Plot with GGPLOT

{% include toc title="In This Lesson" icon="file-text" %}

In this tutorial, we will explore more advanced plotting techniques using `ggplot2`.

Learning Objectives

At the end of this activity, you will be able to:

- Use the `ggplot()` plot function to create custom plots.
- Add labels to x and y axes and a title to your ggplot plot.
- Customize the colors and look of a ggplot plot.

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

In our week 1 homework, we used the quick plot function of ggplot2 to plot our data. In this tutorial, we'll explore ggplot - which offers many more advanced plotting features.

Let's explore the code below to create a quick plot.

```
# load the ggplot2 library for plotting
library(ggplot2)

# download data from figshare
# note that we already downloaded the data to our laptops previously
# but in case you don't have it - re-download it by uncommenting the code below.
#download.file(url = "https://ndownloader.figshare.com/files/7010681",
#              destfile = "data/boulder-precip.csv")

# import data
boulder_precip <- read.csv(file="data/boulder-precip.csv")

# view first few rows of the data
head(boulder_precip)
##     X       DATE PRECIP
## 1 756 2013-08-21    0.1
## 2 757 2013-08-26    0.1
## 3 758 2013-08-27    0.1
## 4 759 2013-09-01    0.0
## 5 760 2013-09-09    0.1
## 6 761 2013-09-10    1.0

# when we download the data we create a dataframe
# view each column of the data frame using its name (or header)
boulder_precip$DATE
```

```
##  [1] "2013-08-21" "2013-08-26" "2013-08-27" "2013-09-01" "2013-09-09"
##  [6] "2013-09-10" "2013-09-11" "2013-09-12" "2013-09-13" "2013-09-15"
## [11] "2013-09-16" "2013-09-22" "2013-09-23" "2013-09-27" "2013-09-28"
## [16] "2013-10-01" "2013-10-04" "2013-10-11"

# view the precip column
boulder_precip$PRECIP
##  [1] 0.1 0.1 0.1 0.0 0.1 1.0 2.3 9.8 1.9 1.4 0.4 0.1 0.3 0.3 0.1 0.0 0.9
## [18] 0.1

# q plot stands for quick plot. Let's use it to plot our data
qplot(x=boulder_precip$DATE,
      y=boulder_precip$PRECIP)
```

