Bayesian Linear Modeling: An introduction using brms and rstan

Lecture notes for MSc Cognitive Science and MSc Linguistics, University of Potsdam, Germany

Shravan Vasishth

2018-10-07

Contents

1 Foundational ideas

1.1 Introduction

 The materials are available from:

- OSF
- github

Acknowledgments: The linear modeling and hierarchical linear modeling chapters follows to some extent the presentation developed by Bruno Nicenboim in https://github.com/vasishth/FGME_Stan_2017. The course notes have benefitted a lot from the lectures and writings of Michael Betancourt.

1.1.1 Intended audience and prerequisites

This course is intended for **graduate students** in all relevant MSc progams related to Cognitive Science at the University of Potsdam, Germany. At the time of writing, this includes Cognitive Science and Linguistics.

The public course home page is here. The university-internal moodle forum for homework submissions and internal communications is here.

I assume here that the graduate student taking this course has elementary numeracy acquired usually at the class 10 level. Calculus and linear algebra are mentioned in the notes but I do not require the student to know these topics. Whenever calculus and linear algebra come up, I will explain what the equations or formulas mean just in time. No **active** ability in these areas is needed. What I do assume is basic (class 10) algebra, basic set theory, and arithmetic ability.

Some very basic knowledge of R is assumed, but not much. If students don't know R, I will provide a quick introduction, although google is one's friend here.

1.1.2 Software needed

Before starting, please install

- R and RStudio
- The R package rstan:
  - Instructions for Windows
  - Instructions for Mac or Linux

- The R package brms

Please talk to me if you have difficulties installing anything, although be warned that my knowledge of Windows is limited to knowing that it exists in the world out there somewhere.

1.1.3 Motivation for course design

Because statistics inherently depends on mathematics, it is very important to get a reasonably formal introduction to the topic. The usual, informal manner in which statistics is taught has been a failure in all Cognitive Science disciplines. We see this in the way statistics is routinely abused in psycholinguistics, psychology, and linguistics, among other areas. Some examples from psycholinguistics are mentioned alongside each point:

- deciding that an effect is "reliable" if $p < 0.05$ in a severely underpowered study; for examples, see Jäger, Engelmann, and Vasishth (2017), Vasishth et al. (2018).
- inconrrectly arguing that the null is true when $p > 0.05$, or arguing that the null is false when is a bit over $p > 0.05$, but the researcher wants desperately that p be less than 0.05; for discussion, see Vasishth and Nicenboim (2016), Nicenboim and Vasishth (2016).
- flexible data analysis to get the result desired: analyze many dependent measures and choose whichever result you like, ignoring the rest; see Malsburg and Angele (2017).
- ignore model assumptions, focusing only on p-value; examples are discussed in Vasishth et al. (2013), Nicenboim, Roettger, and Vasishth (2018), Vasishth et al. (2017).
- flexibly change the research question, theory, or predictions after seeing the data; for discussion of the distinction between confirmatory vs exploratory testing, see Nicenboim et al. (2018).
- remove data selectively, or avoid removing it, depending on the desired result (examples will be provided).

Most people do not make these mistakes intentionally; I have made them too. For examples of mistakes listed above from my own work, see Vasishth and Lewis (2006), Vasishth (2003).

In a modest attempt to at least partly remedy this situation, the present course attempts to provide some understanding of the basic formal ideas behind statistical modeling.

If you find mistakes, please open an issue here.

The central idea we will explore in this course is: how to use Bayes' theorem for data analysis. In order to understand the methodology, some passive understanding of the following topics needs to be in place:

- Basic probability theory:
  - sum and product rule
  - conditional probability
  - independence of events
  - Bayes' rule
- The theory of random variables
  - probability density/mass function $f(x)$ vs cumulative distribution function $F(x)$
  - inverse CDF, $F^{-1}(x)$
  - expectation and variance of (transformations of) random variables
- Probability density/mass functions
  - Ten common pdfs
  - Jointly distributed random variables
  - Sums of random variables
  - Marginal and conditional pdfs
  - Covariance and correlation
  - Multivariate normal distributions
- Maximum likelihood estimation
  - How to find MLEs analytically
  - How to find MLEs using R
  - Visualization of log likelihood

Without a clear understanding of these concepts, generalized confusion is an inevitable outcome. This is why it is so important to expend some energy in understanding these concepts. One common question I get is: *why do I need to know this*? The answer is: Please trust me that you do.

But before we dive in, it may help to step back and get an overview of where we are going in this course.

1.1.4 Preview: Steps in Bayesian analysis

The way we will conduct data analysis is as follows.

- Given data, specify a *likelihood function*.
- Specify *prior distributions* for model parameters.
- Evaluate whether model makes sense, using fake-data simulation, *prior predictive* and *posterior predictive* checks, and (if you want to claim a discovery) calibrating true and false discovery rates.
- Using software, derive *marginal posterior distributions* for parameters given likelihood

function and prior density. I.e., simulate parameters to get *samples from posterior distributions* of parameters using some *Markov Chain Monte Carlo (MCMC) sampling algorithm*.

- Check that the model converged using *model convergence* diagnostics,
- Summarize *posterior distributions* of parameter samples and make your scientific decision.

The above is what you will learn in this course; all the terms introduced above will be explained in these notes and in class. We begin with basic probability theory.

1.2 Brief review of probability theory

The best reference textbooks I know for s first contact with probability theory are Kerns (2018) and Blitzstein and Hwang (2014). My notes below are a bit terse because I am assuming you will look at these if something is unclear.

1.2.1 Axioms of Probability

We assume here a basic knowledge of set theory. Let S be a set of events. For example, for a single coin toss, $S = \{A_1, A_2\}$, where A_1 is the event that we get a heads, and A_2 the event that we get a tails.

1. **Axiom 1** $\mathbb{P}(A) \geq 0$ for any event $A \subset S$.
2. **Axiom 2** $\mathbb{P}(S) = 1$.
3. **Axiom 3** If the events $A_1, A_2, A_3, \ldots$ are disjoint then

$$\mathbb{P}\left(\bigcup_{i=1}^{n} A_i\right) = \sum_{i=1}^{n} \mathbb{P}(A_i) \text{ for every } n, \tag{1}$$

   and furthermore,

$$\mathbb{P}\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mathbb{P}(A_i). \tag{2}$$

Three important propositions:

Proposition 1

Let $E \cup E^c = S$. Then,

$$1 = P(S) = P(E \cup E^c) = P(E) + P(E^c) \tag{3}$$

or:

$$P(E^c) = 1 - P(E) \tag{4}$$

Proposition 2

If $E \subset F$ then $P(E) \leq P(F)$.

Proposition 3

$$P(E \cup F) = P(E) + P(F) - P(EF) \tag{5}$$

1.2.2 Conditional Probability

This is a central concept in this course. The conditional probability of event B given event A, denoted $\mathbb{P}(B \mid A)$, is defined by

$$\mathbb{P}(B \mid A) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(A)}, \quad \text{if } \mathbb{P}(A) > 0. \tag{6}$$

Theorem

For any fixed event A with $\mathbb{P}(A) > 0$,

1. $\mathbb{P}(B|A) \geq 0$, for all events $B \subset S$,
2. $\mathbb{P}(S|A) = 1$, and
3. If B_1, B_2, B_3,... are disjoint events,

then:

$$\mathbb{P}\left(\bigcup_{k=1}^{\infty} B_k \middle| A\right) = \sum_{k=1}^{\infty} \mathbb{P}(B_k|A). \tag{7}$$

In other words, $\mathbb{P}(\cdot|A)$ is a legitimate probability function. With this fact in mind, the following properties are immediate:

For any events A, B, and C with $\mathbb{P}(A) > 0$,

1. $\mathbb{P}(B^c|A) = 1 - \mathbb{P}(B|A)$.
2. If $B \subset C$ then $\mathbb{P}(B|A) \leq \mathbb{P}(C|A)$.
3. $\mathbb{P}[(B \cup C)|A] = \mathbb{P}(B|A) + \mathbb{P}(C|A) - \mathbb{P}[(B \cap C|A)]$.
4. The Multiplication Rule. For any two events A and B,

$$\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B|A). \tag{8}$$

   And more generally, for events A_1, A_2, A_3,..., A_n,

$$\mathbb{P}(A_1 \cap A_2 \cap \cdots \cap A_n) = \mathbb{P}(A_1)\mathbb{P}(A_2|A_1)\cdots\mathbb{P}(A_n|A_1 \cap A_2 \cap \cdots \cap A_{n-1}). \tag{9}$$

1.2.3 Independence of events

(Taken nearly verbatim from Kerns 2018.)

Definition

Events A and B are said to be independent if

$$\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B). \tag{10}$$

Otherwise, the events are said to be dependent.

From the above definition of conditional probability, we know that when $\mathbb{P}(B) > 0$ we may write

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}. \tag{11}$$

In the case that A and B are independent, the numerator of the fraction factors so that $\mathbb{P}(B)$ cancels, with the result:

$$\mathbb{P}(A|B) = \mathbb{P}(A) \text{ when } A, B \text{ are independent}. \tag{12}$$

Proposition 4

If E and F are independent events, then so are E and F^c, E^c and F, and E^c and F^c.

Proof:

Assume E and F are independent. Since $E = EF \cup EF^c$ and EF and EF^c are mutually exclusive,

$$\begin{aligned} P(E) =& P(EF) + P(EF^c) \\ =& P(E)P(F) + P(EF^c) \end{aligned} \tag{13}$$

Equivalently:

$$\begin{aligned} P(EF^c) =& P(E)[1 - P(F)] \\ =& P(E)P(F^c) \end{aligned} \tag{14}$$

1.2.4 Bayes' rule

(Quoted nearly verbatim from Kerns 2018.)

Theorem Bayes' Rule. Let B_1, B_2, $\ldots$, B_n be mutually exclusive and exhaustive and let A be an event with $\mathbb{P}(A) > 0$. Then

$$\mathbb{P}(B_k|A) = \frac{\mathbb{P}(B_k)\mathbb{P}(A|B_k)}{\sum_{i=1}^{n} \mathbb{P}(B_i)\mathbb{P}(A|B_i)}, \quad k = 1, 2, \ldots, n. \tag{15}$$

The proof follows from looking at $\mathbb{P}(B_k \cap A)$ in two different ways. For simplicity, suppose that $P(B_k) > 0$ for all k. Then

$$\mathbb{P}(A)\mathbb{P}(B_k|A) = \mathbb{P}(B_k \cap A) = \mathbb{P}(B_k)\mathbb{P}(A|B_k). \tag{16}$$

Since $\mathbb{P}(A) > 0$ we may divide through to obtain

$$\mathbb{P}(B_k|A) = \frac{\mathbb{P}(B_k)\mathbb{P}(A|B_k)}{\mathbb{P}(A)}. \tag{17}$$

Now remembering that $\{B_k\}$ is a partition (i.e., mutually exclusive and exhaustive), the denominator of the last expression is

$$\mathbb{P}(A) = \sum_{k=1}^{n} \mathbb{P}(B_k \cap A) = \sum_{k=1}^{n} \mathbb{P}(B_k)\mathbb{P}(A|B_k). \tag{18}$$

1.3 Random variable theory

A random variable X is a function $X : S \rightarrow \mathbb{R}$ that associates to each outcome $\omega \in S$ exactly one number $X(\omega) = x$.

S_X is all the x's (all the possible values of X, the support of X). I.e., $x \in S_X$. We can also sloppily write $X \in S_X$.

Good example: number of coin tosses till H

- $X : \omega \rightarrow x$
- ω: H, TH, TTH,…(infinite)
- $x = 0, 1, 2, \dots ; x \in S_X$

Every discrete (continuous) random variable X has associated with it a **probability mass (distribution) function (pmf, pdf)**. I.e., PMF is used for discrete distributions and PDF for continuous. (I will sometimes use lower case for pdf and sometimes upper case. Some books use pdf for both discrete and continuous distributions.)

$$p_X : S_X \rightarrow [0, 1] \tag{19}$$

defined by

$$p_X(x) = P(X(\omega) = x), x \in S_X \tag{20}$$

[**Note**: Books sometimes abuse notation by overloading the meaning of X. They usually have: $p_X(x) = P(X = x), x \in S_X$]

Probability density functions (continuous case) or probability mass functions (discrete case) are functions that assign probabilities or relative frequencies to all events in a sample space.

The expression

$$X \sim g(\cdot) \tag{21}$$

means that the random variable X has pdf/pmf $g(\cdot)$. For example, if we say that $X \sim N(\mu, \sigma^2)$, we are assuming that the pdf is

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp[-\frac{(x-\mu)^2}{2\sigma^2}] \tag{22}$$

We also need a **cumulative distribution function** or cdf because, in the continuous case, P(X=some point value) is zero and we therefore need a way to talk about P(X in a specific range). cdfs serve that purpose.

In the continuous case, the cdf or distribution function is defined as:

$$P(x < X) = F(x < X) = \int_{-\infty}^{X} f(x)\,dx \tag{23}$$

1.3.1 The normalization constant in pdfs

Almost any function can be a pdf as long as it sums to 1 over the sample space. Here is an example of a function that doesn't sum to 1:

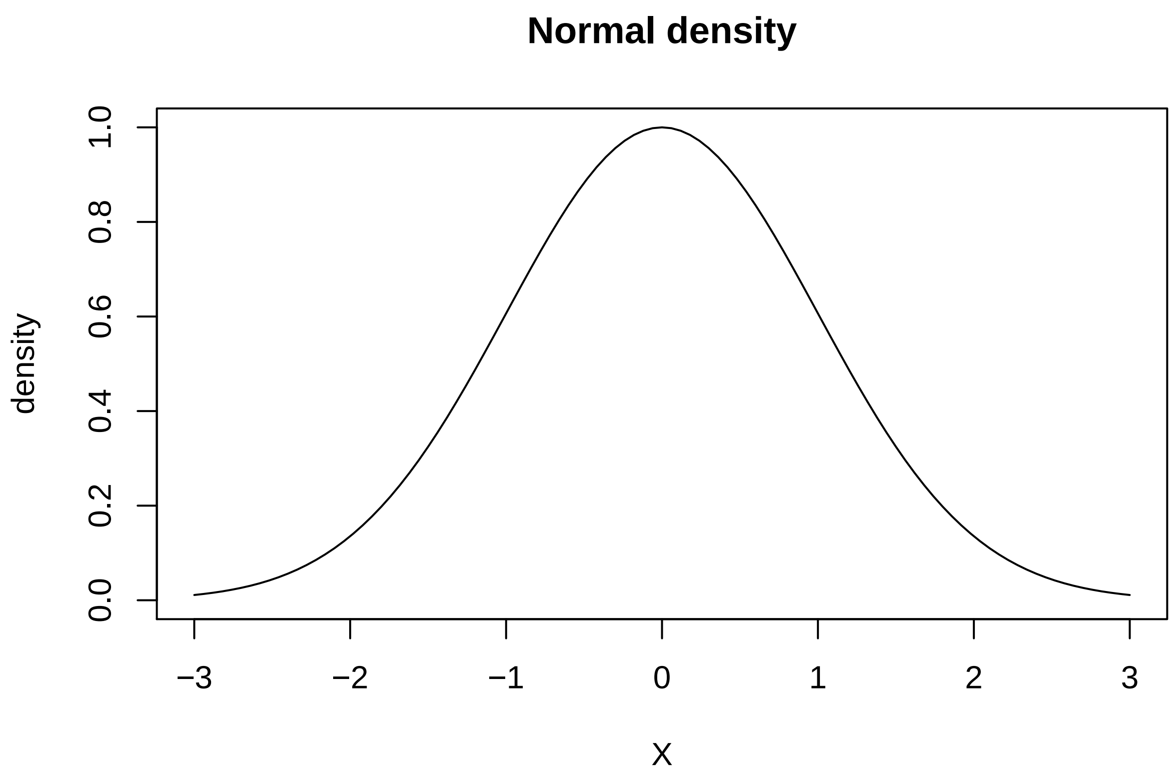
$$f(x) = \exp[-\frac{(x-\mu)^2}{2\sigma^2}] \tag{24}$$

This is the "kernel" of the normal pdf, and it doesn't sum to 1:

```
normkernel <- function(x, mu = 0, sigma = 1) {
    exp((-(x - mu)^2/(2 * (sigma^2))))
}

x <- seq(-10, 10, by = 0.01)

plot(function(x) normkernel(x), -3, 3, main = "Normal density",
    ylim = c(0, 1), ylab = "density", xlab = "X")
```




```
### area under the curve is less than 1:
integrate(normkernel, lower = -Inf, upper = Inf)
```

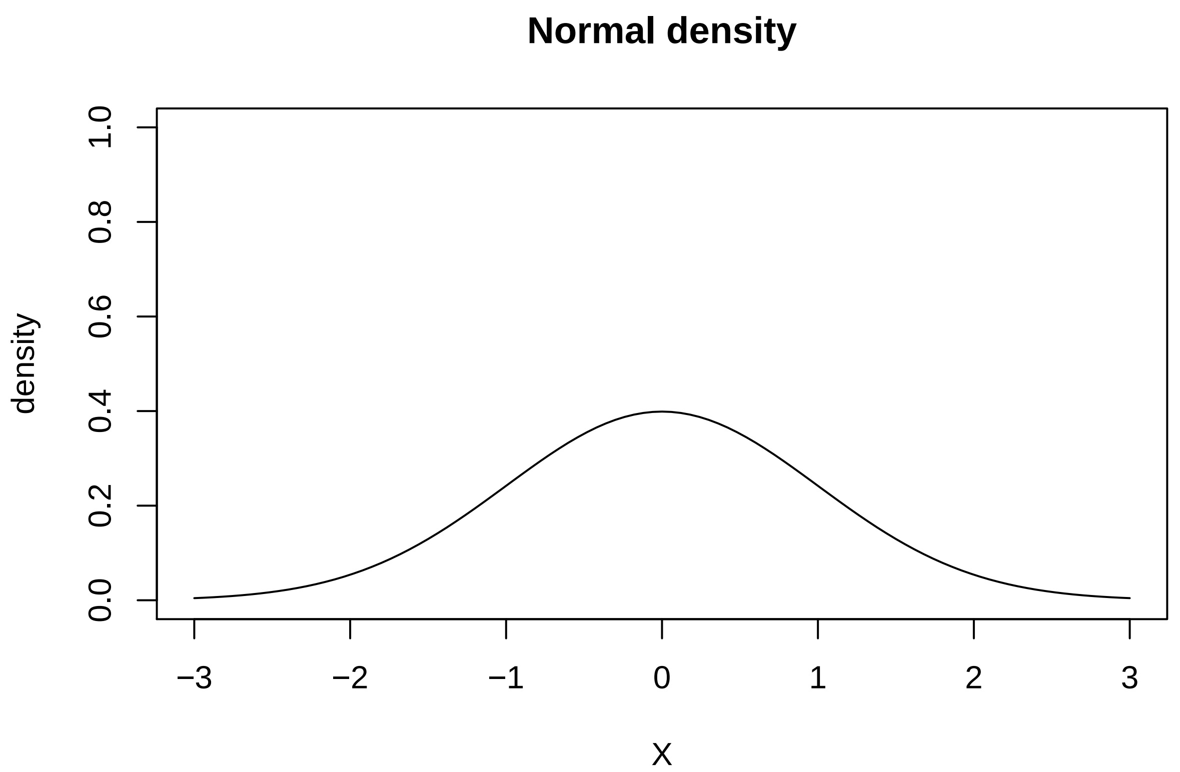
```
## 2.51 with absolute error < 0.00023
```

Adding a normalizing constant makes the above kernel density a pdf.

```
norm <- function(x, mu = 0, sigma = 1) {
    (1/sqrt(2 * pi * (sigma^2))) * exp((-(x - mu)^2/(2 * (sigma^2))))
}

x <- seq(-10, 10, by = 0.01)

plot(function(x) norm(x), -3, 3, main = "Normal density", ylim = c(0,
    1), ylab = "density", xlab = "X")
```




```
### area under the curve sums to 1:
integrate(norm, lower = -Inf, upper = Inf)
```

```
## 1 with absolute error < 0.000094
```

1.3.2 The pdf f(x) and the cdf F(x)

Recall that a random variable X is a function $X : S \to \mathbb{R}$ that associates to each outcome $\omega \in S$ exactly one number $X(\omega) = x$. S_X is all the x's (all the possible values of X, the support of X). I.e., $x \in S_X$.

X is a continuous random variable if there is a non-negative function f defined for all real $x \in (-\infty, \infty)$ having the property that for any set B of real numbers,

$$P\{X \in B\} = \int_B f(x)\,dx \tag{25}$$

Kerns has the following to add about the above:

> Continuous random variables have supports that look like
>
> $$S_X = [a,b] \text{ or } (a,b), \tag{26}$$

or unions of intervals of the above form. Examples of random variables that are often taken to be continuous are:

- the height or weight of an individual,
- other physical measurements such as the length or size of an object, and
- durations of time (usually).

E.g., in psychology and linguistics we take as continous:

1. reading time: Here the random variable X has possible values ω ranging from 0 ms to some upper bound b ms, and the RV X maps each possible value ω to the corresponding number (0 to 0 ms, 1 to 1 ms, etc.).
2. acceptability ratings (technically not true; but people generally treat ratings as continuous, at least in psycholinguistics)
3. Event related potentials

Every continuous random variable X has a probability density function (PDF) denoted f_X associated with it that satisfies three basic properties:

1. $f_X(x) > 0$ for $x \in S_X$,
2. $\int_{x \in S_X} f_X(x)\,\mathrm{d}x = 1$, and
3. $\mathbb{P}(X \in A) = \int_{x \in A} f_X(x)\,\mathrm{d}x$, for an event $A \subset S_X$.

We can say the following about continuous random variables:

- Usually, the set A in condition 3 above takes the form of an interval, for example, $A = [c,d]$, in which case

$$\mathbb{P}(X \in A) = \int_c^d f_X(x)\,\mathrm{d}x. \tag{27}$$

- It follows that the probability that X falls in a given interval is simply the area under the curve of f_X over the interval.
- Since the area of a line $x = c$ in the plane is zero, $\mathbb{P}(X = c) = 0$ for any value c. In other words, the chance that X equals a particular value c is zero, and this is true for any number c. Moreover, when $a < b$ all of the following probabilities are the same:

$$\mathbb{P}(a \leq X \leq b) = \mathbb{P}(a < X \leq b) = \mathbb{P}(a \leq X < b) = \mathbb{P}(a < X < b). \qquad (28)$$

$f(x)$ is the probability density function of the random variable X.

Since X must assume some value, f must satisfy

$$1 = P\{X \in (-\infty, \infty)\} = \int_{-\infty}^{\infty} f(x)\, dx \qquad (29)$$

If $B = [a, b]$, then

$$P\{a \leq X \leq b\} = \int_{a}^{b} f(x)\, dx \qquad (30)$$

If $a = b$, we get

$$P\{X = a\} = \int_{a}^{a} f(x)\, dx = 0 \qquad (31)$$

Hence, for any continuous random variable,

$$P\{X < a\} = P\{X \leq a\} = F(a) = \int_{-\infty}^{a} f(x)\, dx \qquad (32)$$

F is the **cumulative distribution function**. Differentiating both sides in the above equation:

$$\frac{dF(x)}{dx} = f(x) \qquad (33)$$

Just to reiterate this: the density (PDF) is the derivative of the CDF. You can go back and forth between the pdf and the CDF by integrating or differentiating:

$$\int_{a}^{b} f(x)\, dx \Rightarrow F(a) \qquad (34)$$

$$dF(x)/dx = f(x) \qquad (35)$$

$F(x)$ will give you some probability u. The **inverse of the cdf**, $F^{-1}(u)$ gives us back the quantile x such that $F(x) = u$. *This fact will be of great relevance to us.*

1.3.3 Some basic results concerning random variables

1.
$$E[X] = \int_{-\infty}^{\infty} x f(x)\, dx \tag{36}$$

2.
$$E[g(X)] = \int_{-\infty}^{\infty} g(x) f(x)\, dx \tag{37}$$

3.
$$E[aX + b] = aE[X] + b \tag{38}$$

4.
$$Var[X] = E[(X - \mu)^2] = E[X^2] - (E[X])^2 \tag{39}$$

5.
$$Var(aX + b) = a^2 Var(X) \tag{40}$$

So far, we have learnt what a random variable is, and we know that by definition it has a pdf and a cdf associated with it. Why did we go through all this effort to learn all this? The payoff becomes apparent next.

1.3.4 What you can do with a pdf

You can:

1. Calculate the mean:

   Discrete case:

$$E[X] = \sum_{i=1}^{n} x_i p(x_i) \tag{41}$$

   Continuous case:

$$E[X] = \int_{-\infty}^{\infty} x f(x)\, dx \tag{42}$$

2. Calculate the variance:

$$Var(X) = E[X^2] - (E[X])^2 \tag{43}$$

3. Compute quartiles: e.g., for some pdf f(x):

$$\int_{-\infty}^{Q} f(x)\,dx \tag{44}$$

For example, take $f(x)$ to be the normal distribution with mean 0 and sd 1. Suppose we want to know:

$$\int_{0}^{1} f(x)\,dx \tag{45}$$

We can do this in R as follows:[1]

```
pnorm(1)-pnorm(0)
```

Why are we going through these basic results? As Betancourt puts it (see here, at minute 9:55): "Bayesian computations reduce to (computing) expectations and, consequently, to integration." We will be computing expectations a lot in this course, using software. We will never have to do this by hand because we will be working in high-dimensional spaces and will have no choice but to use numerical approximations.

1.4 Ten important distributions

These distributions are generally used quite frequently in Bayesian data analyses, especially in psychology and linguistics applications. For the first few distributions, we show the pdf and cdf. The Binomial and Poisson are discrete distributions, the rest are continuous.

1.4.1 Binomial

Examples: coin tosses, question-response accuracy

Probability mass function:

If we have x successes in n trials, given a success probability p for each trial. If $x \sim Bin(n,p)$.

$$P(x \mid n,p) = \binom{n}{k} p^k (1-p)^{n-k} \tag{46}$$

[1]This is a very important piece of R code here. Make sure you understand the relationship between the integral and the R functions used here.

