Plot and Subset Precipitation Data in R - 2013 Colorado Floods

In this lesson, we will learn how to import a larger dataset, and test our skills cleaning and plotting the data.

Learning Objectives

After completing this tutorial, you will be able to:

- Import a text file into R.
- Plot quantitative time series data using ggplot
- Ensure that NoData values do not interfere with quantitative analysis by setting them to NA in `R`.
- use the na.rm function argument when performing math with large datasets.
- subset data using the dplyr filter() function
- use dplyr pipes to filter data in R.

What you need

You need `R` and `RStudio` to complete this tutorial. Also you should have an `earth-analytics` directory setup on your computer with a `/data` directory with it.

- How to Setup R / RStudio
- Setup your working directory
- Intro to the R & RStudio Interface

R Libraries to Install:

- **ggplot2:** `install.packages("ggplot2")`
- **dplyr:** `install.packages("dplyr")`

Data Download

Download Precipitation Data

Work with Precipitation Data

R Libraries

Let's get started by loading the ggplot2 library and making sure our working directory is set. Be sure to also set `stringsAsFactors` to `FALSE` as shown below.

```
# set your working directory to the earth-analytics directory
# setwd("working-dir-path-here")

# load packages
library(ggplot2) # efficient, professional plots
library(dplyr) # efficient data manipulation
```

```r
# set strings as factors to false for everything
options(stringsAsFactors = FALSE)
```

Import Precipitation Data

We will use a precipitation dataset derived from data accessed through the National Centers for Environmental Information (formerly National Climate Data Center) Cooperative Observer Network (COOP) station 050843 in Boulder, CO. The data time span is: 1 January 2003 through 31 December 2013.

We can use `read.csv()` to import the `.csv` file.

```r
# download the data
# download.file(url = "https://ndownloader.figshare.com/files/7283285",
#               destfile = "data/week2/805325-precip-dailysum_2003-2013.csv")

# import the data
boulder_daily_precip <- read.csv("data/week2/805325precip_dailysum_20032013.csv",
         header = TRUE)


# view first 6 lines of the data
head(boulder_daily_precip)
##   X       DATE DAILY_PRECIP     STATION    STATION_NAME ELEVATION LATITUDE
## 1 1 2003-01-01          0.0 COOP:050843 BOULDER 2 CO US    1650.5 40.03389
## 2 2 2003-02-01          0.0 COOP:050843 BOULDER 2 CO US    1650.5 40.03389
## 3 3 2003-02-03          0.4 COOP:050843 BOULDER 2 CO US    1650.5 40.03389
## 4 4 2003-02-05          0.2 COOP:050843 BOULDER 2 CO US    1650.5 40.03389
## 5 5 2003-02-06          0.1 COOP:050843 BOULDER 2 CO US    1650.5 40.03389
## 6 6 2003-02-07          0.1 COOP:050843 BOULDER 2 CO US    1650.5 40.03389
##   LONGITUDE YEAR JULIAN
## 1 -105.2811 2003      1
## 2 -105.2811 2003     32
## 3 -105.2811 2003     34
## 4 -105.2811 2003     36
## 5 -105.2811 2003     37
## 6 -105.2811 2003     38

# view structure of data
str(boulder_daily_precip)
## 'data.frame':    788 obs. of  10 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ DATE        : chr  "2003-01-01" "2003-02-01" "2003-02-03" "2003-02-05" ...
##  $ DAILY_PRECIP: num  0 0 0.4 0.2 0.1 0.1 0 0 0.3 0.1 ...
##  $ STATION     : chr  "COOP:050843" "COOP:050843" "COOP:050843" "COOP:050843" ...
##  $ STATION_NAME: chr  "BOULDER 2 CO US" "BOULDER 2 CO US" "BOULDER 2 CO US" "BOULDER 2 CO US" ...
##  $ ELEVATION   : num  1650 1650 1650 1650 1650 ...
##  $ LATITUDE    : num  40 40 40 40 40 ...
##  $ LONGITUDE   : num  -105 -105 -105 -105 -105 ...
##  $ YEAR        : int  2003 2003 2003 2003 2003 2003 2003 2003 2003 2003 ...
##  $ JULIAN      : int  1 32 34 36 37 38 41 49 55 58 ...

# are there any unusual / No data values?
summary(boulder_daily_precip$DAILY_PRECIP)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.1000  0.1000  0.2478  0.3000  9.8000
max(boulder_daily_precip$DAILY_PRECIP)
## [1] 9.8
```

About the Data

Viewing the structure of these data, we can see that different types of data are included in this file.

