Advanced Statistical Methods

Bayesian Statistics

Dr Lincoln Colling

Contents

Preface

The aim of this course is to give you an introduction to Bayesian statistics. It is by no means intended to be an exhaustive course, so at the end of it, there will still be a lot for you to learn. However, I do hope that at the end of this workshop you'll have a better understanding of Bayesian statistics, how it differs from Frequentist approaches, and how to incorporate some Bayesian methods

into your research.

In this course we'll primarily focus Bayesian hypothesis testing using Bayes factors. We'll also discuss how Bayesian hypothesis testing differs from Null-hypothesis significance testing. Bayesian hypothesis testing with Bayes factors is just one of many approaches that could fall under the heading on **Bayesian statistics**. Other approaches would include things like Bayesian paramter estimation, and Bayesian regression modelling. These other approaches are definitely worth learning, but there simply aren't within the scope of this short workshop.

How to use this book

You can read through this book just like you would read through any course notes. However, you're get the most out of it if you work along with the code. To work along with the code click on one of the two badges that appear at the start of each chapter.

The first badge will allow you to download the **RMarkdown** file that contains all the code for the examples. You'll be able to load this up in **RStudio**.

The second badge will allow you to open the chapter in **Google Colab**. The interface for **Google Colab** is similar to an **RMardown** document in that it will contain code and text. However, you won't need **R** installed and you'll be able to work with the file directly in your web browser. If you make any changes to the document then you'll be able to save these directly in you Google Drive account. You'll just need to login with a GMail account.

Outline

In the first part of this course will be a refresher on frequentist methods. In particular, we'll talk about the sampling distribution, where it comes from, and how we do inference with it. Following this, we'll cover some of the features of sampling distributions that have been criticised by Bayesian. From the sampling distribution, we'll move on to likelihoods. We'll see that likelihoods don't suffer from some of the problems of sampling distributions, and we'll see how to do inference with likelihoods.

Finally, we'll learn how to compute and interpret Bayes factors, and we'll learn about some of the considerations that go into specifying models for Bayes factors.

```
## Warning: package 'magrittr' was built under R version 4.1.2
```

Chapter 1

Null-hypothesis significance testing

Before diving into Bayesian hypothesis testing, it's worth spending a little time going over Frequentist null-hypothesis significance testing. The reason for this is that I want it to be clear that Bayesian methods and Frequentist methods aren't just two different ways of answering the same question. Rather, I want it to be clear that Bayesian methods and Frequentist methods are asking **fundamentally different questions**. For this reason, we'll start right back at the beginning with p values.

The American Statistical Association (ASA) defines a p value as:

> the probability under a specified statistical model that a statistical summary of the data (e.g., the sample mean difference between two compared groups) would be equal to or more extreme than its observed value (Wasserstein and Lazar 2016)

While this is a perfectly acceptable definition, it is maybe a little tricky to understand. The main reason for this is that the definition contains at least one *ill-defined concept* ("probability") and one tricky concept ("specified statistical model"). To understand what a p value really is, we're going to have to unpack both of these ideas. Along the way, we're going to learn about some other concepts that will also help us understand *Frequentist* inference. And a good grounding in Frequentist inference will also help us understand the distinction between Frequentist inference and Bayesian inference.

1.1 Probability

Most people think of *probability* as a mathematical concept. In a sense it is, but it is also a deeply *philosophical* concept. We deploy the word *probability* in many different *kinds* of situations, and it's not clear whether we mean the same thing in each of them. Some examples of where we use the word probability are when we ask questions like: What is the probability of getting heads on repeated tosses of a fair coin? What is the probability that it will rain tomorrow? What is the probability the accused committed the crime? The word *probability* seemingly refers to different things in each of these situations.

For example, we might suggest that the probability of the getting heads is 0.5, where this 0.5 refers to the *long-run relative frequency* of getting heads. That is, if we were to toss a coin many many times then on around 0.5 (i.e., half) of the tosses the coin would come up heads.

We might use a different notion when thinking about the case of somebody accused of a crime. We might say something like, "we are 90% sure" (probability of .9) that the criminal committed the crime. But what does "90% sure" mean. Does it make sense to think of it as the *relative frequency*? If not, then how else might we think of it? We might, for example, think of it as a *credence* or a *degree of belief* that the proposition is true. Or we might think of it as a *degree of support*. That is, we might say that the available evidence supports the hypothesis that the accused committed the crime with odds of 9-to-1.

This list isn't meant to be exhaustive. The aim is just to highlight that we might sometimes mean different things when we think about probability. It pays to keep this in mind as we move through the course.

1.2 Probability and *p* values

Now that we know that *probability* can mean different things in different situations, what notion of *probability* is at play in ASA's definition of the p value? The common view is to say that it refers to *relative frequencies*. But relative frequencies of **what** over repeats of **what**?

We could possibly re-phrase that definition to say something like this:

> the p value refers to the relative frequency of obtaining a statistical summary of the data as large or larger than the observed value over hypothetical repeats of an experiment described by a specified statistical model

1.2.1 Understanding the *p* through simulation

One method that I think is useful for understanding statistical concept is *simulation*. This is particularly true in the case of p-values, because I definition above refers to *hypothetical repeats*. Simulation means that we can just simulate those *repeats*. To understand how p values work, let's start with a little scenario:

> You've been given a device that can be used to find buried treasure. The device has a numbered dial on it, and there is a little arrow that can point at these numbers. The indicator never stays still, but swings around a bit. You don't know how the device works, except that it behaves *differently* around treasure compared with when there is no treasure present. How can you use this device to find treasure?

This seems like a hard problem. You know very little about the device. You don't know what it's meant to do when it finds treasure, and you don't know what it's meant to do when there isn't any treasure. So how do you go about using it to find treasure?

1.2.1.1 Finding treasure

The first step in using the device is to get a good description of what it does when there isn't any treasure around. To do this, you might just take your device somewhere without treasure. You can then just sit and watch the dial. After a long time watching it, you might notice that although the pointer swings around a lot, *on average* it points at zero. This one bit of information is enough to develop a treasure hunting strategy using this device.

The first step in the strategy is deciding how many readings to take on each hunt. Because the pointer swings around a lot, we'll need to take a couple of readings and then use these to work out an average (which we'll call $\bar{x}$). We're in a hurry so we'll take **10** readings on each hunt.

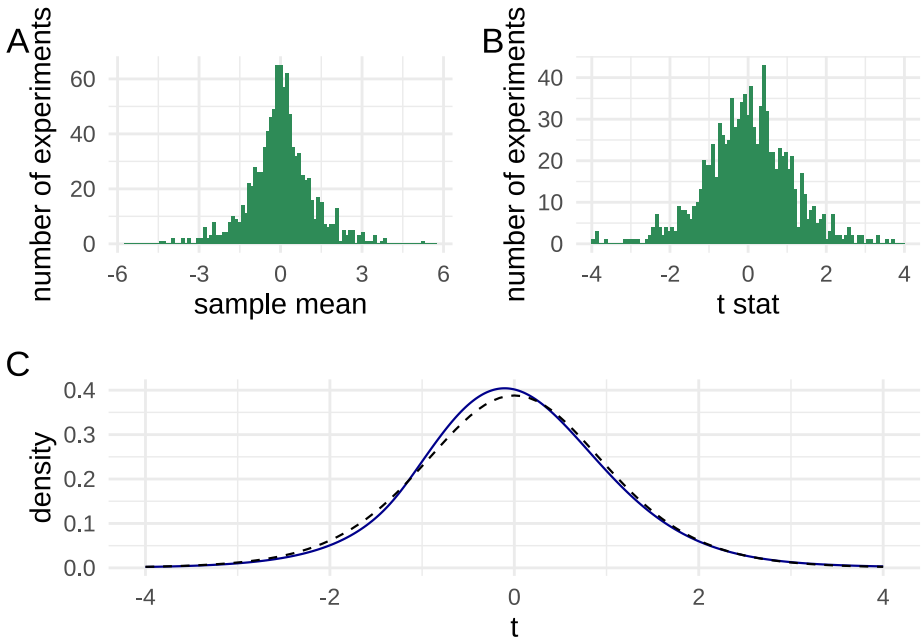
Next, we'll need to scale our average. If our average is 1, then is this close to 0? How about 0.5? Or 5? Or 15? It's impossible to know because you don't know range of the average dial's swings. So your scaling factor should be proportional to the magnitude of the average deviations you've observed (we'll call this scaling factor $s_{\bar{x}}$).

With this information in hand, we have enough to build a statistical model of our device's behaviour. To do this, we just go where there is no treasure and perform the following steps: 1) Take 10 readings; 2) work out an average ($\bar{x}$); 3) scale it by our scaling factor ($s_{\bar{x}}$); write down our scaled measurement (which we'll call t), and repeat! Once we've done this many many times, then we'll have a nice distribution or statistical model of how our device behaves when

Table 1.1: First 10 rows of simulated experiment data

Sample Mean	Sample sd	N	Std Error	t	index
-1.76	3.38	10	1.07	-1.65	1
0.50	4.29	10	1.36	0.37	2
0.08	0.79	10	0.25	0.33	3
-0.87	3.31	10	1.05	-0.83	4
-2.46	3.50	10	1.11	-2.23	5
1.08	3.93	10	1.24	0.87	6
1.09	3.47	10	1.10	1.00	7
0.07	0.75	10	0.24	0.31	8
3.28	5.46	10	1.73	1.90	9
-0.29	3.53	10	1.12	-0.26	10

there isn't any treasure. Of course, we don't have to do this for real. We can just simulate it! Feel free to play around with the simulation, to change the numbers, and to see how this influences our statistical model.



In panel **A** we can see the distribution of the raw averges from our device. In panel **B** the averages have been scaled. Finally, in panel **C**, the histogram has been turned into a density plot. The blue line shows our scaled averages, and the dashed black line shows a t distribution. As we increase the number of experiments we simulate then the two lines should begin to overlap.

1.2.1.2 Using our device

We can use our statistical model of our device (our distribution of t values) to come up with a method for finding treasure. Our statistical model tells us what readings we'll see when **we haven't found treasure** and **how often** we'll see those readings. In the absence of treasure we'll see readings near the middle of the distribution very often and readings near the tails of the distribution less often. We might even say that, in the absence of treasure, it would be pretty **surprising** to see an extreme reading. Now we don't know anything about how the device behaves when it's around treasure, but we know what readings would be *surprising* if it **wasn't** around treasure.

We can use this fact to come up with a treasure hunting rule. When you see a **surprising** reading (that is, if our average of multiple readings, from a single hunt falls in the extreme tails) the we dig for treasure. When you see an **unsurprising** reading, move on to the next spot. Let's try it out!

Our 10 measurements are: 0.25; -0.54; -0.51; -0.75; 0.27; -1.62; -1.29; -0.21; 0.17; -0.83

Our $\bar{x}$ = -0.507

Our $s_{\bar{x}}$ = 0.204

This means that our scaled measurement, t = -2.485

Once we have our scaled reading t, we can ask how **surprising** it is. To do this, we just compare it against the distribution of measurements that we generated when we weren't around treasure.

96.7% of values from our simulation where closer to zero than our current value. Only 3.3% of values where further from zero than our current value.

Once we have a measurement of how surprising our value is, then we just need to set a threshold for when it's surprising enough to warrant digging. We'll call this threshold α, and we'll set it to 5% (for literally no reason in particular).

Now let's try using the rule. We'll do another simulation. We'll simulate many many hunts, and on each hunt there either will be treasure or there won't be treasure. Treasure will occur with the probability of P(treasure). We won't know this value, because we'll just randomly set it. For each hunt, we'll note down whether the rule told us to dig or move one. And we'll also record the ground truth to test the accuracy.

To asses the usefulness of our rule, we can evaluate the accuracy of our rule. There are a few ways to do this. We can look at overall accuracy. We can look at how often we missed treasure when there was treasure. We can look how often we dug for treasure when there wasn't any. Let's take a look at some metrics.

The rule seems to work pretty well in terms of accuracy. But how much is accuracy dependent on the actual probability of finding treasure? Let's run two

Table 1.2: First 10 rows of simulated treasure hunt data

Had treasure?	Did dig?
No	No
No	No
No	No
No	No
No	No
No	No
Yes	Yes
No	No
No	No
No	No

Table 1.3: Our treasure hunting metrics. Accuracy: 0.83

Had treasure?	Did dig?	%
No	No	69.9
No	Yes	3.4
Yes	No	13.9
Yes	Yes	12.8

more quick simulations where we set the probability of treasure actually being present to 1 (treasure all the time) or 0 (treasure none of the time).

But maybe just looking at accuracy isn't the best. After all, there are two ways in which we can be wrong. We can dig when we're not meant to, and we can move on when there's actually treasure. So let's split that accuracy percentage (or rather the

$$1 - accuracy$$

or "error" percentage) into two: 1) Digging when there's no treasure, and 2) moving on without digging when there was treasure. Now let's adjust P(treasure) and see how the two error rates fare.

When there was no treasure at all then the **false alarm rate** was 5.4 and the **miss rate** was 0

Table 1.4: Our treasure hunting metrics. Accuracy: 0.95

Had treasure?	Did dig?	%
No	No	94.6
No	Yes	5.4

Table 1.5: Our treasure hunting metrics. Accuracy: 0.46

Had treasure?	Did dig?	%
Yes	No	54.5
Yes	Yes	45.5

When there was treasure everywhere then the **false alarm rate** was 0 and the **miss rate** was 54.5

We can see that no matter what we do, the false alarm rate (digging when there is no treasure) never goes above ~5%, which is the same value we set for α. This is great because it means that we can with certainty set the upper bound of this error rate. And, we can do so knowing nothing about how much treasure there is to be found or how our device works in the presence of treasure. All we need is: 1) to know that *on average* the device points at zero when there's no treasure around and 2) to sit and watch the device for a long time and just record some scaled measurements that the device produces. In fact, we don't even need to do (2). We can just *pretend* to this by simulating the results, and we only need to input **one parameter**—the same value we that we needed for step 1. Everything else can just be made up.

I'm not going to talk much about the other error rate, because this isn't a course of frequentist inference. But we can estimate it based on some assumptions about how the device behaves *in the presence of treasure*. For example, if we know that treasure of a certain value results in the device pointing on average at 1, then we can calculate the **upper bound** of missing treasures smaller than that value. Trying to estimate the **upper bound** on this error rate is what you're doing when you're doing a **power analysis**. It's generally **a lot** harder to estimate this, because it involves a lot of guesswork. In comparison, estimating the first error rate is rather trivial.

1.3 Summary

What this rather long-winded demonstration was meant to show is that p values are very good at doing one thing. That thing is, controlling how often, in the long run, we'll make a particular kind of error. Deployed in the right context, they're very good at this. This all comes from a simple process: Setting the value of **one parameter**, running pretend experiments, and then comparing our data at hand to results obtained from our pretend experiments to **judge whether our data is surprising** or not.

Of course, our treasure-hunting scenario may not be exactly analogous to how science works. These means that deciding whether p values are useful or not is going to depend on how closely their real-world use case matches their ideal operating environment.

1.3.1 A short note on confidence intervals

I'll mention confidence intervals only briefly, but they follow the *exact* same logic as p-values. Let's say I collect some measurements, work out the average. I could scale this value with my scaling factor, could get a t value. I could then turn to my list of results from the pretend experiments (the sampling distribution) to work out my p value. However, I can also the sampling distribution to construct (the very poorly named) confidence interval.

How could we do this? Looking at the sampling distribution we constructed earlier we would see that values that are more than about 2.23 t units from 0 would be surprising. Using this information, I can ask myself the following question: If my device on average pointed at the current sample average, rather than zero, what data values would be surprising? The answer to this is, of course, values that are more than 2.23 t units from the sample mean. Having an answer in t units isn't very useful. But I know that I converted measurements to t units by scaling readings using the scaling factor $s_{\bar{x}}$. This means we can just un-scale the value in t units back into raw units using the scaling factor calculated from my sample. This means I can say that any values $\pm 2.23 \cdot s_{\bar{x}}$ from the sample mean would be surprising. Any values less than this, or in this range, would be unsurprising if my device, on average, pointed at my current sample mean. I could draw a line through these values, put little tails on this line, and *hey presto* I have a confidence interval.

