

Hands-on non-technical tutorial for Bayesian mixed effects regression

Michael Franke & Timo Roettger

Generalized linear mixed models are very versatile and handy tools for statistical inference. Bayesian approaches to applying these models have recently become increasingly popular. This tutorial provides an accessible, non-technical introduction to the use and feel of Bayesian mixed effects regression models. The focus is on data from a factorial-design experiment.

This tutorial should take you about 1 hour.

Motivation & intended audience

This tutorial provides a very basic introduction to Bayesian regression modeling using R (R Core Team, 2017). We wrote this tutorial with a particular reader in mind. If you have used R before and if you have a basic understanding of linear regression and now you want to find out what a Bayesian approach has to offer, this tutorial is for you. In comparison to other introductions (e.g. Sorensen, Hohensteinb, and Vasishth, 2016), this tutorial remains very conceptual. We don't want to "sell Bayes" to you, and we do not want to scare you away with mathematical details. We just want to give you an impression of how a Bayesian regression analysis looks and feels. So no reason to be afraid! But also: no reason to be bored, because we *will* cover all the essential concepts and we *will* explain how to run and interpret the output of a Bayesian regression analysis using the wonderful R package `brms` written by Paul Buerkner (2016).

If you don't have any experience with regression modeling, you will probably still be able to follow but you might also want to consider doing a crash course. To bring you up to speed, we recommend the excellent two-part tutorial by Bodo Winter (2013) on mixed effects regression in a non-Bayesian—a.k.a. classical or frequentist—paradigm. In a sense, this tutorial could be considered part three of Bodo's nice and lofty introduction. We will, for example use the same data set.

This tutorial contains text boxes (with a gray background) which contain additional background information on some topics. The information is sometimes a bit technical but never absolutely necessary for understanding the main ideas. So, feel free to read or skip any of the text boxes to suit your needs.

To follow this tutorial, you should have R installed on your computer (<https://www.r-project.org>). Unless you already have a favorite editor for tinkering with R scripts, we recommend to try out RStudio (<https://www.rstudio.com>). You will also need some packages,

Remember that you can install a package called XYZ with the command `install.packages("XYZ")`.

which you can import with the following code:

```

1 #####
2 ## package includes and options
3 #####
4
5 # package for convenience functions (e.g. plotting)
6 library(tidyverse)
7
8 # package for Bayesian regression modeling
9 library(brms)
10 # option for Bayesian regression models: use all available cores
    for parallel computing
11 options(mc.cores = parallel::detectCores())
12
13 # set seed
14 set.seed(1702)

```

If you do not want to copy-paste, all code and data for this tutorial is also available for download here: https://github.com/michael-franke/bayes_mixed_regression_tutorial

Data, research questions & hypotheses

Imagine we are experimental researchers. Therefore, we collect data to answer questions of interest about how nature works. For example, we might want to know whether voice pitch differs across female and male speakers, and whether it differs across social contexts (say: informal and polite contexts). — To answer our questions, we come up with a nifty experimental design; we lure a group of people into the lab; we ask them to say different words in different social contexts; we record their voices; and we extract numbers from these recordings, for example, the pitch values of their voices. We then want to find out whether our data provide evidence for any assumed relationships. So far so good.

In this tutorial, we are looking at exactly these kind of data (following Winter, 2013). To load the data into your R environment, run the following code:

```

1 # load the data into variable 'politedata'
2 politedata = read_csv("https://raw.githubusercontent.com/
    michael-franke/bayes_mixed_regression_tutorial/master/code/
    politeness_data.csv")

```

The data is originally from research presented by Winter and Grawunder (2012)

If you are familiar with the previous tutorials by Winter (2013), note that we ‘massaged’ the data a bit, i.e., we renamed variables and removed a line with missing data.

Type `head(politedata)` and you should see the first lines of the imported data:

Here, we show only part of the output that you can see when executing this command.

```

1 > head(politedata)
2   subject gender sentence context pitch
3   <chr>   <chr>  <chr>   <chr>  <dbl>
4 1 F1      F      S1      pol    213.
5 2 F1      F      S1      inf    204.
6 3 F1      F      S2      pol    285.
7 4 F1      F      S2      inf    260.
8 5 F1      F      S3      pol    204.

```

This data set contains information about different subjects, with an anonymous identifier stored in variable `subject`. Because voice pitch is highly dependent on gender (i.e. there are anatomical differences between women and men that affect voice pitch), we stored whether our subjects are F(emale) or M(ale) in variable `gender`. Subjects produced different sentences (stored in variable `sentence`), and the experiment manipulated whether the sentence was produced in a polite or an informal context, indicated by the variable `context`. Crucially, each row contains a measurement of pitch in Hz stored in variable `pitch`.

