Week 3

Santiago Barreda

Contents

0.1 hit in this chapter: 1-way anova: 4 groups . . . 1
0.2 2 way anova: respect the structure . . . 1
0.3 how to check specific contrasts from samples. . . . 1
0.4 heteroskedasticity . . . 1
0.5 parallelism, simple main effects and main effects . . . 1
0.6 interactions: dependent effects . . . 1
0.7 coding . . . 1

1 Comparing many groups **2**
1.1 Data and research questions . . . 2
1.2 Comparing four (or any number of) groups . . . 4
1.2.1 The model . . . 4
1.2.2 Fitting the model and interpreting the results . . . 5
1.2.3 An alternate approach: Sum coding . . . 7
1.2.4 Fitting the model and interpreting the results: sum coding . . . 7

0.1 hit in this chapter: 1-way anova: 4 groups

0.2 2 way anova: respect the structure

0.3 how to check specific contrasts from samples.

0.4 heteroskedasticity

0.5 parallelism, simple main effects and main effects

0.6 interactions: dependent effects

0.7 coding

1 Comparing many groups

Last chapter we talked about comparing two groups. Although it is simple, it is also fundamental and used often: more complicated problems can often be broken down into sets of many two-group questions. However, real experiments don't usually *begin* as two-group questions.

In the last chapter we found reliable, but noisy, difference between women and girls in average f0. In this chapter, we are going to consider the productions of f0 from all four groups in the data. These research questions would traditionally be addressed with a repeated-measures ANOVA.

1.1 Data and research questions

We are still going to work with the Hillenbrand et al. data, and we are going to add a variable indicating 'adultness', and one indicating gender. However, this time we are going to work with all four groups at the same time: `b` (boys), `g` (girls), `m` (men), and `w` (women).

```
library (brms)

url1 = "https://raw.githubusercontent.com/santiagobarreda"
url2 = "/stats-class/master/data/h95_vowel_data.csv"
h95 = read.csv (url(paste0 (url1, url2)))
## make variable that indicates if the talker is an adult
h95$adult = ""
h95$adult[h95$type %in% c('w','m')] = "adult"
h95$adult[h95$type %in% c('g','b')] = "child"
## make variable indicating speaker gender
h95$gender = "female"
h95$gender[h95$type %in% c('b','m')] = "male"

## check cross-tabulation of these predictors
table (h95$gender, h95$adult)
```

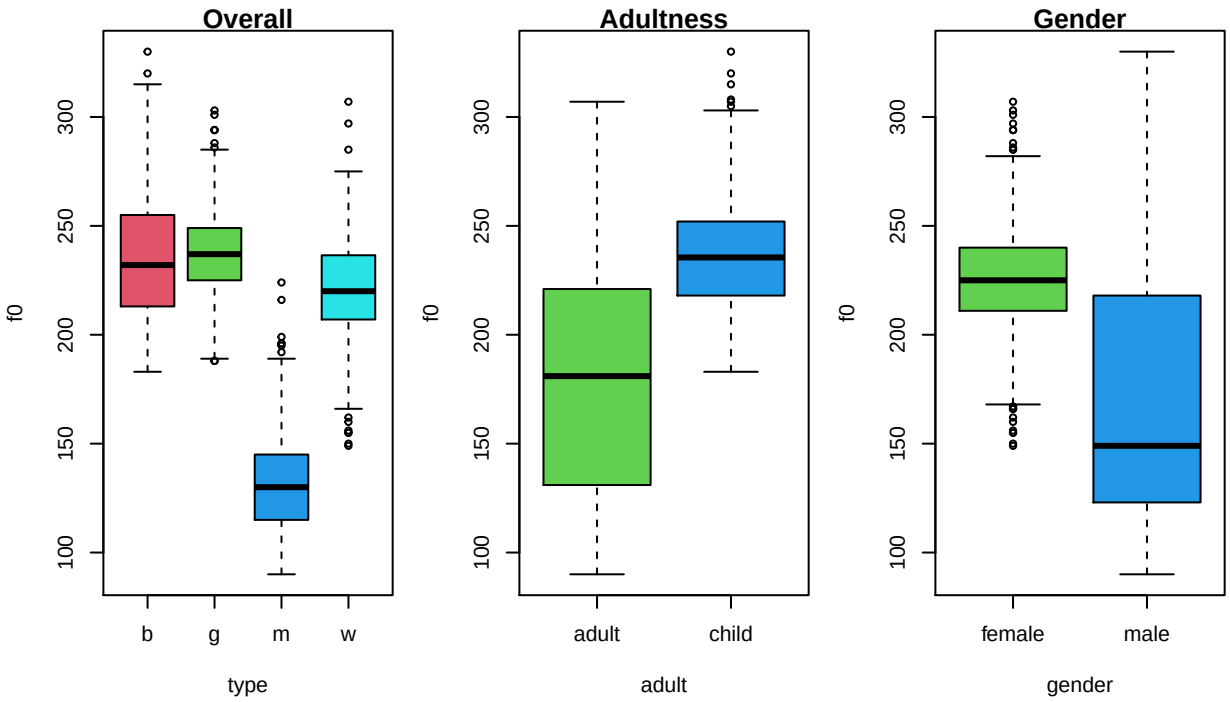
```
##
##          adult child
##   female   576   228
##   male     540   324
```

```
## make speaker number a factor
h95$uspeaker = factor (h95$uspeaker)
```

