Bayesian comparison of two groups

Jelena H. Pantel

2023-01-24

```
library(brms)
library(ggplot2)
library(ggthemes)
library(tidyverse)
library(tidybayes)
```

Overview

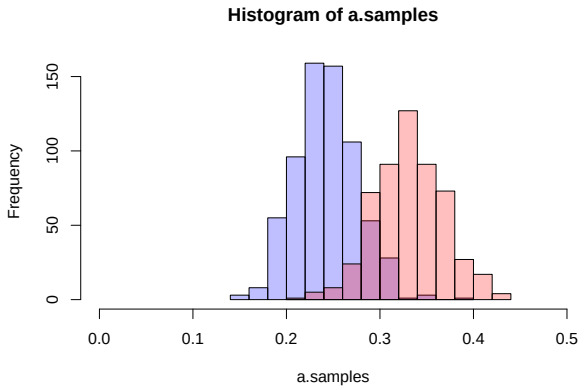
This is a walk-through in Bayesian comparison of two group. Because some of the focal data is bounded between 0-1, we use a Beta distribution to describe the data. The test then is whether the data is better described by 1 common beta distribution, or two distinct beta distributions, one per group.

Example data

I create some false data to mimic the Diversity index data

```
# data
set.seed(42)
prior.alpha <- 3
prior.beta <- 7
a.samples <- rbeta(670, 36 + prior.alpha, 114 + prior.beta)
b.samples <- rbeta(540, 50 + prior.alpha, 100 + prior.beta)
p1 <- hist(a.samples, main = "histogram of proportions")
p2 <- hist(b.samples)
```

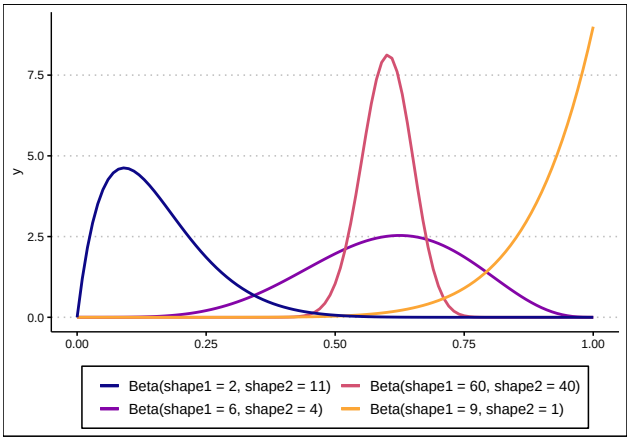
```
plot(p1, col = rgb(0, 0, 1, 1/4), xlim = c(0, 0.5))  # first histogram
plot(p2, col = rgb(1, 0, 0, 1/4), xlim = c(0, 0.5), add = T)  # second histogram
```



How to model the data

We use a beta distribution for the response variable Y, whose shape depends on two parameters: a rate parameter a and a shape parameter b. Beta distributions are often used to model probabilities (and proportions too of course).

```
ggplot2::ggplot() + geom_function(fun = dbeta, args = list(shape1 = 6,
    shape2 = 4), aes(color = "Beta(shape1 = 6, shape2 = 4)"),
    size = 1) + geom_function(fun = dbeta, args = list(shape1 = 60,
    shape2 = 40), aes(color = "Beta(shape1 = 60, shape2 = 40)"),
    size = 1) + geom_function(fun = dbeta, args = list(shape1 = 9,
    shape2 = 1), aes(color = "Beta(shape1 = 9, shape2 = 1)"),
    size = 1) + geom_function(fun = dbeta, args = list(shape1 = 2,
    shape2 = 11), aes(color = "Beta(shape1 = 2, shape2 = 11)"),
    size = 1) + scale_color_viridis_d(option = "plasma", end = 0.8,
    name = "", guide = guide_legend(nrow = 2)) + ggthemes::theme_clean() +
    theme(legend.position = "bottom")
```



We use the R package `brms` for our Bayesian model. The goal is to estimate the rate a and shape b parameters for the dataset, and compare models with shared or separate parameters values depending on group membership.

