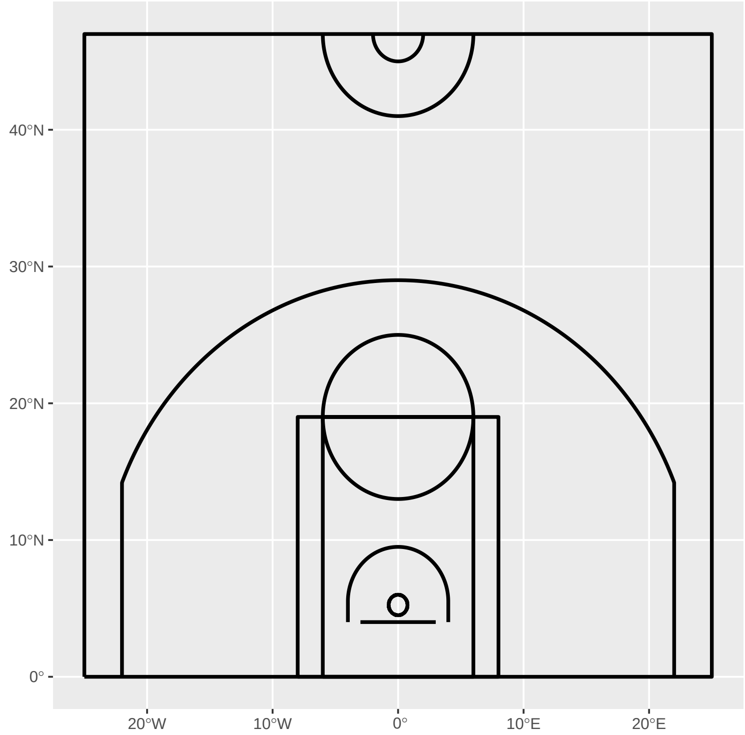
The results below are generated from an R script.

```
# Packages
rm(list=ls())
library(data.table)
library(jsonlite)
library(dplyr)
library(progress)
library(sf)
library(ggplot2)
library(paletteer)
library(stringr)
library(lubridate)
library(gridExtra)
source("helper_functions.R")

# Directory
DATA_DIR <- "./data"

# LOAD DATA
################################################################################

# Load Courts data
full_court <- st_read(file.path(DATA_DIR, "nba-court-lines-05feb2024.gpkg"),
                      layer = "nba-court-lines-05feb2024")

## Reading layer ‘nba-court-lines-05feb2024’ from data source
##   ‘/Users/jonnycodd/Documents/MASTERS/Geo spacial/NBA Project/data/nba-court-lines-05feb2024.gpkg’
##   using driver ‘GPKG’
## Simple feature collection with 12 features and 1 field
## Geometry type: MULTILINESTRING
## Dimension:     XY
## Bounding box:  xmin: -25 ymin: 0 xmax: 25 ymax: 94
## Geodetic CRS:  WGS 84

half_court <- full_court[-1, ]

# Create the basketball court plot
court_plot <- ggplot() +
  geom_sf(data = half_court, color = "black", fill = "transparent",
          linewidth= 1)  # Increase line thickness here
court_plot
```