Often, we are interested in comparing a **dependent variable** (here `pitch`) across different conditions or groups, i.e. **independent variables** (here `gender` and `context`). Before our data collection, we might have formulated concrete predictions about the relationship between the dependent variable and the independent variables. For example, we might have formulated the following three hypotheses:

- H1: Female speakers have a lower average pitch in polite than in informal contexts.
- H2: Male speakers have a lower average pitch in polite than in informal contexts.
- H3: Male speakers have a lower average pitch in informal than female speakers have in polite contexts.

Exploring the data visually

To get a first idea of possible relationships in our data, let's plot them. Figure 1 displays the mean pitch values for each sentence (semi-transparent points) across gender and context. The solid points indicate the average pitch values across all subjects. Looking at the plot, we can see that pitch values from female speakers are generally higher than those from male speakers (points in left column are higher than in the right column). We also see that pitch values in the informal context are slightly higher than those in the polite context (blue points are slightly higher than orange points).

[mf: About Figure 1, could we make sure the plot is readable in gray-scale printing as well?]

Notice that our hypotheses are formulated explicitly as comparisons of means / averages. The statistical model we will use indeed compares means, and this is common practice, albeit a particular assumption worth highlighting. Commonly, this assumption is implicit, with e.g. H1 being simply formulated as something like "Female speakers' pitch is lower in polite than informal contexts."

Extensive plotting is always recommended to start data analysis. You need to know your data inside out. Pictures often reveal complex relationships much better than numbers can.

The code needed to generate the picture in Figure 1 is not reproduced here, but included in the script in the resources for this tutorial: https://github.com/michael-franke/bayes_mixed_regression_tutorial

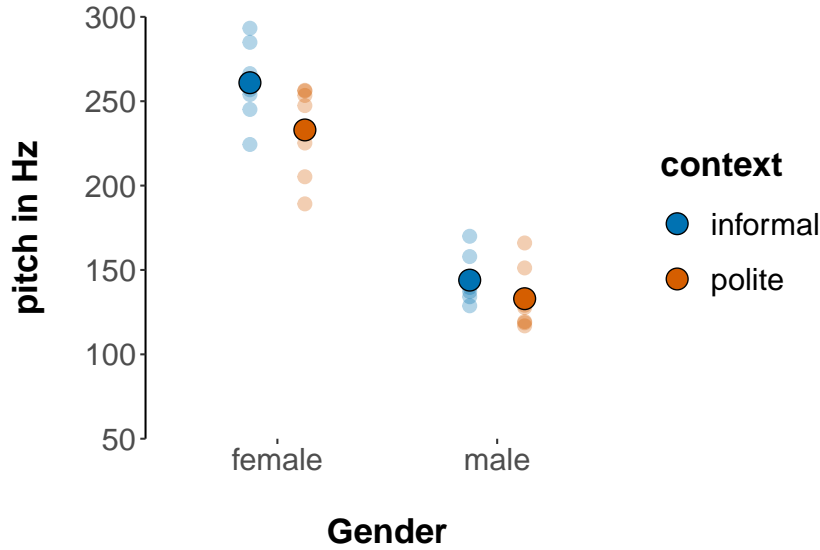


Figure 1: Basic plot of the data displaying overall averages (thick points) and averages for individual sentences (smaller semitransparent points)

Based on keen eye-balling, we might want to shout: “Eureka! The data confirm all of our hypotheses!” But, of course, we need to be more careful. As Bayesians, we would like to translate the data into an expression of **evidence**: does the data provide evidence for our research hypotheses? Or are the observable differences to meager? – Also, notice that there is quite a lot of variability between different sentences (the semi-transparent dots in Figure 1). For example, some values from the informal condition for female speakers (blue points in left column), are lower than their corresponding polite counterparts. Similarly, there could be quite some differences between individual speakers. Consequently, what we want is precise estimates of potential differences between conditions, alongside a measure of how confident we can be in these estimates.

A regression model for our data

Another way of looking at the data in connection with our research hypotheses is displayed in Figure 2. Each cell represents one unique combination of the gender and the context factor, and the table shows the mean pitch value for each cell. Our hypotheses can be related to the comparison between some of these cell-based means. H1 makes a statement about the comparison between cells 1 and 2 (the context effect for female speakers); H2 makes a statement about cells 3 and 4 (the context effect for male speakers); and H3 makes a statement about cells 2 and 3 (the difference between informal male speakers and polite female speakers).

