Tutorial for seqBF()

Emily Line

Thank you for helping us test the function `seqBF()` as part of our package! Please email Emily Line at neuline2@illinois.edu with any feedback you have, whether it be how easy it is to use the function, if you are running into problems, or if you have suggestions of things to add!

Overview of Function

The purpose of this function is to make optional stopping easier to do with t-tests, ANOVAs, and correlations (more analysis types might be added in the future but this is what we're starting with). Optional stopping is a procedure that allows you as a researcher to run analyses as you collect data without affecting the Type I and II error rates. To do this, we use Bayes factors as a "decision" metric instead of p-values. The function helps users identify if they have reached a point in their data collection where they can stop collecting data.

Parameters for function:

Parameters for a function are things the user will have to input or will have the freedom to change. They are set inside the parentheses of the function (e.g., `seq.BF(testtype = "t-test")`). Some of these will be specific to the type of analysis you want to run and are listed under each test type in this tutorial. The parameters that apply to all of the analyses are listed below.

`testtype` - this is the type of test you want to run. It can be a t-test, ANOVA, or correlation.

`threshold` - this is the "cutoff" for the BF when determining if you have reached a point where you can stop collecting data. **Important**: This threshold should be decided *before* you start collecting data. If you change the threshold as you are collecting data and running seqBF() this could lead to inaccurate conclusions. For the purposes of testing you probably have data already, so it's fine to play around with the threshold if you aren't using the results.

`rscale` - this is related to the prior for the Bayes Factor. We have set a default value at $rscale = .5$. This prior distribution is centered on a Cohen's d effect size of 0 and the rscale influences the "spread" of the distribution, with larger values creating a wider spread. The default $rscale = .4$ creates a distribution where most effect sizes are between a Cohen's d of -1 and 1. I am working on writing a more extensive explanation of this, so let me know if you have questions in the meantime. If you don't know what to do with this information, then leave rscale as the default value.

The rest of the parameters are test-specific so I will explain them in more detail later on.

Workflow

This is how we anticipate the workflow to go: A researcher will collect some data to start, and then they will run `seqBF()`, using a t-test model, for example. This will run a t-test and output a Bayes Factor (BF). The output will also tell the researcher if the BF has exceeded the decision threshold. If it has, the researcher can conclude there is support for either an alternative hypothesis or the null hypothesis. If the BF has not exceeded the threshold, then the researcher will collect some more data and run `seqBF()` again. They will continue to collect data and run `seqBF()` periodically until the BF has exceeded the threshold or until they have reached a predetermined maximum sample size (which could related to funding, for example).

For the purposes of testing, you probably have data that has been fully collected already. What would be helpful is if you could take smaller samples from your data to simulated this optional stopping process. I will include some code that can help you do that in the **Simulate Optional Stopping** section.

Setting Up

To get the function up and running, you will need to run a couple things. Once this is in package form it will be a lot simpler, but for now this is the way it is. First, open the R script titled 'supporting_functions_seq.bf'. These are functions that are used within the `seq.BF()` function. You will just need to run this entire script. An easy way to do this is Ctrl+A (selects the entire script) and then press Ctrl+Enter which runs everything selected.

Next, open the R script labeled 'seq.BF' and run everything in that script. This includes the library for the `BayesFactor` package and the `seq.BF()` function itself. You can run things one at a time by selecting the line code is on and pressing Ctrl+Enter to run that line or section of code. At this point, you have everything loaded into your environment to use the `seq.BF()` function. You can test the function in the same script if you want, or you can create a new script to test the function in.

Loading Data

You will need to load your dataset into R in order to test this function. If you're not familiar with loading data into R, this short tutorial is easy to follow.

Testing the Function

Simulate Optional Stopping

If you are not actively collecting data and in a situation where you could use `seq.BF()` as you collect data, it would be helpful to have data that is simulated to look like optional stopping. Essentially, this will involve taking a random sample of your data, testing it, adding more of the dataset to that sample, testing it again, and so forth until your whole dataset is included in the sample.

