Visual Presentation of Data using R

March 2, 2019

1 The Perfect Graphic

1.1 Best Practices

There is no such thing as the perfect graphic, but there are conventions that can be used to guide us to create accurtate, accessible, and visually pleasing grahics. But like many things, it takes some practices.

Here are some general rules:

- Be sure that you introduce the graphic/table with text – i.e. text first, then graphic.
- Cite the graphic/table with a figure or table number.
- Describe the graphic/table with a caption.
- Manage data range and transformations to effectively analyze and display the data.
- Make sure the axes are labeled with appropriate units
- Manage axes label and values font size and orientation to make them easy to reads.
- Avoid graphic titles unless you have more than one panel, i.e. graphics that are side by side or on top of each other.
- Do not connect data points with lines unless you can 'reasonable' interpolate between the points, e.g. a continous data set with some level of autocorrelation.
- Are the graphics accessible? For example, black and white can be better than color in terms of accessibility (universal design) and sustainability.
- Use the caption to describe what the reader is supposed to see in the figure.

1.2 How to Cite Software

In the text, students often make a bigger deal out the software than it deserses. Probably, because we feel like we climbed a big mountain to have some success and want demonstrate that. However, in general, environmental scientists downplay the software, unless they wrote a specific function or library.

Thus, for our purposes, the following is usually sufficient...
"Statistical analysis was conducted using using R (CRAN 2019)."

You don't need to mention how you imported it, used Rstudio, or talk about the functions. In the text, you might mention that you used a linear model, regression, analysis of variance (AOV), but the details of the R code is usually not mentioned.

1.3 Reporting Results

In statistics, we reject the null hypothesis – and don't prove anything. So, we need to be careful how we report our results.

First, when we report statistics, do not report the p-value if the result in non-significant. Just report that null hypothesis cannot be rejected. If the p-value is less then 0.001, just report it as '$p < 0.001$'. If the p-value is between 0.05 and 0.001, then I suggest you report the actual value (rounded to the nearest one-hundredth or thousandth as appropriate.

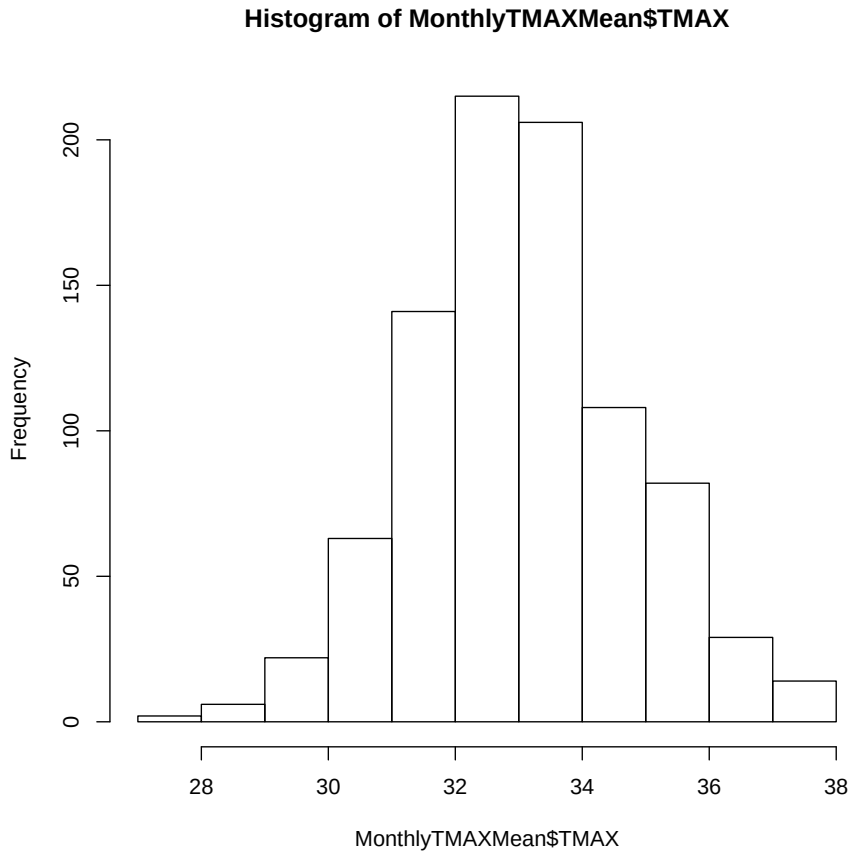
Second, when we report the r^2 value, round to the nearest one-hundredth and describe it as the 'amount of variation explained by the model'. In fact, if you multiply by 100, it's can thought of as a percent variation explained.

