STAT 1341 - Sports Analytics

Final Project

Zach Palmer

Environment Setup and Packages

```
library(tidyverse)
library(dplyr)
library(knitr)
```

Reading in Data

I will be simulating the regular season and playoffs for the 2005-2006 NHL season. I start by reading in the .csv files that contain all of the games throughout recorded league history and the corresponding team information. I then split the file of games into two subsets, one that contains all the games throughout history before the 05-06 season, and another that contains only the games during the 05-06 regular season.

```
# read in the files and fix the unusual column names
scores = read.csv('data/nhl_scores.csv')
elos = read.csv('data/nhl_initial_elos.csv')
names(scores)[names(scores) == 'ï..id'] = 'id'
names(elos)[names(elos) == 'ï..team'] = 'team'

# filter for all games before the 05-06 season
prior_scores = scores %>%
  filter(season < 2005)

# filter for only those games that were part of the 05-06 regular season
current_scores = scores %>%
  filter(season == 2005 & game_type == 'r')

# read in the team statistics for 2005-06 season
team_stats = read.csv('data/team_season_stats.csv')
names(team_stats)[names(team_stats) == 'ï..id'] = 'id'
team_stats = team_stats %>%
  select(id, Team, AvAge, PTS, SRS, SOS, PP., PK., S, SA, S., SV.)
```

Calculating the Preseason Elo Ratings

The following code segment calculates each NHL teams' preseason Elo rating before the start of the 05-06 regular season. I also removed those teams within the "elos" data frame that did not participate in the 05-06 season.

```r
# from lecture 18
weight = 7.1
hfa = 51.63

# Iterate through all games in the sport's history
for(i in 1:nrow(prior_scores)) {
  # Find indices corresponding to home and away teams for current game
  home_index = which(elos$team == prior_scores$home_team[i])
  away_index = which(elos$team == prior_scores$away_team[i])

  # Find home and away team Elo ratings
  home_elo = elos$rating[home_index]
  away_elo = elos$rating[away_index]

  # Calculate home team win probability
  win_prob = 1 / (10 ^ ((away_elo - (home_elo + hfa * prior_scores$neutral[i]))/400) + 1)

  # Calculate actual margin of victory - must be positive
  score_diff = abs(prior_scores$home_score[i] - prior_scores$away_score[i])

  # Determine home team result
  if(prior_scores$home_score[i] > prior_scores$away_score[i]) {
    home_result = 1  # Home team wins
  } else if(prior_scores$home_score[i] < prior_scores$away_score[i]) {
    home_result = 0  # Home team loses
  } else {
    home_result = 0.5  # Tie
  }

  # Calculate amount each team's Elo rating is adjusted by
  home_elo_adjustment = weight * log(score_diff + 1) * (home_result - win_prob)

  # Adjust Elo ratings - add point to winner and subtract points from loser
  elos$rating[home_index] = elos$rating[home_index] + home_elo_adjustment
  elos$rating[away_index] = elos$rating[away_index] - home_elo_adjustment

  # Adjust Elo ratings at end of season to regress 1/3 of the way towards 1500
  if(i < nrow(prior_scores) && prior_scores$season[i + 1] > prior_scores$season[i]) {
    for(j in 1:nrow(elos)) {  # For each team
      if(prior_scores$season[i] >= elos$inaugural_season[j]) {  # Check if team existed
        # Move each team's Elo rating back towards 1500 by 1/3 of the difference
        elos$rating[j] = elos$rating[j] - (elos$rating[j] - 1500)/3
      }
    }

    # Identify all teams that existed at beginning of following season
    existing_teams = elos[which(elos$inaugural_season <= (prior_scores$season[i] + 1)),]

    # Calculate amount each team's Elo rating must be adjusted by to make mean 1500
    expansion_adjustment = -1 * (mean(existing_teams$rating) - 1500)

    # Perform expansion adjustment on teams that existed at beginning of following season
    for(j in 1:nrow(elos)) {  # For each team
```

```
      if((prior_scores$season[i] + 1) >= elos$inaugural_season[j]) {  # Check if team existed
        elos$rating[j] = elos$rating[j] + expansion_adjustment  # Update ratings if so
      }
    }
  }
}

elos = elos %>%
  filter(inaugural_season <= 2005) %>%
  select(-inaugural_season) %>%
  drop_na()
```

The divisions and conferences are up to date rather than correct at the time of the 2005-06 season so I will fix them here.

```
# team column
teams = c("New Jersey Devils", "Philadelphia Flyers", "New York Rangers", "New York Islanders",
          "Pittsburgh Penguins", "Ottawa Senators", "Buffalo Sabres", "Montreal Canadiens",
          "Toronto Maple Leafs", "Boston Bruins", "Carolina Hurricanes", "Tampa Bay Lightning",
          "Winnipeg Jets", "Florida Panthers", "Washington Capitals", "Detroit Red Wings",
          "Nashville Predators", "Columbus Blue Jackets", "Chicago Blackhawks",
          "St. Louis Blues", "Calgary Flames", "Colorado Avalanche", "Edmonton Oilers",
          "Vancouver Canucks", "Minnesota Wild", "Dallas Stars", "San Jose Sharks",
          "Anaheim Ducks", "Los Angeles Kings", "Phoenix Coyotes")

# conference column
conference = c("eastern", "eastern", "eastern", "eastern", "eastern", "eastern",
               "eastern", "eastern", "eastern", "eastern", "eastern", "eastern",
               "eastern", "eastern", "eastern", "western", "western", "western",
               "western", "western", "western", "western", "western", "western",
               "western", "western", "western", "western", "western", "western")

# division column
division = c("atlantic", "atlantic", "atlantic", "atlantic", "atlantic",
             "northeast", "northeast", "northeast", "northeast", "northeast",
             "southeast", "southeast", "southeast", "southeast", "southeast",
             "central", "central", "central", "central", "central",
             "northwest", "northwest", "northwest", "northwest", "northwest",
             "pacific", "pacific", "pacific", "pacific", "pacific")

# points column
points = c(101, 101, 100, 78, 58,
           113, 110, 93, 90, 74,
           112, 92, 90, 85, 70,
           124, 106, 74, 65, 57,
           103, 95, 95, 92, 84,
           112, 99, 98, 89, 81)

# overall data frame
team_divisions = data.frame(teams, conference, division, points)

for (i in 1:nrow(elos)) {
  if (elos[i, "team"] %in% teams) {
```

```
    elos[i, "division"] = team_divisions[which(team_divisions$teams == elos[i, "team"]),
                                         "division"]
    elos[i, "conference"] = team_divisions[which(team_divisions$teams == elos[i, "team"]),
                                           "conference"]
    elos[i, "points"] = team_divisions[which(team_divisions$teams == elos[i, "team"]),
                                       "points"]
  }
}
```