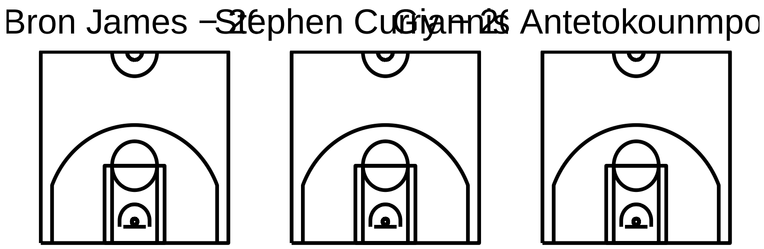
```r
# Load shots data
shots_data <- data.table()
for (year in 2004:2005) {

  file_name <- sprintf("NBA_%d_Shots.csv", year)
  year_data <- fread(file.path(DATA_DIR, file_name))
  shots_data <- rbindlist(list(shots_data, year_data), use.names = TRUE, fill = TRUE)

}

# Make SF point object
shots_data_sf <- st_as_sf(shots_data, coords = c("LOC_X", "LOC_Y"), crs = st_crs(half_court))

## Identify shots in the restricted area
################################################################################

# The restricted area is currently just a semi circle, so we need to connect the
# end points
r_a <- half_court %>%
  filter(Feature=="Restricted area")

# Create back line
back_line <- st_sfc(st_linestring(matrix(c(-4, 4, 4, 4), ncol = 2, byrow = TRUE)), crs = st_crs(r_a))
r_a <- st_union(r_a, back_line) # Combine them

## Warning:  attribute variables are assumed to be spatially constant throughout all geometries

# Create polygon from coordinates
r_a_coords <- st_coordinates(r_a) # Extract coordinates
r_a_coords <- rbind(r_a_coords, r_a_coords[1,]) # Make first and last the same
```

```r
r_a_polygon <- st_polygon(list(r_a_coords)) # Make polygon
r_a_polygon <- st_zm(r_a_polygon) # Only keep x and y coords
r_a_polygon <- st_make_valid(r_a_polygon)
r_a_polygon_sf <- st_sf(geometry = st_sfc(r_a_polygon, crs = st_crs(r_a))) # make sf object

# Identify shots inside restricted area using st_within
inside_r_a <- st_within(shots_data_sf, r_a_polygon_sf)
```

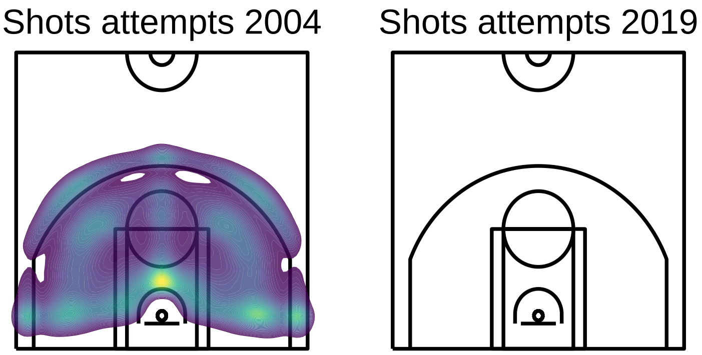
```
## Warning in st_is_longlat(x):  bounding box has potentially an invalid value range for longlat
data
## Warning in st_is_longlat(x):  bounding box has potentially an invalid value range for longlat
data
```

```r
# Create_indicator
shots_data_sf$inside_r_a <- as.integer(lengths(inside_r_a) > 0)
shots_data$inside_r_a <- as.integer(lengths(inside_r_a) > 0)


## Identify shots just behind the 3 point line
################################################################################


# HEAT MAP: per players
################################################################################
# Define a function to create the plot for a player
create_player_plot <- function(shots_data, year, player_name) {

  data <- shots_data %>%
      filter(SEASON_1 == year) %>%
      filter(SHOT_MADE == TRUE) %>%
      filter(str_detect(PLAYER_NAME, player_name))

  title <- paste0(player_name, ' - ', year)

  ggplot() +
    geom_sf(data = half_court, color = "black", fill = "transparent", linewidth = 1) +
    geom_point(data = data, aes(x = LOC_X, y = LOC_Y),
               size = 1, alpha = 0.1) +
    geom_density_2d_filled(data, mapping = aes(x = LOC_X, y = LOC_Y, fill = ..level..),
                           contour_var = "ndensity", breaks = seq(0.03, 1.0, length.out = 80), alpha = .
    scale_x_continuous(limits = c(-27.5, 27.5)) +
    scale_y_continuous(limits = c(0, 45)) +
    labs(title = title) +
    theme(legend.position = "none",
          plot.title = element_text(hjust = 0.5, size = 20),
          axis.title.x = element_blank(),
          axis.title.y = element_blank(),
          axis.text.x = element_blank(),
          axis.text.y = element_blank(),
          axis.ticks = element_blank(),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank())
}

# Create plots for each player
player1 <- create_player_plot(shots_data, 2013, "LeBron James")
player2 <- create_player_plot(shots_data, 2016, "Stephen Curry")
player3 <- create_player_plot(shots_data, 2019, "Giannis Antetokounmpo")

# Arrange plots side by side
side_by_side_plots <- grid.arrange(player1, player2, player3, ncol = 3)
```

```
## Warning:  The dot-dot notation (‘..level..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(level)‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was generated.
```