Our potential research questions are substantially more complicated. Not only are there four groups now, but they also differ along multiple dimensions. This means it is more difficult to make two-group comparisons that ask one single question. For example, the man-girl groups differ according to adultness *and* gender.

We can consider our data in several ways: as four independent groups, or as two 2-groups comparisons (adult vs child, female vs male).

```
par (mfrow = c(1,3))
boxplot (f0 ~ type, data = h95, main = "Overall", ylim = c(90,330), col = 2:5)
boxplot (f0 ~ adult, data = h95, main = "Adultness", ylim = c(90,330), col = 3:4)
boxplot (f0 ~ gender, data = h95, main = "Gender", ylim = c(90,330), col = 3:4)
```

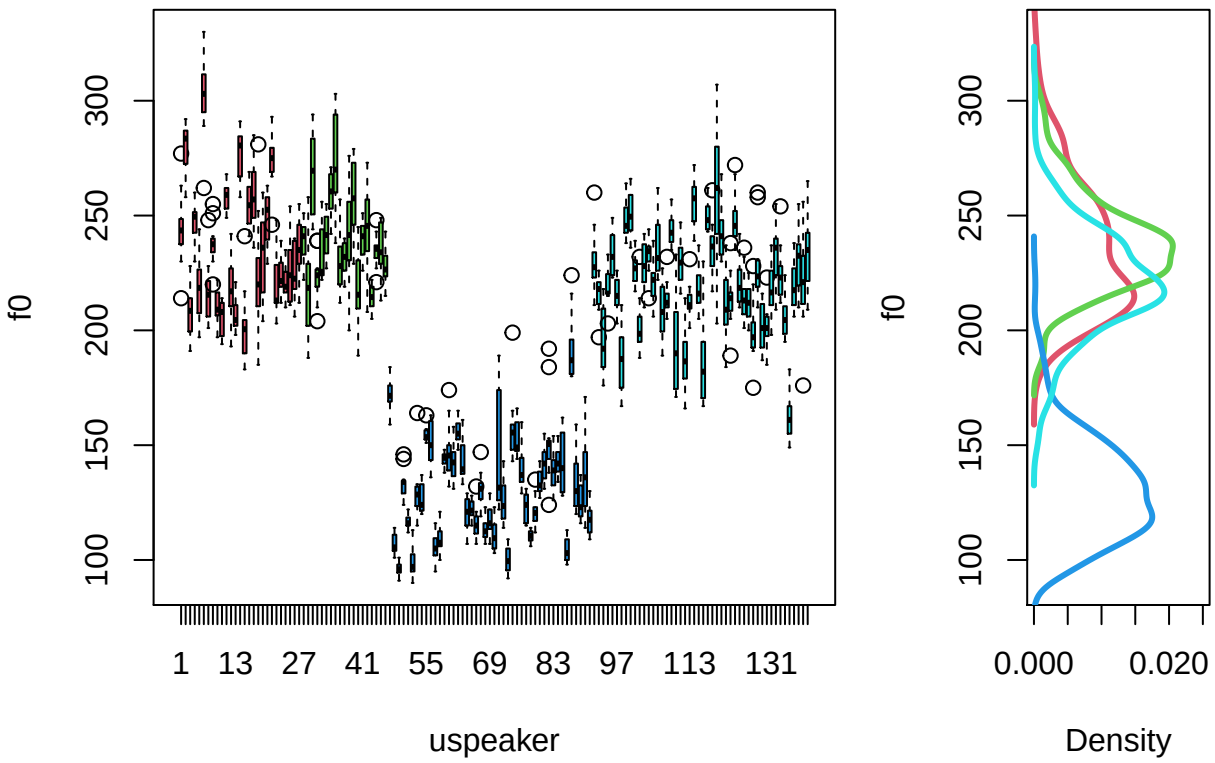


We are going to focus on the 4-way comparison first. Below, we compare between- and within-speaker variation in f0 by boys (red), girls (green), men (blu), and women (cyan). The figure on the right shows the densities of the overall distributions for each group.

```
colors = c(2,3,4,5)[ apply (table(h95$uspeaker, h95$type),1,which.max) ]

par (mfrow = c(1,2)); layout (mat = t(c(1,2)), widths = c(.7,.3))
boxplot (f0 ~ uspeaker, data=h95, col = colors, ylim = c(90,330))

## The density figures are rotated
tmp = density (h95$f0[h95$type=="b"])
plot (tmp$y, tmp$x, lwd = 3, col = 2, ylab = "f0",xlab="Density",
      ylim = c(90,330), xlim = c(0,0.025), type = 'l')
tmp = density (h95$f0[h95$type=="g"]); lines (tmp$y, tmp$x, lwd = 3, col = 3)
tmp = density (h95$f0[h95$type=="m"]); lines (tmp$y, tmp$x, lwd = 3, col = 4)
