How to test hypothesis using Bayes Factor in behavioral sciences

Timo B. Roettger[1] and Michael Franke[2]
[1]Department of Linguistics & Scandinavian Studies, University of Oslo
[2]Department of Linguistics, University of Tübingen

Abstract

Recent times have seen a surge of Bayesian inference across the behavioral sciences. However, the process of testing hypothesis is often conceptually challenging or computationally costly. This tutorial provides an accessible, non-technical introduction that covers the most common scenarios in experimental sciences: Testing the evidence for an alternative hypothesis using Bayes Factor through the Savage Dickey approximation. This method is conceptually easy to understand and computatioanlly cheap.

Keywords: statistics, Bayes, Bayes Factor, Savage Dickey, hypothesis testing, ROPE

1 Introduction

To date, the most common quantitative approach across the experimental sciences is to run an experiment with one or more predictors and statistically test whether the predictors affect the measured variables. Traditionally, these statistical tests have been done within the null hypothesis significance testing framework. Over the last decade or so, however, we have seen more and more statistical approaches within an alternative inferential framework: Bayesian inference. Testing hypothesis within the Bayesian framework is often considered either conceptually challenging, computationally too costly, or both. This tutorial provides an accessible, non-technical introduction to Bayesian hypothesis testing that is easy to understand and computationally cheap.

2 Motivation and intended audience

This tutorial provides a very basic introduction to the topic using R (R Core Team, 2025). 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 Bayesian inference, this tutorial is for you. We will remain mostly conceptual to provide you with a conceptual tool to approach hypothesis testing within Bayesian inference. The form of hypothesis testing that we would like to introduce to you is, however, different from the traditional null hypothesis significance testing in that it requires more thinking about the quantitative nature of your data. This is not a bug but, at least for us, a feature that will allow you to understand both your data and what you can learn from them better. This work stands on the shoulders of giants with many fantastic resources out there. Importantly this tutorial and the accompanying scripts will make use of a few wonderful R package:

Timo B. Roettger https://orcid.org/0000-0003-1400-2739
The authors have no conflict of interest to declare.
Correspondence concerning this article should be addressed to Timo B. Roettger, Email: timo.roettger@iln.uio.no

- The extraordinary `brms` package to run our Bayesian models, written by Paul Buerkner (2016).
- The `bayestestR` package to calculate the Bayes Factors, written by Makowski et al. (2019)
- …

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 tutorial by Bodo Winter (2013) on mixed effects regression in a non-Bayesian —a.k.a. frequentist— paradigm. To then make the transition to Bayesian versions of these regression models, we shamelessly suggest our own tutorial on "Bayesian Regression for Factorial Designs" as a natural follow-up using the same data and Winter (Franke & Roettger 2019). In a sense, the present tutorial on hypothesis testing could be considered the long-awaited sequel of the series started by Winter. For continuity, we will continue to use the original data set.

To actively 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, which you can import with the following code:

```r
# package for convenience functions (e.g. plotting)
library(tidyverse)
library(ggdist)

# package for Bayesian regression modeling
library(brms)

# package for posterior wrangling and plotting
library(tidybayes)

# package for BF calculation and plotting
library(bayestestR)
library(see)
```

3 Data, research questions & hypotheses

This tutorial looks at a data set relevant for investigating whether voice pitch differs across social contexts in Korean. Korean is a language in which the social distance between speakers plays a central role. The way Korean speakers speak depends for example on whether they are in a formal context (e.g. during a consultation with a professor) or an informal context (e.g. chatting with a friend about the holidays) (e.g. Winter et al. 2012)

To load the data into your R environment, run the following code

```r
# TO DO: STORE ONLINE
# TO DO: SIMPLIFY STORED DATA
polite = read_csv("../data/polite.csv") |>
  # remove men
  filter(gender == "F") |>
  # calculate semitones (reference 50 Hz)
  mutate(pitch_ST = 12 * log2(pitch / 50))
```

```
polite
```

```
# A tibble: 126 x 5
   subject gender context  pitch pitch_ST
   <chr>   <chr>  <chr>    <dbl>    <dbl>
 1 F1      F      formal    215.     25.2
 2 F1      F      informal  211.     24.9
 3 F1      F      formal    285.     30.1
 4 F1      F      informal  266.     28.9
 5 F1      F      formal    211.     24.9
 6 F1      F      informal  286.     30.2
 7 F1      F      formal    252.     28.0
 8 F1      F      informal  282.     29.9
 9 F1      F      formal    230.     26.4
10 F1      F      informal  250.     27.9
# i 116 more rows
```

