Basic `ggplot2` Examples

Taylor Mackay

Contents

Introduction

This file contains examples of basic plots created using the `ggplot2` package in `R` and the corresponding code required to create each plot. All examples below require loading `ggplot2`– any other required packages are noted as needed in the included code.

NOTE: The specific style of the plots below is specified by using `theme_bcg` in addition to the other plot options. This calls the code below in order to specify the plot style, font type and size, and center plot titles.

```r
# Setting options for plot formatting, including font type + size, and title
# alignment, using `minimal` theme

theme_bcg <- theme_minimal(base_size = 9, base_family = "Palatino") +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"))
```

Data Used in Examples

Most of the datasets used in the first few examples come directly from the sample datasets included with `R`. Many of the later plots, however, use player-level basketball data from the 2015-2016 season from (https://www.basketball-reference.com). This data set can be downloaded from Github using the following code.

```r
download.file("github.com/mackaytc/plotting/blob/master/basketball.Rda?raw=true",
              "basketball.Rda")

load("basketball.Rda")

library(xtable)

print(xtable(head(nba.data[,1:10])), include.rownames = F)
```

Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA
1	Quincy Acy	PF	25	SAC	59	29	876	119	214
2	Jordan Adams	SG	21	MEM	2	0	15	2	6
3	Steven Adams	C	22	OKC	80	80	2014	261	426
4	Arron Afflalo	SG	30	NYK	71	57	2371	354	799
5	Alexis Ajinca	C	27	NOP	59	17	861	150	315
6	Cole Aldrich	C	27	LAC	60	5	800	134	225

Useful Resources

Useful websites with more information on `R` and `ggplot2` (click bulleted items for link to URL).

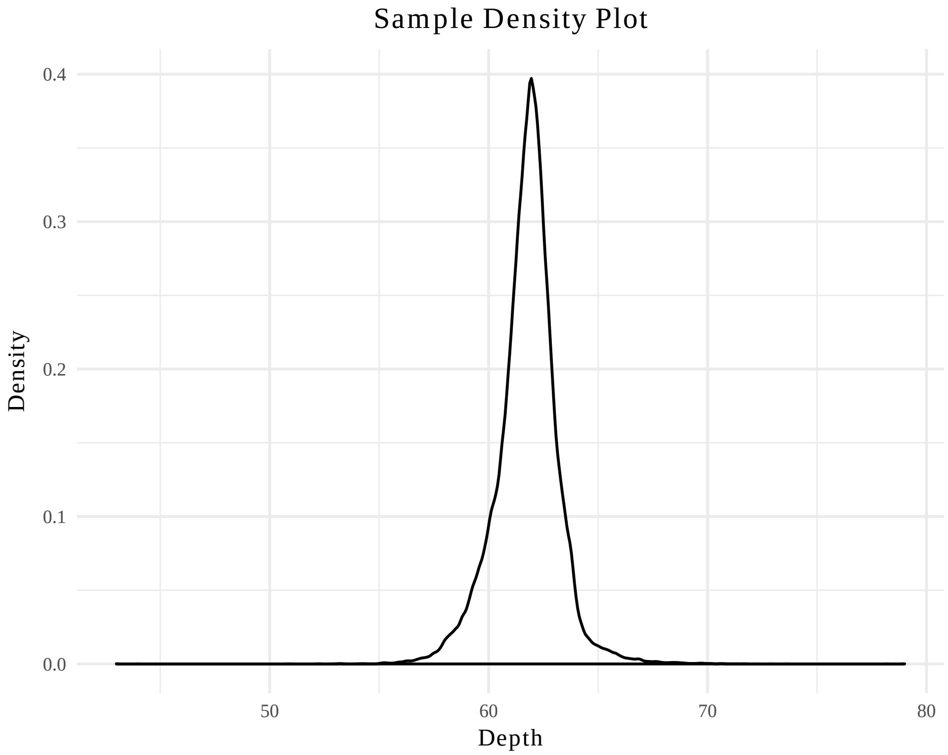
- RStudio `ggplot2` Cheatsheet
  - Two page PDF cheat sheet covering the basics of the `ggplot2` package
- Gallery of `ggplot2` Examples
  - 50 different examples of plots, covering a range of plot types and customizations to things like legends and annotations
- `R` Datasets Package
  - A list of the sample datasets available with `R` that are used in this document. Includes a detailed description of all variables in each dataset.

~

Univariate Plots

Density Plot

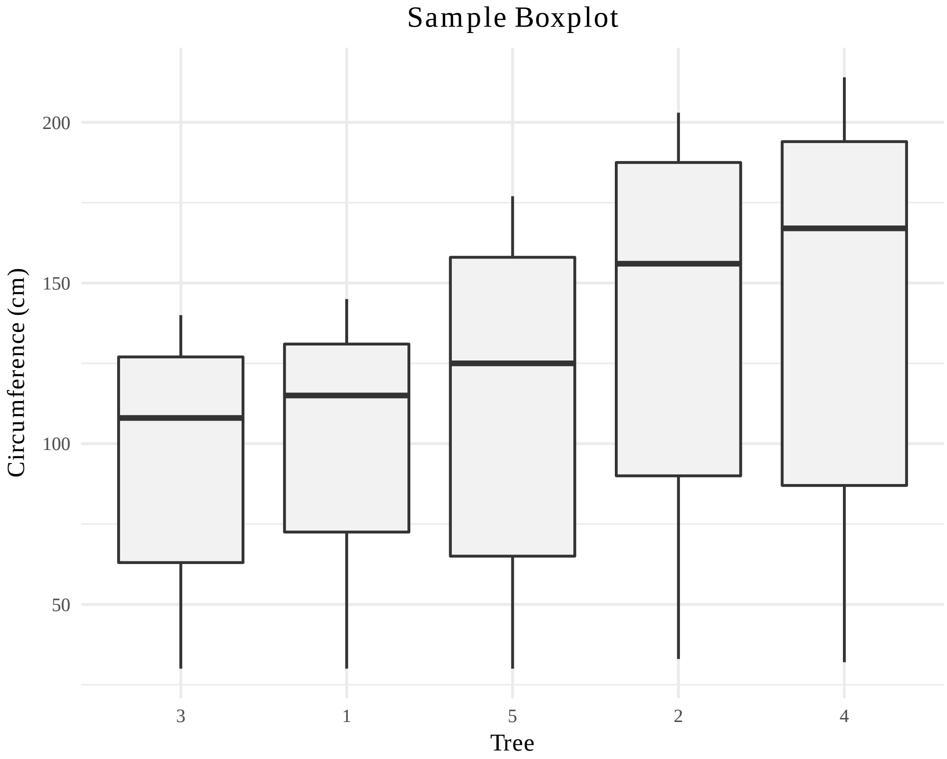
```
# Diamonds samples dataset has prices and other attributes of ~54,000 diamonds.

data(diamonds)

# Generating density plot

ggplot(data = diamonds, aes(x = depth)) + geom_density() +
  labs(title = "Sample Density Plot",
       y = "Density",
       x = "Depth") +
  theme_bcg
```