We have one extra aspect to consider: instead of working directly with a and b, `brms` uses μ and ϕ parameters, where:

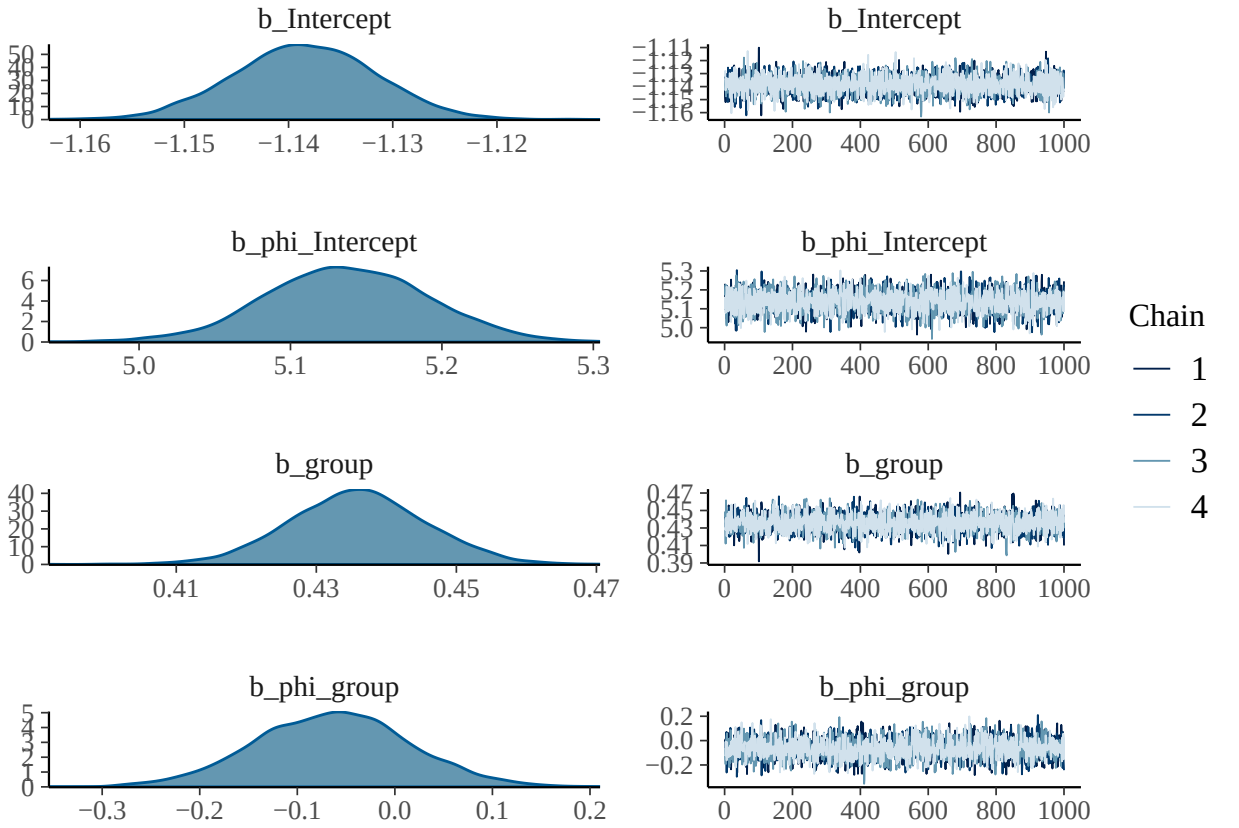
$$Shape1 : a = \mu\phi$$
$$Shape2 : b = (1 - \mu)\phi$$

The formula using the `brms` interface is quite simple, that the values Y are a function of group membership. The notation here with two formulas specifies that both the μ and ϕ terms are a function of group membership.

```
Y <- c(a.samples, b.samples)
group <- c(rep(0, 670), rep(1, 540))
dat <- as.data.frame(cbind(Y, group))
mod2 <- brms::brm(bf(Y ~ group, phi ~ group), data = dat, family = Beta(),
    chains = 4, iter = 2000, warmup = 1000, cores = 4, seed = 1234)
```

