An Introduction to Bayesian Multilevel Models Using brms: A Case Study of Gender Effects on Vowel Variability in Standard Indonesian

Supplementary Materials

Contents

Moderation Analysis

In the current paper, we were interested in knowing whether men and women differ in the way they pronounce Indonesian vowels. Acknowledging that gender differences might be more similar accross repetitions of the same vowel, we allowed the mean variability (and the difference between men and women) to vary by vowel (by including a by-vowel varying intercept and varying slope). This strategy aimed at increasing the precision of our estimation of the gender effect. But it does not allow us to know whether the effect of gender differs according to the vowel and which vowels are more affected by gender. To answer this question, one might add vowel as a constant factor in the model.

This model corresponds to `bmod2` from the main manuscript, to which we added a constant effect for `vowel` as well as an interaction term `gender:vowel`. In R, `vowel` will be internally recoded as a factor, resulting in attributing a specific coefficient by vowel (and for each interaction `gender:vowel`). More precisely, the intercept will be assigned to the first level of the factor, and slopes will be assigned to the other levels.

In the current situation, vowels are alphabetically ordered (/a-e-i-o-u/) so that the intercept represents the mean predicted value of `distance` for vowel /a/ for all participants (without distinction of gender). This model can be implemented using `brms` as follows.

```r
library(brms)

prior <- c(
    prior(normal(0, 10), class = Intercept),
    prior(normal(0, 10), class = b),
    prior(cauchy(0, 10), class = sd),
    prior(cauchy(0, 10), class = sigma)
    )

bmod <- brm(
    distance ~ gender * vowel + (1|subj),
    data = indo, family = gaussian(),
    prior = prior,
    warmup = 2000, iter = 10000,
    chains = 2, cores = parallel::detectCores(),
    control = list(adapt_delta = 0.99)
    )
```

Interpreting the output

We can retrieve the coefficients for the constant effects using the `broom:tidy()` function, which gives the mean of the posterior distribution along with its standard error and credible interval.

```r
library(broom)
tidy(bmod, par_type = "non-varying", prob = 0.95)
```

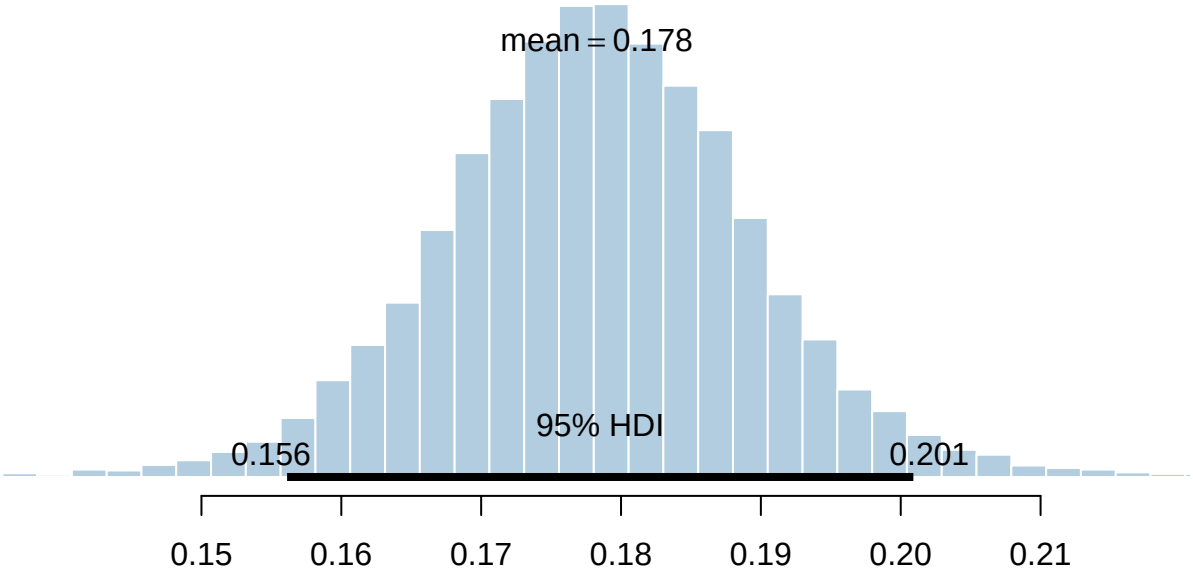
```
##                term     estimate   std.error       lower        upper
## 1         Intercept  0.214739757 0.007860863  0.19916116  0.230479766
## 2            gender -0.072957317 0.015842045 -0.10367968 -0.041889978
## 3          vowelDeD -0.031039652 0.006597524 -0.04387521 -0.018087670
## 4          vowelDiD -0.105302435 0.006652320 -0.11851492 -0.092096961
## 5          vowelDoD -0.021684682 0.006600733 -0.03462236 -0.008690341
## 6          vowelDuD -0.098663191 0.006596785 -0.11144575 -0.085799890
## 7  gender:vowelDeD  0.002757571 0.013348158 -0.02327183  0.028985579
## 8  gender:vowelDiD  0.058119979 0.013120279  0.03235011  0.083810202
## 9  gender:vowelDoD  0.008890188 0.013290617 -0.01708115  0.034721790
## 10 gender:vowelDuD  0.084381869 0.013097641  0.05844936  0.110003042
```

