Tidyverse and ggplot (cont.)

Ed Gonzalez

```
install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/tz/sh20cj15711657_9_1d4v6m00000gn/T//RtmpEjRGTR/downloaded_packages
```

```
install.packages("nycflights13", repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/tz/sh20cj15711657_9_1d4v6m00000gn/T//RtmpEjRGTR/downloaded_packages
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
## Loading required package: timechange
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(nycflights13)

nyc <- nycflights13::flights
```

A little messy, but here I explain what each part of the coding does using the dplyr functions and then creating a new variable within the code where I calculate the arrival

```
nycJan <- nyc %>% # this assigns the following code to a variable
  filter(month == 1) %>% # selecting only for the months that have "1" listed
  filter(dest == "ORD" | dest == "DSM" | dest == "STL" | dest =="MCI" | dest == "MDW") %>% # this allow
  mutate(dep_time = dep_time/100) %>% #this transforms the times into a more recognizable format by add
  unite("dep", c(year, month, day, dep_time), sep="/", remove = FALSE) %>% #unite is merging the variab
  mutate(dep = ymd_hm(dep, tz = "America/New_York", quiet= TRUE)) %>% #this arranges the date and time
  filter(!is.na(dep))

nycJan
```

```
## # A tibble: 2,012 x 20
##    dep                 year month   day dep_t~1 sched~2 dep_d~3 arr_t~4 sched~5
##    <dttm>             <int> <int> <int>   <dbl>   <int>   <dbl>   <int>   <int>
##  1 2013-01-01 05:54:00  2013     1     1    5.54     558      -4     740     728
##  2 2013-01-01 05:58:00  2013     1     1    5.58     600      -2     753     745
##  3 2013-01-01 06:08:00  2013     1     1    6.08     600       8     807     735
##  4 2013-01-01 06:29:00  2013     1     1    6.29     630      -1     824     810
##  5 2013-01-01 06:56:00  2013     1     1    6.56     700      -4     854     850
##  6 2013-01-01 07:09:00  2013     1     1    7.09     700       9     852     832
##  7 2013-01-01 07:15:00  2013     1     1    7.15     713       2     911     850
##  8 2013-01-01 07:39:00  2013     1     1    7.39     745      -6     918     930
##  9 2013-01-01 07:49:00  2013     1     1    7.49     710      39     939     850
## 10 2013-01-01 08:24:00  2013     1     1    8.24     830      -6    1027    1025
## # ... with 2,002 more rows, 11 more variables: arr_delay <dbl>, carrier <chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>, and abbreviated
## #   variable names 1: dep_time, 2: sched_dep_time, 3: dep_delay, 4: arr_time,
## #   5: sched_arr_time
```

```
nycJan <- nyc %>%
  filter(month == 1) %>%
  filter(dest == "ORD" | dest == "DSM" | dest == "STL" | dest == "MCI" | dest == "MDW") %>%
  mutate(dep_time = dep_time/100) %>%
  unite("dep", c(year, month, day, dep_time), sep ="/", remove = FALSE) %>%
  mutate(dep = ymd_hm(dep, tz = "America/New_York", quiet = TRUE)) %>%
  filter(!is.na(dep)) %>%
  mutate(arr_time = arr_time/100) %>%
  unite("arr", c(year, month, day, arr_time), sep ="/", remove = FALSE) %>%
  mutate(arr = ymd_hm(arr, tz = "America/New_York", quiet = TRUE)) %>%
  filter(!is.na(arr)) %>%
  mutate(duration = dep - arr)
```

