Lab 0: (Re)Introduction to R

STAT 244NF: Infectious Disease Modeling

Sample SOLUTIONS

Getting Started

Before you proceed with the lab, please complete the following steps:

1. Under the Environment tab in the upper right window of your RStudio session, click the broom icon. This will clear all objects from your workspace, meaning it will clear the Global Environment.
2. Go to Session > Restart R and Clear Output. This will start a new R session.

You should perform these steps every time you start something new in R. It should help head off some knitting errors.

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com and https://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf.

When you click the **Knit** button at the top of this page a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. An R code chunk looks like this:

```
5 + 2
```

```
## [1] 7
```

```
log(7)
```

```
## [1] 1.94591
```

There are three main ways to run R code. First, whenever you knit the document, all chunks will be run in a "fresh" R session. This means that if your code is out of order, or you have omitted one or more lines of code from your R Markdown file, your code will break and your file will not knit.

However, as you're going along you will also want to run commands in a working session so that you can check that your code runs without having to knit the whole document. To do that, you can run individual code chunks by clicking the green "Play" arrow at the top right corner of the chunk.

You can also select individual lines of code you want to run and choose "Run... Run Selected Line(s)" from the menu at the top of the editor window.

The second two of these approaches will send commands to your Console, at the bottom of the screen. **Except for in times of desperation, you never want to enter commands directly into the Console!** Any

commands you enter directly into the console will run one time only, and will not be a permanent part of your R Markdown document. **Always enter commands you want to save directly into your R Markdown document!!**.

Try out all three of those approaches with the example code chunk above.

Adding Code Chunks

You can add your own code chunks to a document yourself. This can be accomplished by clicking the green C button that has a plus sign in the upper left corner and selecting R from the drop-down menu.

Try adding your own code chunk and use it to find out what the square root of 21 is.

```
sqrt(21)
```

```
## [1] 4.582576
```

Loading Packages

R comes with a decent amount of built-in functionality, but to do anything useful you will need to load *packages* that contain additional functionality. You load packages with the `library` command. Here we will load 3 packages with functionality you will need for this lab: mosaic, dplyr, and ggplot2. Run the code chunk below to load these packages. **If you are working with these packages for the first time, you may need to install them.** One way to do this is run the lines that are commented out (have # in front of them) below; you can run these by removing the #. This only needs to be done once. After the packages are installed, you only need to run the lines with the library function.

```
# install.packages("dplyr")
# install.packages("ggplot2")
# install.packages("incidence")
# install.packages("outbreaks")
# install.packages("readr")

library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.0
```

```
library(incidence)
```

```
## Warning: package 'incidence' was built under R version 4.2.3
```

```
library(outbreaks)
```

```
## Warning: package 'outbreaks' was built under R version 4.3.0
```

```
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.3.0
```

Reading in Data

Data can be read in from a variety of locations, including R packages (like the "outbreaks" package you just loaded), websites, and from files stored on your computer. We will primarily by working with data from the first two locations. For today, we will just load a data set from the outbreaks package.

```
data(dengue_fais_2011)
```

To learn more about the data, run `?dengue_fais_2011` in an R chunk.

Let's take a look at the data. Recall that useful functions include the `head()` function, `dim()` function, and `str()` functions. Run each one of these functions with `dengue_fais_2011` as the input. Describe the output for each function.