We mentioned above that the intercept represents the mean predicted value of `distance` for vowel /a/ for all participants (which is indeed very similar to `mean(indo$distance[indo$vowel=="/a/"])`).

The slope for `gender` represents the predicted difference betwen men and women concerning the

`distance` for vowel /a/. As `gender` was contrast coded, the intercept + 0.5 times the slope for gender gives the model predictions for men concerning vowel /a/, while the intercept + (-0.5) times the slope for gender gives the model prediction for women concerning vowel /a/. One can then study the marginal posterior distribution of each condition, working directly with the posterior samples.

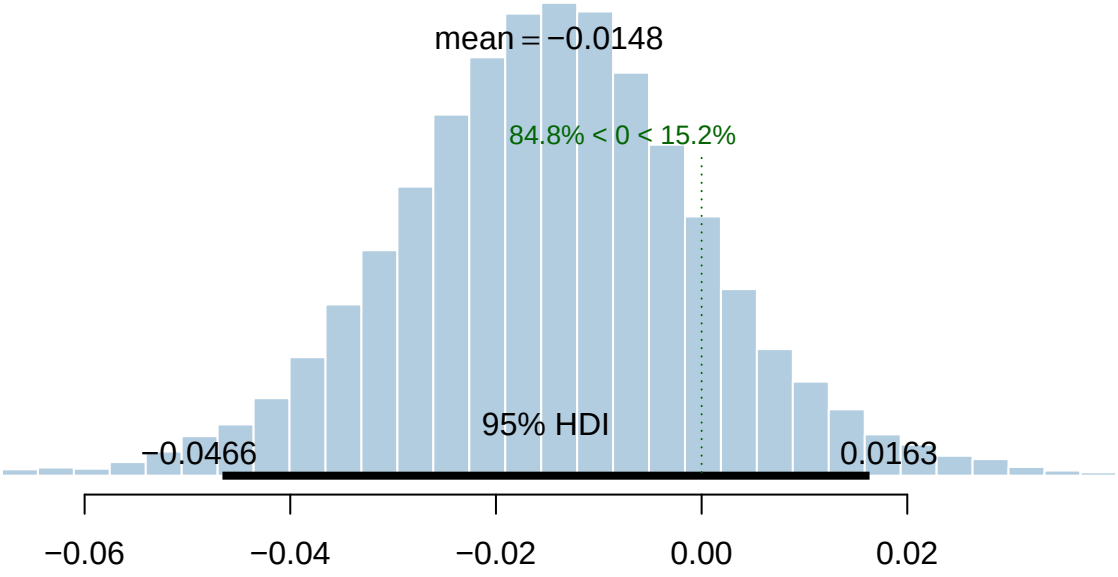
```
library(BEST)

# extracting posterior samples
post <- posterior_samples(bmod, pars = "^b_*")

# extracting predicted values for men
men <- post[["b_Intercept"]] + 0.5 * post[["b_gender"]]

# plotting the posterior distribution of men
plotPost(men, xlab = "", col = "#b3cde0", cex = 1)
```



Following the same strategy, one can investigate any comparison of interest. For instance, if we are interested in the difference between women and men concerning a specific vowel –say vowel /i/–, we can compute the difference between the model predictions for men and women, and plot this distribution. We simply need to know that interaction terms (e.g., `gender:vowelDiD`) represent deviations from the *baseline* slope for `gender`. Thus, we need to add them to obtain the effect of `gender` for a specific vowel.