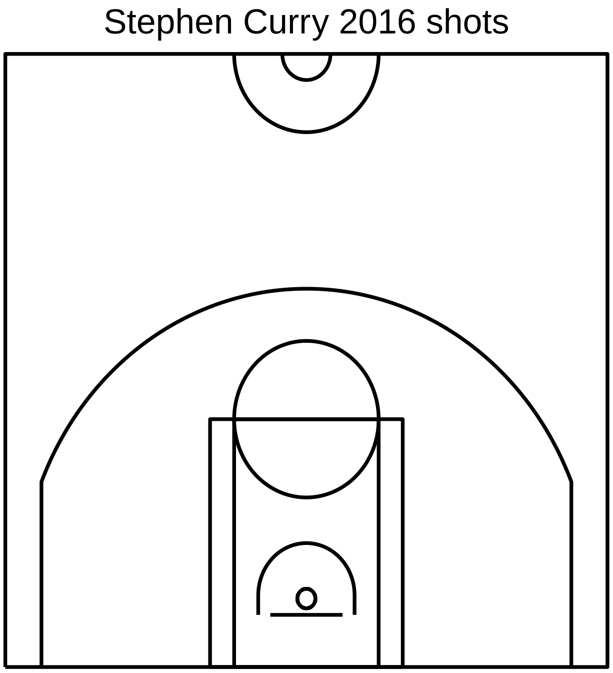


```
print(side_by_side_plots)

## TableGrob (1 x 3) "arrange": 3 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
## 3 3 (1-1,3-3) arrange gtable[layout]

# HEAT MAP:  overall
################################################################################
outside_before <- shots_data %>%
  filter(SEASON_1 == 2004) %>%
  filter(BASIC_ZONE != "Restricted Area")

outside_now <- shots_data %>%
  filter(SEASON_1 == 2019) %>%
  filter(BASIC_ZONE != "Restricted Area")

before <- ggplot() +
  geom_sf(data = half_court, color = "black", fill = "transparent", linewidth = 1) +
  geom_density_2d_filled(outside_before, mapping = aes(x = LOC_X, y = LOC_Y, fill = ..level..),
                         contour_var = "ndensity", breaks = seq(0.1, 1.0, length.out = 50), alpha = .8)
  scale_x_continuous(limits = c(-27.5, 27.5)) +
  scale_y_continuous(limits = c(0, 45)) +
  labs(title = ’Shots attempts 2004’) +
  theme(legend.position = "none",
        plot.title = element_text(hjust = 0.5, size = 20),
        axis.title.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank())

now <- ggplot() +
  geom_sf(data = half_court, color = "black", fill = "transparent", linewidth = 1) +
  geom_density_2d_filled(outside_now, mapping = aes(x = LOC_X, y = LOC_Y, fill = ..level..),
                         contour_var = "ndensity", breaks = seq(0.1, 1.0, length.out = 50), alpha = .8)
  scale_x_continuous(limits = c(-27.5, 27.5)) +
  scale_y_continuous(limits = c(0, 45)) +
  labs(title = ’Shots attempts 2019’) +
  theme(legend.position = "none",
        plot.title = element_text(hjust = 0.5, size = 20),
        axis.title.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank())

side_by_side_plots <- grid.arrange(before, now, ncol = 2)

## Warning:  Removed 445 rows containing non-finite outside the scale range
## (‘stat_density2d_filled()‘).
```