~

Boxplot by Factor Variable

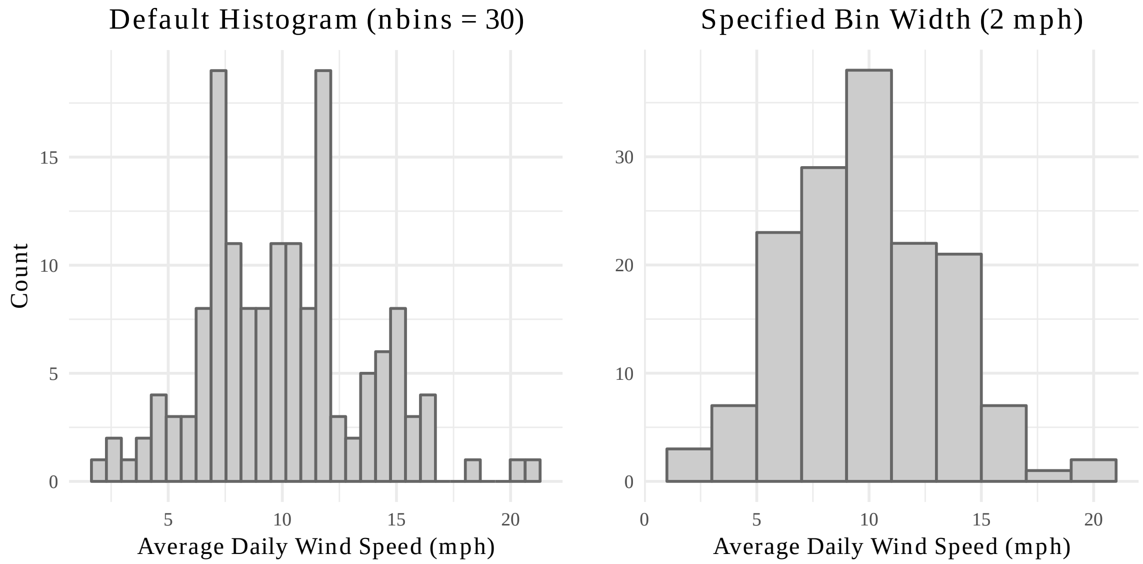
```
# Orange sample data set has 7 measurements of age and circumference for 5
# different oranges (total of 35 observations)

data(Orange)

# Basic boxplot grouped by `tree` factor variable

ggplot(Orange) +
  geom_boxplot(aes(x = Tree, y = circumference), fill = "grey95") +
  labs(title = "Sample Boxplot",
       y = "Circumference (cm)",
       x = "Tree") +
  theme_bcg
```



~

Histograms with Grid Arrange

```
# `gridExtra` allows you to print multiple plots together

library(gridExtra)

# Airquality sample dataset has measurements of temperature, windspeed, and
# daily air quality in New York from May to September, 1973.

data("airquality")

# Default Histogram

p.1 <- ggplot(airquality) +
  geom_histogram(aes(x = Wind), fill = "grey80", color = "grey40") +
  labs(title = "Default Histogram (nbins = 30)",
       y = "Count",
       x = "Average Daily Wind Speed (mph)") +
  theme_bcg

p.2 <- ggplot(airquality) +
  geom_histogram(aes(x = Wind), fill = "grey80", color = "grey40",
                 binwidth = 2) +
  labs(title = "Specified Bin Width (2 mph)",
       y = "",
       x = "Average Daily Wind Speed (mph)") +
  theme_bcg

grid.arrange(p.1, p.2, nrow = 1)
```

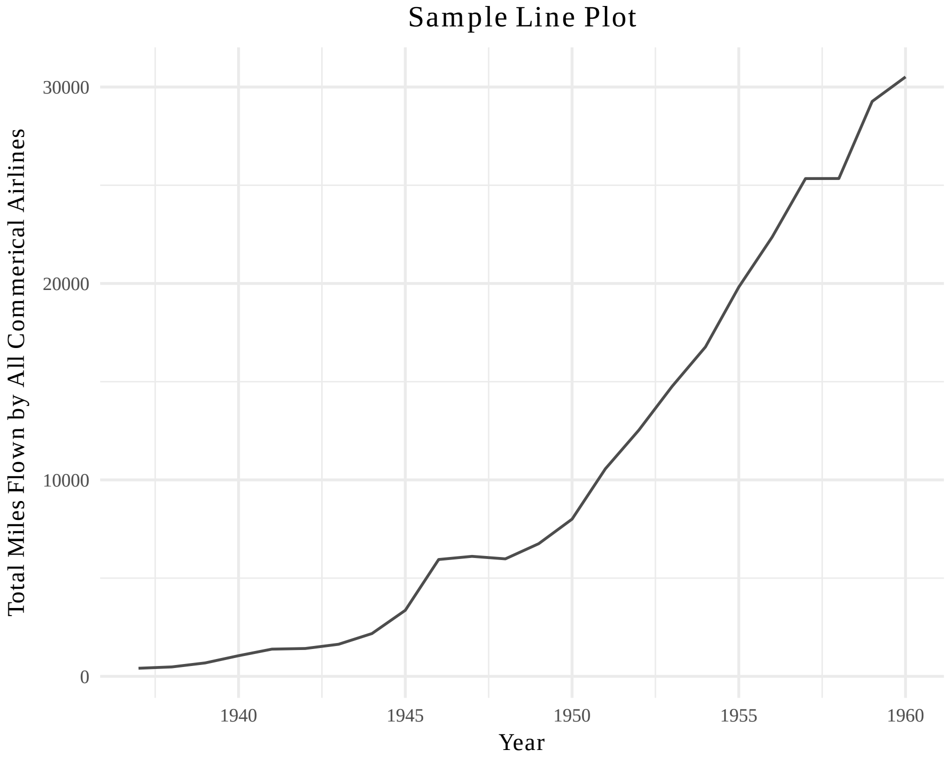


~

Two-Way Plots

Line Plot

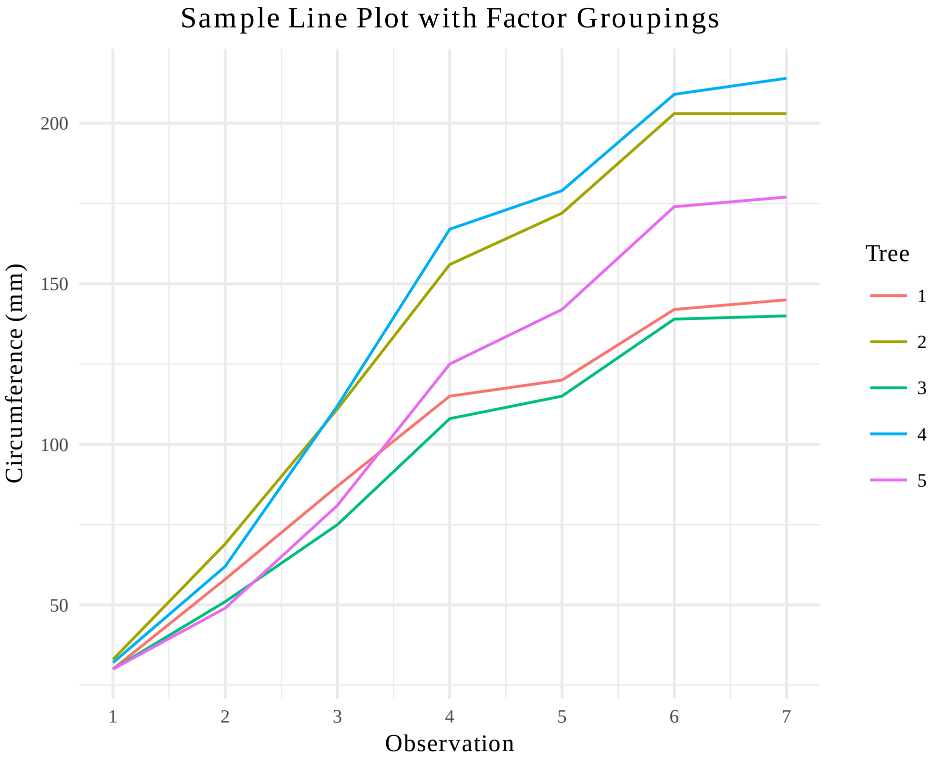
```
# Airmiles data set contains figures for the revenue passenger miles flown by
# commercial airlines in the United States for each year from 1937 to 1960

data(airmiles)

# Line Plot

# Constructing dataframe to pass to `ggplot`

df <- as.data.frame(cbind(1937:1960,
                    airmiles[1:24]))

# Basic line plot

ggplot(df) +
  geom_line(aes(x = V1, y = V2), size = 0.5, color = "grey30") +
  labs(title = "Sample Line Plot",
       y = "Total Miles Flown by All Commerical Airlines",
       x = "Year") +
  theme_bcg
```