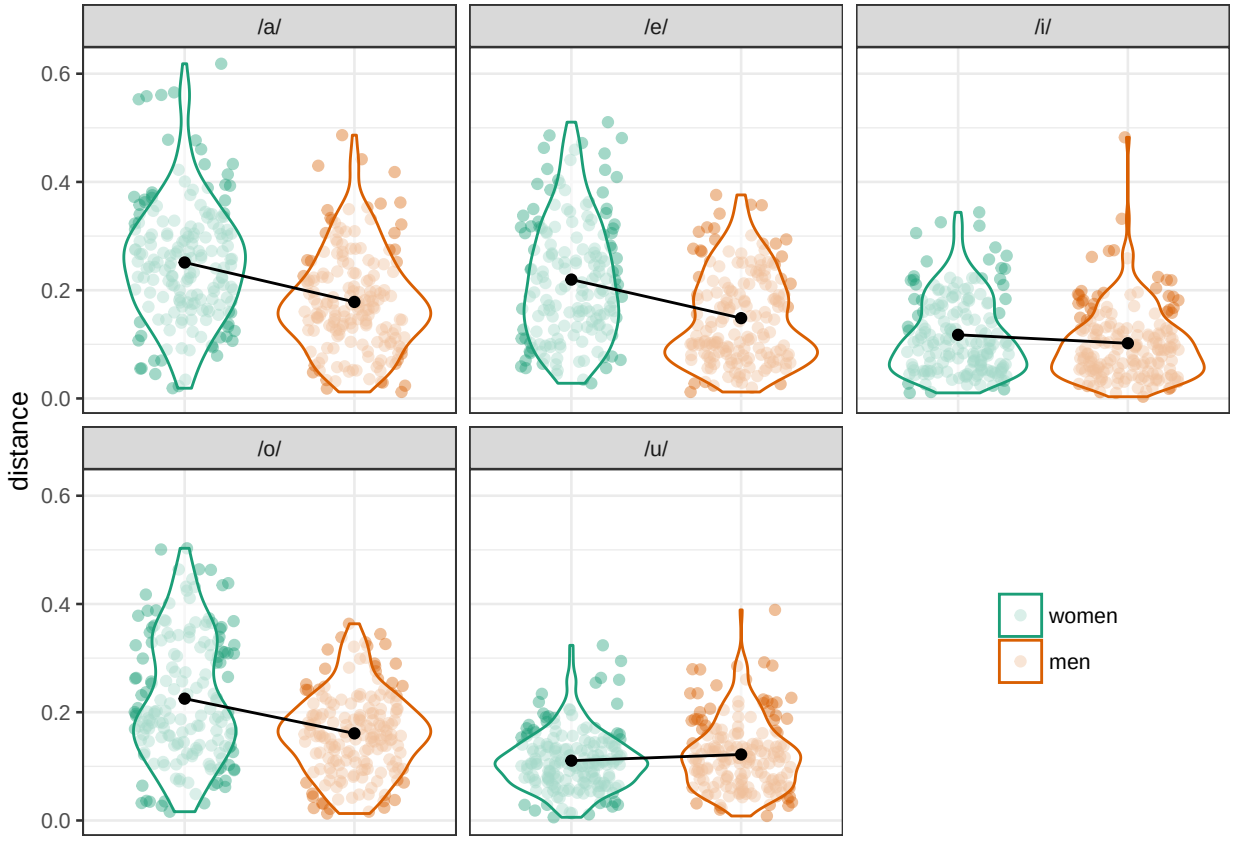
```
# computing the difference between men and women for vowel /i/
diff_i <- post[["b_gender"]] + post[["b_gender:vowelDiD"]]

# plotting it
```

```
plotPost(diff_i, xlab = "", col = "#b3cde0", compVal = 0, cex = 1)
```



Visualising predictions

One of the easiest ways to understand the output of the model is probably to plot its predictions. Below we plot the raw data along with the predictions of the model, represented by a horizontal line.



This is consistent with Figure 1 from the main manuscript. While women generally produce vowels with more variability, this gender effect seems more pronounced for open or mid vowels /a-e-o/, and less pronounced for close vowels /u/ and /i/.

