Exploring the Applicability of Price's Law in the Distribution of Wins Above Replacement Among Major League Baseball Clubs from 2000-2019
Code Archive for Research Project:

Michael Hymowitz

4/27/2021

This RMarkdown file contains all the code used in the research paper: Exploring the Applicability of Price's Law in the Distribution of Wins Above Replacement Among Major League Baseball Clubs from 2000-2019.

Loading in the necessary libraries and datasets

```
set.seed(2008)
```

```
# Loading in the necessary libraries
library(tidyverse)

 # Package and settings for pretty-printing tables
library(gridExtra)
tt <- ttheme_default(colhead=list(fg_params = list(parse=FALSE)))

library(MASS)
select <- dplyr::select # Need this as MASS::select overwrites dplyr::select

library(boot)
```

```
# Loading in the necessary datasets
bat_df_raw <- read_csv("../Data/war_daily_bat.csv") # Baseball-Reference position player WAR table
head(bat_df_raw)
```

```
## # A tibble: 6 x 49
##   name_common    age mlb_ID player_ID year_ID team_ID stint_ID lg_ID    PA     G
##   <chr>        <dbl>  <dbl> <chr>       <dbl> <chr>      <dbl> <chr> <dbl> <dbl>
## 1 David Aards~    22 430911 aardsda01    2004 SFG            1 NL        0    11
## 2 David Aards~    24 430911 aardsda01    2006 CHC            1 NL        3    43
## 3 David Aards~    25 430911 aardsda01    2007 CHW            1 AL        0     2
## 4 David Aards~    26 430911 aardsda01    2008 BOS            1 AL        1     5
## 5 David Aards~    27 430911 aardsda01    2009 SEA            1 AL        0     3
## 6 David Aards~    28 430911 aardsda01    2010 SEA            1 AL        0     4
## # ... with 39 more variables: Inn <chr>, runs_bat <dbl>, runs_br <dbl>,
```

```
## #   runs_dp <dbl>, runs_field <dbl>, runs_infield <chr>, runs_outfield <chr>,
## #   runs_catcher <chr>, runs_good_plays <chr>, runs_defense <dbl>,
## #   runs_position <dbl>, runs_position_p <dbl>, runs_replacement <dbl>,
## #   runs_above_rep <dbl>, runs_above_avg <dbl>, runs_above_avg_off <dbl>,
## #   runs_above_avg_def <dbl>, WAA <chr>, WAA_off <chr>, WAA_def <chr>,
## #   WAR <chr>, WAR_def <chr>, WAR_off <chr>, WAR_rep <chr>, salary <chr>,
## #   pitcher <chr>, teamRpG <chr>, oppRpG <dbl>, oppRpPA_rep <dbl>,
## #   oppRpG_rep <chr>, pyth_exponent <chr>, pyth_exponent_rep <chr>,
## #   waa_win_perc <chr>, waa_win_perc_off <chr>, waa_win_perc_def <chr>,
## #   waa_win_perc_rep <chr>, OPS_plus <chr>, TOB_lg <dbl>, TB_lg <dbl>
```

```r
pit_df_raw <- read_csv("../Data/war_daily_pitch.csv") # Baseball-Reference pitcher WAR table
head(pit_df_raw)
```

```
## # A tibble: 6 x 43
##   name_common    age mlb_ID player_ID year_ID team_ID stint_ID lg_ID     G    GS
##   <chr>        <dbl>  <dbl> <chr>       <dbl> <chr>      <dbl> <chr> <dbl> <dbl>
## 1 David Aards~    22 430911 aardsda01    2004 SFG            1 NL       11     0
## 2 David Aards~    24 430911 aardsda01    2006 CHC            1 NL       45     0
## 3 David Aards~    25 430911 aardsda01    2007 CHW            1 AL       25     0
## 4 David Aards~    26 430911 aardsda01    2008 BOS            1 AL       47     0
## 5 David Aards~    27 430911 aardsda01    2009 SEA            1 AL       73     0
## 6 David Aards~    28 430911 aardsda01    2010 SEA            1 AL       53     0
## # ... with 33 more variables: IPouts <dbl>, IPouts_start <chr>,
## #   IPouts_relief <chr>, RA <dbl>, xRA <dbl>, xRA_sprp_adj <chr>,
## #   xRA_extras_adj <chr>, xRA_def_pitcher <dbl>, PPF <dbl>, PPF_custom <chr>,
## #   xRA_final <chr>, BIP <dbl>, BIP_perc <dbl>, RS_def_total <dbl>,
## #   runs_above_avg <chr>, runs_above_avg_adj <chr>, runs_above_rep <chr>,
## #   RpO_replacement <dbl>, GR_leverage_index_avg <dbl>, WAR <chr>,
## #   salary <chr>, teamRpG <dbl>, oppRpG <chr>, pyth_exponent <chr>,
## #   waa_win_perc <chr>, WAA <chr>, WAA_adj <dbl>, oppRpG_rep <chr>,
## #   pyth_exponent_rep <chr>, waa_win_perc_rep <chr>, WAR_rep <chr>,
## #   ERA_plus <chr>, ER_lg <dbl>
```