tmp = density (h95$f0[h95$type=="w"]); lines (tmp$y, tmp$x, lwd = 3, col = 5)
```



1.2 Comparing four (or any number of) groups

We are first going to treat the four groups as four groups with no internal structure.

1.2.1 The model

We are going to first fit a model that predicts f0 based on speaker type. Unlike in our last example, this is not just a single characteristic that can be represented with a single variable. Actually, the model needs (approximately) one variable per group, but you don't need to worry about this. R treats verbal predictors as 'factors' and assumes that each different label is a different group. Each group of a factor is called a 'level'. Our factor `type` has four levels: `b`,`g`,`m`, and `w`. For models where the predictor is a factor with more than two levels, the model is basically:

$variable \sim group_{[i]}$

where i is a counter variable that goes from 1 to the number of groups, in this case 4. However, we don't usually use a model like this. Instead, just as with the two-group model, the effects associated with different groups are expressed in relation to some reference value. By default, R uses treatment coding which selects a group to be the intercept and then compares each group to this one. In our case our model will use `g` (girls) as the Intercept. This means that our model will return four parameters, the intercept and the difference from each other group mean to the girl's mean. So actually, it is more like:

$variable \sim Intercept + group_{[i]}$, where i goes from $1...(n-1)$ for n total groups.