2 Exploring the Histogram

Data exploration can include many steps, but starting with a histogram gives the researcher the ability to evaluate the distribution of the data.

Below is a default histogram for TMAX values, where we might be able to visually how normally distributed the data might be.

```
hist(MonthlyTMAXMean$TMAX)
```



The default graphic is hideous – so, let's start fixing it.

2.1 Title and Axis Labels

For stand alone figures, we usually add titles, but in papers and lab reports it's a good practice to remove the title and use the caption to describe the graphic. Changes to the title can be made with arguments within the plot command, i.e. 'main=NULL'.

In addition, we can change the x-axis label, with the 'xlab' argument. Specifying the units is also required. And in this case, we want to add the °symbol and create a text string with the axis label in quotes that can be referenced in the hist() funtion.

```
TMAXlabel <- "Maximum Temperature (C)"
hist(MonthlyTMAXMean$TMAX, main=NULL, xlab=TMAXlabel)
```

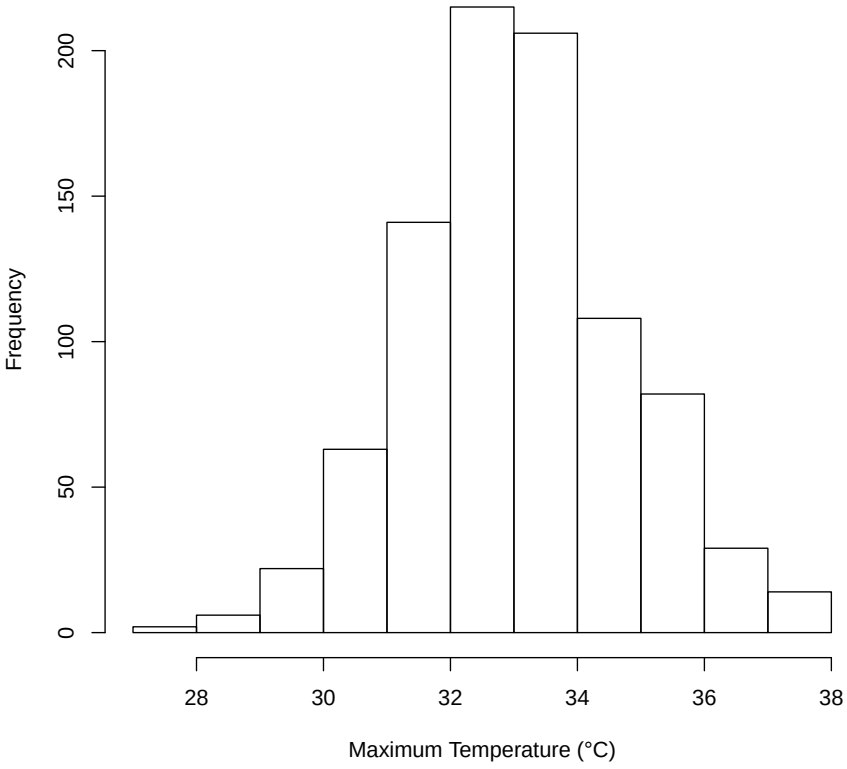


Figure 1: Histogram of Maximum Temperatures (C) (XXX, Thailand, 1940-2018)