Cleaning the datasets and combining them into one over-arching dataset

```r
# Cleaning the datasets
bat_df_clean <- bat_df_raw %>%
    select(name_common, player_ID, year_ID, team_ID, WAR) %>%
    filter(year_ID >= 2000, year_ID <= 2019) %>%
    mutate(WAR = as.numeric(ifelse(WAR == "NULL", 0, WAR)))

pit_df_clean <- pit_df_raw %>%
    select(name_common, player_ID, year_ID, team_ID, WAR) %>%
    filter(year_ID >= 2000, year_ID <= 2019) %>%
    mutate(WAR = as.numeric(ifelse(WAR == "NULL", 0, WAR)))
```

Note that all of the records in the `bat_df_raw` table with `WAR = "NULL"` correspond to entries which have `PA = 0`, meaning they did not have a single plate appearance the entire season. Likewise, all of the records in the `pit_df_raw` table with `WAR = "NULL"` correspond to entries which have `IPOuts = 0`, meaning they did not record a single out while pitching the entire season. Thus, we can presume that those records which have `WAR = "NULL"` did not actually provide any significant contributions to their team, and thus we can safely assign these `WAR` values of 0.

```r
# Combining the two tables into a single table, and grouping records corresponding to the
# sample player for the same team for the same year into a single record
war_df <- bind_rows(bat_df_clean, pit_df_clean) %>%
    group_by(name_common, player_ID, year_ID, team_ID) %>%
    summarize(WAR = sum(WAR), .groups = "keep")
head(war_df)
```