Simulating the Regular Season and Playoffs

Using the altered code from Homework 11, I ran 10000 iterations of a simulation of the 05-06 NHL regular season and playoffs

```
# setup
set.seed(2005)
simulated_season = 2005
iterations = 10000

# Grab the Elo rating, conference, and division for the teams
team_info = elos

# Obtain list of unique conference names and unique division names
conferences = na.omit(unique(team_info$conference))
divisions = na.omit(unique(team_info$division))

# Grab list of regular season games for season being simulated
season_schedule = current_scores

# set up the team_info and summary data frames
team_info = team_info[which(team_info$conference != 'NA'),]
summary = data.frame(matrix(0, ncol = 6, nrow = nrow(team_info)))
colnames(summary) = c("team", "average_points", "playoffs", "div_titles",
                      "conf_champs", "championships")
summary$team = team_info$team

histories = data.frame(matrix(0, ncol = nrow(team_info), nrow = iterations))
colnames(histories) = team_info$team

# first team to dive deeper into
first_team = "Los Angeles Kings"

# Create data frame to store information for team specified above
first_results = data.frame(matrix(ncol = 8, nrow = 0))
colnames(first_results) = c("opponent", "pregame_elo", "win_probability", "result",
                            "team_score", "opponent_score", "elo_adjustment", "postgame_elo")

# second team to dive deeper into
second_team = "St. Louis Blues"

# Create data frame to store information for team specified above
second_results = data.frame(matrix(ncol = 8, nrow = 0))
colnames(second_results) = c("opponent", "pregame_elo", "win_probability", "result",
```

```
                          "team_score", "opponent_score", "elo_adjustment", "postgame_elo")

# third team to dive deeper into
third_team = "Carolina Hurricanes"

# Create data frame to store information for team specified above
third_results = data.frame(matrix(ncol = 8, nrow = 0))
colnames(third_results) = c("opponent", "pregame_elo", "win_probability", "result",
                          "team_score", "opponent_score", "elo_adjustment", "postgame_elo")

count = 0
for (i in 1:iterations) {
  season_stats = team_info[,which(colnames(team_info) != "inaugural_season")]
  season_stats$points = 0
  season_stats$rand = runif(nrow(team_info))

  for (j in 1:nrow(season_schedule)) {
    # Find indices corresponding to home and away teams for current game
    home_index = which(season_stats$team == season_schedule$home_team[j])
    away_index = which(season_stats$team == season_schedule$away_team[j])

    # Find home and away team Elo ratings
    home_elo = season_stats$rating[home_index]
    away_elo = season_stats$rating[away_index]

    # Calculate home team win and tie probabilities
    tie_prob = (1/(sqrt(4 * pi))) * exp(-((away_elo - (home_elo + hfa *
                season_schedule$neutral[j])) ^ 2/160000))
    win_prob = 1/(10 ^ ((away_elo -
              (home_elo + hfa * season_schedule$neutral[j]))/400) + 1) - 0.50 * tie_prob
    u = runif(1)

    if (u < win_prob) {  # Home team wins in regulation
      season_stats$points[home_index] = season_stats$points[home_index] + 2
    } else if (u < win_prob + 0.50 * tie_prob) {  # Home team wins in OT/shootout
      season_stats$points[home_index] = season_stats$points[home_index] + 2
      season_stats$points[away_index] = season_stats$points[away_index] + 1
    } else if (u > win_prob + tie_prob) {  # Away team wins in regulation
      season_stats$points[away_index] = season_stats$points[away_index] + 2
    } else { # Away team wins in OT/shootout
      season_stats$points[home_index] = season_stats$points[home_index] + 1
      season_stats$points[away_index] = season_stats$points[away_index] + 2
    }

    # Calculate actual margin of victory - must be positive
    score_diff = abs(season_schedule$home_score[j] - season_schedule$away_score[j])

    # Determine home team result
    if (season_schedule$home_score[j] > season_schedule$away_score[j]) {
      home_result = 1  # Home team wins
    } else if (season_schedule$home_score[j] < season_schedule$away_score[j]) {
      home_result = 0  # Home team loses
    } else {
```