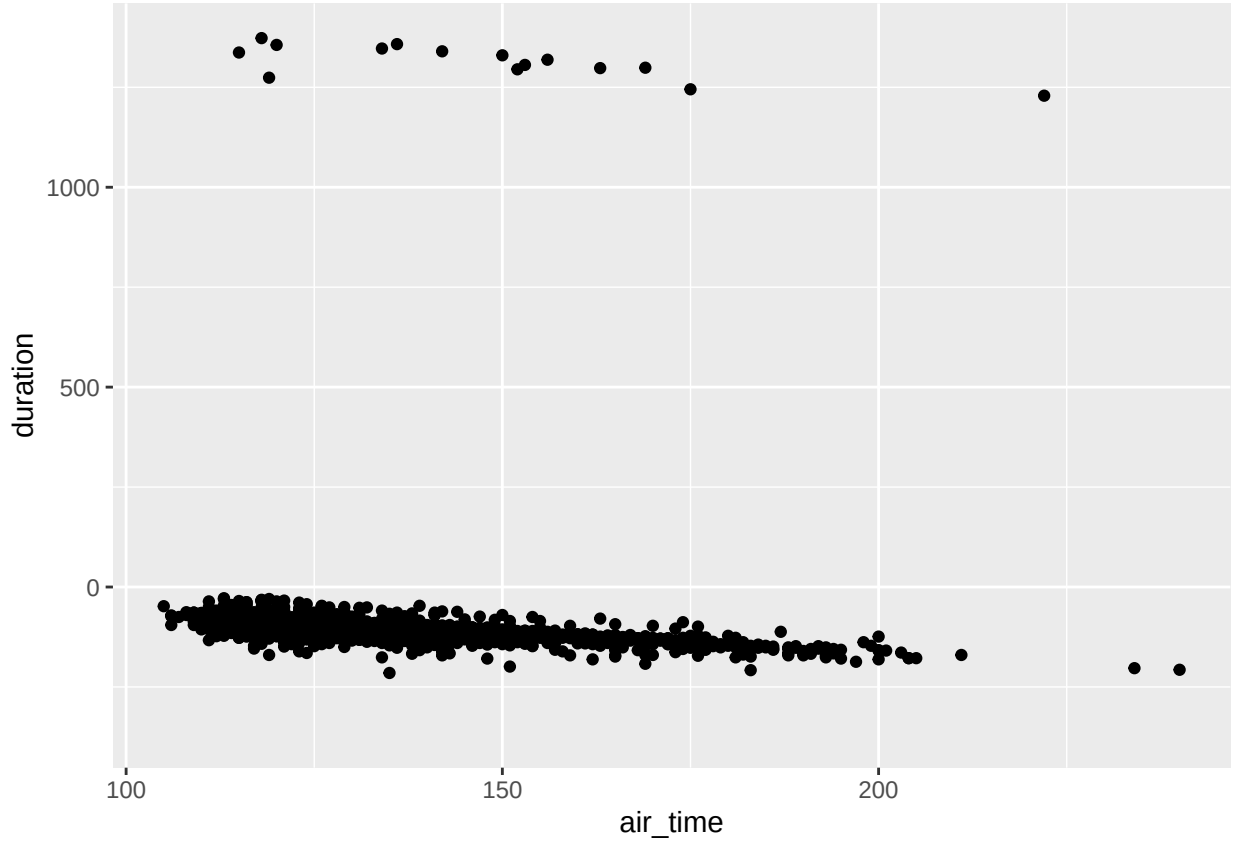
This is an example of how I learned to use ggplot in a more concise way. Seeing a lot of personal growth

```
Jan_plot <- ggplot(nycJan, aes(air_time, duration))

Jan_plot + geom_point()
```

```
## Don't know how to automatically pick scale for object of type <difftime>.
## Defaulting to continuous.

## Warning: Removed 8 rows containing missing values (`geom_point()`).
```



A quick demo of the stringr package using everyone's favorite, Ames Housing data!

```
install.packages("AmesHousing", repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/tz/sh20cj15711657_9_1d4v6m00000gn/T//RtmpEjRGTR/downloaded_packages
```

```
library(AmesHousing)

Ames <- make_ordinal_ames()
```

```
AmesNeigh <-  fct_unique(Ames$Neighborhood)

str_count(AmesNeigh, "[B|b]")
```

```
##  [1] 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 1 0 0 0 1 1 0 0 1 0 0 0 0
```

```
str_count(AmesNeigh, "_")
```

```
##  [1] 1 1 1 0 0 1 0 0 1 1 0 0 0 4 0 0 1 6 1 1 0 1 0 1 0 0 1 0 1
```

Demo of the purr package where I can create a boxplot with variables that have the word "Area" within it

```
Ames %>%
  select(ends_with("Area")) %>%
  boxplot(outline = FALSE)
```