This data set contains anonymous identifiers for individual speakers stored in the variable `subject.` In this tutorial we will only be looking at female speakers btw. Subjects produced different sentences, and the experiment manipulated whether the sentences were produced in a `formal` or an `informal` social context, indicated by the variable `context.` Crucially, each row contains a measurement of pitch in Hz stored in the variable `pitch`. Here we have calculated a derived scale (semitones) of pitch which more closely resembles the way people perceive pitch (`pitch_ST`)

For most analyses of behavioral experiments, researchers are interested in whether an outcome variable is meaningfully affected by at least one manipulated variable and if so how the outcome variable is affected by it. In this case, Winter et al. (2012) wanted to test whether pitch is meaningfully affected by the social context of the utterance.

As a first step, we can explore this question visually:

Figure 1

Empirical distribution of pitch values across contexts

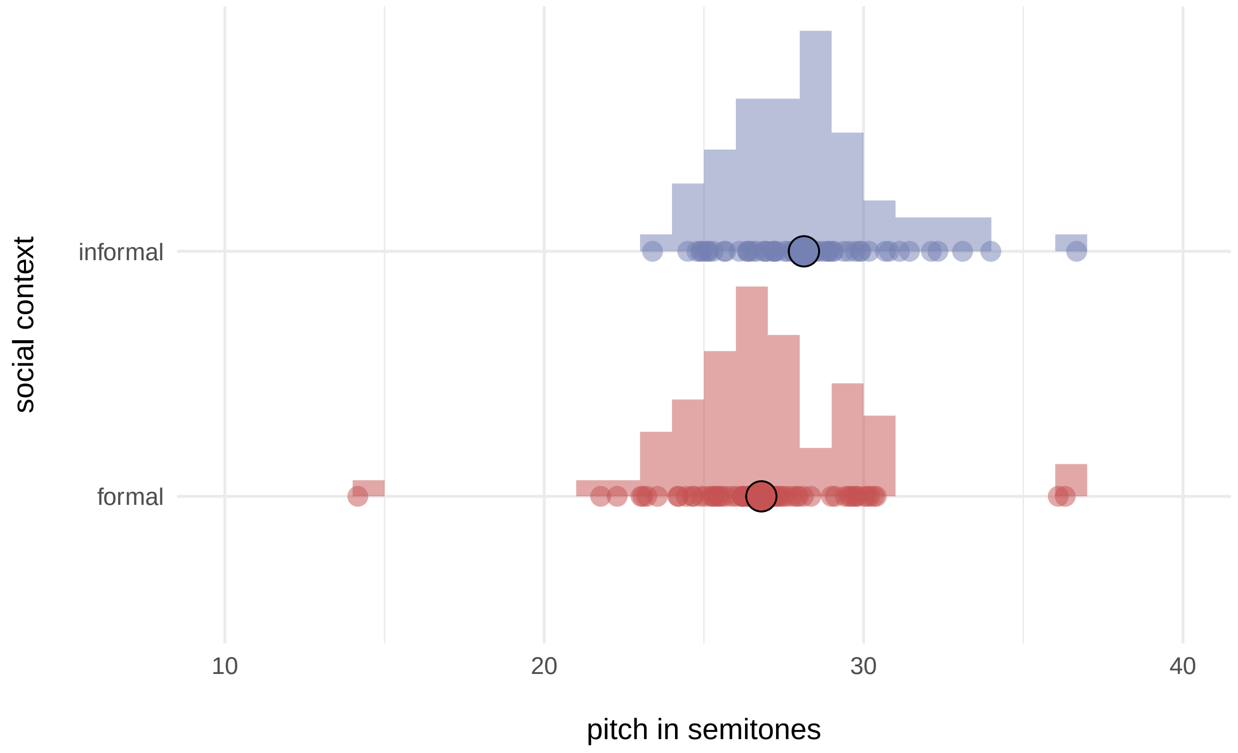


Figure 1 displays the pitch values for all utterances in the dataset across contexts (semi-transparent points). The solid points indicate the average pitch values across all sentences and speakers. Looking at the plot, we can see that voice pitch from utterances in formal contexts are on average slightly lower than those in informal contexts. The red distribution is slightly shifted to the left of the blue distribution by around 1.3 semitones. In other words, speakers tend to slightly lower their voice pitch when speaking in a formal context. But there is also a lot of overlap between the two contexts. Now as Bayesians, we would like to translate the data into an expression of evidence: does the data provide evidence for our research hypotheses?

Let us build a Bayesian linear model to approach an answer to this question. Our first step is to specify the model formula and check which priors need to be specified:

```r
# define linear model formula
# predict pitch by context and allow for that relationship
# to vary between subjects
formula <- bf(pitch_ST ~ context + (1 + context | subject))

# get priors for this model
get_prior(formula, polite)
```

The default priors that brms picks for the Intercept and the variance parameters are mostly reasonable as they are derived from the data, weakly informative and symmetrical. However the prior for our critical parameter `contextinformal` should also be weakly informative (REFERENCE), i.e. the prior assumption about the difference between informal and formal contexts should be that we don't know, but our best guess is that it is close to zero and equally likely to be more or less than zero. So we specify a normal distribution centered on zero for this parameter.

Note: Only for demonstration purposes, we will use default priors for the other parameters. You always should critically reflect on all of your priors.

```
# NOTE: TO MAKE IT EASIER TO FOLLOW WE COULD CONTRAST CODE THE PREDICTOR

# pick a weakly informative prior for the critical parameter
priors <- prior(normal(0, 2),
                class = b,
                coef = "contextinformal")
```