~

Line Plot with Outcome Grouped by Factor Variable

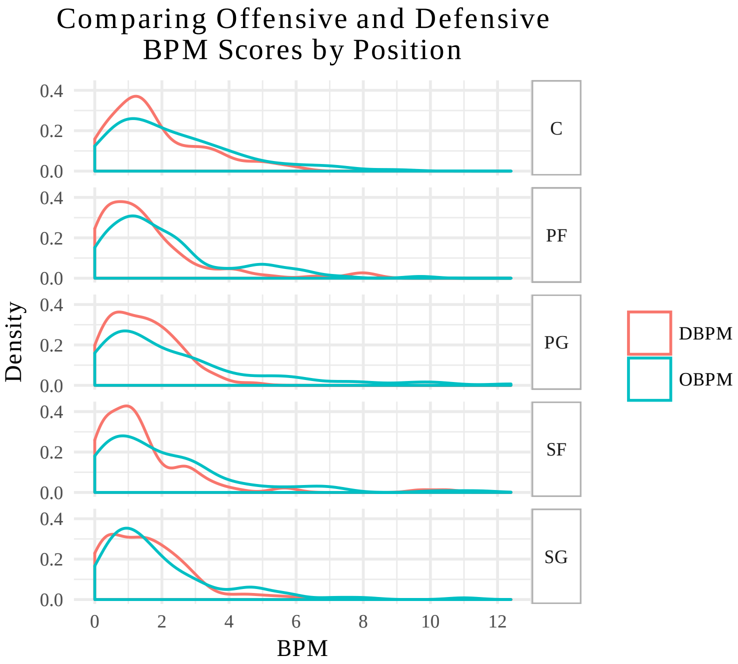
```
# Orange sample data set has 7 measurements of age and circumference for 5
# different oranges (total of 35 observations)

data(Orange)

# Start by creating a observation count by ID variable using `dplyr`. Note that
# data needs to be in *long* form.

library(dplyr)

df <- group_by(Orange, Tree) %>%
  mutate(count = row_number())

# Creating re-ordered `tree` factor variable

df$Tree <- factor(df$Tree, levels = c(1,2,3,4,5))

# Line Plot-- notice options for setting x-axis ticks + legend label

ggplot(df) + geom_line(aes(x = count, y = circumference, color = Tree)) +
  labs(title = "Sample Line Plot with Factor Groupings",
       y = "Circumference (mm)", x = "Observation",
       color = "Tree") +
  scale_x_continuous(breaks=seq(1, 7, 1)) +
  theme_bcg
```



~

Line Plots with Facets to Create Subplots

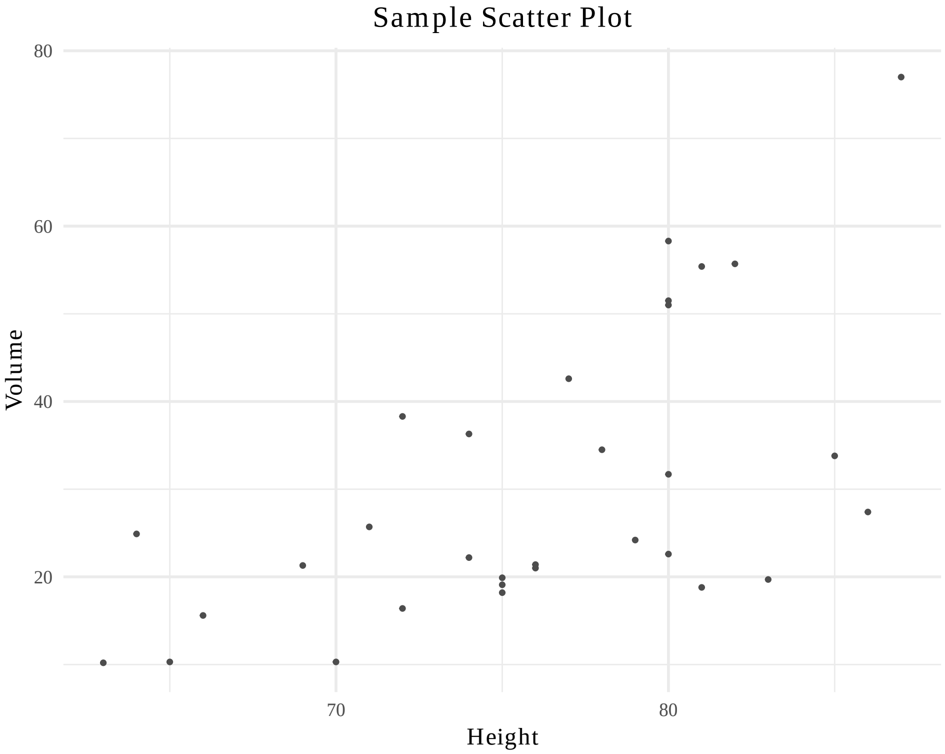
```
# Load sample dataset with 2016 player statistics for all players in NBA

load("basketball.Rda")

# We'll `tidyr` to reshape the data from `wide` to `long` format using the
# `gather` command and create a new dataset where each player in the data set
# has two rows-- one corresponding to their defensive BPM and one corresponding
# to their offensive BPM.

library(dplyr)
library(tidyr)

facet.data <- select(nba.data, Player, Pos, OBPM, DBPM) %>%
  gather(key = c(Player, Pos), value = BPM, OBPM:DBPM) %>%
  rename(stat = `c(Player, Pos)`) %>%
  arrange(Player)

# Facet Plot-- note the formatting options at the bottom to specify facet
# formatting

ggplot(facet.data) + facet_grid(Pos ~ .) +
  geom_density(aes(x = BPM, color = stat)) +
  scale_x_continuous(breaks = seq(0, 12, 2)) +
  scale_y_continuous(breaks = seq(0, 0.4, .2)) +
  labs(title = "Comparing Offensive and Defensive\nBPM Scores by Position",
       x = "BPM", y = "Density", color = "") +
  theme_bcg + theme(strip.text.y = element_text(angle = 0),
                    strip.background = element_rect(color = "grey70", size = 0.5))
```