2.2 Putting Multiple Figures in a Row

To create two graphics in one row, we can change the graphic parameters with the par() function. In this case, we'll create two column panels in one row using the 'mfrow' option and a vector that defines the number of rows and the number of columns. It's often a good idea to set the graphic parameter back to the default afterwards. In this case, I added a title because we have a panel with two graphics. Often people will put letters, e.g. A and B to refer to each one separately, but I prefer to put the actually description in the title, so the reader doesn't have to go back and forth between the caption and the figures.

```
par(mfrow=c(1,2))
hist(MonthlyTMAXMean$TMAX, main='Maximum Temperature', xlab=TMAXlabel)
TMINlabel <- "Minimum Temperature (C)"
hist(MonthlyTMINMean$TMIN, main='Minimum Temperature', xlab=TMINlabel)
```

```
par(mfrow=c(1,1))
```

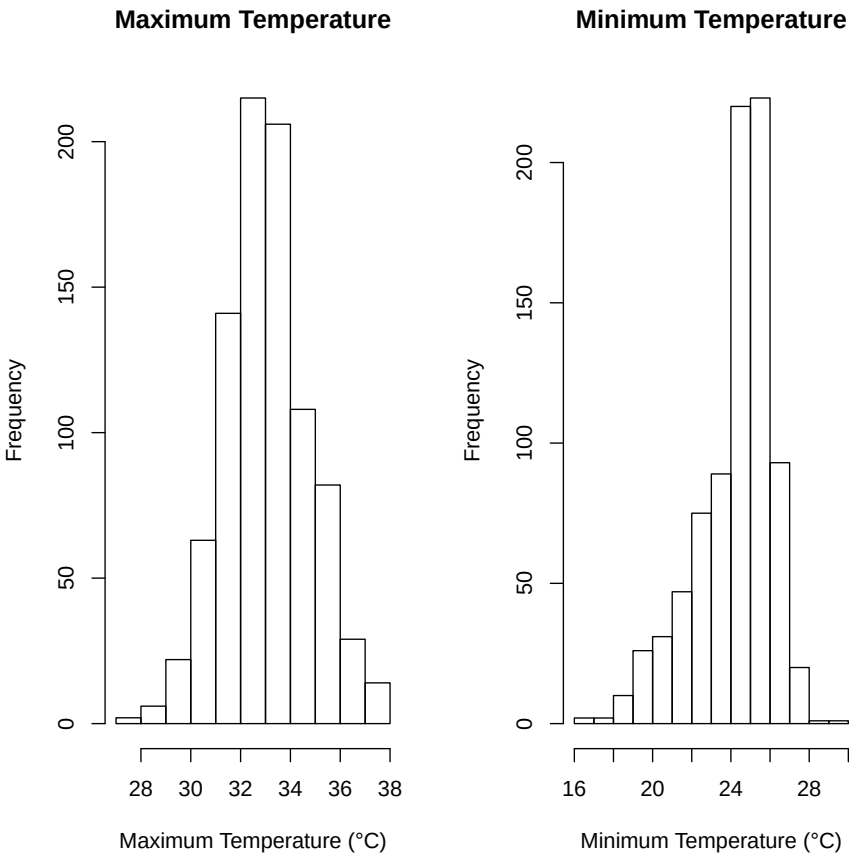


Figure 2: Mean monthly maximum and minimum tempetures (C) ((XXX, Thailand, 1940-2018))