- **STATION** and **STATION_NAME**: Identification of the COOP station.
- **ELEVATION, LATITUDE** and **LONGITUDE**: The spatial location of the station.
- **DATE**: The date when the data were collected in the format: YYYYMMDD. Notice that DATE is currently class `chr`, meaning the data is interpreted as a character class and not as a date.
- **DAILY_PRECIP**: The total precipitation in inches. Important: the metadata notes that the value 999.99 indicates missing data. Also important, hours with no precipitation are not recorded.
- **YEAR**: the year the data were collected
- **JULIAN**: the JULIAN DAY the data were collected.

Additional information about the data, known as metadata, is available in the PRECIP_HLY_documentation.pdf. The metadata tell us that the noData value for these data is 999.99. IMPORTANT: we have modified these data a bit for ease of teaching and learning. Specifically, we've aggregated the data to represent daily sum values and added some noData values to ensure you learn how to clean them!

You can download the original complete data subset with additional documentation here.

Challenge

Using everything you've learned in the previous lessons, import the data, clean it up by assigning noData values to NA and making sure the dates are stored in the correct class. When you are done, plot it using `ggplot()`. Be sure to include a TITLE, and label the X and Y axes.

Some notes to help you along:

- Date: be sure to take of of the date format when you import the data.
- NoData Values: We know that the no data value = 999.99. We can account for this when we read in the data. Remember how?

Your final plot should look something like the plot below.

Data Tip: For a more thorough review of date/time classes, see the NEON tutorial *Dealing With Dates & Times in R - as.Date, POSIXct, POSIXlt.* {: .notice }

Challenge

Take a close look at the plot.

- What does each point represent?
- What are the minimum and maximum precipitation values for the 10 year span?

Subset the Data

If we wanted to zoom in and look at some data over a smaller time period, we can subset it. Let create a subset of data for the time period around the flood between 15 August to 15 October 2013. We will use the `filter()` function in the `dplyr` package to do this.