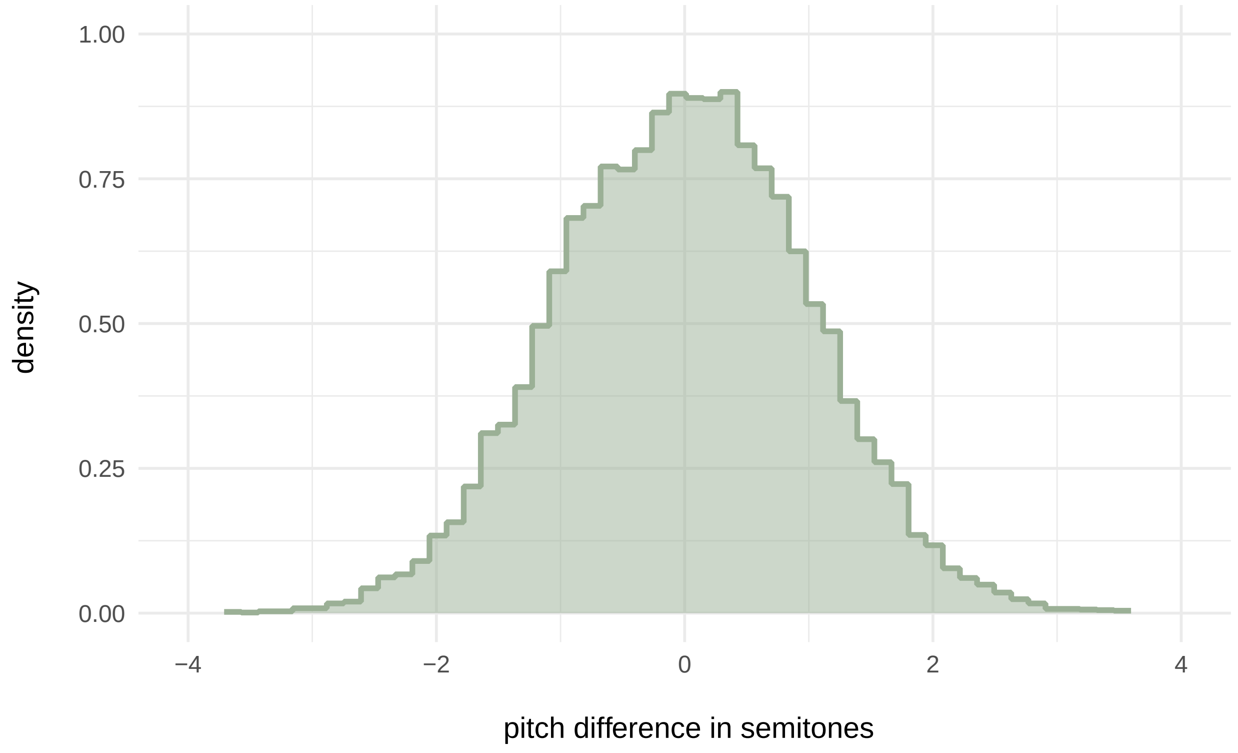
Now we do a so-called prior predictive check, in other words we want to know what the posterior distribution looks like before having seen the data, based on the priors only. This is a useful exercise to make sure that the priors results in reasonable quantitative assumptions. We usually do it for all parameters, but here we will focus only on the critical parameter `contextinformal`, i.e. the difference between formal and informal contexts. Let us also have a look at the predictions for the prior-only model.

```
# NOTE: CAN WE STORE THE SAMPLING PARAMETERS (seed, iter, chains, cores, backend, data)?
#       TO MAKE THE CODE CHUNKS SMALLER?

# run the model
fit_prior <- brm(formula,
           prior = priors,
           family = gaussian(),
           # sample prior only
           sample_prior = "only",
           # common sampling specifications
           file  = "../models/fit_prior",
           seed = 1234,
           iter = 8000,
           chains = 4,
           cores = 4,
           backend = "cmdstanr",
           data = polite)
```

```
# extract prior samples
prior_samples <-
  fit_prior |>
  spread_draws(b_contextinformal)

# plot
ggplot(prior_samples,
       aes(x = b_contextinformal)) +
  stat_histinterval(slab_color = project_colors[11],
                    slab_fill = alpha(project_colors[11], 0.5),
                    fill = NA,
                    color = NA,
                    outline_bars = FALSE) +
  labs(x = "\n pitch difference in semitones",
       y = "density \n") +
  scale_x_continuous(limits =c(-4,4)) +
  theme_minimal()
```

