In [this previous post](https://statisticaloddsandends.wordpress.com/2018/12/11/scraping-nba-game-data-from-basketball-reference-com/), It’s showed how one can scrape top-level NBA game data from [BasketballReference.com](https://www.basketball-reference.com/). In [the post after that](https://statisticaloddsandends.wordpress.com/2018/12/13/recreating-the-nba-lead-tracker-graphic/), I demonstrated how to scrape play-by-play data for one game. After writing those posts, I thought to myself: why not do both? And that is what : **scrape all the box scores for the 2017-18 NBA season and save them to an RDS object.**

The code for web scraping can be found [here](https://github.com/kjytay/misc/blob/master/blog/nba_box_score.R). I don’t think it’s particularly insightful to go through the details line-by-line… What I’ll do instead is explain what is in the saved RDS object which will allow you to make use of the data, and I’ll end off with some visualizations to demonstrate what one can do with this data, specifically:

* ***Do Kevin Durant and Steph Curry get in each other’s way?***
* ***Are the Golden State Warriors really better in the 3rd quarter?***

(Code that I wrote for these questions can be found [here](https://github.com/kjytay/misc/blob/master/blog/2018-12-18_GSW.R).) Let’s load the packages and data (data available as links from [this page](https://github.com/kjytay/misc/tree/master/data)):

library(tidyverse)

game\_df <- readRDS("NBA-2018\_game\_data.rds")

master\_list <- readRDS("NBA-2018\_box\_score.rds")

First, let’s take a quick look at game\_df which contains the top-level game data:

head(game\_df)

#> game\_id date\_game game\_start\_time visitor\_team\_name visitor\_pts home\_team\_name home\_pts

#> 1 201710170CLE 2017-10-17 8:01p Boston Celtics 99 Cleveland Cavaliers 102

#> 2 201710170GSW 2017-10-17 10:30p Houston Rockets 122 Golden State Warriors 121

#> 3 201710180BOS 2017-10-18 7:30p Milwaukee Bucks 108 Boston Celtics 100

#> 4 201710180DAL 2017-10-18 8:30p Atlanta Hawks 117 Dallas Mavericks 111

#> 5 201710180DET 2017-10-18 7:00p Charlotte Hornets 90 Detroit Pistons 102

#> 6 201710180IND 2017-10-18 7:00p Brooklyn Nets 131 Indiana Pacers 140

#> overtimes attendance game\_remarks game\_type

#> 1 20562 Regular

#> 2 19596 Regular

#> 3 18624 Regular

#> 4 19709 Regular

#> 5 20491 Regular

#> 6 15008 Regular

The first column, game\_id, is BasketBallReference.com’s way of uniquely identifying a game. For example, if we wanted to see the results of the game between the Celtics and the Cavs on 17 Oct 2017, we would look up the game\_id (201710170CLE) and go to the URL <https://www.basketball-reference.com/boxscores/201710170CLE.html>. The game\_id is how I was able to scrape the box scores for all the games.

Next, let’s look at the master\_list object which contains all the box score information:

typeof(master\_list)

#> [1] "list"

length(master\_list)

#> [1] 1312

head(names(master\_list))

#> [1] "201710170CLE" "201710170GSW" "201710180BOS" "201710180DAL" "201710180DET" "201710180IND"

It is a list of length 1312, which corresponds to the total number of games (Regular Season and Playoffs) in the 2017-18 season. The keys of the list are the game\_ids, and (as we will see shortly) each element of the list is itself a list of length 5. Let’s take a closer look at the first element of the list:

str(master\_list[[1]])

#> List of 5

#> $ visitor\_basic\_boxscore:'data.frame': 13 obs. of 22 variables:

#> ..$ Player: chr [1:13] "Jaylen Brown" "Kyrie Irving" "Jayson Tatum" "Al Horford" ...

#> ..$ MP : chr [1:13] "39:36" "39:21" "36:32" "32:07" ...

#> ..$ FG : num [1:13] 11 8 5 2 1 5 2 2 0 0 ...

#> ..$ FGA : num [1:13] 23 17 12 7 2 16 6 2 2 1 ...