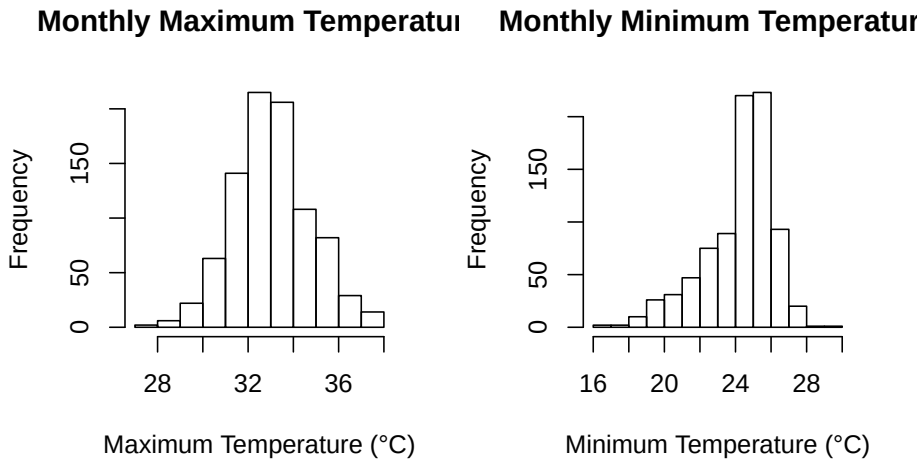


Figure 3: Mean monthly maximum and minimum tempetures (C) ((XXX, Thailand, 1940-2018))

Because the figure is rather distorted, I have constrained the size using a fig.height and fig.width option.

```
par(mfrow=c(1,2))
hist(MonthlyTMAXMean$TMAX, main='Monthly Maximum Temperature', xlab=TMAXlabel)
TMINlabel <- "Minimum Temperature (C)"
hist(MonthlyTMINMean$TMIN, main='Monthly Minimum Temperature', xlab=TMINlabel)
```

```
par(mfrow=c(1,1))
```

3 Boxplot

Box plots are great ways to display quantitative data from controlled experiments. For example, if you had high and low treatment categories and measured some response. Let's use the following example, ants colonies collected from three locations farm, grassland, and forest.