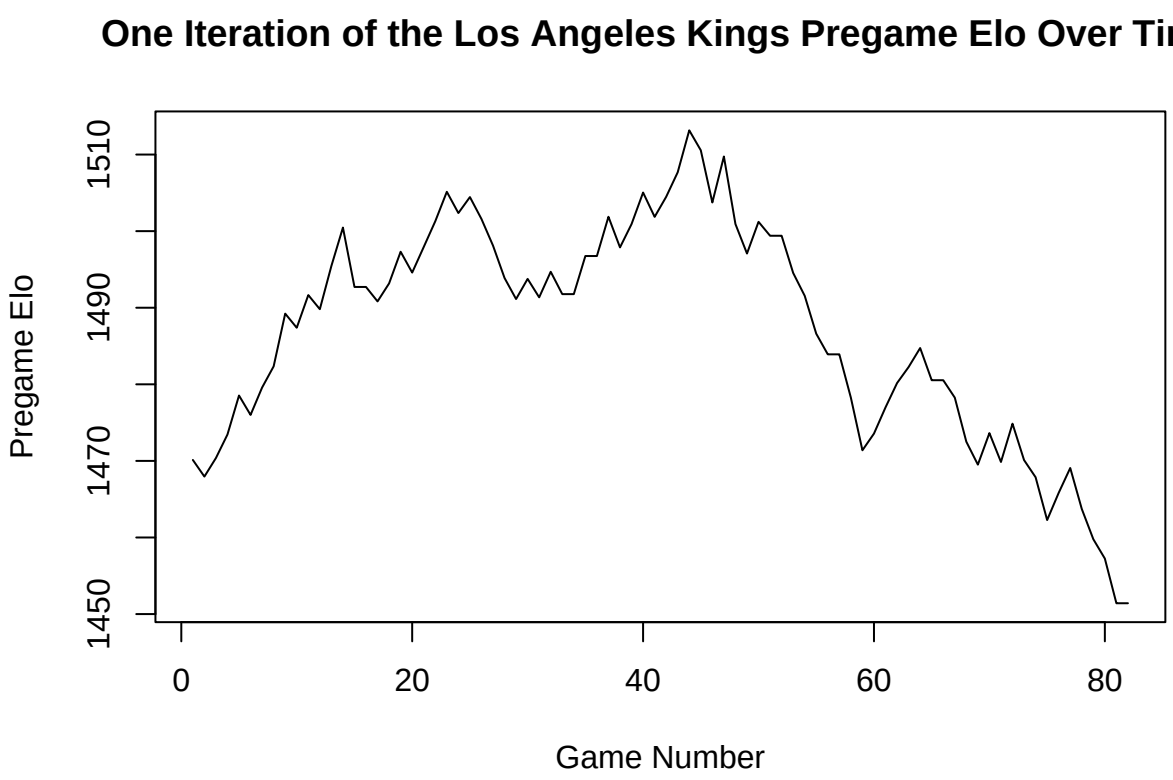
```
    home_result = 0.5  # Tie
  }

  # Calculate amount each team's Elo rating is adjusted by
  home_elo_adjustment = weight * log(score_diff + 1) * (home_result - win_prob)

  # Adjust Elo ratings after game has been simulated to get team's new strength
  season_stats$rating[home_index] = season_stats$rating[home_index] + home_elo_adjustment
  season_stats$rating[away_index] = season_stats$rating[away_index] - home_elo_adjustment

  if (i == 5) {
    count = count + 1
    # Add game information to team result data frame if the first team specified played
    if(season_schedule$home_team[j] == first_team | season_schedule$away_team[j] == first_team) {
      if(season_schedule$home_team[j] == first_team) {  # If specified team was at home
        first_results[nrow(first_results) + 1,] =
          c(season_schedule$away_team[j], season_stats$rating[home_index] - home_elo_adjustment,
            win_prob, home_result, season_schedule$home_score[j], season_schedule$away_score[j],
            home_elo_adjustment, season_stats$rating[home_index])
      } else {  # If specified team was away
        first_results[nrow(first_results) + 1,] =
          c(season_schedule$home_team[j], season_stats$rating[away_index] + home_elo_adjustment,
            1 - win_prob, 1 - home_result, season_schedule$away_score[j],
            season_schedule$home_score[j], -1 * home_elo_adjustment,
            season_stats$rating[away_index])
      }
    }

    # Add game information to team result data frame if the second team specified played
    if(season_schedule$home_team[j] == second_team | season_schedule$away_team[j] == second_team) {
      if(season_schedule$home_team[j] == second_team) {  # If specified team was at home
        second_results[nrow(second_results) + 1,] =
          c(season_schedule$away_team[j], season_stats$rating[home_index] - home_elo_adjustment,
            win_prob, home_result, season_schedule$home_score[j], season_schedule$away_score[j],
            home_elo_adjustment, season_stats$rating[home_index])
      } else {  # If specified team was away
        second_results[nrow(second_results) + 1,] =
          c(season_schedule$home_team[j], season_stats$rating[away_index] + home_elo_adjustment,
            1 - win_prob, 1 - home_result, season_schedule$away_score[j],
            season_schedule$home_score[j], -1 * home_elo_adjustment,
            season_stats$rating[away_index])
      }
    }

    # Add game information to team result data frame if the third team specified played
    if(season_schedule$home_team[j] == third_team | season_schedule$away_team[j] == third_team) {
      if(season_schedule$home_team[j] == third_team) {  # If specified team was at home
        third_results[nrow(third_results) + 1,] =
          c(season_schedule$away_team[j], season_stats$rating[home_index] - home_elo_adjustment,
            win_prob, home_result, season_schedule$home_score[j], season_schedule$away_score[j],
            home_elo_adjustment, season_stats$rating[home_index])
      } else {  # If specified team was away
        third_results[nrow(third_results) + 1,] =
```

```
          c(season_schedule$home_team[j], season_stats$rating[away_index] + home_elo_adjustment,
            1 - win_prob, 1 - home_result, season_schedule$away_score[j],
            season_schedule$home_score[j], -1 * home_elo_adjustment,
            season_stats$rating[away_index])
      }
    }
  }
}

summary$average_points = summary$average_points + season_stats$points

division_winners = data.frame(matrix(ncol = 6, nrow = 0))
colnames(division_winners) = c("team", "conference", "division", "rating", "points", "rand")

non_division_winners = data.frame(matrix(ncol = 6, nrow = 0))
colnames(non_division_winners) = c("team", "conference", "division", "rating", "points", "rand")

num_wild_cards = 5
wild_card_teams = data.frame(matrix(ncol = 6, nrow = 0))
colnames(wild_card_teams) = c("team", "conference", "division", "rating", "points", "rand")

for (div in divisions) {
  div_standings = season_stats[which(season_stats$division == div),]
  div_standings = div_standings[order(-div_standings$points, -div_standings$rand),]
  division_winners = rbind(division_winners, div_standings[1,])
  non_division_winners = rbind(non_division_winners, div_standings[2:nrow(div_standings),])
}

for (conf in conferences) {
  wc_standings = non_division_winners[which(non_division_winners$conference == conf),]
  wc_standings = wc_standings[order(-wc_standings$points, -wc_standings$rand),]
  wild_card_teams = rbind(wild_card_teams, wc_standings[1:num_wild_cards,])
}

division_winners = division_winners[order(division_winners$conference, -division_winners$points,
                                          -division_winners$rand),]

wild_card_teams = wild_card_teams[order(wild_card_teams$conference, -wild_card_teams$points,
                                        -wild_card_teams$rand),]

for (j in 1:nrow(division_winners)) {
  index = which(season_stats$team == division_winners$team[j])
  summary$playoffs[index] = summary$playoffs[index] + 1
  summary$div_titles[index] = summary$div_titles[index] + 1
}

for (team in wild_card_teams$team) {
  index = which(season_stats$team == team)
  summary$playoffs[index] = summary$playoffs[index] + 1
}

games_per_round = c(7, 7, 7, 7)
```