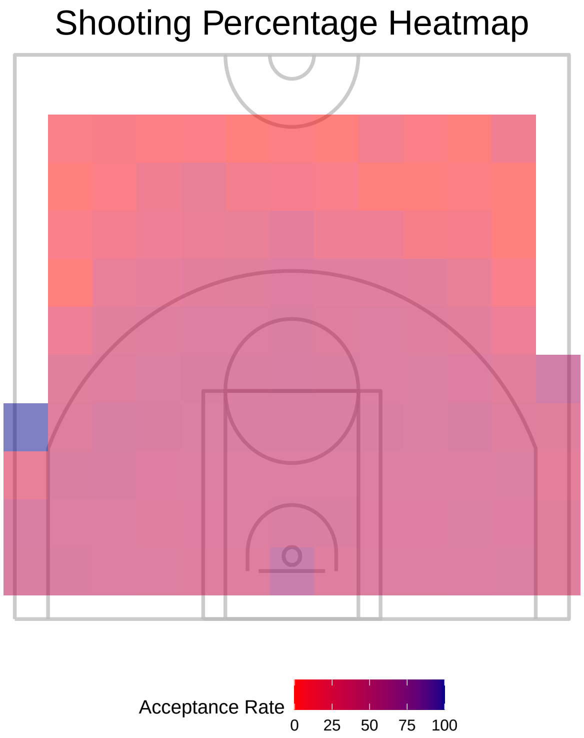

Shots attempts 2004

Shots attempts 2019

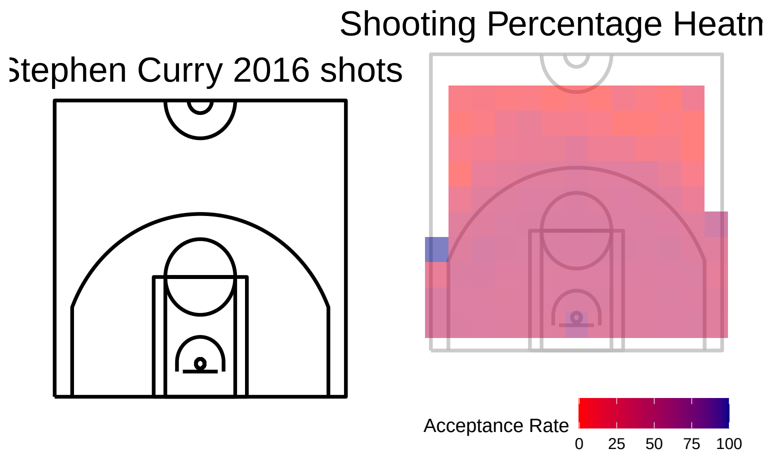
```
print(side_by_side_plots)

## TableGrob (1 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]

# ACCEPTANCE RATE:  Curry
################################################################################
shots_curry <- shots_data %>%
  filter(SEASON_1 == 2016) %>%
  filter(str_detect(PLAYER_NAME, "Stephen Curry")) %>%
  select(PLAYER_NAME, GAME_DATE, EVENT_TYPE, SHOT_MADE, SHOT_TYPE, LOC_X, LOC_Y)

shots_curry

## Empty data.table (0 rows and 7 cols): PLAYER_NAME,GAME_DATE,EVENT_TYPE,SHOT_MADE,SHOT_TYPE,LOC_X...

# Create scatter plot
scatter_plot <- ggplot() +
  geom_sf(data = half_court, color = "black", fill = "transparent", linewidth = 1) +
  geom_point(data = shots_curry, aes(x = LOC_X, y = LOC_Y, color = SHOT_MADE), alpha = 0.3) +  # Add po
  labs(title = "Shots Scatterplot", x = "LOC_X", y = "LOC_Y") +  # Add title and axis labels
  theme_minimal() +
  scale_x_continuous(limits = c(-27.5, 27.5)) +
  scale_y_continuous(limits = c(0, 45)) +
  labs(title = ’Stephen Curry 2016 shots’) +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, size = 20),
        axis.title.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank())

# Print the scatter plot
print(scatter_plot)
```



```
shots_data$LOC_X_simple <- as.integer(shots_data$LOC_X/4)*4
shots_data$LOC_Y_simple <- as.integer(shots_data$LOC_Y/4)*4

shot_summary <- shots_data %>%
  group_by(LOC_X_simple, LOC_Y_simple) %>%
  summarize(total_shots = n(), made_shots = sum(SHOT_MADE))

## `summarise()` has grouped output by 'LOC_X_simple'.  You can override using the `.groups`
## argument.

# Calculate the acceptance rate
shot_summary$acceptance_rate <- shot_summary$made_shots / shot_summary$total_shots * 100

# Create the heatmap plot
heatmap_plot <- ggplot() +
  geom_sf(data = half_court, color = "grey", fill = "transparent", linewidth = 1, alpha=0.8) +  # Plot t
  geom_raster(data = shot_summary, aes(x = LOC_X_simple, y = LOC_Y_simple, fill = acceptance_rate), alph
  scale_fill_gradient(low = "red", high = "darkblue", name = "Acceptance Rate") +  # Customize fill colo
  scale_x_continuous(limits = c(-27.5, 27.5)) +  # Set x-axis limits
  scale_y_continuous(limits = c(0, 45)) +  # Set y-axis limits
  labs(title = "Shooting Percentage Heatmap") +  # Add title
  theme_minimal() +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, size = 20),
        axis.title.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank())

# Print the heatmap plot
print(heatmap_plot)

## Warning:  Removed 132 rows containing missing values or values outside the scale range
## (`geom_raster()`).
```



```
side_by_side_plots <- grid.arrange(scatter_plot, heatmap_plot, ncol = 2)

## Warning:  Removed 132 rows containing missing values or values outside the scale range
## (`geom_raster()`).
```



```
print(side_by_side_plots)

## TableGrob (1 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

The R session information (including the OS info, R version and all packages used):