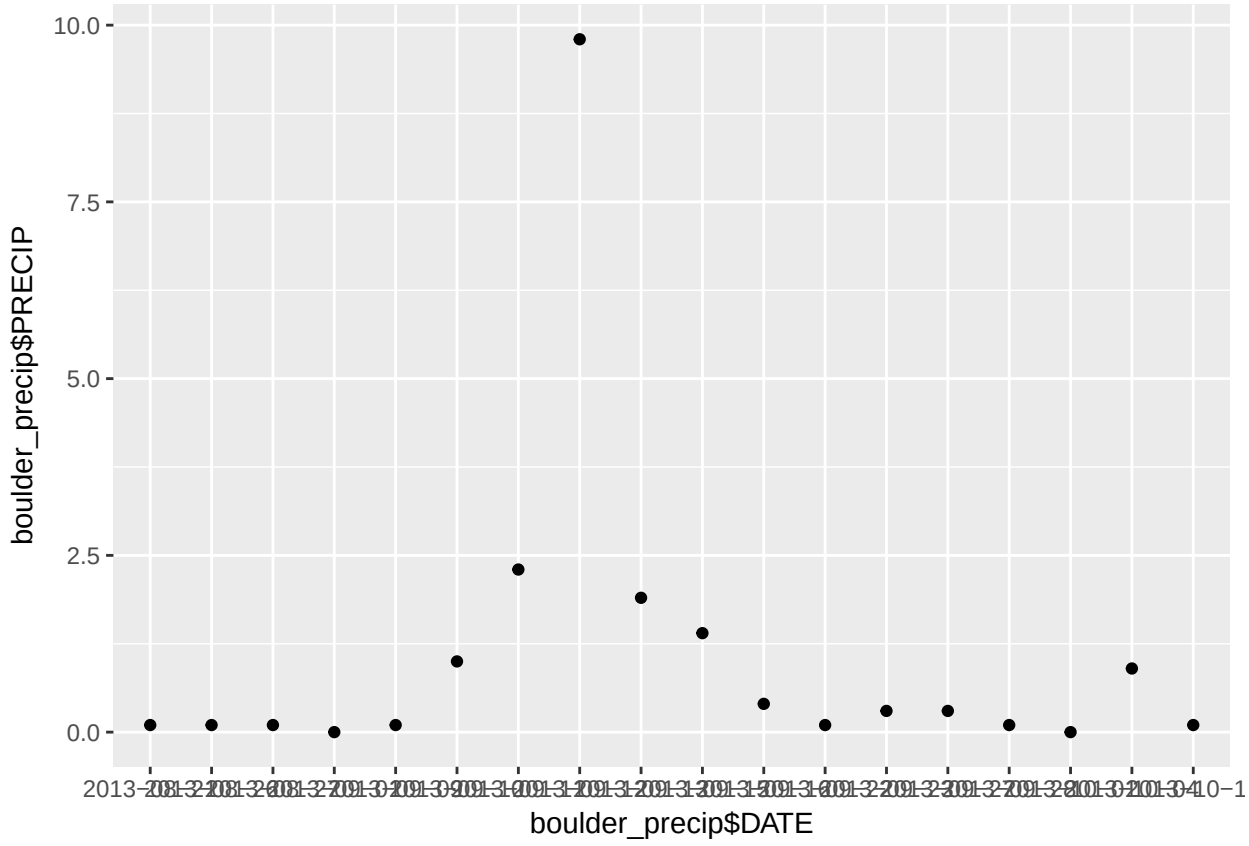

Figure 1: quick plot of precip data

Plotting with ggplot2

`ggplot2` is a plotting package that makes it simple to create complex plots from data in a dataframe. It uses default settings, which help to create publication quality plots with a minimal amount of settings and tweaking.

ggplot graphics are built step by step by adding new elements.

To build a ggplot we need to:

- bind the plot to a specific data frame using the `data` argument

```
ggplot(data = boulder_precip)
```

- define aesthetics (`aes`), by selecting the variables to be plotted and the variables to define the presentation such as plotting size, shape color, etc.,