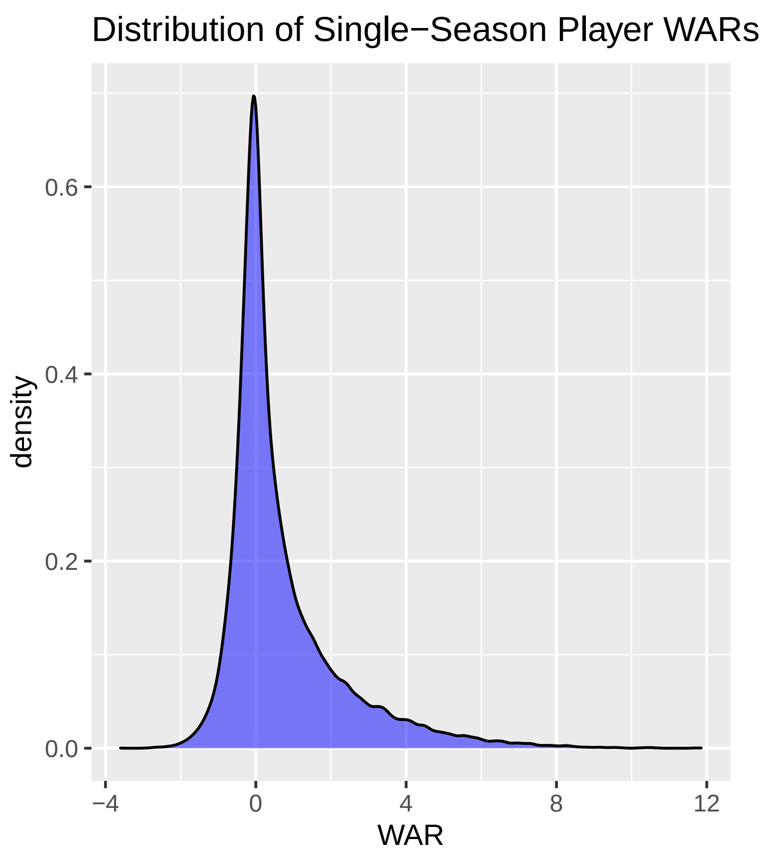
```
## # A tibble: 6 x 5
## # Groups:   name_common, player_ID, year_ID, team_ID [6]
##   name_common player_ID year_ID team_ID   WAR
##   <chr>       <chr>       <dbl> <chr>   <dbl>
## 1 A.J. Achter achteaj01    2014 MIN     -0.03
## 2 A.J. Achter achteaj01    2015 MIN     -0.18
## 3 A.J. Achter achteaj01    2016 LAA      0.48
## 4 A.J. Burnett burnea.01   2000 FLA      1.14
## 5 A.J. Burnett burnea.01   2001 FLA      1.39
## 6 A.J. Burnett burnea.01   2002 FLA      3.98
```

Preliminary graphs of the distribution of the data

```
# This graph and table shows the distribution of single-season player WARs in the sample
single_season_plt <- ggplot(war_df) +
    geom_density(aes(x = WAR), fill = "blue", alpha = 0.5) +
    labs(title = "Distribution of Single-Season Player WARs")

war_distribution_br <- war_df %>%
    mutate(WAR_bucket =
               ifelse(WAR < 0, "< 0",
                      ifelse(WAR < 2, "[0,2)",
                             ifelse(WAR < 5, "[2,5)",
                                    ifelse(WAR < 8, "[5,8)",
                                           "> 8")))),
           WAR_bucket = factor(unique(WAR_bucket),
                               levels = c("< 0", "[0,2)", "[2,5)", "[5,8)", "> 8")))  %>%
    group_by(WAR_bucket) %>%
    summarize(team_size = n()) %>%
    rename(WAR = WAR_bucket,
           `Team Size` = team_size)

single_season_neat <- grid.arrange(single_season_plt,
                                   tableGrob(war_distribution_br, rows=NULL, theme=tt),
                                   nrow=1,
                                   widths = c(6, 4))
```



