

Model 1: fixed

Ben Moolman, Craig Orman, Ethan Pross

Data Prep and Cleaning

```
# Load and clean data
original_tbl <- read.csv("./NBA-BoxScores-2023-2024.csv") |>
  mutate(
    START_POSITION = na_if(START_POSITION, "") |> factor(),
    COMMENT = na_if(COMMENT, "") |> factor(),
    MIN = na_if(MIN, ""),
    MIN = str_replace(MIN, "([0-9]+)\\. [0-9]+:", "\\1:")
  )

# Filter to starters only
starting_dat <- original_tbl |>
  filter(!is.na(START_POSITION))

# Calculate team points per game
team_points <- original_tbl |>
  filter(!is.na(PTS)) |>
  group_by(GAME_ID, TEAM_ID) |>
  summarize(TeamPoints = sum(PTS), .groups = "drop")

# Join with itself to get opponent points
team_vs_opponent <- team_points |>
  inner_join(team_points, by = "GAME_ID", suffix = c("", ".opp")) |>
  filter(TEAM_ID != TEAM_ID.opp) |>
  rename(OPP_TEAM_ID = TEAM_ID.opp, OpponentPoints = TeamPoints.opp)

# Compute average opponent points allowed per team (DRTG)
team_drtg <- team_vs_opponent |>
  group_by(TEAM_ID) |>
  summarize(DRTG_proxy = mean(OpponentPoints), n_games = n(), .groups = "drop")

# Build opponent_map from distinct team-game pairs
game_team_pairs <- original_tbl |>
  select(GAME_ID, TEAM_ID) |>
  distinct()

# Create mapping of TEAM_ID and OPP_TEAM_ID for each game
opponent_map <- game_team_pairs |>
  inner_join(game_team_pairs, by = "GAME_ID") |>
  filter(TEAM_ID.x != TEAM_ID.y) |>
  rename(TEAM_ID = TEAM_ID.x, OPP_TEAM_ID = TEAM_ID.y)
```

```

# Join with defensive ratings (DRTG)
opponent_map <- opponent_map |>
  left_join(team_drtg |> rename(OPP_TEAM_ID = TEAM_ID, OPP_DRTG = DRTG_proxy), by = "OPP_TEAM_ID")

# Merge opponent info into starting dataset and center DRTG
mean_drtg <- mean(team_drtg$DRTG_proxy)

starting_dat <- starting_dat |>
  left_join(opponent_map, by = c("GAME_ID", "TEAM_ID")) |>
  mutate(centered_OPP_DRTG = OPP_DRTG - mean_drtg)

```

Model 1 implementation

Situation:

$$\begin{aligned}
 y_{ijk} &\sim \text{Binom}(n_{ijk}, p_{ik}) \\
 n_{ijk} &\sim \text{fixed } n, \text{ trying mean, median, max} \\
 p_{ik} &= p_i \times \exp(\gamma(\text{DRTG}_k - \bar{\text{DRTG}})) \\
 p_i &\sim \text{Beta}(5, 5)
 \end{aligned}$$

Metropolis Hastings Implementation to Obtain p_i

```

log_q = function(theta, y, n) {
  if (theta < 0 || theta > 1) return(-Inf)
  sum(dbinom(y, size = n, prob = theta, log = TRUE)) + dbeta(theta, 5, 5, log = TRUE)
}

MH_beta_binom = function(current = 0.5, prop_sd = 0.05, n_vec, y_vec, n_iter = 1000) {
  samps = rep(NA, n_iter)
  for (i in 1:n_iter) {
    proposed = rnorm(1, current, prop_sd)
    logr = log_q(proposed, y = y_vec, n = n_vec) - log_q(current, y = y_vec, n = n_vec)
    if (log(runif(1)) < logr) current = proposed
    samps[i] = current
  }
  return(samps)
}

MH_beta_grid_search = function(current = 0.5, n_vec, y_vec, n_iter = 1000) {
  vals <- seq(0.005, 1, by = 0.005)
  effect_sizes <- data.frame(sd = vals, ess = NA)

  for (i in seq_along(vals)) {
    samps <- MH_beta_binom(current = current, prop_sd = vals[i],
                          n_vec = n_vec, y_vec = y_vec, n_iter = n_iter)
    effect_sizes$ess[i] <- effectiveSize(samps)
  }

  best_sd <- effect_sizes$sd[which.max(effect_sizes$ess)]
}

```

```

final_samps <- MH_beta_binom(current = current, prop_sd = best_sd,
                             n_vec = n_vec, y_vec = y_vec, n_iter = n_iter)

return(list(
  samples = final_samps,
  best_sd = best_sd,
  ess_table = effect_sizes
))
}

