Coverage Example

Paul Nguyen

2025-08-25

Coverage Example

To demonstrate one iteration of the coverage experiment, we will fit one instance of the cubic compositional model to one version of simulated data where we know the true event coefficients. For the example, we'll run the simulation with much smaller iterations (100) for faster compilation (normally 2000). The majority of the code is taken from study_coverage.R. You may need to run study/make_gen_data_manual.R to produce the simulated datasets.

```
set.seed(2)
library(tidyverse)
library(readxl)
library(rstan)
library(rlist)
library(tidyselect)
library(forcats)


study = "../../decathlon_simulation"
data_dir = "../../decathlon_simulation/data/"
script_dir = "../../decathlon_simulation/study/"
stan_dir = "../../decathlon_simulation/stan_mods/"

source(paste0(script_dir, "decathlon_funs.R"))
source(paste0(script_dir, "settings_coverage.R"))
sim_data_list <- readRDS(paste0(data_dir, "sim_data_list_manual.RData"))
```

In the study, we use a high performance computing cluster. We will manually set a job id for this example.

```
args = 2
job_id = as.numeric(args[1]) + 1

type = as.character(settings$type[job_id])
comp = as.character(settings$comp[job_id])
iter = as.integer(settings$iter[job_id])


dec_events <- c("hundred_m", "long_jump", "shot_put",
                "high_jump", "four_hundred_m", "hurdles",
                "discus", "pole_vault", "javelin",
                "fifteen_hundred_m")

# make sure to run make_gen_data_manual.R to generate data. already standardized.
sim_data <- sim_data_list[[iter]]
event_sums <- get_event_sums_df(sim_data)
```

```
dec_data_standard <- sim_data
# no predictions, so set arbritrary age, athlete id, vectors
age_vec <- c(20, 20)
athlete_id <- rep(0, length(age_vec))
is_new_athlete <- rep(1, length(age_vec))

sim <- get_comp_cubic_sim(age_vec = age_vec,
                                athlete_id = athlete_id,
                                is_new_athlete = is_new_athlete,
                                decathlon_data = dec_data_standard,
                                event_sums = event_sums,
                                stan_dir = stan_dir,
                                iter = 100, #this is set to 2000 during the actual experiments
                                return_all = T)
```

Now, we check the results.