Figure 2

Prior probability of the effect of context on pitch, i.e. before seeing the data



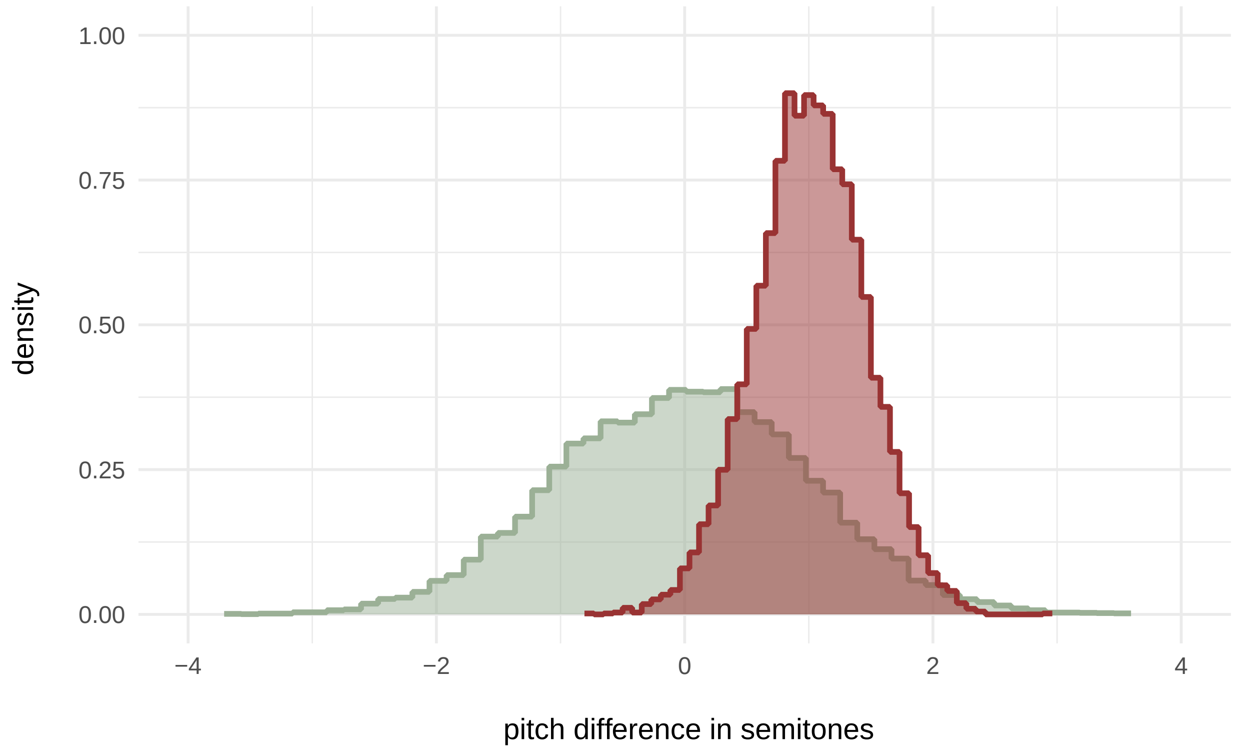
Looking at the distribution, the priors for the effect of context on pitch seems sensible. The most plausible value is zero. Values that are smaller or larger than zero become less plausible the further they are away from zero and values being smaller or larger than zero are equally likely. Good. Before we have seen the data, our model is somewhat pessimistic about the effect of context on on pitch. Now we can run the full model that integrates the likelihood (our data) with the priors and visualize the posteriors for the critical parameter.

```
# run the model
fit <- brm(formula,
           prior = priors,
           family = gaussian(),
           # common sampling specifications
           file  = "../models/fit",
           seed = 1234,
           iter = 4000,
           chains = 4,
           cores = 4,
           backend = "cmdstanr",
           data = polite)
```

```
posterior_plot <-
  fit |>
  spread_draws(b_contextinformal) |>
  ggplot(aes(x = b_contextinformal)) +
    stat_histinterval(data = prior_samples,
                     slab_color = project_colors[11],
                     slab_fill = alpha(project_colors[11], 0.5),
                     fill = NA,
                     color = NA,
                     outline_bars = FALSE) +
    stat_histinterval(slab_color = project_colors[14],
                      slab_fill = alpha(project_colors[14], 0.5),
                      color = NA,
                      outline_bars = FALSE) +
  # stat_slab(data = prior_samples,
  #           lwd = 1,
  #           colour = project_colors[11],
  #           fill = NA) +
  scale_thickness_shared() +
  scale_x_continuous(limits =c(-4,4)) +
  labs(x = "\n pitch difference in semitones",
       y = "density \n") +
  theme_minimal() +
  theme(legend.position = "top")

posterior_plot
```

Figure 3

Posterior probability of the effect of context on pitch, i.e. after seeing the data



The posterior samples suggests that the majority of plausible values after seeing the data are positive, or in other words, informal contexts elicit larger pitch values. Negative values are not very plausible values, but also not completely implausible. We can visually access the amount of of posteriors that fall on the left side of zero by relating the dark blue part of the density curve to the yellow part. Compared to our prior probability (green distribution) for which roughly 50% of posteriors were negative, this decrease in plausibility of negative values is quite noteworthy already.

What we have done here should be quite familiar. We compare our model predictions to a reference point. It is a single point value: zero.But do we really care that much for such point hypotheses? Is zero really that special? We might think so because years of using null hypothesis significance testing has conditioned us to think that way. But this tutorial would like to break this cycle and move forward. Bear with us and let's approach hypothesis testing a bit differently today.