#> ..$ FG% : num [1:13] 0.478 0.471 0.417 0.286 0.5 0.313 0.333 1 0 0 ...

#> ..$ 3P : num [1:13] 2 4 1 0 0 0 1 0 0 0 ...

#> ..$ 3PA : num [1:13] 9 9 2 2 1 4 3 0 1 1 ...

#> ..$ 3P% : num [1:13] 0.222 0.444 0.5 0 0 0 0.333 NA 0 0 ...

#> ..$ FT : num [1:13] 1 2 3 5 0 2 4 2 0 0 ...

#> ..$ FTA : num [1:13] 2 2 3 7 0 3 4 4 0 0 ...

#> ..$ FT% : num [1:13] 0.5 1 1 0.714 NA 0.667 1 0.5 NA NA ...

#> ..$ ORB : num [1:13] 1 2 4 0 0 0 0 2 0 0 ...

#> ..$ DRB : num [1:13] 5 2 6 7 1 9 3 3 0 1 ...

#> ..$ TRB : num [1:13] 6 4 10 7 1 9 3 5 0 1 ...

#> ..$ AST : num [1:13] 0 10 3 5 0 3 2 1 0 0 ...

#> ..$ STL : num [1:13] 2 3 0 0 0 2 4 0 0 0 ...

#> ..$ BLK : num [1:13] 0 0 0 1 0 2 0 1 0 0 ...

#> ..$ TOV : num [1:13] 3 2 1 0 0 2 0 2 0 0 ...

#> ..$ PF : num [1:13] 5 4 4 2 1 2 0 5 1 0 ...

#> ..$ PTS : num [1:13] 25 22 14 9 2 12 9 6 0 0 ...

#> ..$ +/- : num [1:13] -5 -1 6 8 3 -8 4 -14 -10 2 ...

#> ..$ Role : chr [1:13] "Starter" "Starter" "Starter" "Starter" ...

#> $ visitor\_adv\_boxscore :'data.frame': 13 obs. of 17 variables:

#> ..$ Player: chr [1:13] "Jaylen Brown" "Kyrie Irving" "Jayson Tatum" "Al Horford" ...

#> ..$ MP : chr [1:13] "39:36" "39:21" "36:32" "32:07" ...

#> ..$ TS% : num [1:13] 0.523 0.615 0.526 0.446 0.5 0.346 0.58 0.798 0 0 ...

#> ..$ eFG% : num [1:13] 0.522 0.588 0.458 0.286 0.5 0.313 0.417 1 0 0 ...

#> ..$ 3PAr : num [1:13] 0.391 0.529 0.167 0.286 0.5 0.25 0.5 0 0.5 1 ...

#> ..$ FTr : num [1:13] 0.087 0.118 0.25 1 0 0.188 0.667 2 0 0 ...

#> ..$ ORB% : num [1:13] 2.4 4.9 10.5 0 0 0 0 10.1 0 0 ...

#> ..$ DRB% : num [1:13] 13.2 5.3 17.1 22.7 19.9 26.8 16 16.4 0 21.7 ...

#> ..$ TRB% : num [1:13] 7.6 5.1 13.7 10.9 9.5 12.8 7.7 13.1 0 10.4 ...

#> ..$ AST% : num [1:13] 0 46.5 13.4 22.6 0 14.1 15.8 8.1 0 0 ...

#> ..$ STL% : num [1:13] 2.4 3.7 0 0 0 2.8 9.9 0 0 0 ...

#> ..$ BLK% : num [1:13] 0 0 0 2.5 0 4.5 0 4.1 0 0 ...

#> ..$ TOV% : num [1:13] 11.2 10.1 7 0 0 10.4 0 34.7 0 0 ...

#> ..$ USG% : num [1:13] 29.9 22.2 17.3 13.8 16.8 24.3 17.5 13.3 10.2 9.1 ...

#> ..$ ORtg : num [1:13] 89 124 115 111 95 70 129 110 0 0 ...

#> ..$ DRtg : num [1:13] 104 103 108 105 107 96 87 105 112 107 ...

#> ..$ Role : chr [1:13] "Starter" "Starter" "Starter" "Starter" ...