ggplot2

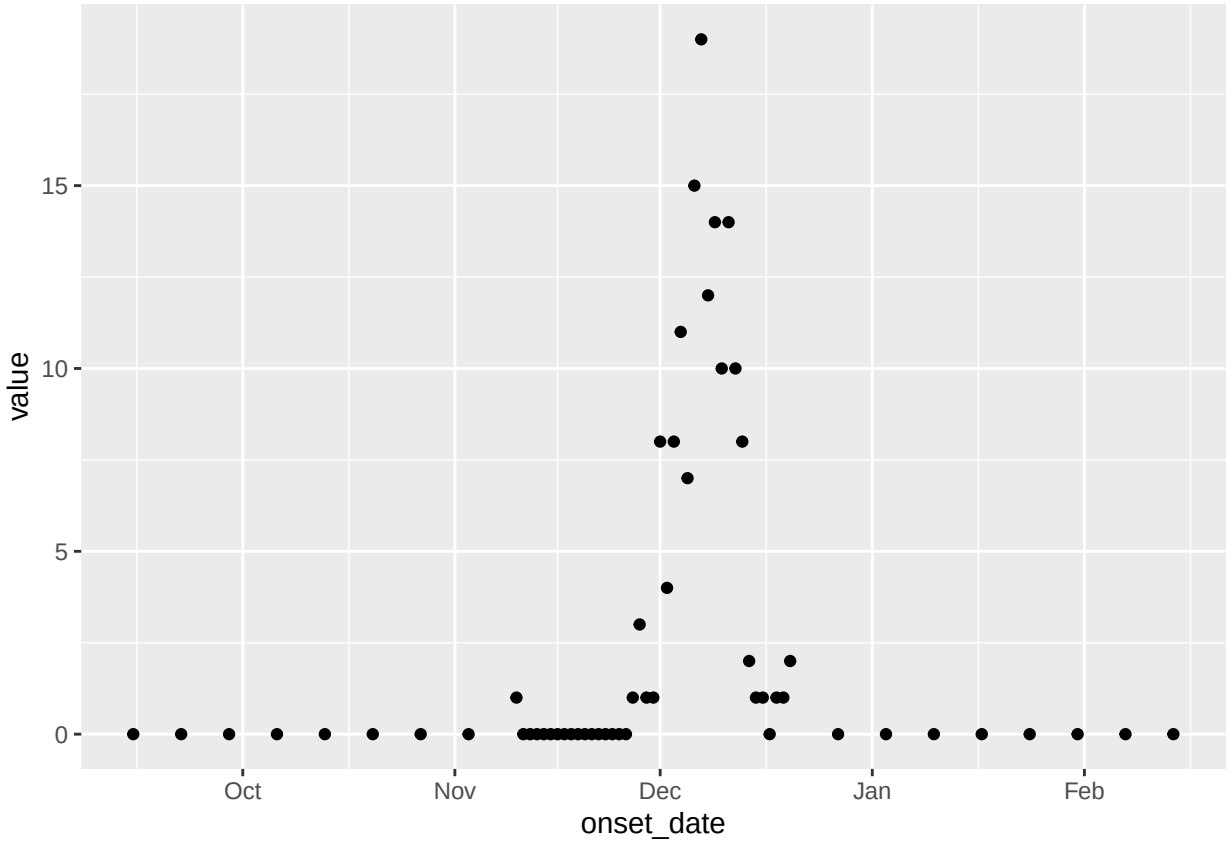
The ggplot2 package is frequently used for data visualization.

```
## Skeleton code - should not run anything

(ggplot(data=<name of data frame>,
        aes(x=<variable for x-axis>, y=<variable for y-axis>,
        color=<variable for color lines>,
        fill=<variable for color area>)) +
        geom_<geometry type>() +
        <optional other things like axis labels, ...>)
```

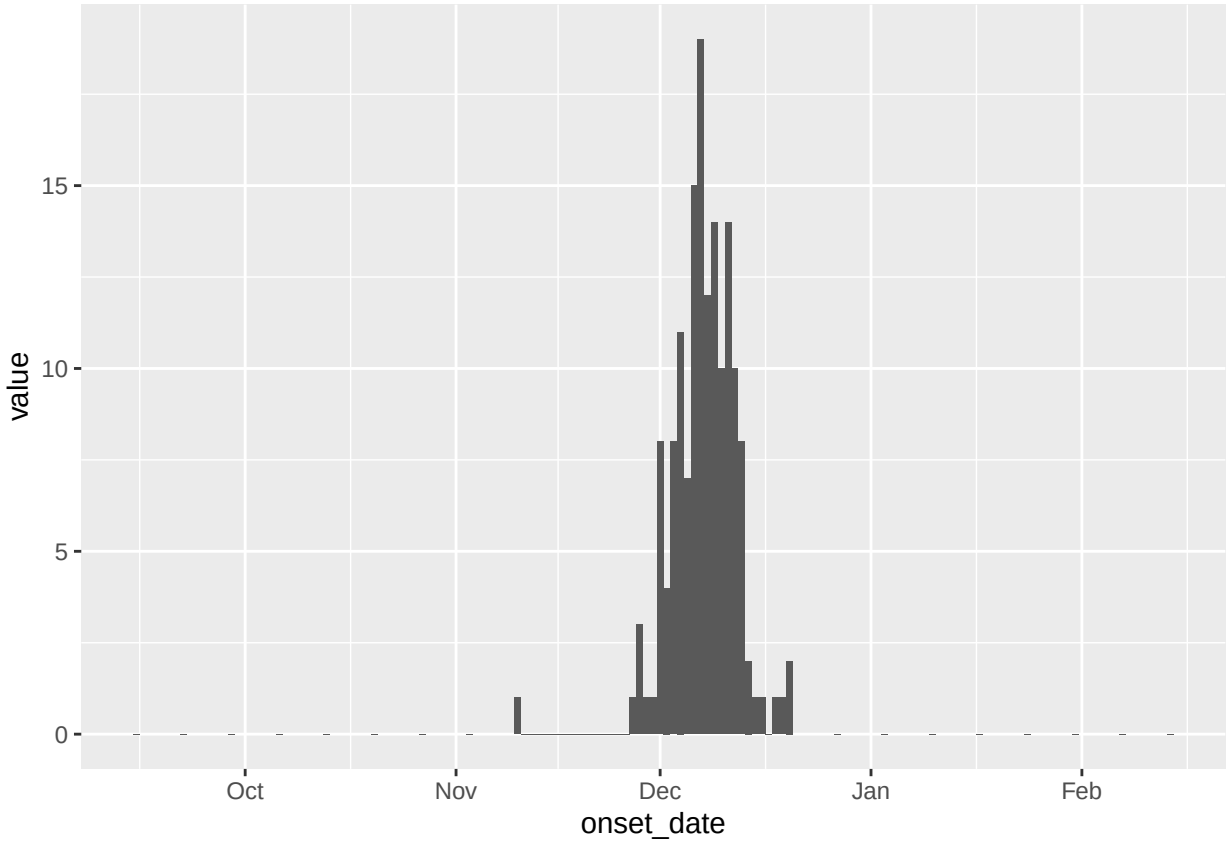
Create a plot of `value` (number of cases) versus `onset_date` using the geom_point geometry.

```
ggplot(data=dengue_fais_2011, aes(x=onset_date, y=value)) +
  geom_point()
```