Now just to be clear, just like with a p value, a confidence interval tells you what values would be surprising/unsurprising on an assumption of a certain value of the parameter. For the p value you set the parameter to 0 (or wherever else the device points when no treasure is around). For the confidence interval, I just set the parameter value to the estimate obtained from the current sample, but it's exactly the same idea.

Hopefully, this should make it clear what the confidence interval does and doesn't tell you. It doesn't tell you about the *probability of a parameter falling within a range* (the common misinterpretation). It tells you *frequency with which data from pretend experiments will fall within a particular range on the assumption that the parameter is equal to the observed value.* At no point are we making inferences about **parameters** or **true values** of parameters. We are holding parameters constant, doing pretend experiments, and then marking out the range of surprising and unsurprising data.

Now that we're all on the same page about p values and confidence intervals, and we have a good idea of where they come from let us the a look at some criticisms of p values.

```
## Warning: package 'magrittr' was built under R version 4.1.2
```

Chapter 2

Criticisms of p values

People have written lots of criticisms of p-values. A lot of these are of the form "p-values are bad because they don't do X", where X is not a design feature of frequentist inference. I'm not interested in these kinds of criticisms, because they seem pretty meaningless. Instead, I think that if we are going to criticise p-values it is better to look at the design features of frequentist inference and find fault there.

So what are the design features? In the last section, we saw how frequentist inference was very good at controlling the kinds of mistakes we made in our treasure hunt. To do this, all we needed was a model of how our treasure detecting device operated. If we only wanted to control *false positives* all we needed was a model of how it operated in the absence of treasure—we didn't even need to know how it behaved when there was treasure around! To build this model we needed one bit of information—that the dial *on average* pointed at 0 when there was no treasure. The whole model could then be built up by running lots of simulations (or pretend experiments) where this parameter (the average reading in the absence of treasure) was the only parameter we needed to set. Just doing this allows us to precisely set an upper-bound on how often we make false positives. That's a pretty powerful property, and it all comes from such a simple principle.

But are there some issues with this simple principle? We can try explore it a bit more and see where things start to break.

2.1 Same measurements from different devices

Let's imagine a new scenario. As before, you have a treasure hunting device (we'll call it d_1). You're using d_1 to hunt for treasure, and using the readings to decide whether to dig or not. At your first treasure hunting spot, you record

the measurements: 1, 0, 1, 3, 0, 1, 4, -1, 3, 4. You then average, and scale these measurements and get a t value of approximately 2.848. You compare this to what you found in your imaginary experiments and find $p = .019$. According to your rule, that means you dig. For far so good.

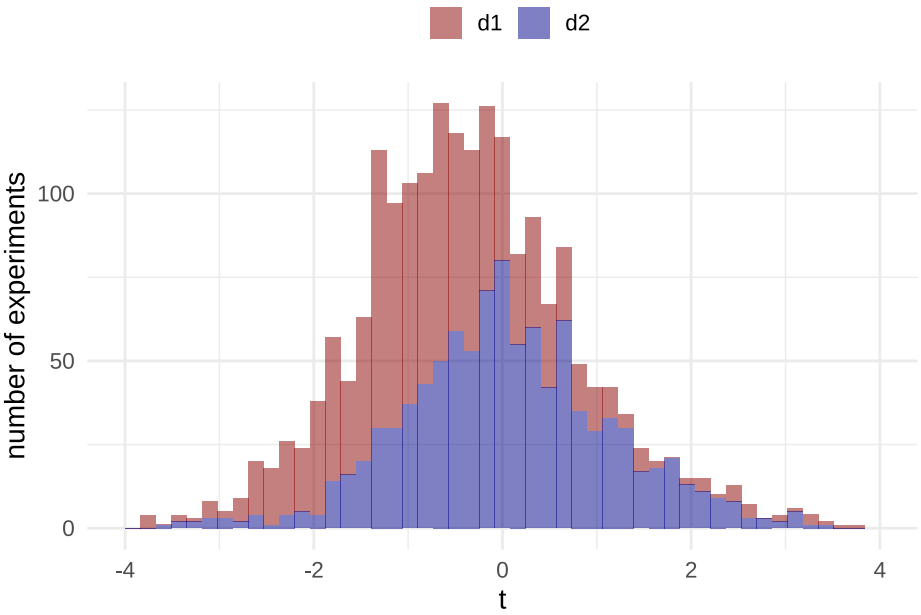
However, before you start digging, I run up to you and tell you that device d_1 is broken. I tested it before you left, and found that d_1 is incapable of measuring values bigger than 6. You look at your measurements again, and to your relief, they don't go anywhere near 6. Your highest measurement is only 4. But should you worry that the device couldn't register values of 6 or higher? And if so, why?

More generally, how would this fault with the device influence your treasure hunting strategy and would it change your view of when you think you should start digging? The intuition here might be a little unclear, so let's modify the example a little bit.

In the modified example, you want to be extra careful to avoid taking a broken device with you, so you take two measurement devices (d_1 and d_2). The devices are identical and, indeed, when you look at the measurements you can see that they've recorded an identical set of 10 numbers. Because the measurements are the same, you just pick whichever device and work out your scaled reading and decide whether to dig.

But not so fast, I again tell you that d_1 is actually broken and it is incapable of recording measurements higher than 6. I also tell you d_2 is working perfectly. What does this do to your inference? Does your inference change depending on whether you decided to look at d_1 or d_2? Remember, that the actual numbers produced by both machines are identical.

If you want to be a good *frequentist* then the answer to this question is a resounding *yes*. Even though d_1 and d_2 produced the exact same measurements, and despite these measurements being accurate, your inference will depend on the device you decided to look at. But why? Understanding the answer to this means going back to the sampling distribution we generated by running pretend experiments. Let's run some new pretend experiments for d_1 and d_2. The stimulations for d_1 will be modified slightly so that all values higher than 6 will be replaced with a 6.



As you can see, the distributions are different. This is because in those pretend experiments, the devices would behave differently. In our actual experiment (this treasure hunt), they didn't behave differently. They behaved exactly the same, and both behaved accurately. Remember, these distributions are what we use to make a judgement about whether our reading is surprising or not. We mark out sections of these distributions to find the range of values that are surprising and the range of values that are unsurprising. Because the shape of these distributions are different, the ranges that we mark out on each of them will be different. And consequently what counts as a surprising/unsurprising value on one distribution might not count as a surprising/unsurprising value on the other one.

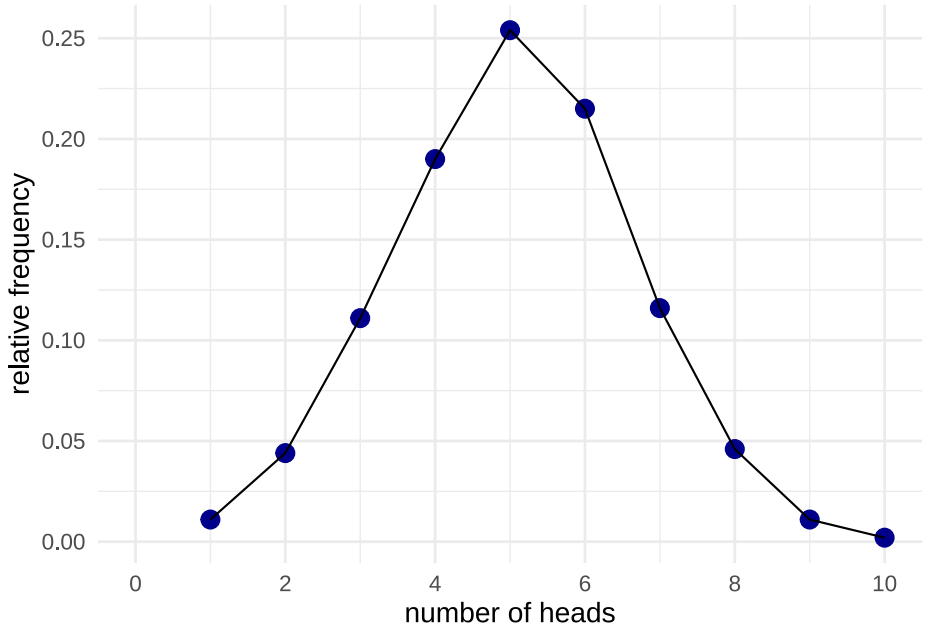
If you're being a frequentist then there's no getting away from the fact that because the devices have the *potential* to behave differently in situations other than the current situation, this *potential difference* must be accounted for. They factor into the calculation of the p value by changing the distributions and, therefore, we need to take account of these potential events in our inferences if we want to maintain our error control properties.

For some, the influence of imaginary events is madness. Jeffreys described this "madness" as follows:

> What the use of P implies, therefore, is that a hypothesis that may be true may be rejected because it has not predicted observable results that have not occurred. This seems a remarkable procedure (Jeffreys, 1961, p. 385)

2.2 The universe of possible events

To see another example of how potential events can influence inferences, let us examine a different scenario. In this scenario, we're going to make judgements about the fairness of a coin (fair coins being defined as coins that show heads with P(heads) = 0.5). We'll use the same procedure as our treasure hunting device. We will flip a coin that we *know* is fair a set number of times (let's say 10 times). We then count up **x** heads out of our total of **n** flips. We then repeat the procedure many many times. We can use this procedure to generate a distribution of possible data. Again, we can just simulate this.



Armed with this distribution, we can start making judgements about actual data. To produce some real data, I'll flip the coin I want to test and, at the end, I'll count up the number of heads. Let's say that I got 8 heads and 2 tails. Now you can make a judgement about whether this data is surprising or not. To do this, all you need to do is compare it to the simulated results above.

The p value for 8 heads in 10 flips is 0.109.

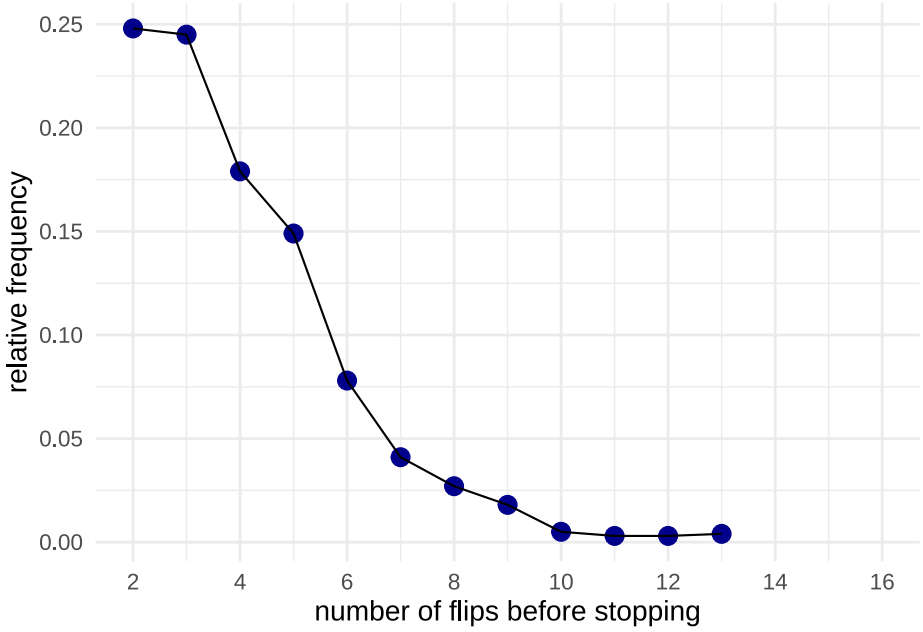
This result is not surprising on the assumption that the coin is fair (i.e., P(heads) = 0.5)

But save your judgement for now, because there's something that I have neglected to tell you. My plan wasn't to flip the coin 10 times. Instead, I decided that I would just flip the coin until it came up tails twice, and it just so happened that on this occasion this meant that I flipped the coin 10 times.

Does this fact change your inference? If our inferences are based on comparing

our actual data to possible data then we need to examine whether this sampling rule changes the possible data that could have been generated. That is, we need to take into account whether the data was generated by deciding to flip the coin 10 times or whether it just so happened that I flipped the coin 10 times, but really "in my head" I was going to stop when I got 2 tails. To so see why we need to re-run the simulations. In the new simulations for each sample we'll continue to flip the coin until it comes up with 2 heads, and then we'll stop. Sometimes this will mean that the coin is flipped 10 times, but sometimes we might flip it more, and sometimes we might flip it less.

We now can count up the relative frequency of getting 2 heads after 2 flips, after 3 flips, 4 flips, and so on. And we can draw a plot of this distribution.



From this new distribution, we can now ask: How often would you need to flip a fair coin 10 or more times before you got two heads? That is, is it surprising that we had to flip it this many times? Let's see how the inference differs.

For a fair coin (P(heads) = 0.5), about 98% of experiments would end before we got to 10 flips. Only 2% of experiments would run this long. Therefore, our result is surprising!

2.3 Summary

What these two examples (the broken device, and stopping rule example) show is that even when presented with the **same data** the inferences we make about

that data will be different if the realm of **possible**, but **not actual** results are different. That is, non-existent results influence our inferences. A broken device that still behaved accurately when we used it influences our inferences, and what we had going on inside our head when we collected our data also made a difference. Based on this, we can go ahead to imagine even more ridiculous examples.

For example, imagine that I build a device that is going to flip a coin to decide whether 1) to flip the coin n times or 2) flip it until it comes up tails x times. The device makes a decision, flips the coin, and it just so happens that on this occasion we get 8 heads and 2 tails. How do I analyse this set of data? Does the realm of possible data include the machine that makes the decision? What if I know what decision the device made? Do I still have to take into account the experiment that wasn't performed? And what if I have the results of two experiments, one that was performed as part of a mixture (using a machine to decide which of the two experiments would be performed) and one that was not performed as part of a mixture. If they yield the same data, then does the fact that one was part of a mixture mean that the conclusions should be different? For a frequentist, these can be pretty uncomfortable questions! In the next section we're going to see if we can find a way out of this bind.

```
## Warning: package 'magrittr' was built under R version 4.1.2
```

Chapter 3

An alternative to p values

Coming up with an alternative to p values requires us to rearrange our thinking a bit. So let's first get straight what we're doing with frequentist inference. In frequentist inference we set some parameter to a certain value (θ), we then generate data from imaginary experiments using that parameter setting, and we then compare our data to the data from those experiments. We then ask the question: "Given that parameter value, how surprising is our data?" At no point are we making inferences *about the value* of θ. We **set** the value, and we ask a question about our data in relation to **all the possible data** that might be generated.

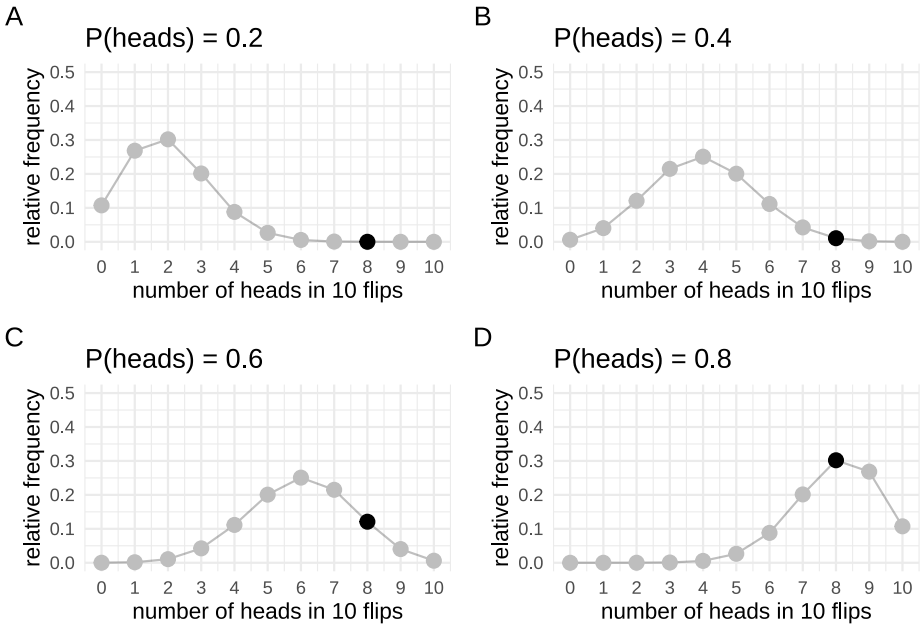
To think about what an alternative might look like, let us think back to our earlier example on the different meanings of probability. With p-values we thought about probability in terms of relative frequency. We were asking "how often?" questions. But I also mentioned another example. The example of being 90% sure that the accused committed a crime. If we want to be rational humans, when we make claims like this what we usually do is examine the evidence. We **compare** whether there is more evidence for the accused's guilt or the accused's innocence. That is, we take the courtroom evidence and examine whether it supports hypothesis 1 (the accused is guilty) or hypothesis 2 (the accused is innocent). To do this we balance of probabilities. Is is more probable that we'd see this evidence if hypothesis 1 was true, or is it more probable that we'd see this evidence if hypothesis 2 was true? (In a civil trial we'd just weigh up the probabilities, but in a criminal trial we'd have to also examine whether this difference in probabilities exceeds some threshold. We'll leave this issue of thresholds for now). Might we be able to apply the same kind of thinking to statistical evidence?

