Guide 2: Cleaning and Pre-Processing Weather Station Data

Marc Los Huertos

February 3, 2024 (ver. 0.4)

1 Introduction

1.1 Goals

The goal of this guide is to provide a step-by-step process for cleaning and pre-processing weather station data. The data is from the Global Historical Climatology Network (GHCN) Daily dataset. The data is available from the National Oceanic and Atmospheric Administration (NOAA) and is available from the National Centers for Environmental Information (NCEI) at

1.2 Background

What is GHCN-Daily?Links to an external site.

1.3 Approach

We need to address several things that might get in the way of our analysis:

1. read station data into R
2. date format (convert to POSIXct, adding month and year as varibles)
3. evaluate missing data
4. units
5. outliers
6. Other stuff??

2 Cleaning and Pre-Processing Functions

2.1 Starting the Process

Before you begin, make sure you have the stations read into R. You can do this by running the following code:

```
ls()

## [1] "CleanUp.fun"          "coverage.fun"          "fixdates.fun"
## [4] "MonthlyAnomalies.fun" "MonthlyNormals.fun"    "MonthlyValues.fun"
## [7] "ReadStations.fun"
```

You should see station1...stationN, where N is the number of stations you have read in.

If the data are not in the R environment, you can read them in using the following but you have the uncompressed files, you can read them in using the following function:

```
## function (datafolder)
## {
##     StationList <- list.files(datafolder, full.names = TRUE,
##         pattern = "\\.csv$")
##     colnames <- c("ID", "DATE", "ELEMENT", "VALUE", "M-FLAG",
##         "Q-FLAG", "S-FLAG", "OBS-TIME")
##     for (i in 1:length(StationList)) {
##         assign(paste0("station", i), read.csv(StationList[i],
##             header = TRUE, col.names = colnames), envir = parent.frame())
##     }
## }
```

Example of how to use the function

```
datafolder = "/home/mwl04747/RTricks/04_Regional_Climate_Trends/Data/SP24"
ReadStations.fun(datafolder)
```

Using str(), make sure the data sets look right!

```
str(station2)
```

2.2 Clean Data

Function to Fix Dates
```
## function (station)
## {
##     station$Ymd = as.Date(as.character(station$DATE), format = "%Y%m%d")
##     station$MONTH = as.numeric(format(station$Ymd, "%m"))
##     station$YEAR = as.numeric(format(station$Ymd, "%Y"))
##     return(station)
## }
```

Example of how to use the function

```
station1a <- fixdates.fun(station1)
```

Evaluation Data Coverage We need to know how much data we have for each station. This is important for the next steps in the process.

```
## function (station, element = "TMAX")
## {
##     Dates.all = data.frame(Ymd = seq.Date(from = min(station$Ymd),
##         to = max(station$Ymd), by = "day"))
##     station.full = merge(Dates.all, station, all = TRUE)
##     station.coverage = sum(!is.na(station.full$VALUE[station.full$ELEMENT ==
##         element]))/length(station.full$VALUE[station.full$ELEMENT ==
##         element]) * 100
##     return(round(station.coverage, 2))
## }
```

Example of how to use the function

```
coverage.fun(station1a)
```

Function to Convert Units

Function to Create Monthly Values For TMAX and TMIN, we want monthly means, for rainfall, we'll want monthly totals.[1]

Here's the function:

```
## function (x = station1)
## {
##     x.TMAX.monthly = aggregate(VALUE ~ MONTH + YEAR, data = subset(x,
##         ELEMENT == "TMAX"), mean)
##     names(x.TMAX.monthly) <- c("MONTH", "YEAR", "TMAX")
##     x.TMIN.monthly = aggregate(VALUE ~ MONTH + YEAR, data = subset(x,
##         ELEMENT == "TMIN"), mean)
##     names(x.TMIN.monthly) <- c("MONTH", "YEAR", "TMIN")
```