#> $ home\_basic\_boxscore :'data.frame': 14 obs. of 22 variables:

#> ..$ Player: chr [1:14] "LeBron James" "Jae Crowder" "Derrick Rose" "Dwyane Wade" ...

#> ..$ MP : chr [1:14] "41:12" "34:44" "31:15" "28:30" ...

#> ..$ FG : num [1:14] 12 3 5 3 4 4 2 3 2 0 ...

#> ..$ FGA : num [1:14] 19 10 14 10 9 7 3 8 3 0 ...

#> ..$ FG% : num [1:14] 0.632 0.3 0.357 0.3 0.444 0.571 0.667 0.375 0.667 NA ...

#> ..$ 3P : num [1:14] 1 1 1 0 1 1 0 0 0 0 ...

#> ..$ 3PA : num [1:14] 5 5 3 1 4 3 0 1 0 0 ...

#> ..$ 3P% : num [1:14] 0.2 0.2 0.333 0 0.25 0.333 NA 0 NA NA ...

#> ..$ FT : num [1:14] 4 4 3 2 6 1 1 0 0 0 ...

#> ..$ FTA : num [1:14] 4 4 4 2 7 1 3 0 0 0 ...

#> ..$ FT% : num [1:14] 1 1 0.75 1 0.857 1 0.333 NA NA NA ...

#> ..$ ORB : num [1:14] 1 1 1 1 3 0 1 0 1 0 ...

#> ..$ DRB : num [1:14] 15 4 3 1 8 4 5 0 1 0 ...

#> ..$ TRB : num [1:14] 16 5 4 2 11 4 6 0 2 0 ...

#> ..$ AST : num [1:14] 9 2 2 3 0 1 2 0 0 0 ...

#> ..$ STL : num [1:14] 0 2 0 0 0 0 0 0 1 0 ...

#> ..$ BLK : num [1:14] 2 0 0 2 0 0 0 0 0 0 ...

#> ..$ TOV : num [1:14] 4 1 2 4 2 0 2 1 1 0 ...

#> ..$ PF : num [1:14] 3 2 2 1 2 4 3 3 3 2 ...

#> ..$ PTS : num [1:14] 29 11 14 8 15 10 5 6 4 0 ...

#> ..$ +/- : num [1:14] 2 7 -7 0 1 7 2 -2 6 -1 ...

#> ..$ Role : chr [1:14] "Starter" "Starter" "Starter" "Starter" ...

#> $ home\_adv\_boxscore :'data.frame': 14 obs. of 17 variables:

#> ..$ Player: chr [1:14] "LeBron James" "Jae Crowder" "Derrick Rose" "Dwyane Wade" ...

#> ..$ MP : chr [1:14] "41:12" "34:44" "31:15" "28:30" ...

#> ..$ TS% : num [1:14] 0.698 0.468 0.444 0.368 0.621 0.672 0.579 0.375 0.667 NA ...

#> ..$ eFG% : num [1:14] 0.658 0.35 0.393 0.3 0.5 0.643 0.667 0.375 0.667 NA ...

#> ..$ 3PAr : num [1:14] 0.263 0.5 0.214 0.1 0.444 0.429 0 0.125 0 NA ...

#> ..$ FTr : num [1:14] 0.211 0.4 0.286 0.2 0.778 0.143 1 0 0 NA ...

#> ..$ ORB% : num [1:14] 2.5 3 3.3 3.7 11 0 5.3 0 8.1 0 ...

#> ..$ DRB% : num [1:14] 35 11.1 9.2 3.4 27 17.5 24.5 0 7.5 0 ...

#> ..$ TRB% : num [1:14] 19.4 7.2 6.4 3.5 19.4 9.1 15.3 0 7.8 0 ...

#> ..$ AST% : num [1:14] 43.7 8.2 10.1 15.3 0 7.5 14.8 0 0 0 ...

#> ..$ STL% : num [1:14] 0 2.8 0 0 0 0 0 0 3.8 0 ...

#> ..$ BLK% : num [1:14] 4.2 0 0 6 0 0 0 0 0 0 ...

#> ..$ TOV% : num [1:14] 16.2 7.8 11.3 26.9 14.2 0 31.6 11.1 25 NA ...