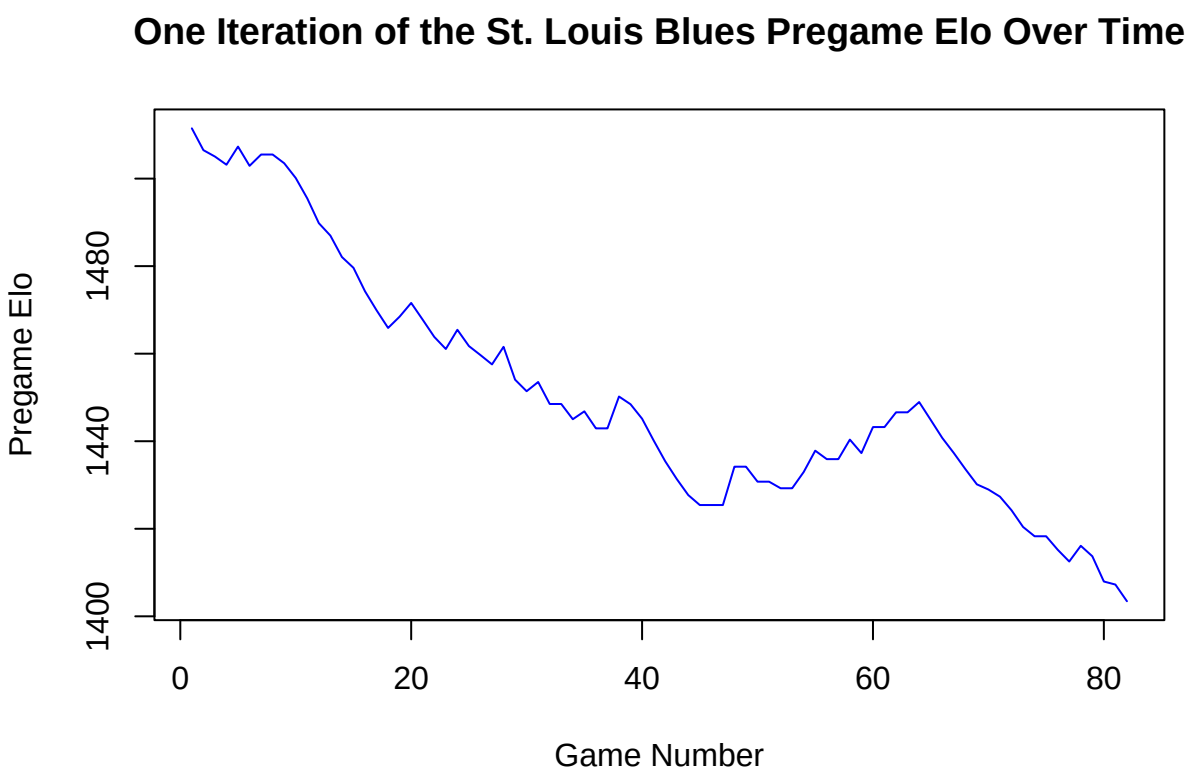
```
playoff_bracket = data.frame(matrix(-Inf, ncol = 6, nrow = 16))
colnames(playoff_bracket) = c("team", "conference", "division", "rating", "points", "rand")
next_round = NULL

#NHL
playoff_bracket[1,] = division_winners[1,]
playoff_bracket[2,] = division_winners[2,]
playoff_bracket[3,] = division_winners[3,]
playoff_bracket[4,] = wild_card_teams[1,]
playoff_bracket[5,] = wild_card_teams[2,]
playoff_bracket[6,] = wild_card_teams[3,]
playoff_bracket[7,] = wild_card_teams[4,]
playoff_bracket[8,] = wild_card_teams[5,]
playoff_bracket[9,] = division_winners[4,]
playoff_bracket[10,] = division_winners[5,]
playoff_bracket[11,] = division_winners[6,]
playoff_bracket[12,] = wild_card_teams[6,]
playoff_bracket[13,] = wild_card_teams[7,]
playoff_bracket[14,] = wild_card_teams[8,]
playoff_bracket[15,] = wild_card_teams[9,]
playoff_bracket[16,] = wild_card_teams[10,]

playoff_bracket$seed = rep(1:8, 2)

# Simulate every round in the playoffs until the championship game/round
for (round in 1:(length(games_per_round) - 1)) {
  for (j in 1:2) {  # Divide 'playoff_bracket' into two halves, separated by conference
    for(k in 1:(nrow(playoff_bracket)/4)) {  # Match 1 seed with 8 seed, 2 seed with 7 seed, etc.
      high_seed_index = 0.5 * nrow(playoff_bracket) * j - (0.5 * nrow(playoff_bracket) - k)
      low_seed_index = 0.5 * nrow(playoff_bracket) * j - (k - 1)

      # Obtain Elo ratings for high and low seeds
      high_seed_elo = playoff_bracket$rating[high_seed_index]
      low_seed_elo = playoff_bracket$rating[low_seed_index]

      # Calculate win probability for each team when they play at home against their playoff opponent
      high_seed_home_win_prob = 1 / (10 ^ ((low_seed_elo - (high_seed_elo + hfa))/400) + 1)
      low_seed_home_win_prob = 1 / (10 ^ ((high_seed_elo - (low_seed_elo + hfa))/400) + 1)

      # Create array of win probabilities where high seed gets 1 more home game than low seed
      win_probs = c(rep(high_seed_home_win_prob, ceiling(games_per_round[round]/2)),
                    1 - rep(low_seed_home_win_prob, floor(games_per_round[round]/2)))

      # Generate random numbers for each game in the round
      u = runif(games_per_round[round])
      # Calculate proportion of games won by higher seed
      high_seed_wins = sum(u < win_probs)/games_per_round[round]

      if(high_seed_wins > 0.50) {  # If high seed won more than 50% of games in series
        # Advance high seed to next round
        next_round = rbind(next_round, playoff_bracket[high_seed_index,])
      } else{  # If low seed won more than 50% of games in series
        # Advance low seed to next round
```

```
          next_round = rbind(next_round, playoff_bracket[low_seed_index,])
        }
      }
    }

    playoff_bracket = next_round  # Reset playoff bracket to consist of all remaining teams

    # Reseed for next round
    playoff_bracket = playoff_bracket[order(playoff_bracket$conference, playoff_bracket$seed),]
    next_round = NULL  # Reset list of teams in subsequent round to an empty data frame
  }


  # Stanley Cup Finals
  playoff_bracket = playoff_bracket[order(-playoff_bracket$points, -playoff_bracket$rand),]

  high_seed_elo = playoff_bracket$rating[1]
  low_seed_elo = playoff_bracket$rating[2]
  high_seed_home_win_prob = 1/(10 ^ ((low_seed_elo - (high_seed_elo + hfa))/400) + 1)
  low_seed_home_win_prob = 1/(10 ^ ((high_seed_elo - (low_seed_elo + hfa))/400) + 1)
  win_probs = c(rep(high_seed_home_win_prob, ceiling(games_per_round[length(games_per_round)]/2)),
                1 - rep(low_seed_home_win_prob, floor(games_per_round[length(games_per_round)]/2)))
  u = runif(games_per_round[4])
  high_seed_wins = sum(u < win_probs)/games_per_round[4]

  if (high_seed_wins > 0.50) {
    champion = playoff_bracket[1,]
  } else {
    champion = playoff_bracket[2,]
  }

  for(team in playoff_bracket$team) {
    index = which(season_stats$team == team)
    summary$conf_champs[index] = summary$conf_champs[index] + 1
  }

  index = which(season_stats$team == champion$team)
  summary$championships[index] = summary$championships[index] + 1
  histories[i,] = season_stats$points
}

summary$average_points = summary$average_points/iterations
```