One way of testing our hypotheses using a Bayesian approach to data analysis, is to ask whether the relevant differences between cell means are

In technical terms, this table is the **design matrix** of our experiment. We have two factors of interest **context** and **gender**, each with two levels. The table shows each combination of levels of all relevant factors. The cells in this table are therefore also called **design cells**.

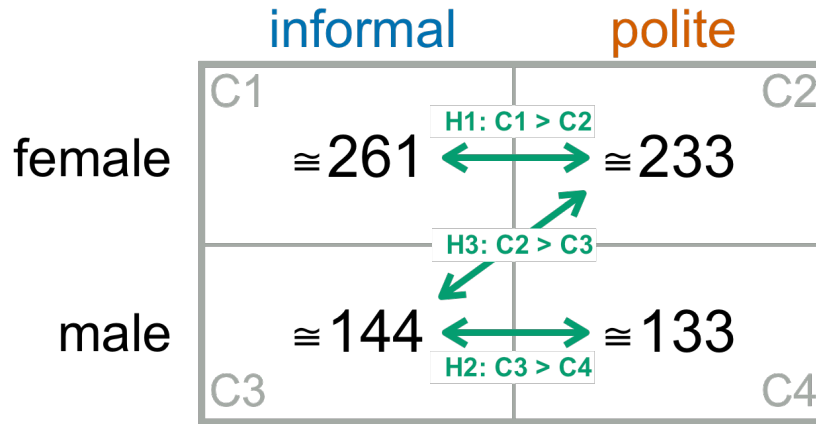


Figure 2: Means of each design cell, together with research hypotheses as statements about ordinal relations between cell means.

credibly different from zero. This is jargon for asking whether, given the observed data, we should believe that the relevant cell means are different from each other. The fact that we are now in Bayesian land should hit us when we hear the phrases “credibly different” and the “should believe”. The Bayesian approach is about updating beliefs based on observations. And we express our beliefs as as probability distributions. At first, this may appear scary or technically involved. But at the end of the day, the intuitions captured by this approach are arguably very natural, and perhaps easier to understand than the reasoning underlying other approaches to statistical inference. Let’s walk through our Bayesian approach step by step.

First, let us look at the **regression model** we want to use. Our regression models assumes that pitch values observed in each cell are sampled from a population that is normally distribution, where each cell c_i has its own mean μ_i . We are interested in the probability of one cell mean being larger than another cell mean, i.e., the probability that $\mu_i > \mu_j$. Put differently, we are interested in the probability that the difference between μ_i and μ_j is larger than zero: $\mu_i - \mu_j > 0$. Figure 3 illustrates the encoding scheme of our cell means in terms of a regression analysis. It assumes that there is a **reference level** for each factor. Here it is the level *female* for the factor *gender* and the level *informal* for the factor *context*.

All cell means can then be expressed in terms of differences between the **intercept** β_0 which is the cell mean of the reference level (here, the cell mean of female subjects in informal contexts) and deviations from this **reference cell** for each individual factor (β_{male} , and β_{polite}), and a so-called **interaction term** $\beta_{\text{pol\&male}}$. In other words, our regression estimates the mean of the

This is so-called **dummy coding** of the regression coefficients. Other coding schemes exist, but are not discussed here.

reference level and estimates how much we need to adjust this mean when we change either the context level (C2), the gender level (C3), or both (C4). Every value that the model estimates is called a parameter.

	informal	polite
female	C1 β_0 reference level	C2 $\beta_0 + \beta_{\text{pol}}$
male	C3 $\beta_0 + \beta_{\text{male}}$	C4 $\beta_0 + \beta_{\text{male}} + \beta_{\text{pol}} + \beta_{\text{pol\&male}}$

Figure 3: Coefficients of a dummy-coded regression model for the factorial 2×2 design.

A Bayesian analysis of a (fixed effects) regression model

Having spelled out a model like the above, a Bayesian analysis asks: what should we believe about the values of the coefficients β_0 , β_{pol} , β_{male} and $\beta_{\text{pol\&male}}$?; what values for these parameters are likely, given the data, the assumed model, and our initial beliefs about the parameters, the so-called **prior beliefs**?

The R package `brms` (Buerkner, 2016) makes it easy to run Bayesian regression models. It uses a very similar formula syntax as related packages for regression analysis. In our case, we want to regress the dependent variable `pitch` against the independent variables `gender` and `context` and their two-way interaction. This model is expressed by the formula:

```
1 # formula for (fixed effects) regression model
2 formulaFE = pitch ~ gender * context
```

Prior beliefs are important to get a Bayesian analysis off the ground; a circumstance which is discussed controversially. For many practical purposes, however, the precise choice of prior is not decisive and tools like the `brms` package which we will use here will default to generically reasonable choices of priors for your model (more on this below).