To understand the concept of statistical evidence, let's go back to our coin flipping example. In our coin flipping example, we collected 10 flips and found 8 heads and 2 tails. Our frequentist analysis asked something like, "is this data surprising?". But we could ask another question. That question might go

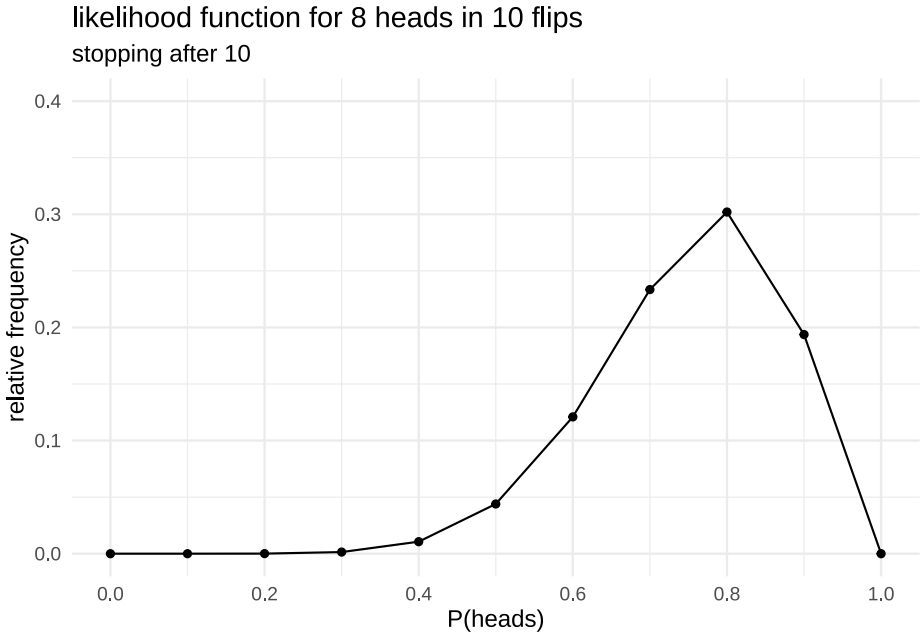
something like this: "Is it more likely that the **bias is 0.6** or that the **bias is 0.8** given that we'd obtained 8 heads in 10 flips?"

To try and answer this question, we'll again create some simulations. We'll start by creating two **sampling distributions**. For now we'll keep things simple and we'll create these sampling distributions on the assumption that I intended to flip the coin 10 times. To create our sampling distributions we'll first set θ to 0.6 and run the simulations, and then we'll set θ to 0.8 and run the simulations. I know the distribution they'll follow, so I'll just compute the distributions directly rather than actually running the simulations.

We can draw the distributions of the possible data that would occur for different values of P(heads) = θ. In each of the plots, our actual observation will be highlighted. Although we're "simulating" all possible observations, you'll see that we're only going to care about our **actual** observation. We will want to know the relative frequency with which **that** result occurs, not the frequency of results that didn't but might've occurred. I'm going to draw several distributions not just two that correspond to the values of θ that we're interested in.



Let's take these plots and create a new one out of them. Since we're just interested in **our specific observation** we'll take all the marked points and put them on a plot of their own. Now we'll still have relative frequency on the y-axis, but on the x-axis we won't have the observation anymore (because we're only focused on one specific outcome). Instead, we'll have θ on the x-axis.



This new plot that we're created illustrates what's know as the **likelihood** function. The likelihood function describes the relationship between values of the parameter and **our data**. It's made up of slices of the sampling distribution- the slices that correspond to our actual observation. Remember that when we were doing inference with the sampling distribution we were looking at the extreme tails of the sampling distribution. That is, we were interested in the entire shape of the sampling distribution. Now we're instead only interested in the thin slice that corresponds to our observation.

3.1 Doing inference with likelihoods

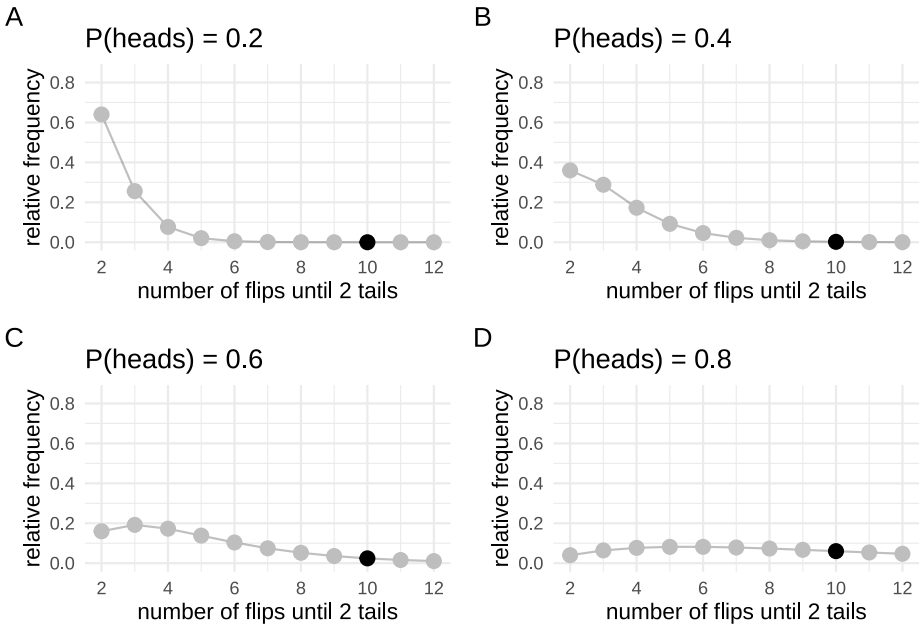
The likelihood plays a key role in Bayesian inference. Inferences on the basis of likelihoods are derived from what is known at the *law of likelihood*. Simply stated, the *law of likelihood* says that for a given pair of hypotheses—for example, $\mathcal{H}_1$ that the coin bias is P(heads)=0.6 and $\mathcal{H}_2$ that the coin bias is P(heads)=0.8—then data support $\mathcal{H}_1$ over $\mathcal{H}_2$ if the likelihood of $\mathcal{H}_1$ exceeds that of $\mathcal{H}_2$. Or, put another way, if our data would be produced more often if $\mathcal{H}_1$ were true than if $\mathcal{H}_2$ were true, then the data provide support for $\mathcal{H}_1$ over $\mathcal{H}_2$ (See Hacking, 1965 Chapter 5, for both formulations).

This definition might seem a little opaque, but we can read these likelihood values straight off our likelihood plot. The height of the likelihood plot, at each value of θ, tells you the probability of obtaining your data given that value of θ. If the likelihood function is higher at $\theta = 0.8$ than $\theta = 0.6$ then the

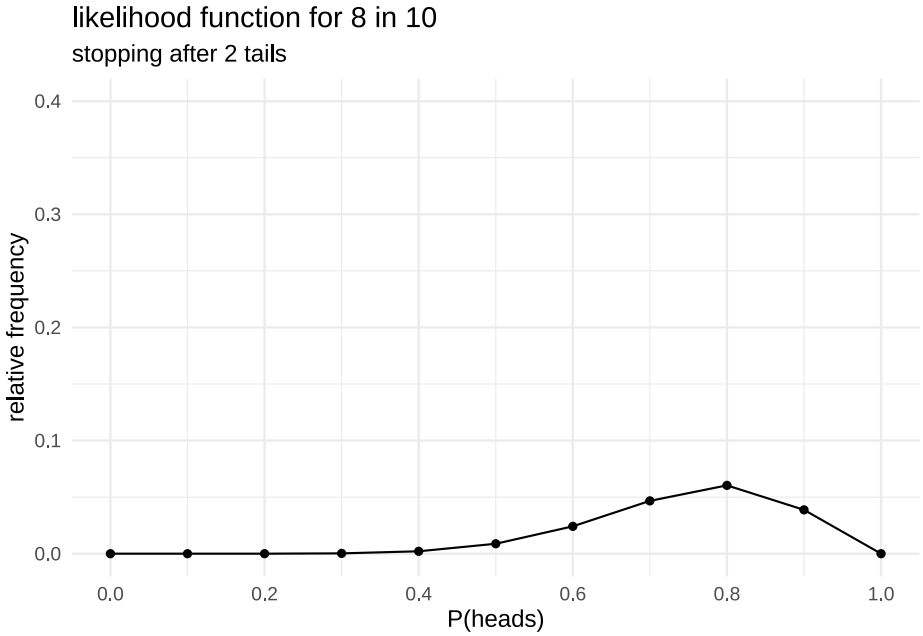
probability of obtaining our data would be higher if θ was 0.8 than it would be if θ was 0.6. Consequently, our data support the hypothesis that $\theta = 0.8$ *over* the hypothesis $\theta = 0.6$. A key point here, that's worth stressing, is that this is a comparison between two specific hypothesis. Does this data support this one specific hypothesis over this other specific hypothesis. What you're doing here is *weighing up probabilities* just like you would do in a courtroom.

3.1.1 A brief detour back to sampling rules

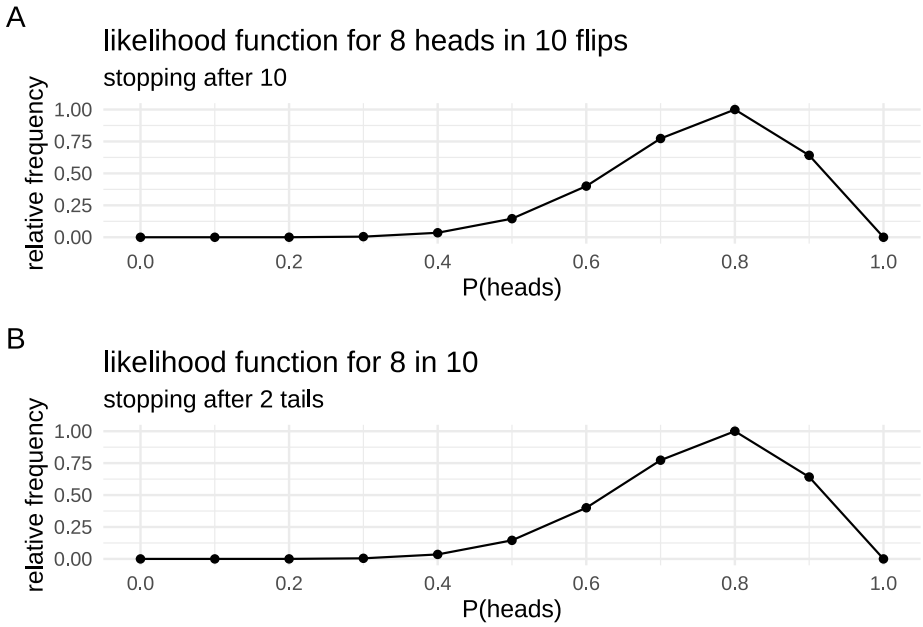
Before we continue, let's just go back to something from the previous section. I made a big deal about how our sampling rules change the shape of the sampling distribution, and that this then changes the inferences that we make. This is the case even if nothing changes about our actual data. But do different sampling rules change the likelihood? To test this out, we'll generate a new set of sampling distributions using the other sampling rule (sampling until we get 2 heads). And from these sampling distributions we'll generate some likelihoods.



We can see that these sampling distributions look very different to the sampling distributions that we generated above. But what we're interested in are just the highlighted points, because we'll use these to generate our likelihood.



The new likelihood might, at first glance, look different to the one we generated earlier, but it's just a scaled version of the earlier likelihood. We can check this just by rescaling the two likelihoods so that they both have a max of 1.



Now that they're been rescaled we can see that they're they same. Note that scaling changes the absolute distance between points on the likelihood, but it

doesn't change the *relative* distance between the points. When we want to know the difference between two likelihood values we take the *ratio* of these two values. The ratio gives us the *relative distance* between the heights on the likelihood function, and the *relative distance* doesn't change with scaling.

More importantly, however, what this demonstrates is that when we do inference with **likelihoods** instead of **sampling distributions**, things like stopping rules, data that wasn't collected but might have been collected, and all those other sorts of things that were tricky about p-values don't come in to play. We only have the worry about **the data we actually have**, and the **likelihood** which relates **parameter values** to **data**.

3.1.2 The likelihood ratio

The likelihood ratio is going to be our measure of evidence of how much the data supports one hypothesis over another. If the likelihood at point one (θ_1) is four times the larger than the likelihood at point two (θ_2) then the data are four times more likely under the hypothesis $\theta = \theta_1$ than the hypothesis $\theta = \theta_2$. Or simply put, the data supports the hypothesis $\theta = \theta_1$ over $\theta = \theta_2$ but a factor of 4 to 1.

Let's look at the likelihood for our actual data and our two hypotheses about the coin bias. Just to drive home the point that the sampling rule doesn't matter, I'm going to work out the likelihood ratio for the sampling rule where I flip the coin 10 times and the sampling rule where I flip the coin until I get 2 heads and just happen to flip it 10 times. We'll see that the absolute values of the likelihoods change (as we saw in the plots above), but that the likelihood ratio between the hypotheses don't change.

To make sure that the numbers work out correctly, I won't use simulations to generate the likelihoods. Instead I'll just generate each likelihood with the relevant formula.

First, for version 1, where I flip the coin 10 times (**binomial** sampling rule).

The likelihood for $\mathcal{H}_1$ (P(heads) = 0.6) is 0.12

The likelihood for $\mathcal{H}_2$ (P(heads) = 0.8) is 0.3

The likelihood ratio is 0.4

The data are 0.4 times more probable under $\mathcal{H}_1$ than $\mathcal{H}_2$

Second, for version 2, where I flip the coin until I get 2 hears (**negative-binomial** sampling rule).

The likelihood for $\mathcal{H}_1$ (P(heads) = 0.6) is 0.1

The likelihood for $\mathcal{H}_2$ (P(heads) = 0.8) is 0.24

The likelihood ratio is 0.4

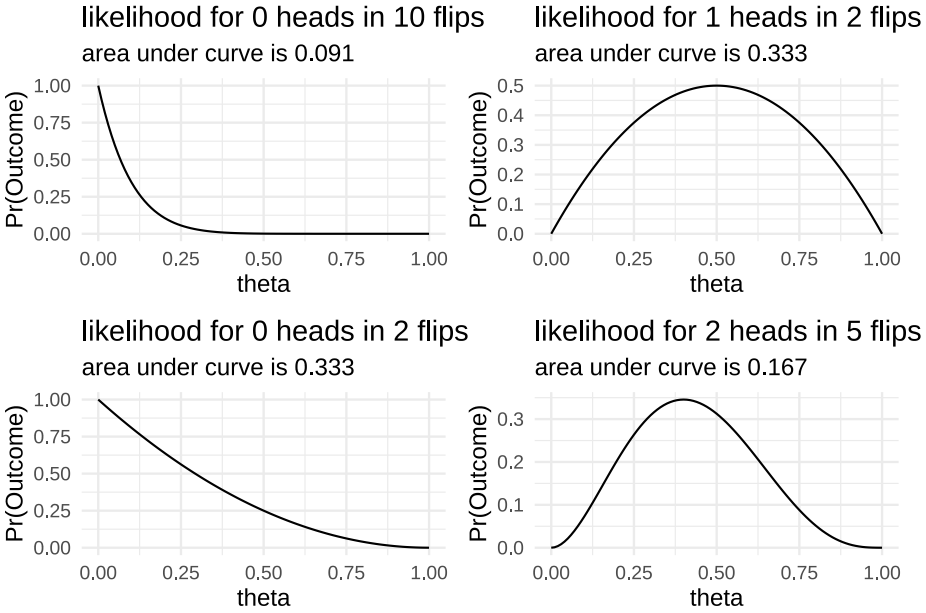
The data are 0.4 times more probable under $\mathcal{H}_1$ than $\mathcal{H}_2$

3.1.3 A note about likelihood functions and probability distributions

One common misconception about likelihood functions is that they're probability distributions. This misconception can come in a few different forms, so it's worth just stressing again what a likelihood function is.