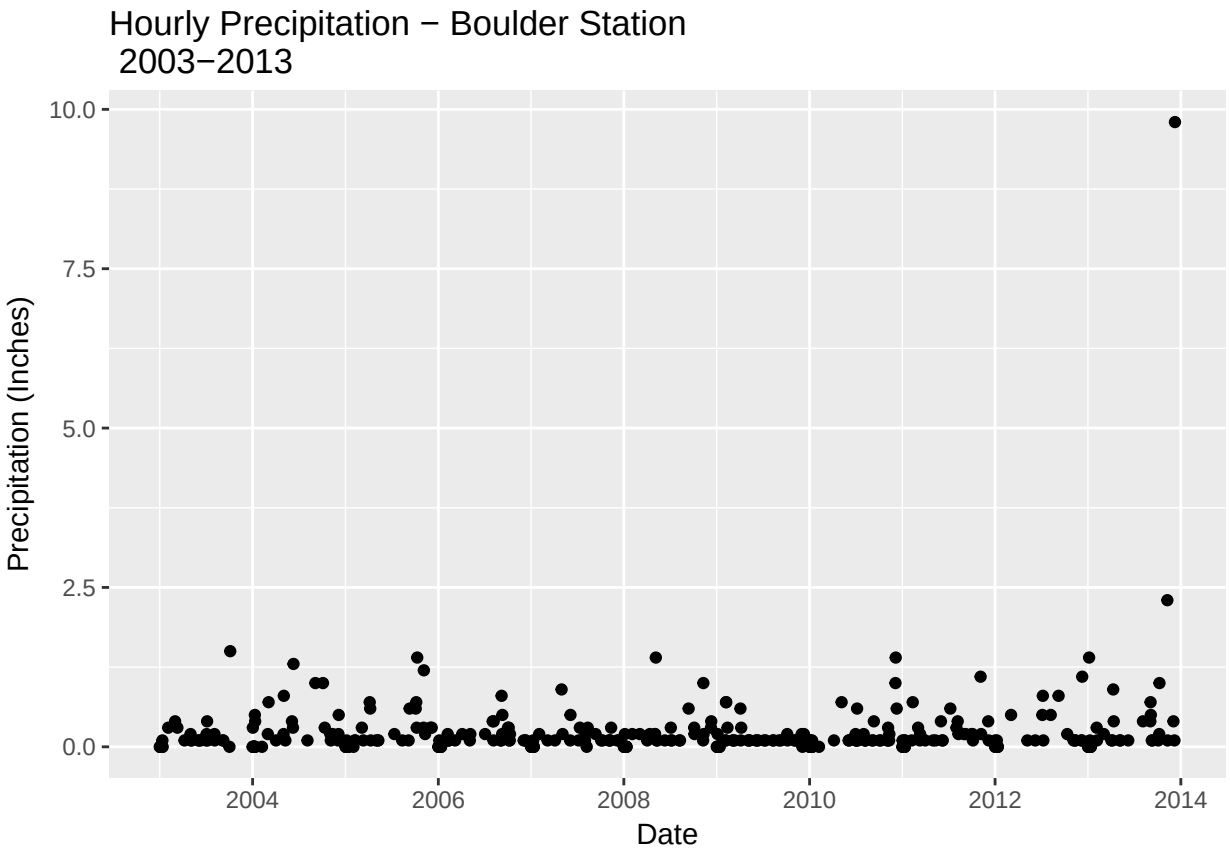

Figure 1: precip plot w fixed dates

Introduction to the Pipe %<%

Pipes let you take the output of one function and send it directly to the next, which is useful when you need to do many things to the same data set. Pipes in R look like %>% and are made available via the magrittr package, installed automatically with dplyr.

If we want to subset our data, we could use the following syntax

```
new_data <- filter(boulder_daily_precip, (DATE >= as.Date('2013-08-15') & DATE <= as.Date('2013-10-15')
```

When we code like this we have to create intermediate data frame outputs. however, pipes allow us to link multiple steps in one line.

```
# subset 2 months around flood
precip_boulder_AugOct <- boulder_daily_precip %>%
                         filter(DATE >= as.Date('2013-08-15') & DATE <= as.Date('2013-10-15'))
```

In the code above, we use the pipe to send the boulder_daily_precip data through a filter setp. In that filter step, we filter our only the rows withing the date range that we specified. Since %>% takes the object on its left and passes it as the first argument to the function on its right, we don't need to explicitly include it as an argument to the filter() function.

```
# check the first & last dates
min(precip_boulder_AugOct$DATE)
## [1] "2013-09-03"
max(precip_boulder_AugOct$DATE)
## [1] "2013-10-09"

# create new plot
precPlot_flood2 <- ggplot(data=precip_boulder_AugOct, aes(DATE,DAILY_PRECIP)) +
  geom_bar(stat="identity") +
  xlab("Date") + ylab("Precipitation (inches)") +
  ggtitle("Daily Total Precipitation Aug - Oct 2013 for Boulder Creek")

precPlot_flood2
```

Challenge

Create a subset from the same dates in 2012 to compare to the 2013 plot. Use the ylim() argument to ensure the y axis range is the SAME as the previous plot - from 0 to 10".

How different was the rainfall in 2012?

HINT: type ?lims in the console to see how the xlim and ylim arguments work.