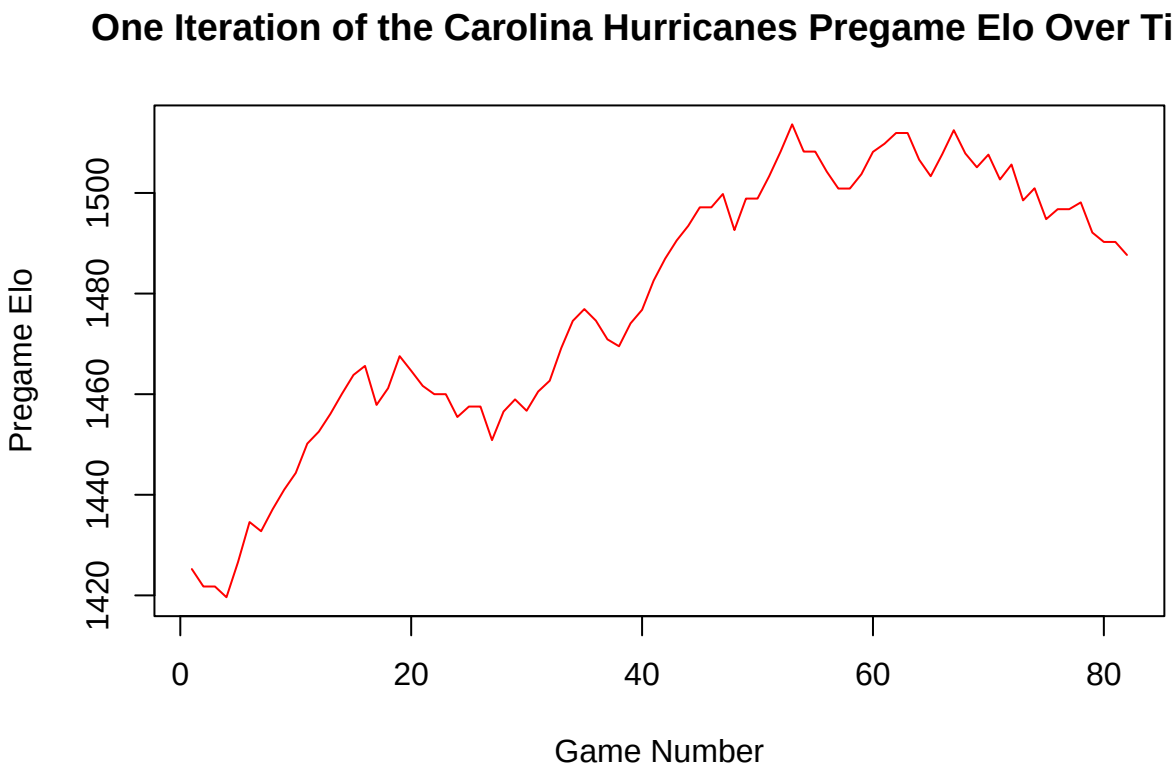
```
plot(first_results$pregame_elo, xlab = "Game Number", ylab = "Pregame Elo",
     main = "One Iteration of the Los Angeles Kings Pregame Elo Over Time", type = "l")
```




```
plot(second_results$pregame_elo, xlab = "Game Number", ylab = "Pregame Elo",
     main = "One Iteration of the St. Louis Blues Pregame Elo Over Time", type = "l", col = "blue")
```




```
plot(third_results$pregame_elo, xlab = "Game Number", ylab = "Pregame Elo",
     main = "One Iteration of the Carolina Hurricanes Pregame Elo Over Time", type = "l", col = "red")
```



Elo Rating by Division

Swap the Winnipeg Jets to the Atlanta Thrashers

```
# Switch the Winnipeg Jets to the Atlanta Thrashers
elos[which(elos$team == "Winnipeg Jets"), "team"] = "Atlanta Thrashers"
summary[which(summary$team == "Winnipeg Jets"), "team"] = "Atlanta Thrashers"
```

```
atlantic = elos %>%
  filter(division == "atlantic")

northeast = elos %>%
  filter(division == "northeast")

southeast = elos %>%
  filter(division == "southeast")

central = elos %>%
  filter(division == "central")

northwest = elos %>%
  filter(division == "northwest")

pacific = elos %>%
  filter(division == "pacific")
```

```
kable(atlantic[, c(1, 4)], caption = "Atlantic Division Preseason Elos")
```

Table 1: Atlantic Division Preseason Elos

team	rating
New York Rangers	1427.300
Pittsburgh Penguins	1371.019
Philadelphia Flyers	1569.541
New York Islanders	1514.499
New Jersey Devils	1564.851

```
kable(northeast[, c(1, 4)], caption = "Northeast Division Preseason Elos")
```

Table 2: Northeast Division Preseason Elos

team	rating
Toronto Maple Leafs	1559.750
Montreal Canadiens	1503.134
Boston Bruins	1523.926
Buffalo Sabres	1491.467
Ottawa Senators	1587.582

```
kable(southeast[, c(1, 4)], caption = "Southeast Division Preseason Elos")
```

Table 3: Southeast Division Preseason Elos

team	rating
Tampa Bay Lightning	1572.033
Florida Panthers	1420.979
Washington Capitals	1416.039
Carolina Hurricanes	1425.234
Atlanta Thrashers	1436.471

```
kable(central[, c(1, 4)], caption = "Central Division Preseason Elos")
```

Table 4: Central Division Preseason Elos

team	rating
Detroit Red Wings	1598.140
Columbus Blue Jackets	1413.183
Chicago Blackhawks	1403.457
St. Louis Blues	1511.479
Nashville Predators	1493.094

```
kable(northwest[, c(1, 4)], caption = "Northwest Division Preseason Elos")
```

Table 5: Northwest Division Preseason Elos

team	rating
Colorado Avalanche	1562.503
Minnesota Wild	1507.042
Vancouver Canucks	1559.794
Calgary Flames	1544.203
Edmonton Oilers	1527.284

```
kable(pacific[, c(1, 4)], caption = "Pacific Division Preseason Elos")
```

Table 6: Pacific Division Preseason Elos

team	rating
Dallas Stars	1553.125
Los Angeles Kings	1470.126
Arizona Coyotes	1421.896
San Jose Sharks	1553.809
Anaheim Ducks	1479.362

Comparison of Actual and Simulated Points

```
atlantic_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "atlantic") %>%
  mutate(residual = points - average_points) %>%
  select(team, average_points, points, residual, rating)
```

```
## Joining, by = "team"
```

```
northeast_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "northeast") %>%
  mutate(residual = points - average_points) %>%
  select(team, average_points, points, residual, rating)
```

```
## Joining, by = "team"
```

```
southeast_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "southeast") %>%
  mutate(residual = points - average_points) %>%
  select(team, average_points, points, residual, rating)
```

```
## Joining, by = "team"
```

Table 7: Atlantic Division Simulation Results

team	average_points	points	residual	rating
New York Rangers	88.4717	100	11.5283	1427.300
Pittsburgh Penguins	59.1479	58	-1.1479	1371.019
Philadelphia Flyers	109.6533	101	-8.6533	1569.541
New York Islanders	92.1233	78	-14.1233	1514.499
New Jersey Devils	103.2777	101	-2.2777	1564.851

```
central_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "central") %>%
  mutate(residual = points - average_points) %>%
  select(team, average_points, points, residual, rating)
```

```
## Joining, by = "team"
```

```
northwest_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "northwest") %>%
  mutate(residual = points - average_points) %>%
  select(team, average_points, points, residual, rating)
```

```
## Joining, by = "team"
```

```
pacific_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "pacific") %>%
  mutate(residual = points - average_points) %>%
  select(team, average_points, points, residual, rating)
```

```
## Joining, by = "team"
```

```
kable(atlantic_summary, caption = "Atlantic Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(northeast_summary, caption = "Northeast Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(southeast_summary, caption = "Southeast Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(central_summary, caption = "Central Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

Table 8: Northeast Division Simulation Results

team	average_points	points	residual	rating
Toronto Maple Leafs	101.8721	90	-11.8721	1559.750
Montreal Canadiens	91.5357	93	1.4643	1503.134
Boston Bruins	91.3294	74	-17.3294	1523.926
Buffalo Sabres	97.5805	110	12.4195	1491.467
Ottawa Senators	120.2373	113	-7.2373	1587.582

Table 9: Southeast Division Simulation Results

team	average_points	points	residual	rating
Tampa Bay Lightning	107.5760	92	-15.5760	1572.033
Florida Panthers	77.9588	85	7.0412	1420.979
Washington Capitals	69.1338	70	0.8662	1416.039
Carolina Hurricanes	91.3078	112	20.6922	1425.234
Atlanta Thrashers	84.3082	90	5.6918	1436.471

Table 10: Central Division Simulation Results

team	average_points	points	residual	rating
Detroit Red Wings	120.5633	124	3.4367	1598.140
Columbus Blue Jackets	66.8851	74	7.1149	1413.183
Chicago Blackhawks	68.0875	65	-3.0875	1403.457
St. Louis Blues	82.0917	57	-25.0917	1511.479
Nashville Predators	95.8771	106	10.1229	1493.094

Table 11: Northwest Division Simulation Results

team	average_points	points	residual	rating
Colorado Avalanche	106.4404	95	-11.4404	1562.503
Minnesota Wild	94.0760	84	-10.0760	1507.042
Vancouver Canucks	103.1826	92	-11.1826	1559.794
Calgary Flames	100.9661	103	2.0339	1544.203
Edmonton Oilers	95.4361	95	-0.4361	1527.284

Table 12: Pacific Division Simulation Results

team	average_points	points	residual	rating
Dallas Stars	106.6797	112	5.3203	1553.125
Los Angeles Kings	87.9840	89	1.0160	1470.126
Arizona Coyotes	76.5776	70	-6.5776	1421.896
San Jose Sharks	101.8775	99	-2.8775	1553.809
Anaheim Ducks	91.0251	98	6.9749	1479.362