#> ..$ USG% : num [1:14] 26 15.9 24.6 22.6 21.4 14.7 13.9 27.3 13.5 0 ...

#> ..$ ORtg : num [1:14] 126 100 88 67 117 138 92 67 102 0 ...

#> ..$ DRtg : num [1:14] 94 98 105 102 100 102 100 107 96 106 ...

#> ..$ Role : chr [1:14] "Starter" "Starter" "Starter" "Starter" ...

#> $ pdp\_df :'data.frame': 114 obs. of 4 variables:

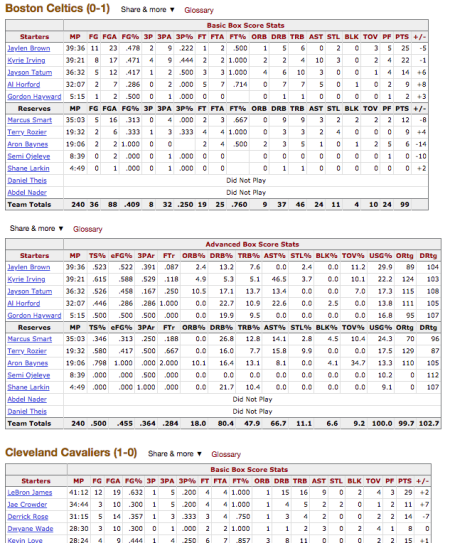
#> ..$ time : num [1:114] 16 72 88 99 131 151 166 191 191 238 ...

#> ..$ visitor: num [1:114] 2 4 4 4 4 4 6 6 6 8 ...

#> ..$ home : num [1:114] 0 0 1 3 5 7 7 8 9 9 ...

#> ..$ period : num [1:114] 1 1 1 1 1 1 1 1 1 1 ...

The first 4 elements of the list (visitor\_basic\_boxscore, visitor\_adv\_boxscore, home\_basic\_boxscore, home\_basic\_boxscore) are data frames for the basic and advanced box scores of both teams. They are the box scores you see on BasketBallReference.com (with some minor preprocessing):



Box scores from BasketballReference.com.

The last element of the list is a data frame containing play-by-play information, much like that in [the previous post](https://statisticaloddsandends.wordpress.com/2018/12/13/recreating-the-nba-lead-tracker-graphic/).

Hopefully some of you will take this data and make some cool data visualizations from it! Below, I will walk through how I use this dataset for two visualizations.

**Do Kevin Durant and Steph Curry get in each other’s way?**

Kevin Durant and Steph Curry are two of the most lethal scorers in today’s game. Does both of them being on the same team hamper the other’s production? This is pretty difficult to pin down what we mean by this statement precisely; we will instead make do with a couple of suggestive plots.

The first plot we can make is a scatterplot of the points scored by these two players. If the relationship is positively correlated, it could mean that they make each other better.

Let’s pull out the game\_ids for Golden State:

team\_name <- "Golden State Warriors"

player1 <- "Stephen Curry"

player2 <- "Kevin Durant"

# get game\_ids

game\_rows <- which(game\_df$visitor\_team\_name == team\_name |

game\_df$home\_team\_name == team\_name)

game\_ids <- game\_df[game\_rows, "game\_id"]

Next, we set up a data frame consisting of 4 columns: the game\_id, game\_type (regular or playoff), and points scored by players 1 and 2 respectively.

points\_df <- data.frame(matrix(NA, ncol = 4, nrow = length(game\_ids)))

names(points\_df) <- c("game\_id", "game\_type", player1, player2)

The loop below builds up the data frame row by row (not the most R-like way of programming but it gets the job done…):

for (i in 1:nrow(points\_df)) {

id <- game\_ids[i]

points\_df[i, 1] <- id

points\_df[i, 2] <- game\_df[game\_rows[i], "game\_type"]

