The Illusion of Certainty in Meta-Analysis

Bradley Kolb

Packages

```
library(dplyr)
library(here)
```

Stan model

```
// varying (random) effect meta-analysis
// adapted from www.mc-stan.org/docs/stan-users-guide/measurement-error.html
data {
  int<lower=0> J; // num studies
  array[J] int<lower=0> n_t; // num cases, treatment
  array[J] int<lower=0> r_t; // num events, treatment
  array[J] int<lower=0> n_c; // num cases, control
  array[J] int<lower=0> r_c; // num events, control

  int<lower=0> estimate_posterior; // switch for estimating posterior vs running prior predi
  int<lower=0> priors; // switch for checking sensitivity of posterior to alternative specifi
}
transformed data {
  array[J] real y; // log odds ratio for each study
  for (j in 1:J) {
    y[j] = log(r_t[j]) - log(n_t[j] - r_t[j])
    - (log(r_c[j]) - log(n_c[j] - r_c[j]));
  }

  array[J] real<lower=0> se; // standard error of y (inverse variance method)
  for (j in 1:J) {
    se[j] = sqrt(1.0 / r_t[j] + 1.0 / (n_t[j] - r_t[j])
```

```
    + 1.0 / r_c[j] + 1.0 / (n_c[j] - r_c[j]));
  }
}
parameters {
  real mu; // mean treatment effect
  real<lower=0> tau; // deviation of treatment effects from the mean
  vector<offset=mu,multiplier=tau>[J] theta; // trial-specific treatment effects
}
model {
  if (estimate_posterior == 1) {
  y[1:J] ~ normal(theta[1:J], se[1:J]);
  }

  theta[1:J] ~ normal(mu, tau);
  if (priors == 1) { // standard normal
    mu ~ std_normal();
    tau ~ std_normal();
  } else { // CDSR
    mu ~ student_t(3.8, 0, 0.48);
    tau ~ lognormal(-1.44, 0.79);
  }
}
generated quantities {
  // pooling metrics
  vector[J] se2 = square(to_vector(se)); // approximate sampling variance for each study
  real se2_hat = sum(se2) / J; // average approximate sampling variance across all studies
  real<lower=0> i2 = square(tau) / (square(tau) + se2_hat); // proportion of total variance i
  vector[J] p = 1 - (square(tau) / (square(tau) + se2)); // proportion of variance in the tri

  // posterior predictive distribution
  real theta_new = normal_rng(mu, tau);

  // event probabilities
  real mu_gt_0 = mu > 0;
  real theta_new_gt_0 = theta_new > 0;

  // Universe of 100 possible future studies
  array[100] real theta_100;
  for (i in 1:100) {
    theta_100[i] = normal_rng(mu, tau);
  }
}
```