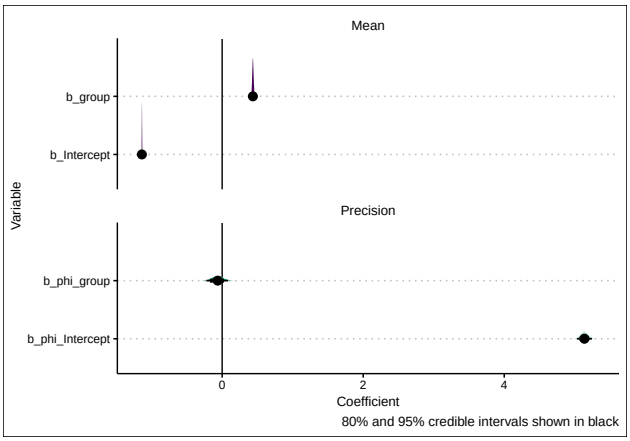
Results summary and visualization

```
summary(mod2)
#>  Family: beta
#>   Links: mu = logit; phi = log
#> Formula: Y ~ group
#>          phi ~ group
#>    Data: dat (Number of observations: 1210)
#>   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
#>          total post-warmup draws = 4000
#>
#> Population-Level Effects:
#>               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
#> Intercept        -1.14      0.01    -1.15    -1.13 1.00     4808     3430
#> phi_Intercept     5.14      0.05     5.03     5.24 1.00     3831     3269
#> group             0.44      0.01     0.42     0.46 1.00     4376     3277
#> phi_group        -0.06      0.08    -0.22     0.09 1.00     3864     2986
#>
#> Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
#> and Tail_ESS are effective sample size measures, and Rhat is the potential
#> scale reduction factor on split chains (at convergence, Rhat = 1).
plot(mod2)
```



We can interpret this to show that the two parameters differ depending on group, as the μ terms `b_Intercept` and `b_group` do not overlap, and the ϕ terms `b_phi_Intercept` and `b_phi_group` do not overlap. We can also see from the trace plots at right that the 4 chains converged, and therefore the parameter estimates are reliable samples of the posterior distributions. We can visualize these by showing the posterior distributions of distribution parameters by group: but *note* these coefficients are in their transformed scale: logits for the mean (μ), logs for the precision (ϕ).

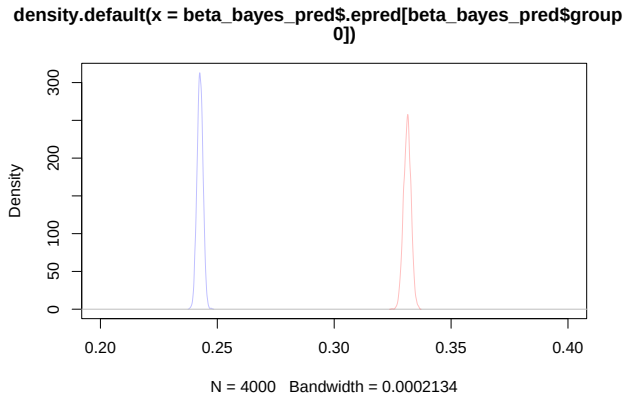
```
post_beta <- mod2 %>%
    tidybayes::gather_draws(`b_.*`, regex = TRUE) %>%
    mutate(component = ifelse(str_detect(.variable, "phi_"),
        "Precision", "Mean"), intercept = str_detect(.variable,
        "Intercept"))

ggplot2::ggplot(post_beta, aes(x = .value, y = fct_rev(.variable),
    fill = component)) + geom_vline(xintercept = 0) + stat_halfeye(aes(slab_alpha = intercept),
    .width = c(0.8, 0.95), point_interval = "median_hdi") + scale_fill_viridis_d(option = "viridis",
    end = 0.6) + scale_slab_alpha_discrete(range = c(1, 0.4)) +
    guides(fill = "none", slab_alpha = "none") + labs(x = "Coefficient",
    y = "Variable", caption = "80% and 95% credible intervals shown in black") +
    facet_wrap(vars(component), ncol = 1, scales = "free_y") +
    theme_clean()
```



To view the estimated posterior distributions for the Y values (these are predictions made from the models):

```
# Plug a dataset where quota is FALSE and TRUE into the
# model
beta_bayes_pred <- mod2 %>%
    epred_draws(newdata = tibble(group = c(0, 1)))
p1 <- density(beta_bayes_pred$.epred[beta_bayes_pred$group ==
    0])
p2 <- density(beta_bayes_pred$.epred[beta_bayes_pred$group ==
    1])
```