First, we can tell a likelihood function isn't a probability distribution, because for a probability distribution the area under the curve would have to sum to 1. Each point on a probability distribution gives the probability of a specific event. The whole curve describes all the events that could happen, and the area under the curve gives the probability that one of the possible events happens. That is, it is the sum of all the individual probabilities of the different events.

In the plot below, we can see the likelihood functions for different events (different numbers of heads in 10 flips). We can see that the area under the curve varies in each case. If these were probability distributions then the area under the curve in each case would be 1.



This misconception about likelihood functions being probability distribution often takes the form of thinking that the likelihood function tells use the probability of the parameter being a specific value. That is, it tells us that there's a higher probability that $\theta = \theta_1$ than $\theta = \theta_2$, given our data. Put another way, this misconception states that the likelihood tells us $p(\theta|y)$. This quantity, however, is what's know as the **posterior probability**. Rather, the likelihood

tells us the reserve conditional, or $p(y|\theta)$. That is, it tells use the probability of obtaining our data given different values of the parameter.

To emphasise that the likelihood is not a probability distribution it is often denoted $\mathcal{L}(\theta|y)$.

3.2 Testing more complex hypotheses

So we've seen that comparing likelihoods (by taking their ratio) can tell us which hypothesis is better supported by the data. However, there's a couple of problems with what we've done up until now. First, how do we know explicitly set a threshold for when we would start digging for treasure. Is there also a threshold for likelihood ratios? To answer this question, we're going to have to take into account a lot of additional factors. And the answer to this question is probably going to be context-dependent. For example, if we're placing bets on hypotheses, we're probably going to want to take into account the relative pay-offs. If we're using evidence to decide somebody's guilt in a court case, we're probably going to want to take into account things like "reasonable doubt". In short, there's not a straight forward answer to this question, so we'll set it aside for now. Instead, we'll turn to the second problem.

The second problem with what we've done up until now is that we've just been comparing single point hypothesis. We can can say, for example, whether the data supports P(heads) = 0.5 over the hypothesis P(heads) = 0.8, and we can quantify this level of support. But usually, we are not comparing two simple hypotheses like this. Our hypotheses take a more complex form like: "Is the coin fair?"

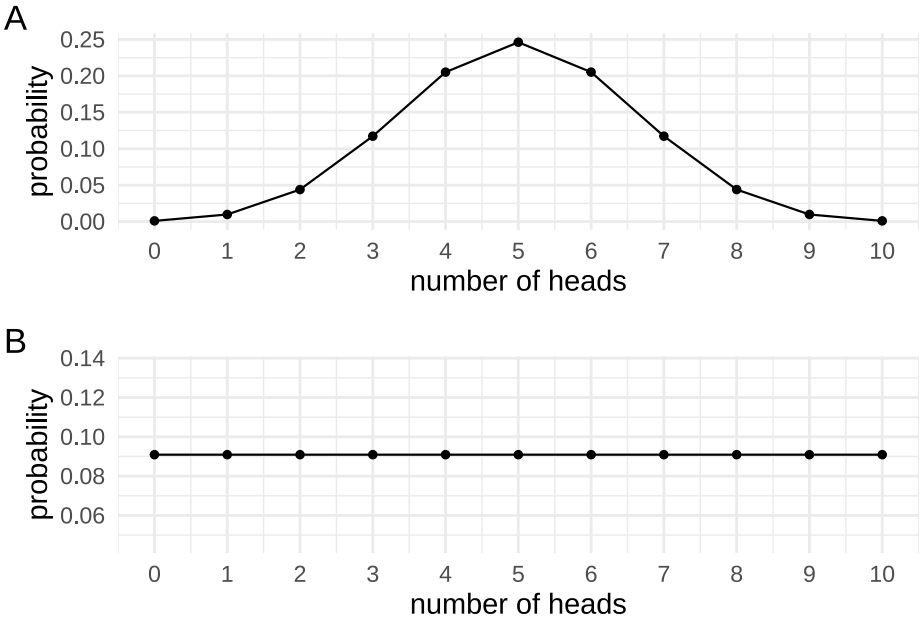
How might we go about answering this question?

To come up with a way to answer this question we're going to think about hypotheses in terms of **predictions**. Our first hypothesis, $\mathcal{H}_0$, will be that the coin is fair. And we'll say a fair coin has a bias of 0.5. What do we predict will happen if we flip the coins 10 times? Most of the time it'll show around about 5 heads and 5 tails, but it will also rarely show 1 head and 9 heads etc. If we plotted it, it would just be our sampling distribution from before.

For $\mathcal{H}_1$, that the coin isn't fair, what do we predict will happen if we flipped it 10 times? Before we can work this out we need to think a little bit about what it means for a coin not to be fair. For now, let's say that it means that it can have some bias between 0 and 1, but that we don't know what it is. For our fair coin, if we collected a very large number of samples the most common outcome would be 5 heads and 5 tails, but would be the most common outcome with our unfair coin? Would it be 5 heads and 5 tails? Would it be 0 heads? 1 head? 9 heads? Do we have any grounds for **predicting** that one outcome would be more common than another outcome? We arguably do not. If so, then if I asked which of the 11 possible outcomes (from a sample of 10 coin flips) is

more probable than the others you might say none. If none of the outcomes are more probable than any of the other outcomes, and given that there's 11 possible outcomes, then our prediction must be that each outcome has a 1 in 11 chance of occurring.

Below, we can see plots of our two predictions. First, what we would predict if we knew the coin bias was 0.5, and second what we would predict if we had no reason for favouring one outcome over another.



Now that we have a intuition for hypotheses in terms of predictions, let use formalise it a bit. And instead of thinking about all the data that might be produced let's just try and think about the probability of obtaining our data of 8 heads in 10 flips. If a coin is fair, then in 10 coin flips there are exactly 2^{10} possible sequences and 45 of these sequences would give 8 heads in 10 flips. Therefore, if the coin is fair, then the probability of obtaining our result of 8 heads in 10 flips in $\frac{45}{1024}$, or about 0.044. I've worked this out exactly, but we'd get the same value if we ran the simulations, or if we just looked at the likelihood function at $\theta = 0.5$. This is after all, what the likelihood function tells use: the probability of obtaining our data for a given value of the parameter.

Now on to the more complex example where the coin bias is some unknown value between 0 and 1. What now is the probability of obtaining our data. One good strategy of dealing with unknowns is to average across the possibilities. For example, if I didn't know what the coin bias was, but I knew it could either be 0.5 or 0.6, then to work out the probability of obtaining our current data I could just work out the probability of obtaining our current data if the bias was 0.5 (~0.044), and then work out the probability of obtaining our current data if

0.6 (~0.121), and then just average them together (~0.082). Again, I just take the values from the likelihood function at $\theta = 0.5$ and $\theta = 0.6$ and aveage them together.

But in our example it's not just the case that the bias of the coin could be 0.5 or 0.6. For our second hypothesis we said it could be any value between 0 and 1. That is, hypothesis is the set $\Theta = \{\theta_1, \theta_2, ..., \theta_n\}$ where each θ_1 to θ_n is some value between 0 and 1. To keep things simple for now, we'll say that $\Theta = \{\theta_1 = \frac{0}{10}, \theta_2 = \frac{1}{10}, ..., \theta_{11} = \frac{10}{10}\}$. An average is just a sum where each value is multiplied by $\frac{1}{n}$, therefore, the average across these 11 values would be:

$$\sum_{i=1}^{11} \mathcal{L}(\theta_i|\mathbf{y}) \cdot \frac{1}{11}$$

This gives a value of approximately $\frac{1}{12}$, which is pretty close to the value of $\frac{1}{11}$ we worked out earlier. Why is it not the same? Well, earlier, we said it could be **any** value between 0 and 1. We're only looking at 11 values. Let's instead look at 101 values between 0 and 1. Now $\Theta = \{\theta_1 = \frac{0}{100}, \theta_2 = \frac{1}{100}, ..., \theta_{101} = \frac{100}{100}\}$. Now we get a value that's even closer to $\frac{1}{11}$. To get to exactly $\frac{1}{11}$, however, we're going to have to look at even more points. Instead of spacing the points out by $\frac{1}{10}$ or $\frac{1}{100}$, we're going to need infinitesimally small spacing. That's means we just switch out the sum for an integral, but the logic is the same. We're still just taking an average.

$$\int_{\theta \in \Theta} \mathcal{L}(\theta|\mathbf{y}) d(\theta)$$

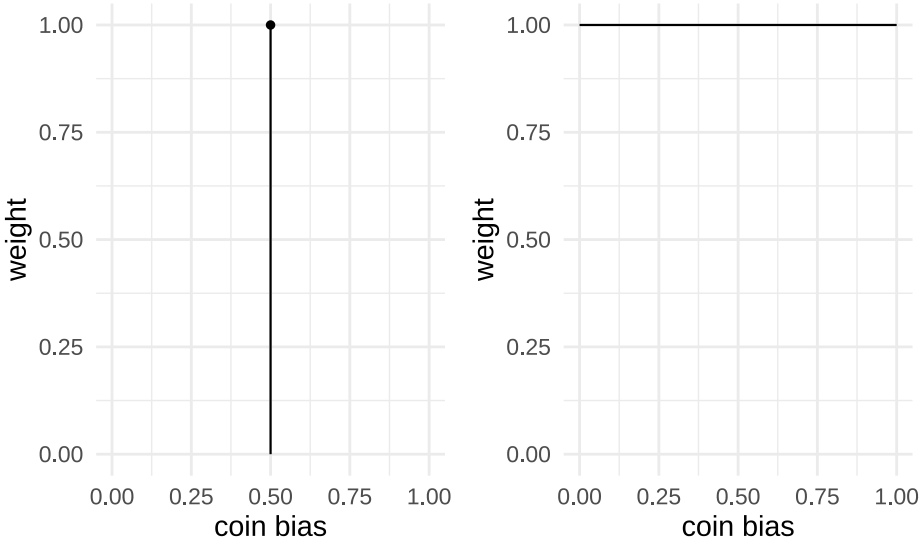
Now that we're taking an integral, we get exactly $\frac{1}{11}$.

Now that we have these two values: First, $\frac{45}{1024}$, which gives the probability of obtaining 8 heads in 10 flips if $\theta = 0.5$, and second, $\frac{1}{11}$, which the probability of obtaining 8 head in 10 flips if θ was some unknown value between 0 and 1, what can we do with them? Well, we can just take the ratio! Just like we did with the two simple point hypotheses, we can also take the ratio between our simple point hypothesis and our more complex hypothesis. Taking this ratio tells us that's we'd be $2\frac{34}{495}$ times more likely to see our data if θ was some unknown value between 0 and 1 than if $\theta = 0.5$.

Thinking back to the *law of likelihood* that we covered at the start of this section, we said if our data would be produced more often if $\mathcal{H}_1$ were true than if $\mathcal{H}_2$ were true, then the data provide support for $\mathcal{H}_1$ over $\mathcal{H}_2$. This is exactly the number that we've just worked out.

3.2.1 There's more than one way to average

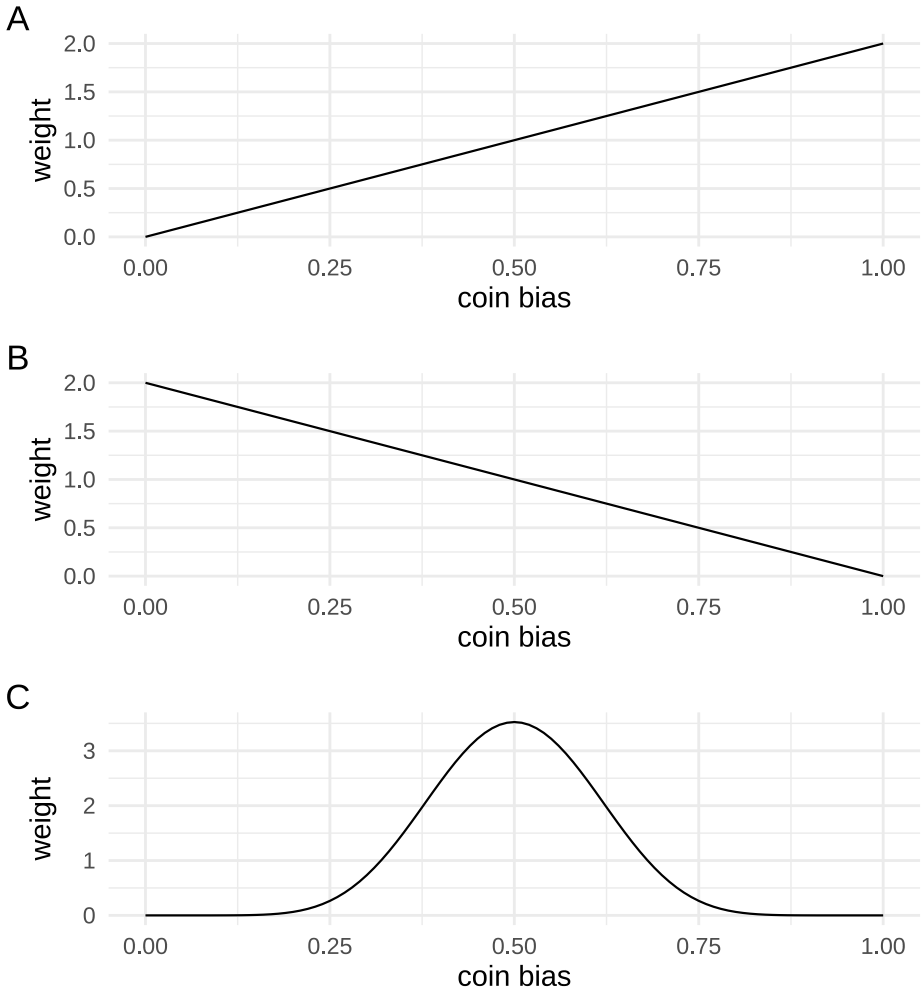
Above we worked out two values. The second number we calculated by averaging the likelihood function, but the first number we calculated by just taking a single point on the likelihood function. So one involved an average and the other did not. Or did it? We can actually think of both as involving an average of the likelihood function. They're just different kinds of average. We can view both as taking a *weighted average*, where different values contribute more or less to the average. For the second number, all values in the average were *weighted equally*. That is, it was just like a regular average. For the first, it can be viewed as taking an average where the likelihood value for $\theta = 0.5$ is given a weight of 1, and all over values of given a weight of 0. We could visualise these weighting in the plots below.



These *weightings* can be thought of as *probability distributions*. We are going to call these **priors**. Mathematically, they represents the weights that we apply to the values that we average together. But what do they represents *conceptually*. One way to think of them is that they *represent our beliefs about the parameter value* (in this case the *coin bias*). Or, that they represent our *model of the hypothesis*-that is, they represent what the hypothesis has to say about the parameter value. So the fair coin hypothesis represents is a model that says *the coin bias is exactly 0.5*. The other hypothesis is a model that say *all values of the coin bias between 0 and 1 are equally probable*.