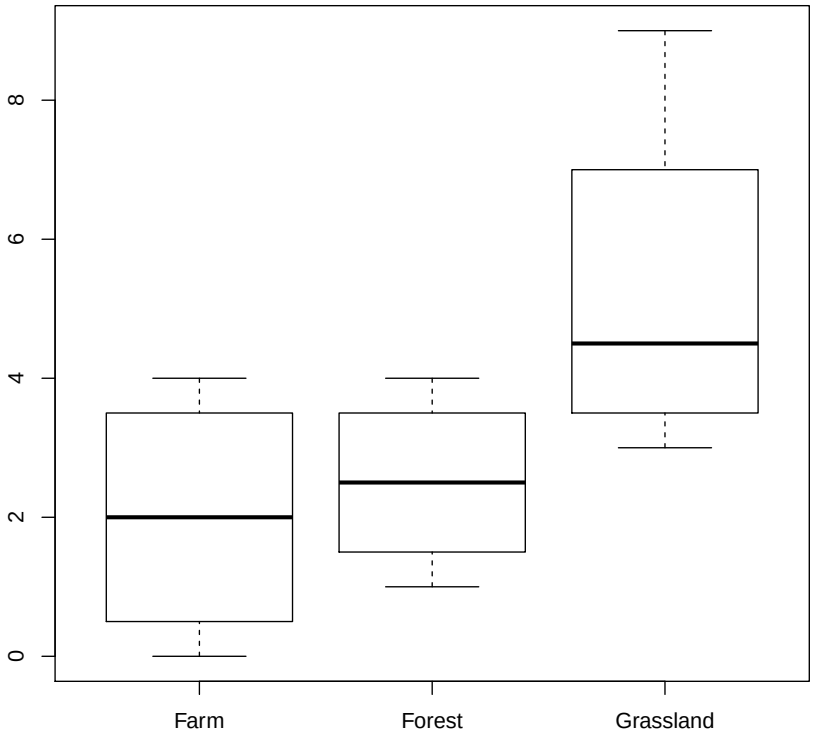


Figure 4: Number of ant colonies by habitat type

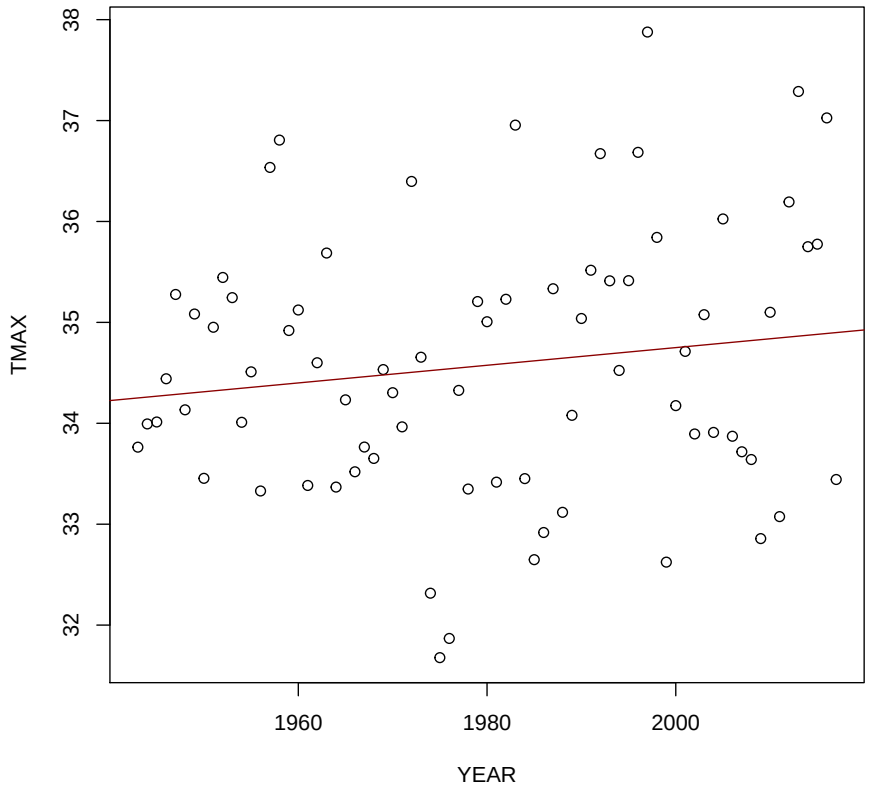
4 Scatter Plot – Non-time series

4.1 Scatter Plot – Time Series

Time series data sets often require special treatment. For example, data are often autocorrelated, thus data can be represented by lines instead of points. However, this requires some careful thought.

When we are graphing temperature data from one year to the next, we are averaging many days and connecting one year to the next might be appropriate if the data series is long enough, e.g. more than 50 years. However, with shorter time-series, i.e. 15 or less, connect the years with a line might be problematic.

```
plot(TMAX ~ YEAR, data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,])
abline(coef(lm(TMAX ~ YEAR, data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,])),
    col='darkred')
```



Let's fix the y-axis label as we did above (TMAX is not a very helpful label!). Furthermore, the x-axis needs to be calmed down some, so let's change the case

for these. We will also change the symbols to make it less busy with the 'pch' argument. You can look online to see the choices one has in R.

I am also not impress with the vertical orientation of the y-axis, so it's important to change these as well.

Finally, it's important that the image works in black and white. So, let's see if we can modify the graphic to make it less resource intensive. Finally, let's add a caption and reference to the figure (Figure 5).

```
ylabel <- "Maximum Temperature (C)"
plot(TMAX ~ YEAR, data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,],
     ylab=ylabel, xlab='Year', pch=20, las=1, col='gray')

abline(coef(lm(TMAX ~ YEAR,
    data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,])), col='black')
```