To simulate this process, we will create a series of "sub-datasets" that represent each step along the sampling procedure.

I am using the `puzzles` dataset from the `BayesFactor` package as an example. This has puzzle completion time data from Hayes (1994). To see a full description of the dataset, type `help(puzzles)` into the console (make sure the `BayesFactor` package is loaded). We start with the full dataframe:

```
data(puzzles)
full_data <- puzzles

head(full_data)
```

```
##   RT ID shape         color
## 1 49  1 round monochromatic
## 2 47  2 round monochromatic
## 3 46   3 round monochromatic
## 4 47   4 round monochromatic
## 5 48   5 round monochromatic
## 6 47   6 round monochromatic
```

Then, decide how many sub-datasets you want. For testing, it would be helpful to test this with different sizes of "subsets". I will choose 3 subsets for this example, which will result in 4 total samples when you include the full dataset. The first subset you take will be the second-to-last, so I will label it accordingly as Sample C. The full dataset has 48 observations, so I will set the second-to-last dataset to have 10 fewer observations. Thus, I set the size equal to 38. The third-to-last sample will be another 10 observations fewer, but this time we are sampling from Sample C instead of the full dataset. The first (or fourth-to-last) sample will only have 18 observations in it, and it will be sampled from Sample B. This will represent our first round of data collection. We can double check the size of each dataframe by using `nrow()`. The smallest sample should be your "first" sample, and the largest sample (your full dataset) should be your "last" sample.

```r
sampleC <- full_data[sample(nrow(full_data), size=38), ]
sampleB <- sampleC[sample(nrow(sampleC), size=28), ]
sampleA <- sampleB[sample(nrow(sampleB), size=18), ]

nrow(sampleA)
```

```
## [1] 18
```

```r
nrow(sampleB)
```

```
## [1] 28
```

```r
nrow(sampleC)
```

```
## [1] 38
```

```r
nrow(full_data)
```

```
## [1] 48
```

Now that we have our subsets, we are ready to test!

T-tests

For t-tests, you will need to set `testtype = "test"`. `seq.BF` takes some different parameters for t-tests.

`paired` - set this as `FALSE` or `TRUE`, depending on if you are running a paired samples t-test

Independent Sample t-tests

For independent sample t-tests, you will need to enter a formula indicating the dependent and independent variables. `formula` - this will take the form of "y ~ x" where y is the dependent variable and x is the independent variable (with two levels). `data` - the dataframe your variables are in

Paired Samples t-test

`x` - data for the first observation in the pair `y` - data for the second observation in the pair **Note**: Even those these variables are entered in x/y form, it is not setting either variable to a dependent/independent variable

You can also change the parameters described earlier.

We start with our first "sample" in our stopping procedure, Sample A. This is an independent sample t-test, so we need to enter the formula. In this case, `RT` is our dependent variable and `shape` is our independent variable, so we enter the formula as `RT ~ shape`. We also specify which dataset our sample originated from. The formula will stay the same as we change the datasets across our different sample increments. So while we specify `sampleA` here for the dataset, we will change that as we "collect more data."

```
seq.BF(testtype = "t-test",
       formula = RT ~ shape,
       data = sampleA,
       paired = FALSE)
```

```
## Bayes factor has not exceeded the threshold of 7.
##
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.

## Bayes factor analysis
## --------------
## [1] Alt., r=0.5 : 2.794991 ±0.01%
##
## Against denominator:
##   Null, mu1-mu2 = 0
## ---
## Bayes factor type: BFindepSample, JZS
```

After running `seq.BF()` with Sample A, we see the BF is not greater than 7. Let's see what it looks like when we add 10 more observations to our sample:

```
seq.BF(testtype = "t-test",
       formula = RT ~ shape,
       data = full_data,
       paired = FALSE)
```

```
## Bayes factor has not exceeded the threshold of 7.
##
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.
## Bayes factor analysis
## --------------
## [1] Alt., r=0.5 : 0.7287808 ±0%
##
## Against denominator:
##   Null, mu1-mu2 = 0
## ---
## Bayes factor type: BFindepSample, JZS
```