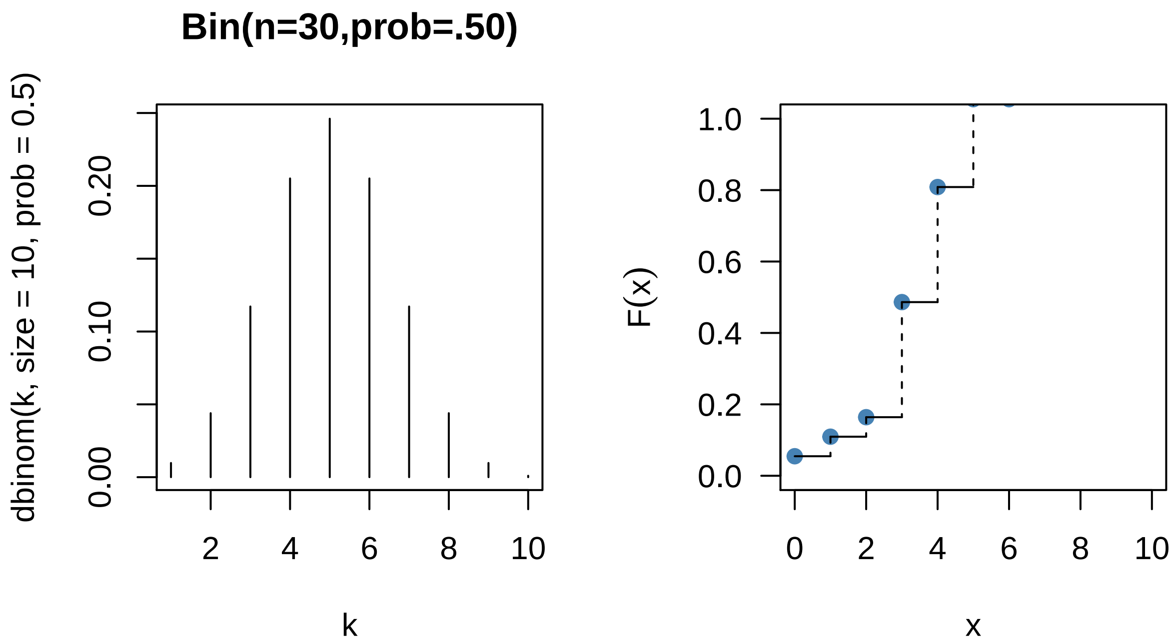


Figure 1: Example of the pmf Bin(n=10,prob=.50).

[Recall that: $\binom{n}{k} = \frac{n!}{(n-r)!r!}$. Hence, given x and n, this term will be a constant.]

Figure 1 shows the pmf and cdf.

Expectation and variance:

The mean is np and the variance $np(1-p)$.

When $n = 1$ we have the Bernoulli distribution.

Relevant functions in R

```
###pmf:
dbinom(x, size, prob, log = FALSE)
### cdf:
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)
### inverse cdf:
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)
### pseudo-random generation of samples:
rbinom(n, size, prob)
```

Notational conventions: A binomial distribution, n trials each with probability θ of occurring, is written $Bin(\theta, n)$. Given a random variable with this distribution, we can write $R \mid \theta, n \sim Bin(\theta, n)$ or $p(r \mid \theta, n) = Bin(\theta, n)$, where r is the realization of R. We can drop the conditioning in $R \mid \theta, n$, so that we can write: given $R \sim Bin(\theta, n)$, what is $Pr(\theta_1 < \theta < \theta_2 \mid r, n)$.

1.4.2 Poisson

Examples: traffic accidents, typing errors, customers arriving in a bank, number of fixations in reading.

Probability density function

Let λ be the average number of events in the time interval $[0,1]$. Let the random variable X count the number of events occurring in the interval. Then:

$$f_X(x) = \mathbb{P}(X = x) = \mathrm{e}^{-\lambda}\frac{\lambda^x}{x!}, \quad x = 0, 1, 2, \dots \tag{47}$$

Expectation and variance

$$E[X] = \lambda \tag{48}$$

$$Var(X) = \lambda \tag{49}$$

Relevant functions in R

```
dpois(x, lambda)
ppois(q, lambda, lower.tail = TRUE)
qpois(p, lambda, lower.tail = TRUE)
rpois(n, lambda)
```

1.4.3 Uniform

Example: All outcomes have equal probability.

Probability density function:

A random variable (X) with the continuous uniform distribution on the interval (α, β) has PDF

$$f_X(x) = \begin{cases} \frac{1}{\beta - \alpha}, & \alpha < x < \beta, \\ 0, & \text{otherwise} \end{cases} \tag{50}$$

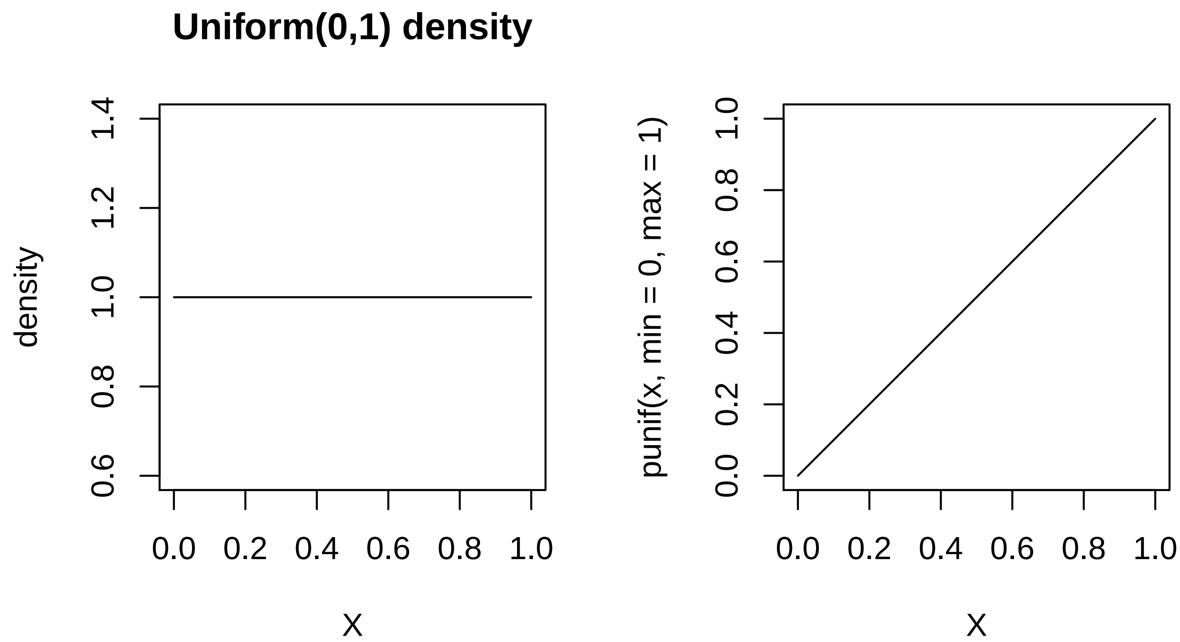


Figure 2: Uniform(0,1) distribution, pdf and cdf.

The associated R function is $\mathsf{dunif}(\mathtt{min} = a, \mathtt{max} = b)$. We write $X \sim \mathsf{unif}(\mathtt{min} = a, \mathtt{max} = b)$. Due to the particularly simple form of this PDF we can also write down explicitly a formula for the CDF F_X:

$$F_X(a) = \begin{cases} 0, & a < 0, \\ \frac{a-\alpha}{\beta-\alpha}, & \alpha \leq t < \beta, \\ 1, & a \geq \beta. \end{cases} \tag{51}$$

The pdf and cdf are show in Figure 2.

Expectation and variance:

$$E[X] = \frac{\beta + \alpha}{2} \tag{52}$$

$$Var(X) = \frac{(\beta - \alpha)^2}{12} \tag{53}$$

Relevant functions in R

```
dunif(x, min = 0, max = 1, log = FALSE)
punif(q, min = 0, max = 1, lower.tail = TRUE, log.p = FALSE)
```

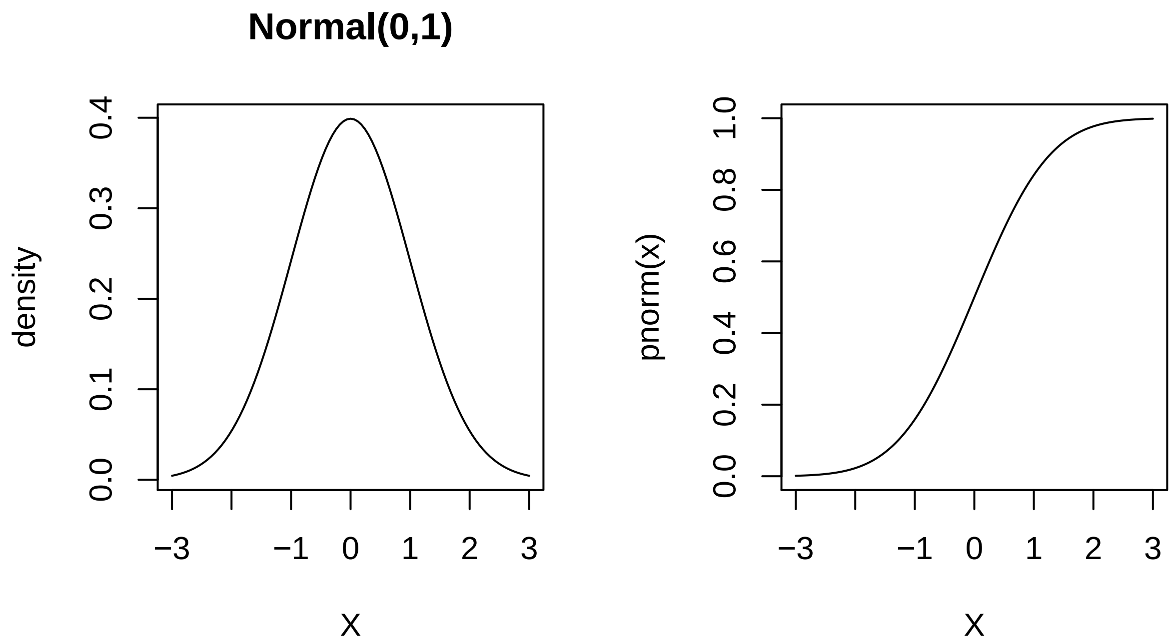


Figure 3: Normal distribution.

```
qunif(p, min = 0, max = 1, lower.tail = TRUE, log.p = FALSE)
runif(n, min = 0, max = 1)
```

1.4.4 Normal

Examples: heights, weights of people

Probability density function

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}, \quad -\infty < x < \infty. \tag{54}$$

We write $X \sim \mathsf{norm}(\mathtt{mean} = \mu, \mathtt{sd} = \sigma)$, and the associated R function is `dnorm(x, mean = 0, sd = 1)`.

If X is normally distributed with parameters μ and σ^2, then $Y = aX + b$ is normally distributed with parameters $a\mu + b$ and $a^2\sigma^2$.

Special case: Standard or unit normal random variable

If X is normally distributed with parameters μ and σ^2, then $Z = (X - \mu)/\sigma$ is normally distributed with parameters $0, 1$.

We conventionally write $\Phi(x)$ for the CDF:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-y^2}{2}} dy \quad \text{where } y = (x-\mu)/\sigma \tag{55}$$

Old-style (pre-computer era) printed tables give the values for positive x; for negative x we do:

$$\Phi(-x) = 1 - \Phi(x), \quad -\infty < x < \infty \tag{56}$$

If Z is a standard normal random variable (SNRV) then

$$p\{Z \leq -x\} = P\{Z > x\}, \quad -\infty < x < \infty \tag{57}$$

Since $Z = ((X-\mu)/\sigma)$ is an SNRV whenever X is normally distributed with parameters μ and σ^2, then the CDF of X can be expressed as:

$$F_X(a) = P\{X \leq a\} = P\left(\frac{X-\mu}{\sigma} \leq \frac{a-\mu}{\sigma}\right) = \Phi\left(\frac{a-\mu}{\sigma}\right) \tag{58}$$

The standardized version of a normal random variable X is used to compute specific probabilities relating to X (it is also easier to compute probabilities from different CDFs so that the two computations are comparable).

Expectation and variance

$$E[X] = \mu \tag{59}$$

$$Var(X) = \sigma^2 \tag{60}$$

Relevant functions in R

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```

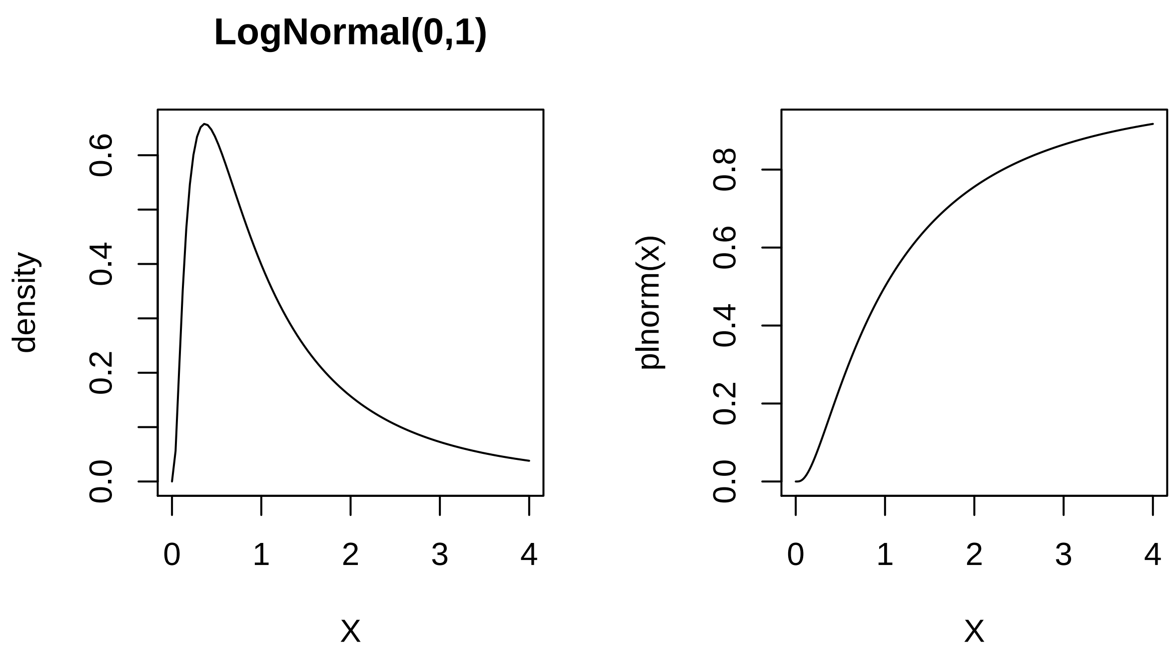


Figure 4: LogNormal(0,1) distribution.

1.4.5 Log-Normal

Examples: reaction time, reading time

Probability density function

$$f_X(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{\frac{-(\log x - \mu)^2}{2\sigma^2}}, \quad 0 < x < \infty. \tag{61}$$

```
op <- par(mfrow = c(1, 2), pty = "s")
plot(function(x) dlnorm(x), 0, 4, main = "LogNormal(0,1)", ylab = "density",
    xlab = "X")
x <- seq(0, 4, by = 0.001)
plot(x, plnorm(x), main = "", type = "l", xlab = "X")
```

Note:

-μ, σ are on the log scale. -If $X \sim LogNormal(\mu, \sigma)$, this is the same as saying that $\log(X)$ is normally distributed.

Expectation and variance

$$E[X] = \mu + \sigma/2 \tag{62}$$

$$Var(X) = \sigma^2 \tag{63}$$

Relevant functions in R

```
dlnorm(x, meanlog = 0, sdlog = 1)
plnorm(q, meanlog = 0, sdlog = 1, lower.tail = TRUE)
qlnorm(p, meanlog = 0, sdlog = 1, lower.tail = TRUE)
rlnorm(n, meanlog = 0, sdlog = 1)
```

1.4.6 Beta

Example: Distribution of probability of success.

Probability density function

This is a generalization of the continuous uniform distribution. Think of parameter a as number of successes, and parameter b as number of failures.

$$f(x) = \begin{cases} \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1} & \text{if } 0 < x < 1 \\ 0 & \text{otherwise} \end{cases}$$

where

$$Beta(a,b) = \int_0^1 x^{a-1}(1-x)^{b-1}\,dx$$

We write $X \sim \texttt{beta}(\texttt{shape1} = a, \texttt{shape2} = b)$.

Expectation and variance

$$E[X] = \frac{a}{a+b} \text{ and } Var(X) = \frac{ab}{(a+b)^2(a+b+1)}. \tag{64}$$

Relevant functions in R

```
dbeta(x, shape1, shape2)
pbeta(q, shape1, shape2)
qbeta(p, shape1, shape2)
rbeta(n, shape1, shape2)
```

1.4.7 Exponential

Examples: Waiting time for an arrival from a Poisson process (e.g., reading times)

Probability density function

For some $\lambda > 0$,

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x \geq 0 \\ 0 & \text{if } x < 0. \end{cases}$$

Expectation and variance

$$E[X] = \frac{1}{\lambda} \tag{65}$$

$$Var(X) = \frac{1}{\lambda^2} \tag{66}$$

Relevant functions in R

```
dexp(x, rate=1)
pexp(q, rate=1)
qexp(p, rate=1)
rexp(n, rate=1)
```

1.4.8 Gamma

Examples: reading times, distribution of inverse of variance.

Connection to Poisson: if X measures the length of time until the first event occurs in a Poisson process with rate λ then $X \sim \texttt{exp}(\texttt{rate} = \lambda)$. If we let Y measure the length of time until the α^{th} event occurs then $Y \sim \texttt{gamma}(\texttt{shape} = \alpha, \texttt{rate} = \lambda)$. When α is an integer this distribution is also known as the **Erlang** distribution.

The Chi-squared distribution is the Gamma distribution with $\lambda = 1/2$ and $\alpha = n/2$, where n is an integer:

Probability density function

This is a generalization of the exponential distribution. We say that X has a Gamma distribution and write $X \sim \texttt{gamma}(\texttt{shape} = \alpha, \texttt{rate} = \lambda)$, where $\alpha > 0$ (called shape) and $\lambda > 0$ (called rate). It has PDF

$$f(x) = \begin{cases} \frac{\lambda e^{-\lambda x}(\lambda x)^{\alpha-1}}{\Gamma(\alpha)} & \text{if } x \geq 0 \\ 0 & \text{if } x < 0. \end{cases}$$

$\Gamma(\alpha)$ is called the gamma **function** (note: it's lower case gamma, and it's a function, not a distribution):

$$\Gamma(\alpha) = \int_0^\infty e^{-y} y^{\alpha-1}\, dy = (\alpha-1)\Gamma(\alpha-1)$$

Note that for integral values of n, $\Gamma(n) = (n-1)!$ (follows from above equation).

Expectation and variance

$$E[X] = \alpha/\lambda \tag{67}$$

$$Var(X) = \alpha/\lambda^2 \tag{68}$$

Relevant functions in R

```
dgamma(x, rate, scale = 1/rate)
pgamma(q, rate, scale = 1/rate)
qgamma(p, rate, scale = 1/rate)
rgamma(n, rate, scale = 1/rate)
```

```r
x <- seq(0, 4, by = 0.01)
plot(x, gamma.fn(x), type = "l")
```

```r
x <- seq(0, 100, by = 0.01)
plot(x, gamma.fn(x), type = "l")
```

1.4.9 Student's t

Examples: reading times.

Probability density function

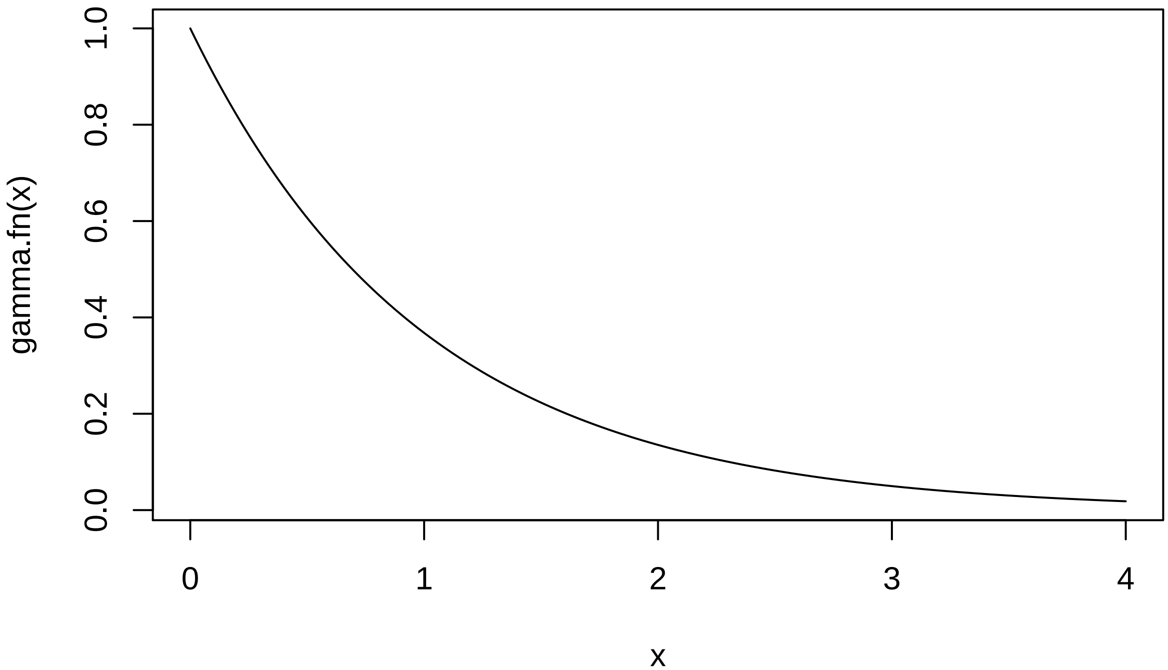


Figure 5: The Gamma distribution.

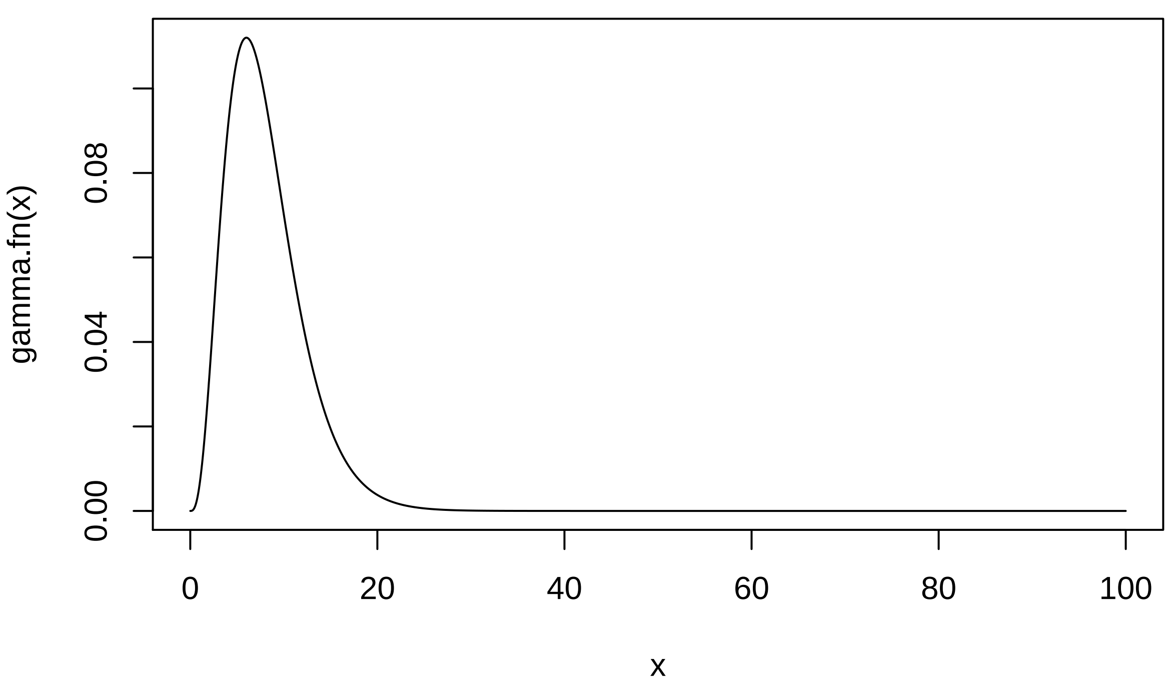


Figure 6: The chi-squared distribution.

A random variable X with PDF

$$f_X(x) = \frac{\Gamma[(r+1)/2]}{\sqrt{r\pi}\,\Gamma(r/2)} \left(1 + \frac{x^2}{r}\right)^{-(r+1)/2}, \quad -\infty < x < \infty \tag{69}$$

is said to have Student's t distribution with r degrees of freedom, and we write $X \sim \mathtt{t}(\mathtt{df} = r)$.

We will write $X \sim t(\mu, \sigma^2, r)$, where r is the degrees of freedom $(n-1)$, where n is sample size.

Expectation and variance

$$E[X] = 0 \text{ if } n > 1 \text{ otherwise undefined} \tag{70}$$

$$Var(X) = \frac{n}{n-2}, \text{ when } n > 2 \text{ otherwise undefined} \tag{71}$$

Relevant functions in R

```
dt(x, df)
pt(q, df)
qt(p, df)
rt(n, df)
```

1.4.10 Summary of distributions

Distribution	PMF/PDF and Support	Expected Value	Variance
Binomial $Bin(n,p)$	$P(X=k)=\binom{n}{k}p^k q^{n-k}$ $k \in \{0,1,2,\dots n\}$	np	npq
Poisson $Pois(\lambda)$	$P(X=k)=\frac{e^{-\lambda}\lambda^k}{k!}$ $k \in \{0, 1, 2, \dots\}$	λ	λ
Uniform $Unif(a,b)$	$f(x)=\frac{1}{b-a}$ $x \in (a,b)$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$
Normal $Normal(\mu,\sigma)$	$f(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-(x-\mu)^2/(2\sigma^2)}$ $x \in (-\infty,\infty)$	μ	
Log-Normal $LogNormal(\mu,\sigma)$	$\frac{1}{x\sigma\sqrt{2\pi}}e^{-(\log x-\mu)^2/(2\sigma^2)}$ $x \in (0,\infty)$	$\theta = e^{\mu+\sigma^2/2}$	$\theta^2(e^{\sigma^2}-1)$
Beta Beta(a,b)	$f(x)=\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}x^{a-1}(1-x)^{b-1}$ $x \in (0,1)$	$\mu=\frac{a}{a+b}$	$\frac{\mu(1-\mu)}{(a+b+1)}$
Exponential $Exp(\lambda)$	$f(x)=\lambda e^{-\lambda x}$ $x \in (0,\infty)$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$
Gamma $Gamma(a,\lambda)$	$f(x)=\frac{1}{\Gamma(a)}(\lambda x)^a e^{-\lambda x}\frac{1}{x}$ $x \in (0,\infty)$	$\frac{a}{\lambda}$	$\frac{a}{\lambda^2}$
Student-t $t(n)$	$\frac{\Gamma((n+1)/2)}{\sqrt{n\pi}\Gamma(n/2)}(1+x^2/n)^{-(n+1)/2}$ $x \in (-\infty,\infty)$	0 if $n>1$	$\frac{n}{n-2}$ if $n>2$

1.5 Jointly distributed random variables

1.5.1 Discrete case

[This section is an extract from Kerns.]

Consider two discrete random variables X and Y with PMFs f_X and f_Y that are supported on the sample spaces S_X and S_Y, respectively. Let $S_{X,Y}$ denote the set of all possible observed **pairs** (x,y), called the **joint support set** of X and Y. Then the **joint probability mass function** of X and Y is the function $f_{X,Y}$ defined by

$$f_{X,Y}(x,y) = \mathbb{P}(X = x, Y = y), \quad \text{for } (x,y) \in S_{X,Y}. \tag{72}$$

Every joint PMF satisfies

$$f_{X,Y}(x,y) > 0 \text{ for all } (x,y) \in S_{X,Y}, \tag{73}$$

and

$$\sum_{(x,y)\in S_{X,Y}} f_{X,Y}(x,y) = 1. \tag{74}$$

It is customary to extend the function $f_{X,Y}$ to be defined on all of $\mathbb{R}^2$ by setting $f_{X,Y}(x,y) = 0$ for $(x,y) \notin S_{X,Y}$.

