flow
awf <- awf %>% mutate(rqc = (flow - dflow_1)/flow)
```

### Get final master temp dataframe

```
master.temp <- awf[complete.cases(cbind(awf$mean_temp,awf$temp)),] %>%
  select(location, date, station_name, country,
         year, month = month.x, day = day.x,
         water = temp, air = mean_temp, dmean_1, dmean_2, dmean_3,
         dmean_4, dmean_5, flow, dflow_1, rqc, cair)

master.temp <- master.temp[complete.cases(master.temp[,8:18]),]
```

## Get training and testing

### Subsetting seasonal scales

Now we want to subset three master dataframes that contains seasonal-scale data. We categorize the data into four different seasonal categories:

1. spring: 3,4,5
2. summer: 6,7,8
3. fall: 9,10,11
4. winter: 12,1,2

```
master.sum <- master.temp %>%
  filter(month == 6 | month == 7 | month == 8)

master.win <- master.temp %>%
  filter(month == 12 | month == 1 | month == 2)

master.spring <- master.temp %>%
  filter(month == 3 | month == 4 | month == 5)

master.fall <- master.temp %>%
  filter(month == 9 | month == 10 | month == 11)

master.annual <- master.temp %>%
  filter(month != 12 & month != 1 & month != 2)
```

### Identify good year/month data

We want the seasonal data to be greater than 60 days, annual data (winter removed) more than 180 days.

```
## $bigcreek
##  [1] "2000" "2001" "2002" "2003" "2004" "2005" "2006" "2007" "2008" "2009"
## [11] "2012" "2013" "2014"
##
## $bigotter
## [1] "2012" "2013" "2014"
##
## $fox
## [1] "2011" "2012" "2013" "2014"
##
## $genesee
## [1] "2011" "2012" "2013"
##
```

```
## $humber
##  [1] "1998" "1999" "2000" "2001" "2002" "2003" "2005" "2006" "2007" "2008"
## [11] "2009" "2011"
##
## $mississagi
## [1] "2011" "2013"
##
## $nipigon
## [1] "2008" "2009"
##
## $portage
## [1] "2011" "2012"
##
## $saginaw
## [1] "2013" "2014"
##
## $still
## [1] "2002" "2004" "2005" "2008"
##
## $stlouis
## [1] "2011" "2012" "2013" "2014"
##
## $vermilion
## [1] "2012" "2013" "2014"
```

```
## $bigcreek
##  [1] "2001" "2002" "2003" "2004" "2005" "2006" "2007" "2008" "2009" "2012"
## [11] "2013" "2014"
##
## $bigotter
## [1] "2012" "2013" "2014"
##
## $fox
## [1] "2011" "2012" "2013" "2014"
##
## $genesee
## [1] "2011" "2012" "2013"
##
## $humber
##  [1] "1999" "2000" "2001" "2002" "2003" "2005" "2006" "2007" "2008" "2009"
## [11] "2011" "2013"
##
## $mississagi
## [1] "2011" "2013" "2014"
##
## $nipigon
## [1] "2008" "2009"
##
## $portage
## [1] "2011" "2012"
##
## $saginaw
## [1] "2013" "2014"
##
```

```
## $still
## [1] "2005" "2008"
##
## $stlouis
## [1] "2012" "2013"
##
## $vermilion
## [1] "2012" "2013" "2014"

## $bigcreek
##  [1] "2000" "2001" "2002" "2003" "2004" "2005" "2006" "2007" "2008" "2009"
## [11] "2012" "2013" "2014"
##
## $bigotter
## [1] "2012" "2013" "2014"
##
## $fox
## [1] "2011" "2012" "2013" "2014"
##
## $genesee
## [1] "2011" "2012" "2013"
##
## $humber
##  [1] "1998" "1999" "2000" "2001" "2002" "2003" "2005" "2006" "2007" "2008"
## [11] "2009" "2011"
##
## $mississagi
## [1] "2011" "2012" "2013"
##
## $nipigon
## [1] "2008" "2009"
##
## $portage
## [1] "2011" "2012"
##
## $saginaw
## [1] "2014"
##
## $still
## [1] "2002" "2004" "2005" "2008"
##
## $stlouis
## [1] "2011" "2012" "2013" "2014"
##
## $vermilion
## [1] "2013" "2014"