In this case, the BF went down, indicating the data we added contributed towards support for the null hypothesis more than the alternative hypothesis. We haven't exceeded the threshold of greater than 7 or less than 1/7 yet, so let's add 10 more to the sample.

```
seq.BF(testtype = "t-test",
       formula = RT ~ shape,
       data = sampleC,
       paired = FALSE)
```

```
## Bayes factor has not exceeded the threshold of 7.
##
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.

## Bayes factor analysis
## --------------
## [1] Alt., r=0.5 : 0.4703883 ±0%
##
## Against denominator:
##   Null, mu1-mu2 = 0
## ---
## Bayes factor type: BFindepSample, JZS
```

We see that the BF changed a little, but still has not exceeded the threshold. Let's see what is looks like when we add 10 more observations and reach our final sample size.

```
seq.BF(testtype = "t-test",
       formula = RT ~ shape,
       data = full_data,
       paired = FALSE)
```

```
## Bayes factor has not exceeded the threshold of 7.
##
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.
## Bayes factor analysis
## --------------
## [1] Alt., r=0.5 : 0.7287808 ±0%
##
## Against denominator:
##   Null, mu1-mu2 = 0
## ---
## Bayes factor type: BFindepSample, JZS
```

So we've reached the full sample size of this dataset and still haven't exceeded the threshold. If we were doing this in practice, we would keep collecting data until we reached the threshold for either the alternative or null hypothesis.

Correlations

For correlations, you will need to set `testtype = "correlation"`.

`x` - one of the continuous variables in your correlation (does not imply that it's an independent variable) `y` - the other continuous variable in your correlation (does not imply that it's a dependent variable)

When running correlations with seq.BF, you will only enter an x and y value, with no dataset specified. This means you will have to specify where x and y are coming from. We can do that before running the `seq.BF()` function to make things a little easier. Since the puzzles dataset I used for the simulation example doesn't have data that would work for correlations, I am going to use the Iris dataset to show how this can work. To extract a column or variable from a dataset, simply put the name of the dataset first (e.g., `sampleA`), followed by a $ sign and the name of the variable (`sampleA$RT`). Make sure to do this for each sample you have (e.g., `sampleB$RT` when you move on to using Sample B).

I won't go through the sequential component of the analysis but it works the same way as t-tests and ANOVAs. Note that the input is still in x and y form. This is not setting an "independent" or "dependent" variable but just serving as labels in this case. The results should be the same if you switch the x and y values. For this example, I use the `iris` dataset from base R.

```r
y = iris$Sepal.Length
x = iris$Sepal.Width
seq.BF(testtype = "correlation",
       y = y,
       x = x,
       rscale = .5)
```

```
## Bayes factor has not exceeded the threshold of 7.
##
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.
```

Bayes factor analysis

[1] Alt., r=0.5 : 0.4151844 ±0%

Against denominator: Null, rho = 0 — Bayes factor type: BFcorrelation, Jeffreys-beta*

ANOVAs

For ANOVAs, you will need to set `testtype = "anova"` (make sure this is lower-case).

`formula` - specifies your model. Should take the form of `Y ~ X` (see explanation below for more formula information) `data` - the dataframe that the formula should pull data from. **Note**: Make sure the variable names in your formula match the names in your dataframe `whichRandom` - if you have random factors in your ANOVA model, you need to indicate that they are random factors with this parameter. For example, if you have a random factor called `schools` you would put `whichRandom = "schools"` as a parameter in `seq.BF()`. `rscaleRandom` - this is the `BayesFactor` package's parameter for the prior scale for standardized random effects. The default is set at "nuisance".