~

Scatter Plot

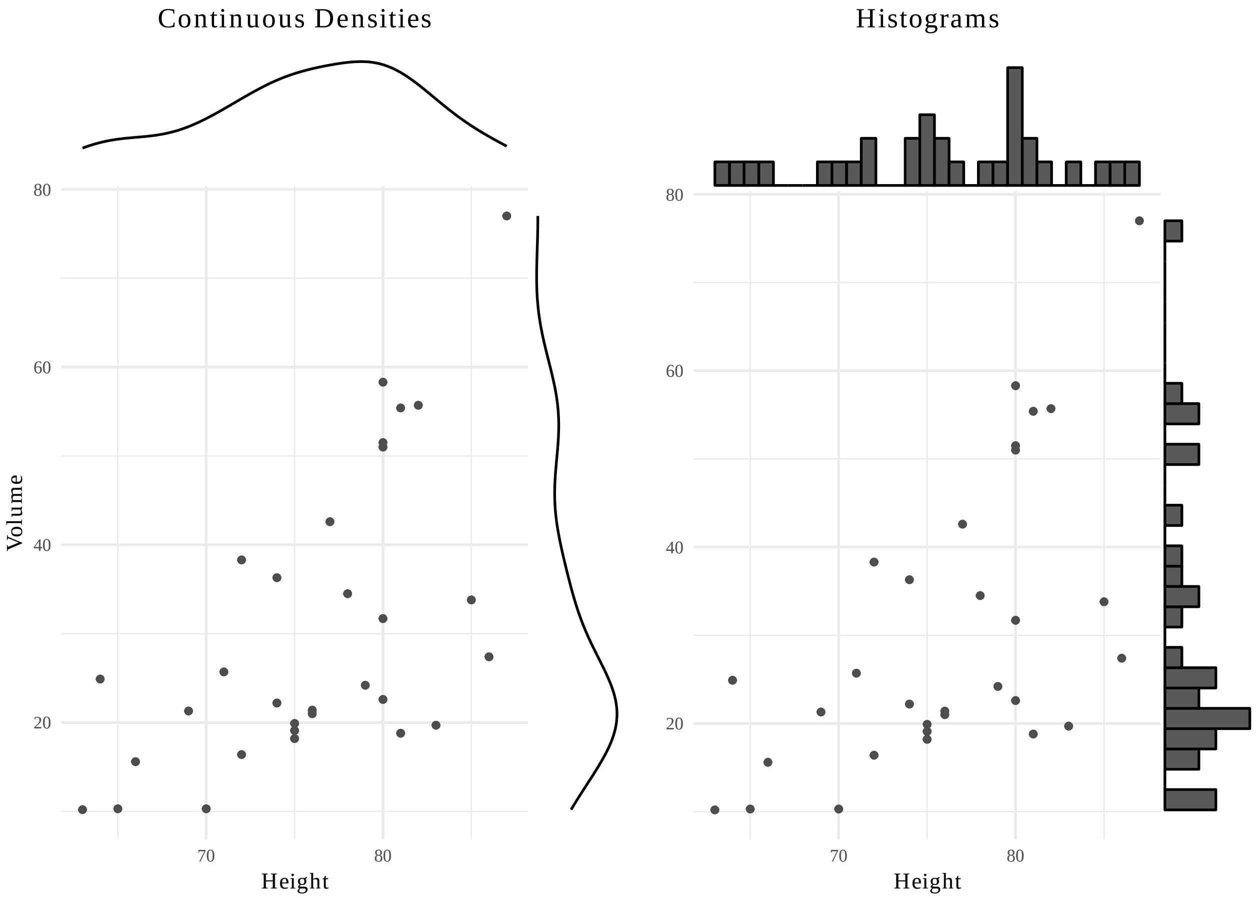
```
# Trees sample dataset has measurements of the height, weight, and length of
# 31 observations.

data(trees)

# Scatter Plot with size and color of points specified

ggplot(trees) +
  geom_point(aes(x = Height, y = Volume), size = 0.5, color = "grey30") +
  labs(title = "Sample Scatter Plot",
       y = "Volume",
       x = "Height") +
  theme_bcg
```



~

Scatter Plot with Marginal Densities

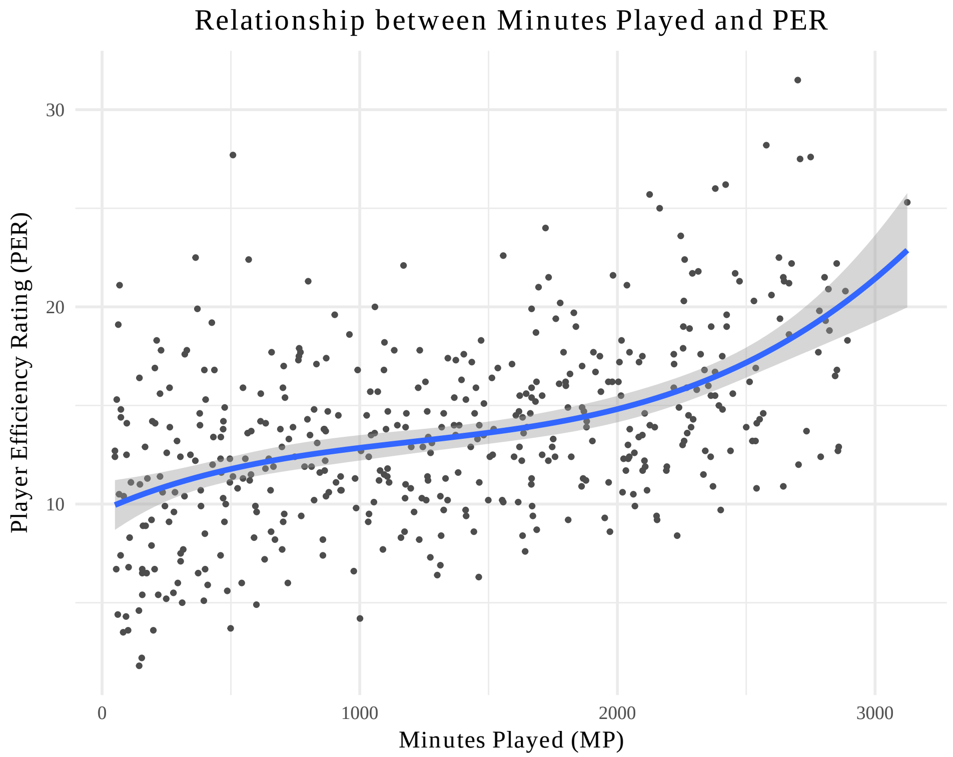
```r
# Trees sample dataset has measurements of the height, weight, and length of
# 31 observations.

data(trees)

# Scatter Plot with Marginal Densities

# To generate this graph we'll use the `ggExtra` package

suppressWarnings(library(ggExtra))

# Basic scatter plot

plot.1 <- ggplot(trees, aes(x = Height, y = Volume)) +
  geom_point(size = 1, color = "grey30") +
  labs(title = "Continuous Densities",
       y = "Volume",
       x = "Height") +
  theme_bcg

# Adding marginal densities

p.1 <- ggMarginal(plot.1, type = "density")

# Basic scatter plot

plot.2 <- ggplot(trees, aes(x = Height, y = Volume)) +
  geom_point(size = 1, color = "grey30") +
  labs(title = "Histograms",
       y = "",
       x = "Height") +
  theme_bcg

p.2 <- ggMarginal(plot.2, type = "histogram")

grid.arrange(p.1, p.2, nrow = 1)
```


Continuous Densities
Histograms
80
60
40
20
Volume
70
80
Height
80
60
40
20
70
80
Height