```
kable(northwest_summary, caption = "Northwest Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(pacific_summary, caption = "Pacific Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

Simulated Playoff Results

```
atlantic_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "atlantic") %>%
  select(-average_points, -points, -conference, -division)
```

```
## Joining, by = "team"
```

```
northeast_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "northeast") %>%
  select(-average_points, -points, -conference, -division)
```

```
## Joining, by = "team"
```

```
southeast_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "southeast") %>%
  select(-average_points, -points, -conference, -division)
```

Table 13: Atlantic Division Simulation Results

team	playoffs	div_titles	conf_champs	championships	rating
New York Rangers	3313	137	100	13	1427.300
Pittsburgh Penguins	2	0	0	0	1371.019
Philadelphia Flyers	9848	6849	1172	355	1569.541
New York Islanders	5056	328	45	6	1514.499
New Jersey Devils	9235	2686	1151	391	1564.851

```
## Joining, by = "team"
```

```
central_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "central") %>%
  select(-average_points, -points, -conference, -division)
```

```
## Joining, by = "team"
```

```
northwest_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "northwest") %>%
  select(-average_points, -points, -conference, -division)
```

```
## Joining, by = "team"
```

```
pacific_summary = summary %>%
  inner_join(elos) %>%
  filter(division == "pacific") %>%
  select(-average_points, -points, -conference, -division)
```

```
## Joining, by = "team"
```

```
kable(atlantic_summary, caption = "Atlantic Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(northeast_summary, caption = "Northeast Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(southeast_summary, caption = "Southeast Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(central_summary, caption = "Central Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

Table 14: Northeast Division Simulation Results

team	playoffs	div_titles	conf_champs	championships	rating
Toronto Maple Leafs	8927	469	719	240	1559.750
Montreal Canadiens	4757	41	210	50	1503.134
Boston Bruins	4701	32	70	13	1523.926
Buffalo Sabres	7578	197	1141	441	1491.467
Ottawa Senators	10000	9261	4527	2497	1587.582

Table 15: Southeast Division Simulation Results

team	playoffs	div_titles	conf_champs	championships	rating
Tampa Bay Lightning	9737	9048	695	194	1572.033
Florida Panthers	467	24	2	0	1420.979
Washington Capitals	42	0	0	0	1416.039
Carolina Hurricanes	4668	768	121	25	1425.234
Atlanta Thrashers	1669	160	47	11	1436.471

Table 16: Central Division Simulation Results

team	playoffs	div_titles	conf_champs	championships	rating
Detroit Red Wings	9996	9849	4891	3455	1598.140
Columbus Blue Jackets	6	0	0	0	1413.183
Chicago Blackhawks	19	0	0	0	1403.457
St. Louis Blues	806	2	0	0	1511.479
Nashville Predators	6271	149	234	87	1493.094

Table 17: Northwest Division Simulation Results

team	playoffs	div_titles	conf_champs	championships	rating
Colorado Avalanche	9431	4473	971	453	1562.503
Minnesota Wild	5354	479	165	55	1507.042
Vancouver Canucks	8886	2589	364	150	1559.794
Calgary Flames	8287	1844	903	438	1544.203
Edmonton Oilers	6125	615	264	91	1527.284

Table 18: Pacific Division Simulation Results

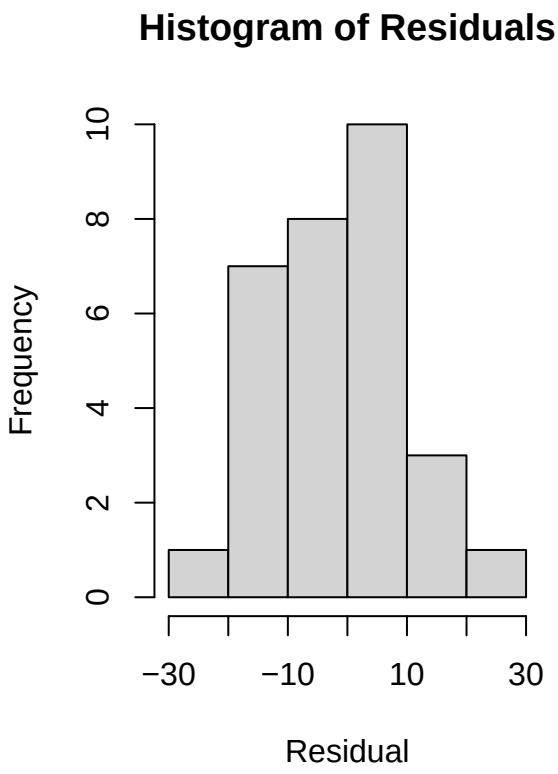
team	playoffs	div_titles	conf_champs	championships	rating
Dallas Stars	9513	6289	1095	531	1553.125
Los Angeles Kings	2570	210	16	4	1470.126
Arizona Coyotes	280	10	1	0	1421.896
San Jose Sharks	8573	3071	939	437	1553.809
Anaheim Ducks	3883	420	157	63	1479.362

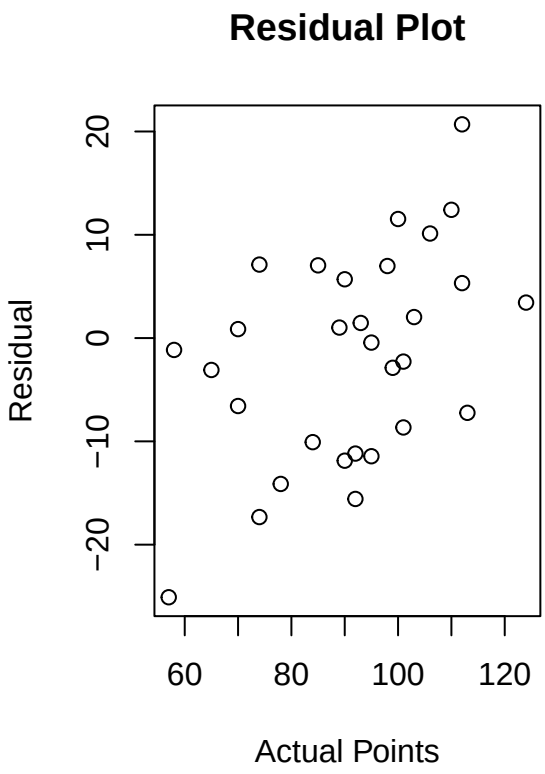
```
kable(northwest_summary, caption = "Northwest Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(pacific_summary, caption = "Pacific Division Simulation Results",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