Info Box 1 provides some background on prior beliefs, likelihood function and posterior beliefs.

The Bayesian model can then be fitted with the function `brm` from the `brms` package. We only need to specify the formula and supply the data:

```

1 # run regression model in brms
2 modelFE = brm(
3   formula = formulaFE,
4   data = politedata
5 )

```

The `brms` package uses the probabilistic programming language `Stan` in the background. What `brms` does is translating your syntax into `Stan` code and executes it. The `Stan` code is then translated to C++ (hence the message about “compiling C++” when you run this code). Conceptually, `Stan` obtains samples from the posterior distribution, based on an algorithm called *Hamiltonian Monte Carlo*. This is an instance of a more general class of algorithms, called *Markov Chain Monte Carlo* methods. The purpose of these methods is to return representative samples from the posterior distribution (without actually having to compute the exact distribution, which might be intractable) [tr: I suggest deleting the brackets. Difficult to understand and not relevant].

You can type in `modelFE` in order to get a summary of the model fit. It should look much like the following output (note that your values might be slightly different due to stochastic variation in the sampling procedure).

```

1 > modelFE
2 Family: gaussian
3 Links: mu = identity; sigma = identity
4 Formula: pitch ~ gender * context
5 Data: politedata (Number of observations: 83)
6 Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
7         total post-warmup samples = 4000
8
9 Population-Level Effects:
10      Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
11 Intercept          260.68      8.07   244.99   276.78     2409 1.00
12 genderM           -116.09     11.44  -138.37   -93.80     2094 1.00
13 contextpol        -27.38     11.33   -50.01    -5.98     2092 1.00
14 genderM:contextpol   15.74     16.38   -17.03    49.13     1831 1.00
15
16 Family Specific Parameters:
17      Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
18 sigma        36.15      2.87    30.93    42.40     3652 1.00
19
20 Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
21 is a crude measure of effective sample size, and Rhat is the potential
22 scale reduction factor on split chains (at convergence, Rhat = 1).

```

This summary looks very much like regression model summaries of non-Bayesian model. Lines 2–5 give us information about the model and the data used. Lines 6 and 7 tell us about the sampling procedure. We have a total of 4000 samples from the posterior distribution, obtained from 4 chains all of which had 2000 iterations but discarded the first 1000 as warmup (more

on this below) [tr: chains, iterations, warm-up need to be elaborated]. Lines 9–14 contain information about our parameters of interest, i.e. what we are most interested in for evaluating our hypotheses. We will discuss them in detail below. Lines 16–18 contain information that look similar to those in lines 9–14. This is the estimation of the standard deviation `sigma`, describing the variance of the assumed normal distributions (which describe the distribution of measures in each design cell). Finally, lines 20–22 contain general information about the model fit and the information presented in this summary.

Let us look at lines 9–14 in detail now. What these lines give us is a table with four rows, each of which corresponds to a parameter in the model, namely the coefficients shown in Figure 3. The variable `Intercept` refers to our β_0 , which represents the mean of our reference level in cell 1 (female speakers in polite contexts). The variable `genderM` corresponds to our β_{male} , `contextpol` corresponds to our β_{pol} , and `genderM:contextpol` is the interaction term $\beta_{\text{pol}\&\text{male}}$. For each of these parameters, the table contains very useful summary statistics based on the samples returned from the model fit. Here we are interested in the information in columns `Estimate`, that is the estimate of each parameter's mean. We are also interested in `l-95% CI` and `u-95% CI`, which give us the lower and upper bound of the **95% credible interval** for each parameter. More details about the information given in the other columns can be found in Info Box 2.

For our purposes, the information about 95% credible intervals is most interesting. Take the parameter `contextpol`, corresponding to our coefficient β_{pol} . This parameter corresponds to the estimated adjustment of the mean of the reference level when we change the context level to polite. The 95% CI is roughly [-50;-6]. This means that we would take values outside of this interval to be sufficiently unlikely to consider plausible values. This means that a very special parameter value for this parameter is implausible, namely 0. In other words, this analysis suggests that we should not believe that 0 is a very likely value for the coefficient β_{pol} ; rather we should believe that β_{pol} is most likely negative. — Hurray! This directly addresses our first research hypothesis. In a research paper we could now write: “Based on the regression model, the data suggests that H1 is likely true.”