At the start of this section we said that for our alternative for a fair coin we'd say that all values of the coin bias were equally likely, and this is what we'd mean by an unfair coin. But this is only one possible model of an unfair coin. We might actually think that if a coin is unfair then it'll show heads far more often than tails. Or, we might think that unfair coins will show tails more often than

heads. We might even think that unfair coins will behave very similarly to fair coins, but they'll just outcomes of 5 head and 5 tails a little bit less often than the $\frac{252}{1024}$ that we'd see with a perfectly fair coin. These are all different *models* that we might have about unfair coins. We can represent these hypotheses in terms of what they say about the coin bias parameter. That is, we can represent them as *weights* or *priors*. We'll learn more in the next section about how to specify these, but for now I'll just generate some plot.



In panel **A**, values of the coin bias closer to 1 (show heads all the time) and given more weight than values closer to 0 (never show heads). This means we except the coin to show heads more often. In panel **B**, we see the opposite. Finally, in panel **C**, we weight values closer to 0.5 (fair) higher than values closer to 0 or 1. That is, we don't think the coin bias is exactly 0.5, but we think values closer to 0.5 are more probable than values further away from 0.5.

Now that we're taking a weighted average, our earlier formula before:

$$\int_{\theta \in \Theta} \mathcal{L}(\theta|\mathbf{y}) d(\theta)$$

Now just becomes:

$$\int_{\theta \in \Theta} \mathcal{L}(\theta|\mathbf{y}) p(\theta) d\theta$$

In words we'd read this as:

> *The probability of obtaining our data under the specified model is equal to the integral of the likelihood (the model of the data) multiplied by the prior (the weights).*

We might denote this as $p(Y|\mathcal{M}_i)$ or simply $\mathcal{M}_i$. When comparing two models—for example, $\mathcal{M}_1$ and $\mathcal{M}_0$, we take the ratio as follows:

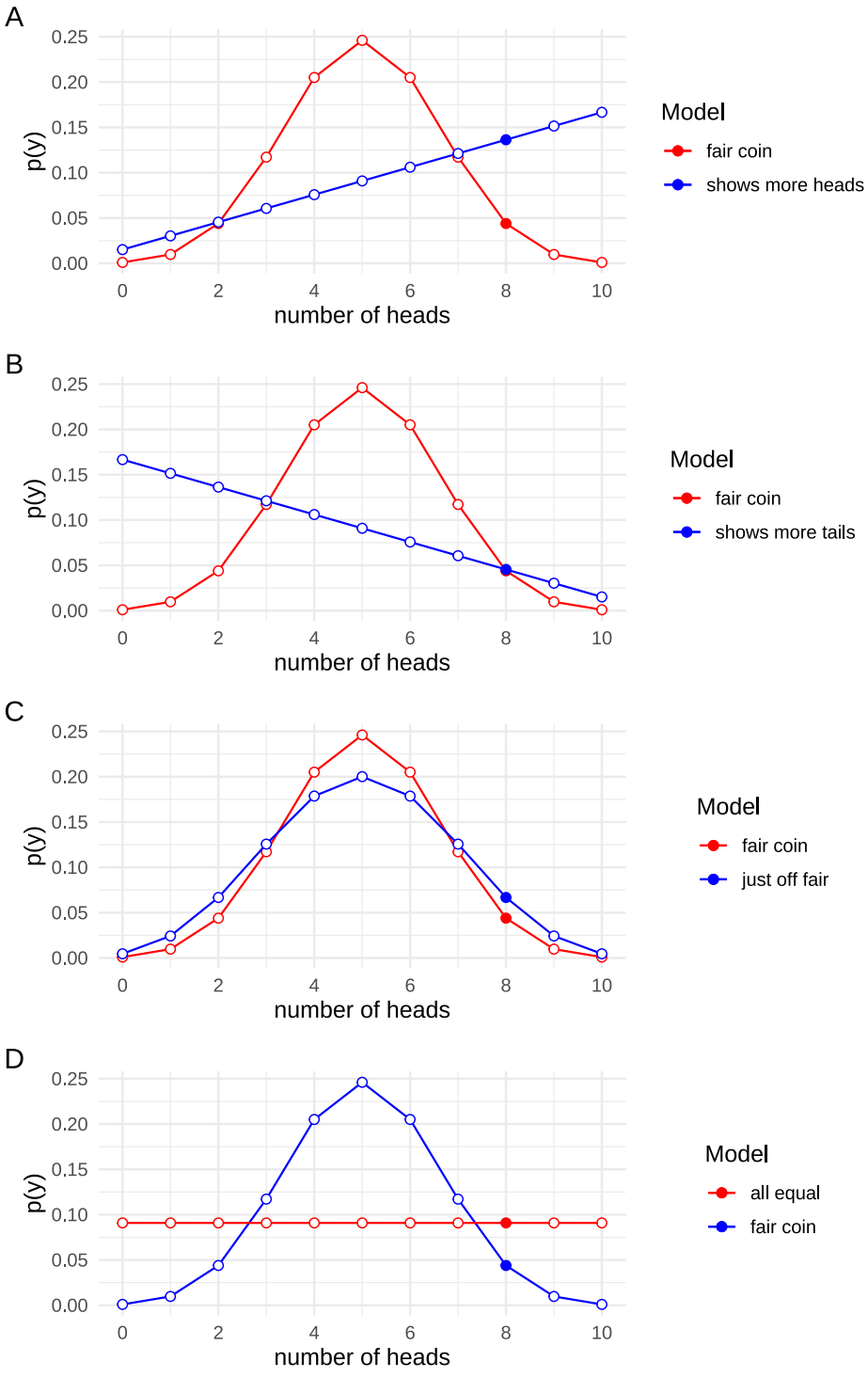
$$\frac{\mathcal{M}_1}{\mathcal{M}_0}$$

Try not to be too intimidated by the formula above. It just means that we're working out the probability of obtaining our data for a given value of the parameter, and that we're doing this for a range of parameter values. And finally, we're taking a weighted average of these. There's only 3 parts to the formula.

1. The likelihood, which tells us the probability of obtained our data at a specific value of the parameter: $\mathcal{L}(\theta|\mathbf{y})$
2. The prior, which determines the weights for weighted average of the likelihood values: $p(\theta)$
3. The integral, which performs the "averaging" across all the different values of the parameter range: $\int_{\theta \in \Theta} ...d\theta$

3.2.1.1 Visualising predictions

We can also represent these different models of the coin bais in terms of what outcomes we'd predict, just like we did with the earlier predictions. In the next section, we'll also learn about how to turns priors into predictions, but for now we'll just look at some plots. In each of these plots we'll show what we would predict if the coin was exactly fair overlaid on each of these different models of *unfairness*. In each of the plots we'll highlight our actual outcome of 8 heads in 10 flips.


A
0.25
0.20
0.15
0.10
0.05
0.00
p(y)
0
2
4
6
8
10
number of heads
Model
fair coin
shows more heads
B
0.25
0.20
0.15
0.10
0.05
0.00
p(y)
0
2
4
6
8
10
number of heads
Model
fair coin
shows more tails
C
0.25
0.20
0.15
0.10
0.05
0.00
p(y)
0
2
4
6
8
10
number of heads
Model
fair coin
just off fair
D
0.25
0.20
0.15
0.10
0.05
0.00
p(y)
0
2
4
6
8
10
number of heads
Model
all equal
fair coin

In panel **A**, we see the predictions from our fair coin model against our predictions from a model where the coin shows heads more often. In panel **B**, we see the fair coin predictions against a coin that shows tails more often. In panel **C** we see our fair coin model again a model where the bias is just slightly off from fair. And finally in panel **D**, we see the predictions of the fair coin model against a model where we have no reason for thinking that one outcome is more likely than any other outcome.

In each of these panels we can weigh up the evidence for whether our data support one model over the other by looking at whether our data would be produced more often if $\mathcal{H}_1$ were true than if $\mathcal{H}_2$ were true. That is,w we can see whether the data provide support for $\mathcal{H}_1$ over $\mathcal{H}_2$ just by looking at whether the blue highlighted point is higher (more probable) than the red highlighted point (less probable).

```
## Warning: package 'magrittr' was built under R version 4.1.2
```

Chapter 4

The Bayes factor

The primary aim of this course is to learn how to compute and interpret Bayes factors. But what is a Bayes factor? Well it turns out that we've already computed a Bayes factor. The Bayes factor is just the ratio that we computed in the previous section. The Bayes factor is a metric that compares the relative probability of obtaining our data under one model compared to another.

When we computed these ratios, the Bayes factor, in the previous section, it was made up of two ingredients.

1. We had our likelihood that related parameter values to our data. It told us the relative probability of obtaining our data under different values of the the parameter (the coin bias)
2. We had priors, which assigned different probabilities to the different values of the parameter. These served as our hypotheses about the parameter (the coin bias), and they served as the *weights* for our average of the likelihood. We had one prior for each hypothesis.

To perform the computation itself, we multiplied the prior by the likelihood, and took the weighted average, by taking the integral. Mathematically, we did the following:

$$\mathcal{M}_H = \int_{\theta \in \Theta_H} \mathcal{L}_H(\theta|\mathbf{y}) p(\theta) d\theta$$

We did this for each hypothesis (e.g., $\mathcal{M}_0$ and $\mathcal{M}_1$), and then took the ratio $\frac{\mathcal{M}_0}{\mathcal{M}_1}$. And this ratio was the Bayes factor.

$$\text{BF}_{01} = \frac{\mathcal{M}_0}{\mathcal{M}_1}$$

4.1 Computing Bayes factors with `bayesplay`

To actually compute Bayes factors we're going to use an `R` package called `bayesplay`. The `bayesplay` package allows you to specify likelihoods and priors, and to perform some operations on them. The operations, described above.

Let's step through some `R` to see how we would actually do it:

First, we specify the likelihood. We'll specify a likelihood of the **binomial** family. It requires 2 inputs. The number of heads, and the number of flips.

```
data_model <- likelihood(family = "binomial", successes = 8, trials = 10)
```

Second, we'll specify the two priors. For the first, we'll set a **point** prior at 0.5 to represent our fair coin hypothesis.

```
fair_coin <- prior(family = "point", point = 0.5)
```

Next, we'll use a **uniform** prior to represent equal weights for all values between 0 and 1. There's two ways we can do this. First, we can use the **uniform** family.

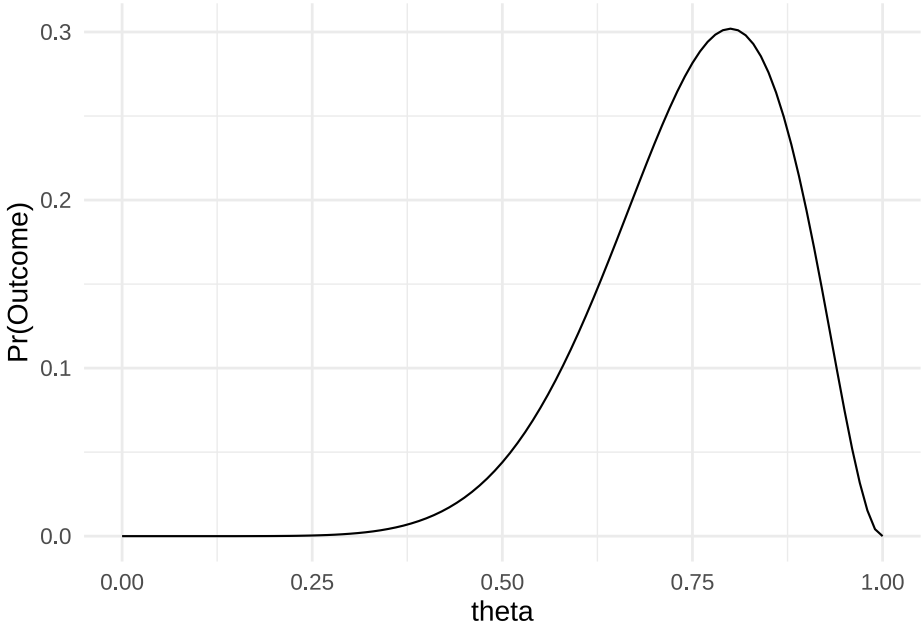
```
unfair_coin <- prior(family = "uniform", min = 0, max = 1)
```

But we can also use the **beta** family. The result will be the same in either case.

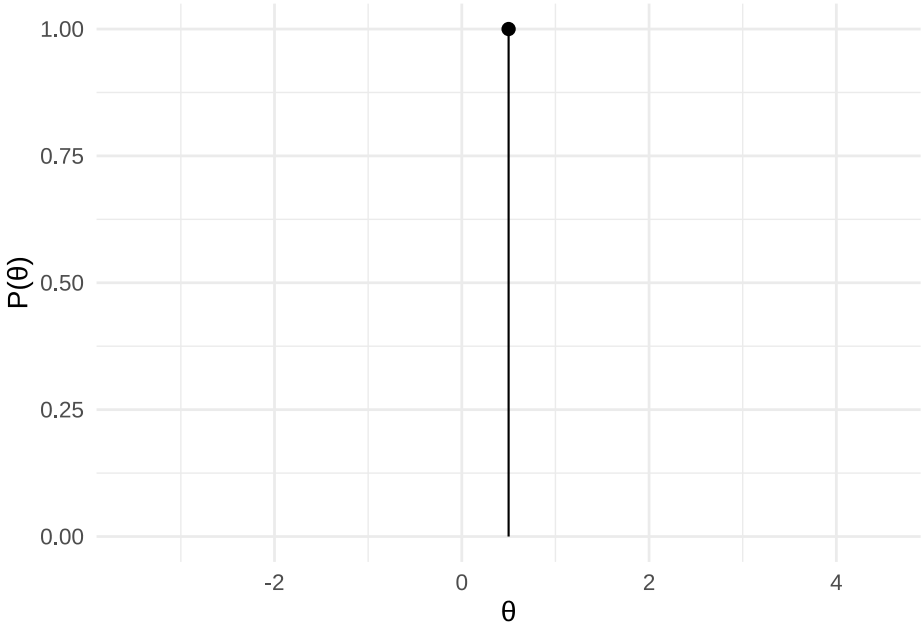
```
unfair_coin <- prior(family = "beta", alpha = 1, beta = 1)
```

Since we've been visualizing everything so far, we can also visualise the likelihood and priors we've just defined. To do this, we just use the `plot()` function.

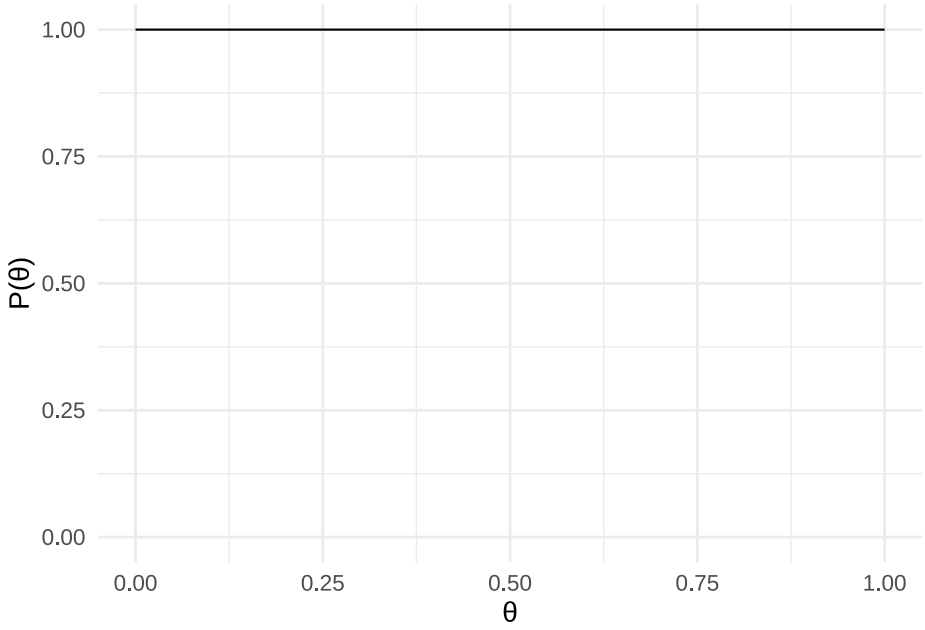
```
plot(data_model) +
  theme_minimal(14)
```




```
plot(fair_coin) +
  theme_minimal(14)
```



```
plot(unfair_coin) +
  theme_minimal(14)
```

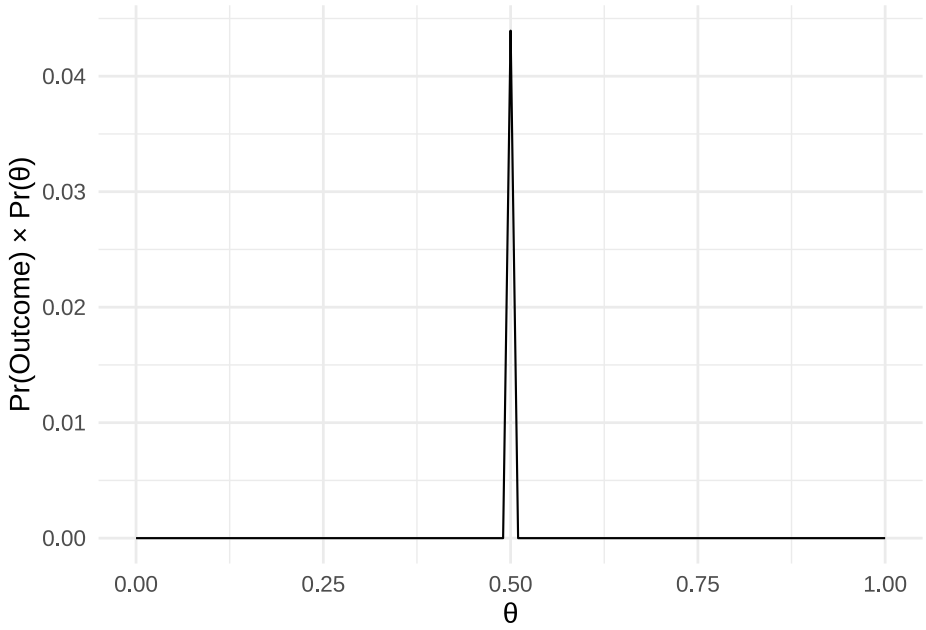


The next step was to multiply the likelihood by the prior. We'll do this for our likelihood and each of the prior.