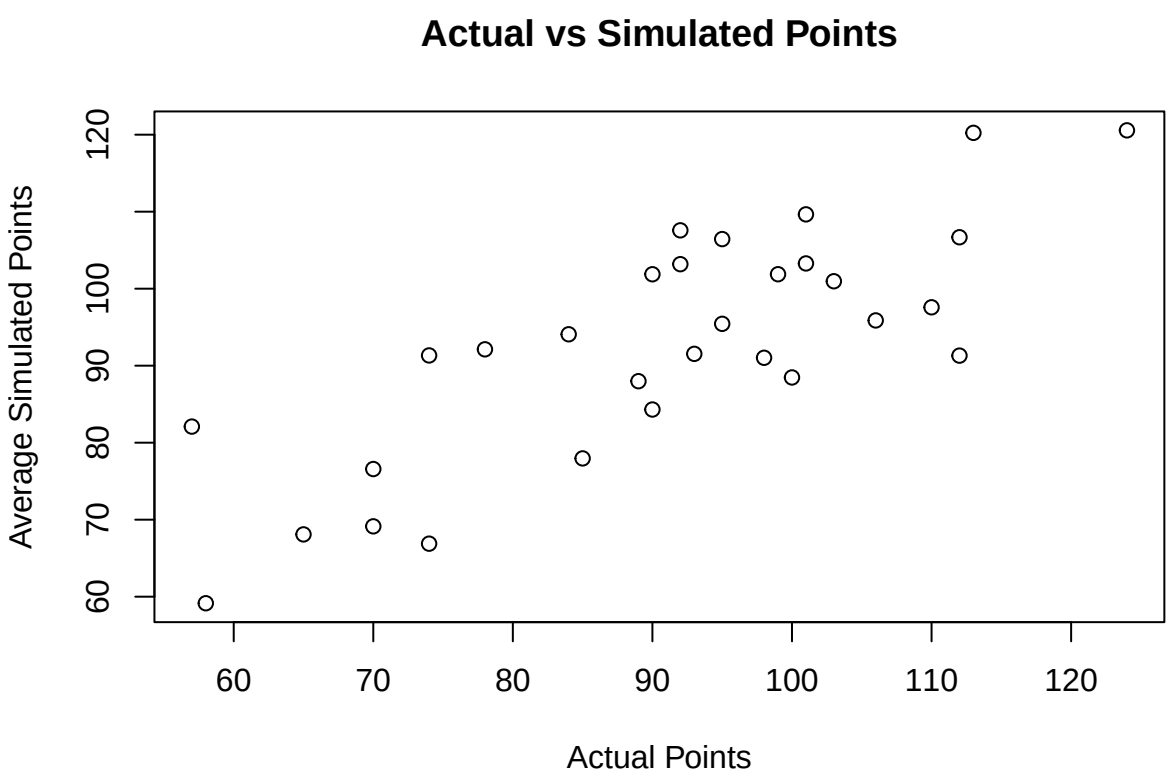
Comparison Graphics

```
summary = summary %>%
  mutate(actual_pts = elos$points,
         pts_diff = actual_pts - average_points)

par(mfrow = c(1, 2))
hist(summary$pts_diff, xlab = "Residual", main = "Histogram of Residuals")
plot(summary$actual_pts, summary$pts_diff, xlab = "Actual Points", ylab = "Residual",
     main = "Residual Plot")
```






```
plot(summary$actual_pts, summary$average_points, xlab = "Actual Points",
     ylab = "Average Simulated Points", main = "Actual vs Simulated Points")
```




```
cor(summary$actual_pts, summary$average_points)
```

```
## [1] 0.8018191
```

Actual Team Season Statistics

```
team_stats = team_stats %>%
  mutate(S.SA = S/(S + SA),
         PDO = (SV. * 100) + S.) %>%
  select(id, Team, PTS, PP., PK., S.SA, PDO)
```

```
league_avg = team_stats %>%
  summarise(avg.pts = sum(PTS)/n(),
            avg.PP = sum(PP.)/n(),
            avg.PK = sum(PK.)/n(),
            avg.S.SA = sum(S.SA)/n(),
            avg.PDO = sum(PDO)/n())

kable(league_avg, caption = "Actual League Average Statistics for the Regular Season",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

Table 19: Actual League Average Statistics for the Regular Season

avg.pts	avg.PP	avg.PK	avg.S.SA	avg.PDO
91.36667	17.642	82.35267	0.5003432	100

Table 20: Season Statistics for the Carolina Hurricanes

Team	PP.	PK.	S.SA	PDO	diff.PP	diff.PK	diff.S.SA	diff.PDO
Carolina Hurricanes	17.89	81.8	0.5055446	100.9	0.248	-0.5526667	0.0052014	0.9

```
comparison_stats = team_stats %>%
  filter(Team == first_team | Team == second_team | Team == third_team) %>%
  mutate(diff.PP = PP. - league_avg$avg.PP,
         diff.PK = PK. - league_avg$avg.PK,
         diff.S.SA = S.SA - league_avg$avg.S.SA,
         diff.PDO = PDO - league_avg$avg.PDO) %>%
  select(-id, -PTS)

comparison_stats
```

```
##                  Team   PP.   PK.      S.SA   PDO diff.PP    diff.PK
## 1 Carolina Hurricanes 17.89 81.80 0.5055446 100.9   0.248 -0.5526667
## 2   Los Angeles Kings 14.23 78.73 0.4933031  99.3  -3.412 -3.6226667
## 3     St. Louis Blues 14.65 82.21 0.4819401  97.0  -2.992 -0.1426667
##      diff.S.SA diff.PDO
## 1  0.005201381      0.9
## 2 -0.007040062     -0.7
## 3 -0.018403029     -3.0
```

```
kable(comparison_stats[1, ], caption = "Season Statistics for the Carolina Hurricanes",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(comparison_stats[2, ], caption = "Season Statistics for the Los Angeles Kings",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

```
kable(comparison_stats[3, ], caption = "Season Statistics for the St. Louis Blues",
      format = "latex", booktabs = TRUE, longtable = FALSE)
```