In the context of this chapter, the PMFs f_X and f_Y are called the **marginal PMFs** of X and Y, respectively. If we are given only the joint PMF then we may recover each of the marginal PMFs by using the Theorem of Total Probability: observe

$$\begin{aligned} f_X(x) &= \mathbb{P}(X = x), & (75)\\ &= \sum_{y\in S_Y} \mathbb{P}(X = x, Y = y), & (76)\\ &= \sum_{y\in S_Y} f_{X,Y}(x,y). & (77) \end{aligned}$$

By interchanging the roles of X and Y it is clear that

$$f_Y(y) = \sum_{x\in S_X} f_{X,Y}(x,y). \tag{78}$$

Given the joint PMF we may recover the marginal PMFs, but the converse is not true. Even if we have **both** marginal distributions they are not sufficient to determine the joint PMF; more information is needed.

Associated with the joint PMF is the **joint cumulative distribution function** $F_{X,Y}$ defined by

$$F_{X,Y}(x,y) = \mathbb{P}(X \le x, Y \le y), \quad \text{for } (x,y) \in \mathbb{R}^2.$$

The bivariate joint CDF is not quite as tractable as the univariate CDFs, but in principle we could calculate it by adding up quantities of the form in Equation~72. The joint CDF is typically not used in practice due to its inconvenient form; one can usually get by with the joint PMF alone.

Example:

Roll a fair die twice. Let X be the face shown on the first roll, and let Y be the face shown on the second roll. For this example, it suffices to define

$$f_{X,Y}(x,y) = \frac{1}{36}, \quad x = 1,\ldots,6,\ y = 1,\ldots,6.$$

The marginal PMFs are given by $f_X(x) = 1/6$, $x = 1,2,\ldots,6$, and $f_Y(y) = 1/6$, $y = 1,2,\ldots,6$, since

$$f_X(x) = \sum_{y=1}^{6} \frac{1}{36} = \frac{1}{6}, \quad x = 1,\ldots,6,$$

and the same computation with the letters switched works for Y.

Here, and in many other ones, the joint support can be written as a product set of the support of X "times" the support of Y, that is, it may be represented as a cartesian product set, or rectangle, $S_{X,Y} = S_X \times S_Y$ where $S_X \times S_Y = \{(x,y) :\ x \in S_X,\ y \in S_Y\}$. This form is a necessary condition for X and Y to be **independent** (or alternatively **exchangeable** when $S_X = S_Y$). But please note that in general it is not required for $S_{X,Y}$ to be of rectangle form.

1.5.2 Continuous case

For random variables X and y, the **joint cumulative pdf** is

$$F(a,b) = P(X \leq a, Y \leq b) \quad -\infty < a,b < \infty \tag{79}$$

The **marginal distributions** of F_X and F_Y are the CDFs of each of the associated RVs:

1. The CDF of X:

$$F_X(a) = P(X \leq a) = F_X(a,\infty) \tag{80}$$

2. The CDF of Y:

$$F_Y(a) = P(Y \leq b) = F_Y(\infty,b) \tag{81}$$

Definition 1. ***Jointly continuous***: *Two RVs X and Y are jointly continuous if there exists a function $f(x,y)$ defined for all real x and y, such that for every set C:*

$$P((X,Y) \in C) = \iint_{(x,y)\in C} f(x,y)\, dx\, dy \tag{82}$$

*$f(x,y)$ is the **joint PDF** of X and Y.*

Every joint PDF satisfies

$$f(x,y) \geq 0 \textit{ for all } (x,y) \in S_{X,Y}, \tag{83}$$

and

$$\iint_{S_{X,Y}} f(x,y)\, \mathrm{d}x\, \mathrm{d}y = 1. \tag{84}$$

For any sets of real numbers A and B, and if $C = \{(x,y) : x \in A, y \in B\}$, it follows from equation~82 that

$$P((X \in A, Y \in B) \in C) = \int_B \int_A f(x,y)\, dx\, dy \tag{85}$$

Note that

$$F(a,b) = P(X \in (-\infty,a]), Y \in (-\infty,b])) = \int_{-\infty}^{b} \int_{-\infty}^{a} f(x,y)\, dx\, dy \tag{86}$$

Differentiating, we get the joint pdf:

$$f(a,b) = \frac{\partial^2}{\partial a \partial b} F(a,b) \tag{87}$$

One way to understand the joint PDF:

$$P(a < X < a+da, b < Y < b+db) = \int_{b}^{d+db} \int_{a}^{a+da} f(x,y)\, dx\, dy \approx f(a,b) da db \tag{88}$$

Hence, $f(x,y)$ is a measure of how probable it is that the random vector (X,Y) will be near (a,b).

1.5.3 Marginal probability distribution functions

If X and Y are jointly continuous, they are individually continuous, and their PDFs are:

$$
\begin{aligned}
P(X \in A) =& P(X \in A, Y \in (-\infty, \infty)) \\
&= \int_A \int_{-\infty}^{\infty} f(x,y)\, dy\, dx \\
&= \int_A f_X(x)\, dx
\end{aligned}
\tag{89}
$$

where

$$
f_X(x) = \int_{-\infty}^{\infty} f(x,y)\, dy \tag{90}
$$

Similarly:

$$
f_Y(y) = \int_{-\infty}^{\infty} f(x,y)\, dx \tag{91}
$$

1.5.4 Independent random variables

Random variables X and Y are independent iff, for any two sets of real numbers A and B:

$$
P(X \in A, Y \in B) = P(X \in A) P(Y \in B) \tag{92}
$$

In the jointly continuous case:

$$
f(x,y) = f_X(x) f_Y(y) \quad \text{for all } x, y \tag{93}
$$

A necessary and sufficient condition for the random variables X and Y to be independent is for their joint probability density function (or joint probability mass function in the discrete case) $f(x,y)$ to factor into two terms, one depending only on x and the other depending only on y. %This can be stated as a proposition:

Example from Kerns:
Let the joint PDF of (X,Y) be given by

$$
f_{X,Y}(x,y) = \frac{6}{5}\left(x + y^2\right), \quad 0 < x < 1,\ 0 < y < 1.
$$

The marginal PDF of X is

$$
\begin{aligned}
f_X(x) &= \int_0^1 \frac{6}{5}\left(x+y^2\right) \mathrm{d}y, \\
&= \frac{6}{5}\left(xy+\frac{y^3}{3}\right)\Bigg|_{y=0}^{1}, \\
&= \frac{6}{5}\left(x+\frac{1}{3}\right),
\end{aligned}
$$

for $0 < x < 1$, and the marginal PDF of Y is

$$
\begin{aligned}
f_Y(y) &= \int_0^1 \frac{6}{5}\left(x+y^2\right) \mathrm{d}x, \\
&= \frac{6}{5}\left(\frac{x^2}{2}+xy^2\right)\Bigg|_{x=0}^{1}, \\
&= \frac{6}{5}\left(\frac{1}{2}+y^2\right),
\end{aligned}
$$

for $0 < y < 1$.

In this example the joint support set was a rectangle $[0,1]\times[0,1]$, but it turns out that X and Y are not independent. This is because $\frac{6}{5}\left(x+y^2\right)$ cannot be stated as a product of two terms ($f_X(x)f_Y(y)$).

1.5.5 Sums of independent random variables

(Taken nearly verbatim from Ross 2002.)

Suppose that X and Y are independent, continuous random variables having probability density functions f_X and f_Y. The cumulative distribution function of $X+Y$ is obtained as follows:

$$
\begin{aligned}
F_{X+Y}(a) =& P(X+Y \leq a) \\
=& \iint\limits_{x+y\leq a} f_{XY}(x,y)\,dx\,dy \\
=& \iint\limits_{x+y\leq a} f_X(x) f_Y(y)\,dx\,dy \\
=& \int_{-\infty}^{\infty}\int_{-\infty}^{a-y} f_X(x) f_Y(y)\,dx\,dy \\
=& \int_{-\infty}^{\infty}\int_{-\infty}^{a-y} f_X(x)\,dx f_Y(y)\,dy \\
=& \int_{-\infty}^{\infty} F_X(a-y) f_Y(y)\,dy
\end{aligned} \tag{94}
$$

The CDF F_{X+Y} is the **convolution** of the distributions F_X and F_Y.

If we differentiate the above equation, we get the pdf f_{X+Y}:

$$\begin{aligned} f_{X+Y} =& \frac{d}{dx}\int_{-\infty}^{\infty} F_X(a-y) f_Y(y)\,dy \\ =& \int_{-\infty}^{\infty} \frac{d}{dx} F_X(a-y) f_Y(y)\,dy \\ =& \int_{-\infty}^{\infty} f_X(a-y) f_Y(y)\,dy \end{aligned} \tag{95}$$

1.5.6 Conditional distributions

1.5.6.1 Discrete case

Recall that the conditional probability of B given A, denoted $\mathbb{P}(B \mid A)$, is defined by

$$\mathbb{P}(B \mid A) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(A)}, \quad \text{if } \mathbb{P}(A) > 0. \tag{96}$$

If X and Y are discrete random variables, then we can define the conditional PMF of X given that $Y = y$ as follows:

$$\begin{aligned} p_{X|Y}(x \mid y) =& P(X = x \mid Y = y) \\ =& \frac{P(X = x, Y = y)}{P(Y = y)} \\ =& \frac{p(x,y)}{p_Y(y)} \end{aligned} \tag{97}$$

for all values of y where $p_Y(y) = P(Y = y) > 0$.

The **conditional cumulative distribution function** of X given $Y = y$ is defined, for all y such that $p_Y(y) > 0$, as follows:

$$\begin{aligned} F_{X|Y} =& P(X \leq x \mid Y = y) \\ =& \sum_{a \leq x} p_{X|Y}(a \mid y) \end{aligned} \tag{98}$$

If X and Y are independent then

$$p_{X|Y}(x \mid y) = P(X = x) = p_X(x) \tag{99}$$

See the examples starting p. 264 of Ross (2002).

1.5.6.2 Continuous case

(Taken almost verbatim from Ross 2002.)

If X and Y have a joint probability density function $f(x,y)$, then the conditional probability density function of X given that $Y = y$ is defined, for all values of y such that $f_Y(y) > 0$,by

$$f_{X|Y}(x \mid y) = \frac{f(x,y)}{f_Y(y)} \tag{100}$$

We can understand this definition by considering what $f_{X|Y}(x \mid y)\,dx$ amounts to:

$$\begin{aligned} f_{X|Y}(x \mid y)\,dx =& \frac{f(x,y)}{f_Y(y)}\frac{dxdy}{dy} \\ =& \frac{f(x,y)dxdy}{f_Y(y)dy} \\ =& \frac{P(x < X < d + dx, y < Y < y + dy)}{y < P < y + dy} \end{aligned} \tag{101}$$

1.5.7 Covariance and correlation

There are two very special cases of joint expectation: the **covariance** and the **correlation**. These are measures which help us quantify the dependence between X and Y.
Definition 2. *The **covariance** of X and Y is*

$$Cov(X,Y) = \mathbb{E}(X - \mathbb{E}X)(Y - \mathbb{E}Y). \tag{102}$$

Shortcut formula for covariance:

$$\text{Cov}(X,Y) = \mathbb{E}(XY) - (\mathbb{E}X)(\mathbb{E}Y). \tag{103}$$

The Pearson product moment correlation between X and Y is the covariance between X and Y rescaled to fall in the interval $[-1,1]$. It is formally defined by

$$\text{Corr}(X,Y) = \frac{\text{Cov}(X,Y)}{\sigma_X \sigma_Y}. \tag{104}$$

The correlation is usually denoted by $\rho_{X,Y}$ or simply ρ if the random variables are clear from context. There are some important facts about the correlation coefficient:

1. The range of correlation is $-1 \leq \rho_{X,Y} \leq 1$.
2. Equality holds above ($\rho_{X,Y} = \pm 1$) if and only if Y is a linear function of X with probability one.

1.5.8 Multivariate normal distributions

This is a very important distribution that we will need in linear mixed models.

Consider the bivariate case first. Suppose we have two univariate random variables $U0 \sim Normal(\mu_0, \sigma_{u0})$ and $U1 \sim Normal(\mu_1, \sigma_{u1})$ that have covariance $\rho\sigma_{u0}\sigma_{u1}$. We write this as:

$$\begin{pmatrix} U_0 \\ U_1 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_0 \\ \mu_1 \end{pmatrix}, \Sigma \right) \tag{105}$$

The variances and covariances between the two random variables are described by a 2×2 variance-covariance matrix. The diagonals have the variances, and the off-diagonals have the covariance between the two random variables.

$$\Sigma = \begin{pmatrix} \sigma_{u0}^2 & \rho\sigma_{u0}\sigma_{u1} \\ \rho\sigma_{u0}\sigma_{u1} & \sigma_{u1}^2 \end{pmatrix} \tag{106}$$

We develop some graphical intuition about bivariate distributions next.

1.5.8.1 Graphical intuition for the bivariate case

If we have two independent random variables U0, U1, and we examine their joint distribution, we can plot a 3-d plot which shows, u0, u1, and f(u0,u1).

Bivariate distribution with no correlation (independent random variables): E.g., $u0 \sim N(0,1)$ and $u1 \sim N(0,1)$, with two independent random variables. See Figure 7.

```
library(mvtnorm)
u0 <- u1 <- seq(from = -3, to = 3, length.out = 30)
Sigma1 <- diag(2)
f <- function(u0, u1) dmvnorm(cbind(u0, u1), mean = c(0, 0),
    sigma = Sigma1)
z <- outer(u0, u1, FUN = f)
```

```
persp(u0, u1, z, theta = -30, phi = 30, ticktype = "detailed")
```

Bivariate distribution with positive correlation: see Figure 8.

```
Sigma2 <- matrix(c(1, 0.6, 0.6, 1), byrow = FALSE, ncol = 2)
f <- function(u0, u1) dmvnorm(cbind(u0, u1), mean = c(0, 0),
    sigma = Sigma2)
z <- outer(u0, u1, FUN = f)
```

```
persp(u0, u1, z, theta = -30, phi = 30, ticktype = "detailed")
```

Bivariate distribution with negative correlation: see Figure 9.

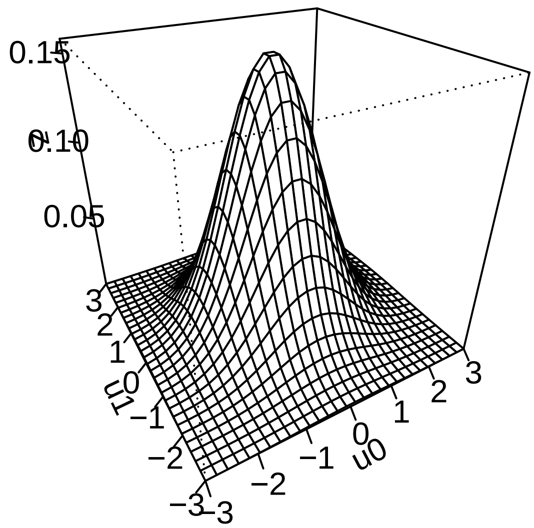


Figure 7: Visualization of two uncorrelated random variables.

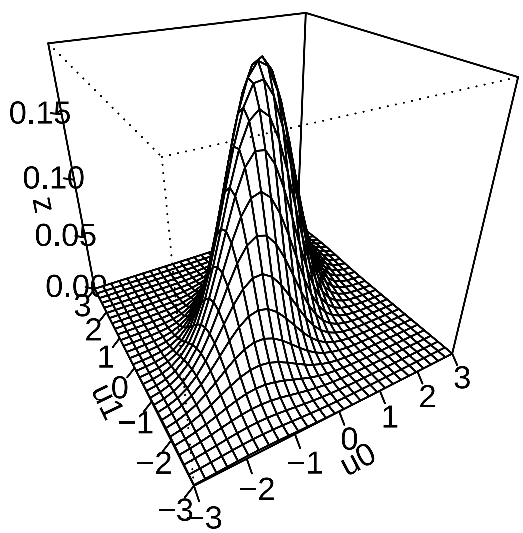


Figure 8: Visualization of two positively correlated random variables.

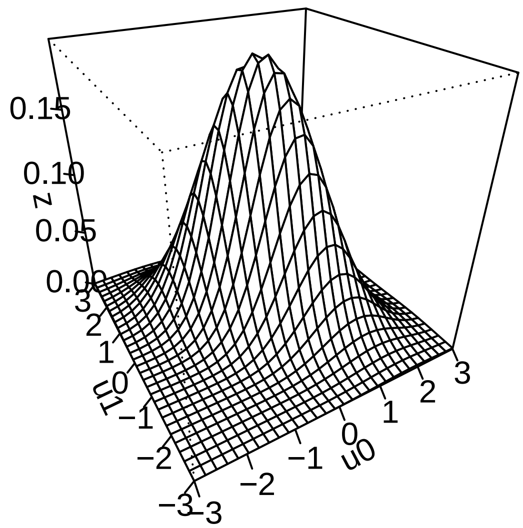


Figure 9: Visualization of two negatively correlated random variables.

```
Sigma3 <- matrix(c(1, -0.6, -0.6, 1), byrow = FALSE, ncol = 2)
f <- function(u0, u1) dmvnorm(cbind(u0, u1), mean = c(0, 0),
    sigma = Sigma3)
z <- outer(u0, u1, FUN = f)
```

```
persp(u0, u1, z, theta = -30, phi = 30, ticktype = "detailed")
```

1.5.8.2 Visualizing conditional distributions in a bivariate

You can run the following code to get a visualization of what a conditional distribution looks like when we take "slices" from the conditioning random variable:

```
bivn <- mvrnorm(1000, mu = c(0, 1), Sigma = matrix(c(1, 0, 0,
    2), 2))
bivn.kde <- kde2d(bivn[, 1], bivn[, 2], n = 50)

for (i in 1:50) {
    plot(bivn.kde$z[i, 1:50], type = "l", ylim = c(0, 0.1))
    Sys.sleep(0.5)
}
```

If you run this code, you will see "slices" from the bivariate distribution.

1.5.8.3 Formal definition of the multivariate normal

Having acquired a graphical intuition, we turn to the formal definitions. Recall that in the univariate normal distribution:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e\{-\frac{(\frac{(x-\mu)}{\sigma})^2}{2}\} \quad -\infty < x < \infty \tag{107}$$

We can write the power of the exponential as:

$$\left(\frac{(x-\mu)}{\sigma}\right)^2 = (x-\mu)(x-\mu)(\sigma^2)^{-1} = (x-\mu)(\sigma^2)^{-1}(x-\mu) = Q \tag{108}$$

Generalizing this to the multivariate case:

$$Q = (x-\mu)'\Sigma^{-1}(x-\mu) \tag{109}$$

Σ is a variance-covariance matrix, and Σ^{-1} is its inverse.

So, for multivariate case:

$$f(x) = \frac{1}{\sqrt{2\pi det(\Sigma)}} e\{-Q/2\} \quad -\infty < x_i < \infty, i = 1, \ldots, n \tag{110}$$

$det(\Sigma)$ is the determinant of the matrix.

Properties of the multivariate normal (MVN) X:

- Linear combinations of X are normal distributions.
- All subsets of X's components have a normal distribution.
- Zero covariance implies independent distributions.
- Conditional distributions are normal.

1.6 Maximum likelihood estimation

1.6.1 Discrete case

Suppose the observed independent sample values are $x_1, x_2, \ldots, x_n$ from some random variable with pmf $P(\cdot)$ that has a parameter θ. The probability of getting these particular values is

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = f(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n; \theta) \tag{111}$$

i.e., the function f is the value of the joint probability **distribution** of the random variables $X_1, \ldots, X_n$ at $X_1 = x_1, \ldots, X_n = x_n$. Since the sample values have been observed and are fixed, $f(x_1, \ldots, x_n; \theta)$ is a function of θ. The function f is called a **likelihood function**.

1.6.2 Continuous case

Here, f is the joint probability **density**, the rest is the same as above.

Definition 3. *If $x_1, x_2, \ldots, x_n$ are the values of a random sample from a population with parameter θ, the **likelihood function** of the sample is given by*

$$L(\theta) = f(x_1, x_2, \ldots, x_n; \theta) \tag{112}$$

for values of θ within a given domain. Here, $f(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n; \theta)$ is the joint probability distribution or density of the random variables $X_1, \ldots, X_n$ at $X_1 = x_1, \ldots, X_n = x_n$.

So, the method of maximum likelihood consists of maximizing the likelihood function with respect to θ. The value of θ that maximizes the likelihood function is the **MLE** (maximum likelihood estimate) of θ.

1.6.3 Finding maximum likelihood estimates for different distributions

Example 1

Let X_i, $i = 1, \ldots, n$ be a random variable with PDF $f(x; \sigma) = \frac{1}{2\sigma} exp(-\frac{|x|}{\sigma})$. Find $\hat{\sigma}$, the MLE of σ.

$$L(\sigma) = \prod f(x_i; \sigma) = \frac{1}{(2\sigma)^n} exp(-\sum \frac{|x_i|}{\sigma}) \tag{113}$$

Let ℓ be log likelihood (log lik). The log likelihood is much easier to work with, because products become sums. Then:

$$\ell(x; \sigma) = \sum \left[-\log 2 - \log \sigma - \frac{|x_i|}{\sigma} \right] \tag{114}$$

Differentiating and equating to zero to find maximum:

$$\ell'(\sigma) = \sum \left[-\frac{1}{\sigma} + \frac{|x_i|}{\sigma^2} \right] = -\frac{n}{\sigma} + \frac{|x_i|}{\sigma^2} = 0 \tag{115}$$

Rearranging the above, the MLE for σ is:

$$\hat{\sigma} = \frac{\sum |x_i|}{n} \tag{116}$$

Example 2: Poisson

$$L(\mu;x) = \prod \frac{\exp^{-\mu}\mu^{x_i}}{x_i!} \tag{117}$$

$$= \exp^{-\mu}\mu^{\sum x_i}\frac{1}{\prod x_i!} \tag{118}$$

Log lik:

$$\ell(\mu;x) = -n\mu + \sum x_i \log\mu - \sum \log y! \tag{119}$$

Differentiating:

$$\ell'(\mu) = -n + \frac{\sum x_i}{\mu} = 0 \tag{120}$$

Therefore:

$$\hat{\lambda} = \frac{\sum x_i}{n} \tag{121}$$

Example 3: Binomial

$$L(\theta) = \binom{n}{x}\theta^x(1-\theta)^{n-x} \tag{122}$$

Log lik:

$$\ell(\theta) = \log\binom{n}{x} + x\log\theta + (n-x)\log(1-\theta) \tag{123}$$

Differentiating:

$$\ell'(\theta) = \frac{x}{\theta} - \frac{n-x}{1-\theta} = 0 \tag{124}$$

Thus:

$$\hat{\theta} = \frac{x}{n} \tag{125}$$

Example 4: Normal

Let $X_1,\ldots,X_n$ constitute a random variable of size n from a normal population with mean μ and variance σ^2, find joint maximum likelihood estimates of these two parameters.

$$L(\mu;\sigma^2) = \prod N(x_i;\mu,\sigma) \tag{126}$$
$$= (\frac{1}{2\pi\sigma^2})^{n/2}\exp(-\frac{1}{2\sigma^2}\sum(x_i-\mu)^2) \tag{127}$$
$$\tag{128}$$

Taking logs and differentiating with respect to μ and σ^2, we get:

$$\hat{\mu} = \frac{1}{n}\sum x_i = \bar{x} \tag{129}$$

and

$$\hat{\sigma}^2 = \frac{1}{n}\sum(x_i-\bar{x})^2 \tag{130}$$

1.6.4 Visualizing likelihood and maximum log likelihood for normal

For simplicity consider the case where $N(\mu=0,\sigma^2=1)$.

```
op<-par(mfrow=c(1,2),pty="s")
plot(function(x) dnorm(x,log=F), -3, 3,
      main = "Normal density",#ylim=c(0,.4),
              ylab="density",xlab="X")
abline(h=0.4)
plot(function(x) dnorm(x,log=T), -3, 3,
      main = "Normal density (log)",#ylim=c(0,.4),
              ylab="density",xlab="X")
abline(h=log(0.4))
```

1.6.5 MLE using R

1.6.5.1 One-parameter case

Estimating θ for the binomial distribution: Let's assume we have the result of 10 coin tosses. We know that the MLE is the number of successes divided by the sample size:

```
x <- rbinom(10, 1, prob = 0.5)
sum(x)/length(x)
```

```
## [1] 0.5
```

We will now get this number using MLE. We do it numerically to illustrate the principle. First, we define a negative log likelihood function for the binomial. Negative because the function we will use to optimize does minimization by default, so we just flip the sign on the log likelihood to convert the maximum to a minimum.

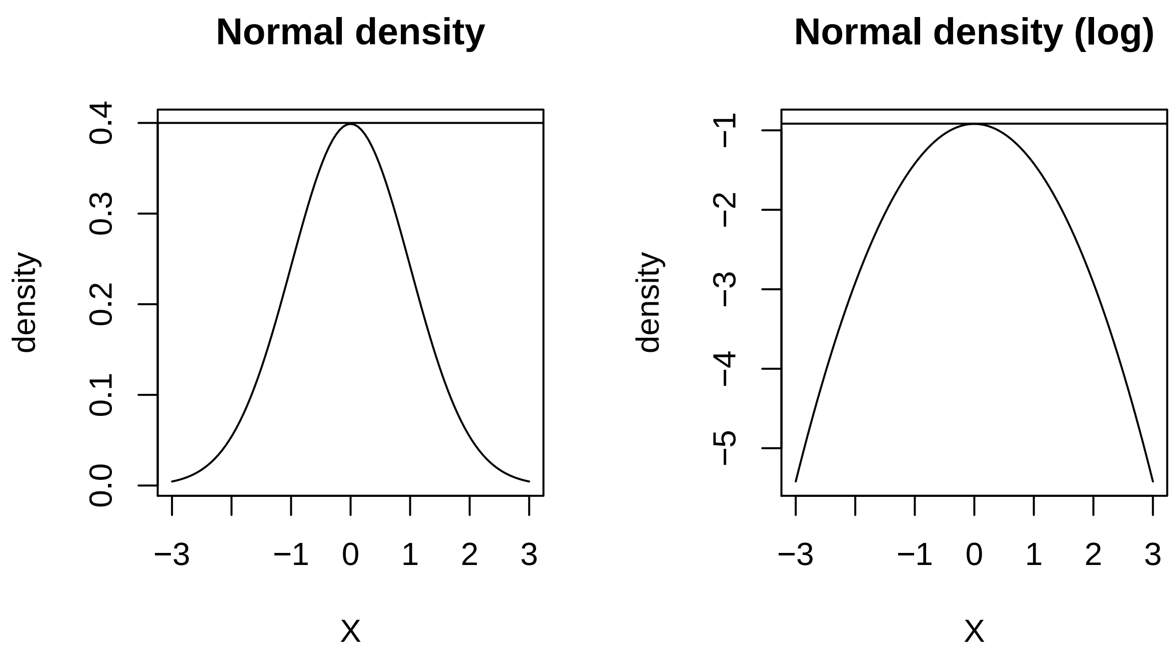


Figure 10: Maximum likelihood and log likelihood.

```
negllbinom <- function(p, x) {
    -sum(dbinom(x, size = 1, prob = p, log = T))
}
```

Then we run the optimization function:

```
optimize(negllbinom, interval = c(0, 1), x = x)
```

```
## $minimum
## [1] 0.5
##
## $objective
## [1] 6.93
```

1.6.5.2 Two-parameter case

Here is an example of MLE using R. Note that in this example, we could have analytically figured out the MLEs. Instead, we are doing this numerically. The advantage of the numerical approach becomes obvious when the analytical way is closed to us.