# get the correct basic box score

if (game\_df[game\_rows[i], "visitor\_team\_name"] == team\_name) {

boxscore <- master\_list[[id]]$visitor\_basic\_boxscore

} else {

boxscore <- master\_list[[id]]$home\_basic\_boxscore

}

# get player points

if (player1 %in% boxscore$Player) {

points\_df[i, 3] <- subset(boxscore, Player == player1)$PTS

}

if (player2 %in% boxscore$Player) {

points\_df[i, 4] <- subset(boxscore, Player == player2)$PTS } } head(points\_df)

#> game\_id game\_type Stephen Curry Kevin Durant

#> 1 201710170GSW Regular 22 20

#> 2 201710200NOP Regular 28 22

#> 3 201710210MEM Regular 37 29

#> 4 201710230DAL Regular 29 25

#> 5 201710250GSW Regular 30 29

#> 6 201710270GSW Regular 20 31

Next, we make the scatterplot (notice how we can use the get function to refer to the columns programmatically). We force the axes to start at 0 and the aspect ratio to be 1, and include the 45 degree line for reference.

ggplot(data = points\_df, aes(x = get(player1), y = get(player2))) +

geom\_point(alpha = 0.5) +

geom\_smooth(se = FALSE) +

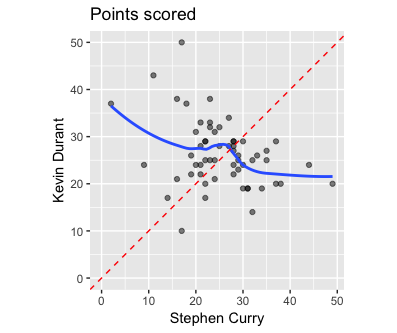
geom\_abline(slope = 1, intercept = 0, lwd = 0.5, col = "red", lty = 2) +

scale\_x\_continuous(limits = c(0, NA)) +

scale\_y\_continuous(limits = c(0, NA)) +

labs(x = player1, y = player2, title = "Points scored") +

coord\_fixed()



One might say, wouldn’t it be better to compare games where KD and Curry play, vs. games where only KD plays? If KD scores more if he is playing with Curry, that could suggest that playing together makes him more productive. To make these plots, we introduce two new columns which are Boolean variables representing whether a given player played in that game or not. We then make boxplots to compare the points scored depending on whether the other player played.

# ignore games where both didn't play

points\_df2 <- points\_df[!(is.na(points\_df[[3]]) & is.na(points\_df[[4]])), ]

points\_df2[[paste(player1, "played")]] <- !is.na(points\_df2[[3]])

points\_df2[[paste(player2, "played")]] <- !is.na(points\_df2[[4]])

# boxplots

ggplot(data = points\_df2, aes(x = get(paste(player1, "played")), y = get(player2))) +

geom\_boxplot() +

geom\_jitter(height = 0, alpha = 0.5, col = "blue") +

labs(x = paste(player1, "played?"), y = "Points scored",

title = paste("Points", player2, "scored"))

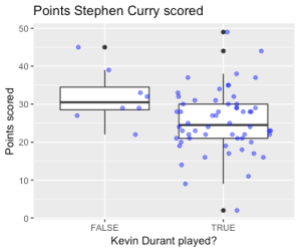
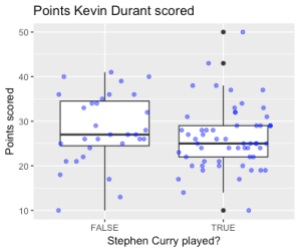
ggplot(data = points\_df2, aes(x = get(paste(player2, "played")), y = get(player1))) +

geom\_boxplot() +

geom\_jitter(height = 0, alpha = 0.5, col = "blue") +

labs(x = paste(player2, "played?"), y = "Points scored",

title = paste("Points", player1, "scored"))



From the plots, it looks like each player scores a bit more when the other person is not playing, although there is quite a bit of variance.

Instead of looking at points scored, we could look at more sophisticated measures of production. For example, the NBA computes individual offensive and defensive ratings for each game based on a complicated formula (see [here](https://www.basketball-reference.com/about/ratings.html) for details). These measures are provided as ORtg and DRtg in the advanced box score.