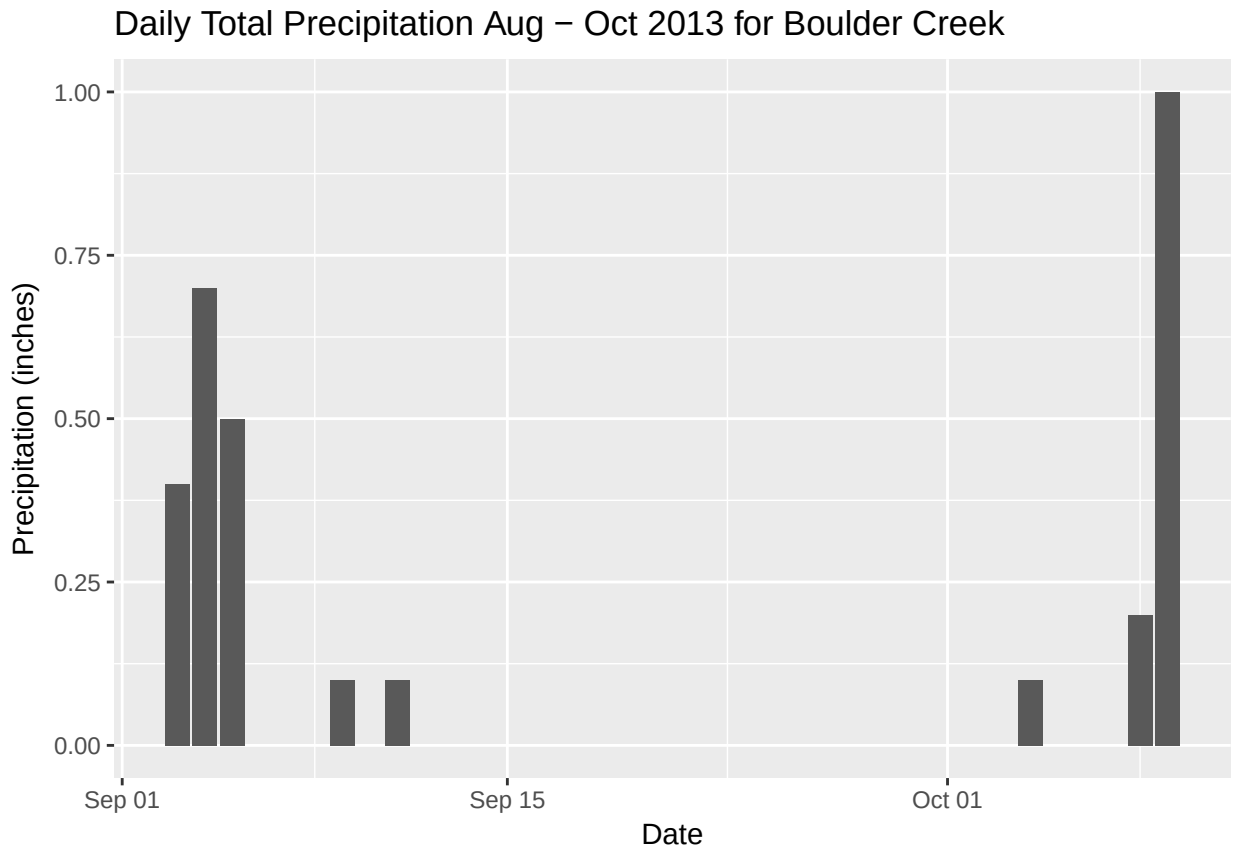

Figure 2: precip plot subset

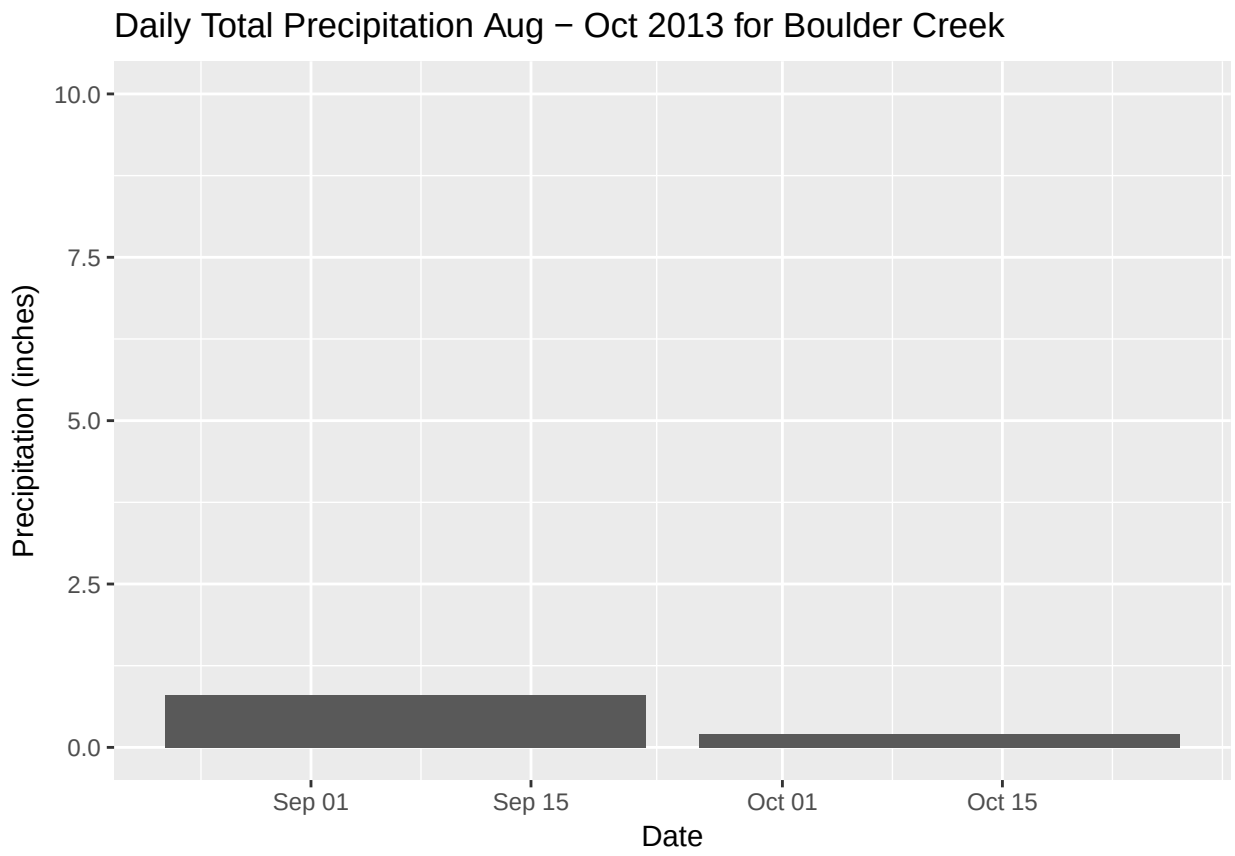

Figure 3: precip plot subset 2