```

Data Prep for Exploring LeBron James vs GSW (Golden State Warriors)

```

# Filter to LeBron James's games
# LeBron PLAYER_ID is 2544
lebron_dat <- starting_dat |>
  filter(PLAYER_ID == 2544)

# Centered DRTG values for each opponent team LeBron faced
# Only has 28 rows because LeBron only played against 28 out of 30 NBA teams
# LeBron did not play against Milwaukee Bucks (or the Lakers, his own team)
lebron_team_drtg <- lebron_dat |>
  group_by(OPP_TEAM_ID) |>
  summarize(centered_OPP_DRTG = first(centered_OPP_DRTG), .groups = "drop") |>
  arrange(OPP_TEAM_ID)

# Store response and trial vectors
Y <- lebron_dat$FGM
N_vec <- lebron_dat$FGA

mh_result <- MH_beta_grid_search(current = 0.5,
                                y_vec = Y,
                                n_vec = N_vec,
                                n_iter = 1000)

p_samples <- mh_result$samples
mh_result$best_sd # tuning parameter chosen

## [1] 0.035

drtg_vec <- lebron_team_drtg$centered_OPP_DRTG
gamma_val <- 0.01 # fixed tuning value

# Compute p_ik matrix: rows = posterior draws, cols = opponent teams
p_ik_matrix <- outer(
  p_samples,
  drtg_vec,
  FUN = function(p, drtg) p * exp(gamma_val * drtg)) |> pmin(1) # clip to max 1

# Posterior mean FG% per opponent
p_ik_mean <- colMeans(p_ik_matrix)

```

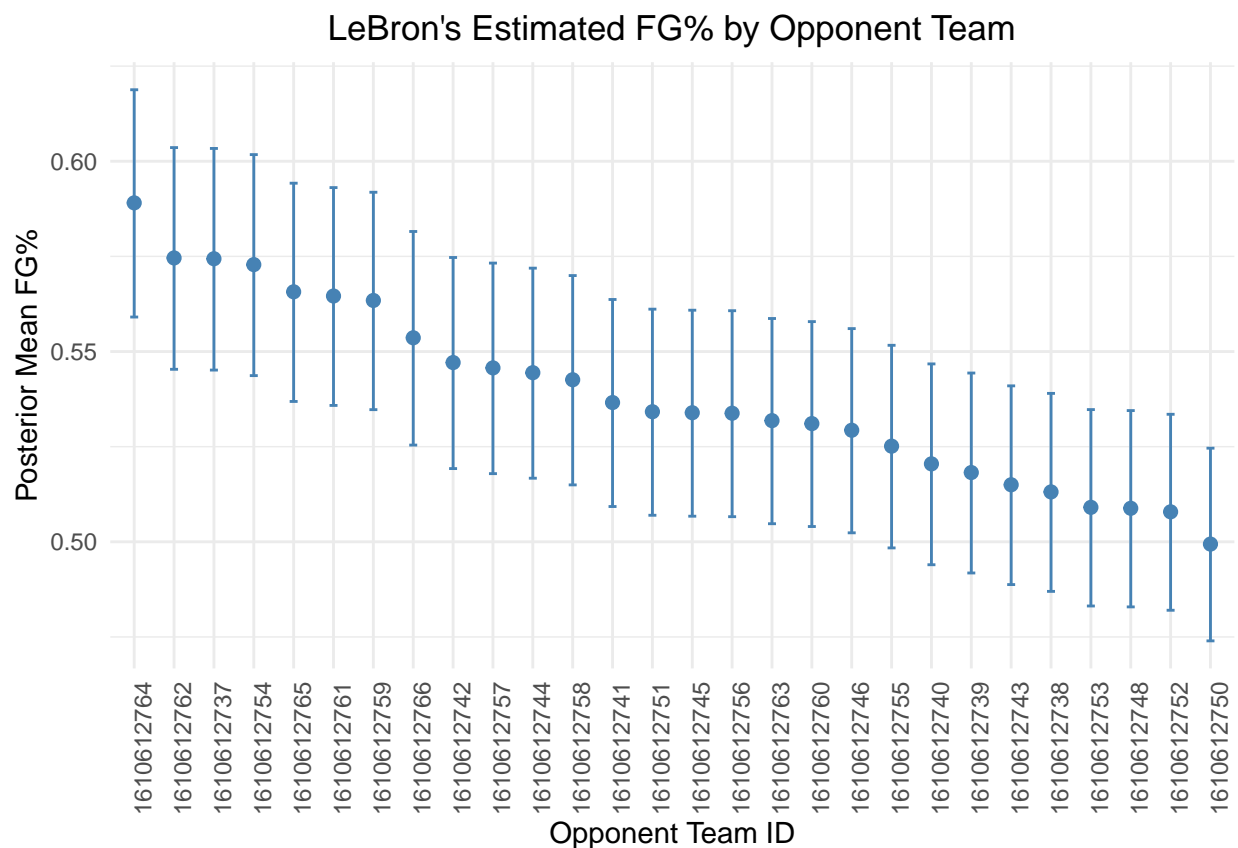
```

# Credible intervals
p_ik_CI <- apply(p_ik_matrix, 2, quantile, probs = c(0.025, 0.975))

# Combine into a table
p_ik_summary <- data.frame(
  OPP_TEAM_ID = lebron_team_drtg$OPP_TEAM_ID,
  p_ik_mean = p_ik_mean,
  p_ik_lower = p_ik_CI[1,],
  p_ik_upper = p_ik_CI[2,]
)

ggplot(p_ik_summary, aes(x = reorder(as.factor(OPP_TEAM_ID), -p_ik_mean), y = p_ik_mean)) +
  geom_point(color = "steelblue", size = 2) +
  geom_errorbar(aes(ymin = p_ik_lower, ymax = p_ik_upper), width = 0.2, color = "steelblue") +
  labs(
    title = "LeBron's Estimated FG% by Opponent Team",
    x = "Opponent Team ID",
    y = "Posterior Mean FG%"
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 90, hjust = 1),
    plot.title = element_text(hjust = 0.5)
  )