## $bigcreek
##  [1] "2000" "2001" "2002" "2003" "2004" "2005" "2006" "2007" "2008" "2009"
## [11] "2012" "2013"
##
## $bigotter
## [1] "2012" "2013"
##
## $fox
```

```
## [1] "2011" "2012" "2013" "2014"
##
## $genesee
## [1] "2011" "2012" "2013"
##
## $humber
##  [1] "1998" "1999" "2000" "2001" "2002" "2003" "2005" "2006" "2007" "2008"
## [11] "2009" "2011"
##
## $mississagi
## [1] "2010" "2011" "2012" "2013"
##
## $nipigon
## [1] "2008" "2009"
##
## $portage
## [1] "2011" "2012"
##
## $saginaw
## [1] "2012" "2014"
##
## $still
## [1] "2002" "2004"
##
## $stlouis
## [1] "2011" "2012" "2013" "2014"
##
## $vermilion
## [1] "2012" "2013" "2014"
```

```
## $bigcreek
##  [1] "2001" "2002" "2004" "2005" "2006" "2007" "2008" "2009" "2012" "2013"
##
## $bigotter
## [1] "2013"
##
## $fox
## [1] "2012" "2013" "2014"
##
## $genesee
## [1] "2011" "2012" "2013"
##
## $humber
##  [1] "1999" "2000" "2001" "2002" "2003" "2005" "2006" "2007" "2008" "2009"
## [11] "2011"
##
## $mississagi
## [1] "2011" "2012" "2013"
##
## $nipigon
## [1] "2008" "2009"
##
## $portage
## [1] "2011" "2012"
```

```
##
## $saginaw
## [1] "2012" "2014"
##
## $stlouis
## [1] "2012"
##
## $vermilion
## [1] "2012" "2013"
```

### Get training and testing

Similarly, get training and testing for the specific season.

```
fold = 10

## Get training and testing for annual
annual <- get.traintest(master.annual, fulyear, 10)

combined_training_list <- annual[[1]]
combined_testing_list <- annual[[2]]


## Get training and testing for each season
sum <- get.traintest(master.sum, fulsum, 10)
win <- get.traintest(master.win, fulwin, 10)
spr <- get.traintest(master.spring, fulspring, 10)
fall <- get.traintest(master.fall, fulfall, 10)

combined_training_list_sp <- spr[[1]]
combined_testing_list_sp <- spr[[2]]

combined_training_list_su <- sum[[1]]
combined_testing_list_su <- sum[[2]]

combined_training_list_fa <- fall[[1]]
combined_testing_list_fa <- fall[[2]]

combined_training_list_w <- win[[1]]
combined_testing_list_w <- win[[2]]


## Create a list to store all the combined training and testing lists
grand_training <- list(combined_training_list_sp, combined_training_list_su,
                       combined_training_list_fa, combined_training_list_w,
                       combined_training_list)

grand_testing <- list(combined_testing_list_sp, combined_testing_list_su,
                      combined_testing_list_fa, combined_testing_list_w,
                      combined_testing_list)
```

## GLOBAL: futureStream database

We import weekly modelled water temperature data from the 14 tributary locations.

```r
## Import all 14 files
bigcreek <-read.csv("weekly modeled water temperature_new/bigcreek_model.csv") %>%
  mutate(X = "bigcreek")
bigotter <- read.csv("weekly modeled water temperature_new/bigotter_model.csv") %>%
  mutate(X = "bigotter")
fox <- read.csv("weekly modeled water temperature_new/fox_model.csv") %>%
  mutate(X = "fox")
genesee <- read.csv("weekly modeled water temperature_new/genesee_model.csv") %>%
  mutate(X = "genesee")
humber <- read.csv("weekly modeled water temperature_new/humber_model.csv") %>%
  mutate(X = "humber")
longpoint <- read.csv("weekly modeled water temperature_new/lp_model.csv") %>%
  mutate(X = "longpoint")
mississagi <- read.csv("weekly modeled water temperature_new/mississagi_model.csv") %>%
  mutate(X = "mississagi")
nipigon <- read.csv("weekly modeled water temperature_new/nipigon_model.csv") %>%
  mutate(X = "nipigon")
portage <- read.csv("weekly modeled water temperature_new/pb_model.csv") %>%
  mutate(X = "portage")
portdover <- read.csv("weekly modeled water temperature_new/portdover_model.csv") %>%
  mutate(X = "portdover")
saginaw <- read.csv("weekly modeled water temperature_new/saginaw_model.csv") %>%
  mutate(X = "saginaw")
still <- read.csv("weekly modeled water temperature_new/still_model.csv") %>%
  mutate(X = "still")
stlouis <- read.csv("weekly modeled water temperature_new/st_louis_model.csv") %>%
  mutate(X = "stlouis")
vermilion <- read.csv("weekly modeled water temperature_new/vermilion_model.csv") %>%
  mutate(X = "vermilion")