```
ggplot(data = boulder_precip, aes(x = DATE, y = PRECIP))
```

- add `geoms` – graphical representation of the data in the plot (points, lines, bars). To add a geom to the plot use `+` operator:

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
  geom_point()
```

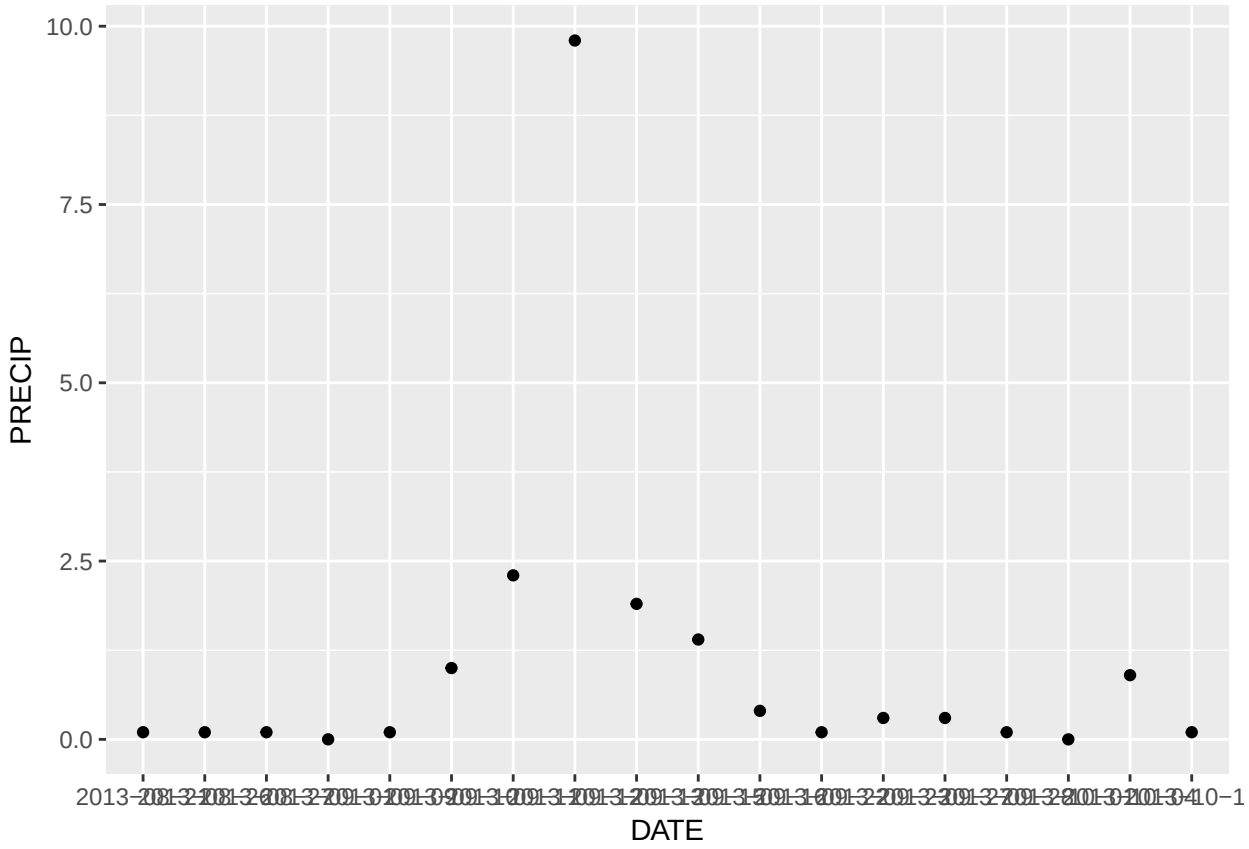

Figure 2: ggplot boulder precip

The `+` in the `ggplot2` package is particularly useful because it allows you to modify existing `ggplot` objects. This means you can easily set up plot "templates" and conveniently explore different types of plots, so the above plot can also be generated with code like this:

```
# Create the plot object (nothing will render on your screen)
precip_plot <-  ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP))

# Draw the plot
precip_plot + geom_point()
```

We can also apply a color to our points

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_point(color = "blue")
```

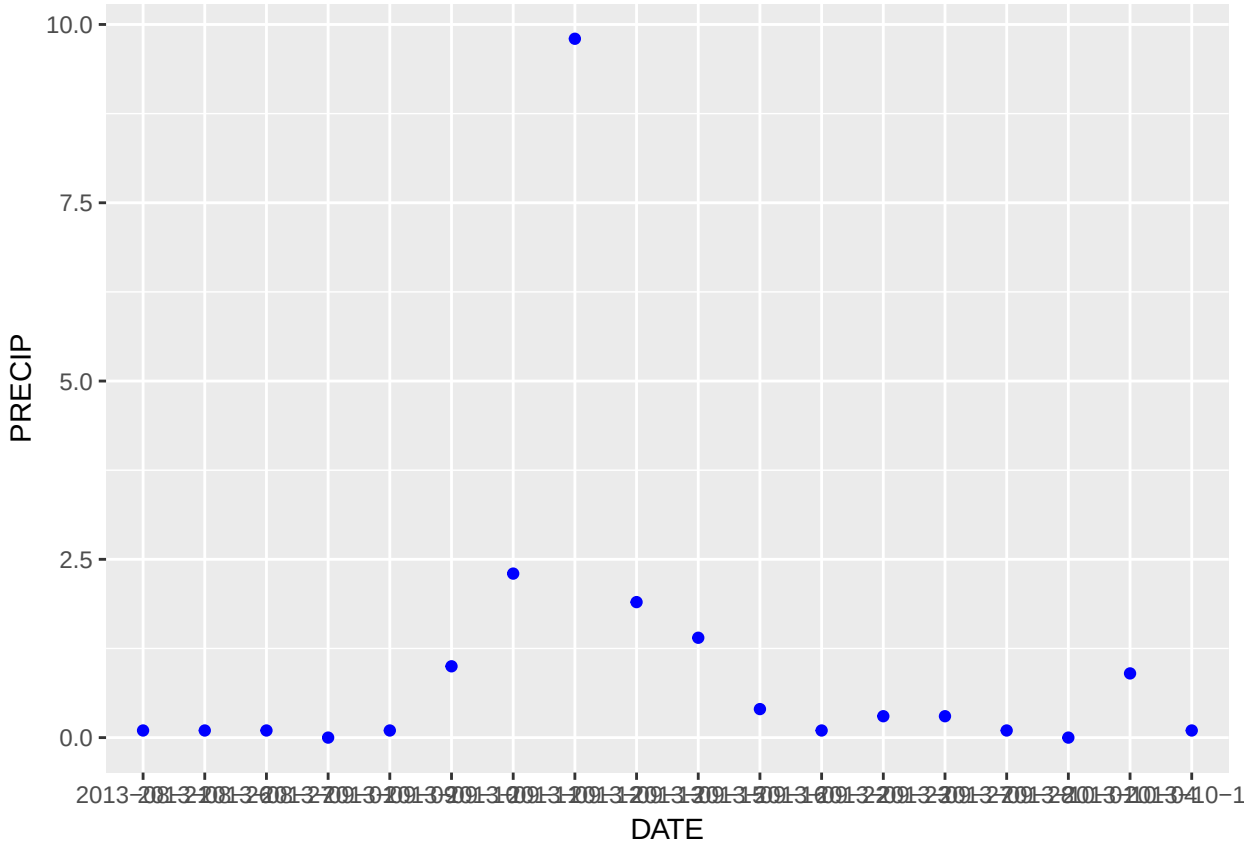

Figure 3: ggplot with blue points

And adjust the transparency.

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_point(alpha=.5, color = "blue")
```

Or to color each value in the plot differently:

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_point(alpha = 0.9, aes(color=PRECIP))
```

We can turn our plot into a bar plot.

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_bar(stat="identity")
```

Turn the bar outlines blue

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_bar(stat="identity", color="blue")
```

Change the fill to bright green.