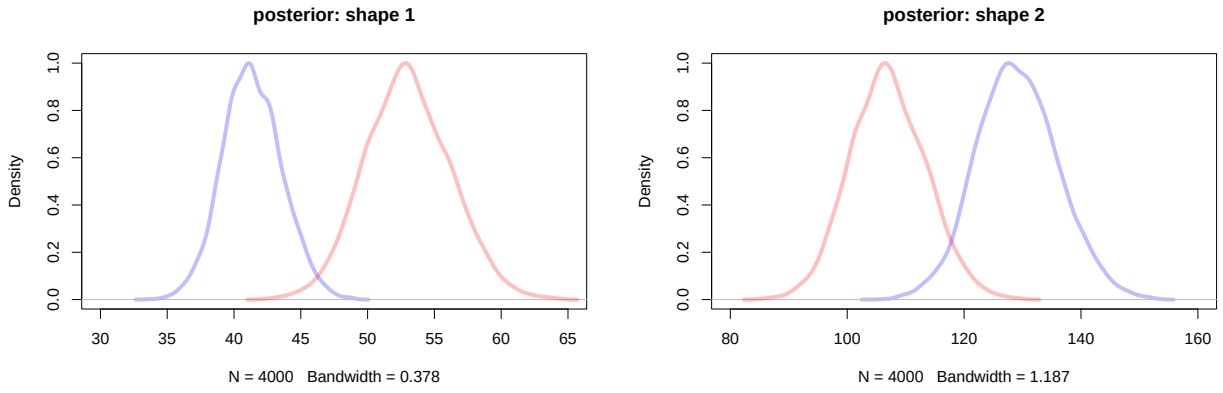
```
plot(p1, col = rgb(0, 0, 1, 1/4), xlim = c(0.2, 0.4))  # first histogram
lines(p2, col = rgb(1, 0, 0, 1/4))  # second histogram
```



We take one more step, to view the posterior distributions for the beta shape1 and shape2 parameters by group:

```
# Get posterior distributions of model parameters
post <- cbind(unlist(as_draws(mod2, "b_Intercept")), unlist(as_draws(mod2,
    "b_phi_Intercept")), unlist(as_draws(mod2, "b_group")), unlist(as_draws(mod2,
    "b_phi_group")))
post <- as.data.frame(post)
colnames(post) <- c("b_Intercept", "b_phi_Intercept", "b_group",
    "b_phi_group")
post$shape1_g1 <- NA
```

```
post$shape2_g1 <- NA
post$shape1_g2 <- NA
post$shape2_g2 <- NA
# Use the exp() and plogis() functions to get the values of
# the rate and shape parameters, taking the model intercept
# and group coefficients into account posterior
# distribution for shape1 and shape 2 parameters, group 1
post$mu_g1 <- plogis(post$b_Intercept)
post$phi_g1 <- exp(post$b_phi_Intercept)
post$shape1_g1 <- post$mu_g1 * post$phi_g1
post$shape2_g1 <- (1 - post$mu_g1) * post$phi_g1
# posterior distribution for shape1 and shape 2 parameters,
# group 2
post$mu_g2 <- plogis(post$b_Intercept + post$b_group)
post$phi_g2 <- exp(post$b_phi_Intercept + post$b_phi_group)
post$shape1_g2 <- post$mu_g2 * post$phi_g2
post$shape2_g2 <- (1 - post$mu_g2) * post$phi_g2

p1 <- density(post$shape1_g1)
p1$y = p1$y/max(p1$y)
p2 <- density(post$shape1_g2)
p2$y = p2$y/max(p2$y)
plot(p1, col = rgb(0, 0, 1, 1/4), xlim = c(30, 65), lwd = 4,
    main = "posterior: shape 1")  # first histogram
lines(p2, col = rgb(1, 0, 0, 1/4), lwd = 4)  # second histogram

p1 <- density(post$shape2_g1)
p1$y = p1$y/max(p1$y)
p2 <- density(post$shape2_g2)
p2$y = p2$y/max(p2$y)
plot(p1, col = rgb(0, 0, 1, 1/4), xlim = c(80, 160), lwd = 4,
    main = "posterior: shape 2")  # first histogram
lines(p2, col = rgb(1, 0, 0, 1/4), lwd = 4)  # second histogram
```



Here we can see most clearly that the shape1 (a) and shape2 (b) parameters of the underlying beta distributions are quite different from one another.