# Combine into one dataframe
futurestream.temp <- rbind(bigcreek,bigotter,fox,genesee,humber,longpoint,
                           mississagi,nipigon,portage,portdover,saginaw,
                           still,stlouis,vermilion) %>%
  rename(location = X, week = weeks, preds.futureS = temperature.avg) %>%
  arrange(location)
```

## REGIONAL: non-linear model with 5 day average temperature

### Fit non-linear model

```r
## starting parameters
coef.spring <- c(alpha=30, gamma=0.05,beta=10)
coef.summer <- c(alpha=25, gamma=0.05,beta=10)
coef.fall <- c(alpha=30, gamma=0.05,beta=10)
coef.annual <- c(alpha=20, gamma=0.05,beta=9)

coef.list <- list(coef.spring, coef.summer, coef.fall, NA, coef.annual)


spring.r <- vector("list", fold)
summer.r <- vector("list", fold)
fall.r <- vector("list", fold)
```

```r
winter.r <- vector("list", fold)
annual.r <- vector("list", fold)

grand.nonlinear <- list(spring.r, summer.r, fall.r, winter.r, annual.r)


## Iteration starts here
for (season in c(1,2,3,5)) {

  ## Get current season and its correspnding training/testing
  train <- grand_training[[season]]
  test <- grand_testing[[season]]

  ## 10 fold iteration starts here
  for (i in 1:fold){

    compare <- NA
    # select current dataset, and all unique location levels
    current.training <- train[[i]]
    current.testing <- test[[i]]
    compare <- current.testing

    # model training and predicting
    model <- nlme(water ~ alpha / (1 + exp(gamma * (beta - cair))),
                  fixed = list(alpha~1,gamma~1,beta~1),
                  random = gamma ~ 1|location,
                  start = coef.list[[season]],
                  data = current.training, na.action = na.omit,
                  control = list(msMaxIter = 200))

    compare$preds.ar <- as.vector(predict(
      model, newdata = current.testing, re.form = ~1|location))

    grand.nonlinear[[season]][[i]] <- compare
  }
}
```

**Convert into weekly output**

```r
# new dataframe to store all weekly results
grand.week = grand.nonlinear

for (season in c(1,2,3,5)) {

  for(i in 1:fold) {
    ## Convert lag5 model prediction to weekly timescale
    df.days <- grand.nonlinear[[season]][[i]]
    df.days$week <- week(df.days$date)
    df.days$week[df.days$week == 53] <- 52 #need to convert week 53 to 52

    df.week <- merge(
      aggregate(preds.ar~week+location,FUN=mean,data=df.days,na.action=na.omit),
      aggregate(water~week+location,FUN=mean,data=df.days,na.action=na.omit),
```

```
      all=TRUE) %>%
      na.omit() %>%
      arrange(location, week)

    comp.week <- merge(df.week, futurestream.temp, by=c("location","week")) %>%
      arrange(location, week)

    # store it back into the grand.nonlinear list
    grand.week[[season]][[i]] <- comp.week
  }
}
```

## Compare between GLOBAL and REGIONAL

### Calculate RMSE for each seasonal scale

```
# new function - different from other files...
cal.rmse <- function(out.list, fold, switch) {

  # data frame to store the results
  r <- matrix(NA, nrow=fold, ncol=length(unique(out.list[[1]]$location)))
  colnames(r) <- unique(out.list[[1]]$location)

  # 10 iterations
  for (i in 1:fold){
   compare <- out.list[[i]]

   # calculate rmse
   for (loc in unique(compare$location)) {
     compare.now <- subset(compare, location == loc) %>% na.omit()

     if (switch=="r"){
       r[i,loc] <-
         round(sqrt(mean((compare.now$water-compare.now$preds.ar)^2)),2)
     } else if (switch=="g"){
       r[i,loc] <-
         round(sqrt(mean((compare.now$water-compare.now$preds.futureS)^2)),2)
     }
    }
  }
  return(r)
}