3.1 Grounding hypotheses in regions of practical equivalences

NOTE NEED TO FIND A BETTER JND SOURCE

Above we claimed that we wanted to test "whether pitch is **meaningfully affected** by the social context of the utterance". We snuck the word meaningfully in there for a reason. But what does "meaningful" mean? This is really a good questions and (un)fortunately requires quite a bit of thinking. This tutorial deals with speech data. Speech is, in spoken languages at least, THE vehicle to transmit linguistic information in order to communicate with each other. Speech is also very complex and very noisy: Not everything that can be measured in the acoustic signal matters for the listener. For example, if something cannot be perceived reliably, it is at least conceivable that it might play little to no role in communication. While speech sciences has a rich research tradition to estimate what can and what cannot be reliably heard, exact estimates depends on a lot of moving parts. But there is evidence that pitch in static conditions with resynthesised speech might be around 5% (Isačenko and Schädlich 1970). That means, pitch changes below 0.05 semitones can not be used to discriminate two sounds reliably. Such thresholds are referred to as Just Noticeable Differences (JNDs) and can be used to define what constitutes meaningful differences when we look at speech data. So we could interpret the original hypothesis the following way: If a pitch difference is below the JND, it is not meaningful. So instead of testing against a point-zero hypothesis, we can test against a range of parameter values that are equivalent to the null value for practical purposes. In our case, let us be extra conservative and double the reported JND to 0.1, so values between `-0.1` and `0.1` are simply meaningless. Such ranges are called regions of practical equivalence (ROPEs, Kruschke, 2010, 2018).

```
rope <- c(-0.1,0.1)
```

With a ROPE being defined, we can now test our hypothesis "whether pitch is **meaningfully affected** by the social context of the utterance" using Bayes Factor:

4 Testing hypothesis using Bayes Factor

4.1 What is Bayes Factor

Bayes Factors (henceforth: BFs) allow us to quantify relative evidence of one model compared to another.

4.2 Approximating BF with Savage Dickey

4.3 BF for a specified Region of Practical Equivalence (ROPE)

Instead of doing it by hand, we can calculate the Savage Dickey ratio with the `bayesfactor_parameters()` function from the `bayesfactorR` package. What happens behind the scenes is that the function will sample posteriors from your specified model based on priors only (so before seeing any data) and calculates the posterior probability of the specified `null` hypothesis (here the range specified by our ROPE).

```
#|warning: FALSE
#|message: FALSE

BF_1 <- bayesfactor_parameters(posterior = fit,
                               null = rope,
                               parameter = "b_contextinformal")
```