~

Scatter Plot with Fitted Line

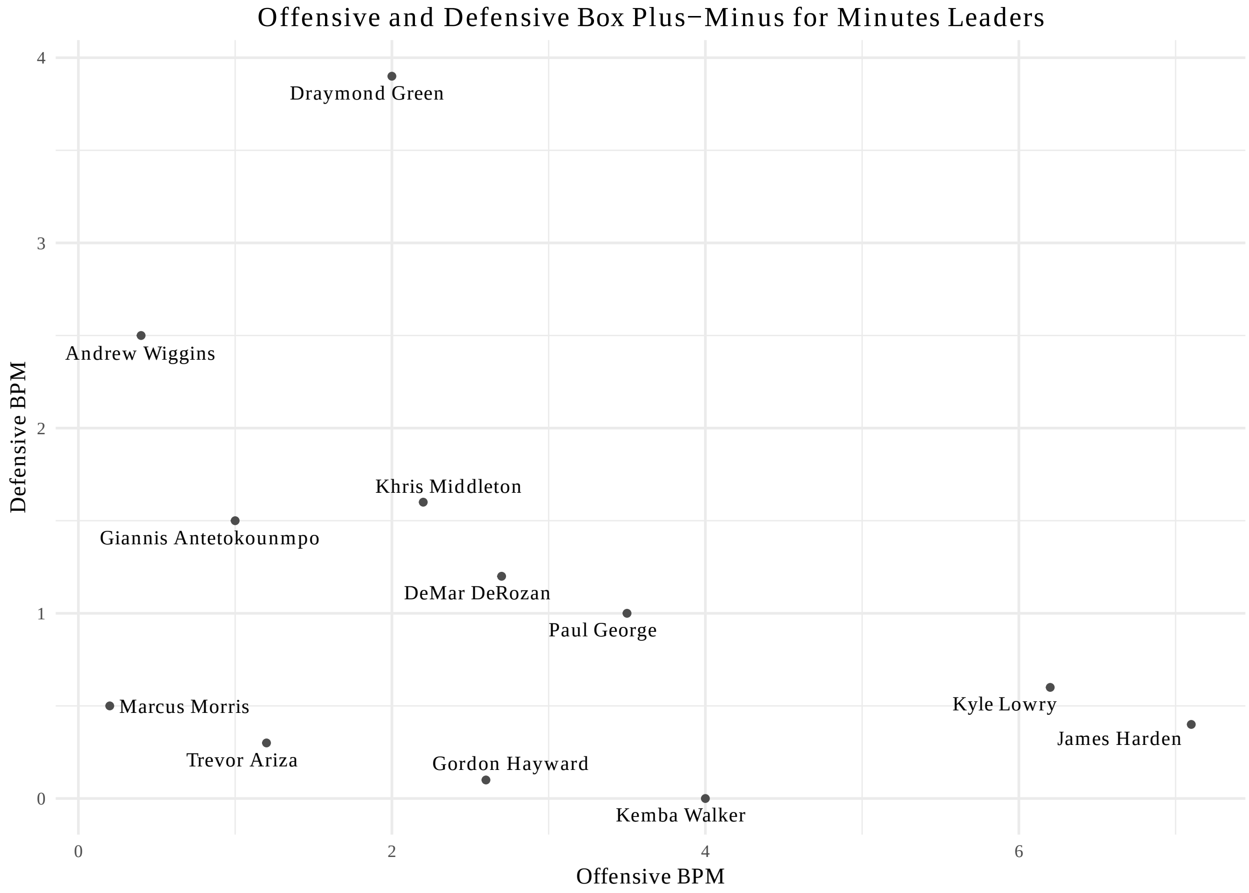
```r
# Load sample dataset with 2016 player statistics for all players in NBA

load("basketball.Rda")

# Scatter Plot with Minutes Played and Player Efficiency Rating (PER)

# Data is filtered to only players with at least one full game (48 minutes)
# worth of playing time during the season. `geom_smooth` options set to display
# a 3rd-degree polynomial fitted line with SE bands displayed

library(dplyr)

ggplot(filter(nba.data, MP > 48), aes(x = MP, y = PER)) +
  geom_point(size = 0.5, color = "grey30") +
  geom_smooth(method = "lm", formula = y ~ x + I(x^2) + I(x^3), se = TRUE) +
  labs(title = "Relationship between Minutes Played and PER",
       y = "Player Efficiency Rating (PER)",
       x = "Minutes Played (MP)") +
  theme_bcg
```



~

Scatter Plot with (Neatly) Labeled Points

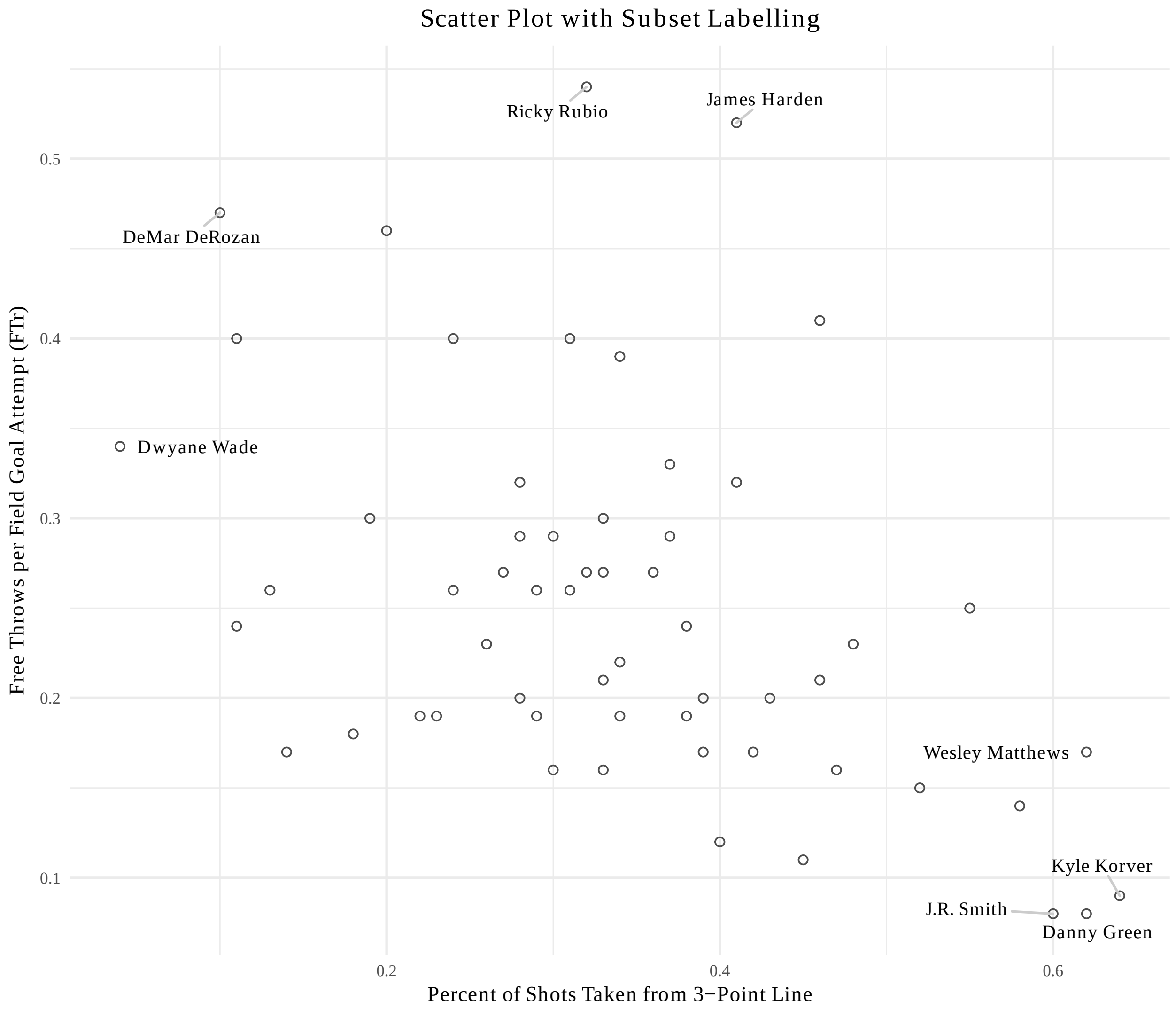
```
# Load sample dataset with 2016 player statistics for all players in NBA

load("basketball.Rda")

# Plot Relationship between Offensive and Defensive Box Plus-Minus (BPM)

# We can use the `ggrepel` package for neatly formatted point labelling

library(ggrepel)

# Use `dplyr` to select + sort the top 10 players by total minutes played

arrange(filter(nba.data, MP > 2000),-MP)[1:12,] %>%
  ggplot(aes(x = OBPM, y = DBPM)) +
  geom_point(size = 1, color = "grey30") +
  geom_text_repel(aes(label = Player), family = "Palatino", size = 2.8,
                  segment.color = NA) +
  labs(title = "Offensive and Defensive Box Plus-Minus for Minutes Leaders",
       y = "Defensive BPM", x = "Offensive BPM") + theme_bcg
```