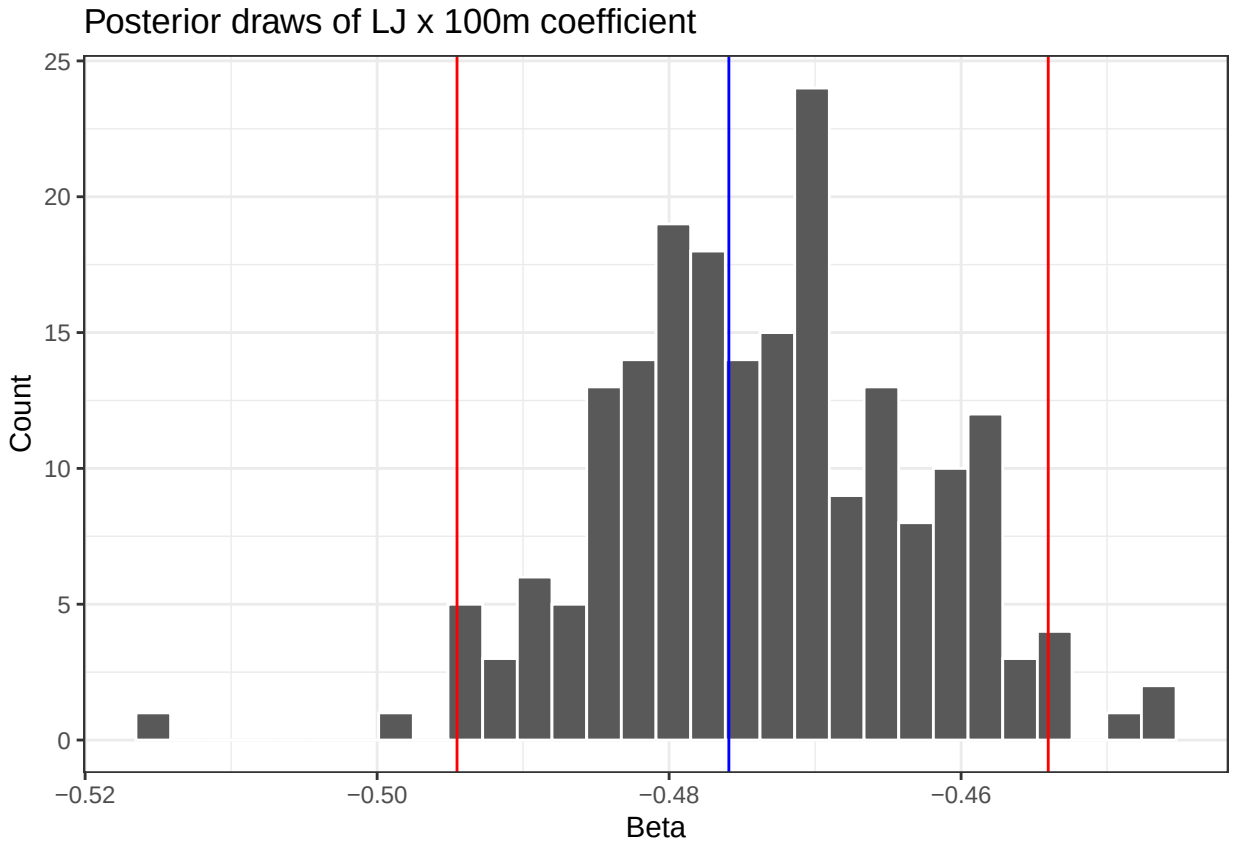
```
# check the coefficients for each event.
bounds_df <- data.frame()
for (i in 1:length(sim$sims_list)) {
  event = dec_events[i]
  target_sim <- sim$sims_list[[i]]
    if (type == "cubic"){
      # get quantiles for betas for age coefficients and event coefficents.
        q_beta1 <- quantile(target_sim$beta1, probs = c(.025, .975) )
        q_beta2 <- quantile(target_sim$beta2, probs = c(.025, .975) )
        q_beta3 <- quantile(target_sim$beta3, probs = c(.025, .975) )
        q_beta_age <- data.frame(target = event,
                                 predictor = c("age", "age2", "age3"),
                                 lb = c(q_beta1[1], q_beta2[1], q_beta3[1]),
                                 ub = c(q_beta1[2], q_beta2[2], q_beta3[2]))
    } else { #spline
      q_beta_age_mat <- apply(X = target_sim$beta_age,
                              MARGIN = 2,
                              FUN = quantile, probs = c(0.025, .975))
      q_beta_age <- data.frame(target = event,
                               predictor = paste0("age", 1:dim(q_beta_age_mat)[2]),
                               lb = q_beta_age_mat[1,],
                               ub = q_beta_age_mat[2,])
    }
  if (comp == "simple"){
    q_beta_y <- NA
  } else{ # compositional
    if (event == "hundred_m") {
      q_beta_y <- NA
    } else{
      q_beta_y <- apply(X = target_sim$betaY,
                        MARGIN = 2,
                        FUN = quantile, probs = c(0.025, .975))
      colnames(q_beta_y) <- dec_events[1:(i-1)]
      q_beta_y_lb <- data.frame(t(q_beta_y))[,1]
      q_beta_y_ub <- data.frame(t(q_beta_y))[,2]
      q_beta_y <- data.frame(target = event,
                             predictor = dec_events[1:(i-1)],
```

```
                              lb = q_beta_y_lb,
                              ub = q_beta_y_ub)%>%
        mutate(type = type,
               comp = comp,
               iter = iter)
    }
  }
  q_beta_age <- q_beta_age %>%
    mutate(type = type,
           comp = comp,
           iter = iter)
  bounds_df <- rbind(bounds_df, q_beta_age,
                     q_beta_y) %>%
    mutate(type = type,
           comp = comp,
           iter = iter) %>%
    drop_na()
}


# load the true coefficients
sim_beta_coef_list <- readRDS(paste0(data_dir, "beta_list_sim_manual.RData"))

# check if true coefficients fall within credible intervals.
for (i in 1:nrow(bounds_df)) {
  target_id <- bounds_df$target[i]
  predictor_id <- bounds_df$predictor[i]
  if (predictor_id == "age2") {
    predictor_id = "I(age^2)"
  } else if (predictor_id == "age3"){
    predictor_id = "I(age^3)"
  }
  beta_coef <- sim_beta_coef_list[[target_id]][predictor_id]
  bounds_df$true_coef[i] <- beta_coef
}
bounds_df <- bounds_df %>%
  mutate(cov_check = (true_coef < ub) & (true_coef > lb))
```

To create the coverage table in the paper, we run the compositional cubic model for all 200 datasets, and calculate the proportion of iterations in which the credible interval correctly contains the true coefficient. We can also display the credible interval for particular β for the current iteration.

```
# the effect of 100m on long jump
beta_df <- data.frame(sim$sims_list$long_jump_sims$betaY)
colnames(beta_df) <- "Beta"


ggplot(data = beta_df, mapping = aes(x = Beta)) +
  geom_histogram(color = "white") +
  geom_vline(xintercept = c(quantile(beta_df$Beta, c(0.025, 0.975))),
             color = "red") +
  geom_vline(xintercept = sim_beta_coef_list$long_jump["hundred_m"] ,
             color = "blue") +
  labs(x = "Beta",
```

```
      y = "Count",
      title = "Posterior draws of LJ x 100m coefficient") +
theme_bw()
```



In the plot above, we mark 95% Credible interval for 100m coefficient marked in red, with the true value of β in blue. In this instance, we see that the interval contains β when modeling the long jump.