WAR	Team Size
< 0	10923
[0,2)	12908
[2,5)	3554
[5,8)	680
> 8	82

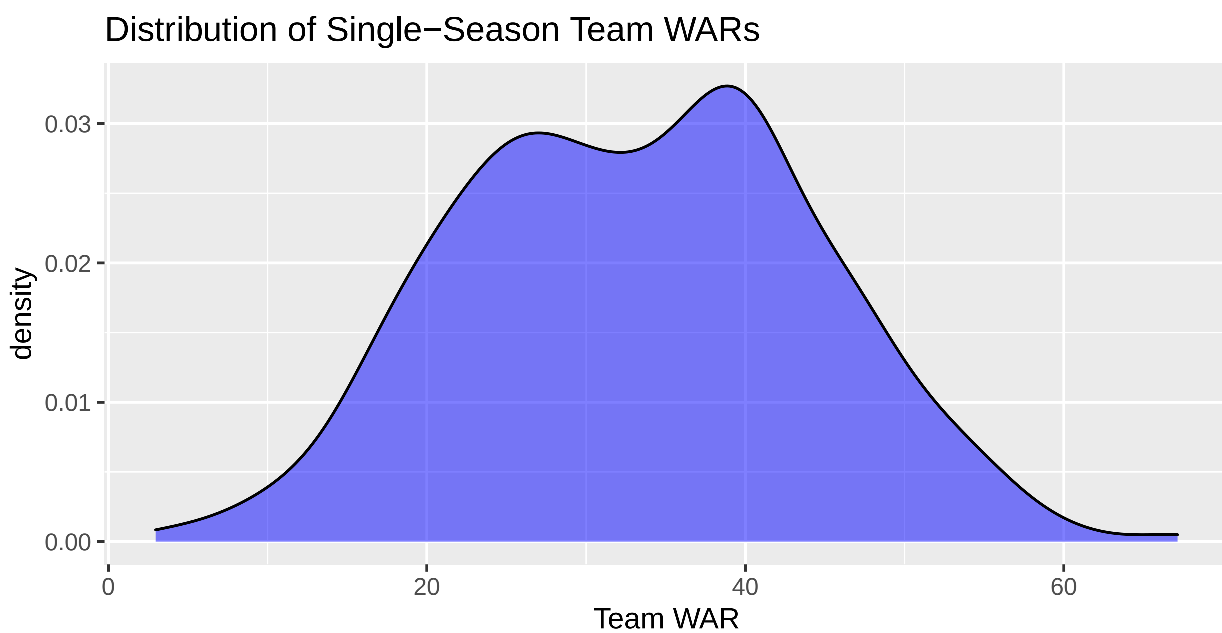
```r
# ggsave("team_WAR_distribution", team_wars_neat, device = "png")
```

```r
# This graph and table shows the distribution of single-season team WARs in the sample
team_wars <- war_df %>%
    mutate(team_year = str_c(team_ID, year_ID, sep = "_")) %>%
    group_by(team_year) %>%
    summarize(team_WAR = sum(WAR))

team_wars_plt <- ggplot(team_wars) +
    geom_density(aes(x = team_WAR), fill = "blue", alpha = 0.5) +
    labs(x = "Team WAR",
         title = "Distribution of Single-Season Team WARs")

team_wars_summary <- team_wars %>%
    .$team_WAR %>%
    summary %>%
    as.matrix %>%
    t() %>%
    round(2)

team_wars_neat <- grid.arrange(team_wars_plt,
                               tableGrob(team_wars_summary, rows=NULL, theme=tt),
                               nrow=2,
                               heights = c(3, 1))
```



Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
2.97	24.98	33.24	33.28	40.99	67.14

```
# ggsave("team_WAR_distribution", team_wars_neat, device = "png")
```