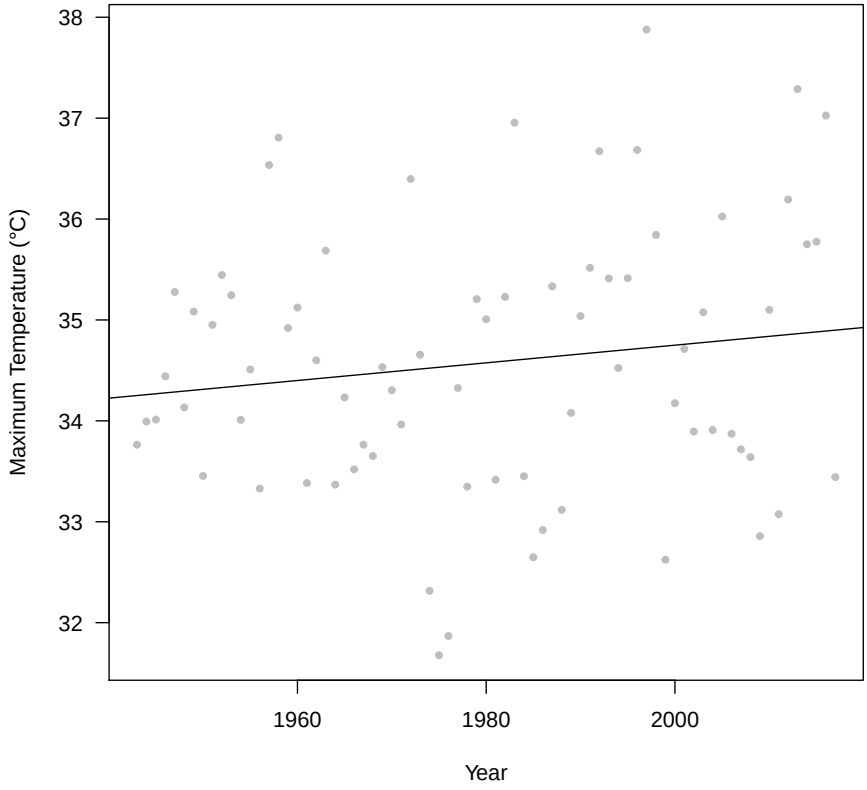


Figure 5: Monthly Average of Daily Maximum Temperatures (°C). Notice the slightly darker line in the x-axis for the middle section. I am not sure how to get rid of this, but it bugs me!

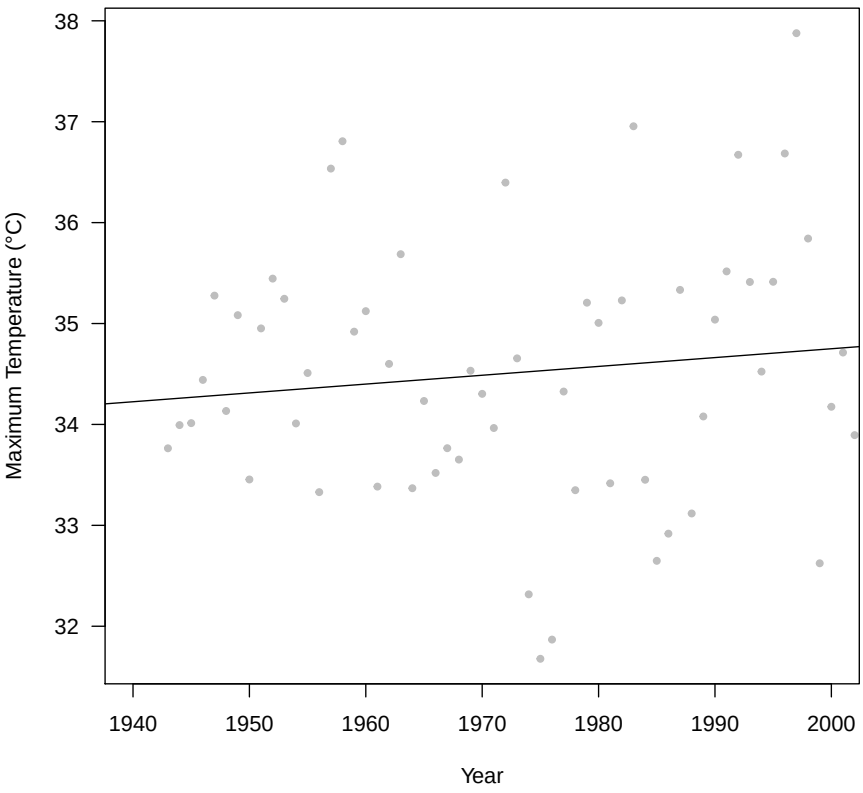


Figure 6: Monthly Average of Daily Maximum Temperatures (C).