The code below pulls out the ORtg and DRtg values for KD and Curry:

points\_df <- data.frame(matrix(NA, ncol = 6, nrow = length(game\_ids)))

names(points\_df) <- c("game\_id", "game\_type",

sapply(c(player1, player2), function(x) paste(x, "ORtg")),

sapply(c(player1, player2), function(x) paste(x, "DRtg")))

for (i in 1:nrow(points\_df)) {

id <- game\_ids[i]

points\_df[i, 1] <- id

points\_df[i, 2] <- game\_df[game\_rows[i], "game\_type"]

# get the correct advanced box score

if (game\_df[game\_rows[i], "visitor\_team\_name"] == team\_name) {

boxscore <- master\_list[[id]]$visitor\_adv\_boxscore

} else {

boxscore <- master\_list[[id]]$home\_adv\_boxscore

}

# get player ORtg & DRtg

if (player1 %in% boxscore$Player) {

points\_df[i, 3] <- subset(boxscore, Player == player1)$ORtg

points\_df[i, 5] <- subset(boxscore, Player == player1)$DRtg

}

if (player2 %in% boxscore$Player) {

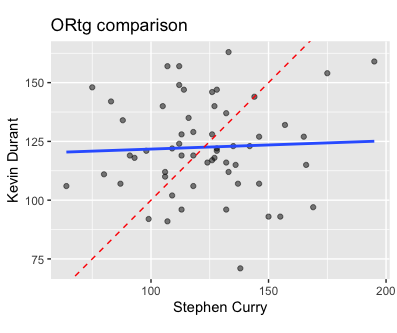
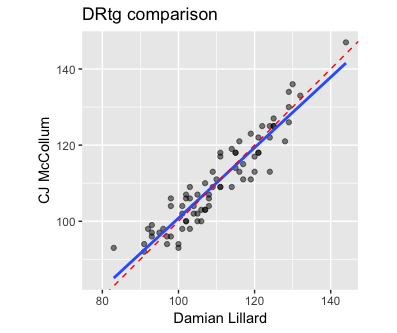
points\_df[i, 4] <- subset(boxscore, Player == player2)$ORtg

points\_df[i, 6] <- subset(boxscore, Player == player2)$DRtg

}

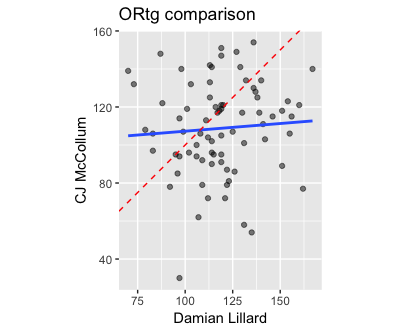
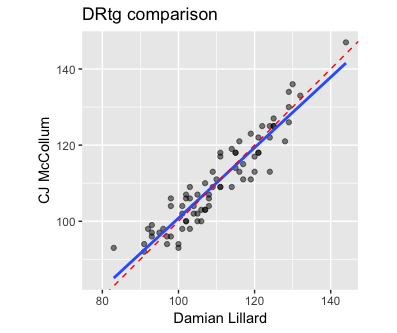
}

And here are the plots (along with a linear fit):

This was pretty interesting to me: it looks like the ORtgs are independent of each other while the DRtgs are highly correlated with each other. Might this suggest that defense is more of a team activity, while offense can be driven by individuals?

There are of course many other visualizations we could do to explore this question. We might also want to see how the KD-Curry pairing compares with other pairings. Because of the way the code is written, it’s really easy to generate these figures for other pairings: we just need to update the team\_name, player1 and player2 variables and rerun the code. As an example, here are the ORtg and DRtg scatterplots for Damian Lillard and C.J. McCollum of the Portland Trail Blazers:

We see similar trends here.

**Are the Golden State Warriors really better in the 3rd quarter?**

Last season, it seemed like the Warriors were often down at the half, only to come roaring back in the third quarter. Is this really their best quarter? ([This NYT article](https://www.nytimes.com/2018/05/31/sports/warriors-third-quarter.html) says yes and suggests why that might be the case.) To answer this question, we’ll make a lead tracker graphic (much like the [previous post](https://statisticaloddsandends.wordpress.com/2018/12/13/recreating-the-nba-lead-tracker-graphic/)) but (i) we will do it for all games GSW played in, and (ii) we reset the score differential to 0 at the beginning of each period.