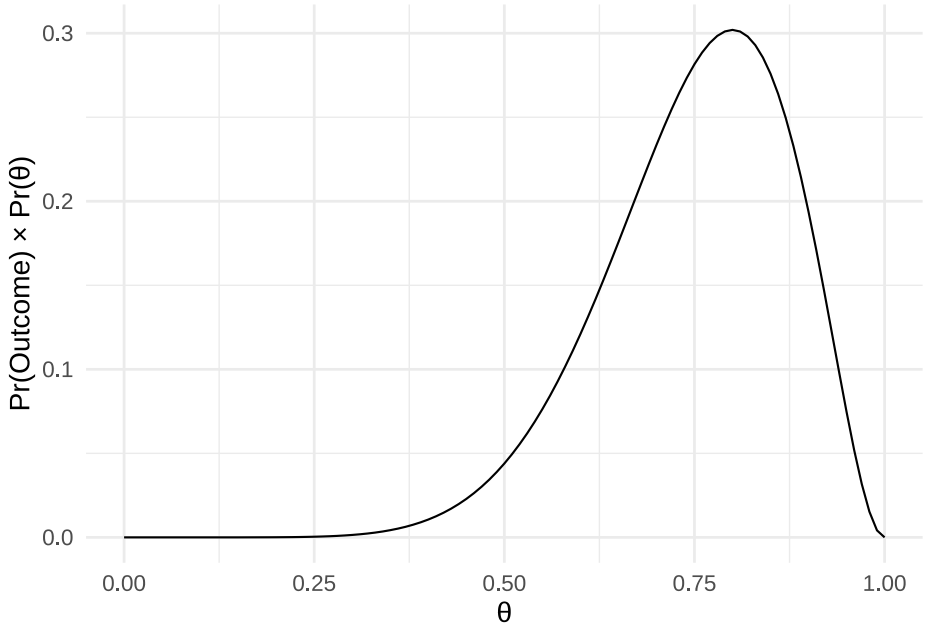
```
m0 <- data_model * fair_coin
m1 <- data_model * unfair_coin
```

Plotting these isn't super informative, but we can do it anyway.

```
plot(m0) +
  theme_minimal(14)
```




```
plot(m1) +
  theme_minimal(14)
```



The next step is to work our the area under each of these curves. That is, work out that integral. To do this, we just use the `integral()` function.

```
int_m0 <- integral(m0)
int_m1 <- integral(m1)
```

And finally, we just take the ratio of these two values to get the Bayes factor.

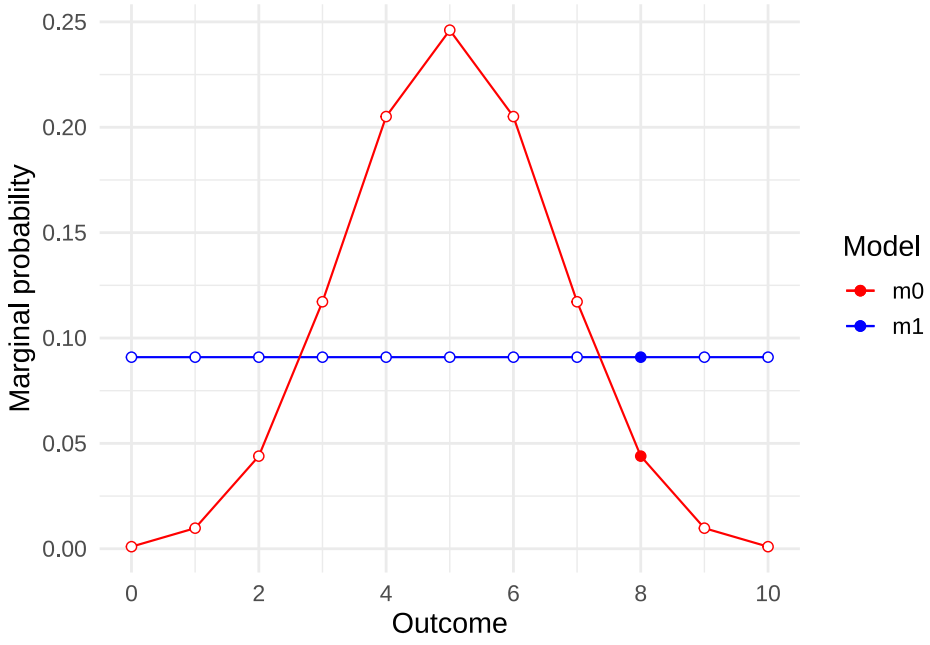
```
bf <- int_m1 / int_m0
```

The bayes factor is 2.069

And that's all there is to it.

However, we can do a little more. In the previous section we saw plots that showed the predictions of each model and highlighted our specific observation. We can also generate these easily with `bayesplay`. We simply use the `visual_compare()` function, and give the two models (the likelihood $\times$ the prior) as inputs.

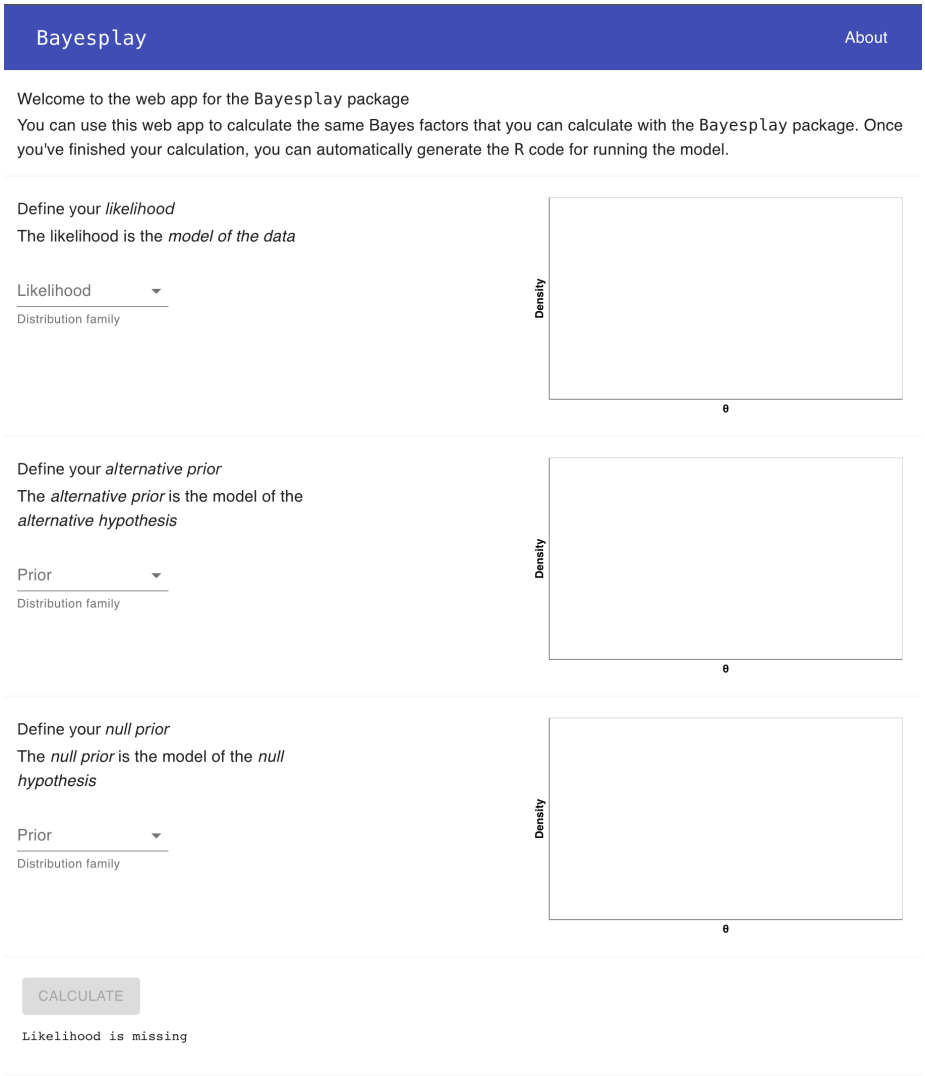
```
visual_compare(m1, m0) +
  theme_minimal(14)
```



4.2 Computing Bayes factors with Bayesplay-Web

If you're not super proficient with `R`, then you can use the **Bayesplay-Web** app to compute Bayes factors. The **Bayesplay-Web** will even generate the `R` code for you. To access the web-app go to bayesplay.mindsci.net.

Below is an image of the **Bayesplay-Web** interface.



Define your *likelihood*

The likelihood is the *model of the data*

Likelihood

binomial

Distribution family

You can use the binomial likelihood when your observation is the number of successes in a number of trials

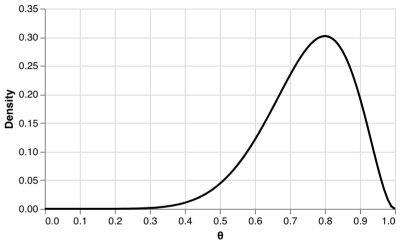
Parameters

successes

8

trials

10



Then we define the prior for unfair coin.

Define your *alternative prior*

The *alternative prior* is the model of the *alternative hypothesis*

Prior

beta

Distribution family

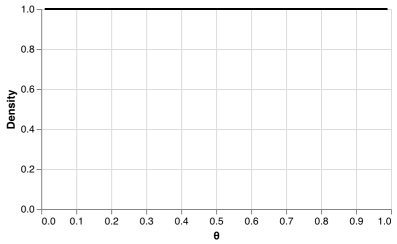
Parameters

alpha

1

beta

1



And then the prior for the fair coin.

Define your *null prior*

The *null prior* is the model of the *null hypothesis*

Prior

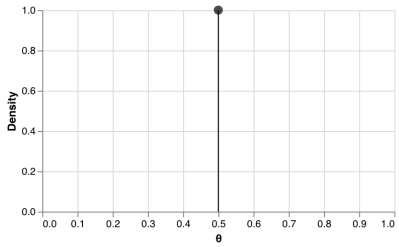
point

Distribution family

Parameters

point

0.5



And then we click **Calculate** to get our answer.

CALCULATE

BF_{10} = 2.0686869

BF_{01} = 0.4833984

View advanced output

Show R code

The BF10 value shows the evidence for the alternative model over the null

model. The BF01 value is the inverse, and shows the evidence for the null model over the alternative model"

Toggling the **Show R Code** button will show you the `R` code you need to compute the model.

And finally, clicking on **View advanced output** will take you to a new screen where you can view some additional output including the model predictions.

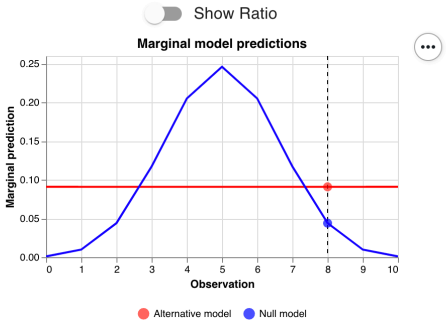


Figure 4: The marginal model predictions

To reproduce this analysis you can bookmark this page or you can view the corresponding R code below.

Show R code

4.3 Moving beyond coin flips

In the next section, we'll learn how to calculate Bayes factors with different kinds of likelihoods for different kinds of data that we might encounter. We'll also go in to more detail about different kinds of priors we might want to use and why we might want to use them.

In the meantime, you can play around with the **web-app** and have a look through the documentation for the `R` package (available at bayesplay.github.io/bayesplay/). You'll have to submit `R` code for the assessment (either written yourself or generated with the web-app) so it pays to familiarise yourself with it.

```
## Warning: package 'magrittr' was built under R version 4.1.2
```

Chapter 5

Moving beyond coin flips

In the previous section (see An alternative to p values, and The Bayes factor), we were introduced to the concept of the **likelihood**. In these sections, we specifically covered the **binomial** likelihood, which can be used for working out Bayes factors for samples of Bernoulli trials—that is, trials with two discrete outcomes like heads and tails or successes and failures. We used this specifically for computing our Bayes factors for hypotheses about coin flips. Although these Bayes factors could be used anywhere where we might ordinarily use a frequentist **binomial test** it is still rather limited in scope. Therefore, in this section, we'll cover Bayes factors that can be used in other situations. Specifically, we'll focus primarily on situations where we're interested in **differences between means**—that is, situations where we might otherwise use a t-test or ANOVA.

5.1 Choosing a likelihood

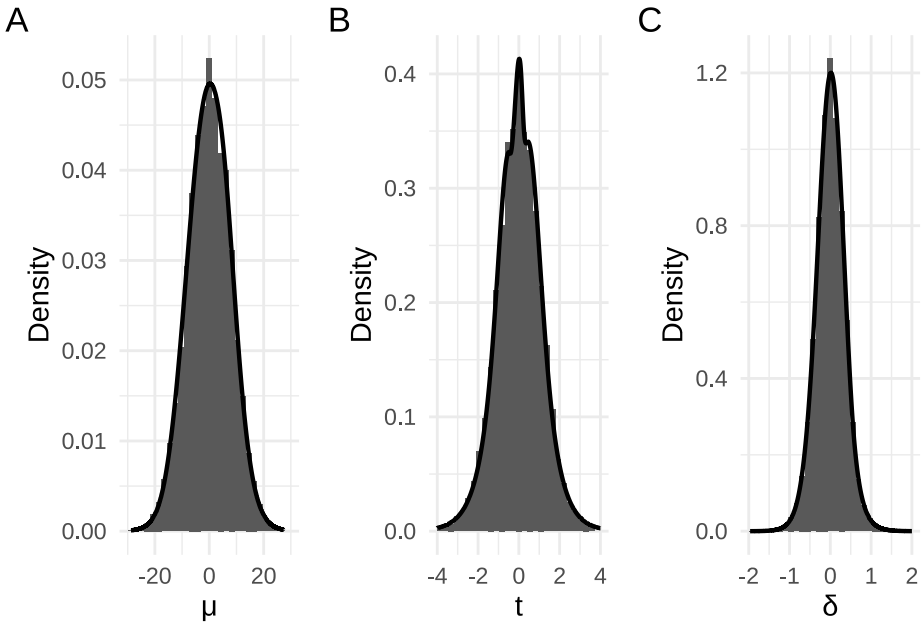
In our initial example on hunting treasure (see Null hypothesis significance testing) our treasure hunting device worked by, on average, pointing at 0 when there was no treasure around and, on average, pointing at some other value when there was treasure around.