Set up simulation

```
run_single_sim <- function(seed,
                           n_trials = 10,
                           true_mu = 0.7,
                           true_tau = 0.7,
                           n_range = c(200, 200)) {

  set.seed(seed)

  # Generate varying sample sizes
  n_per_trial <- round(runif(n_trials, n_range[1], n_range[2]))

  # Generate true effects for each trial
  true_effects <- rnorm(n_trials, true_mu, true_tau)

  # Generate trial data
  data <- list(
    J = n_trials,
    n_t = n_per_trial,
    r_t = rbinom(n_trials, n_per_trial, plogis(true_effects)),
    n_c = n_per_trial,
    r_c = rbinom(n_trials, n_per_trial, plogis(0)),
    estimate_posterior = 1,
    priors = 1
  )

  observed_effects <-  log(data$r_t) - log(data$n_t - data$r_t) -
    (log(data$r_c) - log(data$n_c - data$r_c))

  observed_se <- sqrt(1/data$r_t + 1/(data$n_t - data$r_t)
               + 1/data$r_c + 1/(data$n_c - data$r_c)
          )

  observed_z <- abs(observed_effects/observed_se)

  # Fit model
  fit <- compiled_model$sample(
    data = data,
    seed = seed,
    chains = 4,
    parallel_chains = 4,
```

```
  refresh = 0,
  init = 1,
  adapt_delta = 0.99
)

print(fit$diagnostic_summary())
print(fit$summary(variables = c("mu", "tau"), "rhat", "ess_bulk"))

# Extract posterior samples
draws <- fit$draws()
summ <- fit$summary()
mu_samples <- as.vector(draws[,,"mu"])
theta_new_samples <- as.vector(draws[,,"theta_new"])
theta_100_samples <- fit$draws("theta_100", format = "matrix")

# Calculate metrics
coverage_mu <- mean(true_mu > quantile(mu_samples, 0.025) &
                      true_mu < quantile(mu_samples, 0.975))
coverage_theta <- mean(true_effects > quantile(theta_new_samples, 0.025) &
                         true_effects < quantile(theta_new_samples, 0.975))

# Return results
list(
  observations = list(
    observed_effects = observed_effects,
    observed_se = observed_se,
    observed_z = observed_z
  ),
  true_effects = true_effects,
  diagnostics = list(
    rhat = max(summ$rhat, na.rm=TRUE),
    min_ess = min(summ$ess_bulk, na.rm=TRUE)
  ),
  coverage = list(
    mu = coverage_mu,
    theta = coverage_theta
  ),
  samples = list(
    mu = fit$draws("mu", format = "matrix"),
    theta_new = fit$draws("theta_new", format = "matrix"),
    theta_100 = fit$draws("theta_100", format = "matrix")
  ),
```

```
    probs = c(
      mu_gt_0 = mean(mu_samples > 0),
      theta_new_gt_0 = mean(theta_new_samples > 0)
    )
  )
}

# Aggregate results
summarize_results <- function(results) {
  coverage_mu <- mean(sapply(results, function(x) x$coverage$mu))
  coverage_theta <- mean(sapply(results, function(x) x$coverage$theta))
  prob_data <- lapply(results, function(x) {
    mu_prob <- x$probs[1]  # mu_gt_0
    theta_prob <- x$probs[2]  # theta_new_gt_0
    c(mu_prob, theta_prob, mu_prob - theta_prob)
  })
  prob_comparisons <- do.call(rbind, prob_data)
  colnames(prob_comparisons) <- c("mu", "theta", "diff")

  list(
    coverage = list(mu = coverage_mu, theta = coverage_theta),
    prob_summary = list(
      mean_diff = mean(prob_comparisons[,"diff"]),
      sd_diff = sd(prob_comparisons[,"diff"]),
      range_diff = range(prob_comparisons[,"diff"])
    ),
    prob_distributions = prob_comparisons
    )
}
```

Run simulation

```
# experiment one
source("functions.r")
library(dplyr)
library(cmdstanr)

model <- cmdstan_model("model.stan")
n_sims <- 20
```