Creating a table of the θ values

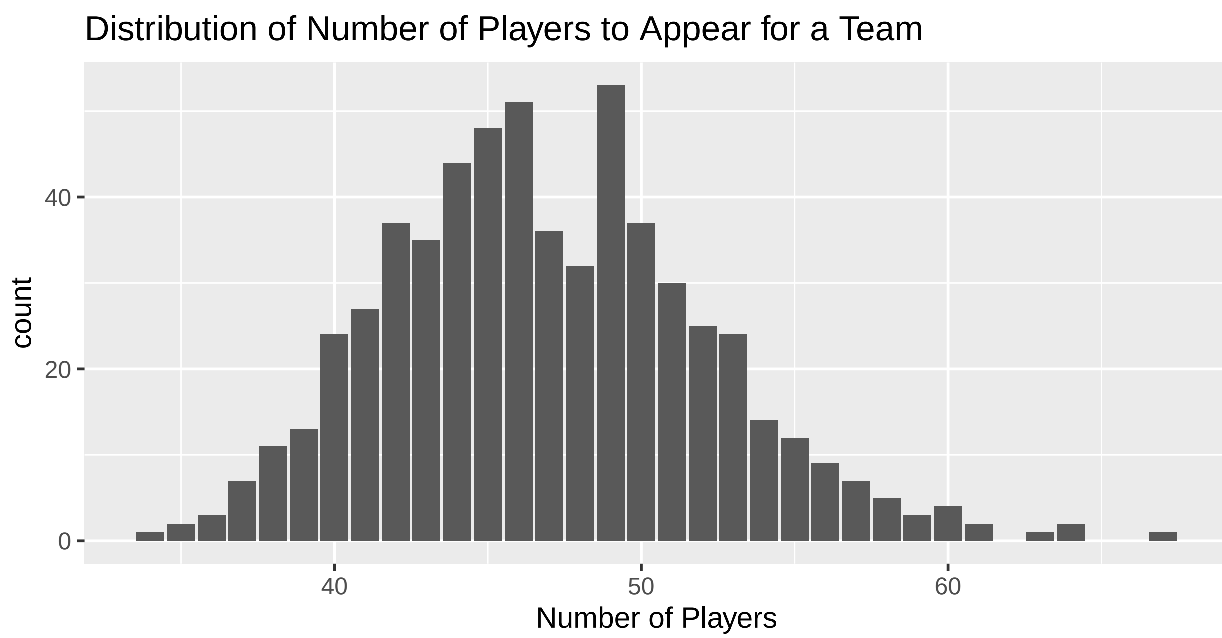
```
# Creating table which notes how many players it took for each team to surpass 50% of
# that team's war
price_analysis <- war_df %>%
    mutate(team_year = str_c(team_ID, year_ID, sep = "_")) %>%
    arrange(team_year, desc(WAR)) %>%
    group_by(team_year) %>%
    left_join(team_wars, by = "team_year") %>%
    mutate(row_num = row_number(),
           team_size = n(),
           sqrt_team_size = sqrt(team_size),
           cWAR = cumsum(WAR),
           pWAR = cWAR / team_WAR) %>%
    filter(pWAR >= .5) %>%
    slice(1) %>%
    ungroup %>%
    mutate(price_is_true = (row_num <= sqrt_team_size)) %>%
    select(team_ID, year_ID, team_year, row_num, team_size, cWAR, team_WAR, pWAR,
           price_is_true)
head(price_analysis)
```

```
## # A tibble: 6 x 9
##   team_ID year_ID team_year row_num team_size  cWAR team_WAR  pWAR price_is_true
##   <chr>     <dbl> <chr>       <int>     <int> <dbl>    <dbl> <dbl> <lgl>
## 1 ANA        2000 ANA_2000        3        45  20.6     37.5 0.548 TRUE
## 2 ANA        2001 ANA_2001        6        38  21.4     39.2 0.546 TRUE
## 3 ANA        2002 ANA_2002        6        40  30.1     54.9 0.549 TRUE
## 4 ANA        2003 ANA_2003        5        43  17.1     33.7 0.507 TRUE
## 5 ANA        2004 ANA_2004        6        38  22.4     42.5 0.526 TRUE
## 6 ARI        2000 ARI_2000        3        41  16.3     32.0 0.511 TRUE
```

```
# This graph and table shows the distribution of team sizes in the sample
team_sizes_plt <- ggplot(price_analysis) +
    geom_bar(aes(team_size)) +
    labs(x = "Number of Players",
         title = "Distribution of Number of Players to Appear for a Team")

team_sizes_summary <- price_analysis %>%
    .$team_size %>%
    summary %>%
        as.matrix %>%
    t() %>%
    round(2)

team_sizes_neat <- grid.arrange(team_sizes_plt,
                               tableGrob(team_sizes_summary, rows=NULL, theme=tt),
                               nrow=2,
                               heights = c(3, 1))
```



Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
34	43	46	46.91	50	67

```
# ggsave("team_sizes_distribution", team_sizes_neat, device = "png")
```