```



Now we will look at LeBron against Stephen Curry predictions (against Golden State Warriors, TEAM_ID

```
= 1610612744)
```

```
# Get centered DRTG for GSW (Golden State Warriors)
drtg_gsw <- lebron_team_drtg$centered_OPP_DRTG[lebron_team_drtg$OPP_TEAM_ID == 1610612744]
gamma_val <- 0.01 # fixed tuning value
p_ik_gsw <- p_samples * exp(gamma_val * drtg_gsw)
p_ik_gsw <- pmin(p_ik_gsw, 1) # ensure FG% stays within [0,1]

mean(p_ik_gsw)
```

```
## [1] 0.5444422
```

```
quantile(p_ik_gsw, c(0.025, 0.975))
```

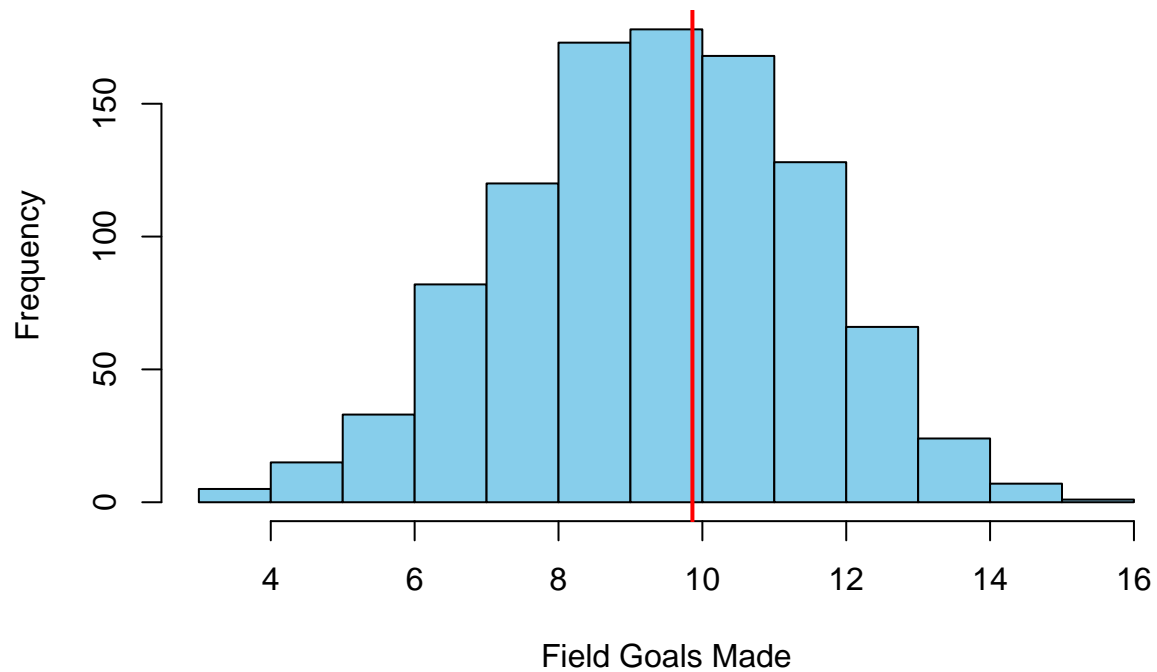
```
##      2.5%      97.5%
## 0.5167151 0.5719141
```

```
# FGAs for LeBron
n_mean <- round(mean(lebron_dat$FGA, na.rm = TRUE))
n_median <- round(median(lebron_dat$FGA, na.rm = TRUE))
n_max <- max(lebron_dat$FGA, na.rm = TRUE)

# Simulations using different n_i values
set.seed(5440)
fgm_sim_mean <- rbinom(length(p_ik_gsw), size = n_mean, prob = p_ik_gsw)
fgm_sim_median <- rbinom(length(p_ik_gsw), size = n_median, prob = p_ik_gsw)