Lognormal and Skew-Normal models

In the following, we fit a model with the same structure as `bmod5` (the last model from the main manuscript), except that we use a different likelihood (i.e., distribution for the residuals). In this case, we fit a lognormal and a skew normal models. Note that the only modification we bring to `bmod5` is that we change the `family` argument in the `brm` function, replacing `gaussian()` by `lognormal()` or `skew_normal()` (but see the `brms` package documentation for more details).

```
priors <- c(
    prior(normal(0, 10), class = Intercept),
    prior(normal(0, 10), class = b, coef = gender),
    prior(cauchy(0, 10), class = sd),
    prior(cauchy(0, 10), class = sigma),
    prior(lkj(2), class = cor)
    )
```

```
bmod5 <- brm(
    distance ~ gender + (1|subj) + (1+gender|vowel) + (1|subj:vowel),
    data = indo,
    family = gaussian(),
    prior = priors,
    warmup = 2000, iter = 10000,
    chains = 2, cores = parallel::detectCores(),
    control = list(adapt_delta = 0.99)
    )

bmod6 <- brm(
    distance ~ gender + (1|subj) + (1+gender|vowel) + (1|subj:vowel),
    data = indo,
    family = lognormal(),
    prior = priors,
    warmup = 2000, iter = 1e4,
    chains = 2, cores = parallel::detectCores(),
    control = list(adapt_delta = 0.99)
)

bmod7 <- brm(
    distance ~ gender + (1|subj) + (1+gender|vowel) + (1|subj:vowel),
    data = indo,
    family = skew_normal(),
    prior = priors,
    warmup = 2000, iter = 1e4,
    chains = 2, cores = parallel::detectCores(),
    control = list(adapt_delta = 0.99)
)
```

Model comparison

Once we have fitted these models, we can compare them using the `LOO` function (see section 3 on model comparison).

```
LOO(bmod5, bmod6, bmod7, cores = parallel::detectCores() )
```

```
##                  LOOIC    SE
## bmod5         -3591.14 68.13
## bmod6         -3697.56 73.42
## bmod7         -3690.61 64.56
## bmod5 - bmod6   106.42 60.15
```

```
## bmod5 - bmod7    99.46 26.26
## bmod6 - bmod7    -6.96 46.50
```

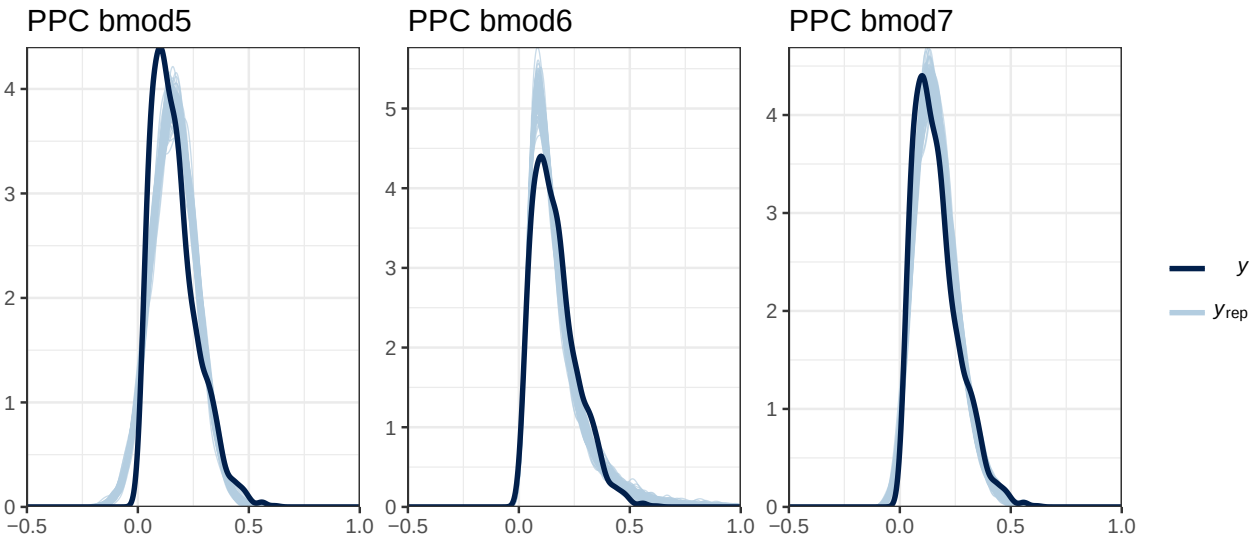
This comparison reveals that replacing the Normal likelihood by a lognormal or a skew-normal one, while adding a supplementary parameter to estimate (the alpha parameter for the skew-normal), induced a considerable decrease in LOOIC value (`bmod6` and `bmod7` LOOIC values are approximately 100 points below the LOOIC value of `bmod5`), although being associated with quite a large uncertainty (as expressed by the SE).