How likely is it that β_{pol} is smaller than 0? —Instead of simply making a binary thumbs-up / thumbs-down decision, it would be even cooler, if we could put a number to it. As Bayesians, we fortunately can. To see how this works, let us have a more intimate look at the samples that the `brm` function returns. We can access the samples of a model fitted with `brm` with the function `posterior_samples`:

```
1 # extract posterior samples
2 post_samples_FE = posterior_samples(modelFE)
3 head(post_samples_FE)
```

If the model failed to converge or other problems occurred, you would see an informative message in the last part of this summary.

Intuitively, the 95% credible interval is the range of values that we can often practically consider credible enough to care about.

The output of this could look like this:

```
1 > head(post_samples_FE)
2   b_Intercept b_genderM b_contextpol b_genderM:contextpol   sigma   lp__
3 1    255.3   -106.4    -24.7          15.9 34.3 -420.1
4 2    252.5   -118.7    -15.7          22.7 35.4 -421.2
5 3    254.1   -119.8    -15.9          15.5 35.7 -420.6
6 4    270.1   -114.0    -33.7          26.9 36.3 -423.1
7 5    275.4   -122.9    -37.1          23.5 37.5 -422.0
8 6    281.4   -135.4    -43.4          25.5 38.9 -423.3
```

What you see here is the top 6 rows of a data frame with columns for each parameter and 4000 rows, corresponding to each sample of that parameter (so our sampling method has generated 4000 likely values for each parameter). We can use these samples to produce a density plot. The plot in Figure 4 shows, for each of the four main model parameters an estimate of the posterior density. Each curve shows how much credence we should put on particular parameter values. For example, we see that our beliefs concerning plausible values for the mean of cell 1 (female speakers in polite contexts, the reference cell) should hover around 261, spreading from about 240 to 280. We also see that all values that receive substantial probability density for `contextpol` (our β_{pol}) are negative (as captured in the 95% CI discussed above). Zero is estimated to be a rather unlikely value for this parameter.

The column `lp__` contains the log-probability of the data for the parameterization in each row. This is useful for model comparison and model criticism but not important for our current adventures.

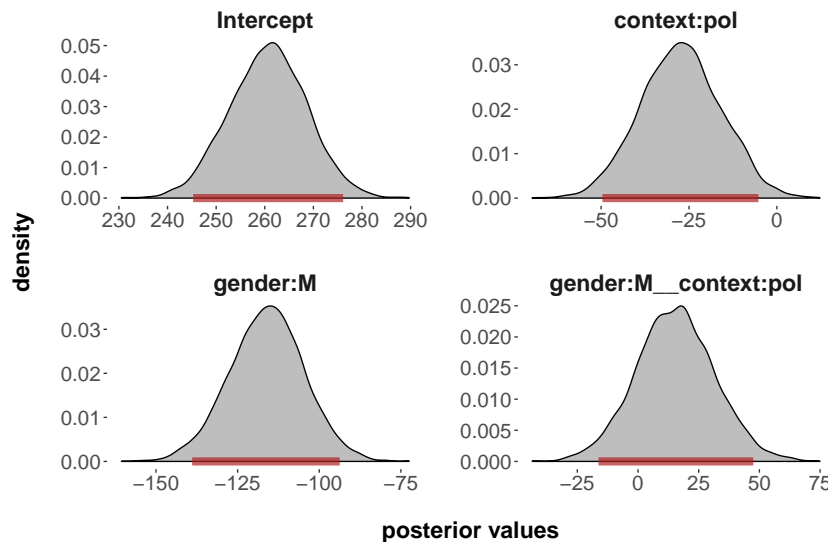


Figure 4: Posterior density of parameter values in the fixed-effects regression model. The thick red lines indicate the 95% credible intervals, i.e., the range of parameter values that it is reasonable to believe in.

Now, here comes a nice gadget. Based on the samples obtained for `contextpol` (β_{pol}), it is very easy to estimate our belief that β_{pol} is indeed negative. We simply have to calculate the proportion of samples that were negative, that's all. For instance, with the code below, which reveals that the posterior probability, given the data, that $\beta_{\text{pol}} < 0$ is about 0.99275, so

very close to 1!

```
1 # proportion of negative samples for parameter p_contextpol
2 # this number approximates P(beta_pol < 0 | model, data)
3 mean(post_samples_FE$b_contextpol < 0)
```

As an interim summary, we have seen how to run a Bayesian regression analysis with the `brms` package and deal with its output. We have also seen that the output can be interpreted in very intuitive ways (e.g., “The probability of H1, given our model, priors, and data, is more than .99”).