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_bar(stat="identity", color="blue", fill="green")
```

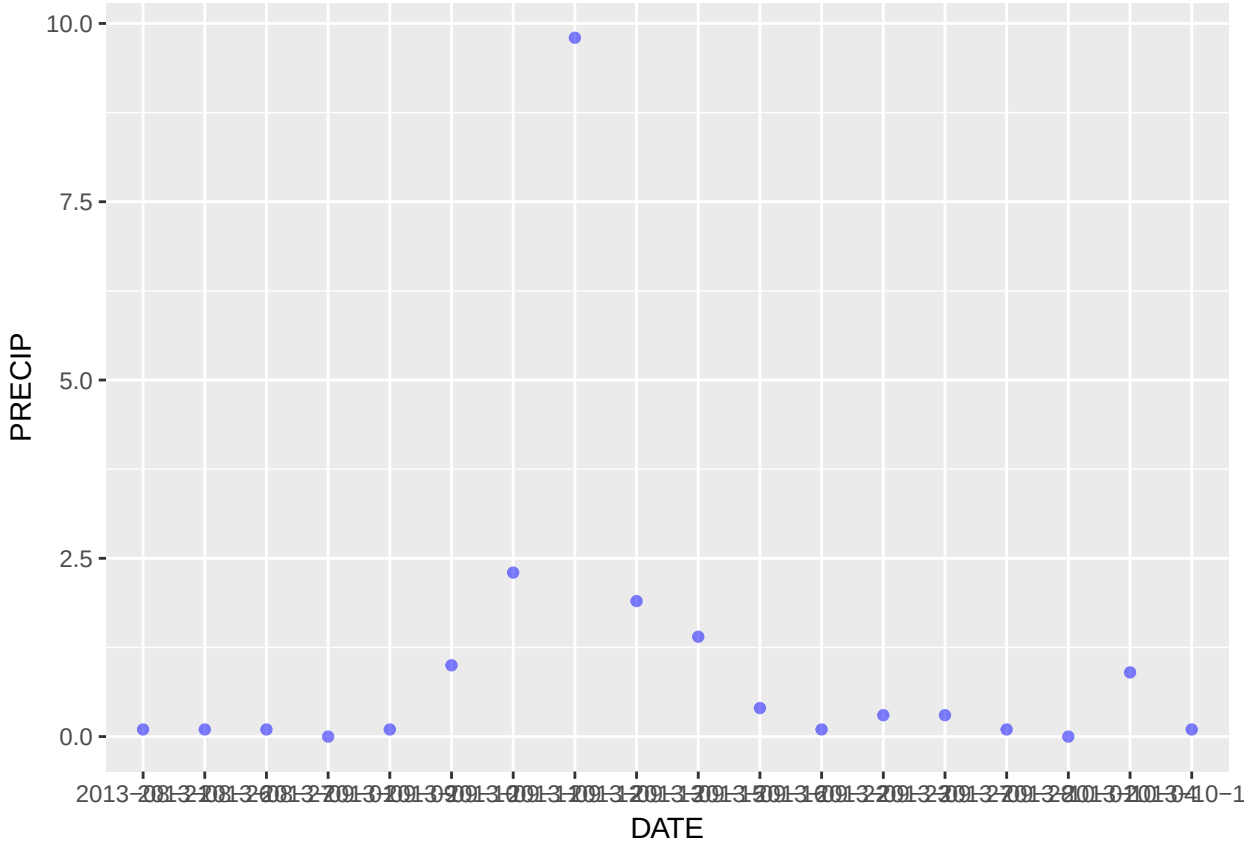

Figure 4: ggplot with blue points and alpha

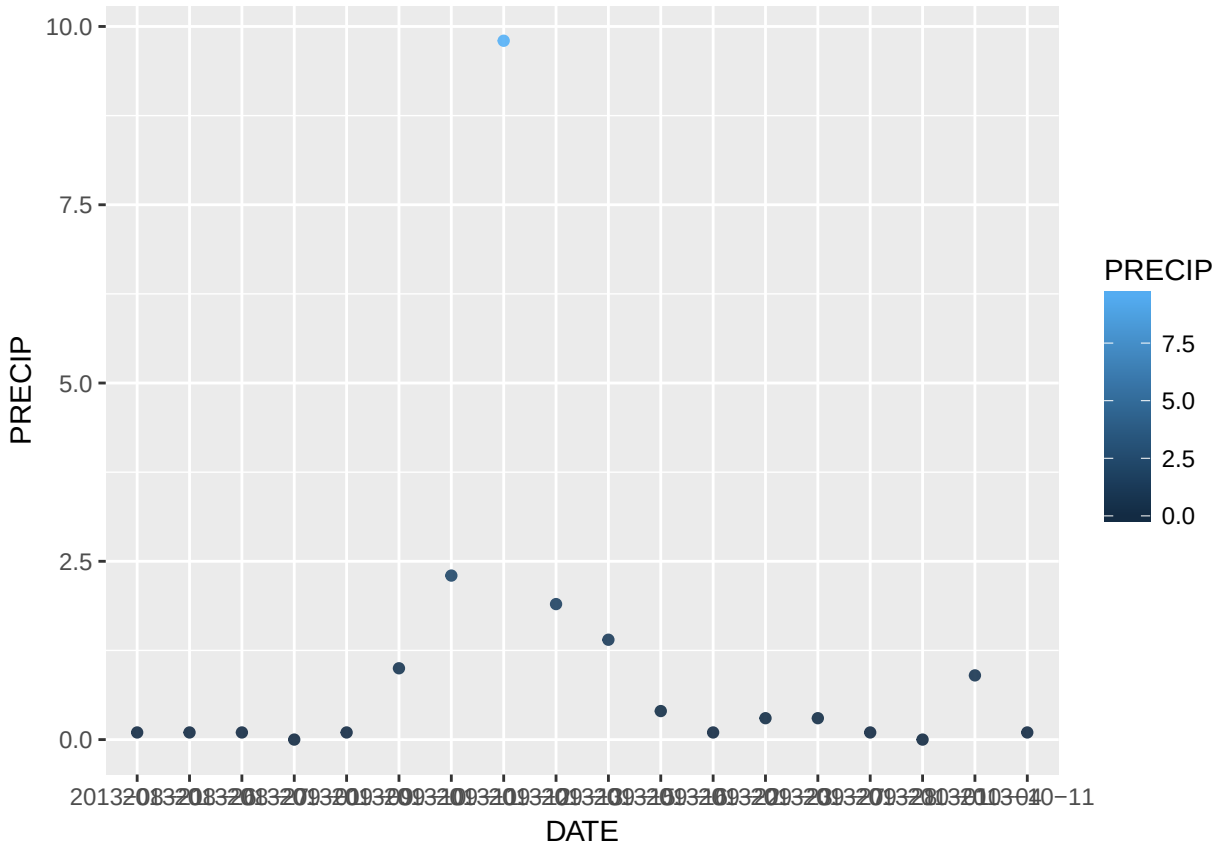