1.2.2 Fitting the model and interpreting the results

We can check out the mean and standard deviations for the data to set prior probabilities for the model parameters.

```
mean (h95$f0)
```

```
## [1] 197.027
```

```
sd (h95$f0)
```

```
## [1] 51.76277
```

And fit a model that incorporates this information.

```
options (contrasts = c("contr.treatment","cont.treatment"))
set.seed (1)
model =
  brm (f0 ~ type + (1|uspeaker), data = h95, chains = 4, cores = 4,
       warmup = 1000, iter = 6000, thin = 8,
       prior = c(set_prior("student_t(3, 195, 100)", class = "Intercept"),
                                set_prior("student_t(3, 0, 100)", class = "b"),
                                set_prior("student_t(3, 0, 100)", class = "sd")))
#  saveRDS (model, "model.RDS")
```

```
## load and inspect pre-fit model
model = readRDS ("model.RDS")
model
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: f0 ~ type + (1 | uspeaker)
##    Data: h95 (Number of observations: 1668)
## Samples: 4 chains, each with iter = 6000; warmup = 1000; thin = 8;
##          total post-warmup samples = 2500
##
## Group-Level Effects:
## ~uspeaker (Number of levels: 139)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)    20.89      1.30    18.53    23.54 1.00     1251     1840
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept   235.83      4.26   227.51   244.11 1.01      641     1309
## typeg         2.51      6.41   -10.06    15.24 1.01      829     1472
## typem      -104.48      5.54  -115.27   -93.75 1.02      620     1118
## typew       -15.47      5.30   -25.63    -5.35 1.01      432     1122
##
## Family Specific Parameters:
```

```
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma    11.87      0.21    11.45    12.28 1.00     2559     2440
##
## Samples were drawn 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).
```

We recover the estimated overall mean using the 'hypothesis function, and compare this to the actual group:

```
mean (h95$f0)
```

```
## [1] 197.027
```

```
hypothesis (model, "(Intercept*4 + typeg + typem + typew)/4 = 0")
```

```
## Hypothesis Tests for class b:
##                 Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1 ((Intercept*4+typ... = 0   206.47      1.94   202.61   210.41         NA
##   Post.Prob Star
## 1        NA    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

We can use the same function to recover the estimated group means, and compare these to the actual group means:

```
hypothesis (model, c("Intercept = 0",
                     "Intercept + typeg = 0",
                     "Intercept + typem = 0",
                     "Intercept + typew = 0"))
```

```
## Hypothesis Tests for class b:
##              Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob
## 1       (Intercept) = 0   235.83      4.26   227.51   244.11         NA        NA
## 2 (Intercept+typeg) = 0   238.34      4.87   228.60   247.57         NA        NA
## 3 (Intercept+typem) = 0   131.35      3.29   124.91   137.76         NA        NA
## 4 (Intercept+typew) = 0   220.36      3.08   214.51   226.36         NA        NA
##   Star
## 1    *
## 2    *
## 3    *
## 4    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
tapply (h95$f0, h95$type, mean)
```

```
##        b        g        m        w
## 236.0741 238.3509 131.2185 220.4010
```

In these Bayesian models, we can actually compare any groups we want in this model, using comparisons of the posterior samples (as shown in chapter 2). For example, the difference between girls and men can be found by asking of one minus the other equals 0 (which would be true if these were identical):

```
hypothesis (model, "typeg - typem = 0")
```

```
## Hypothesis Tests for class b:
##          Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob Star
## 1 (typeg-typem) = 0   106.99      5.92    95.31   118.51         NA        NA    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

1.2.3 An alternate approach: Sum coding

Under treatment coding, everything is compared to an arbitrary reference level. This can be bad for more complicated models, because sometimes effects can become difficult to interpret. Researchers often prefer what is called 'sum-'coding'. Under sum coding, the intercept of the model represents the overall average value. Then, the different group effects represent the difference to the overall mean.

Our model still looks like this:

$variable \sim Intercept + group_{[i]}$, where i goes from $1...(n-1)$ for n total groups.

However, the 'missing' group is actually not represented anywhere in the model. The last group from each factor is actually omitted from the model, but it is recoverable based on the effects that *are* present. The omitted level of any factor will have a value equal to:

$-(group_{[i]} + group_{[i+1]} + ... + group_{[n-1]})$

That is, the missing effect will be equal to the negative sum of the coefficients that are present. You just ass up all the coefficients you can see and flip the sign on it. By default, R drops the **last** level from your factor. In our case, it will drop the `w` level.