~

Scatter Plot with Selectively Labeled Points

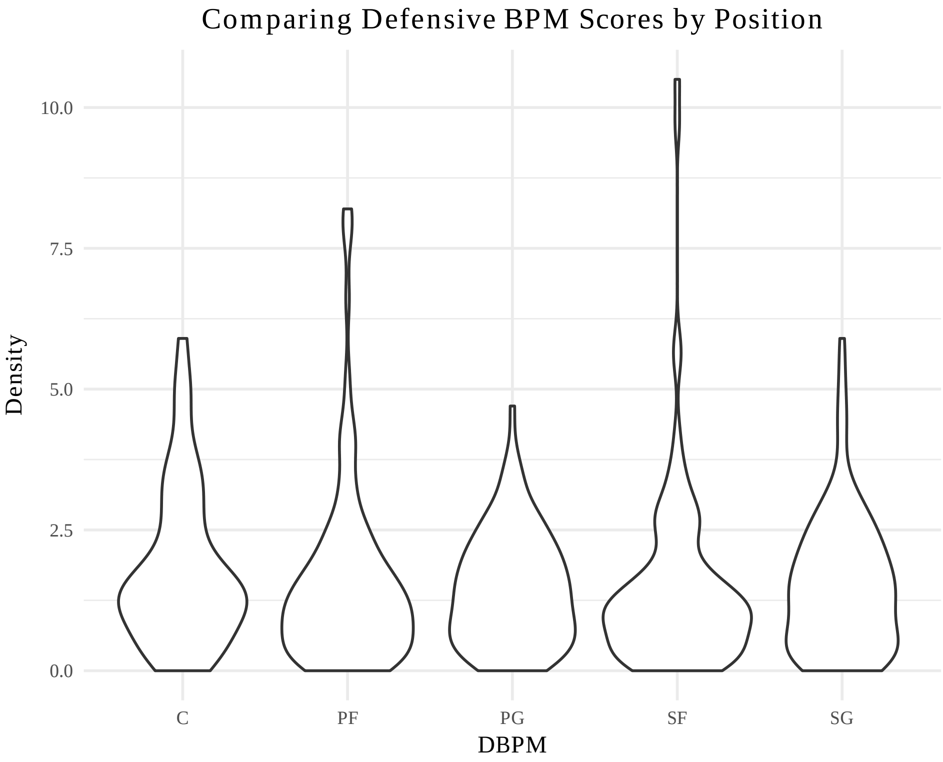
```r
# Load sample dataset with 2016 player statistics for all players in NBA

load("basketball.Rda")

# Comparing True Shooting Pct and Three Point Attempt Rate

# We'll use the `ggrepel` package to get neatly formatted labels

library(ggrepel)

# Subsetting data to just look at guards with over 2000 minutes of game time

plt.data <- filter(nba.data, (Pos == "PG" | Pos == "SG") & MP > 2000)

# We want to look at highest and lowest rated players by free throw attempt
# rate (FTr) and 3 pt attempt rate (X3PAr), defined as 5th and 95th percentiles

x.min <- quantile(plt.data$X3PAr, seq(0, 1, 0.05))[2]
x.max <- quantile(plt.data$X3PAr, seq(0, 1, 0.05))[20]
y.min <- quantile(plt.data$FTr, seq(0, 1, 0.05))[2]
y.max <- quantile(plt.data$FTr, seq(0, 1, 0.05))[20]

# Subset plotting dataset to just the observations that we want to have names
# included on the plot

lbls.df <- filter(plt.data, X3PAr < x.min | X3PAr > x.max |
                  FTr > y.max | FTr < y.min)

# Scatter Plot

ggplot(plt.data, aes(x = X3PAr, y = FTr)) +
  geom_point(shape = 1, color = "grey30") +
  geom_text_repel(data = lbls.df, aes(x = X3PAr, y = FTr, label = Player),
                  family = "Palatino", box.padding = 0.5, size = 2.8,
                  segment.color = "grey80") +
  labs(title = "Scatter Plot with Subset Labelling",
       x = "Percent of Shots Taken from 3-Point Line",
       y = "Free Throws per Field Goal Attempt (FTr)") +
  theme_bcg
```


Scatter Plot with Subset Labelling
Ricky Rubio
James Harden
DeMar DeRozan
Dwyane Wade
Wesley Matthews
Kyle Korver
J.R. Smith
Danny Green
0.5
0.4
0.3
0.2
0.1
0.2
0.4
0.6
Free Throws per Field Goal Attempt (FTr)
Percent of Shots Taken from 3−Point Line

~

Violin Plot by Factor Variable

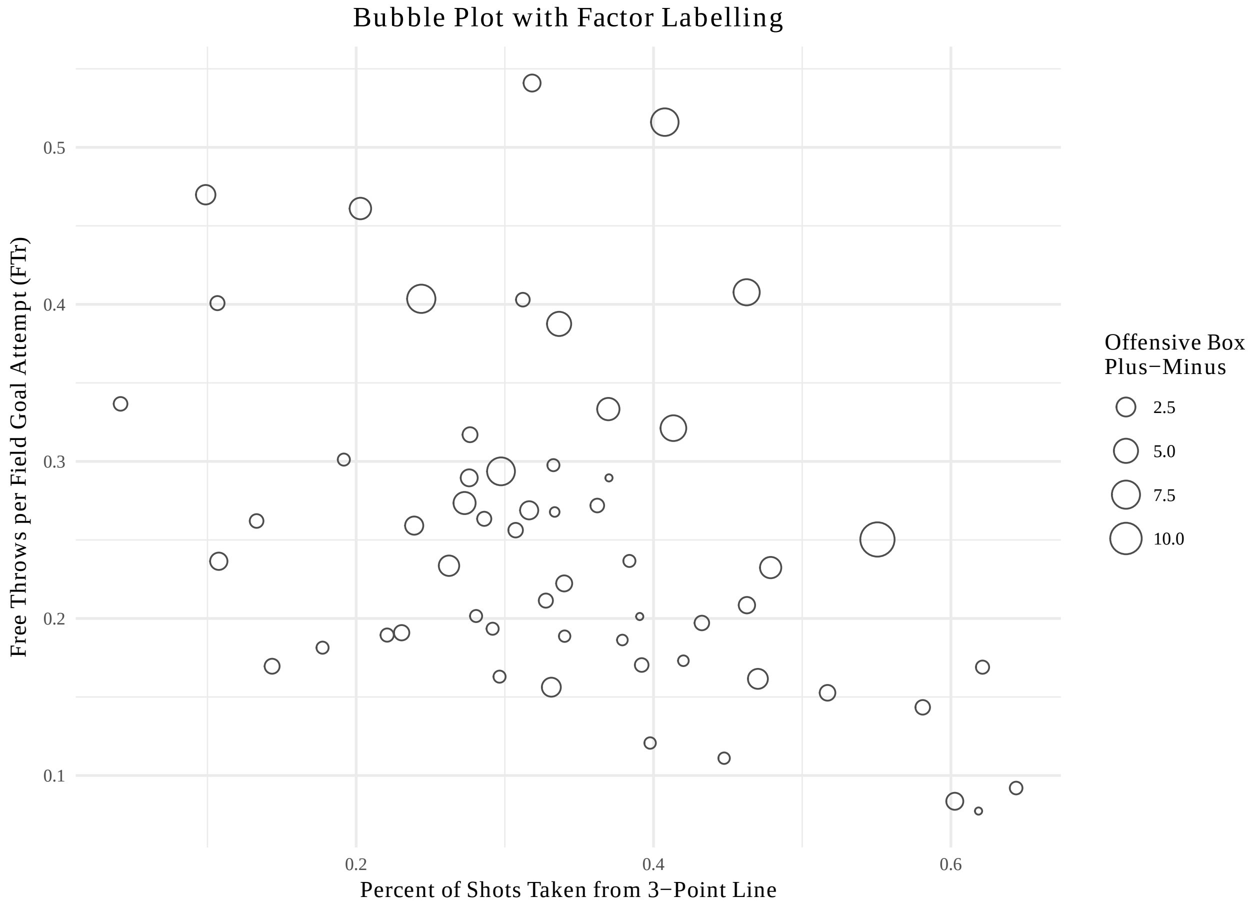
```
# Load sample dataset with 2016 player statistics for all players in NBA

load("basketball.Rda")

# Violin Plot by Factor Variable-- Splitting Players by Position

ggplot(nba.data) +
  geom_violin(aes(x = Pos, y = DBPM)) +
  labs(title = "Comparing Defensive BPM Scores by Position",
       x = "DBPM", y = "Density", color = "") +
  theme_bcg + theme(strip.text.y = element_text(angle = 0),
                    strip.background = element_rect(color = "grey70", size = 0.5))
```