Simulations

To go about demonstrating the applicability of the bootstrap and jackknife methods described in the Method section of the report, this section runs simulations of these methods on a simulated dataset both with the assumption of all team's having independent θ values, as well as simulations with this assumption relaxed, and demonstrate that this method is still reliable. In each of these different settings, we will create 1,000 samples of 30 θ values to mimic our dataset. To construct the theta value of any given sample, we will first use a multivariate normal distribution random number generator, with the correlation matrix coming from a first-order autoregressive model and an inputted non-positive correlation value ρ ("Generating correlation…").

```
# Code from: https://www.r-bloggers.com/2020/02/generating-correlation-matrix-for-ar1-model/
ar1_cor <- function(n, rho) {
    exponent <- abs(matrix(1:n - 1, nrow = n, ncol = n, byrow = TRUE) -
                    (1:n - 1))
    rho^exponent
}
```

```
# Running the simulations with varying levels of correlation between the teams
conf_ints <- map_dfr(c(seq(0, -0.9, length.out = 10)), function(rho) {

    Sigma <- ar1_cor(30, rho)

    test_df <- tibble(team_num = rep(1:30, 1000)) %>%
        arrange(team_num) %>%
        mutate(year = rep(1:1000, 30),
               teamID = str_c("team", team_num, year, sep = "_")) %>%
        arrange(year, team_num) %>%
        group_by(year) %>%
        mutate(theta = mvrnorm(1, rep(0, 30), Sigma) ^ 2) # generating chi-sqared df = 1


    B <- 1000

    mean_boot <- function(data, index) {
        mean(data[index])
    }
    boot_mean <- boot(test_df$theta, statistic = mean_boot, R = B)
    tibble(rho = rho,
           conf_int_0.025 = boot.ci(boot_mean, type = c("basic"))$basic[4],
           conf_int_0.975 = boot.ci(boot_mean, type = c("basic"))$basic[5]) %>%
        mutate(covers_1 = ifelse(conf_int_0.025 <= 1 & conf_int_0.975 >= 1,
                                 "TRUE",
                                 "FALSE"))
}) %>%
    mutate(conf_int_0.025 = signif(conf_int_0.025, 4),
           conf_int_0.975 = signif(conf_int_0.975, 4)) %>%
    rename(Rho = rho,
           `Lower Bound` = conf_int_0.025,
           `Upper Bound` = conf_int_0.975,
           `Interval Covers 1` = covers_1)

conf_ints_neat <- grid.arrange(tableGrob(conf_ints, rows=NULL, theme=tt))
```

Rho	Lower Bound	Upper Bound	Interval Covers 1
0.0	0.9848	1.0170	TRUE
−0.1	0.9949	1.0290	TRUE
−0.2	0.9714	1.0020	TRUE
−0.3	0.9739	1.0050	TRUE
−0.4	0.9792	1.0110	TRUE
−0.5	0.9796	1.0120	TRUE
−0.6	0.9786	1.0110	TRUE
−0.7	0.9772	1.0080	TRUE
−0.8	0.9920	1.0230	TRUE
−0.9	0.9673	0.9996	FALSE

```
# ggsave("simulation_results", conf_ints_neat, device = "png")
```

Analysis

```r
# Creating thetas, a table of the theta values for each team in the sample
thetas <- price_analysis %>%
    mutate(theta = log(row_num) / log(team_size)) %>%
    group_by(team_year) %>%
    summarize(team_WAR = first(team_WAR),
              team_size = first(team_size),
              theta = first(theta))
head(thetas)
```

```
## # A tibble: 6 x 4
##   team_year team_WAR team_size theta
##   <chr>        <dbl>     <int> <dbl>
## 1 ANA_2000      37.5        45 0.289
## 2 ANA_2001      39.2        38 0.493
## 3 ANA_2002      54.9        40 0.486
## 4 ANA_2003      33.7        43 0.428
## 5 ANA_2004      42.5        38 0.493
## 6 ARI_2000      32.0        41 0.296
```

```r
# Setting the number of bootstrap replications
B <- 1000
```

```r
# Using bootstrap to estimate the mean of theta-star
mean_boot <- function(data, index) {
    mean(data[index])
}
boot_mean <- boot(thetas$theta, statistic = mean_boot, R = B)
boot.ci(boot_mean, type = "basic")
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_mean, type = "basic")
##
## Intervals :
## Level      Basic
## 95%   ( 0.3451,  0.3592 )
## Calculations and Intervals on Original Scale
```