## Annual
annual.compare <- data.frame(
  regional = colMeans(cal.rmse(grand.week[[5]], fold, "r"), na.rm=T),
  global = colMeans(cal.rmse(grand.week[[5]], fold, "g"), na.rm=T))

colMeans(annual.compare)
```

```
## regional   global
## 2.227583 3.320750
```

```r
knitr::kable(annual.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 2.006 | 2.783 |
| bigotter | 1.751 | 3.592 |
| fox | 2.815 | 3.872 |
| genesee | 1.674 | 2.204 |
| humber | 2.024 | 3.602 |
| mississagi | 2.331 | 2.148 |
| nipigon | 3.395 | 4.630 |
| portage | 2.871 | 4.188 |
| saginaw | 1.681 | 3.795 |
| still | 2.544 | 2.588 |
| stlouis | 1.896 | 3.900 |
| vermilion | 1.743 | 2.547 |

```r
## Spring
spring.compare <- data.frame(
  regional = colMeans(cal.rmse(grand.week[[1]], fold, "r"), na.rm=T),
  global = colMeans(cal.rmse(grand.week[[1]], fold, "g"), na.rm=T))

colMeans(spring.compare)
```

```
## regional   global
## 2.105917 3.164000
```

```r
knitr::kable(spring.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 1.564 | 1.991 |
| bigotter | 1.893 | 2.162 |
| fox | 2.357 | 2.527 |
| genesee | 2.090 | 2.435 |
| humber | 2.178 | 3.306 |
| mississagi | 1.342 | 1.290 |
| nipigon | 0.539 | 4.046 |
| portage | 2.637 | 5.042 |
| saginaw | 1.911 | 3.248 |
| still | 3.306 | 4.254 |
| stlouis | 2.898 | 3.145 |
| vermilion | 2.556 | 4.522 |

```r
## Summer
summer.compare <- data.frame(
  regional = colMeans(cal.rmse(grand.week[[2]], fold, "r"), na.rm=T),
  global = colMeans(cal.rmse(grand.week[[2]], fold, "g"), na.rm=T))

colMeans(summer.compare)
```

```
## regional   global
## 1.832111 3.593417
```

```r
knitr::kable(summer.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 1.099 | 3.507 |
| bigotter | 1.303 | 5.141 |
| fox | 1.664 | 4.132 |
| genesee | 1.878 | 1.200 |
| humber | 1.693 | 4.822 |
| mississagi | 1.416 | 1.812 |
| nipigon | 3.587 | 5.618 |
| portage | 2.344 | 2.417 |
| saginaw | 3.003 | 5.990 |
| still | 1.469 | 1.820 |
| stlouis | 1.279 | 4.612 |
| vermilion | 1.250 | 2.050 |

```r
## Fall
fall.compare <- data.frame(
  regional = colMeans(cal.rmse(grand.week[[3]], fold, "r"), na.rm=T),
  global = colMeans(cal.rmse(grand.week[[3]], fold, "g"), na.rm=T))

colMeans(fall.compare)
```

```
## regional   global
## 2.124583 2.789833
```

```r
knitr::kable(fall.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 2.527 | 2.544 |
| bigotter | 1.659 | 2.466 |
| fox | 3.236 | 3.511 |
| genesee | 1.986 | 1.610 |
| humber | 1.961 | 2.670 |
| mississagi | 1.846 | 1.863 |
| nipigon | 1.308 | 3.510 |
| portage | 3.214 | 4.444 |
| saginaw | 1.504 | 3.540 |
| still | 2.243 | 1.490 |
| stlouis | 1.798 | 3.343 |
| vermilion | 2.213 | 2.487 |

### Calculate NSC for each seasonal scale

```r
# new function - different from other files...
cal.nsc <- function(out.list, fold, switch) {

  # data frame to store the results
  r <- matrix(NA, nrow=fold, ncol=length(unique(out.list[[1]]$location)))
  colnames(r) <- unique(out.list[[1]]$location)
```

```r
  # 10 iterations
  for (i in 1:fold){
   compare <- out.list[[i]]