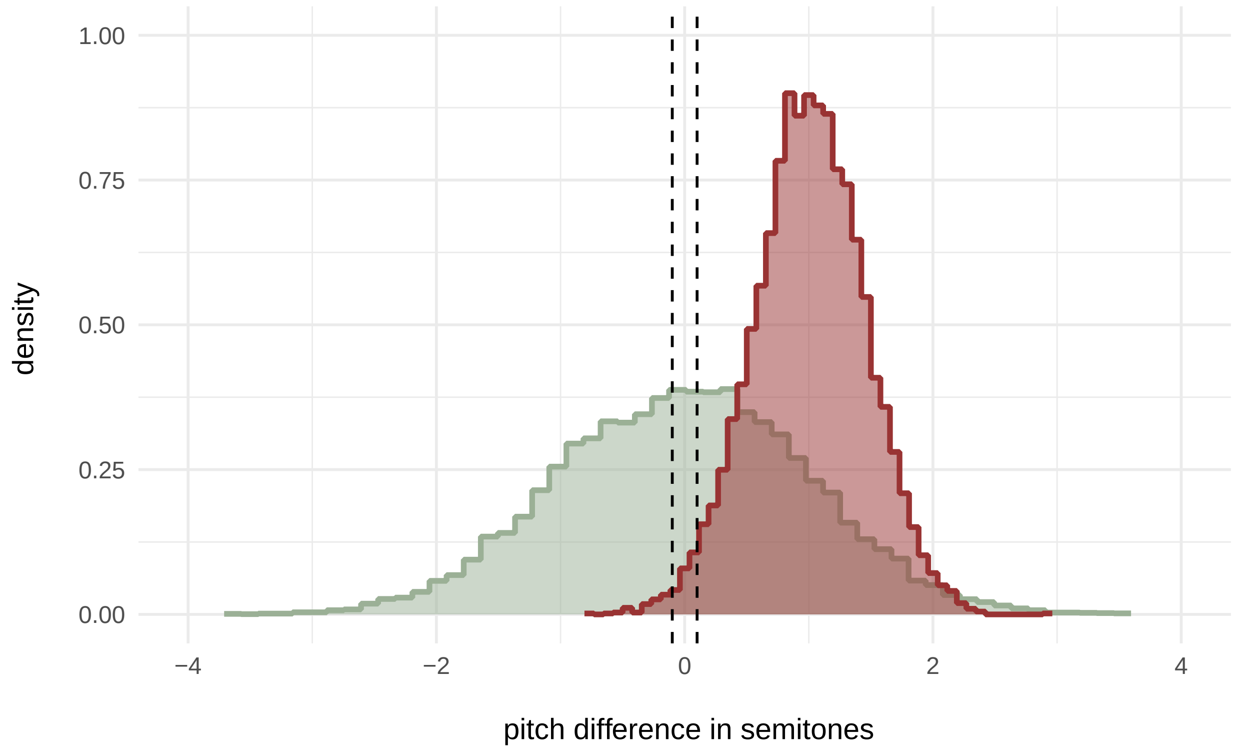
```
Sampling priors, please wait...
```

Before interpreting the number we get, let us visually explore what our BF corresponds to.

```
posterior_plot +
  geom_vline(xintercept = c(rope[1], rope[2]),
             lty = "dashed")
```

Figure 4

Prior and posterior probability of the effect of context on pitch relative to the ROPE (-0.1, 0.1)



What the BF does is relating two numbers: (a) The prior probability of parameter values outside the rope, i.e. the proportion of the green distribution that falls outside the dashed lines, and (b) the posterior probability of parameter values outside the rope, i.e. the proportion of the red distribution that falls outside the dashed lines. Eye-balling the plot, we can maybe already see that more of the red distribution is outside the ROPE than of the green distribution.

```
BF_1
```

```
Bayes Factor (Null-Interval)

Parameter       |    BF
-----------------------
contextinformal | 5.63

* Evidence Against The Null: [-0.100, 0.100]
```

To be exact, 5.6 times for of the red distribution is outside the ROPE than of the green distribution.