fgm_sim_max <- rbinom(length(p_ik_gsw), size = n_max, prob = p_ik_gsw)

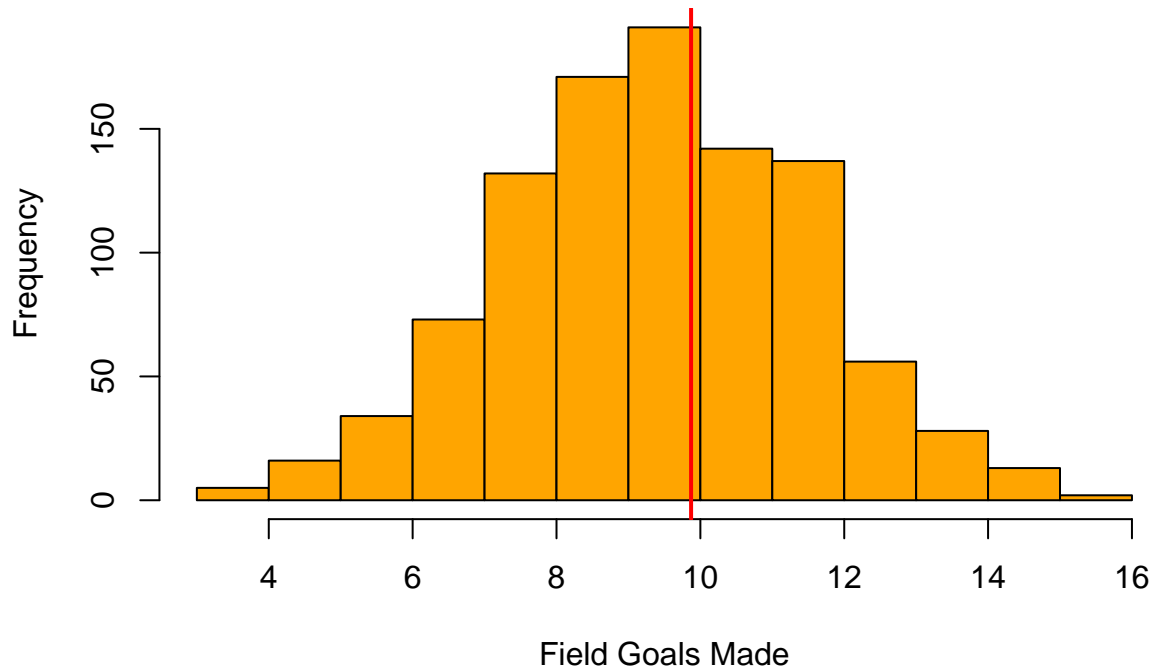
# Plot 1: Mean FGA
hist(fgm_sim_mean,
     main = paste("Simulated FGM vs GSW (n_i = mean =", n_mean, ")"),
     xlab = "Field Goals Made",
     col = "skyblue", breaks = 15)
abline(v = mean(fgm_sim_mean), col = "red", lwd = 2)
```

Simulated FGM vs GSW ($n_i = \text{mean} = 18$)



```
# Plot 2: Median FGA
hist(fgm_sim_median,
     main = paste("Simulated FGM vs GSW ( $n_i = \text{median} =$ ", n_median, ")"),
     xlab = "Field Goals Made",
     col = "orange", breaks = 15)
abline(v = mean(fgm_sim_median), col = "red", lwd = 2)
```

Simulated FGM vs GSW ($n_i = \text{median} = 18$)



```
# Plot 3: Max FGA
hist(fgm_sim_max,
     main = paste("Simulated FGM vs GSW ( $n_i = \text{max} =$ ", n_max, ")"),
     xlab = "Field Goals Made",
     col = "lightgreen", breaks = 15)
abline(v = mean(fgm_sim_max), col = "red", lwd = 2)
```

Simulated FGM vs GSW ($n_i = \max = 27$)