Figure 5: ggplot with colored points

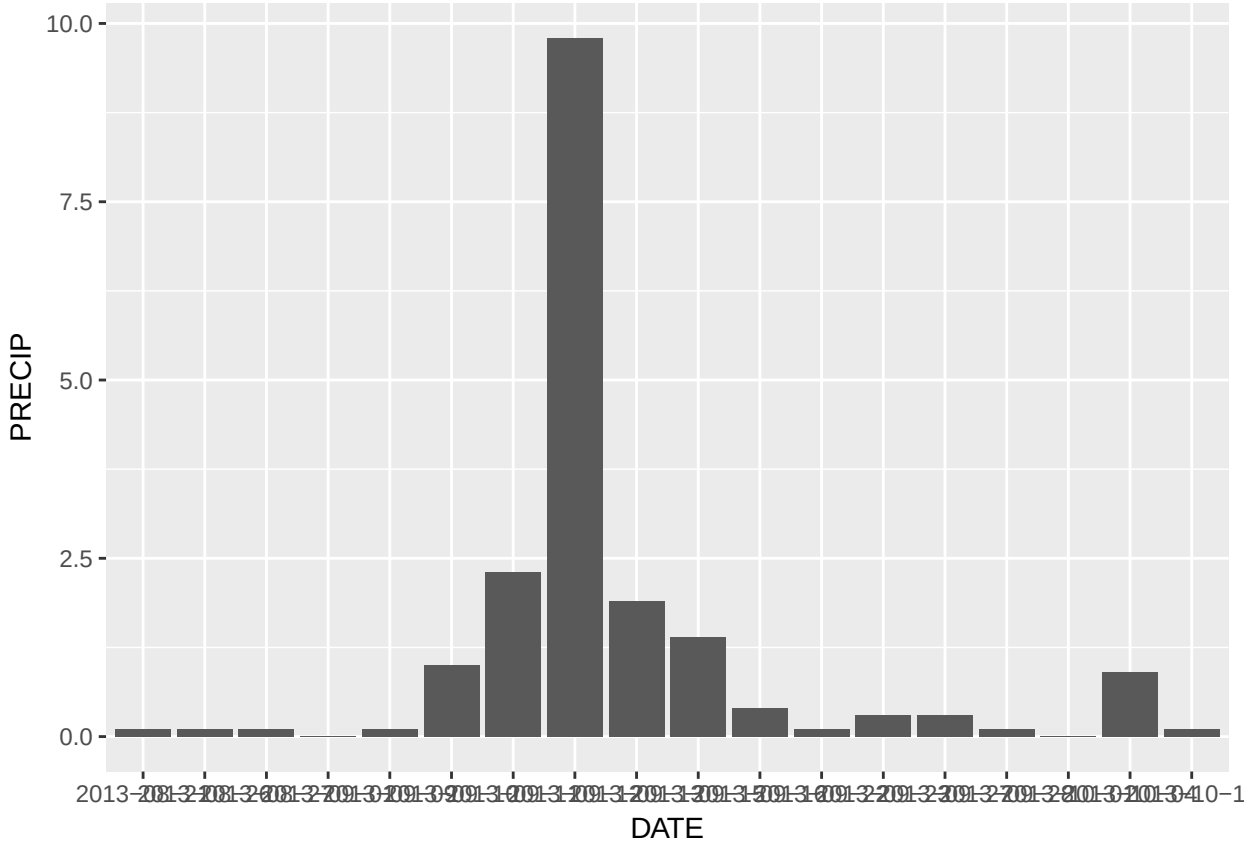

Figure 6: ggplot with bars

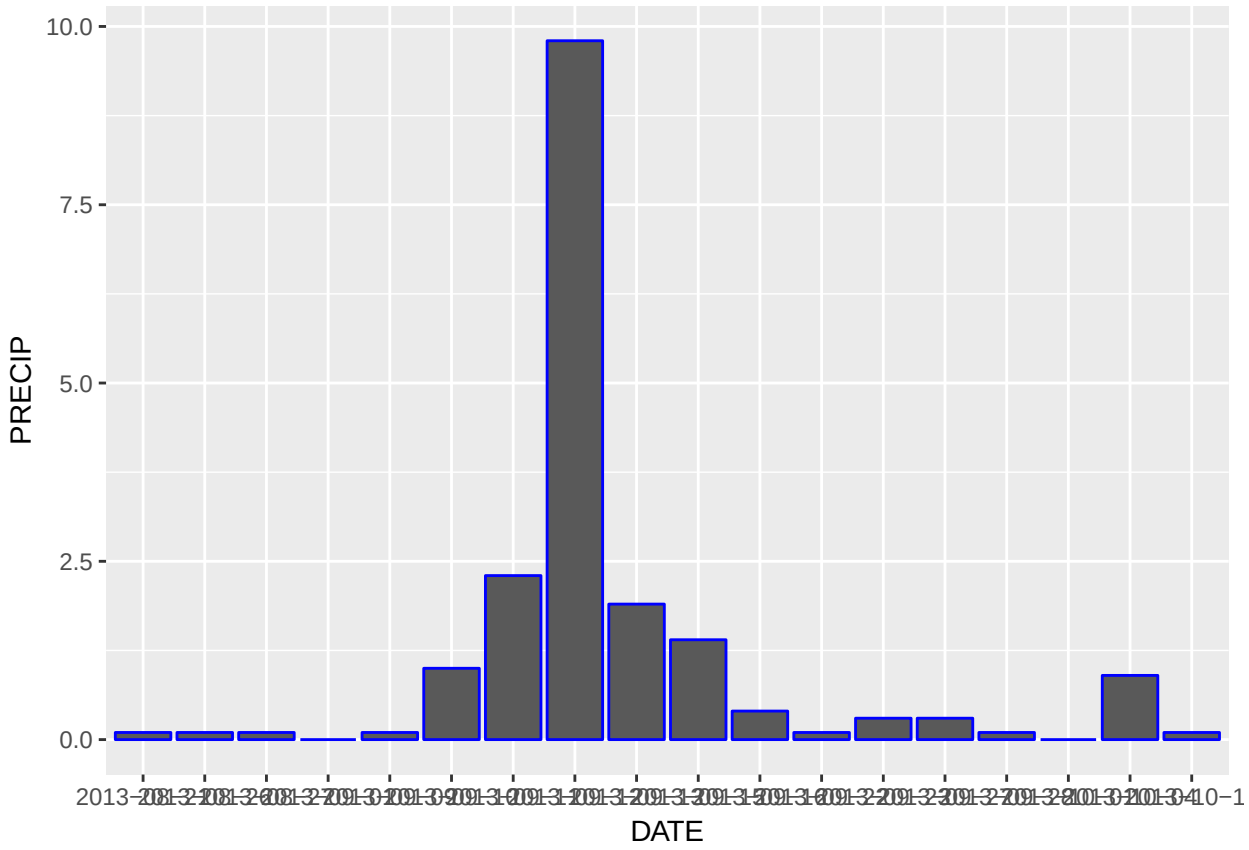

Figure 7: ggplot with blue bars

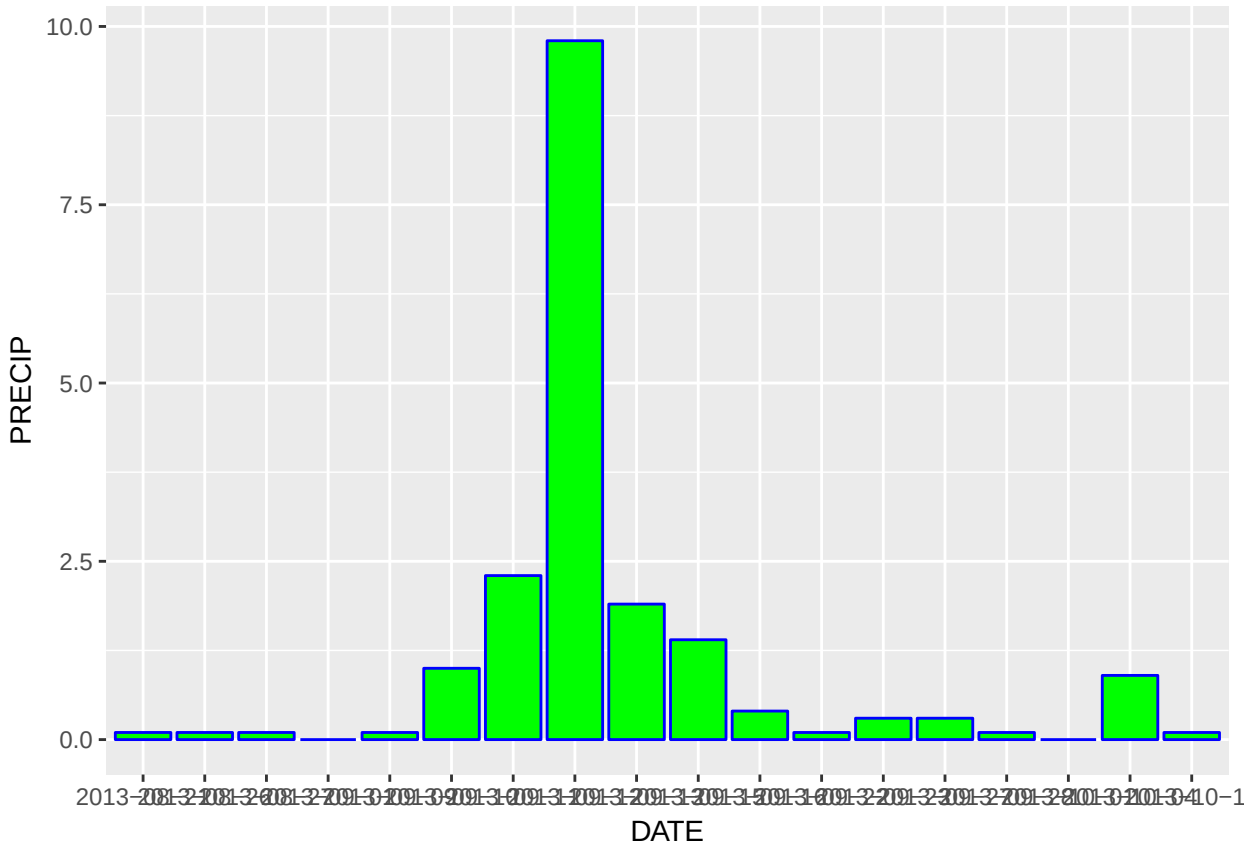

Figure 8: ggplot with green bars

Adding Labels

You can add labels to your plots as well. Let's add a title, and x and y labels.

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_point(alpha = 0.9, aes(color=PRECIP)) +
    ggtitle("Precipitation - Boulder, Colorado 2013")
```