Therefore, a 95% confidence interval for the mean value of the distribution of θ^* is (0.3451, 0.3592).

```r
# Using nested bootstrap to estimate the median of theta-star
median_idx <- function(data, idx) {
    median(data[idx])
}

median_nested <- function(data, idx) {
    xstar <- data[idx]
```

```r
    meds <- boot(xstar, median_idx, R = 100)$t # just the T values
    return(c(median(xstar), var(meds)))
}

boot_median <- boot(thetas$theta, median_nested, R = B)
boot.ci(boot_median, type = "basic")
```

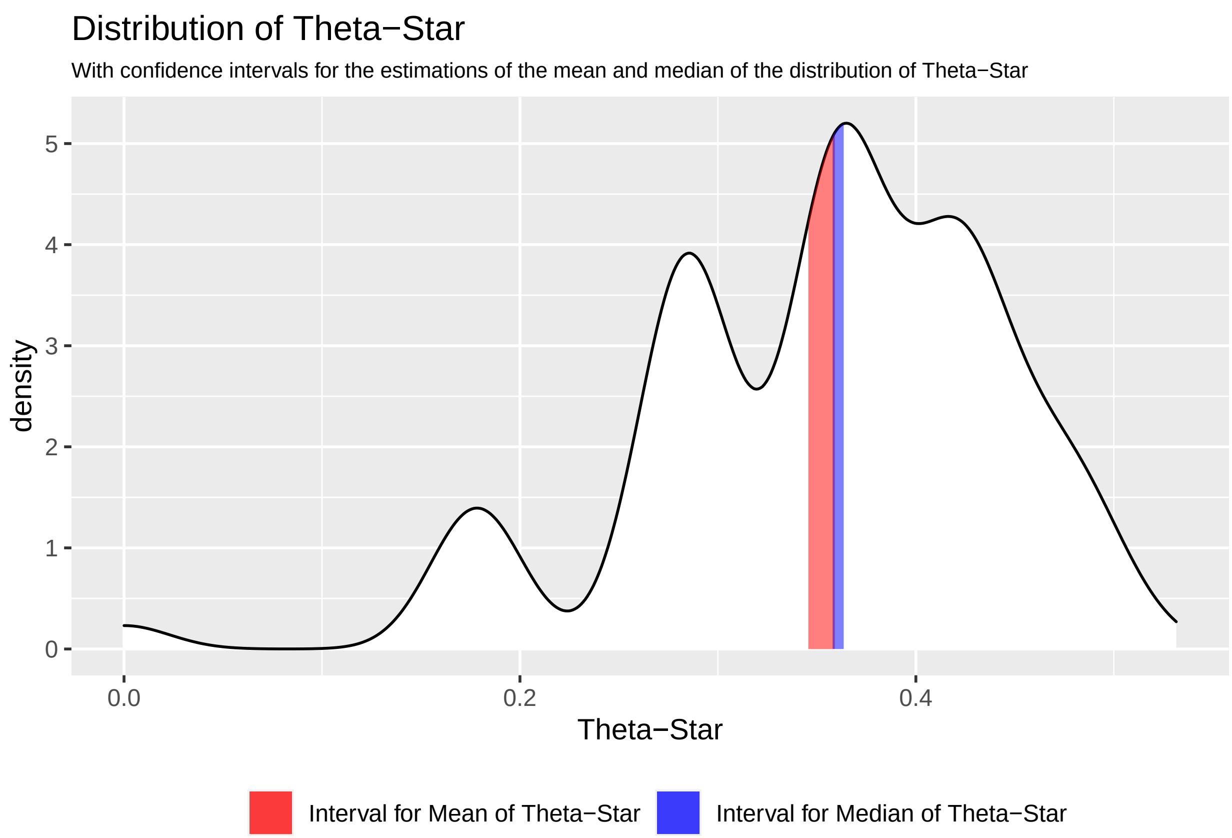
```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_median, type = "basic")
##
## Intervals :
## Level      Basic
## 95%   ( 0.3578,  0.3641 )
## Calculations and Intervals on Original Scale
```