Assume that you have some data that was generated from a normal distribution, with mean 500, and standard deviation 50. Let's say you have 100 data points.

```
data <- rnorm(100, mean = 500, sd = 50)
```

Let's assume we don't know what the mean and standard deviation are. Now, of course you know how to estimate these using the standard formulas. But right now we are going to estimate them using MLE.

We first write down the negation of the log likelihood function. We take the negation because the optimization function we will be using (see below) does minimization by default, so to get the maximum with the default setting, we just change the sign.

The function `nllh.normal` takes a vector `theta` of parameter values, and a data frame `data`.

```
nllh.normal <- function(theta, data) {
    ## decompose the parameter vector to its two parameters:
    m <- theta[1]
    s <- theta[2]
    ## read in data
    x <- data
    n <- length(x)
    ## log likelihood:
    logl <- sum(dnorm(x, mean = m, sd = s, log = TRUE))
    ## return negative log likelihood:
    -logl
}
```

Here is the negative log lik for mean = 40, sd 4, and for mean = 800 and sd 4:

```
nllh.normal(theta = c(40, 4), data)
```

```
## [1] 651313
```

```
nllh.normal(theta = c(800, 4), data)
```

```
## [1] 300834
```

As we would expect, the negative log lik for mean 500 and sd 50 is much smaller (due to the sign change) than the two log liks above:

```
nllh.normal(theta = c(500, 50), data)
```

```
## [1] 532
```

Basically, you could sit here forever, playing with combinations of values for mean and sd to find the combination that gives the optimal log likelihood. R has an optimization function that does this for you. We have to specify some sensible starting values:

```
opt.vals.default <- optim(theta <- c(700, 40), nllh.normal,
    data = data, hessian = TRUE)
```

Finally, we print out the estimated parameter values that maximize the likelihood:

```
(estimates.default <- opt.vals.default$par)
```

```
## [1] 493.8  49.2
```

Knowledge of maximum likelihood estimation will be needed in the next chapter.

2 Introduction to Bayesian data analysis

Recall Bayes' rule:
Theorem 1. ***Bayes' Rule****. Let B_1, B_2, ..., B_n be mutually exclusive and exhaustive and let A be an event with $\mathbb{P}(A) > 0$. Then*

$$\mathbb{P}(B_k|A) = \frac{\mathbb{P}(B_k)\mathbb{P}(A|B_k)}{\sum_{i=1}^{n} \mathbb{P}(B_i)\mathbb{P}(A|B_i)}, \quad k = 1, 2, \ldots, n. \tag{131}$$

When A and B are observable events, we can state the rule as follows:

$$p(A \mid B) = \frac{p(B \mid A)p(A)}{p(B)} \tag{132}$$

Note that $p(\cdot)$ is the probability of an event.

When looking at probability distributions, we will encounter the rule in the following form.

$$f(\theta \mid \text{data}) = \frac{f(\text{data} \mid \theta)f(\theta)}{f(y)} \tag{133}$$

Here, $f(\cdot)$ is a probability density, not the probability of a single event. $f(y)$ is called a "normalizing constant", which makes the left-hand side a probability distribution.

$$f(y) = \int f(x,\theta)\,d\theta = \int f(y \mid \theta)f(\theta)\,d\theta \tag{134}$$

If θ is a discrete random variable taking one value from the set $\{\theta_1, \ldots, \theta_n\}$, then

$$f(y) = \sum_{i=1}^{n} f(y \mid \theta_i)P(\theta = \theta_i) \tag{135}$$

Without the normalizing constant, we have the relationship:

$$f(\theta \mid \text{data}) \propto f(\text{data} \mid \theta)f(\theta) \tag{136}$$

Note that the likelihood $L(\theta; \text{data})$ (our data is fixed) is proportional to $f(\text{data} \mid \theta)$, and that's why we can refer to $f(\text{data} \mid \theta)$ as the likelihood in the following Bayesian incantation:

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior} \tag{137}$$

Our central goal is going to be to derive the posterior distribution and then summarize its properties (mean, median, 95% credible interval, etc.).

Usually, we don't need the normalizing constant to understand the properties of the posterior distribution. That's why Bayes' theorem is often stated in terms of the proportionality shown above.

Incidentally, this is supposed to be the moment of great divide between frequentists and Bayesians: the latter assign a probability distribution to the parameter, the former treat the parameter as a point value.

Two examples will clarify how we can use Bayes' rule to obtain the posterior. Both examples involve so-called conjugate priors, which are defined as follows:

> Given the likelihood $f(x \mid \theta)$, if the prior $f(\theta)$ results in a posterior $f(\theta \mid x)$ that has the same form as $f(\theta)$, then we call $f(\theta)$ a conjugate prior.

2.1 Example 1: Binomial Likelihood, Beta prior, Beta posterior

This is a contrived example, just meant to provide us with a smooth entry into Bayesian data analysis. Suppose that an individual with aphasia answered 46 out of 100 questions correctly in a particular sentence comprehension task. The research question is, what is the probability that their average response is greater than 0.5, i.e., above chance.

The likelihood function will tell us $P(\text{data} \mid \theta)$:

```
dbinom(46, 100, 0.5)
```

```
## [1] 0.058
```

Note that

$$P(\text{data} \mid \theta) \propto \theta^{46}(1-\theta)^{54} \tag{138}$$

So, to get the posterior, we just need to work out a prior distribution $f(\theta)$.

$$f(\theta \mid \text{data}) \propto f(\text{data} \mid \theta) f(\theta) \tag{139}$$

For the prior, we need a distribution that can represent our uncertainty about the probabiliy θ of success. The Beta distribution is commonly used as prior for proportions. We say that the Beta distribution is conjugate to the binomial density; i.e., the two densities have similar functional forms.

The pdf is

$$f(x) = \begin{cases} \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1} & \text{if } 0 < x < 1 \\ 0 & \text{otherwise} \end{cases}$$

where

$$B(a,b) = \int_0^1 x^{a-1}(1-x)^{b-1}\, dx$$

In R, we write $X \sim \texttt{beta}(\texttt{shape1} = \alpha, \texttt{shape2} = \beta)$. The associated R function is $\texttt{dbeta}(\texttt{x}, \texttt{shape1}, \texttt{shape2})$.

The mean and variance are

$$E[X] = \frac{a}{a+b} \text{ and } Var(X) = \frac{ab}{(a+b)^2\,(a+b+1)}. \tag{140}$$

The Beta distribution's parameters a and b can be interpreted as (our beliefs about) prior successes and failures, and are called **hyperparameters**. Once we choose values for a and b, we can plot the Beta pdf. Here, we show the Beta pdf for three sets of values of a,b:

```
plot(function(x) dbeta(x, shape1 = 1, shape2 = 1), 0, 1, main = "Beta density",
    ylab = "density", xlab = "X", ylim = c(0, 3))

text(0.5, 1.1, "a=1,b=1")

plot(function(x) dbeta(x, shape1 = 3, shape2 = 3), 0, 1, add = T)
text(0.5, 1.6, "a=3,b=3")


plot(function(x) dbeta(x, shape1 = 6, shape2 = 6), 0, 1, add = T)
text(0.5, 2.6, "a=6,b=6")
```

As Figure 11 shows, as the a,b values are increased, the spread decreases.

If we don't have much prior information, we could use a=b=1; this gives us a uniform prior; this is called an uninformative prior or non-informative prior (although having no prior knowledge is, strictly speaking, not uninformative). If we have a lot of prior knowledge and/or a strong belief that θ has a particular value, we can use a larger a,b to reflect our greater certainty about the parameter. Notice that the larger our parameters a and b, the narrower the spread of the distribution; this makes sense because a larger sample size (a greater number of successes a, and a greater number of failures b) will lead to more precise estimates.

The central point is that the Beta distribution can be used to define the prior distribution of θ.

Just for the sake of illustration, let's take four different beta priors, each reflecting increasing certainty.

1. Beta(a=2,b=2)
2. Beta(a=3,b=3)
3. Beta(a=6,b=6)
4. Beta(a=21,b=21)

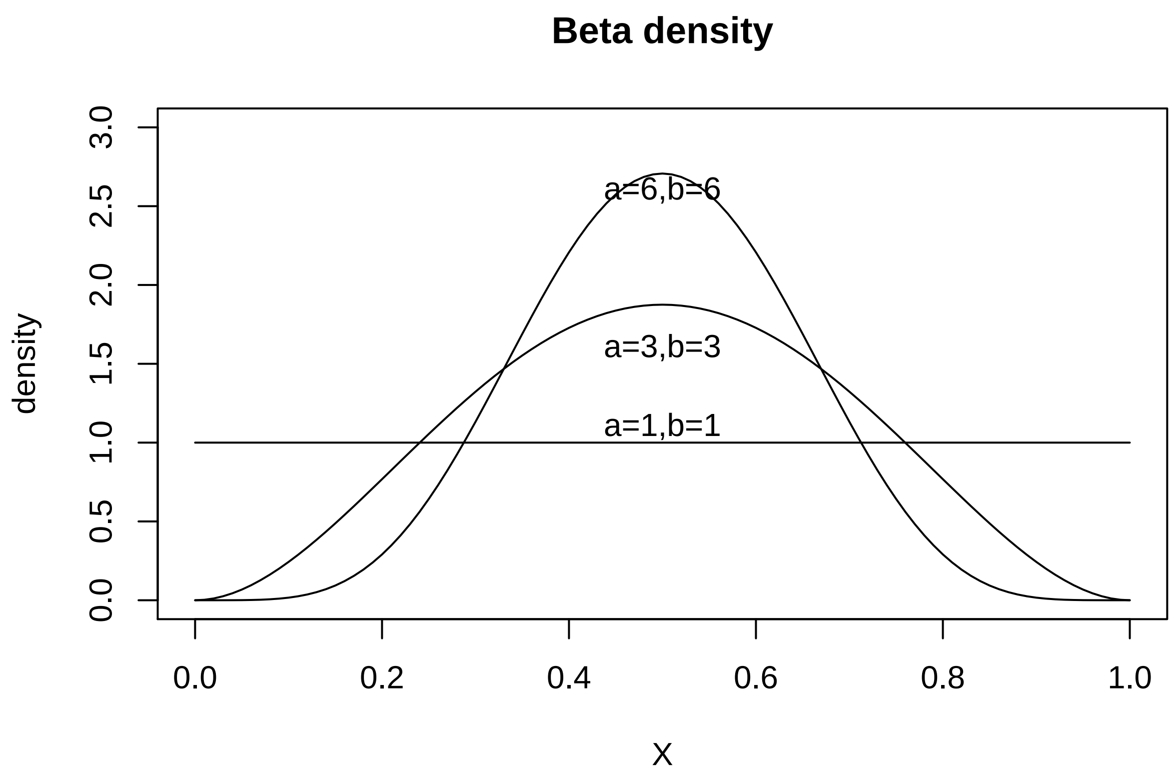


Figure 11: Examples of the beta distribution with different parameter values.

Each reflects a belief that $\theta = 0.5$, with varying degrees of (un)certainty. Now we just need to plug in the likelihood and the prior:

$$f(\theta \mid \text{data}) \propto f(\text{data} \mid \theta) f(\theta) \tag{141}$$

The four corresponding posterior distributios would be:

$$f(\theta \mid \text{data}) \propto [\theta^{46}(1-\theta)^{54}][\theta^{2-1}(1-\theta)^{2-1}] = \theta^{47}(1-\theta)^{55} \tag{142}$$

$$f(\theta \mid \text{data}) \propto [p^{46}(1-p)^{54}][p^{3-1}(1-p)^{3-1}] = p^{48}(1-p)^{56} \tag{143}$$

$$f(\theta \mid \text{data}) \propto [\theta^{46}(1-\theta)^{54}][\theta^{6-1}(1-\theta)^{6-1}] = \theta^{51}(1-\theta)^{59} \tag{144}$$

$$f(\theta \mid \text{data}) \propto [\theta^{46}(1-\theta)^{54}][\theta^{21-1}(1-\theta)^{21-1}] = \theta^{66}(1-\theta)^{74} \tag{145}$$

We can now visualize each of these triplets of priors, likelihoods and posteriors. Note that I use the beta to model the likelihood because this allows me to visualize all three (prior, lik., posterior) in the same plot. The likelihood function is actually as shown in Figure 12:

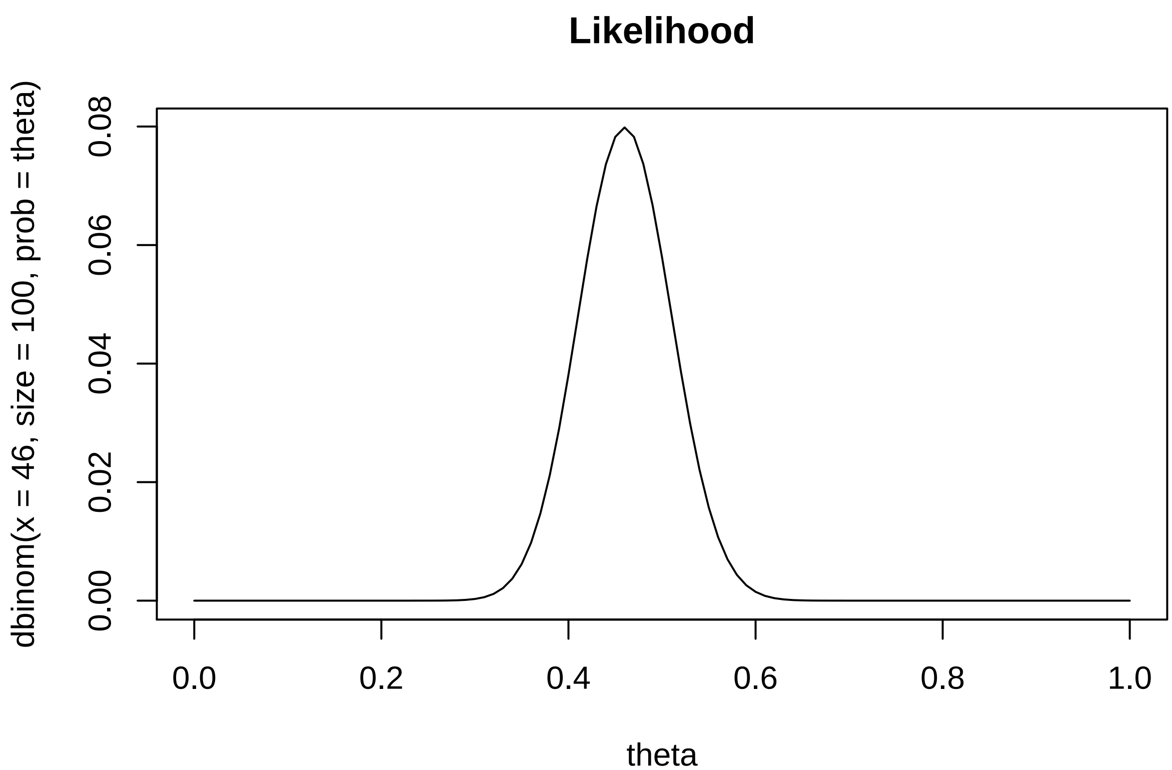


Figure 12: Binomial likelihood function.

```
theta=seq(0,1,by=0.01)

plot(theta,dbinom(x=46,size=100,prob=theta),
     type="l",main="Likelihood")
```

We can represent the likelihood in terms of the Beta as well, as shown in Figure 13:

```
plot(function(x)
  dbeta(x,shape1=46,shape2=54),0,1,
              ylab="",xlab="X")
```

2.2 Example 2: Poisson Likelihood, Gamma prior, Gamma posterior

This is also a contrived example. Suppose we are modeling the number of times that a speaker says the word "the" per day.

The number of times x that the word is uttered in one day can be modeled by a Poisson distribution:

$$f(x \mid \theta) = \frac{\exp(-\theta)\theta^x}{x!} \tag{146}$$

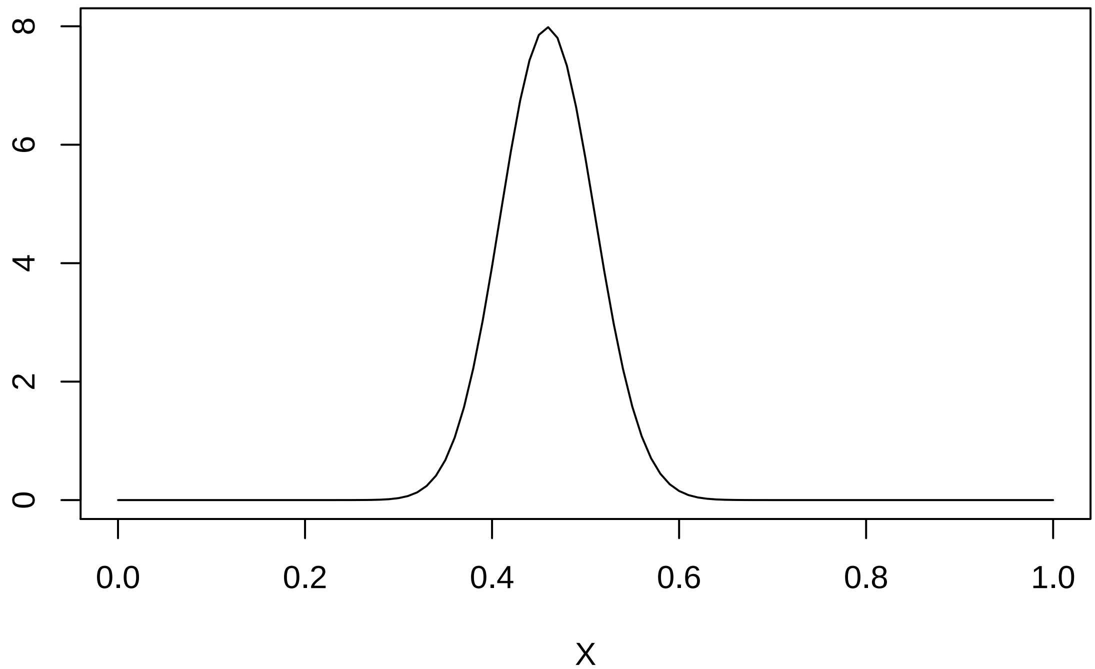


Figure 13: Using the beta distribution to represent a binomial.

where the rate θ is unknown, and the numbers of utterances of the target word on each day are independent given θ.

We are told that the prior mean of θ is 100 and prior variance for θ is 225. This information could be based on the results of previous studies on the topic.

In order to visualize the prior, we first fit a Gamma density prior for θ based on the above information.

Note that we know that for a Gamma density with parameters a, b, the mean is $\frac{a}{b}$ and the variance is $\frac{a}{b^2}$. Since we are given values for the mean and variance, we can solve for a,b, which gives us the Gamma density.

If $\frac{a}{b} = 100$ and $\frac{a}{b^2} = 225$, it follows that $a = 100 \times b = 225 \times b^2$ or $100 = 225 \times b$, i.e., $b = \frac{100}{225}$.

This means that $a = \frac{100 \times 100}{225} = \frac{10000}{225}$. Therefore, the Gamma distribution for the prior is as shown below (also see Figure 14):

$$\theta \sim Gamma(\frac{10000}{225}, \frac{100}{225}) \tag{147}$$

```
x <- 0:200
plot(x, dgamma(x, 10000/225, 100/225), type = "l", lty = 1,
    main = "Gamma prior", ylab = "density", cex.lab = 2, cex.main = 2,
    cex.axis = 2)
```

A distribution for a prior is **conjugate** if, multiplied by the likelihood, it yields a posterior that has

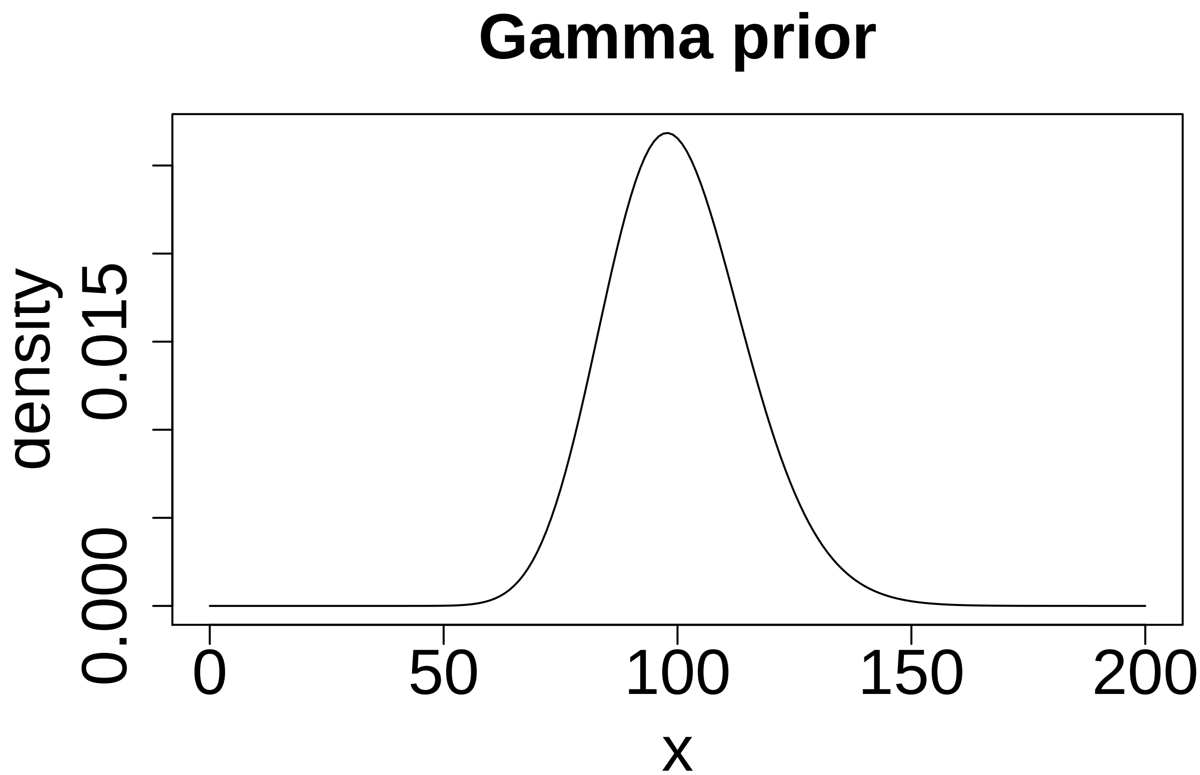


Figure 14: The Gamma prior for the parameter theta.

the distribution of the same family as the prior.

It turns out that the Gamma distribution is a conjugate prior for the Poisson distribution. For the Gamma distribution to be a conjugate prior for the Poisson, the posterior needs to have the general form of a Gamma distribution. We derive this conjugacy result below. The proof just involves very simple algebra.

Given that

$$\text{Posterior} \propto \text{Prior Likelihood} \tag{148}$$

and given that the likelihood is:

$$\begin{aligned} L(\mathbf{x} \mid \theta) &= \prod_{i=1}^{n} \frac{\exp(-\theta)\theta^{x_i}}{x_i!} \\ &= \frac{\exp(-n\theta)\theta^{\sum_i^n x_i}}{\prod_{i=1}^{n} x_i!} \end{aligned} \tag{149}$$

we can compute the posterior as follows:

$$\text{Posterior} = \left[\frac{\exp(-n\theta)\theta^{\sum_i^n x_i}}{\prod_{i=1}^n x_i!}\right]\left[\frac{b^a\theta^{a-1}\exp(-b\theta)}{\Gamma(a)}\right] \tag{150}$$

We can disregard the terms $x!, \Gamma(a), b^a$, which do not involve θ, we have

$$\begin{aligned}\text{Posterior} \propto &\exp(-n\theta)\theta^{\sum_i^n x_i}\theta^{a-1}\exp(-b\theta)\\ =&\theta^{a-1+\sum_i^n x_i}\exp(-\theta(b+n))\end{aligned} \tag{151}$$

We can now figure out the parameters of the posterior distribution, and show that it will be a Gamma distribution.

First, note that the Gamma distribution in general is $Gamma(a,b) \propto \theta^{a-1}\exp(-\theta b)$. So it's enough to state the above as a Gamma distribution with some parameters a, b.

If we equate $a^* - 1 = a - 1 + \sum_i^n x_i$ and $b^* = b + n$, we can rewrite the above as:

$$\theta^{a^*-1}\exp(-\theta b^*) \tag{152}$$

This means that $a^* = a + \sum_i^n x_i$ and $b^* = b + n$. We can find a constant k such that the above is a proper probability density function, i.e.:

$$\int_{-\infty}^{\infty} k\theta^{a^*-1}\exp(-\theta b^*) = 1 \tag{153}$$

Thus, the posterior has the form of a Gamma distribution with parameters $a^* = a + \sum_i^n x_i, b^* = b + n$. Hence the Gamma distribution is a conjugate prior for the Poisson.

2.2.1 Concrete example given data

Suppose we know that the number of "the" utterances is $115, 97, 79, 131$. We can derive the posterior distribution as follows.

The prior is Gamma(a=10000/225,b=100/225). The data are as given; this means that $\sum_i^n x_i = 422$ and sample size $n = 4$. It follows that the posterior is

$$\begin{aligned}Gamma(a^* = a + \sum_i^n x_i, b^* = b + n) =&Gamma(10000/225 + 422, 4 + 100/225)\\ =&Gamma(466.44, 4.44)\end{aligned} \tag{154}$$

The mean and variance of this distribution can be computed using the fact that the mean is $\frac{a*}{b*} = 466.44/4.44 = 104.95$ and the variance is $\frac{a*}{b*^2} = 466.44/4.44^2 = 23.66$.

```r
### load data:
data <- c(115, 97, 79, 131)

a.star <- function(a, data) {
    return(a + sum(data))
}

b.star <- function(b, n) {
    return(b + n)
}

new.a <- a.star(10000/225, data)
new.b <- b.star(100/225, length(data))

### post. mean
(post.mean <- new.a/new.b)
```

```
## [1] 105
```

```r
### post. var:
(post.var <- new.a/(new.b^2))
```

```
## [1] 23.6
```

Now suppose you get one new data point:

```r
new.data <- c(200)
```

We can compute the parameters of the posterior Gamma distributions using the function we wrote above:

```r
new.a.2 <- a.star(new.a, new.data)
new.b.2 <- b.star(new.b, length(new.data))

### new mean
(new.post.mean <- new.a.2/new.b.2)
```

```
## [1] 122
```

```r
### new var:
(new.post.var <- new.a.2/(new.b.2^2))
```

```
## [1] 22.5
```

Thus, new data can be used with the previous posterior distribution as prior, to compute the new posterior.