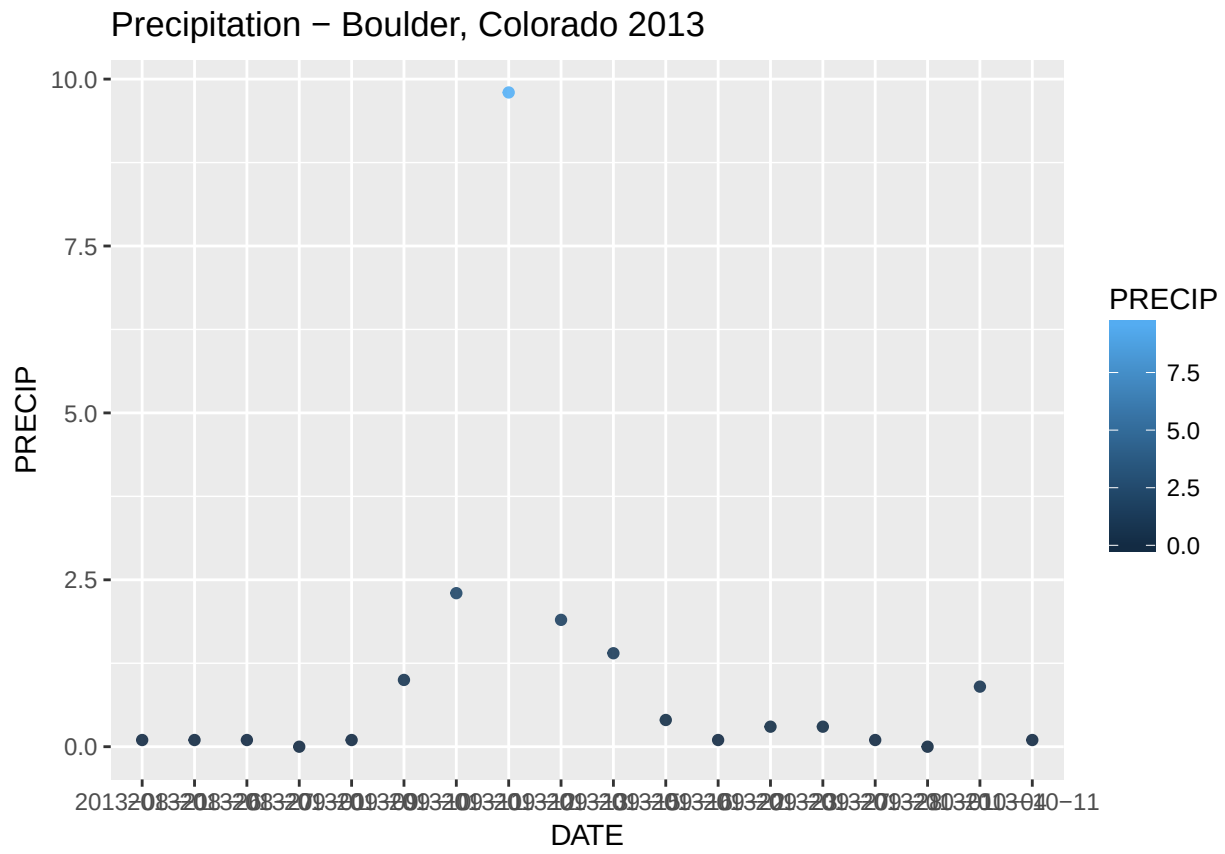

Figure 9: ggplot with labels

x and y labels...

```
ggplot(data = boulder_precip,  aes(x = DATE, y = PRECIP)) +
    geom_point(alpha = 0.9, aes(color=PRECIP)) +
    ggtitle("Precipitation - Boulder, Colorado 2013") +
    xlab("x label here") + ylab("y label here")
```

More on customizing your plots

There are many different tutorials out there on customizing ggplot plots. A few are listed below.

- Data carpentry ggplot2
- R Cookbook

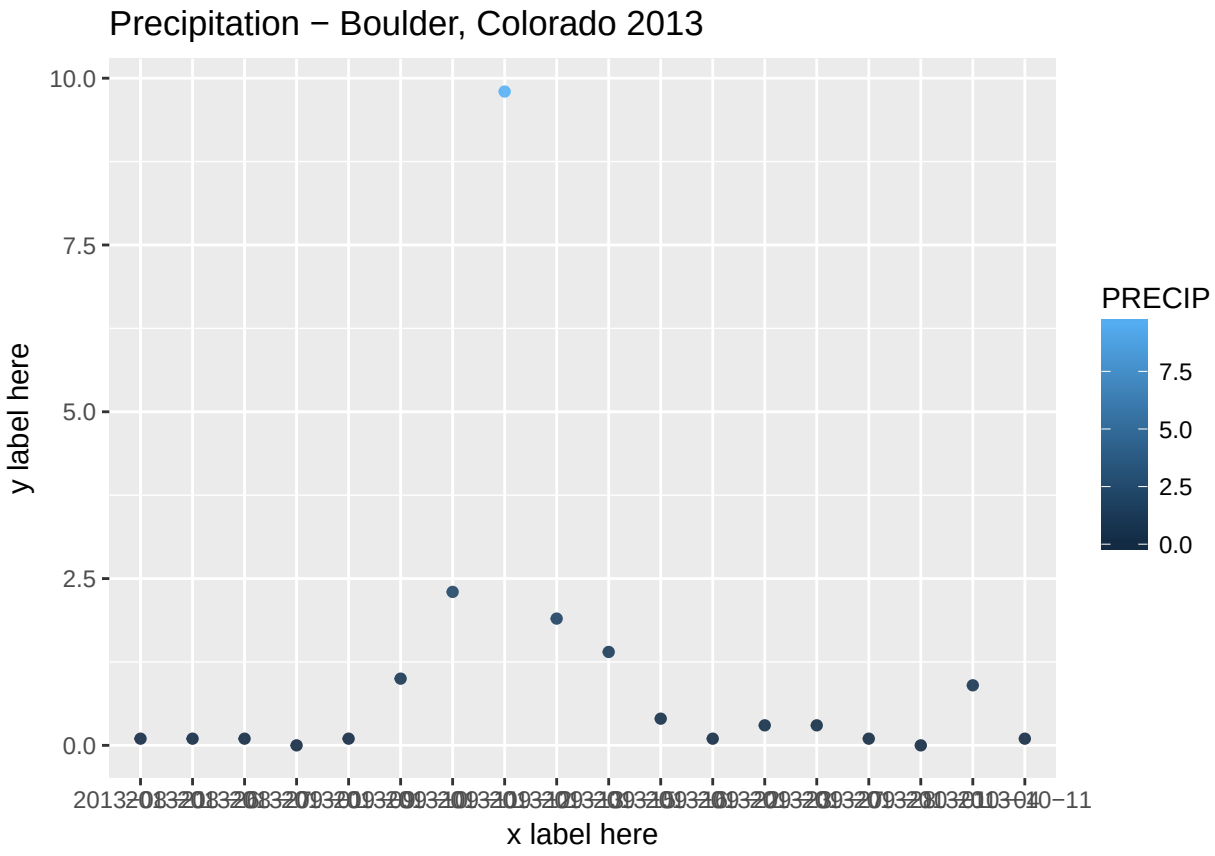

Figure 10: ggplot with title and labels

Optional challenge

Customize the plot that we created in last weeks homework as follows:

- Change the colors of the plot
- Plot the data using points `geom_point()` AND bars `geom_bar()`
- Add a title to your plot and x and y axis labels.