Unfortunately, what we have not seen yet is what our model and data say about hypotheses 2 or 3. This is because there is no single parameter in the (dummy-coded) regression model that corresponds to the differences between cells 3 and 4 (for hypothesis 2) and cells 2 and 3 (for hypothesis 3). Notice that this problem is not specific to Bayesian analyses, but inherent in the way the regression coefficients were set up. However, unlike in more frequentist/classical analysis, the Bayesian approach allows to recover information about any derived measure from the obtained samples. Here’s how:

Take hypothesis 3 which requires us to compare cells 2 and 3. The hypothesis states that $\beta_0 + \beta_{\text{pol}} > \beta_0 + \beta_{\text{male}}$, which reduces to $\beta_{\text{pol}} > \beta_{\text{male}}$. We can approximate the posterior probability that this is true based on the samples that we obtained for our model in the same general way as before, namely [tr: I think this would be easier to follow if we’d keep the reference level in the equation] [mf: aaargh, I cannot do it; that’s rock-bottom for me; ;-) it’s explained in the paragraph above why we *can* leave it out; please ...]:

```
1 # proportion of samples where the mean for cell 2 was bigger
2 # than that of cell 3
3 # this number approximates P(beta_pol > beta_male | model, data)
4 mean(post_samples_FE$b_contextpol > post_samples_FE$b_genderM)
```

Based on the posterior samples we obtained, this estimate is 1. That’s a strong result. If the model were true, then, given the data, our certainty that hypothesis 3 is true should be pretty much almost at ceiling.

To conclude this section, the Bayesian approach to regression modeling allows us to retrieve all direct comparisons between cells in a factorial design (all possible comparisons). It also allows us to retrieve quantitative information about our hypotheses which is accessible, and easy to communicate and understand. We can calculate (estimated) posterior probability that a particular hypothesis holds.

The `faintr` package

To make the comparison of pairs of cells even easier and applicable for even bigger factorial designs, this tutorial comes with a little R package,

A potential way of testing different hypotheses of the kind we have set out here, is to run different regression analyses, each with a different reference cell. This is a rather unhandy work flow. It wouldn’t help with hypothesis 3 either, which compares the cell means in our design matrix “diagonally”: there is no way of changing the reference level of either factor such that dummy coding gives us a single coefficient as the difference between cells 2 and 3.

the `faintr` package. You can install the package from GitHub with the `devtools` package, as follows:

```
1 # package to allow installation from github
2 library(devtools)
3 # package with convenience function for Bayesian regression
  models for factorial designs
4 install_github("michael-franke/bayes_mixed_regression_tutorial/
  faintr") # install from GitHub
5 library(faintr)
```

The name `faintr` is indicative of the possibility that the package might break down unexpectedly (we might consider renaming after more extensive testing), but also alludes to “factorial design” and “interpretation” somehow.

The `faintr` package provides two (hopefully) useful functions. The function `extract_posterior_cell_means` takes as input the output of a factorial-design regression model fitted with `brm`. It outputs samples for all design cell means, and a comparison of all design cells against each other. The function `get_cell_comparison` takes the same kind of model fit as input, together with a specification of which two cells to compare against each other. Using the latter function, the source code provides a convenient function to produce the posterior probability of the three hypothesis relevant for this tutorial:

```
1 > get_posterior_beliefs_about_hypotheses_new(modelFE)
2 # A tibble: 3 x 2
3   hypothesis                                probability
4   <chr>                                     <dbl>
5 1 Female-polite < Female-informal          0.993
6 2 Male-polite < Male-informal              0.842
7 3 Male-informal < Female-polite            1
```

Adding random effects

In our experiment, we measured pitch multiple times for each subject (since they produced multiple sentences). We also have multiple measures for each sentence (as each sentence was produced by multiple speakers). A crucial assumption of linear regression models is the independence assumptions. Many before us have covered this aspect of linear models before us (e.g. Winter, 2013; Clark, 1973) so we won’t discuss this important issue here. In a nutshell, we need to inform our model about these dependencies between observations. The way we’re going to handle this is to add random effects to our model, just as we do in frequentist frameworks. Random effects are additional parameters that our Bayesian model estimates.