Table 21: Season Statistics for the Los Angeles Kings

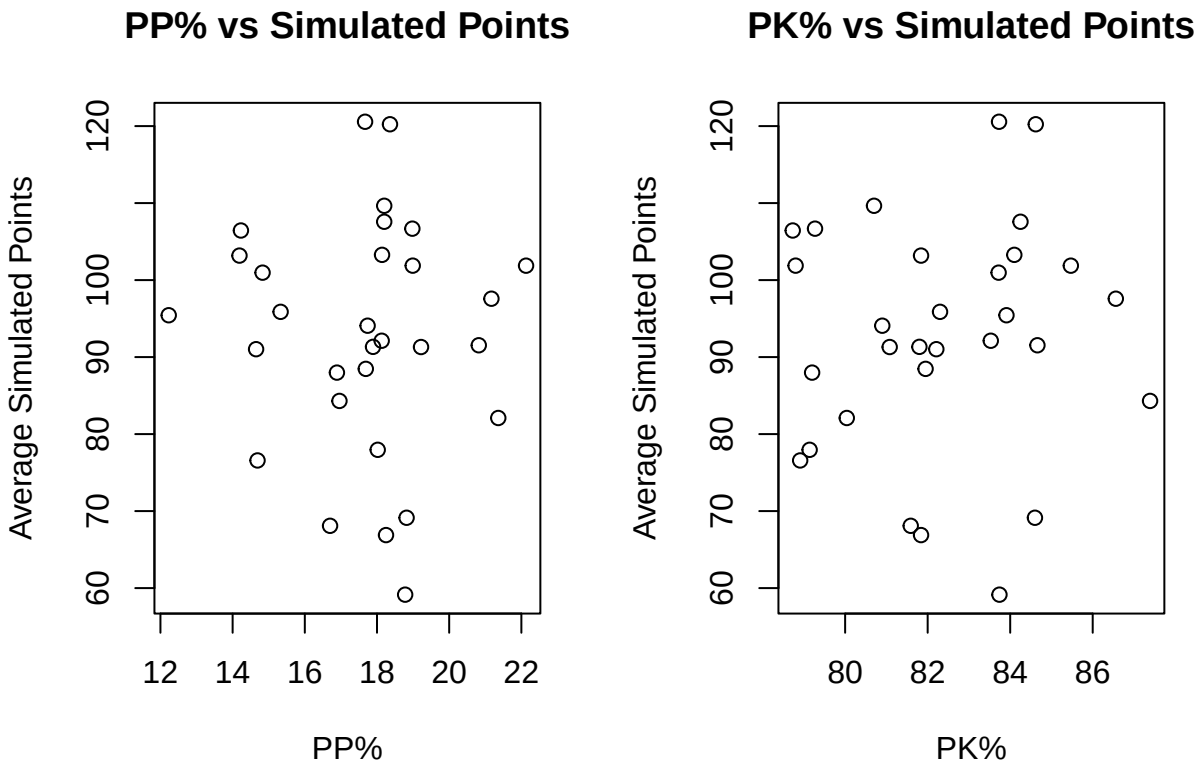
	Team	PP.	PK.	S.SA	PDO	diff.PP	diff.PK	diff.S.SA	diff.PDO
2	Los Angeles Kings	14.23	78.73	0.4933031	99.3	-3.412	-3.622667	-0.0070401	-0.7

Table 22: Season Statistics for the St. Louis Blues

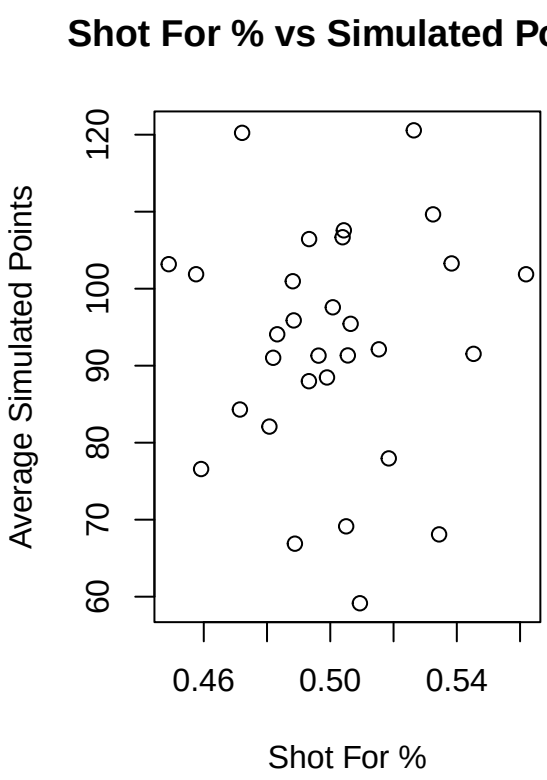
	Team	PP.	PK.	S.SA	PDO	diff.PP	diff.PK	diff.S.SA	diff.PDO
3	St. Louis Blues	14.65	82.21	0.4819401	97	-2.992	-0.1426667	-0.018403	-3

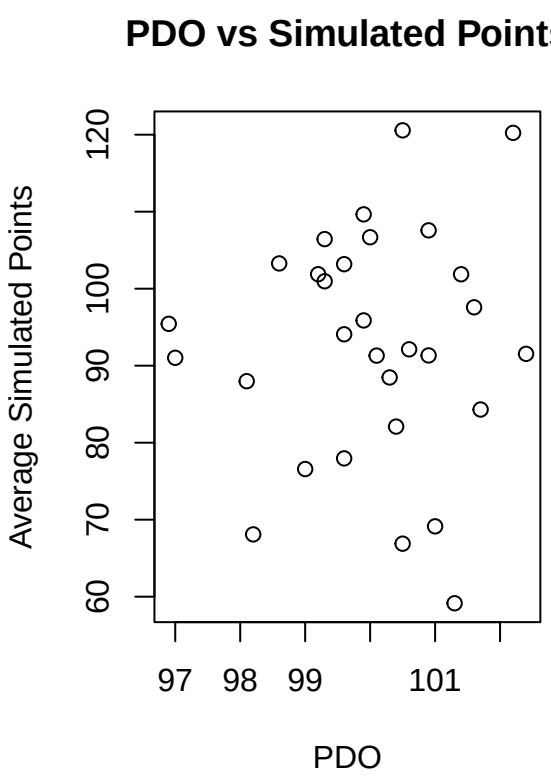
Simulation vs. Statistics Graphics

```
par(mfrow = c(1, 2))

plot(team_stats$PP., summary$average_points, xlab = "PP%",
     ylab = "Average Simulated Points", main = "PP% vs Simulated Points")


plot(team_stats$PK., summary$average_points, xlab = "PK%",
     ylab = "Average Simulated Points", main = "PK% vs Simulated Points")
```




```
plot(team_stats$S.SA, summary$average_points, xlab = "Shot For %",
     ylab = "Average Simulated Points", main = "Shot For % vs Simulated Points")

plot(team_stats$PDO, summary$average_points, xlab = "PDO",
     ylab = "Average Simulated Points", main = "PDO vs Simulated Points")
```






```
cor(team_stats$PP., summary$average_points)
```

```
## [1] -0.04339668
```

```
cor(team_stats$PK., summary$average_points)
```

```
## [1] 0.08122941
```

```
cor(team_stats$S.SA, summary$average_points)
```

```
## [1] 0.02585755
```

```
cor(team_stats$PDO, summary$average_points)
```

```
## [1] 0.05311544
```