   # calculate rmse
   for (loc in unique(compare$location)) {
     compare.now <- subset(compare, location == loc) %>% na.omit()

     if (switch=="r"){
       r[i,loc] <-
         1 - sum((compare.now$water - compare.now$preds.ar)^2) / sum((compare.now$water - mean(compare.r
     } else if (switch=="g"){
       r[i,loc] <-
         1 - sum((compare.now$water - compare.now$preds.futureS)^2) / sum((compare.now$water - mean(comp
     }
    }
  }
  return(r)
}


## Annual
annual.compare <- data.frame(
  regional = colMeans(cal.nsc(grand.week[[5]], fold, "r"), na.rm=T),
  global = colMeans(cal.nsc(grand.week[[5]], fold, "g"), na.rm=T))

colMeans(annual.compare)
```

```
##  regional    global
## 0.8735882 0.7173597
```

```r
knitr::kable(annual.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 0.832 | 0.696 |
| bigotter | 0.857 | 0.422 |
| fox | 0.887 | 0.766 |
| genesee | 0.954 | 0.914 |
| humber | 0.881 | 0.646 |
| mississagi | 0.913 | 0.926 |
| nipigon | 0.671 | 0.388 |
| portage | 0.765 | 0.512 |
| saginaw | 0.962 | 0.811 |
| still | 0.871 | 0.868 |
| stlouis | 0.947 | 0.787 |
| vermilion | 0.942 | 0.871 |

```r
## Spring
spring.compare <- data.frame(
  regional = colMeans(cal.nsc(grand.week[[1]], fold, "r"), na.rm=T),
  global = colMeans(cal.nsc(grand.week[[1]], fold, "g"), na.rm=T))

colMeans(spring.compare)
```

```
##  regional    global
## 0.8002597 0.1538867
```

```r
knitr::kable(spring.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 0.873 | 0.815 |
| bigotter | 0.755 | 0.705 |
| fox | 0.832 | 0.736 |
| genesee | 0.882 | 0.804 |
| humber | 0.845 | 0.646 |
| mississagi | 0.892 | 0.903 |
| nipigon | 0.901 | -4.125 |
| portage | 0.581 | -0.581 |
| saginaw | 0.924 | 0.782 |
| still | 0.606 | 0.359 |
| stlouis | 0.727 | 0.589 |
| vermilion | 0.784 | 0.214 |

```r
## Summer
summer.compare <- data.frame(
  regional = colMeans(cal.nsc(grand.week[[2]], fold, "r"), na.rm=T),
  global = colMeans(cal.nsc(grand.week[[2]], fold, "g"), na.rm=T))

colMeans(summer.compare)
```

```
##   regional     global
##  0.2243522 -3.4693223
```

```r
knitr::kable(summer.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 0.470 | -4.895 |
| bigotter | -0.366 | -19.690 |
| fox | 0.330 | -3.222 |
| genesee | 0.387 | 0.717 |
| humber | -0.061 | -8.953 |
| mississagi | 0.674 | 0.459 |
| nipigon | 0.097 | -1.300 |
| portage | -0.748 | -0.887 |
| saginaw | 0.459 | -1.151 |
| still | 0.237 | -0.397 |
| stlouis | 0.760 | -1.875 |
| vermilion | 0.451 | -0.437 |

```r
## Fall
fall.compare <- data.frame(
  regional = colMeans(cal.nsc(grand.week[[3]], fold, "r"), na.rm=T),
  global = colMeans(cal.nsc(grand.week[[3]], fold, "g"), na.rm=T))

colMeans(fall.compare)
```

```
##   regional    global
## 0.8137708 0.6714939
```

```r
knitr::kable(fall.compare, digits = 3)
```

| | regional | global |
|---|---|---|
| bigcreek | 0.575 | 0.523 |
| bigotter | 0.750 | 0.526 |
| fox | 0.747 | 0.708 |
| genesee | 0.902 | 0.937 |
| humber | 0.751 | 0.512 |
| mississagi | 0.892 | 0.880 |
| nipigon | 0.899 | 0.296 |
| portage | 0.638 | 0.313 |
| saginaw | 0.955 | 0.760 |
| still | 0.828 | 0.928 |
| stlouis | 0.937 | 0.801 |
| vermilion | 0.892 | 0.873 |