We first set up a data frame that records the game\_id, whether the team was at home, and whether they won (we might want to use latter two variables in our plotting later):

team\_name <- "Golden State Warriors"

game\_rows <- which(game\_df$visitor\_team\_name == team\_name |

game\_df$home\_team\_name == team\_name)

game\_ids <- game\_df[game\_rows, "game\_id"]

# set up game data frame and play-by-play list

df <- data.frame(matrix(NA, ncol = 3, nrow = length(game\_ids)))

names(df) <- c("game\_id", "home", "win")

df$game\_id <- game\_ids

df$home <- game\_df[game\_rows, "home\_team\_name"] == team\_name

df$win <- (df$home & game\_df[game\_rows, "home\_pts"] > game\_df[game\_rows, "visitor\_pts"]) |

((!df$home) & game\_df[game\_rows, "home\_pts"] < game\_df[game\_rows, "visitor\_pts"])

Next, we create a list with keys being the game\_ids and the value being the play-by-play data frame from the master list we scraped earlier. We standardize the column names so that GSW’s score is labeled team and the opponent’s score is labeled opp. (Note: At the time of writing, BasketBallReference.com had incorrect data for the game 201710200NOP, so I have removed that game from our data.)

pbp\_list <- list()

for (i in 1:nrow(df)) {

id <- game\_ids[i]

pdp <- master\_list[[id]]$pdp\_df

if (df[i, "home"]) {

names(pdp) <- c("time", "opp", "team", "period")

} else {

names(pdp) <- c("time", "team", "opp", "period")

}

pbp\_list[[id]] <- pdp

}

# NOTE: mistake in 201710200NOP. original data on website is wrong

# https://www.basketball-reference.com/boxscores/pbp/201710200NOP.html

# so we remove it

df <- df[-which(df$game\_id == "201710200NOP"), ]

pbp\_list[["201710200NOP"]] <- NULL

We create a function parse\_pbp which helps us reset the score at the beginning of every quarter. To make the lines in our eventual visualization smoother, parse\_pbp also adds additional rows for the beginning and end of every quarter.

parse\_pbp <- function(pbp) {

pbp <- rbind(0, pbp)

new\_pbp <- pbp

# get points scored in the period

last\_opp <- 0; last\_team <- 0; last\_period <- 0 for (i in 2:nrow(pbp)) { if (pbp[i, "period"] > last\_period + 1) {

last\_period <- last\_period + 1

last\_opp <- pbp[i-1, "opp"]

last\_team <- pbp[i-1, "team"]

}

new\_pbp$opp[i] <- pbp$opp[i] - last\_opp

new\_pbp$team[i] <- pbp$team[i] - last\_team

}

# add extra rows to denote beginning and end of periods

num\_period <- max(new\_pbp$period)

for (i in 1:num\_period) {

end\_row <- new\_pbp[max(which(new\_pbp$period == i)), ]

end\_row[1] <- 12 \* 60 \* min(i, 4) + 5 \* 60 \* max(i-4, 0)

beg\_row <- c(0, 0, 0, i)

beg\_row[1] <- 12 \* 60 \* min(i-1, 4) + 5 \* 60 \* max(i-1-4, 0)

new\_pbp <- rbind(new\_pbp, beg\_row)

new\_pbp <- rbind(new\_pbp, end\_row)

}

new\_pbp <- new\_pbp[order(new\_pbp$time), ]

new\_pbp$adv <- with(new\_pbp, team - opp)

new\_pbp[-1, ]

}

pbp\_list <- lapply(pbp\_list, parse\_pbp)

Finally, we make the list into a dataframe:

pbp\_df <- data.frame(matrix(ncol = 6, nrow = 0))

names(pbp\_df) <- c("game\_id", "home", "win", "time", "period", "adv")

for (i in seq\_along(pbp\_list)) {

xx <- pbp\_list[[i]]

xx$game\_id <- df$game\_id[i]

xx$home <- df$home[i]

xx$win <- df$win[i]

pbp\_df <- rbind(pbp\_df, xx) } head(pbp\_df) #> time opp team period adv game\_id home win