Now, what if we only want to display part of the data. We can limit the x-axis range using the 'xlim' argument, where we create a vector of for the start and end of the range.

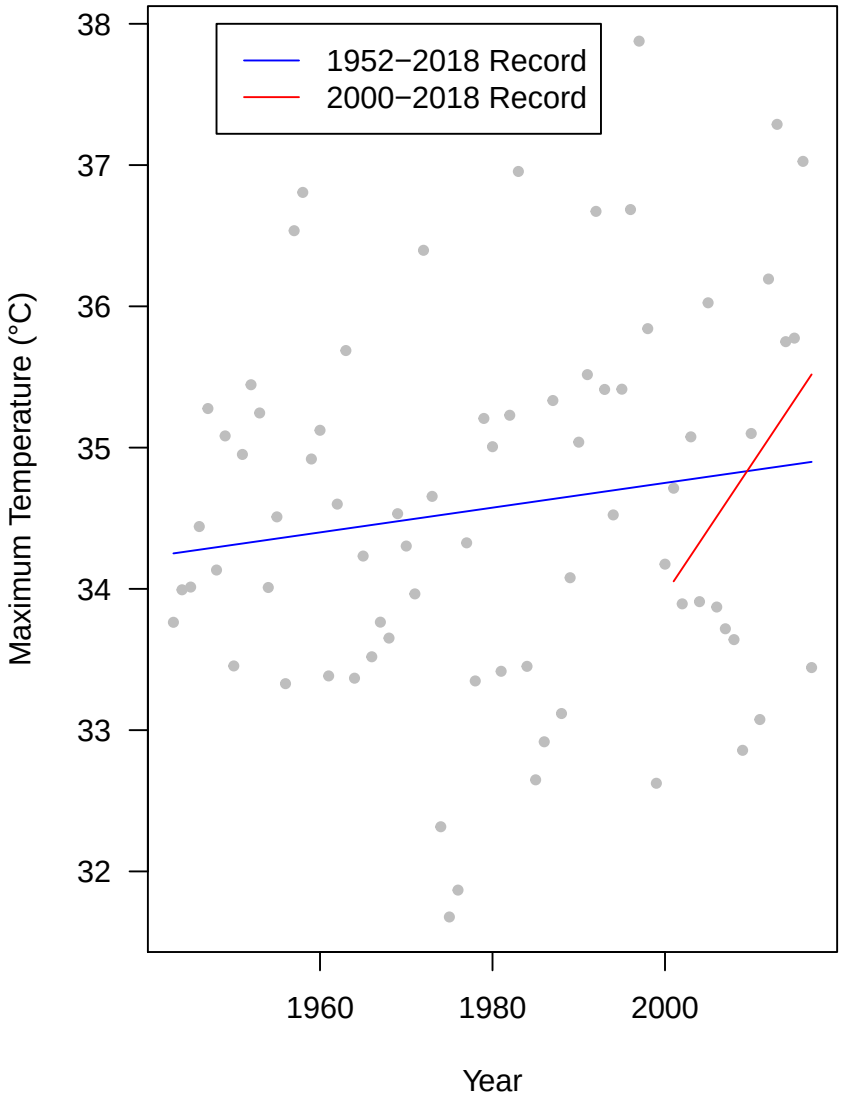
```
ylabel <- "Maximum Temperature (C)"
plot(TMAX ~ YEAR, data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,],
     xlim=c(1940, 2000),
     ylab=ylabel, xlab='Year', pch=20, las=1, col='gray')

abline(coef(lm(TMAX ~ YEAR,
    data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,])), col='black')
```

Alternatively, you may want to creat a best fit line that only covers the range for the existing data without extrapolating, which is usually a very good idea

for most scientific endeavors!