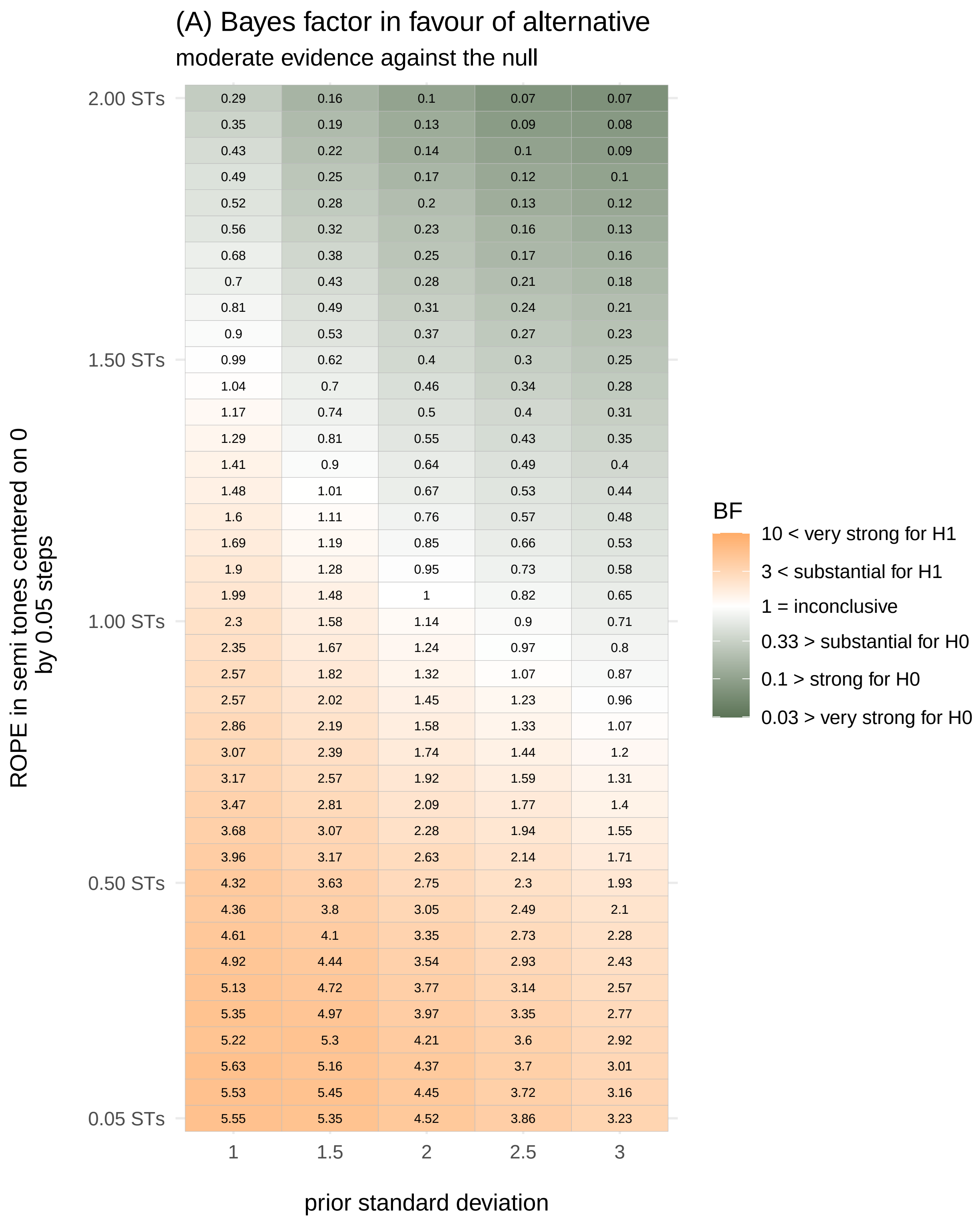
That means the model that has seen the data provide 5.6 times more evidence for pitch being outside of the ROPE, or in other words, it is 5.6 times more likely (after having seen the data), that context affects pitch meaningfully. According to Jeffrey's (1961) criteria for interpreting BFs, this value corresponds to moderate evidence for the alternative hypothesis.

Now as you probably have guessed already, all these probabilities are very much dependent on the priors of the model, so it is important to evaluate the robustness of our BF-based interpretation across a range of sensible priors. And as long as we are not a 100% sure about our ROPEs, we might as well explore the robustness of the BF across different ROPEs. We won't bore you with the code for that process, but you can follow it along in our scripts. Let us assume the following five ROPE intervals centered on zero: 0.05, 0.1, 0.5, 1, 2. We assume the following five prior values for the width of the standard deviation of the critical parameter (centered on zero): 1 1.5, 2, 2.5, 3. These are all sensible prior widths assuming that medium to strong effects in either direction are plausible.

```
Rows: 200 Columns: 3
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): interval_range
dbl (2): prior_sd, BF

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Figure 5

Bayes Factors for a range of priors and a range of ROPEs



The matrix of BFs is visualized in Figure X. By comparing the BF values along the y-axis, we can see that the calculated BFs are heavily dependent on the chosen ROPE. We here chose theoretically quite different ROPEs. For ROPEs above one semitone, BF values become less stable and range from anecdotal evidence for the alternative to strong evidence for the null hypothesis, dependent on the prior.

By comparing the BF values along the x-axis, we can see that the calculated BFs are relatively robust for different standard deviations of the critical prior. However, we can also see that the BF values decrease with the width of the priors. This is not surprising and a known phenomenon, often discussed under the Jeffreys-Lindley paradox (Lindley 1957). The more diffuse the priors are (i.e. wider priors), the larger is the probability that a specific parameter values is not compatible with the data.

4.4 BF for point hypothesis

don’t lol

4.5 Write up

Do’s Do think think about sensible priors think about sensible ropes think about the smallest effect sizes of interest instead of testing point-0

Don’ts Don’t fall into the trap of discrete thresholds. Don’t hack ropes