2.2.2 The posterior is a weighted mean of prior and likelihood

We can express the posterior mean as a weighted sum of the prior mean and the maximum likelihood estimate of θ.

The posterior mean is:

$$\frac{a*}{b*} = \frac{a + \sum x_i}{n + b} \tag{155}$$

This can be rewritten as

$$\frac{a*}{b*} = \frac{a + n\bar{x}}{n + b} \tag{156}$$

Dividing both the numerator and denominator by b:

$$\frac{a*}{b*} = \frac{(a + n\bar{x})/b}{(n + b)/b} = \frac{a/b + n\bar{x}/b}{1 + n/b} \tag{157}$$

Since a/b is the mean m of the prior, we can rewrite this as:

$$\frac{a/b + n\bar{x}/b}{1 + n/b} = \frac{m + \frac{n}{b}\bar{x}}{1 + \frac{n}{b}} \tag{158}$$

We can rewrite this as:

$$\frac{m + \frac{n}{b}\bar{x}}{1 + \frac{n}{b}} = \frac{m \times 1}{1 + \frac{n}{b}} + \frac{\frac{n}{b}\bar{x}}{1 + \frac{n}{b}} \tag{159}$$

This is a weighted average: setting $w_1 = 1$ and $w_2 = \frac{n}{b}$, we can write the above as:

$$m\frac{w_1}{w_1 + w_2} + \bar{x}\frac{w_2}{w_1 + w_2} \tag{160}$$

A n approaches infinity, the weight on the prior mean m will tend towards 0, making the posterior mean approach the maximum likelihood estimate of the sample.

In general, in a Bayesian analysis, as sample size increases, the likelihood will dominate in determining the posterior mean.

Regarding variance, since the variance of the posterior is:

$$\frac{a*}{b*^2} = \frac{(a + n\bar{x})}{(n + b)^2} \tag{161}$$

as n approaches infinity, the posterior variance will approach zero: more data will reduce variance (uncertainty).

2.3 Summary

We saw two examples where we can do the computations to derive the posterior using simple algebra. There are several other such simple cases. However, in realistic data analysis settings, we cannot specify the posterior distribution as a particular density. We can only specify the priors and the likelihood.

For such cases, we need to use MCMC sampling techniques so that we can sample from the posterior distributions of the parameters.

We will discuss three approaches for sampling:

- Gibbs sampling using inversion sampling
- Metropolis-Hasting
- Hamiltonian Monte Carlo

2.4 MCMC sampling

2.4.1 The inversion method for sampling

This method works when we know the closed form of the pdf we want to simulate from and can derive the inverse of that function.

Steps:

1. Sample one number u from $Unif(0,1)$. Let $u = F(z) = \int_L^z f(x)\,dx$ (here, L is the lower bound of the pdf f).
2. Then $z = F^{-1}(u)$ is a draw from $f(x)$.

2.4.1.1 Example 1: Samples from Standard Normal

Take a sample from the Uniform(0,1):

```
u <- runif(1, min = 0, max = 1)
```

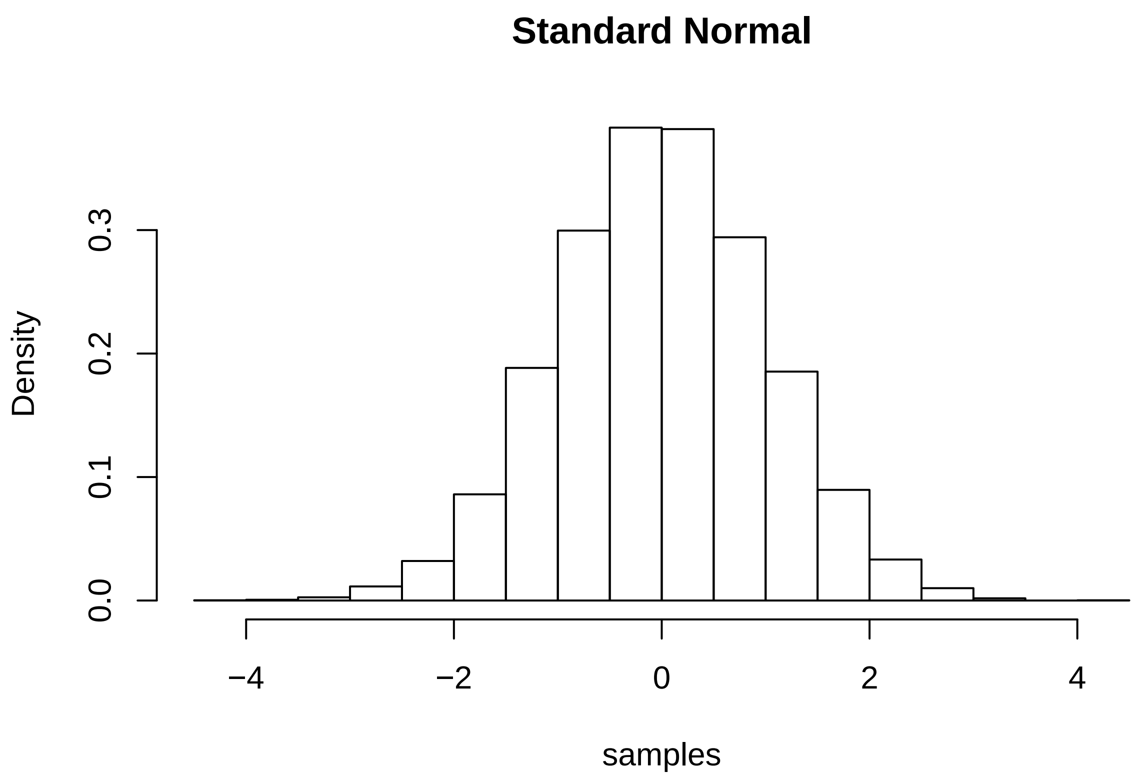
Let f(x) be a Normal density—we want to sample from this density. The inverse of the CDF in R is qnorm. It takes as input a probability and returns a quantile.

```
qnorm(u)
```

```
## [1] 1.67
```

If we do this repeatedly, we will get samples from the Normal distribution (here, the standard normal).

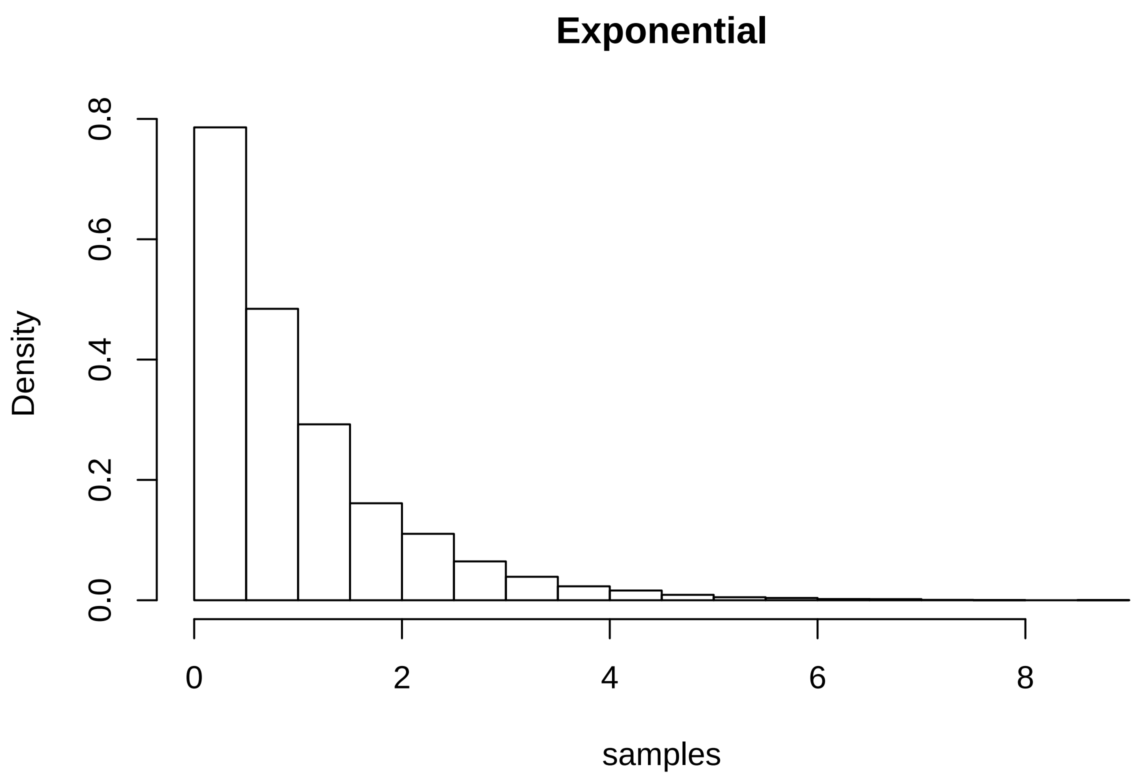
```
nsim <- 10000
samples <- rep(NA, nsim)
for (i in 1:nsim) {
    u <- runif(1, min = 0, max = 1)
    samples[i] <- qnorm(u)
}
hist(samples, freq = FALSE, main = "Standard Normal")
```



2.4.1.2 Example 2: Samples from Exponential or Gamma

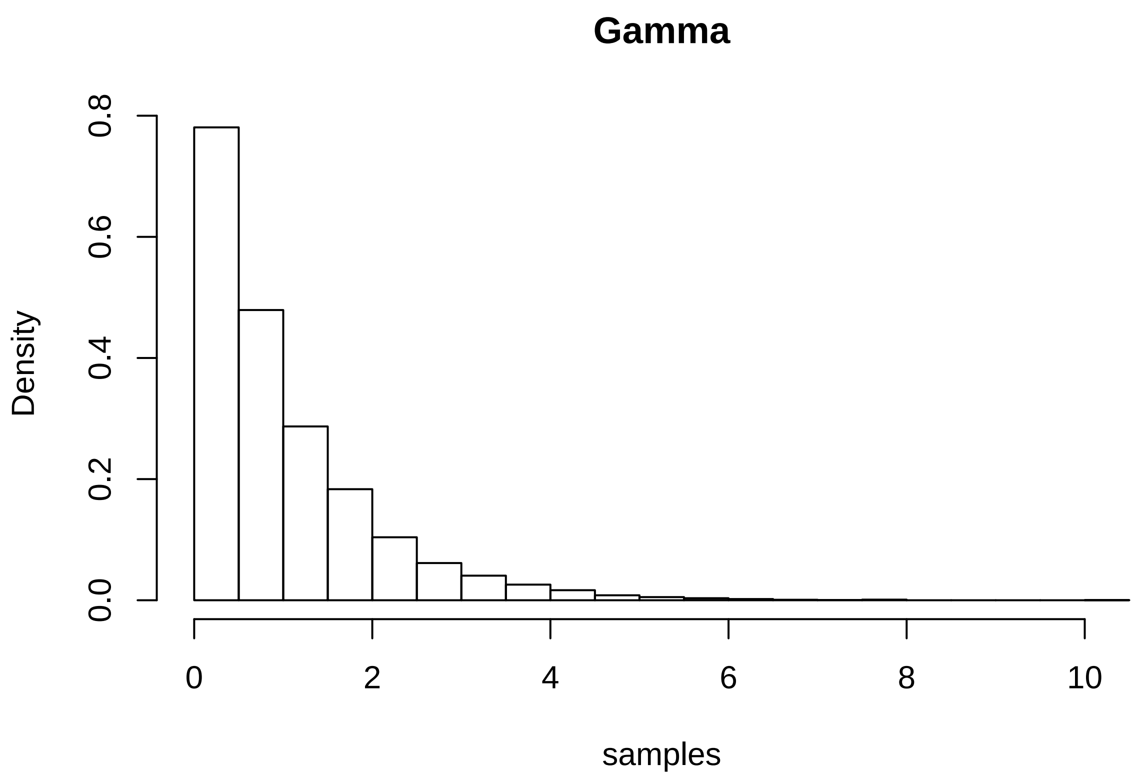
Now try this with the exponential with rate 1:

```
nsim <- 10000
samples <- rep(NA, nsim)
for (i in 1:nsim) {
    u <- runif(1, min = 0, max = 1)
    samples[i] <- qexp(u)
}
hist(samples, freq = FALSE, main = "Exponential")
```



Or the Gamma with rate and shape 1:

```
nsim <- 10000
samples <- rep(NA, nsim)
for (i in 1:nsim) {
    u <- runif(1, min = 0, max = 1)
    samples[i] <- qgamma(u, rate = 1, shape = 1)
}
hist(samples, freq = FALSE, main = "Gamma")
```



2.4.1.3 Example 3

Let $f(x) = \frac{1}{40}(2x+3)$, with $0 < x < 5$. Now, we can't just use the family of q functions in R, because this density is not defined in R.

We have to draw a number from the uniform distribution and then solve for z, which amounts to finding the inverse function:

$$u = \int_0^z \frac{1}{40}(2x+3) \tag{162}$$

```
u <- runif(1000, min = 0, max = 1)

z <- (1/2) * (-3 + sqrt(160 * u + 9))
```

This method can't be used if we can't find the inverse, and it can't be used with multivariate distributions.

2.4.2 Gibbs sampling

Gibbs sampling is a very commonly used method in Bayesian statistics. Here is how it works.

Let Θ be a vector of parameter values, let length of Θ be k. Let j index the j-th iteration.

Algorithm:

1. Assign some starting values to Θ:

   $\Theta^{j=0} \leftarrow S$

2. Set $j \leftarrow j+1$

3. 1. Sample $\theta_1^j \mid \theta_2^{j-1} \dots \theta_k^{j-1}$.

   2. Sample $\theta_2^j \mid \theta_1^j \theta_3^{j-1} \dots \theta_k^{j-1}$.

$\vdots$

   k. Sample $\theta_k^j \mid \theta_1^j \dots \theta_{k-1}^j$.

4. Return to step 1.

2.4.2.1 Example: A simple bivariate distribution

Assume that our bivariate (joint) density is:

$$f(x,y) = \frac{1}{28}(2x+3y+2) \tag{163}$$

Using the methods discussed in the Foundations chapter, it is possible to analytically work out the conditional distributions from the joint distribution:

$$f(x \mid y) = \frac{f(x,y)}{f(y)} = \frac{(2x+3y+2)}{6y+8} \tag{164}$$

$$f(y \mid x) = \frac{f(x,y)}{f(x)} = \frac{(2x+3y+2)}{4y+10} \tag{165}$$

The Gibbs sampler algorithm is:

1. Set starting values for the two parameters $x=-5, y=-5$. Set j=0.
2. Sample x^{j+1} from $f(x \mid y)$ using inversion sampling. You need to work out the inverse of $f(x \mid y)$ and $f(y \mid x)$ first. To do this, for $f(x \mid u)$, we have find z_1:

$$u = \int_0^{z_1} \frac{(2x+3y+2)}{6y+8}\, dx \tag{166}$$

And for $f(y \mid x)$, we have to find z_2:

$$u = \int_0^{z_2} \frac{(2x+3y+2)}{4y+10}\, dy \tag{167}$$

It doesn't matter if you can't solve this integral; the solution is given in the code below.

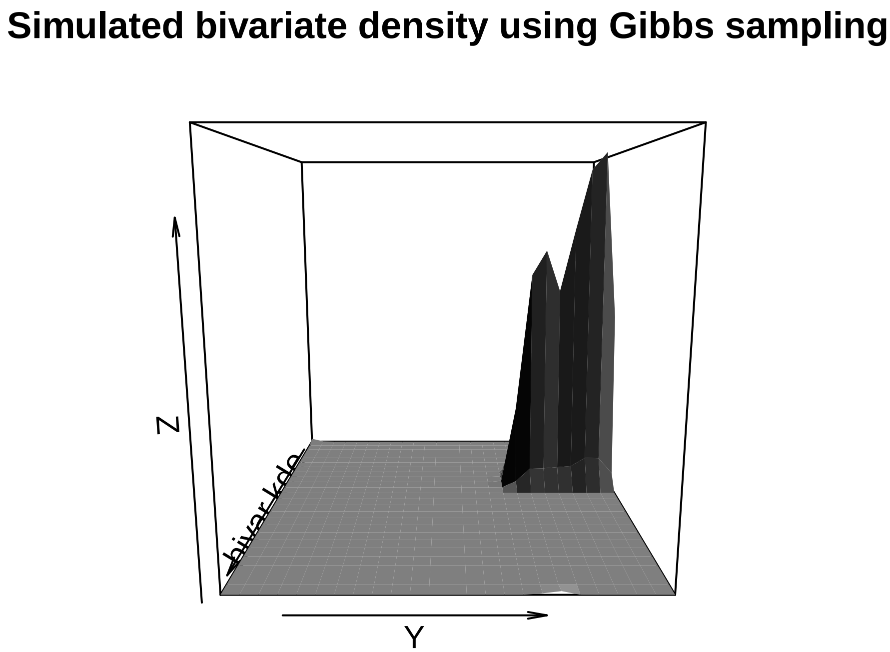


Figure 15: Example of posterior distribution of a bivariate distribution.

```
# R program for Gibbs sampling using inversion method
# program by Scott Lynch, modified by SV:
x <- rep(NA, 2000)
y <- rep(NA, 2000)
x[1] <- -5
y[1] <- -5

for (i in 2:2000) {
    # sample from x | y
    u <- runif(1, min = 0, max = 1)
    x[i] <- sqrt(u * (6 * y[i - 1] + 8) + (1.5 * y[i - 1] +
        1) * (1.5 * y[i - 1] + 1)) - (1.5 * y[i - 1] + 1)
    # sample from y | x
    u <- runif(1, min = 0, max = 1)
    y[i] <- sqrt((2 * u * (4 * x[i] + 10))/3 + ((2 * x[i] +
        2)/3) * ((2 * x[i] + 2)/3)) - ((2 * x[i] + 2)/3)
}
```

You can run this code to visualize the simulated posterior distribution. See Figure 15.

```
library(MASS)
bivar.kde <- kde2d(x, y)
persp(bivar.kde, phi = 10, theta = 90, shade = 0, border = NA,
    main = "Simulated bivariate density using Gibbs sampling")
```

A central insight here is that knowledge of the conditional distributions is enough to simulate from the joint distribution, provided such a joint distribution exists.

2.5 Hamiltonian Monte Carlo

Instead of Gibbs sampling, Stan uses this more efficient sampling approach. HMC works well for the high-dimensional models we will fit (hierarchical models). Gibbs sampling faces difficulties with some of the complex hierarchical models we will be fitting later. HMC will always succeed for these complex models.

One limitation of HMC (which Gibbs sampling does not have) is that HMC only works with continuous parameters (not discrete parameters).

For our purposes, it is enough to know what sampling using MCMC is, and that HMC gives us posterior samples efficiently. A good reference explaining HMC is Neal and others (2011). However, this paper is technically very demanding. More intuitively accessible introductions are available via Michael Betancourt's home page: https://betanalpha.github.io/. In particular, this video is helpful: https://youtu.be/jUSZboSq1zg.

2.5.1 HMC demonstration

The HMC algorithm takes as input the log density and the gradient of the log density. In Stan, these will be computed internally; the user doesn't need to do any computations.

For example, suppose the log density is $f(\theta) = -\frac{\theta^2}{2}$. Its gradient is $f'(\theta) = -\theta$. Setting this gradient to 0, and solving for θ, we know that the maximum is at 0. We know it's a maximum because the second derivative, $f''(\theta) = -1$, is negative. See Figure 16.

This is the machinery we learnt in the foundations chapter (recall how we found MLEs in particular).

```
theta <- seq(-4, 4, by = 0.001)
plot(theta, -theta^2/2, type = "l", main = "Log density")
```

We first write down the Radford Neal algorithm for HMC; details can be ignored in the next piece of code below, which I got from Jarad Niemi's github repository.

```
## Radford Neal algorithm:
HMC_neal <- function(U, grad_U, e, L, current_theta) {
    theta = current_theta
    omega = rnorm(length(theta), 0, 1)
    current_omega = omega
    omega = omega - e * grad_U(theta)/2
    for (i in 1:L) {
        theta = theta + e * omega
        if (i != L)
            omega = omega - e * grad_U(theta)
```

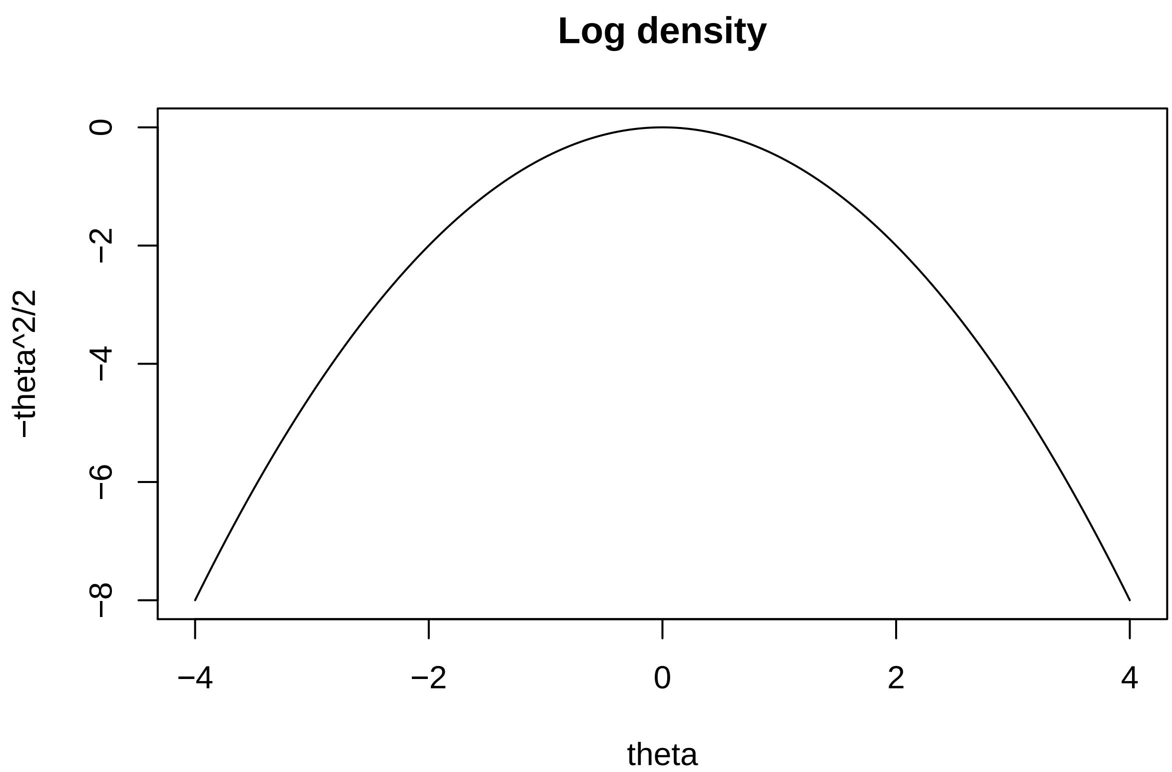


Figure 16: Example log density.

```
    }
    omega = omega - e * grad_U(theta)/2
    omega = -omega
    current_U = U(current_theta)
    current_K = sum(current_omega^2)/2
    proposed_U = U(theta)
    proposed_K = sum(omega^2)/2
    if (runif(1) < exp(current_U - proposed_U + current_K -
        proposed_K)) {
        return(theta)
    } else {
        return(current_theta)
    }
}

HMC <- function(n_reps, log_density, grad_log_density, tuning,
    initial) {
    theta = rep(0, n_reps)
    theta[1] = initial$theta
    for (i in 2:n_reps) theta[i] = HMC_neal(U = function(x) -log_density(x),
        grad_U = function(x) -grad_log_density(x), e = tuning$e,
```

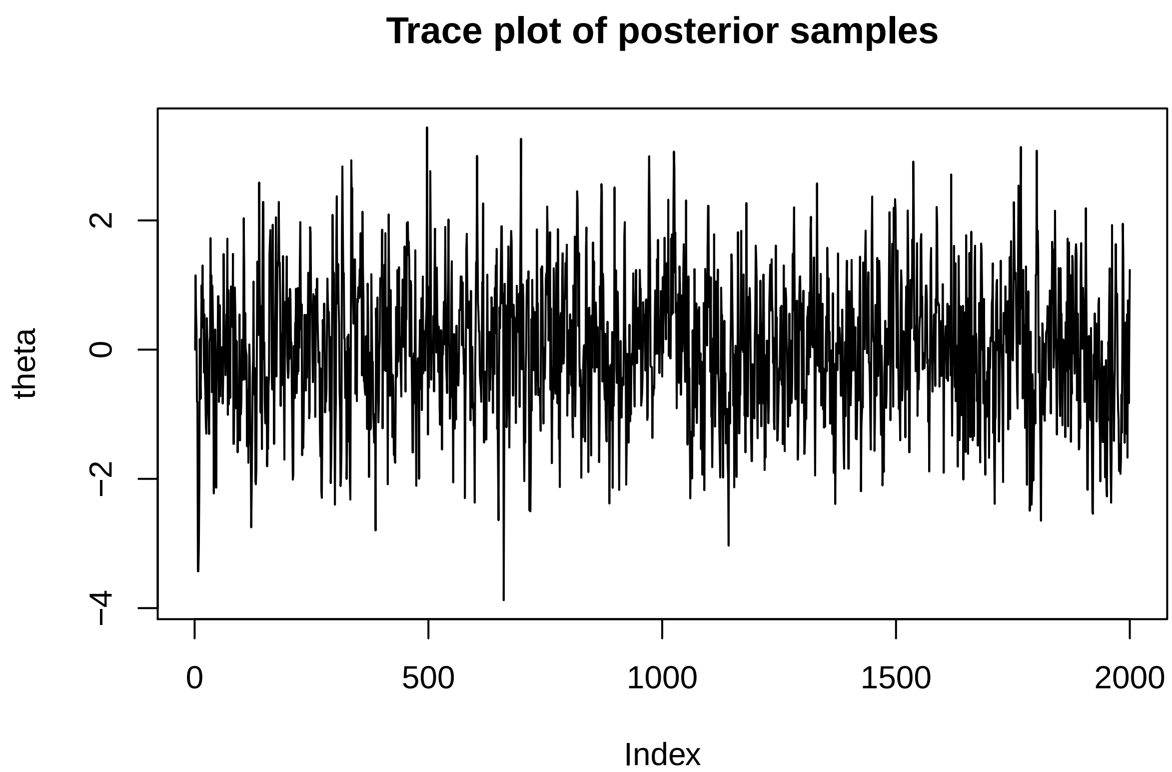


Figure 17: An example of a trace plot.

```
        L = tuning$L, theta[i - 1])
    theta
}
```

Then, we use the HMC function above to take 2000 samples from the posterior. We drop the first few (typically, the first half) samples, which are called warm-ups. The reason we drop them is that the initial samples may not yet be sampling from the posterior.

```
theta <- HMC(n_reps = 2000, log_density = function(x) -x^2/2,
    grad_log_density = function(x) -x, tuning = list(e = 1,
        L = 1), initial = list(theta = 0))
```

Figure 17 shows a **trace plot**, the trajectory of the samples over 2000 iterations. This is called a **chain**. When the sampler is correctly sampling from the posterior, we see a characteristic "fat hairy caterpillar" shape, and we say that the sampler has **converged**. You will see later what a failed convergence looks like.

```
plot(theta, type = "l", main = "Trace plot of posterior samples")
```

When we fit Bayesian models, we will always run four parallel chains. This is to make sure that even if we start with four different initial values chosen randomly, the chains all end up sampling from the same distribution. When this happens, we see that the chains overlap visually, and we say that the chains are **mixing**.