Therefore, a 95% confidence interval for the median value of the distribution of θ^* is (0.3578, 0.3641).

```r
# Plotting the mean and median confidence intervals with the distribution of theta-star
theta_densities <- tibble(x = density(thetas$theta, n = 600)$x,
                          y = density(thetas$theta, n = 600)$y) %>%
    mutate(theta = thetas$theta,
           `Mean Interval` =
               x > boot.ci(boot_mean, type = "basic")$basic[4] &
               x < boot.ci(boot_mean, type = "basic")$basic[5],
           `Median Interval` =
               x > boot.ci(boot_median, type = "basic")$basic[4] &
               x < boot.ci(boot_median, type = "basic")$basic[5])

theta_densities_plt <-
    ggplot(theta_densities, aes(x = x, y = y)) +
    geom_density(aes(x = theta, y = ..density..), fill = "white") +
    geom_area(data = filter(theta_densities , `Mean Interval`),
              mapping = aes(fill = "Interval for Mean of Theta-Star"), alpha = 0.5) +
    geom_area(data = filter(theta_densities , `Median Interval`),
              mapping = aes(fill = "Interval for Median of Theta-Star"), alpha = 0.5) +
    scale_fill_manual("", values = c("Interval for Mean of Theta-Star" = "red",
                                     "Interval for Median of Theta-Star" = "blue")) +
    labs(x = "Theta-Star",
         y = "density",
         title = "Distribution of Theta-Star",
         subtitle =
             "With confidence intervals for the estimations of the mean and median of the distribution o
    theme(plot.subtitle = element_text(size = 8),
          legend.position = "bottom")

theta_densities_plt
```




```
# ggsave("theta_densities_plt", theta_densities_plt, device = "png", height = 4, width = 7)
```

```
# Same plot as above, except using the Greek letter for theta instead of "Theta"
# (Cannot knit this document with the Unicode "\u03B8", uncomment this code block to see/save plot)

# theta_densities_greek_letter_plt <-
#     ggplot(theta_densities, aes(x = x, y = y)) +
#     geom_density(aes(x = theta, y = ..density..), fill = "white") +
#     geom_area(data = filter(theta_densities , `Mean Interval`),
#               mapping = aes(fill = "Interval for Mean of \u03B8*"), alpha = 0.5) +
#     geom_area(data = filter(theta_densities , `Median Interval`),
#               mapping = aes(fill = "Interval for Median of \u03B8*"), alpha = 0.5) +
#     scale_fill_manual("", values = c("Interval for Mean of \u03B8*" = "red",
#                                      "Interval for Median of \u03B8*" = "blue")) +
#     labs(x = "\u03B8*",
#          y = "density",
#          title = "Distribution of \u03B8*",
#          subtitle =
#              "With confidence intervals for the estimations of the mean and median of the distributio
#     theme(plot.subtitle = element_text(size = 8),
#           legend.position = "bottom")
#
# theta_densities_greek_letter_plt
# # ggsave("theta_densities_greek_letter_plt", theta_densities_greek_letter_plt,
# #        device = "png", height = 4, width = 7)
```

```r
# Using the sample variance formula and the jackknife to estimate the variance of theta-star
var_hat <-
    sum(map_dbl(thetas$theta,
                function(theta) (theta - mean(thetas$theta)) ^ 2)) /
    (length(thetas$theta) - 1)
var_hat
```

```
## [1] 0.008402942
```

```r
# Using jackknife to estimate variance of var_hat
num_teams <- nrow(thetas)
tj <- map_dbl(1:num_teams, function(i) {
    var(thetas$theta[-i])
})
var_var_hat <- (num_teams - 1) / num_teams * sum((tj - mean(tj))^2)
var_var_hat
```

```
## [1] 4.275429e-07
```

```r
# Using var_hat and var_var_hat to construct a 95% confidence interval for the
# variance of the distribution of theta-star
var_hat_conf_int <- c(var_hat - 1.96 * sqrt(var_var_hat), var_hat + 1.96 * sqrt(var_var_hat))
var_hat_conf_int
```

```
## [1] 0.007121361 0.009684523
```

Therefore, a 95% confidence interval for the variance of the distribution of θ^* is (0.007121, 0.009685).