1.2.4 Fitting the model and interpreting the results: sum coding

We fit the same model but with sum coding this time.

```
options (contrasts = c("contr.sum","cont.sum"))
set.seed (1)
model_sum_coding =
  brm (f0 ~ type + (1|uspeaker), data = h95, chains = 4, cores = 4,
       warmup = 1000, iter = 6000, thin = 8,
       prior = c(set_prior("student_t(3, 195, 100)", class = "Intercept"),
```

```
                            set_prior("student_t(3, 0, 100)", class = "b"),
                            set_prior("student_t(3, 0, 100)", class = "sd")))
##  saveRDS (model_sum_coding, "model_sum_coding.RDS")
```

```
## load and inspect pre-fit model
model_sum_coding = readRDS ("model_sum_coding.RDS")
model_sum_coding
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: f0 ~ type + (1 | uspeaker)
##    Data: h95 (Number of observations: 1668)
## Samples: 4 chains, each with iter = 6000; warmup = 1000; thin = 8;
##          total post-warmup samples = 2500
##
## Group-Level Effects:
## ~uspeaker (Number of levels: 139)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)    20.92      1.29    18.56    23.58 1.00     1301     1785
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept   206.46      1.91   202.63   210.06 1.00      801     1491
## type1        29.63      3.50    22.81    36.46 1.00      623     1583
## type2        31.63      3.94    23.99    39.33 1.00      710     1294
## type3       -75.20      2.95   -81.41   -69.54 1.00      770     1010
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma    11.87      0.22    11.45    12.31 1.00     2650     2606
##
## Samples were drawn 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).
```

The Intercept clearly matches our previous mean effect. Since these effects are deviations from the overall mean, we can compare them to the same in our raw data:

```
hypothesis (model_sum_coding, c("type1 = 0",
                                "type2 = 0",
                                "type3 = 0",
                                "(type1+type2+type3) = 0"))
```

```
## Hypothesis Tests for class b:
##                 Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1              (type1) = 0    29.63      3.50    22.81    36.46         NA
## 2              (type2) = 0    31.63      3.94    23.99    39.33         NA
## 3              (type3) = 0   -75.20      2.95   -81.41   -69.54         NA
## 4 ((type1+type2+typ... = 0   -13.94      2.81   -19.43    -8.33         NA
##   Post.Prob Star
## 1        NA    *
## 2        NA    *
```

```
## 3        NA    *
## 4        NA    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
## grop means
tapply (h95$f0, h95$type, mean)
```

```
##        b        g        m        w
## 236.0741 238.3509 131.2185 220.4010
```

```
## centered means
tapply (h95$f0, h95$type, mean) - mean (h95$f0)
```

```
##         b         g         m         w
##  39.04710  41.32390 -65.80846  23.37406
```

We can compare the effects estimated by the two models in their natural state and they seem quite different. None of the parameters match at all:

```
fixef (model)
```

```
##              Estimate Est.Error       Q2.5      Q97.5
## Intercept  235.829201  4.264990  227.51347 244.106840
## typeg        2.508074  6.407977  -10.05737  15.244656
## typem     -104.481059  5.540981 -115.26785 -93.753055
## typew      -15.469124  5.301906  -25.63031  -5.353257
```

```
fixef (model_sum_coding)
```

```
##            Estimate Est.Error      Q2.5     Q97.5
## Intercept 206.46403  1.909610 202.62966 210.05696
## type1      29.62692  3.504781  22.80578  36.46396
## type2      31.63321  3.938962  23.99464  39.33010
## type3     -75.19547  2.948541 -81.41356 -69.54104
```