[1]Brody/Evyln: We need code to exclue months with missing data, these might not be represenative of the month if missing, especically for PRCP!

```
##     x.PRCP.monthly = aggregate(VALUE ~ MONTH + YEAR, data = subset(x,
##         ELEMENT == "PRCP"), sum)
##     names(x.PRCP.monthly) <- c("MONTH", "YEAR", "PRCP")
##     return(list(x.TMAX.monthly, x.TMIN.monthly, x.PRCP.monthly))
## }
```

Example of how to use the function:

```
station1.monthly <- MonthlyValues.fun(station1a)
```

Function to Create Anomalies Here's the function:

```
## Error in print(Anomalies.fun):  object 'Anomalies.fun' not
found
```

Example of how to use the function:

```
station1.monthly.Anomalies <- Anomalies.fun(station1a)

## Error in Anomalies.fun(station1a):  could not find function
"Anomalies.fun"
```

First, I tested each line on station1. I will then create a function to clean the data and apply it to each station.

```
# station1£VALUE = station1£VALUE/10  # Correct Values Units
# Fix Date format
station1$Ymd = as.Date(as.character(station1$DATE), format = "%Y%m%d")
str(station1)

station1$MONTH = as.numeric(format(station1$Ymd, "%m"))
station1$YEAR = as.numeric(format(station1$Ymd, "%Y"))
station1.monthly = aggregate(VALUE ~ MONTH + YEAR,
                   data = subset(station1, ELEMENT == "TMAX"), mean)

# create baseline (normals) dataset
station1.normals = subset(station1,
                           Ymd >= "1961-01-01" & Ymd <= "1990-12-31")
station1.normals.monthly = aggregate(VALUE ~ MONTH,
                   data = subset(station1.normals, ELEMENT == "TMAX"), mean)
names(station1.normals.monthly) <- c("MONTH", "NORMALS")

station1.anomaly = merge(station1.monthly,
                         station1.normals.monthly, by = "MONTH")
```

```
station1.anomaly$ANOMALY =
  station1.anomaly$VALUE - station1.anomaly$NORMALS
```

3 Code for QA/QC and Preparing for our Analyses

Function is probably sensitive to missing values, need to work on that!

```
x=station1
cleandataframe.fun <- function(x){
  #x£VALUE = x£VALUE/10
  x$Ymd = as.Date(as.character(x$DATE), format = "%Y%m%d")
  x$MONTH = as.numeric(format(x$Ymd, "%m"))
  x$YEAR = as.numeric(format(x$Ymd, "%Y"))

  x.TMAX.monthly = aggregate(VALUE ~ MONTH + YEAR,
            data = subset(x, ELEMENT == "TMAX"), mean)
  names(x.TMAX.monthly) <- c("MONTH", "YEAR", "TMAX")
  x.TMIN.monthly = aggregate(VALUE ~ MONTH + YEAR,
            data = subset(x, ELEMENT == "TMIN"), mean)
  names(x.TMIN.monthly) <- c("MONTH", "YEAR", "TMIN")
  x.PRCP.monthly = aggregate(VALUE ~ MONTH + YEAR,
            data = subset(x, ELEMENT == "PRCP"), sum)
  names(x.PRCP.monthly) <- c("MONTH", "YEAR", "PRCP")

  x.normals = subset(x, Ymd >= "1961-01-01" & Ymd <= "1990-12-31")
  x.TMAX.normals.monthly = aggregate(VALUE ~ MONTH,
            data = subset(x.normals, ELEMENT == "TMAX"), mean)
  names(x.TMAX.normals.monthly) <- c("MONTH", "NORMALS")
  x.TMIN.normals.monthly = aggregate(VALUE ~ MONTH,
            data = subset(x.normals, ELEMENT == "TMIN"), mean)
  names(x.TMIN.normals.monthly) <- c("MONTH", "NORMALS")
  x.PRCP.normals.monthly = aggregate(VALUE ~ MONTH,
            data = subset(x.normals, ELEMENT == "PRCP"), sum)
  names(x.PRCP.normals.monthly) <- c("MONTH", "NORMALS")


  x.TMAX.anomaly = merge(x.TMAX.monthly, x.TMAX.normals.monthly, by = "MONTH")
  x.TMAX.anomaly$TMAX.anomaly = x.TMAX.anomaly$TMAX - x.TMAX.anomaly$NORMALS

  x.TMIN.anomaly = merge(x.TMIN.monthly, x.TMIN.normals.monthly, by = "MONTH")
  x.TMIN.anomaly$TMIN.anomaly = x.TMIN.anomaly$TMIN - x.TMIN.anomaly$NORMALS
  x.PRCP.anomaly = merge(x.PRCP.monthly, x.PRCP.normals.monthly, by = "MONTH")
  x.PRCP.anomaly$PRCP.anomaly = x.PRCP.anomaly$PRCP - x.PRCP.anomaly$NORMALS

  TEMP <- merge(x.TMAX.anomaly, x.TMIN.anomaly, by = c("MONTH", "YEAR") )
  x.anomaly <- merge(TEMP, x.PRCP.anomaly, by = c("MONTH", "YEAR"))[,c(1:3, 5:6, 8:9, 11)]
  library(lubridate)
  #x.anomaly£Ym1 = as.Date(paste(x.anomaly£YEAR, x.anomaly£MONTH), format="%Y %m")
  x.anomaly$Ym1 =  lubridate::myd(paste(x.anomaly$MONTH, x.anomaly$YEAR, "1"))
  str(x.anomaly)
  return(x.anomaly)
}
```