Posterior predictive checking

Another useful diagnostic of the model's predictive abilities is known as *posterior predictive checking* (PPC) and consists in comparing observed data to data simulated from the posterior distribution (e.g., Gelman et al., 2013). The idea behing PPC is quite simple: if a model is a good fit, then we should be able to use it to generate data that resemble the data we observed (Gabry, Simpson, Betancourt, & Gelman, 2017). This is implemented in `brms` with the `pp_check` method, that provides various ways of visualising posterior predictive checks. Below we compare the posterior predictions of the three models we fitted previously (n = 100 simulated samples for each model), to the distribution of the original dataset.

```
pp_bmod5 <-
    brms::pp_check(bmod5, nsamples = 1e2) +
    ggtitle("PPC bmod5") +
    theme_bw(base_size = 10) +
    theme(legend.position = "none") +
    xlim(-0.5, 1)

pp_bmod6 <-
    brms::pp_check(bmod6, nsamples = 1e2) +
    ggtitle("PPC bmod6") +
    theme_bw(base_size = 10) +
    theme(legend.position = "none") +
    xlim(-0.5, 1)

pp_bmod7 <-
    brms::pp_check(bmod7, nsamples = 1e2) +
    ggtitle("PPC bmod7") +
    theme_bw(base_size = 10) +
    xlim(-0.5, 1)

library(patchwork) # adding ggplots
```

```
pp_bmod5 + pp_bmod6 + pp_bmod7
```



This confirms that the Normal model (`bmod5`), while being a convenient and easy to interpret model, can be improved to better predict the particular features of our data. Using a lognormal or a skew-normal likelihood improves the predictions of the model (i.e., the simulated samples seem to be closer to the observed data).

Session information

```
sessionInfo()
```

```
## R version 3.4.3 (2017-11-30)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS High Sierra 10.13.3
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
```

```
##  [1] patchwork_0.0.1    BEST_0.5.0         HDInterval_0.1.3
##  [4] broom_0.4.3        brms_2.1.4         Rcpp_0.12.15
##  [7] bindrcpp_0.2       phonR_1.0-7        forcats_0.2.0
## [10] stringr_1.2.0      dplyr_0.7.4        purrr_0.2.4
## [13] readr_1.1.1        tidyr_0.8.0        tibble_1.4.2
## [16] ggplot2_2.2.1.9000 tidyverse_1.2.1
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-131         matrixStats_0.53.0   xts_0.10-1
##  [4] lubridate_1.7.1      threejs_0.3.1        httr_1.3.1
##  [7] rprojroot_1.3-2      rstan_2.17.3         tools_3.4.3
## [10] backports_1.1.2      R6_2.2.2             DT_0.3
## [13] lazyeval_0.2.1       colorspace_1.3-2     withr_2.1.1.9000
## [16] gridExtra_2.3        mnormt_1.5-5         Brobdingnag_1.2-4
## [19] compiler_3.4.3       cli_1.0.0            rvest_0.3.2
## [22] xml2_1.2.0           shinyjs_1.0          labeling_0.3
## [25] colourpicker_1.0     scales_0.5.0.9000    dygraphs_1.1.1.4
## [28] mvtnorm_1.0-7        psych_1.7.8          digest_0.6.15
## [31] StanHeaders_2.17.2   foreign_0.8-69       rmarkdown_1.8
## [34] base64enc_0.1-3      pkgconfig_2.0.1      htmltools_0.3.6
## [37] htmlwidgets_1.0      rlang_0.1.6.9003     readxl_1.0.0
## [40] rstudioapi_0.7       shiny_1.0.5          bindr_0.1
## [43] zoo_1.8-1            jsonlite_1.5         crosstalk_1.0.0
## [46] gtools_3.5.0         inline_0.3.14        magrittr_1.5
## [49] loo_1.1.0            bayesplot_1.4.0      Matrix_1.2-12
## [52] munsell_0.4.3        abind_1.4-5          stringi_1.1.6
## [55] yaml_2.1.16          plyr_1.8.4           grid_3.4.3
## [58] parallel_3.4.3       crayon_1.3.4         miniUI_0.1.1
## [61] lattice_0.20-35      haven_1.1.1          hms_0.4.1
## [64] knitr_1.19           pillar_1.1.0         igraph_1.1.2
## [67] markdown_0.8         shinystan_2.4.0      codetools_0.2-15
## [70] reshape2_1.4.3       stats4_3.4.3         rstantools_1.4.0
## [73] glue_1.2.0           evaluate_0.10.1      modelr_0.1.1
## [76] httpuv_1.3.5         cellranger_1.1.0     gtable_0.2.0
## [79] assertthat_0.2.0     mime_0.5             xtable_1.8-2
## [82] coda_0.19-1          rsconnect_0.8.5      rjags_4-6
## [85] shinythemes_1.1.1    bridgesampling_0.4-0
```

References

Gabry, J., Simpson, D., Betancourt, M., & Gelman, A. (2017). *Visualization in Bayesian workflow.*

http://doi.org/10.1007/978-1-4612-1694-0_16

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian Data Analysis, Third Edition*. CRC Press.