The [NBA playoffs](https://www.nba.com/playoffs#/) are in full swing! A total of 16 teams are competing in a playoff-format competition, with the winner of each best-of-7 series moving on to the next round. In each matchup, two teams play 7 basketball games against each other, and the team that wins more games progresses. Of course, we often don’t have to play all 7 games: we can determine the overall winner once either team reaches 4 wins.

During one of the games, a commentator made a remark along the lines of ***“you have no idea how hard it is for the better team to lose in a best-of-7 series”***. In this post, I’m going to try and quantify how hard it is to do so! I will do this not via analytical methods, but by [Monte Carlo simulation](https://en.wikipedia.org/wiki/Monte_Carlo_method). (You can see the code all at once [here](https://github.com/kjytay/misc/blob/master/blog/2019-04-22_best_of_n.R).)

Let’s imagine an idealized set-up, where for any given game, team A has probability pof beating team B. This probability is assumed to be the same from game to game, and the outcome of each game is assumed to be independent of the others. With these assumptions, the number of games that team A wins has a binomial distribution \text{Binom}(7, p).

In our simulation, for each p, we will generate a large number of \text{Binom}(7, p)random values and determine the proportion of them that are \geq 4: that would be our estimate for the winning probability of the 7-game series. ***How many random values should we draw?*** We should draw enough so that we are reasonably confident of the accuracy of the proportion estimate. If we draw Nsamples, then the plug-in estimate of standard error is \sqrt{\hat{p}(1 - \hat{p})/N}. In what follows, we will plot estimates with error bars indicating \pm 1standard error.

First, let’s set up a function that takes in three parameters: the number of simulation runs (simN), the total number of games in a series (n), and the probability that team A wins (p). The function returns the proportion of the simN runs which team A wins.

sim\_fn <- function(simN, n, p) {

if (n %% 2 == 0) {

stop("n should be an odd number")

}

mean(rbinom(simN, n, p) > n / 2)

}

Next, we set up a data frame with each row representing a different win probability for team A. Because of symmetry inherent in the problem, we focus on win probabilities which are \geq 0.5.

n <- 7

df <- data.frame(p = seq(0.5, 1, length.out = 26), n = n)

For a start, we set simN to be 100. For each row, we run simN simulation runs to get the win probability estimate and compute the associated standard error estimate. We also compute the upper and lower 1 standard error deviations from the probability estimate, to be used when plotting error bars.

set.seed(1043)

simN <- 100

df$win\_prob <- apply(df, 1, function(x) sim\_fn(simN, x[2], x[1]))

# compute std err & p\_hat \pm 1 std err

df$std\_err <- with(df, sqrt(win\_prob \* (1 - win\_prob) / simN))

df$lower <- with(df, win\_prob - std\_err)

df$upper <- with(df, win\_prob + std\_err)

Here is the code for the plot and the plot itself:

library(ggplot2)

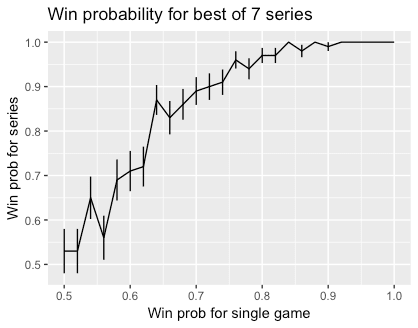
ggplot(df, aes(x = p, y = win\_prob)) +

geom\_line() +

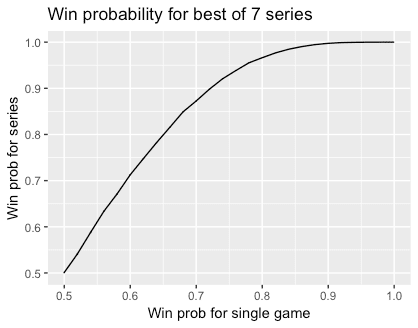
geom\_linerange(aes(ymin = lower, ymax = upper)) +

labs(title = paste("Win probability for best of", n, "series"),

x = "Win prob for single game", y = "Win prob for series")



We can see that the line is still quite wiggly with large error bars, suggesting that 100 simulation runs is not enough. (Indeed, we expect the graph to be monotonically increasing.) Below, we run 100,000 simulations for each value of p instead (with the same random seed):



We get a much smoother line, and the error bars are so small we can hardly see them. My conclusion here is that while it is harder for the weaker team to win a best-of-7 series that a single game, the odds are not insurmountable. For example, a team that has a 70% chance of winning any one game, which is a huge advantage, still has about a 13% chance of losing a best-of-7 series: not insignificant!

How do these probabilities change if we consider best-of-n series for different values of n? The code below is very similar to that above, except that our data frame contains data for n equal to all the odd numbers from 1 to 15, not just n = 7.

p <- seq(0.5, 1, length.out = 26)

n <- seq(1, 15, length.out = 8)

df <- expand.grid(p, n)

names(df) <- c("p", "n")

set.seed(422)

simN <- 100000

df$win\_prob <- apply(df, 1, function(x) sim\_fn(simN, x[2], x[1]))

# compute std err & p\_hat \pm 1 std err

df$std\_err <- with(df, sqrt(win\_prob \* (1 - win\_prob) / simN))

df$lower <- with(df, win\_prob - std\_err)

df$upper <- with(df, win\_prob + std\_err)

ggplot(df, aes(x = p, y = win\_prob, col = factor(n))) +

geom\_line() +

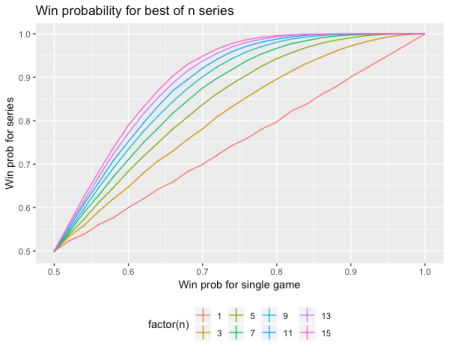
geom\_linerange(aes(ymin = lower, ymax = upper)) +

labs(title = paste("Win probability for best of n series"),

x = "Win prob for single game", y = "Win prob for series") +

theme(legend.position = "bottom")

Here is the plot:



As we expect, the series win probabilities increase as n increases. Also, the n = 1 graph corresponds to the y = xline. It looks like the win probabilities don’t change much from best-of-9 to best-of-15.