For example, we have seen several papers that select parts of the record to make dubious claims.

```
ylabel <- "Maximum Temperature (C)"
plot(TMAX ~ YEAR, data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,],
     #xlim=c(1940, 2000),
     ylab=ylabel, xlab='Year', pch=20, las=1, col='gray')

MonthlyTMAX.lm = (lm(TMAX ~ YEAR,
    data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5, ]))
interpolated = predict(MonthlyTMAX.lm,
      MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5,])

lines(MonthlyTMAXMean$YEAR[MonthlyTMAXMean$MONTH==5],
      interpolated, col='blue')

MonthlyTMAX.lm2 = (lm(TMAX ~ YEAR,
    data=MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5 &
            MonthlyTMAXMean$YEAR>2000, ]))

interpolated2 = predict(MonthlyTMAX.lm2,
            MonthlyTMAXMean[MonthlyTMAXMean$MONTH==5 &
            MonthlyTMAXMean$YEAR>2000,])

lines(MonthlyTMAXMean$YEAR[MonthlyTMAXMean$MONTH==5 &
            MonthlyTMAXMean$YEAR>2000], interpolated2, col='red')

legend(1948, 38, legend=c("1952-2018 Record", "2000-2018 Record"),
       lty=1, col=c("blue", "red"))
```


1952−2018 Record
2000−2018 Record
Maximum Temperature (°C)
38
37
36
35
34
33
32
1960
1980
2000
Year

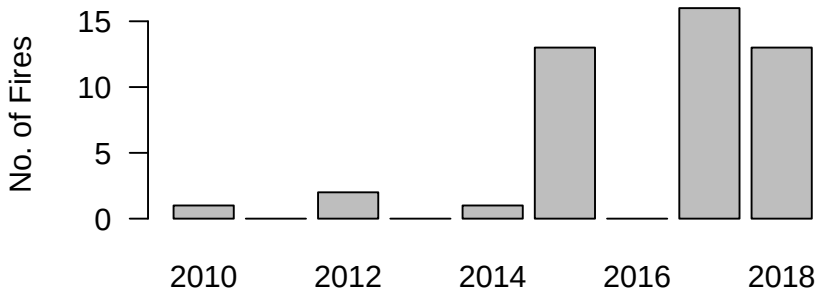


Figure 7: Number of Unsafe Air Quality Days in Washington State due to Fires.

5 Bar Graphs

Rainfall data and other types of 'count' data are often best displayed as bar-charts.

```
fires = data.frame(Year = 2010:2018, Fires = c(1, 0, 2, 0, 1, 13, 0, 16, 13))
barplot(fires$Fires, names.arg = fires$Year, las=1, ylab="No. of Fires")
```

Notice the figure cuts out every-other year. You'll need to decide if the reader will be bothered by that. If so, you can reduce the x-axis labels fonts or increase the width of the chart.

6 Tables

Sometimes (often times?), tables are better than graphs. In the case below, I summarized the grades for the first drafts using a table and a histogram. Which do you think is more effective?

```
print(xtable(grades_tab,
          caption='Table of First Blog Draft Scores', table.placement =''),
          caption.placement = "top")
```

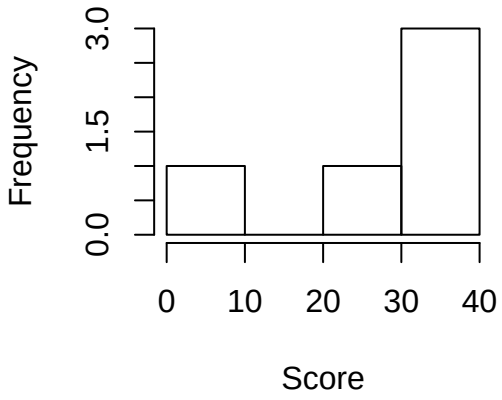


Figure 8: Histogram of First Blog Draft Scores

	V1
(0,10]	1
(10,20]	0
(20,30]	1
(30,40]	3
(40,50]	0

```
hist(grades, main=NULL, xlab="Score")
```