#> 124 0 0 0 1 0 201710170GSW TRUE FALSE

#> 2 13 2 0 1 -2 201710170GSW TRUE FALSE

#> 3 55 2 1 1 -1 201710170GSW TRUE FALSE

#> 4 55 2 2 1 0 201710170GSW TRUE FALSE

#> 5 90 2 5 1 3 201710170GSW TRUE FALSE

#> 6 104 5 5 1 0 201710170GSW TRUE FALSE

We can now make the line plot easily. We use the interaction function to group the line plots by both game\_id and quarter, color the lines by whether GSW won or not, and manually define the breaks to be at the end of the periods:

periods <- unique(pbp\_df$period)

x\_value <- ifelse(periods <= 4, 12 \* 60 \* periods,

12 \* 60 \* 4 + 5 \* 60 \* (periods - 4))

x\_label <- ifelse(periods <= 4, paste0("Q", periods),

paste0("OT", periods - 4))

ggplot(pbp\_df, aes(x = time, y = adv)) +

geom\_line(aes(col = win, group = interaction(game\_id, period)),

lwd = 0.1) +

geom\_smooth(aes(group = period), col = "black", se = FALSE) +

scale\_x\_continuous(breaks = x\_value, labels = x\_label) +

scale\_color\_manual(values = c("#ff6600", "#3366ff")) +

labs(title = paste("Point Advantage by Quarter,", team\_name)) +

theme\_minimal() +

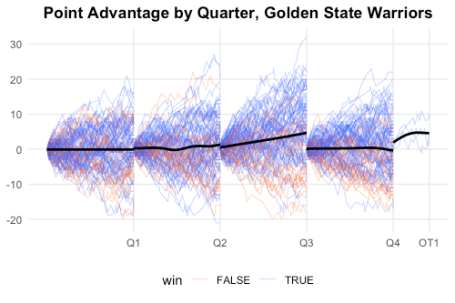
theme(plot.title = element\_text(face = "bold", hjust = 0.5),

axis.title.x = element\_blank(), axis.title.y = element\_blank(),

panel.grid.minor.x = element\_blank(),

panel.grid.minor.y = element\_blank(),

legend.position = "bottom")



It may not be so obvious in this plot, but there certainly seems to be a bit of a difference in the third quarter. On average, the Warriors are even or up 1-2 points, but in the third quarter GSW gains almost 5 points. The difference is more obvious in the plot below, where we zoom in on the y-axis. (Note that we have to use coord\_cartesian instead of scale\_y\_continuous to zoom in: scale\_y\_continuous will remove the points outside the range before plotting, which results in the incorrect result for the geom\_smooth layer.

ggplot(pbp\_df, aes(x = time, y = adv)) +

geom\_line(aes(col = win, group = interaction(game\_id, period)),

lwd = 0.1) +

geom\_smooth(aes(group = period), col = "black", se = FALSE) +

scale\_x\_continuous(breaks = x\_value, labels = x\_label) +

scale\_color\_manual(values = c("#ff6600", "#3366ff")) +

coord\_cartesian(ylim = c(-10, 10)) +

labs(title = paste("Point Advantage by Quarter,", team\_name)) +

theme\_minimal() +

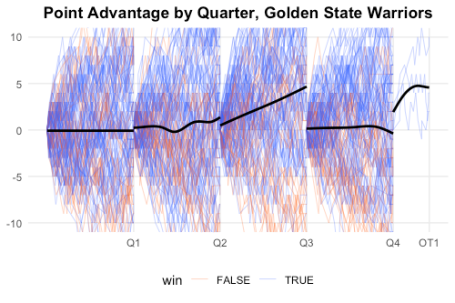
theme(plot.title = element\_text(face = "bold", hjust = 0.5),

axis.title.x = element\_blank(), axis.title.y = element\_blank(),

panel.grid.minor.x = element\_blank(),

panel.grid.minor.y = element\_blank(),

legend.position = "bottom")



The advantage becomes even more obvious when we facet by whether GSW won or not: in wins, they gain almost 10 points in the 3rd quarter, compared to something like 1-3 points in each of the other quarters.