3.1 Apply Function to All Stations

So far, I have only run function for 1 station, but I suspect you can figure out how to run it for each one!

```
station1.clean= cleandataframe.fun(station1)
```

3.2 Plot Anomaly

Graphic has lots of issues. more next time! But here's a start.

```
options(scipen=5)
par(mar=c(4,6,2,5))

plot(ANOMALY ~ YEAR, data = subset(station1.TMAX, MONTH == 1),
     las=1, pch=19, col = "blue", cex=.5, #xlab = "Year",
     ylab = "Maximum Temp Anomaly (C)",
     main="January Maximum Temp Anomaly")
mtext("Maximum Temp Anomaly (C)", side = 2, line = 3)
temp.lm = lm(ANOMALY ~ YEAR, data = subset(station1.TMAX, MONTH == 1))
abline(coef(temp.lm), col = "red")
```

4 QA/QC

4.1 Missing Data

```
# determine percent missing in station1
station1.TMAX.coverage = sum(!is.na(station1$VALUE[station1$ELEMENT=="TMAX"]))/length(statio
# function to determine percent missing
coverage.fun <- function(station, element){
  Dates.all = data.frame(Ymd=seq.Date(from=min(station$Ymd), to=max(station$Ymd), by="day"))
  station.full = merge(Dates.all, station, all = TRUE)
  station.coverage = sum(!is.na(station.full$VALUE[station.full$ELEMENT==element]))/
    length(station.full$VALUE[station.full$ELEMENT==element])*100
  return(round(station.coverage,2))
}

coverage.data(station1, "TMAX")
coverage.data(station2, "TMAX")

Date.full = data.frame(Ymd=seq.Date(from=min(station1$Ymd), to=max(station1$Ymd), by="day"))
str(Date.full)

station1.full = merge(Date.full, station1, all = TRUE)
coverage.fun(station1, "TMAX")
coverage.fun(station2, "TMAX")
```

5 Next Steps

This is all we need to do so far. Next week, we'll look at different way to visualize the data!

I'll save all the station data into csv files, then use them in the next guide to clean, process, and visualize data.

```
write.csv(station1, file = paste0(here::here("04_Regional_Climate_Trends", "Data", "SP24", '
write.csv(station2, file = paste0(here::here("04_Regional_Climate_Trends", "Data", "SP24", '
write.csv(station3, file = paste0(here::here("04_Regional_Climate_Trends", "Data", "SP24", '
write.csv(station4, file = paste0(here::here("04_Regional_Climate_Trends", "Data", "SP24", '
write.csv(station5, file = paste0(here::here("04_Regional_Climate_Trends", "Data", "SP24", '
```