~

Bubble Plot

```
# Load sample dataset with 2016 player statistics for all players in NBA

load("basketball.Rda")

# Comparing True Shooting Pct and Foul Rate

# Subsetting data to just look at guards with over 2000 minutes of game time

plt.data <- filter(nba.data, (Pos == "PG" | Pos == "SG") & MP > 2000)

# `geom_jitter()` is a variation of `geom_point()` that prevents points from
# "piling up" when they're close to one another. We can set bubble (point)
# size using the `aes(size = ...)` call below.

ggplot(plt.data, aes(x = X3PAr, y = FTr)) +
  geom_jitter(aes(size = OBPM), color = "grey30", shape = 1) +
  labs(title = "Bubble Plot with Factor Labelling",
       x = "Percent of Shots Taken from 3-Point Line",
       y = "Free Throws per Field Goal Attempt (FTr)",
       size = "Offensive Box \nPlus-Minus") +
  theme_bcg
```

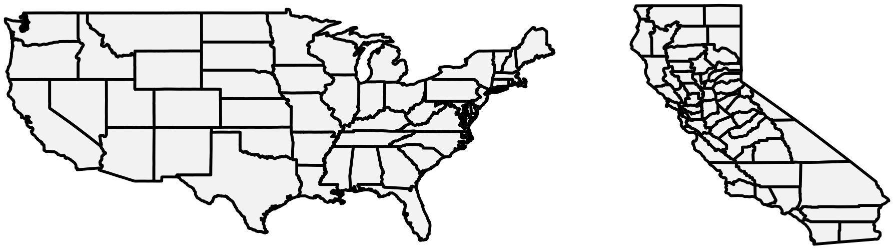


~

Map Plots

Basic Map of the United States and Counties of California

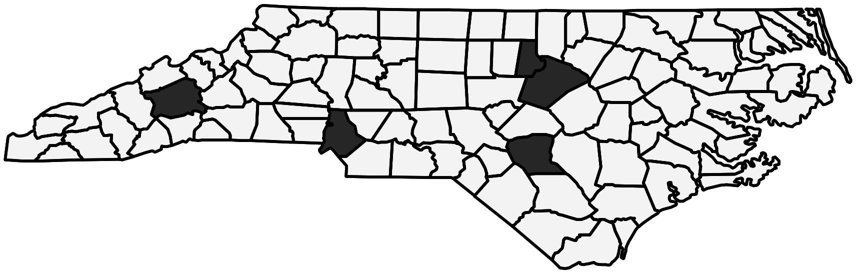
```r
# Packages for map plotting-- `ggthemes` loads the `theme_minimal()` style
# used below, while `gridExtra` lets you control layouts of displayed plots.
# The `maps` package contains geographic data.

library(maps)
library(ggthemes)
library(gridExtra)

# Loading geographic data for states + counties in the US. `map_data()` takes
# the series of points-based data provided by the `maps` package and converts
# into a df that is readable via ggplot2

counties <- map_data("county")
states   <- map_data("state")

# Theme below removes all unneccessary axes and tick marks from plots

ditch_the_axes <- theme(axis.text = element_blank(), axis.line = element_blank(),
                        axis.ticks = element_blank(), panel.border = element_blank(),
                        panel.grid = element_blank(), axis.title = element_blank())

# Basic state-level plot of the United States. Note the use of `coord_fixed()`
# to prevent distortion along the x / y axes-- setting alternative parameter
# options here "stretches" the along the y-axis

p.1 <- ggplot(states, mapping = aes(x = long, y = lat, group = group)) +
              coord_fixed(1) + geom_polygon(color = "black", fill = "gray95") +
              theme_minimal() + ditch_the_axes

# Plot of the counties within California

p.2 <- ggplot(subset(counties, region == "california"),
              mapping = aes(x = long, y = lat, group = group)) +
              coord_fixed(1) + geom_polygon(color = "black", fill = "gray95") +
              theme_minimal() + ditch_the_axes

# `grid.arrange()` to display them next to one another

grid.arrange(p.1, p.2, widths = 2:1)
```



~

Plotting Discrete Variables on a Map

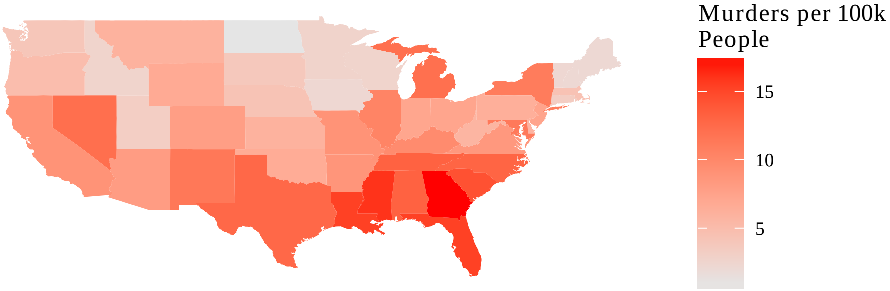
```
library(maps)
library(ggthemes)
library(gridExtra)

# Subset north carolina data from `counties` data in previous example

nc.counties <- map_data("county") %>%
  subset(region == "north carolina")

# We want to a plot that highlights the counties in North Carolina that have a
# particular type of law in effect

# We can start with a basic map of the counties in NC

nc.map <- ggplot(nc.counties, mapping = aes(x = long, y = lat, group = group)) +
  coord_fixed(1) + geom_polygon(color = "black", fill = "white") +
  theme_minimal() + ditch_the_axes

# Now we can create a variable set equal to 1 if a particular county has the
# type of law we're interested in

btb.counties <- c("buncombe", "cumberland", "durham", "mecklenburg", "wake")

nc.counties$btb.law <- 0

# Set btb.law dummy variable equal to 1 if the county name is in the list
# of `btb.counties`

nc.counties[nc.counties$subregion %in% btb.counties, ]$btb.law <- 1

# Creating map with counties that have law in effect shaded in

nc.map +
  geom_polygon(data = nc.counties, aes(fill = as.factor(btb.law)),
               color = alpha("black", 0.2)) +
  scale_colour_discrete() +
  scale_fill_manual(values = alpha(c("gray95", "black"), .85)) +
  geom_polygon(color = "black", fill = NA) +
  theme_bcg + ditch_the_axes + theme(legend.position="none")
```