Although the results look quite different, running hierarchical models with random effects with `brms` is very similar to the look and feel of non-Bayesian approaches. Here is a function call to a model with the maximal random effect structure licensed by the design:

```

1 # hierarchical model with the maximal RE structure licensed by
  the design
2 # (notice that factor 'gender' does not vary for a given value
  of variable 'subject')
3 model_MaxRE = brm(formula = pitch ~ gender * context +
4                    (1 + gender * context | sentence) +
5                    (1 + context | subject),
6                    data = politedata,
7                    control = list(adapt_delta = 0.9))

```

The outcome of this is model fit is shown here:

```

1 > model_MaxRE
2 Family: gaussian
3 Links: mu = identity; sigma = identity
4 Formula: pitch ~ gender * context + (1 + gender * context | sentence) + (1 + context | subject)
5 Data: politedata (Number of observations: 83)
6 Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
7         total post-warmup samples = 4000
8
9 Group-Level Effects:
10 ~sentence (Number of levels: 7)
11
12      Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
13 sd(Intercept)      21.84    9.72   7.40  44.90      2116 1.00
14 sd(genderM)        11.13    9.07   0.49  33.29      2435 1.00
15 sd(contextpol)     15.04   11.24   0.55  42.49      1637 1.00
16 sd(genderM:contextpol) 16.55   13.43   0.76  47.63      2173 1.00
17 cor(Intercept,genderM) -0.23    0.44  -0.90   0.71      4808 1.00
18 cor(Intercept,contextpol) 0.01    0.41  -0.74   0.79      4658 1.00
19 cor(genderM,contextpol) -0.06    0.44  -0.83   0.77      2940 1.00
20 cor(Intercept,genderM:contextpol) -0.10   0.43  -0.84   0.73      5216 1.00
21 cor(genderM,genderM:contextpol) -0.03    0.44  -0.82   0.80      3335 1.00
22 cor(contextpol,genderM:contextpol) -0.15    0.44  -0.87   0.74      3057 1.00
23
24 ~subject (Number of levels: 6)
25
26      Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
27 sd(Intercept)      36.11   18.30  14.60  84.84      1210 1.00
28 sd(contextpol)       9.17    8.73   0.32  32.53      2290 1.00
29 cor(Intercept,contextpol) 0.03    0.58  -0.93   0.95      4608 1.00
30
31 Population-Level Effects:
32
33      Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
34 Intercept      261.92   25.73  213.11  312.17      1088 1.00
35 genderM       -116.78   35.19 -188.21  -50.38      1011 1.00
36 contextpol     -27.16   12.44  -51.30   -2.22      2782 1.00
37 genderM:contextpol 15.13   16.84  -17.62   47.40      2863 1.00
38
39 Family Specific Parameters:
40      Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
41 sigma      24.96    2.38   20.73   30.07      3676 1.00
42
43 Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
44 is a crude measure of effective sample size, and Rhat is the potential
45 scale reduction factor on split chains (at convergence, Rhat = 1).

```

The lines 29–34 again give the estimates of the fixed-effects coefficients.

The mean estimates look very similar to the ones that we obtained in the fixed-effect only model above. However, not surprisingly, our uncertainty surrounding these estimates is larger. We now also get information about the parameters implied by the specified random effect structure. Lines 9–21 cover the by-sentence random effects, lines 23–27 cover the by-subject random effects. We see from the 95% credible intervals that the only parameters implied by the random effects structure that does seem to receive sufficient a posteriori credence on non-trivial values are the by-sentence and by-subject random intercepts. [tr: wait? I don't understand. Where do you get this from? The random intercepts and slopes are SDs, meaning they are bound to zero, so from the CIs, you cannot infer that they are relevant or not. You can for the correlation terms, however, they are rarely systematic] [mf: I think you can infer that they are not noticeably different from zero; but I agree this is a subtle point and maybe not to be touched here]

To check the probability of our hypotheses of interest, we can use the `faintr` package again:

```
1 > get_posterior_beliefs_about_hypotheses_new(model_MaxRE)
2 # A tibble: 3 x 2
3   hypothesis                                probability
4   <chr>                                     <dbl>
5 1 Female-polite < Female-informal          0.981
6 2 Male-polite < Male-informal              0.830
7 3 Male-informal < Female-polite            0.988
```

Priors

... in progress ...

References

- Buerkner, Paul-Christian (2016). “brms: An R package for Bayesian multi-level models using Stan”. In: *Journal of Statistical Software* 80.1, pp. 1–28.
- Clark, Herbert H (1973). “The language-as-fixed-effect fallacy: A critique of language statistics in psychological research”. In: *Journal of verbal learning and verbal behavior* 12.4, pp. 335–359.
- R Core Team (2017). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. URL: <https://www.R-project.org/>.
- Sorensen, Tanner, Sven Hohensteinb, and Shravan Vasishth (2016). “Bayesian linear mixed models using Stan: A tutorial for psychologists, linguists, and cognitive scientists”. In: *The Quantitative Methods for Psychology*.