Create a plot of `value` (number of cases) versus `onset_date` using the geom_col geometry.

```
ggplot(data=dengue_fais_2011, aes(x=onset_date, y=value)) +
  geom_col()
```

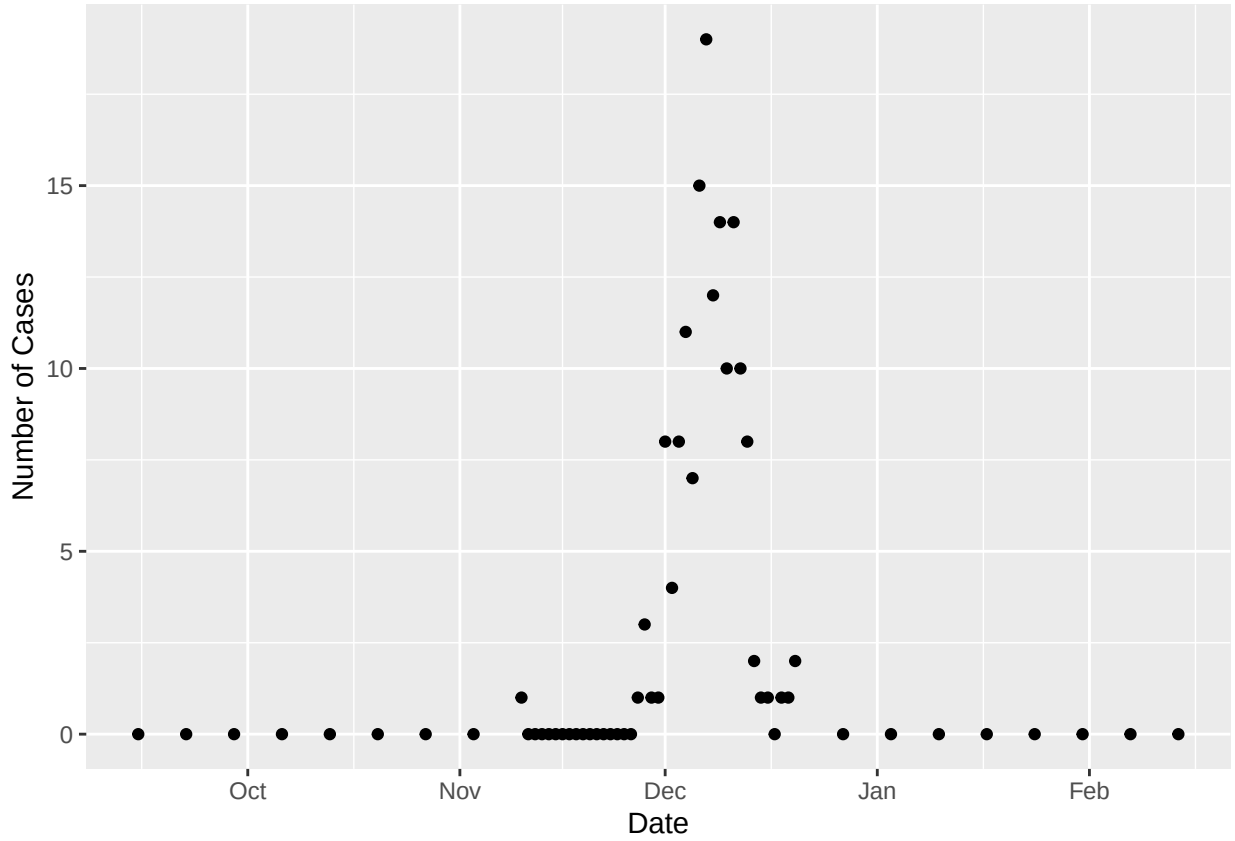


What is the difference between these plots?