To work out whether our device was, on average, pointing at a particular value we collected a sample of a fixed size (10 in our first example) and then worked out the average of these values. For example, we might've collected a sample of 10 values as follows:

Sample data: -14, -5.8, 39, 1.8, 3.2, 42.9, 11.5, -31.6, -17.2, -11.1

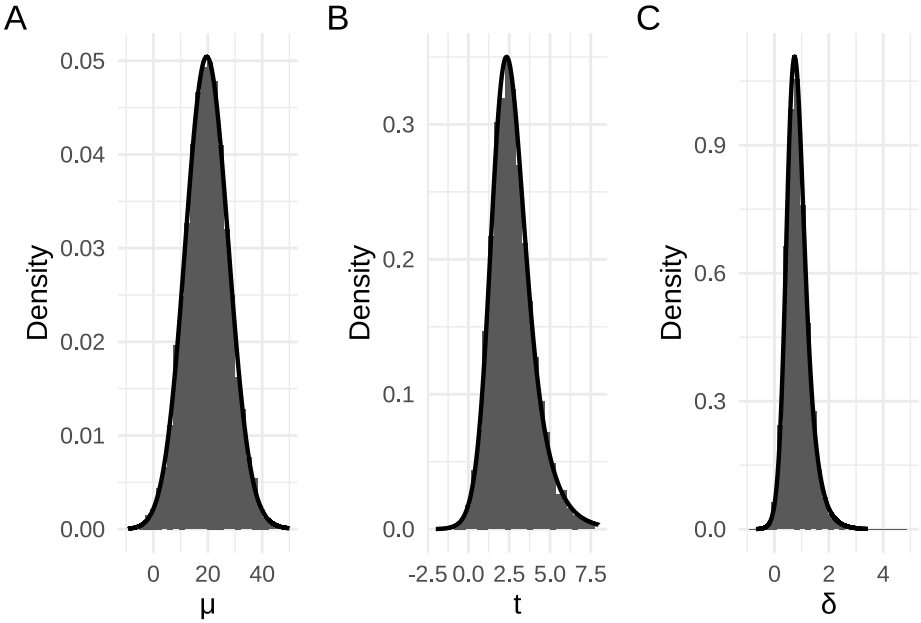
Sample mean: 1.87

After collecting a large number of samples, we could plot our averages as follows:



In the above examples, we've set the *parameter* of interest (the mean) to 0. The above plots are just examples of the sampling distribution when the parameter is 0. Panel **A** shows the sampling distribution of the *raw means*. Panel **B** shows the sampling distribution of the raw means re-scaled to t values. Finally, panel **C** shows the sampling distribution of the raw means re-scaled to *Cohen's* d values (where $\delta = \frac{\mu}{\sigma}$). All these plots approximately follow the shape of a *normal distribution* or a *t distribution*. We could use these plots to work out p values, just as we previously did.

But, as we did with the coin flip examples, when working with *likelihoods*, we're interested in the probability of obtaining our data under different values of the parameter. That is, we must consider the probability of obtaining our current data not just in the case where the parameter of interest is 0, but also where the parameter of interest is some other value. For example, the plots below have been generated by setting the parameter value–that is, the value to which the device, on average points—to a raw mean of approximately 19.6. We've also set the average spread of the values—that is, the standard deviation–to 25. Consequently, when rescaled to a t value, this would result in an average t of $\frac{19.6}{\frac{25}{\sqrt{10}}}$. And when rescaled to a d value, this would result in an average d value of $\frac{19.6}{25}$.



For these plots, panel **A** again shows the *raw means*, with panel **B** showing the t values, and panel **C** showing the Cohen's d values. The distribution in Panel **A** now approximately follows the shape of a *normal distribution* or a *scaled and shifted t distribution*. Panel **B** and **C** now follow the shape of slightly differently scaled versions of the *non-central t distribution*.

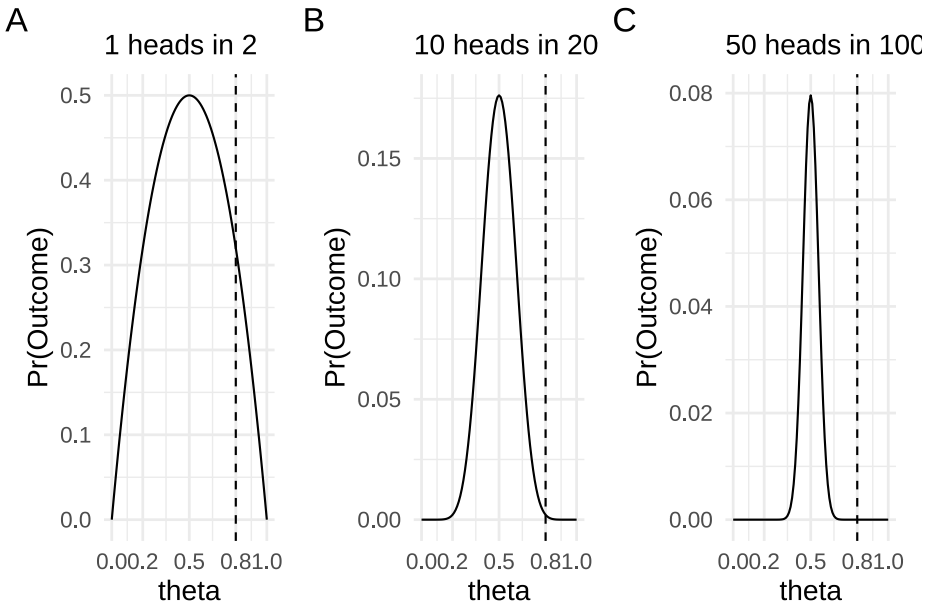
In the section that follows, we'll discover and learn how to use these likelihoods: The *normal* likelihood, the *scaled and shifted t* likelihood, and three versions of the *non-central t* likelihood (which `bayesplay` calls the *non-central t*, *non-central d*, and *non-central d2* likelihoods).

5.2 The variance of likelihoods

When we were examining coin flips, we saw that the sampling distribution (in the flip until **n** flips case) followed the *binomial distribution*. And when we wanted to make inferences about parameter values (the coin bias) we used the *binomial* likelihood. Our data, which we used to make our inference consisted of, first, the number of heads and, second, the number of flips.

These two values controlled different aspects of the shape of the likelihood function. First, the number of heads that we observed more-or-less controlled where the peak of the likelihood function was located. Second, the number of flips, or the sample size, controlled how spread out the likelihood function was. In the plots below we can see three cases of observing $\frac{n}{2}$ heads in 2, 20, and 100 flips. In all three plots, the likelihood function is peaked at 0.5 and drops off as we

move away from 0.5. The rate of this drop off, however, is steeper as the sample size increases.



What is the important intuition here? To get a handle on the intuition we can think of the extreme cases. When the coin is very biased—for example, it shows heads 0.8 of the time—then it will still sometimes show tails. It might even sometimes show tails on the first flip. In fact, it'll show tails on the first flip 0.2 of the time. Therefore, we wouldn't be that surprised if after two flips we have one head and one tail, because it won't be such an uncommon occurrence. But in the situation where we're making 100 flips, it now becomes more and more unlikely that we'd see equal numbers of heads and tails if the coin bias really was 0.8. We can put numbers to it by calculating the likelihood ratio between 0.5 and 0.8 for each of the three sample sizes.

When there are 2 trials, the likelihood ratio between $= 0.5$ and $= 0.8$ is 1.56.

When there are 20 trials, the likelihood ratio between $= 0.5$ and $= 0.8$ is 86.74.

When there are 100 trials, the likelihood ratio between $= 0.5$ and $= 0.8$ is 4909093465.3.

For all the likelihoods that we'll examine in this section, when defining the likelihoods we'll have one value that represents our observation: the mean we observe, the t value we observe, or the d value we observe. This will be analogous to the number of heads we observe in the coin flip example. And we'll have a value (or values) that defines how peaked or spread out the likelihood will be: this could be the sample size, the degrees of freedom and/or standard deviation.

In more technical terms, all the likelihoods that we'll examine will have one

parameter that we're making inferences about: the *mean*, the t value, or the d value. But this parameter will also have a *variance* associated with it. And as we'll see in the examples below, depending on whether we're interested in *raw means*,t values, or d values, the sampling distributions are slightly different shapes. Therefore, the *likelihoods* that we use in each case will be slightly different. We'll look at each of these in turn.

5.3 Inferences about raw means

When we're interested in making inferences about the *raw means* we have two choices available to us. The most straightforward choice is to choose the *normal* likelihood. The shape of the *normal* likelihood is controlled by **two** values. The first value is our **observed mean**. This value controls the location of the peak of the likelihood function. The second value is the **standard deviation of the mean**. The **standard deviation of the mean** is more commonly known as the **standard error of the mean**. We can work out the *standard error* using the following formula:

$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}},$$

where σ is the standard deviation of the population. Usually, we don't know the value of σ, so we estimate it using s, or the standard deviation of our sample.

To see how defining a *normal* likelihood works in practice, we'll generate our data. From this, we'll work out the mean of our sample, and we'll estimate the standard error (the standard deviation of the mean).

To generate our data, we'll set up a data generating process (you can think of this as **the population**) and we'll draw a sample of 10 values from this. Our data generating process will have a μ (mean of the population) of 19.6, and a σ (standard deviation of the population) of 25.

Sample data: 5.59, 13.85, 58.57, 21.36, 22.83, 62.48, 31.12, -12.03, 2.43, 8.46

Mean of sample: 21.47

Standard deviation of sample: 23.84

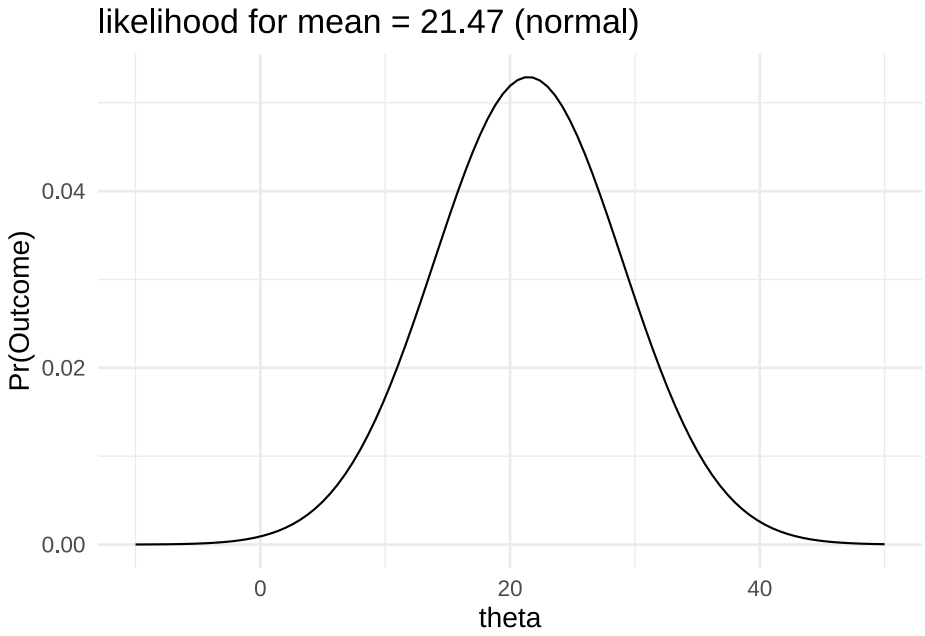
Size of sample: 10

Standard deviation of sampling distribution (standard error): 7.54

Note that the two values that we want are the *mean of the sample* and the *standard deviation of the mean*. A common confusion is that you want the *standard deviation of the sample*. This is not the value that we want, and we only calculate it because we can use it to estimate the *standard deviation of the population* and, from this, the *standard deviation of the mean*.

Now we can define the likelihood, and we can plot it.

```
data_model <- likelihood(family = "normal", mean = 21.47, sd = 7.54)

plot(data_model) +
  labs(title = "likelihood for mean = 21.47 (normal)") +
  xlim(-10, 50) +
  theme_minimal(14)
```



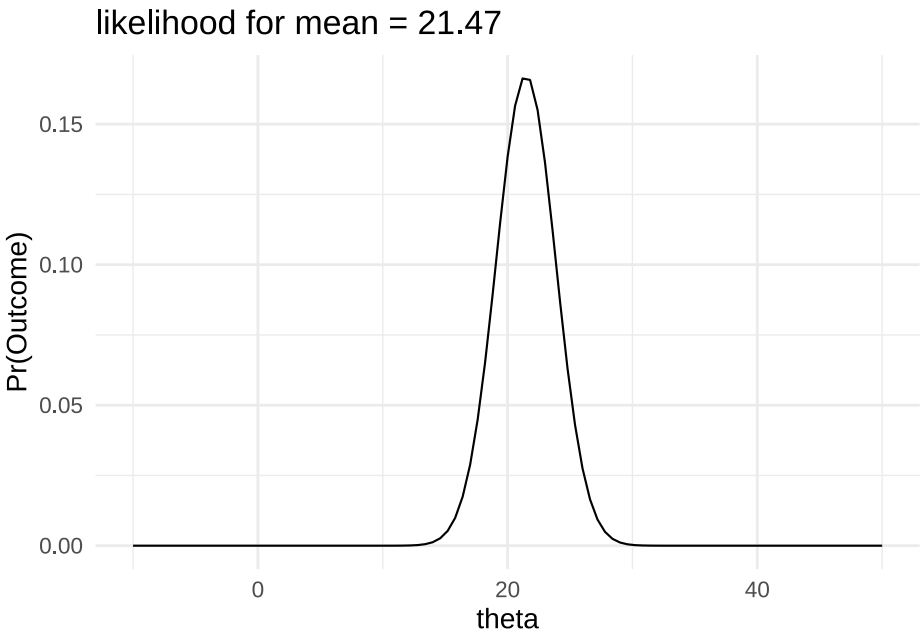
From our likelihood plot we can see that our data would be generated more often if the mean of the data generating process was 21 and less often if the mean of the data generating process was 30.

In fact, we can put a number to it and say that:

The data would be produced 1.89 times more often if the mean of the population was 21 than it would be if the mean of the population was 30.

The likelihood function will get wider or narrower when the *standard deviation of the mean* changes. The two factors that control the *standard deviation of the mean* are the sample size and the standard deviation of the population. In the example below we'll keep our estimate of the *standard deviation of the population* the same but we'll increase the sample size. Consequently, the *standard deviation of the mean* will decrease and our likelihood function will become narrower.

```
data_model <- likelihood(family = "normal", mean = 21.47, sd = 2.384)

plot(data_model) +
  labs(title = "likelihood for mean = 21.47") +
  xlim(-10, 50) +
  theme_minimal(14)
```



And this would also make a change to any likelihood ratio we could calculate. For example, calculating the new likelihood ratio comparing $\mu = 30$ and $\mu = 21$ would give the following result:

The data would be produced 590.92 times more often if the mean of the population was 21 than it would be if the mean of the population was 30.

In the preceding examples, we were modelling our data with a *normal* likelihood. And we were estimating the *standard deviation of the mean* using the *standard deviation of our sample*. We did this, because we didn't know the actual standard deviation of our data generating process. We'd need to know this value if we wanted to exactly calculate the *standard deviation of the mean*.

Or at least, we ordinarily don't know the standard deviation of our data generating process. However, because I set it up, I know it is 25, because this is the value I set it to. Therefore, our estimate was an under estimate. Typically, we'll underestimate the *standard deviation of the population*. These under-estimates will be worse when the sample size is small. As our sample size increases then the two values will, by definition, match.