Winter, Bodo (2013). *Linear models and linear mixed effects models in R with linguistic applications*. URL: <https://arxiv.org/abs/1308.5499>.

Winter, Bodo and S. Grawunder (2012). “The Phonetic Profile of Korean Formality”. In: *Journal of Phonetics* 40, pp. 808–815.

Bayesian inference: priors, likelihoods and posteriors

Jones is a rational scientist. She has recently inherited her grandma's lucky coin. Grandma used this coin many times during Jones' childhood to determine whether Jones was allowed a sweet or not. Jones suspects that grandma's coin might be a trick coin, but she is not sure. She is determined to find out. How? Well, naturally, by rationally updating her *prior beliefs* about the coin's bias to obtain a new *posterior belief* based on empirical observation (outcomes of coin flips). Central to this updating is Jones' *likelihood function*, which encodes how likely each relevant coin bias may have generated the observed data. — Sounds fancifully abstract? It's actually fairly intuitive. Consider this example.

Prior beliefs. Jones knows that there are two factories who produce coins. One produces fair coins, the other produces coins which give heads three times more often than tails. She believes that it's equally likely that the coin is from either factory. Numerically, Jones' *prior beliefs* can be written as, where $\theta \in [0; 1]$ is the coin's bias: $P(\theta = 1/2) = 1/2$, and $P(\theta = 3/4) = 1/2$.

Likelihood. The bias θ is, by definition, the probability of the coin landing heads on the next trial. Let's assume that Jones tosses the coin only once (hm, maybe not so rational a scientist after all? or just too busy?). Let D be the set of potential outcomes of this experiment, namely $D = \{\text{heads}, \text{tails}\}$. The *likelihood function* determines the likelihood of observing each observation $d \in D$ for each θ , which in our case is just rather trivial: $P(D = \text{heads} | \theta) = \theta$ and $P(D = \text{tails} | \theta) = 1 - \theta$.

Posterior beliefs. Jones observes that the coin landed heads. What should she believe now? By *Bayes rule* her posterior beliefs are defined as:

$$P(\theta | D = \text{heads}) = \frac{P(\theta)P(D = \text{heads} | \theta)}{\sum_{\theta'} P(\theta')P(D = \text{heads} | \theta')}$$

Jones' posterior belief that the coin is twice as likely to land heads is therefore:

$$\begin{aligned} P(\theta = 3/4 | D = \text{heads}) &= \\ \frac{P(\theta = 3/4) P(D = \text{heads} | \theta = 3/4)}{P(\theta = 3/4) P(D = \text{heads} | \theta = 3/4) + P(\theta = 1/2) P(D = \text{heads} | \theta = 1/2)} &= \\ \frac{1/2 \cdot 3/4}{1/2 \cdot 3/4 + 1/2 \cdot 1/2} = \frac{3/8}{7/8} = \frac{3}{5} \end{aligned}$$

After making her observation, rational Jones believes that the bias towards heads has a probability of about 0.6, unlike her prior belief of 0.5.

Info Box 1: Priors, likelihood and posteriors in Bayesian inference.

Information displayed in the summary of a `brm` model fit

The first column *Estimate* gives the mean of the obtained samples, thereby approximating the mean of the posterior distribution (beliefs we should hold) about each parameter. For example, the parameter `Intercept` is estimated to have a mean of about 261, which (here) coincides with the mean of the data points in cell 1, as shown in Figure 2. The other columns give further useful information. *Est.Error* is the estimation error, an indication of the certainty we should have about the whole inference procedure. The columns `l-95% CI` and `u-95% CI` give the lower and upper bound of the **95% credible interval** for each parameter, estimated from the posterior samples. The column `Eff. Sample`, for efficient samples, gives a rough measure of how many of all the samples we took (4000 in our case) are contributing non-redundant information to our estimation. [tr: what does non-redundant mean concretely. Remember I am a noob reading this] [mf: I'll leave this comment unaddressed here and pick it up later in an Info Box on sampling etc.] The higher this number, the better. Finally, `Rhat` is a measure of whether the samples obtained are likely representative of the true distribution. Concretely, it indicates whether the four chains we ran in parallel all ended up with the same results, so to speak. If this column contains values bigger than 1.1 this is an indication that your model fit has not converged. (If your model output indicates non-convergence, you may want to increase the number of samples, but you should also consider the possibility that you are trying to fit a model which cannot be “trained” based on the (perhaps too little) data and the particular method of posterior sampling. For common regression analyses, this will usually entail considering a simpler model (e.g., with fewer explanatory factors, less (correlated) random effects, etc.))

Info Box 2: Information in summaries of `brm` model fits.