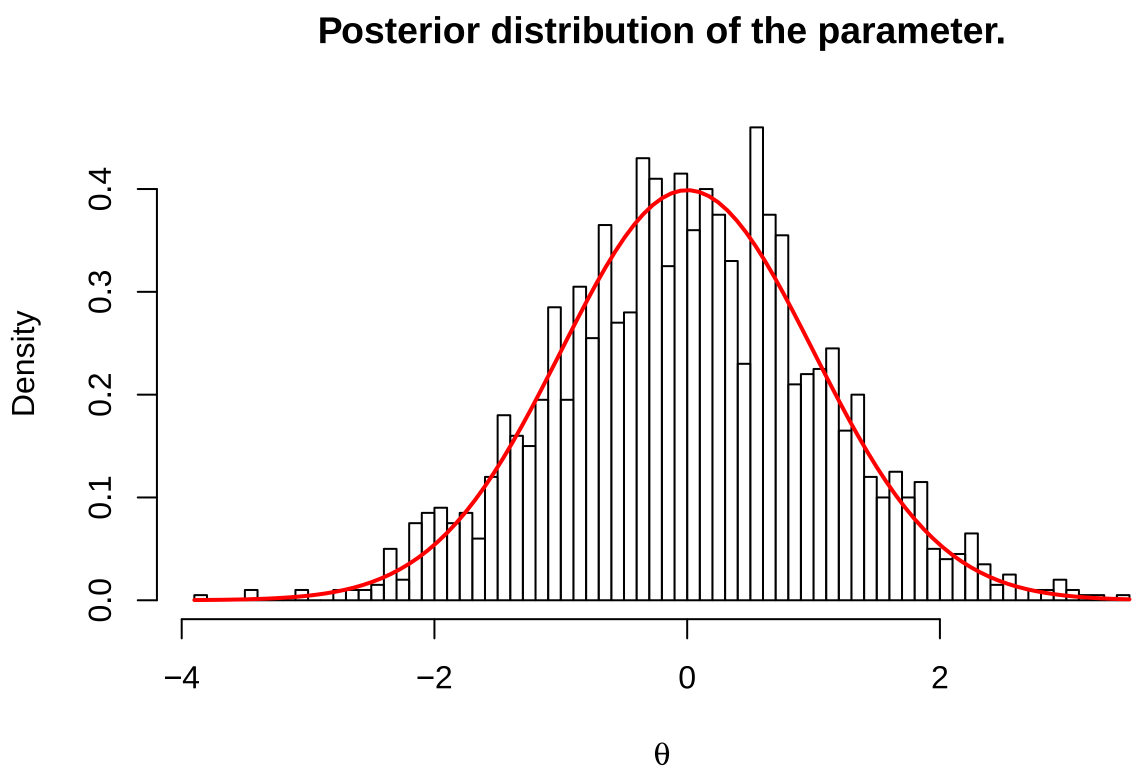


Figure 18: Sampling from the posterior using HMC. The red curve shows the distribution Normal(0,1).

Figure 18 shows the posterior distribution of θ. We are not discarding samples here because the sampler converges quickly in this simple example.

```
hist(theta, freq=F, 100,

     main="Posterior distribution of the parameter.",
     xlab=expression(theta))
curve(dnorm, add=TRUE, col='red', lwd=2)
```

In the modeling we do in the following pages, the Stan software will do the sampling for us.

2.6 Further readings

Some good books introducing Bayesian data analysis methods are the following:

- McElreath, R. (2015). Statistical Rethinking. Texts in Statistical Science. This book is currently the best textbook in existence, and assumes no calculus or linear algebra knowledge.
- Kruschke, J. (2014). Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan. Academic Press. This book is specifically designed for a psychology audience. I have only read parts of it and I find it very useful as a reference.
- Lynch, S. M. (2007). Introduction to applied Bayesian statistics and estimation for social scientists. Springer Science & Business Media. This book assumes knowledge of calculus and linear algebra. It gives a classical, statistician's introduction to Bayes. I highly recommend this book to people who know some (undergraduate math level) calculus and linear algebra.

2.7 Exercises

1. Suppose we are given that the pdf of θ, which ranges from 0 to 1, is proportional to θ^2. This means that there is some constant c (the constant of proportionality) such that $1 = c\int_0^1 \theta^2\, d\theta$.
   (a) Find c.
   (b) Find the mean, median (hint: what is the median in terms of quantiles?) and variance of the above pdf.
   (c) Find the 95% credible interval, i.e., the lower and upper values in $P(lower < \theta < upper) = 0.95$.
2. Exercise on the Binomial distribution: Toss a coin 10 times and compute, using `pbinom`, the probability of getting the total numbers of heads you got, assuming that the coin is fair.
   Hint: given sample size n, your assumed probability of a heads *prob*, and the number of heads you got x, the `pbinom` function delivers $P(X \leq x)$, the probability of getting x heads or less. In other words, pbinom is the **cumulative distribution function** (textbooks often call this simply **distribution function**).
   Here is how you can compute $P(X \leq x)$:
   `pbinom(x,size=n,p=prob)`
   Note that you have to compute $P(X = x)$!
   Based on what everyone finds, write down the number of cases we have of each possible outcome, along with the theoretical probability.
3. Given $X \sim f(\cdot)$, where $f(\cdot)$ is (a) $Unif(0,10)$, (b) $N(\mu = 100, \sigma^2 = 20)$, (c) $Binom(p = .6, n = 20)$. Find the probability of $P(X < 3), P(X > 11), P(X = 6)$ for each distribution.
4. Plot the priors, likelihoods, and posterior distributions in the four Beta-Binomial cases discussed in the notes.

3 Linear modeling

In this chapter, we will fit linear models of the following type. Suppose y is a vector of continuous responses; assume for now that it is coming from a normal distribution:

$y \sim Normal(\mu, \sigma)$

This is the simple linear model:

$y = \mu + \varepsilon$ where $\varepsilon \sim Normal(0, \sigma)$

There are two parameters, μ, σ, so we need priors on these. We expand on this simple model next.

Recall from the foundations chapter that the way we will conduct data analysis is as follows.

- Given data, specify a *likelihood function*.
- Specify *prior distributions* for model parameters.
- Evaluate whether model makes sense, using fake-data simulation, *prior predictive* and *posterior predictive* checks, and (if you want to claim a discovery) calibrating true and false discovery rates.
- Using software, derive *marginal posterior distributions* for parameters given likelihood function and prior density. I.e., simulate parameters to get *samples from posterior distributions* of parameters using some *Markov Chain Monte Carlo (MCMC) sampling algorithm*.
- Check that the model converged using *model convergence* diagnostics,
- Summarize *posterior distributions* of parameter samples and make your scientific decision.

We will now work through some specific examples to illustrate how the data analysis process works.

3.1 Example 1: A single subject pressing a button repeatedly

As a first example, we will fit a simple linear model to some reaction time data.

The file `button_press.dat` contains data of a subject pressing the space bar without reading in a self-paced reading experiment.

3.1.1 Preprocessing of the data

```
noreading_data <- read.table(header = F, "data/button_press.dat",
    encoding = "latin1")
noreading_data <- noreading_data[c("V2", "V3", "V5", "V6", "V8")]
colnames(noreading_data) <- c("type", "item", "wordn", "word",
    "y")
tail(noreading_data)
```

```
##       type item wordn          word   y
## 356 filler    3     0    Vielleicht 214
```

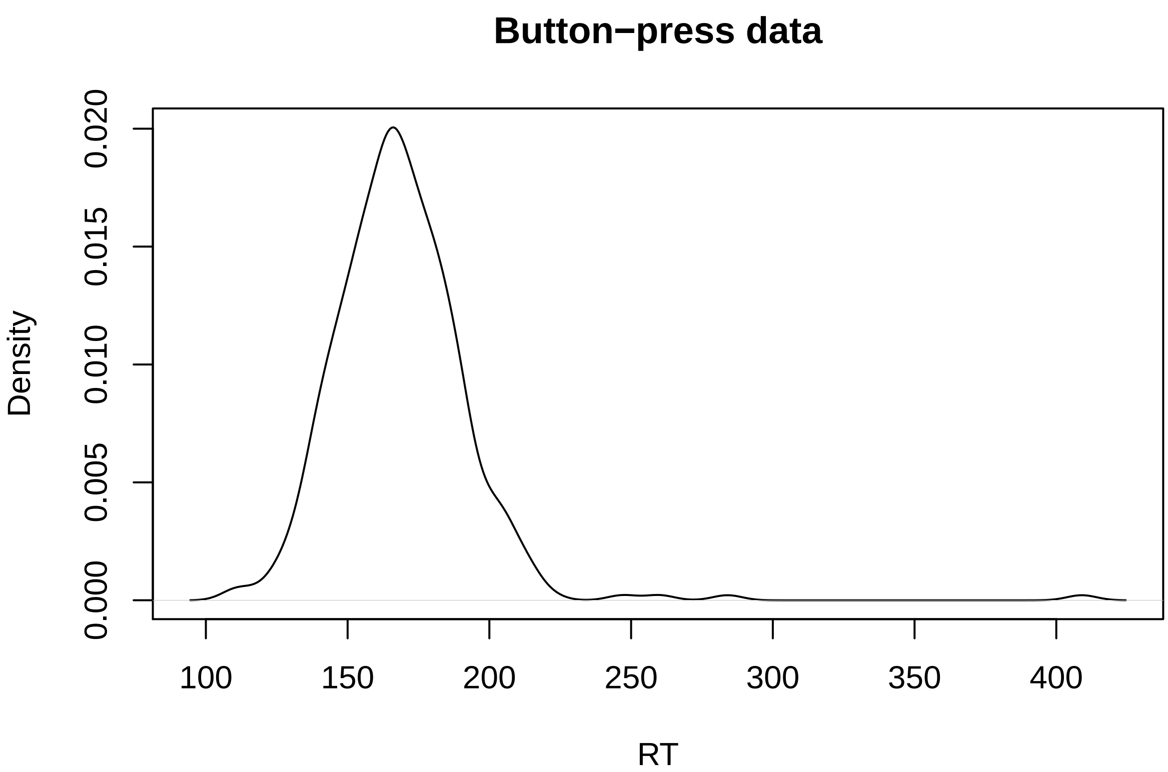


Figure 19: Visualizing the data.

```
## 357 filler    3     1          haben 182
## 358 filler    3     2 die_Zahnärztin 179
## 359 filler    3     3        aus_Bonn 177
## 360 filler    3     4  die_Patienten 183
## 361 filler    3     5       verklagt. 162
```

```
summary(noreading_data$y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     110     156     166     169     181     409
```

```
class(noreading_data)
```

```
## [1] "data.frame"
```

3.1.2 Visualizing the data

It is a good idea to look at the distribution of the data before doing anything else. See Figure 19.

```
plot(density(noreading_data$y), main = "Button-press data",
    xlab = "RT")
```

The data looks a bit skewed, but we ignore this for the moment.

3.1.3 Define the likelihood function

Let's model the data with the following assumptions:

- There is a true underlying time, μ, that the participant needs to press the space-bar.
- There is some noise in this process.
- The noise is normally distributed (this assumption is questionable given the skew but; we fix this assumption later).

This means that the likelihood for each observation i will be:

$$y_i \sim Normal(\mu, \sigma) \tag{168}$$

where $i = 1 \dots N$.

This is just the simple linear model:

$$y = \mu + \varepsilon \text{ where } \varepsilon \sim Normal(0, \sigma) \tag{169}$$

3.1.4 Define the priors for the parameters

We are going to use the following priors for the two parameters in this model:

$$\begin{aligned} \mu &\sim Normal(0, 2000) \\ \sigma &\sim Normal(0, 500) \text{ truncated so that } \sigma > 0 \end{aligned} \tag{170}$$

In order to decide on a prior for the parameters, always visualize them first. See Figure 20.

```
op <- par(mfrow = c(1, 2), pty = "s")
x <- seq(-4000, 4000, by = 1)
plot(x, dnorm(x, mean = 0, sd = 2000), type = "l", main = expression(paste("Prior for "
    mu)), xlab = expression(mu))
x <- seq(0, 4000, by = 1)
plot(x, dnorm(x, mean = 0, sd = 500), type = "l", main = expression(paste("Prior for ",
    sigma)), xlab = expression(sigma))
```

The prior for μ expresses the belief that the underlying mean button-pressing time could be both positive and negative, and given that the scale of the prior (in this case the standard deviation of the normal distribution) is 2000, we are $\approx 68\%$ certain that the true value would be between 2000 ms and -2000 ms and $\approx 95\%$ certain that it would be between -4000 ms and 4000 ms (two standard deviations away from zero). We know this because:

```
pnorm(2000, mean = 0, sd = 2000) - pnorm(-2000, mean = 0, sd = 2000)
```

```
## [1] 0.683
```

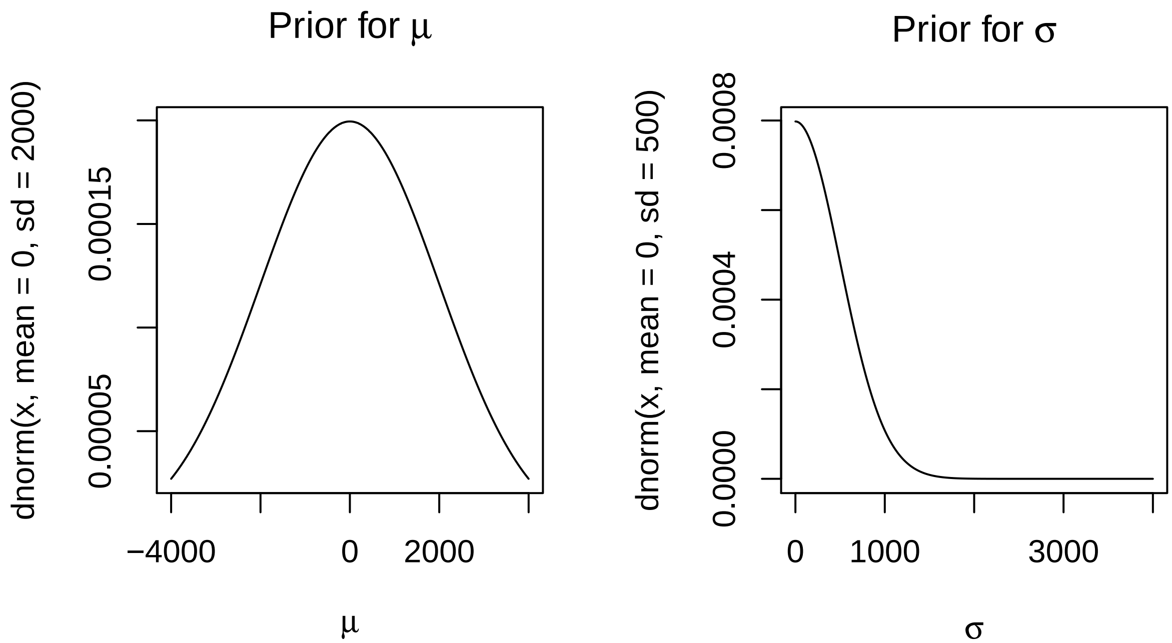


Figure 20: Visualizing the priors for example 1.

```
pnorm(4000, mean = 0, sd = 2000) - pnorm(-4000, mean = 0, sd = 2000)
```

```
## [1] 0.954
```

But we obviously know that button-pressing time can't be negative! So we have more prior information than what we are using for informing the model. We'll discuss this below.

Regarding the prior for σ: The values must be positive, and we are $\approx 68\%$ certain that the true value would be between 0 ms and 500 ms and $\approx 95\%$ certain that it would be between 0 ms and 1000 ms. **How would you check this using pnorm?**

3.1.5 Prior predictive checks

With these priors, we are going to generate something called the **prior predictive distribution**. This helps us check whether the priors make sense.

Formally, we want to know the density $f(\cdot)$ of data points $y_1,\dots,n$, given a vector of priors Θ. In our example, $\Theta = \langle \mu, \sigma \rangle$. The prior predictive density is:

$$f(y_1,\dots,y_n) = \int f(y_1)\cdot f(y_2)\cdots f(y_n) f(\Theta)\, d\Theta \tag{171}$$

In essence, we integrate out the parameters. Here is one way to do it in R:

- Take a sample from each of the priors
- Generate data using those samples

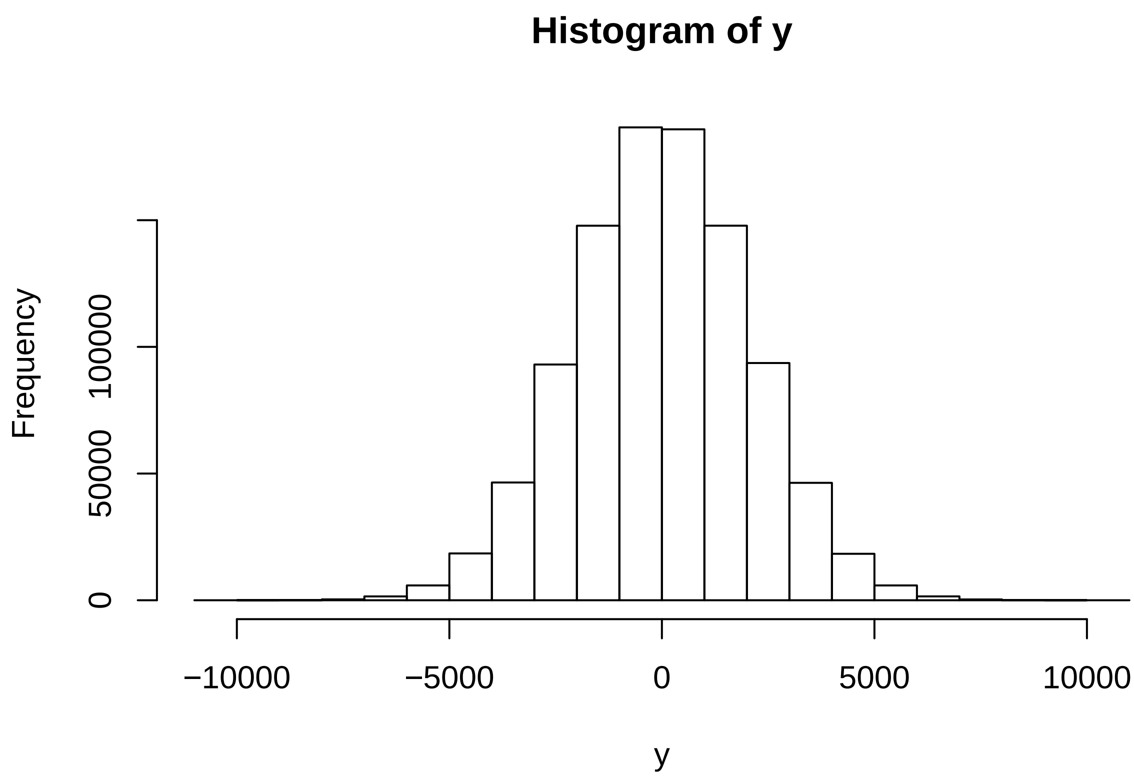


Figure 21: First attempt at prior predictive distribution of the data, model m1.

- Repeat until you have some data
- Plot the prior predictive density

```
nsim <- 1000000
y <- rep(NA, nsim)
mu <- rnorm(nsim, mean = 0, sd = 2000)
sigma <- abs(rnorm(nsim, mean = 0, sd = 500))

for (i in 1:nsim) {
    y[i] <- rnorm(1, mean = mu[i], sd = sigma[i])
}

hist(y)
```

This prior predictive distribution in Figure 21 shows that the prior for μ is not realistic: how can button press time have negative values?

We can try to redefine the prior for μ to have only positive values, and then check again (Figure 22). We still get some negative values, but that is because we are assuming that

$y \sim Normal(\mu, \sigma)$

which will have negative values for small μ and large σ.

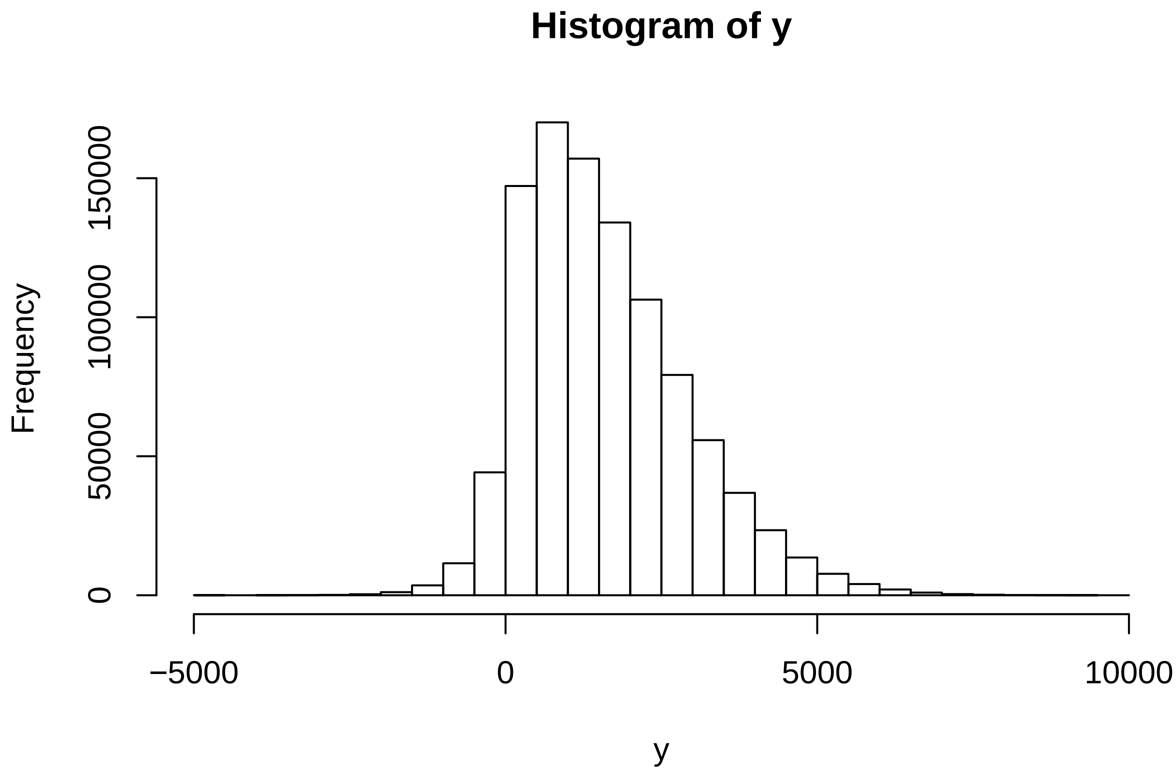


Figure 22: Second attempt at prior predictive distribution of the data, model m1.

```
nsim <- 1000000
y <- rep(NA, nsim)
mu <- abs(rnorm(nsim, mean = 0, sd = 2000))
sigma <- abs(rnorm(nsim, mean = 0, sd = 500))

for (i in 1:nsim) {
    y[i] <- rnorm(1, mean = mu[i], sd = sigma[i])
}

hist(y)
```

This prior predictive distribution in Figure 22 looks reasonable for now.

Incidentally, we can generate a prior predictive distribution using Stan as follows.

First, we define a Stan model that defines the priors and defines how the data are to be generated. The details of the code below will be explained in class. Documentation on Stan is available at mc-stan.org.

```
priorpred <- "data {
  int N;
}
parameters {
```

```
real<lower=0> mu;
real<lower=0> sigma;
}
model {
  // priors
\tmu ~ normal(0,2000);
\tsigma ~ normal(0,500);
}
generated quantities {
  vector[N] y_sim;
  // prior predictive
  for(i in 1:N) {
    y_sim[i] = normal_rng(mu,sigma);
  }
}
"
```

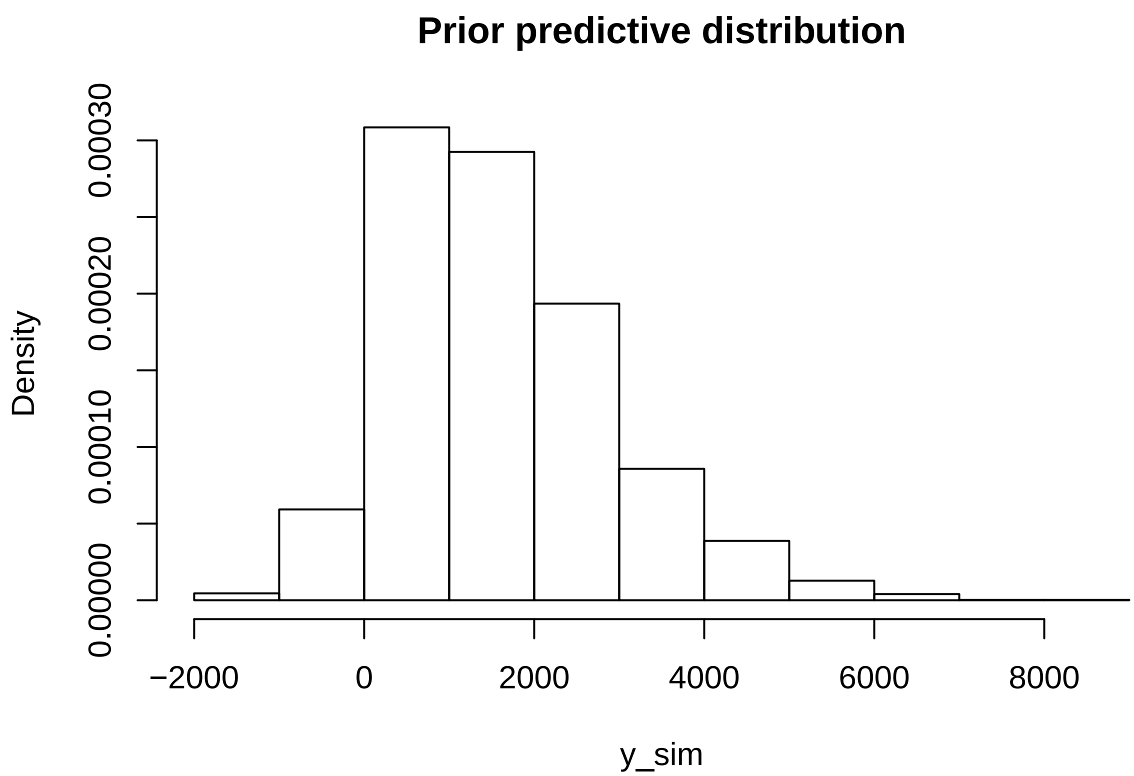
Then we generate the data:

```
## load rstan
library(rstan)
options(mc.cores = parallel::detectCores())

## generate 100 data sets
dat <- list(N = 100)

## fit model:
m1priorpred <- stan(model_code = priorpred, data = dat, chains = 4,
    iter = 2000)

## extract and plot one of the data-sets:
y_sim <- extract(m1priorpred, pars = "y_sim")
hist(y_sim$y_sim[, 1], main = "Prior predictive distribution",
    xlab = "y_sim", freq = FALSE)
```



Having satisfied outselves that the priors mostly make sense, we now fit the model to fake data. The goal here is to ensure that the model recovers the true underlying parameters.