~

Plotting Continuous Variables on a Map

```
# Loading geographic data for states + counties in the US. `map_data()` takes
# the series of points-based data provided by the `maps` package and converts
# into a df that is readable via ggplot2

states.data <- map_data("state")

# Crime data comes from `USArrests` sample data set

data("USArrests")

USArrests$region <- tolower(rownames(USArrests))

# Use `dplyr` to merge data sets using region (state) as ID variable

map.df <- inner_join(states.data, USArrests, by = "region")

# Theme below removes all unneccessary axes and tick marks from plots

ditch_the_axes <- theme(axis.text = element_blank(), axis.line = element_blank(),
                        axis.ticks = element_blank(), panel.border = element_blank(),
                        panel.grid = element_blank(), axis.title = element_blank())

# Plotting State Murder Rate Data


# Mapping state level murder counts. Check out the URL below for ggplot colors
#   http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/

ggplot(map.df, aes(x = long, y = lat, group = group)) +
  geom_polygon(color = "white", fill = "white") +
  geom_polygon(data = map.df, aes(fill = Murder)) +
  scale_fill_gradient(low = "grey90", high = "red") +
  coord_fixed(1) + labs(fill = "Murders per 100k\nPeople") +
  theme_minimal(base_size = 9, base_family = "Palatino") +
  ditch_the_axes
```

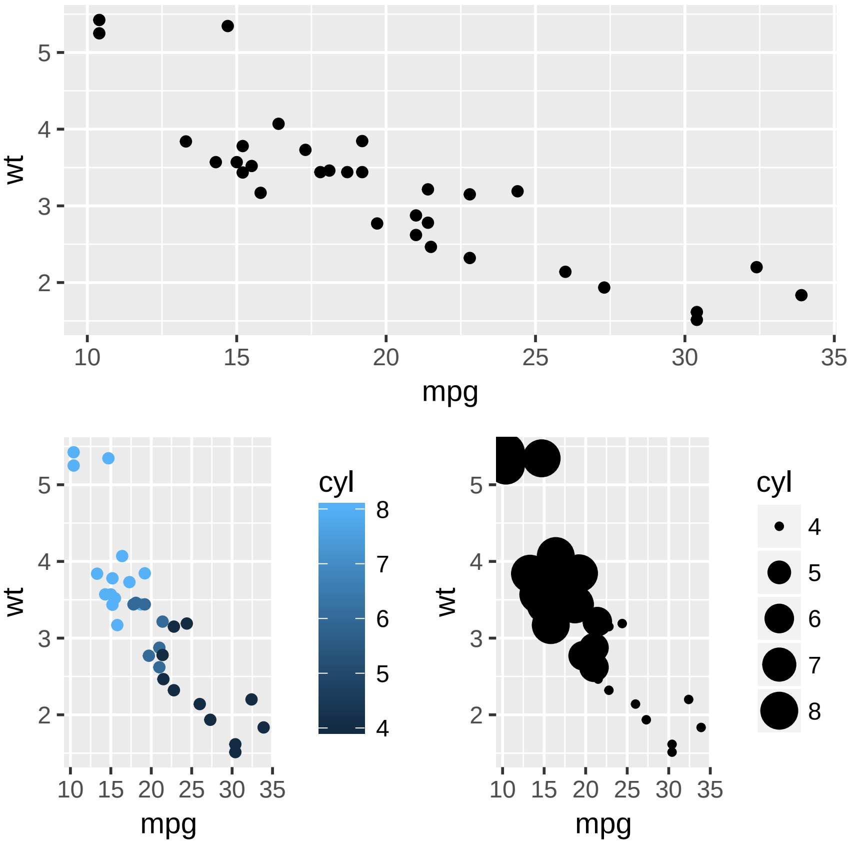


~

Using the qplot Function

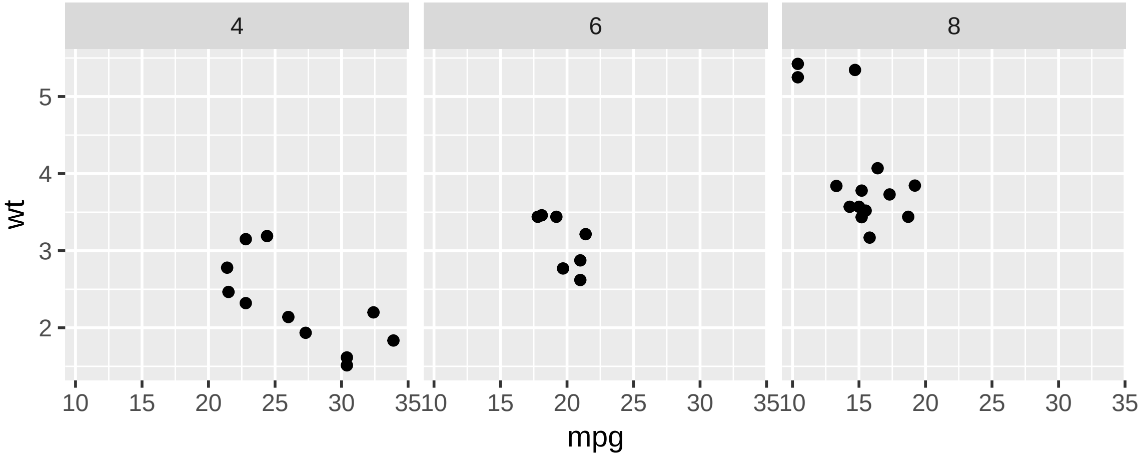
Introductory Examples

```
library(gridExtra)

# Sample `cars` dataset

data("mtcars")

# Basic `qplot` examples

# Scatter plot

p.1 <- qplot(mpg, wt, data = mtcars)

# Scatter plot with factor variable categorization

p.2 <- qplot(mpg, wt, data = mtcars, color = cyl)

# Chooding `size` instead of `color`

p.3 <- qplot(mpg, wt, data = mtcars, size = cyl)

grid.arrange(p.1, p.2, p.3, layout_matrix = rbind(c(1,1),c(2,3)),
             nrow = 2, ncol = 2)
```

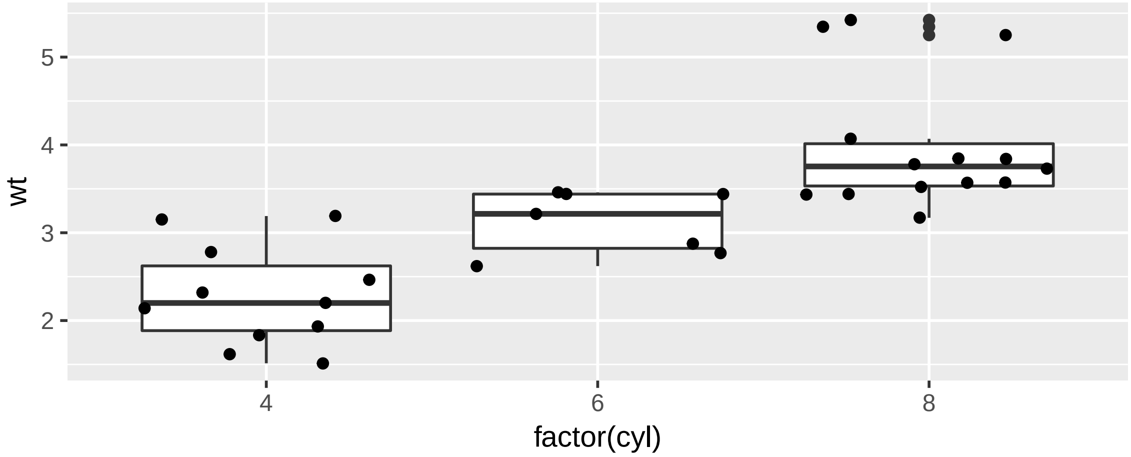


Additional Examples

```r
# Sample `cars` dataset

data("mtcars")

# Some additional plots

# Facet plot by factor variable

qplot(mpg, wt, data = mtcars, facets = .~cyl)
```



```r
# Specifying multiple `geom` options

qplot(factor(cyl), wt, data = mtcars, geom = c("boxplot", "jitter"))
```


wt
factor(cyl)