However, if we transform them appropriately, we can see that they actually are giving us the same information:

```
hypothesis (model, c("(Intercept*4 + typeg + typem + typew)/4 = 0",
                     "Intercept = 0",
                     "Intercept + typeg = 0",
                     "Intercept + typem = 0",
                     "Intercept + typew = 0"))
```

```
## Hypothesis Tests for class b:
##                 Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
```

```
## 1 ((Intercept*4+typ... = 0   206.47      1.94   202.61   210.41         NA
## 2          (Intercept) = 0   235.83      4.26   227.51   244.11         NA
## 3    (Intercept+typeg) = 0   238.34      4.87   228.60   247.57         NA
## 4    (Intercept+typem) = 0   131.35      3.29   124.91   137.76         NA
## 5    (Intercept+typew) = 0   220.36      3.08   214.51   226.36         NA
##   Post.Prob Star
## 1        NA    *
## 2        NA    *
## 3        NA    *
## 4        NA    *
## 5        NA    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
hypothesis (model_sum_coding, c("Intercept = 0",
                                "Intercept + type1 = 0",
                                "Intercept + type2 = 0",
                                "Intercept + type3 = 0",
                                "Intercept - (type1+type2+type3) = 0"))
```

```
## Hypothesis Tests for class b:
##                 Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio
## 1          (Intercept) = 0   206.46      1.91   202.63   210.06         NA
## 2    (Intercept+type1) = 0   236.09      4.10   228.03   244.19         NA
## 3    (Intercept+type2) = 0   238.10      4.84   228.62   247.51         NA
## 4    (Intercept+type3) = 0   131.27      3.17   124.75   137.58         NA
## 5 (Intercept-(type1... = 0   220.40      2.94   214.76   226.16         NA
##   Post.Prob Star
## 1        NA    *
## 2        NA    *
## 3        NA    *
## 4        NA    *
## 5        NA    *
## ---
## 'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
## '*': For one-sided hypotheses, the posterior probability exceeds 95%;
## for two-sided hypotheses, the value tested against lies outside the 95%-CI.
## Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
set.seed (1)
model_adult =
  brm (f0 ~ adult + (1|uspeaker), data = h95, chains = 4, cores = 4,
       warmup = 1000, iter = 6000, thin = 8,
       prior = c(set_prior("student_t(3, 195, 100)", class = "Intercept"),
                                set_prior("student_t(3, 0, 100)", class = "b"),
                                set_prior("student_t(3, 0, 100)", class = "sd")))
saveRDS (model_adult, "model_adult.RDS")
```

```
## load pre-fit model
model_adult = readRDS ("model_adult.RDS")
```

```
set.seed (1)
model_gender =
  brm (f0 ~ gender + (1|uspeaker), data = h95, chains = 4, cores = 4,
       warmup = 1000, iter = 6000, thin = 8,
       prior = c(set_prior("student_t(3, 195, 100)", class = "Intercept"),
                                set_prior("student_t(3, 0, 100)", class = "b"),
                                set_prior("student_t(3, 0, 100)", class = "sd")))
saveRDS (model_gender, "model_gender.RDS")
```

```
## load pre-fit model
model_gender = readRDS ("model_gender.RDS")
```

```
set.seed (1)
model_both =
  brm (f0 ~ adult + gender + (1|uspeaker), data = h95, chains = 4, cores = 4,
       warmup = 1000, iter = 6000, thin = 8,
       prior = c(set_prior("student_t(3, 195, 100)", class = "Intercept"),
                                set_prior("student_t(3, 0, 100)", class = "b"),
                                set_prior("student_t(3, 0, 100)", class = "sd")))
saveRDS (model_both, "model_both.RDS")
```

```
## load pre-fit model
model_both = readRDS ("model_both.RDS")
```

```
set.seed (1)
model_interaction =
  brm (f0 ~ adult * gender + (1|uspeaker), data = h95, chains = 4, cores = 4,
       warmup = 1000, iter = 6000, thin = 8,
       prior = c(set_prior("student_t(3, 195, 100)", class = "Intercept"),
                                set_prior("student_t(3, 0, 100)", class = "b"),
                                set_prior("student_t(3, 0, 100)", class = "sd")))
saveRDS (model_interaction, "model_interaction.RDS")
```

```
## load pre-fit model
model_interaction = readRDS ("model_interaction.RDS")
```