3.1.6 Fake-data simulation and modeling

Next, we write the Stan model, adding a likelihood in the model block:

```
m1 <- "data {
  int N;
  real y[N]; // data
}
parameters {
real<lower=0> mu;
real<lower=0> sigma;
}
model {
// priors
mu ~ normal(0,2000);
sigma ~ normal(0,500);
// likelihood
y ~ normal(mu,sigma);
}
generated quantities {
  vector[N] y_sim;
```

```
 // posterior predictive
  for(i in 1:N) {
    y_sim[i] = normal_rng(mu,sigma);
  }
}
"
```

Then generate fake data with known parameter values (we decide what these are):

```
set.seed(123)
N <- 500
true_mu <- 400
true_sigma <- 125
y <- rnorm(N, true_mu, true_sigma)

y <- round(y)
fake_data <- data.frame(y = y)
dat <- list(y = y, N = N)
```

Finally, we fit the model:

```
## fit model:
m1rstan <- stan(model_code = m1, data = dat, chains = 4, iter = 2000)

## extract posteriors:
posteriors <- extract(m1rstan, pars = c("mu", "sigma"))
```

Figure 23 shows that the true parameters that we defined when generating the fake data fall within the posterior distributions. This shows that the model can in principle recover the parameters. (One should do several fake data simulations to check that the model consistently recovers the true parameters.)

```
op <- par(mfrow = c(1, 2), pty = "s")
hist(posteriors$mu, main = expression(paste("posterior for ",
    mu)), freq = FALSE, xlab = "")
abline(v = 400)
hist(posteriors$sigma, main = expression(paste("posterior for ",
    sigma)), freq = FALSE, xlab = "")
abline(v = 125)
```

3.1.7 Posterior predictive checks

Once we have the posterior distribution $f(\Theta \mid y)$, we can derive the predictions based on this posterior distribution:

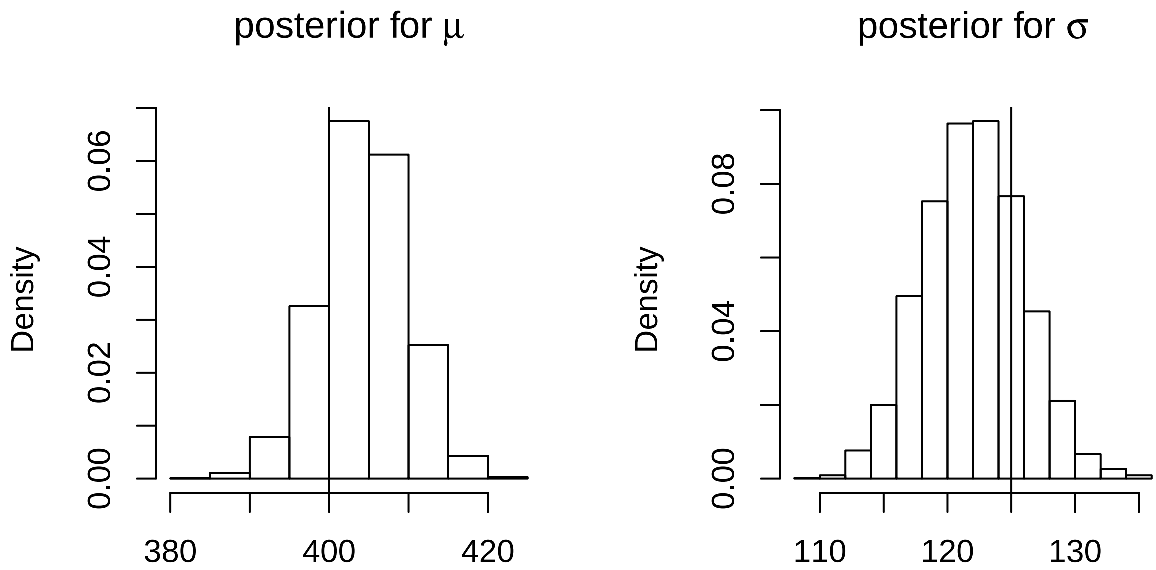


Figure 23: Posteriors from fake data, model m1. Vertical lines show the true values of the parameters.

$$p(y_{pred} \mid y) = \int p(y_{pred}, \Theta \mid y)\, d\Theta = \int p(y_{pred} \mid \Theta, y) p(\Theta \mid y)\, d\Theta \tag{172}$$

Assuming that past and future observations are conditionally independent given Θ, i.e., $p(y_{pred} \mid \Theta, y) = p(y_{pred} \mid \Theta)$, we can write:

$$p(y_{pred} \mid y) = \int p(y_{pred} \mid \Theta) p(\Theta \mid y)\, d\Theta \tag{173}$$

Note that we are conditioning y_{pred} only on y, we do not condition on what we don't know (Θ); **we integrate out the unknown parameters**.

This posterior predictive distribution is different from the frequentist approach, which gives only a predictive distribution of y_{pred} given our estimate of θ (a point value).

In the Stan code above, we have already generated the posterior predictive distribution, in the generated quantities block:

```
generated quantities {
  vector[N] y_sim;
 // posterior predictive
  for(i in 1:N) {
    y_sim[i] = normal_rng(mu,sigma);
  }
}
```

3.1.8 Implementing model in brms

An alternative to using rstan as we did above is the package brms. The advantage with using brms is that many of the details of model-specification are hidden from the user; the price paid is loss of transparency, and reduced flexibility in modeling. brms is a good software for fitting canned models, but for customized models you will always need Stan, so it is good to know both syntaxes. I personally do all my research using Stan almost exclusively.

First, load the brms package; this packge runs Stan (Stan Development Team 2017) in the background. For an introduction to this package, see Bürkner (2017), and https://github.com/paul-buerkner/brms.

```
library(brms)
```

This model is expressed in brms in the following way. First, define the priors:

```
priors <- c(set_prior("normal(0, 2000)", class = "Intercept"),
    set_prior("normal(0, 500)", class = "sigma"))
```

Then, define the generative process assumed:

```
m1brms <- brm(y ~ 1, noreading_data, prior = priors, iter = 2000,
    warmup = 1000, chains = 4, family = gaussian(), control = list(adapt_delta = 0.99))
```

1. The term `family = gaussian()` makes explicit the underlying likelihood function that is implicit in `lme4`. Other linking functions are possible, exactly as in the `glmer` function in `lme4`.
2. The term `prior` takes as argument the list of priors we defined earlier. Although this specification of priors is optional, the researcher should always explicitly specify each prior. Otherwise, `brms` will define a prior by default, which may or may not be appropriate.
3. The term `iter` refers to the number of iterations that the sampler makes to sample from the posterior distribution of each parameter (by default 2000). See the discussion on HMC earlier.
4. The term `warmup` refers to the number of iterations from the start of sampling that are eventually discarded (by default half of `iter`).
5. The term `chains` refers to the number of independent runs for sampling (by default four).
6. The term `control` refers to some optional control parameters for the sampler, such as `adapt_delta`, `max_treedepth`, and so forth, to be discussed later.

3.1.9 Summarizing the posteriors, and convergence diagnostics

The summary displayed below show summary statistics over the marginal posterior distributions of the parameters in the model. The summary shows posterior means, standard deviations (`sd`), quantiles, Monte Carlo standard errors (`se_mean`), split Rhats, and effective sample sizes (`n_eff`).

The summaries are computed after removing the warmup and merging together all chains. Notice that the se_mean is unrelated to the se of an estimate in the frequentist model, and we will ignore it in this course.

```
summary(m1brms)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: y ~ 1
##    Data: noreading_data (Number of observations: 361)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##     ICs: LOO = NA; WAIC = NA; R2 = NA
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept   168.63      1.31   166.01   171.25       2150 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma    25.04      0.95    23.25    26.97       1868 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

A graphical summary of posterior distributions of model m1 is shown in Figure 24:

```
stanplot(m1brms, type = "hist")
```

The trace plots in Figure 25 show how well the four chains are mixing:

```
stanplot(m1brms, type = "trace")
```

An alternative way to plot is shown in Figure **??**

```
plot(m1brms)
```

3.1.10 MCMC diagnostics: Convergence problems and Stan warnings

Because we are using MCMC methods to sample from the posterior distributions, we need to make sure that the model has converged.

The most important checks or MCMC diagnostics are the following:

- The chains should look like a straight "fat hairy caterpillar": the chains should bounce around the same values and with the same variance.

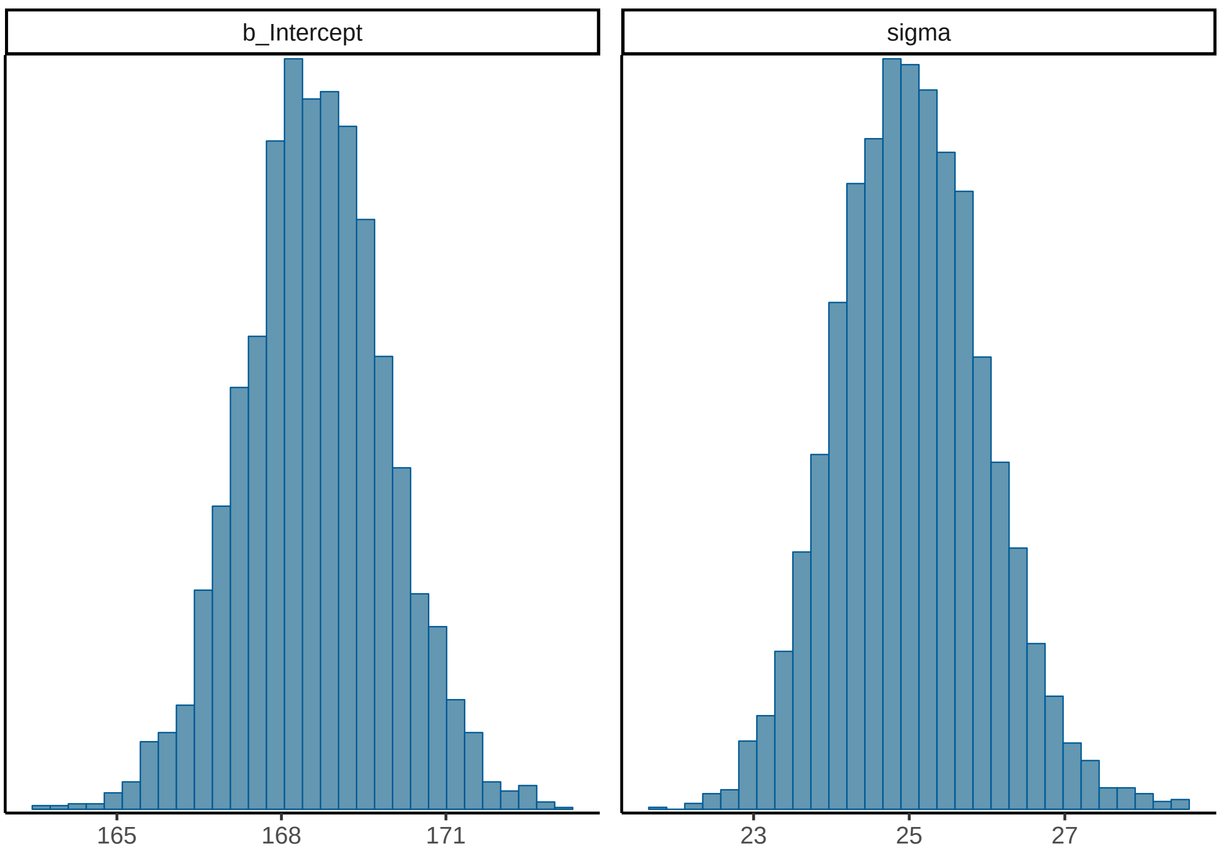


Figure 24: Posterior distributions of the parameters in model m1.

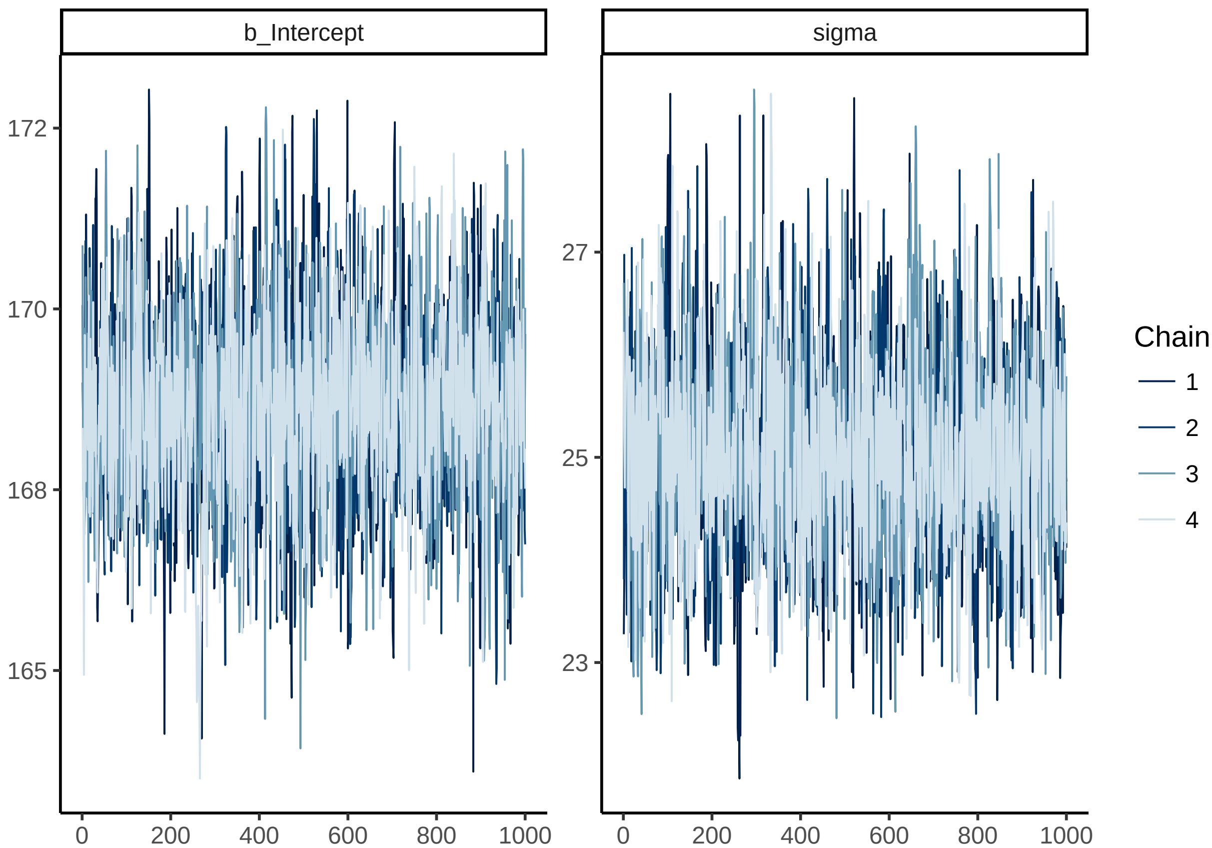


Figure 25: Trace plots in model m1.

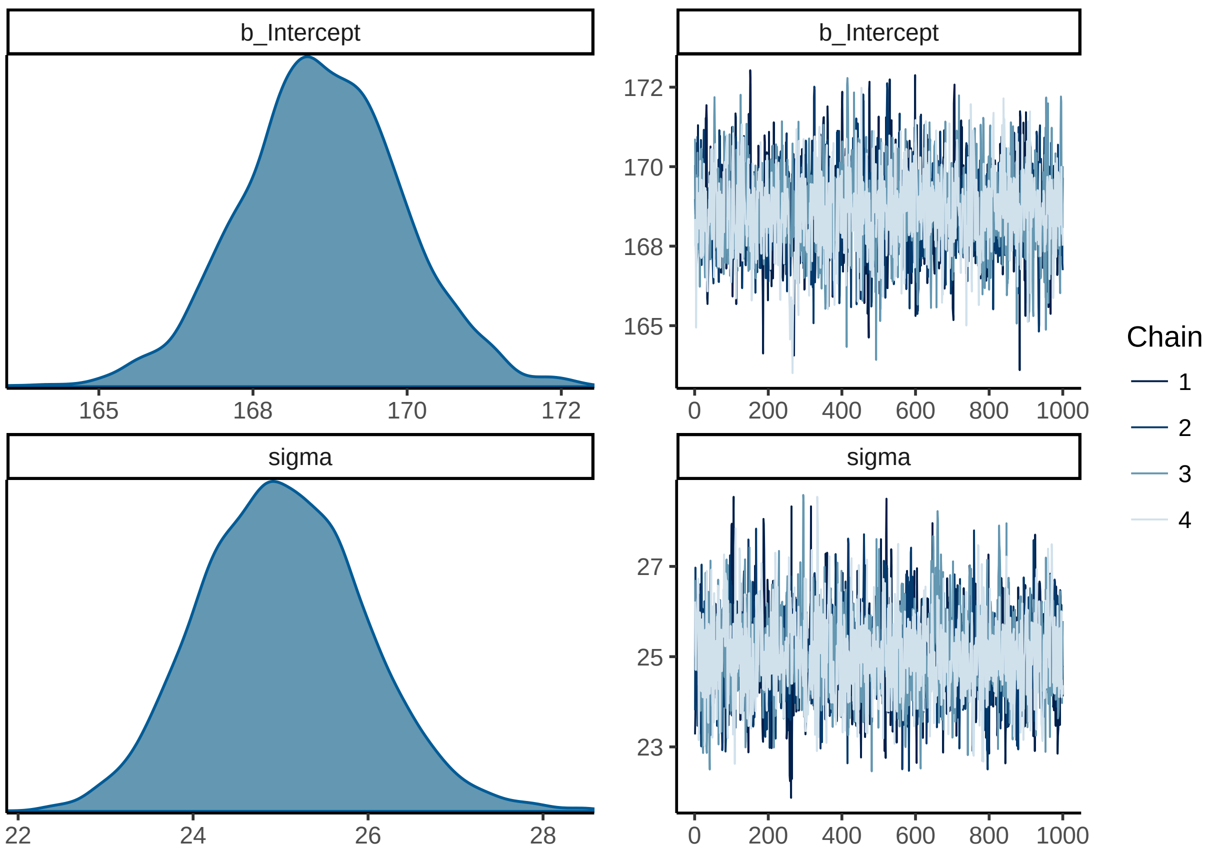


Figure 26: Posterior distributions and trace plots in model m1.

- The R-hat statistic, $\hat{R}$s, of the parameters should be close to one (as a rule of thumb less than 1.1). This indicates that the chains have mixed and they are traversing the same distribution. $\hat{R}$s are printed in the summary in the column `Rhat` (see section 11.4 of Gelman et al. 2014). These are the ratio of between to within chain variance.
- The effective sample size, n_{eff} should be large enough. The effective sample size is an estimate of the number of independent draws from the posterior distribution. Since the samples are not independent, n_{eff} will generally be smaller than the total number of samples, N. How large n_{eff} should be depends on the summary statistics that we want to use. But as a rule of thumb, $n_{eff}/N > 0.1$.
- Check that the software does not warnings such as divergent transitions, Bayesian fraction of missing information (BFMI) that was too low, etc. These warning may indicate that the sampler is not adequately exploring the parameter space. If you see these warnings, consult http://mc-stan.org/misc/warnings.html

For useful graphical checks see https://cran.r-project.org/web/packages/bayesplot/vignettes/MCMC-diagnostics.html

These issues **should not be ignored**! See the Appendix 3.5 for some troubleshooting ideas to solve them.

3.1.11 Summarizing the posterior distribution: posterior probabilities and the credible interval

We are assuming that there's a true underlying time it takes to press the space bar, μ, and there is normally distributed noise with distribution Normal(0,σ) that generates the different RTs. All this is encoded in our likelihood by assuming that RTs are distributed with an unknown true mean μ (and an unknown standard deviation σ).

The objective of the Bayesian model is to learn about the plausible values of μ, or in other words, to get a distribution that encodes what we know about the true mean of the distribution of RTs, and about the true standard deviation, σ, of the distribution of RTs.

Our model allows us to answer questions such as:

What is the probability that the underlying value of the mindless press of the space bar would be over, say 170 ms?

As an example, consider this model that we ran above:

```
priors <- c(set_prior("normal(0, 2000)", class = "Intercept"),
    set_prior("normal(0, 500)", class = "sigma"))

m1brms <- brm(y ~ 1, noreading_data, prior = priors, iter = 2000,
    warmup = 1000, chains = 4, family = gaussian(), control = list(adapt_delta = 0.99))
```

```
## Compiling the C++ model
```

```
## Start sampling
```

We now compute the posterior probability $Prob(\mu > 170)$:

```
mu_post <- posterior_samples(m1brms, pars = c("b_Intercept"))$b_Intercept
mean(mu_post > 170)
```

```
## [1] 0.145
```

The credible interval

The 95% credible interval can be extracted for μ as follows:

```
posterior_interval(m1brms, pars = c("b_Intercept"))
```

```
##              2.5% 97.5%
## b_Intercept  166   171
```

This type of interval is also known as a *credible interval*. A credible interval demarcates the range within which we can be certain with a certain probability that the "true value" of a parameter lies given the data and the model. This is very different from the frequentist confidence interval! See for discussion, Hoekstra et al. (2014) and Morey et al. (2016).

The percentile interval is a type of credible interval (the most common one), where we assign equal probability mass to each tail. We generally report 95% credible intervals. But we can extract any interval, a 73% interval, for example, leaves 13.5% of the probability mass on each tail, and we can calculate it like this:

```
quantile(mu_post, prob = c(0.135, 0.865))
```

```
## 13.5% 86.5%
##   167   170
```

3.1.11.1 Influence of priors and sensitivity analysis

Everything was normally distributed in our example (or truncated normal), but the fact that we assumed that RTs were normally distributed is completely unrelated to our (truncated) normally distributed priors. Let's try a uniform prior on μ with a low boundary of 0 and a high boundary of 5000. Here, we assume that every value between 0 and 5000 is equally likely. In general, this is a bad idea for two reasons: (i) it is computationally expensive (the sampler has a larger parameter space to search), and (ii) it is providing information that we know is not sensible (every value between 0 and 5000 cannot be equally likely). But in our very simple example these priors will give use the same posterior as with the normal priors.

$$\begin{aligned} \mu &\sim Uniform(0,5000) \\ \sigma &\sim Uniform(0,5000) \end{aligned} \tag{174}$$

```
priors <- c(set_prior("uniform(0, 5000)", class = "Intercept"),
    set_prior("normal(0, 500)", class = "sigma"))
```

```r
m2 <- brm(y ~ 1, noreading_data, prior = priors, iter = 2000,
    chains = 4, family = gaussian(), control = list(adapt_delta = 0.99))
```

```
## Compiling the C++ model
## Start sampling
```

```r
summary(m2)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: y ~ 1
##    Data: noreading_data (Number of observations: 361)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##     ICs: LOO = NA; WAIC = NA; R2 = NA
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept   168.62      1.28   166.16   171.13       2361 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma    24.98      0.95    23.16    26.89       2306 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

In general, we don't want our priors to have too much influence on our posterior. This is unless we have *very* good reasons for having informative priors, such as a very small sample and a lot of prior information; an example would be if we have data from an impaired population, which makes it hard to increase our sample size.

We usually center the priors on 0 and we let the likelihood dominate in determining the posterior. This type of prior is called *weakly informative prior*. Notice that a uniform prior is not a weakly informative prior, it assumes that every value is equally likely, zero is as likely as 5000.

You should always do a *sensitivity analysis* to check how influential the prior is: try different priors and verify that the posterior doesn't change drastically (for a published example, see Vasishth et al. 2013). See also Exercise 1a.

3.2 Example 2: Investigating adaptation effects

More realistically, we might have run the small experiment to find out whether the participant tended to speedup (practice effect) or slowdown (fatigue effect) while pressing the space bar.

3.2.1 Preprocessing the data

We need to have data about the number of times the space bar was pressed for each observation, and add it to our list. It's a good idea to center the number of presses (a covariate) to have a clearer interpretation of the intercept. In general, centering predictors is always a good idea, for interpretability and for computational reasons.

```
# We create the new column in the data frame
noreading_data$presses <- 1:nrow(noreading_data)
# We center the column
noreading_data$c_presses <- noreading_data$presses - mean(noreading_data$presses)
```

3.2.2 Probability model

Our model changes, because we have a new parameter.

$$y_i \sim Normal(\alpha + presses_i \cdot \beta, \sigma) \tag{175}$$

where $i = 1 \dots N$

And we are going to use the following priors:

$$\begin{aligned} &\alpha \sim Normal(0, 2000) \\ &\beta \sim Normal(0, 500) \\ &\sigma \sim Normal(0, 500) \text{ truncated so that } \sigma > 0 \end{aligned} \tag{176}$$

We are basically fitting a linear model, α represents the intercept (namely, the grand mean of the RTs), and β represents the slope. What information are the priors encoding? Do the priors make sense?