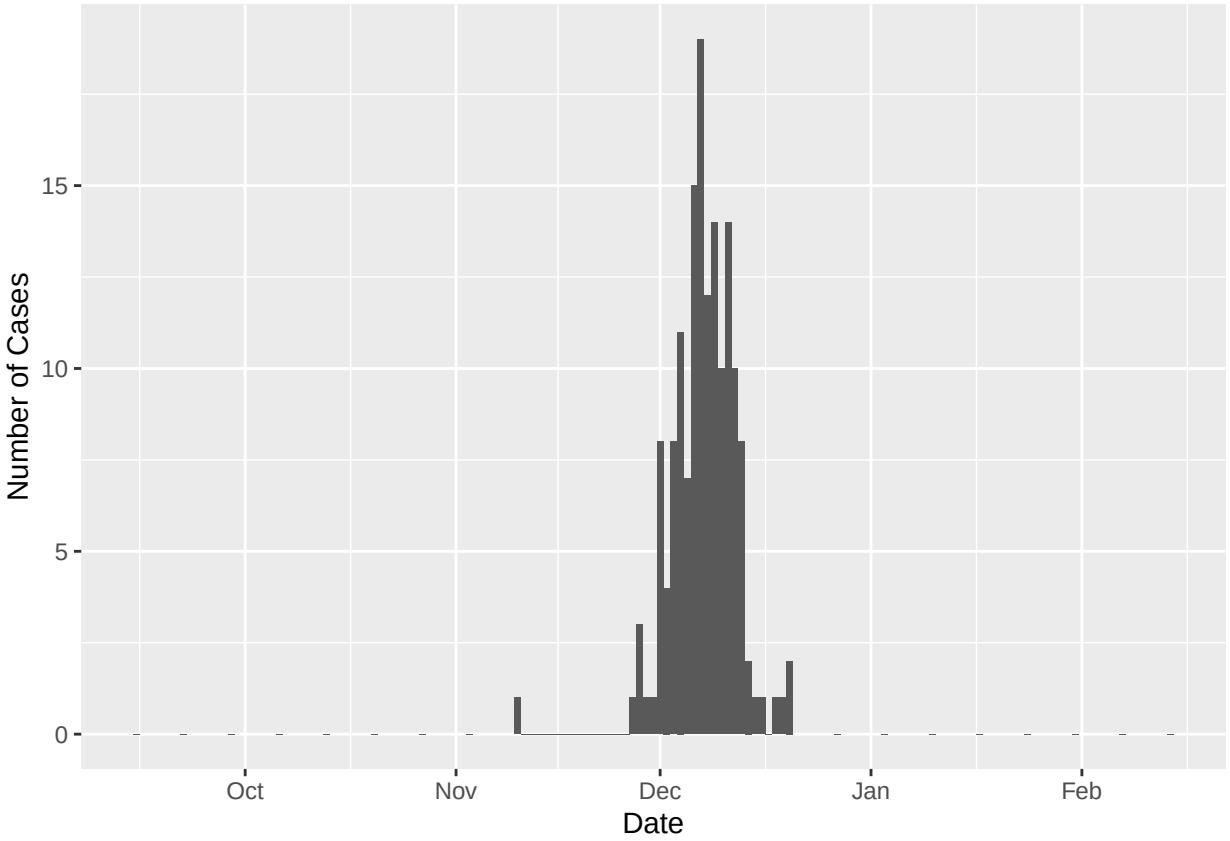
This is just a question about how changing small aspects about the code change the plot output. For the first plot, `geom_point` allows us to display the case counts as points. For the second plot, `geom_col` allows us to display the case counts as bars. This second one is similar to `geom_hist` but since the counts are already aggregated, rather than recorded one case per line, `geom_col` works better for us.

For both your plots, change the label on the x-axis to "Date" and change the label on the y-axis to "Number of Cases". You may find this resource helpful if you are not immediately sure how to do this: .

```
ggplot(data=dengue_fais_2011, aes(x=onset_date, y=value)) +
  geom_point() +
  xlab("Date") +
  ylab("Number of Cases")
```




```
ggplot(data=dengue_fais_2011, aes(x=onset_date, y=value)) +
  geom_col() +
  xlab("Date") +
  ylab("Number of Cases")
```



The plots that you just made will soon have significance to us as we study infectious diseases and model their behavior. A plot that shows the relationship between number of new cases and the onset date is called an incidence curve. We will define incidence more formally in lecture.