```
sessionInfo()

## R version 4.3.1 (2023-06-16)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Sonoma 14.0
##
## Matrix products: default
## BLAS:   /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versio
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  LAPACK ve
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: Europe/Rome
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] gridExtra_2.3     lubridate_1.9.3   stringr_1.5.1     paletteer_1.6.0
##  [5] ggplot2_3.5.0     sf_1.0-15         progress_1.2.3    dplyr_1.1.4
##  [9] jsonlite_1.8.8    data.table_1.15.2
##
## loaded via a namespace (and not attached):
##  [1] s2_1.1.6           utf8_1.2.4         generics_0.1.3     class_7.3-22
##  [5] KernSmooth_2.23-22 stringi_1.8.3      hms_1.1.3          magrittr_2.0.3
##  [9] evaluate_0.23      grid_4.3.1         timechange_0.3.0   e1071_1.7-14
## [13] DBI_1.2.2          rematch2_2.1.2     fansi_1.0.6        viridisLite_0.4.2
## [17] scales_1.3.0       isoband_0.2.7      cli_3.6.2          rlang_1.1.3
## [21] crayon_1.5.2       units_0.8-5        munsell_0.5.0      withr_3.0.0
## [25] tools_4.3.1        colorspace_2.1-0   vctrs_0.6.5        R6_2.5.1
## [29] proxy_0.4-27       lifecycle_1.0.4    classInt_0.4-10    MASS_7.3-60.0.1
## [33] pkgconfig_2.0.3    pillar_1.9.0       gtable_0.3.4       glue_1.7.0
## [37] Rcpp_1.0.12        xfun_0.42          tibble_3.2.1       tidyselect_1.2.1
## [41] highr_0.10         rstudioapi_0.15.0  knitr_1.45         farver_2.1.1
## [45] labeling_0.4.3     wk_0.9.1           compiler_4.3.1     prettyunits_1.2.0
```

```r
Sys.time()
```

```
## [1] "2024-03-27 12:16:49 CET"
```