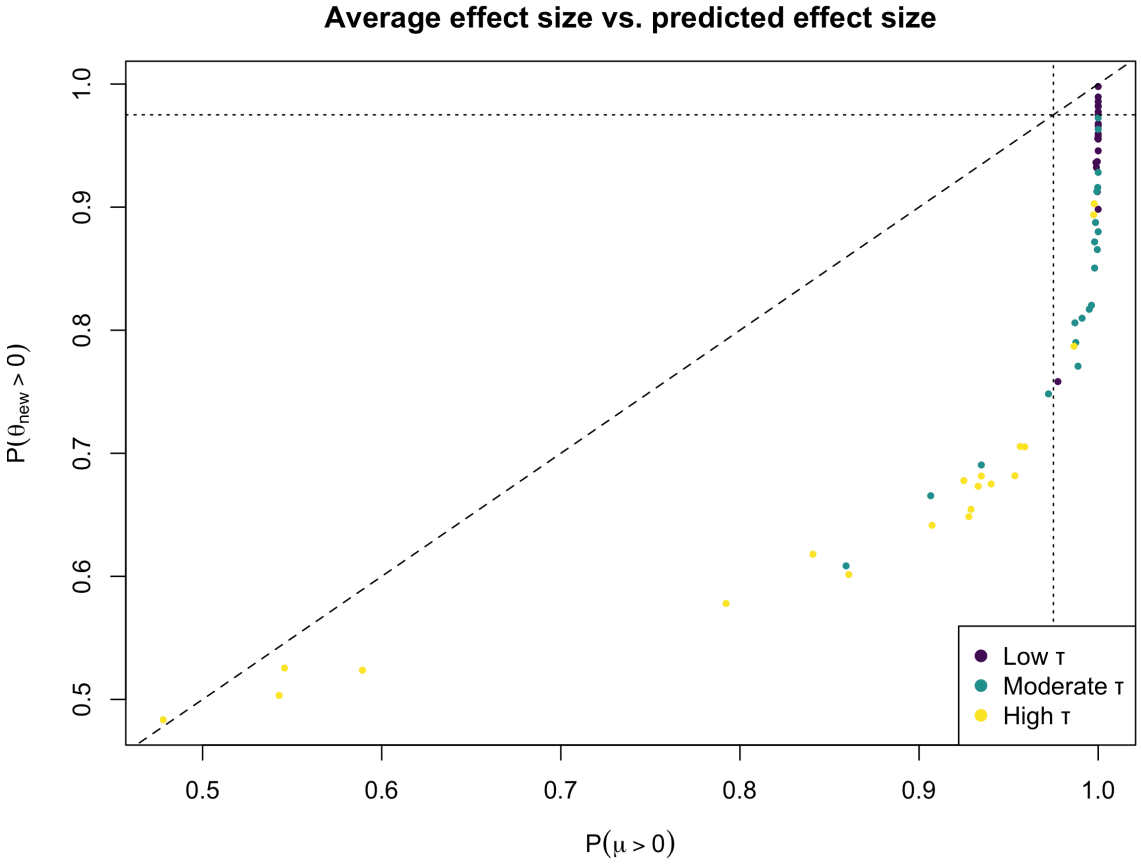
```
results_low <- lapply(1:n_sims, function(x) run_single_sim(seed = x, true_tau = 0.35))
results_mod <- lapply(1:n_sims, function(x) run_single_sim(seed = x, true_tau = 0.7))
results_high <-  lapply(1:n_sims, function(x) run_single_sim(seed = x, true_tau = 1.4))

probs_df <- rbind(
  cbind(do.call(rbind, lapply(results_low, function(x) x$probs)), condition="low"),
  cbind(do.call(rbind, lapply(results_mod, function(x) x$probs)), condition="moderate"),
  cbind(do.call(rbind, lapply(results_high, function(x) x$probs)), condition="high")
) %>% as.data.frame()

probs_df$condition <- factor(probs_df$condition,
                             levels = c("low", "moderate", "high"))

saveRDS(probs_df, "experiment_one.RDS")
```

```
probs_df <- readRDS(here("experiment_one.RDS"))
colors <- viridis::viridis(3)[c(1,2,3)]
par(mar = c(4, 4, 3, 2))
plot(probs_df$mu_gt_0, probs_df$theta_new_gt_0,
     xlab = expression(P(mu > 0)),
     ylab = expression(P(theta[new] > 0)),
     main = "Average effect size vs. predicted effect size",
     col = colors[probs_df$condition],
     pch = 19,
     cex = 0.5)
abline(0, 1, lty = 2)
abline(v = .975, lty = 3)
abline(h = .975, lty = 3)
legend("bottomright",
       legend = c("Low  ", "Moderate  ", "High  "),
       col = colors,
       pch = 19)
```


Average effect size vs. predicted effect size
P(θnew > 0)
P(μ > 0)
0.5
0.6
0.7
0.8
0.9
1.0
Low τ
Moderate τ
High τ