The ANOVA input will take the form of a `formula`. This allows you to include two-way ANOVAs or ANOVAs that have a random effects component. First, lets create a formula variable to use as input. The lefthand side of the formula should only have the dependent variable. In this case, that is RT. Then the formula will have a ~ sign, which stands for "distributed as". The righthand side of the formula will include all of your independent variables. If you want a two-way ANOVA that considers the shape and color of puzzle pieces as independent variables, you could have `RT ~ shape + color` if you do *not* want to consider an interaction, otherwise you could have the formula `RT ~ shape*color` if you *do* want to consider an interaction. This is the formula I will use below. Once you have a formula saved to a variable, you can continue to use that formula even as you change the data argument in `seq.BF()`. It is not tied to a given dataset, as long as the data has the variables you use in the formula.

```
anova_form <- RT ~ shape*color
```

With our formula variable specified, we can begin the sequential testing procedure: Starting with Sample A, I run seq.BF() with each increased sample size.

```
seq.BF(testtype = "anova",
       formula = anova_form,
       data = sampleA,
       whichRandom = "ID",
       rscale = .5)
```

```
## ANOVA models usually have more than one Bayes Factor. Each Bayes factor corresponds to an
##  effect in your model.
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.

## Bayes factor analysis
## --------------
## [1] shape                       : 2.980904  ±0.01%
## [2] color                       : 0.6863331 ±0%
## [3] shape + color               : 2.086764  ±1.62%
## [4] shape + color + shape:color : 1.215149  ±10.85%
##
## Against denominator:
##   Intercept only
## ---
## Bayes factor type: BFlinearModel, JZS
```

```
seq.BF(testtype = "anova",
       formula = anova_form,
       data = sampleB,
       whichRandom = "ID",
       rscale = .5)
```

```
## ANOVA models usually have more than one Bayes Factor. Each Bayes factor corresponds to an
##  effect in your model.
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.

## Bayes factor analysis
## --------------
## [1] shape                         : 0.8271964 ±0%
## [2] color                         : 0.4732304 ±0%
## [3] shape + color                 : 0.4617108 ±0.73%
## [4] shape + color + shape:color : 0.2197968 ±0.98%
##
## Against denominator:
##   Intercept only
## ---
## Bayes factor type: BFlinearModel, JZS
```

```
seq.BF(testtype = "anova",
       formula = anova_form,
       data = sampleC,
       whichRandom = "ID",
       rscale = .5)
```

```
## ANOVA models usually have more than one Bayes Factor. Each Bayes factor corresponds to an
##  effect in your model.
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.

## Bayes factor analysis
## --------------
## [1] shape                         : 0.3751499 ±0%
## [2] color                         : 0.6957189 ±0.01%
## [3] shape + color                 : 0.264305  ±1.45%
## [4] shape + color + shape:color : 0.1128279 ±3.88%
##
## Against denominator:
##   Intercept only
## ---
## Bayes factor type: BFlinearModel, JZS
```

```
seq.BF(testtype = "anova",
       formula = anova_form,
       data = full_data,
       whichRandom = "ID",
       rscale = .5)
```

```
## ANOVA models usually have more than one Bayes Factor. Each Bayes factor corresponds to an
##  effect in your model.
## A Bayes factor less than  7 or greater than  0.14  does not constitute evidence for the
##  alternative or null hypothesis, respectively.
##
##   We recommend you keep sampling if you want to make inferences about an
##   effect that has a Bayes factor that does not exceed the threshold for the
##   alternative or the null hypothesis.
```

```
## Bayes factor analysis
## --------------
## [1] shape                         : 0.6114754 ±0.01%
## [2] color                         : 0.6114754 ±0.01%
## [3] shape + color                 : 0.3864658 ±0.82%
## [4] shape + color + shape:color : 0.1392385 ±2.35%
##
## Against denominator:
##   Intercept only
## ---
## Bayes factor type: BFlinearModel, JZS
```

You can see that as the sample size increases (from Sample A to Sample B, for example), the BFs fluctuate. In this scenario, even when we reach the point of using the full dataset, there are no BFs that exceed the threshold of 7. One BF is close to exceed the threshold for the null hypothesis, which is $1/7 = 0.14$. If we were running this experiment in practice, we would continue to collect data until we reached a threshold or until we reached our prespecified maximum sample size.