There are a few approaches that we can take to dealing with this issue. First, we can just do nothing. This is the most straightforward approach, and it is also a very common approach. The second approach, is to apply a correction as follows:

$$s_{\bar{x}} = \frac{s}{\sqrt{n}} \times \left(1 + \frac{20}{\text{df}^2}\right),$$

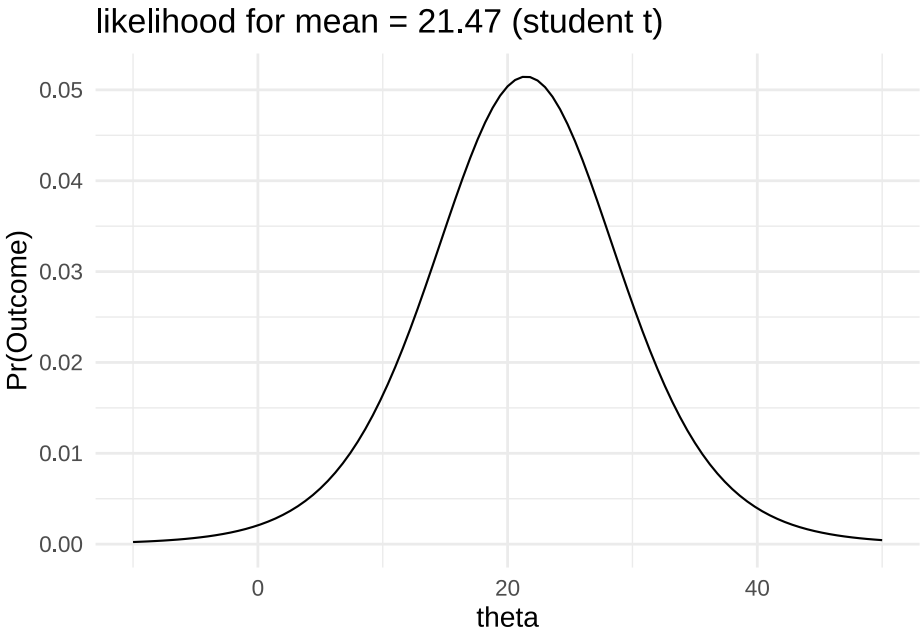
where df are the degrees of freedom for the corresponding t test. Dienes (2014, p 11) provides more details on this approach. Dienes (2014) also provides good guidance on using Bayes factors, and I would recommend reading it for the assessment.

Finally, the third approach, is to employ a *scaled and shifted t likelihood* instead of a *normal likelihood*. The *scaled and shifted t likelihood* has fatter tails than the *normal likelihood*, which accounts for the fact that our *normal likelihood* tends to be narrower than it ought to be.

In `bayesplay` we can use the *scaled and shifted t likelihood* by setting the likelihood **family** to **student_t*. When using this likelihood, one additional value will need to be set. This is the **df** value, which is the same as the **df** value that we would use for the correction approach above.

We'll recompute our previous example using this likelihood family.

```
data_model <- likelihood(family = "student_t", mean = 21.47, sd = 7.54, df = 9)

plot(data_model) +
  labs(title = "likelihood for mean = 21.47 (student t)") +
  xlim(-10, 50) +
  theme_minimal(14)
```



As you can see the two likelihood functions look very similar.

The data would be produced 1.94 times more often if the mean of the population was 21 than it would be if the mean of the population was 30.

This likelihood ratios are also very similar.

The difference in the likelihood ratio between the *normal* likelihood and the *student_t* likelihood is about 0.05

5.4 Inferences about effect sizes

A very popular alternative approach to modelling data in terms of the observed mean is to instead model data in terms of standardized effect sizes on the raw means themselves. This gets around the problem of the unknown variance; however, it has an added benefit in that it can place results from very different experiments on a common scale. For example, results from a study of reaction times might have values between 500 and 2000 ms and results from a study on test scores might have values that range between 40 and 90. By re-scaling mean values to *standardised means*—that is, to effect sizes—we can be more certain that values will fall somewhere between -10 / 10, and more typically between -1/1. When we talk more about priors, we'll see that this rescaling will help us to come up with priors that will work in a wide range of settings and with a wide range of experiments and types of data. However, we'll also see that when we want to come up with priors for specific situations, thinking in terms of **standardised effects** rather than actual differences in data can get confusing.

To define a likelihood based on effect size, we first need to work out the effect size. There are two formulas for effect sizes, depending on whether we have data from one group (or from paired samples) or whether we have data from two groups.

If we have data from paired samples, then we first work out the pair-wise differences. Following this, we proceed as we would for the one-sample case. To work out the effect size for the one sample case, we use the following formula:

$$d = \frac{m}{s},$$

Where m is the mean of the sample, and s is the standard deviation of the sample.

If we have data from two groups then the formula is a little more complex. For the two group case, the formula is as follows:

$$d = \frac{m_1 - m2}{s_{\text{pooled}}},$$

where s_{pooled} is given as follows:

$$s_{\text{pooled}} = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}},$$

and where m_1/m_2, s_1/s_2, and n_1/n_2, are the mean, standard deviation, and sample size for group 1 and group 2. In the worked examples below we'll see that we can use the `effsize` package in R to work out this effect size.

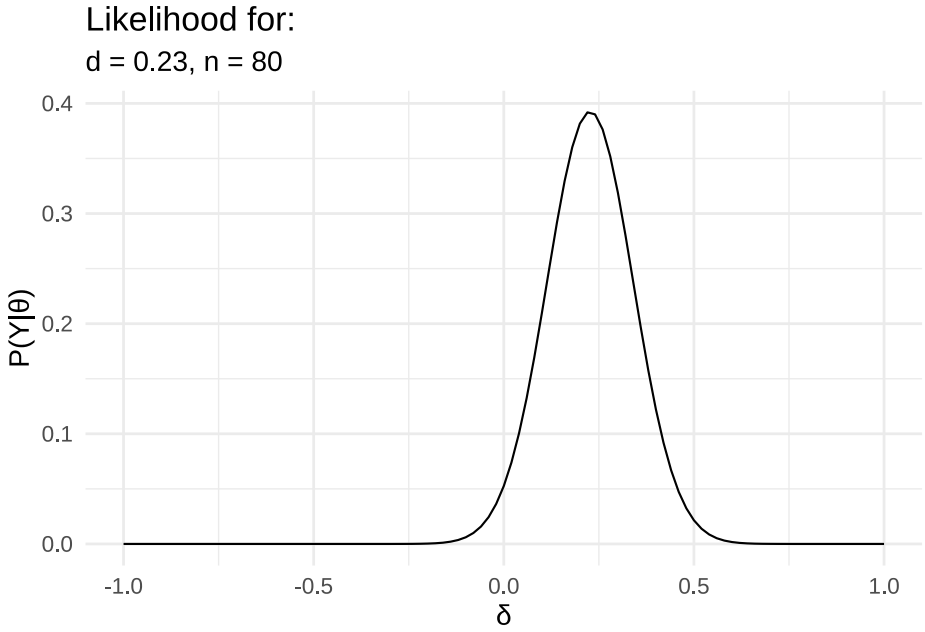
Once we have the effect size, then the only other value that we need is the sample size. We have one sample size in the one group case (sample size or number of pairs), and for the two group case we'll have the sample size of each group. We'll walk through a couple of examples using simulated data.

In the following example, we have data from an experiment looking at memory for words. The words were presented under two conditions: an *emotional* condition and a *neutral* condition. We're interested in knowing whether there is a difference in recognition accuracy between the two conditions. This is the kind of data we'd ordinarily analyse with a t test.

We'll load the data, work out the effect size, define the likelihood, and then plot it.

```
# First load the data
word_data <- readr::read_csv("https://files.mindsci.net/word_data.csv",
  show_col_types = FALSE
```

```
)

# Now we'll work out the effect size and n
summary_data <- word_data %>%
  pivot_wider(1:3, names_from = "condition", values_from = "accuracy") %>%
  mutate(diff = emotional - neutral) %>%
  summarise(m = mean(diff), s = sd(diff), n = n()) %>%
  mutate(d = m / s, t = m / (s / sqrt(n)))


# Now define the likelihood and plot it
data_model <- likelihood(
  family = "noncentral_d",
  d = summary_data$d,
  n = summary_data$n
)

plot(data_model) +
  labs(
    x = " ", y = "P(Y| )",
    title = "Likelihood for:",
    subtitle = glue::glue("d = {round(summary_data$d,2)}, n = {summary_data$n}")
  ) +
  theme_minimal(14) +
  scale_x_continuous(limits = c(-1, 1), breaks = seq(-1, 1, .5))
```



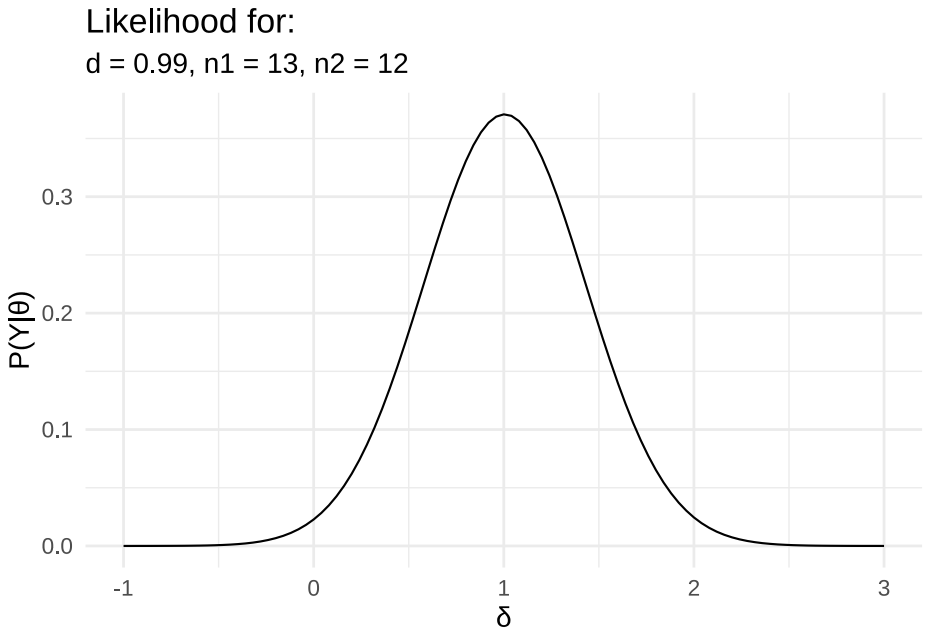
In the second example, we'll look at the two group case. For this example we'll use some simulated data to match some data from an experiment I conducted many years ago. In this task, people were asked to watch an animated avatar performing a movement, they were asked to synchronise a button press with critical points in the movement, and the timing error was measured. The animated avatar moved in two difference ways. In one condition it moved like a human. In the other condition, the dynamics of the movement were altered so that it moved like a robot. All participants viewed both kinds of movements. In addition to this within subjects factor, there was also a between subjects factor. Before viewing any of the movement, participants were split into two groups. One group was given experience actually performing the movement they would later observe, while the other group was not.

This is the kind of data that would ordinarily be analysed using a 2×2 mixed ANOVA. However, I was particularly interested in the **interaction**. The **interaction** just examines whether the difference between **condition 1** and **condition 2** is different between **group 1** and **group 2**. In the example below, I've already worked out the difference in the timing error for **condition 1** and **condition 2**, and now we just have to compare this difference between the two groups.

As with the earlier example, we'll load the data, work out the effect size, define the likelihood, and then plot it.

```
# First load the data
motor_exp <- readr::read_csv("https://files.mindsci.net/motor_exp.csv",
  show_col_types = FALSE
```

```
)


# Now we'll work out the effect size and n
summary_data <- motor_exp %>%
  dplyr::group_by(group) %>%
  dplyr::summarise(m = mean(r_diff), s = sd(r_diff), n = n()) %>%
  tidyr::pivot_wider(names_from = "group", values_from = c("m", "s", "n"))

md_diff <- summary_data$m_exper - summary_data$m_naive
sd_pooled <- sqrt((((summary_data$n_exper - 1) * summary_data$s_exper^2) +
  ((summary_data$n_naive - 1) * summary_data$s_naive^2)) /
  (summary_data$n_exper + summary_data$n_naive - 2))
d <- md_diff / sd_pooled

# or we can use the effsize package
# you'll just need to install it before you use it
# you can install it with the following command
# install.package("effsize")
#
# and then use it as follows
# d <- effsize::cohen.d(motor_exp$r_diff, motor_exp$group)$estimate

# sample_size <- motor_exp %>%
#   dplyr::group_by(group) %>%
#   dplyr::summarise(n = n())

# Now define the likelihood and plot it
data_model <- likelihood(
  family = "noncentral_d2",
  d = d,
  n1 = summary_data$n_exper,
  n2 = summary_data$n_naive
)

plot(data_model) +
  labs(
    x = " ", y = "P(Y| )",
    title = "Likelihood for:",
    subtitle = glue::glue("d = {round(d,2)}, n1 = {summary_data$n_exper}, n2 = {summary_data$n_na
  ) +
  theme_minimal(14) +
  scale_x_continuous(limits = c(-1, 3), breaks = seq(-1, 3, 1))
```

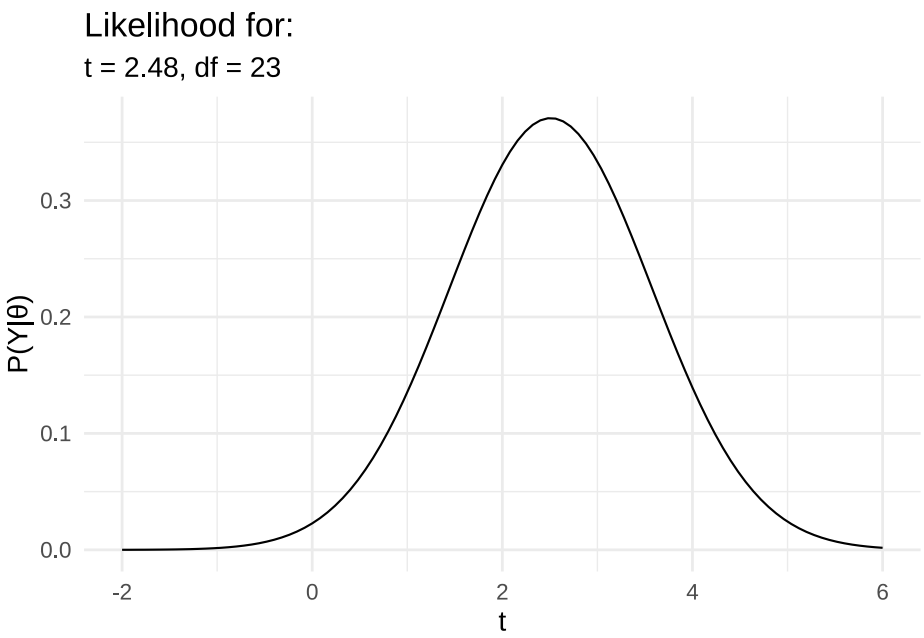


5.5 Inferences about t values

Finally, we'll repeat the last analysis, but in this case we'll model the data in terms of t rather than d.

```
# Load the data again just in case
motor_exp <- readr::read_csv("https://files.mindsci.net/motor_exp.csv",
  show_col_types = FALSE
)

# Run the t test and extract the t value and df
t_test_res <- t.test(r_diff ~ group, motor_exp, var.equal = TRUE) %>%
  broom::tidy() %>%
  dplyr::select(statistic, parameter)

# Now define the likelihood

data_model <- likelihood(
  family = "noncentral_t",
  t = t_test_res$statistic,
  df = t_test_res$parameter
)
```

```
# And plot it
plot(data_model) +
  labs(
    x = "t", y = "P(Y| )",
    title = "Likelihood for:",
    subtitle = glue::glue("t = {round(t_test_res$statistic,2)}, df = {t_test_res$parameter}")
  ) +
  theme_minimal(14) +
  scale_x_continuous(limits = c(-2, 6), breaks = seq(-2, 6, 2))
```



Using the *non-central t* likelihood might seem a little easier to use, because it requires less work upfront because we can just use the `t.test` function to work out the t statistic instead of having to work out the d value. However, as we'll see in the section on **priors**, there as advantages to using the *non-central t* likelihood. This disadvantage is primarily to do with the fact that the t value can change dramatically with sample size—that is, very large sample sizes can result in very large t values even if the mean difference between conditions or groups stays constant. For the *non-central d* and *non-central d2* likelihoods this isn't an issue. The d value will stay the same even if the sample size increases and instead, the likelihood will just get narrower.

I've covered the *non-central t* likelihood for completeness, but it's almost never of any real use.