We'll write this in brms as follows.

```
priors <- c(set_prior("normal(0, 2000)", class = "Intercept"),
    set_prior("normal(0, 500)", class = "b", coef = "presses"),
    set_prior("normal(0, 500)", class = "sigma"))
```

```
m2 <- brm(y ~ 1 + presses, noreading_data, prior = priors, iter = 2000,
    chains = 4, family = gaussian(), control = list(adapt_delta = 0.99))
```

```
## Compiling the C++ model
```

```
## Start sampling
```

```
summary(m2)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: y ~ 1 + presses
##    Data: noreading_data (Number of observations: 361)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##     ICs: LOO = NA; WAIC = NA; R2 = NA
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept   152.50      2.46   147.61   157.33       3399 1.00
## presses       0.09      0.01     0.07     0.11       3273 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma    23.23      0.86    21.67    25.00       3106 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

3.2.3 Summarizing the posterior and inference

How can we answer our research question? What is the effect of pressing the bar on the participant's reaction time?

We'll need to examine what happens with β. The summary gives us the relevant information:

```
m2_post_samp_b <- posterior_samples(m2, "^b")
beta_samples <- m2_post_samp_b$b_presses
beta_mean <- mean(beta_samples)
quantiles_beta <- quantile(beta_samples, prob = c(0.025, 0.975))
beta_low <- quantiles_beta[1]
beta_high <- quantiles_beta[2]
```

We learn that the most likely values of β will be around the mean of the posterior 0.089, and we can be 95% certain that the true value of β *given the model and the data* lies between 0.066 and 0.111.

We see that as the number of times the space bar is pressed increases, the participant becomes slower. If we want to determine how likely it is that the participant was slower rather than faster, we can examine the proportion of samples above zero:

```
mean(beta_samples > 0)
```

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

We would report this in a paper as $\hat{\beta} = 0.089$, 95% CrI = $[0.066, 0.111]$, $P(\beta > 0) \approx 1$. Plotting the posterior as a histogram is always a good idea.

Can we really conclude that there is a fatigue effect? It depends on how much we expect the fatigue to affect the RTs. Here we see that only after 100 button presses do we see a slowdown of 9 ms on average (0.09×100).

We need to consider whether the size of this effect has any scientific relevance by considering the previous literature. Sometimes this requires a meta-analysis. See Jäger, Engelmann, and Vasishth (2017), Nicenboim, Roettger, and Vasishth (2018) for examples. For examples of the use of this prior knowledge, see Nicenboim et al. (Submitted).

3.2.4 Posterior predictive checks

Let's say we know that our model is working as expected, since we already used fake data to test the recovery of the parameters (this will be a homework assignment).

We will now examine the *descriptive adequacy* of the models (Shiffrin et al. 2008; Gelman et al. 2014, Chapter 6): the observed data should look plausible under the *posterior predictive distribution*, as discussed above. The posterior predictive distribution is composed of one dataset for each sample from the posterior. (So it will generate as many datasets as iterations we have after the warm-up.) Achieving descriptive adequacy means that the current data could have been predicted by the model. Passing a test of descriptive adequacy is not strong evidence in favor of a model, but a major failure in descriptive adequacy can be interpreted as strong evidence against a model (Shiffrin et al. 2008).

To do posterior predictive checks for our last example, using brms, we need to do:

```
pp_check(m2, nsamples = 100) + theme(text = element_text(size = 16),
    legend.text = element_text(size = 16))
```

We'll use the values generated by our model to verify whether the general shape of the actual distribution matches the distributions from some of the generated datasets. Let's compare the real data against 100 of the predicted 4000 datasets.

Are the posterior predicted data similar to the real data?

Figure 27 shows that our dataset seems to be more skewed to the right than our predicted datasets. This is not too surprising, we assumed that the likelihood was a normal distribution, but latencies are not very normal-like, they can't be negative and can be arbitrarily long.

3.2.5 Using the log-normal likelihood

Since we know that the latencies shouldn't be normally distributed, we can choose a more realistic distribution for the likelihood. A good candidate is the log-normal distribution since a variable (such as time) that is log-normally distributed takes only positive real values.

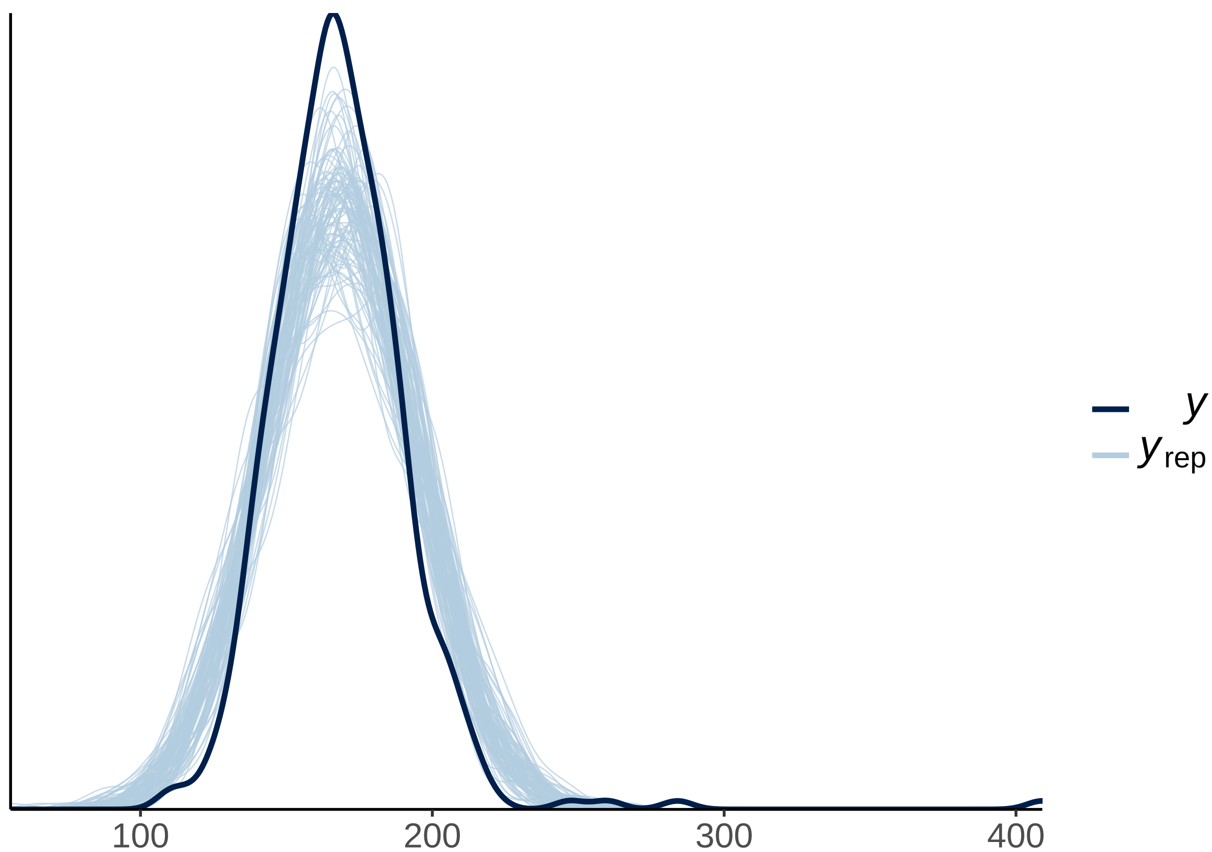


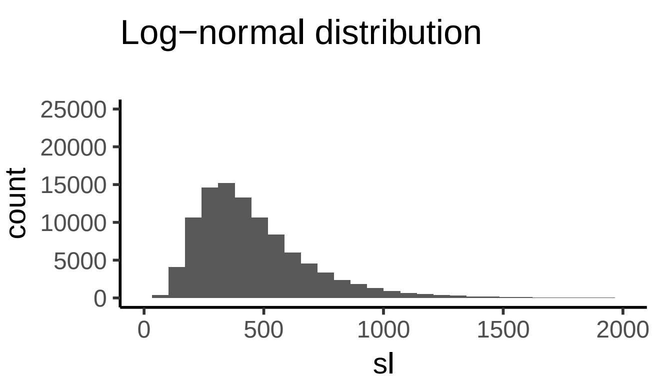
Figure 27: Posterior predictive check of model m2.

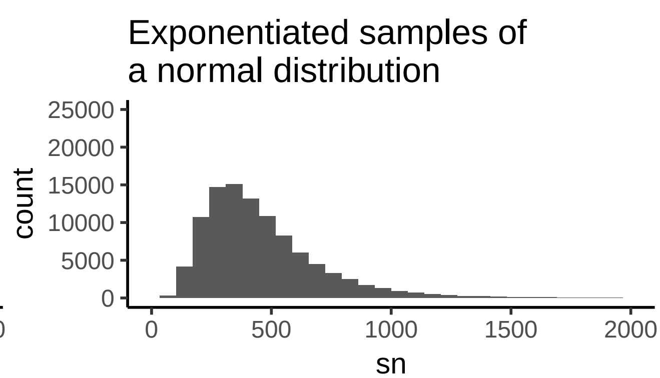
If Y is log-normally distributed, this means that $log(Y)$ is normally distributed.[2] Something important to notice is that the log-normal distribution is defined using again μ and σ, but this corresponds to the mean and standard deviation of the normally distributed logarithm $log(Y)$. Thus μ and σ are on a different scale than the variable that is log-normally distributed.

This also means that you can create a log-normal distribution by exponentiating the samples of a normal distribution. See Figure **??**.

```
mu <- 6
sigma <- 0.5
N <- 100000
# We generate N random samples from a log-normal
# distribution.
sl <- rlnorm(N, mu, sigma)
lognormal_plot <- ggplot(data.frame(samples = sl), aes(sl)) +
    geom_histogram() + ggtitle("Log-normal distribution\n") +
    ylim(0, 25000) + xlim(0, 2000)
# We generate N random samples from a normal distribution,
# and then we exponentiate them
sn <- exp(rnorm(N, mu, sigma))
normalplot <- ggplot(data.frame(samples = sn), aes(sn)) + geom_histogram() +
    ggtitle("Exponentiated samples of\na normal distribution") +
    ylim(0, 25000) + xlim(0, 2000)

plot(lognormal_plot)
plot(normalplot)
```





3.2.6 Re-fit the model assuming a log-normal likelihood

If we assume that RTs are log-normally distributed, we'll need to change our model:

$$Y_i \sim LogNormal(\alpha + presses_i \cdot \beta, \sigma) \tag{177}$$

[2] In fact, $log_e(Y)$ or $ln(Y)$, but we'll write it as just $log()$

where $i = 1 \ldots N$

But now the scale of our priors needs to change! They are no longer in milliseconds.

$$\begin{aligned}
\alpha &\sim Normal(0, 10) \\
\beta &\sim Normal(0, 1) \\
\sigma &\sim Normal(0, 2) \text{ truncated so that } \sigma > 0
\end{aligned} \tag{178}$$

The interpretation of the parameters changes and it is more complex than if we were dealing with a linear model that assumes a normal likelihood:

- α. In our previous linear model, α represented the grand mean (or the grand median since in a normal distribution both coincide), and was equivalent to our previous μ (since β was multiplied by 0). But now, the grand mean needs to be calculated in the following way, $\exp(\alpha + \sigma^2/2)$. Interestingly, the grand median will just be $exp(\alpha)$,[3] and we could assume that this represents the underlying time it takes to press the space bar if there would be no noise, that is, if σ had no effect. This also means that the prior of α is not in milliseconds, but in log(milliseconds).

- β. In a linear model, β represents the slowdown for each time the space bar is pressed. Now β is the effect on the log-scale, and the effect in milliseconds depends on the intercept α: $\exp(\alpha + \beta) - \exp(\alpha)$. Notice that the log is not linear and the effect of β will have more impact on milliseconds as the intercept grows. For example, if we start with (i) $\exp(5) = 148$, and we add 0.1 in log-scale, $\exp(5 + 0.1) = 164$, we end up with a difference of 15 ms; if we start with (ii) $\exp(6) = 400$, and we add 0.1, $\exp(6 + 0.1) = 445$, we end up with a difference of 45 ms. You can also see this graphically below, in Figure 28.

- σ. This is the standard deviation of the normal distribution of $log(y)$.

```
ms_diff <- function(Intercept) {
    exp(Intercept + 0.1) - exp(Intercept)
}
df <- tibble::data_frame(Intercept = seq(0.1, 15, 0.01), ms = ms_diff(Intercept))
ggplot(df, aes(x = Intercept, y = ms)) + geom_point() + scale_y_continuous("Difference
```

3.2.7 What kind of information are the priors encoding?

- For α: We are 95% certain that the grand median of the RTs will be between ≈ 0 and 485165195 milliseconds. This is a (very-)weakly regularizing prior because it won't affect our results, but it will down-weight values for the grand median of the RTs that are extremely large, and won't allow the grand median to be negative. We calculate the previous range by back-transforming the values that lie between two standard deviations of the prior (2×10) to millisecond scale: $exp(-10 \times 2)$ and $exp(10 \times 2)$).

[3] You can check in Wikipedia (https://en.wikipedia.org/wiki/Log-normal_distribution) why.

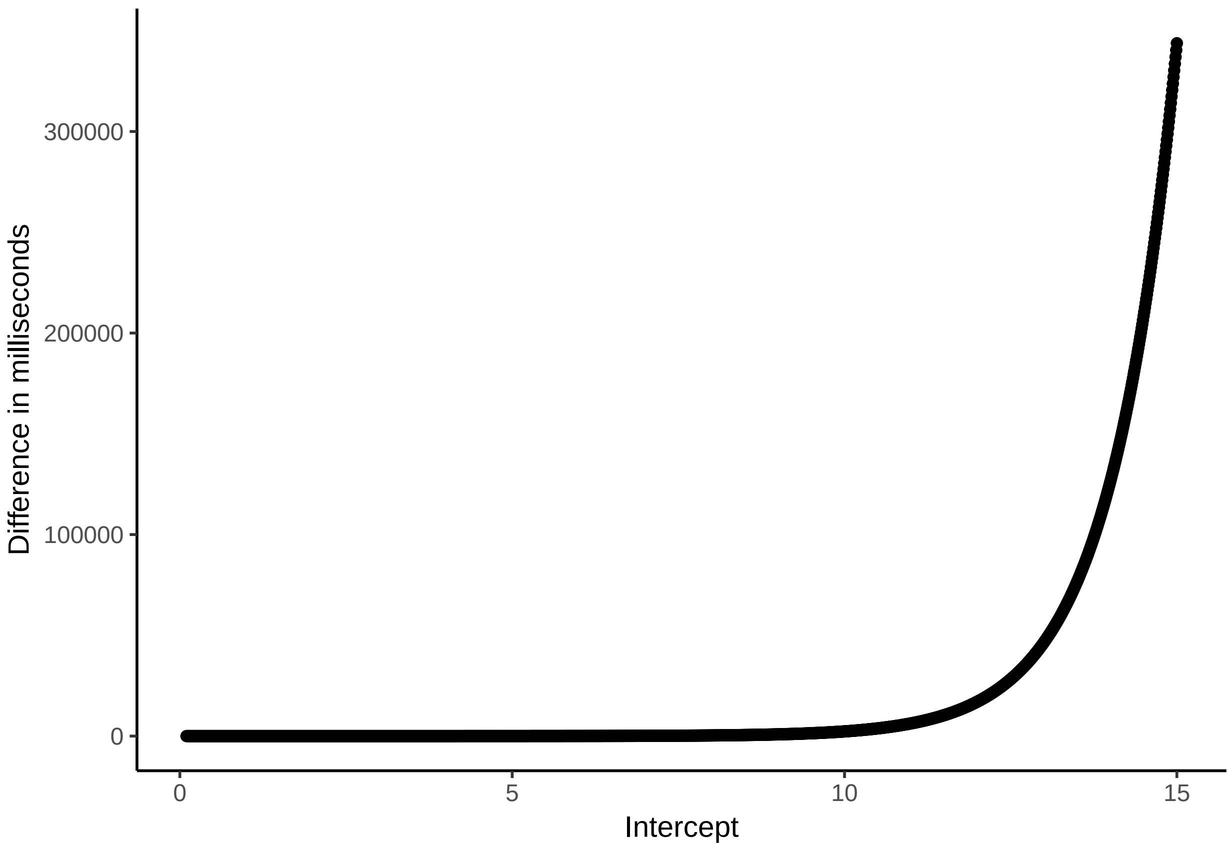


Figure 28: Back-transforming to milliseconds.

- For β: This is more complicated, because the effect on milliseconds will depend on the estimate of α. However, we can assume some value for α and it will be enough to be in the right order of magnitude. So let's assume 500 ms. That will mean that we are 95% certain that the effect of pressing the space bar will be between -491 and 26799 milliseconds. It is asymmetric because the log-scale is asymmetric. But the prior is weak enough so that if we assume 1000 or 100 instead of 500, the possible estimates of β will still be contained in the prior distribution. We calculate this by first finding out the value in milliseconds when we are two standard deviations away in both directions: (2×2), that is $\exp(500 - 2 \times 2)$ and $\exp(500 + 2 \times 2)$, and we subtract from that the value of α that we assumed, 500: $\exp(500 - 2 \times 2) - 500$ and $\exp(500 + 2 \times 2) - 500$.
- For σ. This indicates that we are 95% certain that the standard deviation of $log(y)$ will be between 0 and 2. So 95% of the RTs will be between $\exp(\log(500) - 1 \times 2) = 68$ and $exp(log(500) + 1 \times 2) = 3695$.

What happens if we replace 500 by 100, and by 1000? What happens if it is 10 instead? Does it still makes sense?

We'll code the model as follows.

```
priors_log <- c(set_prior("normal(0, 10)", class = "Intercept"),
    set_prior("normal(0, 1)", class = "b", coef = "presses"),
    set_prior("normal(0, 2)", class = "sigma"))
```

```
m2_logn <- brm(y ~ 1 + presses, noreading_data, prior = priors_log,
    iter = 2000, chains = 4, family = lognormal(), control = list(adapt_delta = 0.99,
        max_treedepth = 15))
```

```
## Compiling the C++ model
```

```
## Start sampling
```

```
summary(m2_logn)
```

```
##  Family: lognormal
##   Links: mu = identity; sigma = identity
## Formula: y ~ 1 + presses
##    Data: noreading_data (Number of observations: 361)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##     ICs: LOO = NA; WAIC = NA; R2 = NA
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept     5.02      0.01     5.00     5.05       2329 1.00
## presses       0.00      0.00     0.00     0.00       3258 1.00
##
## Family Specific Parameters:
```

```
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.12      0.00     0.12     0.13       1214 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

We fit the model, and check its convergence as usual (this will be a homework assignment).

3.2.8 Summarizing the posterior and inference

Next, we turn to the question of what we can report as our results, and what we can conclude from the data.

We can summarize the posterior and do inference as discussed in Example 1. If we want to talk about the effect estimated by the model, we summarize the posterior of β in the following way: $\hat{\beta} = 0.000523$, 95% CrI = $[0.0004, 0.00065]$, $P(\beta > 0) \approx 1$

But in most cases, the effect is easier to interpret in milliseconds. We generated the effect of 1 press in the generated quantities block, which is not the same as the linear model's β. Our generated estimate will tell us the estimate of the slowdown produced by pressing the space bar in the middle of the experiment once, assuming that the RTs are log-normally distributed: 0.07937 ms, 95% CrI = $[0.06184, 0.09603]$. Coincidentally, it is close to the same value as before, but this is not always the case, and since it's not linear the effect won't be the same across the whole experiment; see Exercise ?a.

3.2.9 Posterior predictive checks and distribution of summary statistics

We can now verify whether our predicted datasets look more similar to the real dataset. See Figure 29.

```r
pp_check(m2_logn, nsamples = 100) + theme(text = element_text(size = 16),
    legend.text = element_text(size = 16))
```

Are the posterior predicted data now more similar to the real data?

It seems so, but it's not easy to tell. Another way to examine this would be to look at the *distribution of summary statistics*. The idea is to compare the distribution of representative summary statistics for the datasets generated by different models and compare them to the observed statistics. Since we suspect that the log-normal distribution may capture the long tail, we could use the maximum as a summary statistics. We could generate 100 posterior predictive data-sets, and then compute the maximum each time and plot the distribution of the maximum, comparing it with the maximum value (409) in the data.

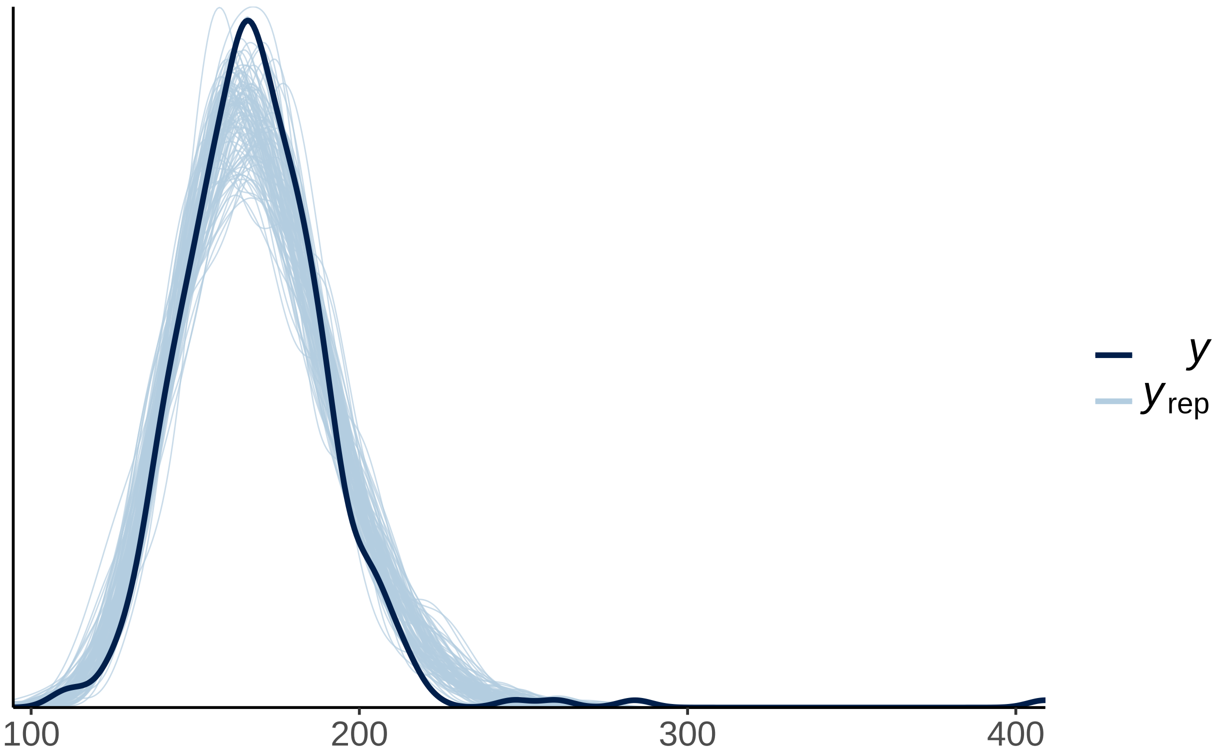


Figure 29: Posterior predictive check.

```
m2_logn <- brm(y ~ 1 + presses, noreading_data, prior = priors_log,
    iter = 2000, chains = 4, family = lognormal(), control = list(adapt_delta = 0.99,
        max_treedepth = 15))
```

```
## Compiling the C++ model
```

```
## Start sampling
```

```
postsamp <- posterior_samples(m2_logn, "^b", add_chain = TRUE)

nsim <- 100
maximum <- rep(NA, nsim)
for (i in 1:nsim) {
    ppm2_logn <- predict(m2_logn)
    maximum[i] <- max(as.vector(ppm2_logn))
}
```

```
hist(maximum, main = "Distribution of maximum values", freq = FALSE)
```

Figure 30 shows that we are unable to capture the maximum value of the observed data, so there's still room for improving the model. We will return to this question later in the course.

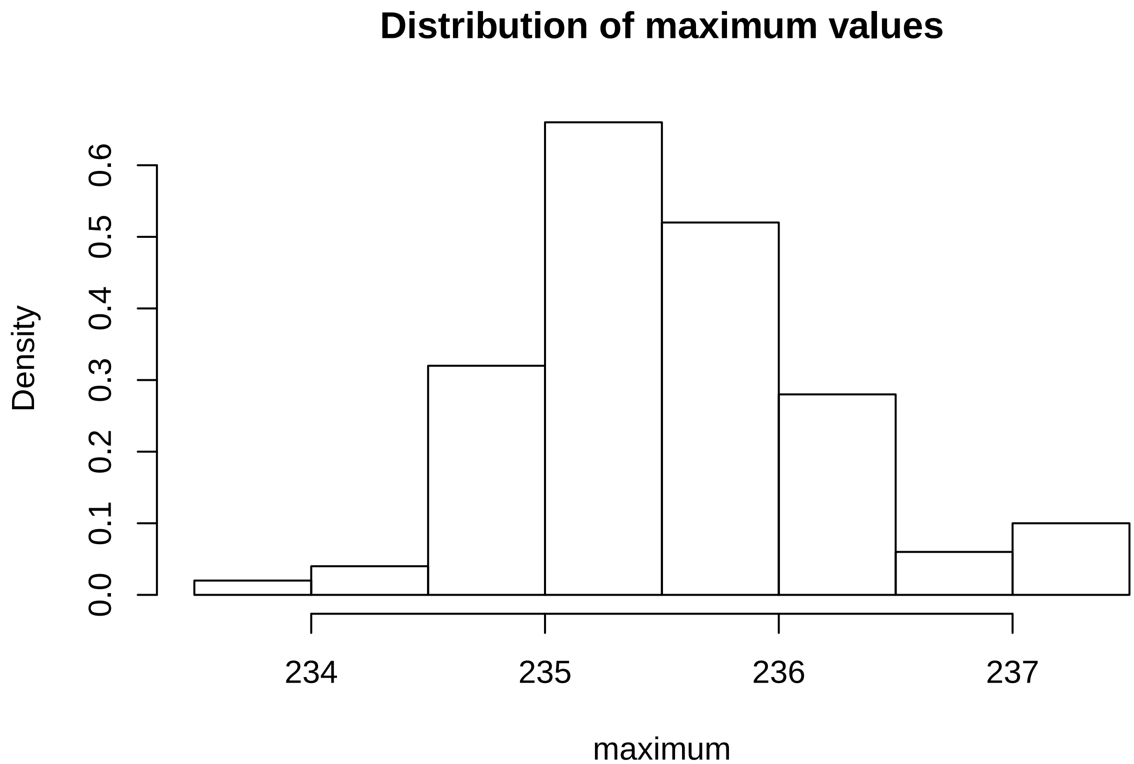


Figure 30: Distribution of maximum values in a posterior predictive check. The maximum in the data is 409 ms.

3.2.10 General workflow

This is the general workflow that we recommend for a Bayesian model.

1. Define the full probability model:
    a. Decide on the likelihood.
    b. Decide on the priors.
    c. Write the brms or Stan model.
2. Do prior predictive checks to determine if priors make sense.

3. Check model using fake data simulations:
    a. Simulate data with known values for the parameters.
    b. Fit the model and do MCMC diagnostics.
    c. Verify that it recovers the parameters from simulated data.
4. Fit the model with real data and do MCMC diagnostics.
5. Evaluate the model's fit (e.g., posterior predictive checks, distribution of summary statistics). This may send you back to 1.
6. Inference/prediction/decisions.
7. Conduct model comparison if there's an alternative model (to be discussed later).

3.3 Key concepts

- Basic brms and Stan syntax.
- How to identify convergence problems (MCMC diagnostics).
- Normal and log-normal distributions.
- Summaries of the posterior and inference.
- Model evaluation using prior and posterior predictive checks.
- General workflow for doing Bayesian analysis.

3.4 Exercises

1. For the model in Section 3.1.
   (a) Recall that when we use the Beta distribution as a prior for the θ parameter in a Binomial, the two parameters of the Beta indicate previous outcomes and not only the most likely value of θ. For example, $Beta(2,2)$ and $Beta(50,50)$ both indicate that .5 is the most likely value of θ, but $Beta(50,50)$ indicates that we are more certain about it than $Beta(2,2)$. Assume that you are quite certain that the average accuracy in the task is 80% for individuals with aphasia. How would you change the priors for the model in 3.1? Fit the model several times, assuming the same average accuracy as prior information, but varying the amount of uncertainty? When do the results change?
   (b) Change the data to `qresp_data <- list(c = 460, N = 1000)`, and repeat 1a with the same priors.
   (c) Simulate data assuming that the true θ is exactly .5 and (i) `N = 100`, (ii) `N = 1000` and run the `firstmodel.stan`. (Tip: use the function `rbinom` to simulate data.) Is the true value of θ inside 95% credible interval?
2. For the model in Section **??**
   (a) Change the priors of μ and σ in `noreading_1.stan` to a Cauchy and a truncated Cauchy distributions. A Cauchy distribution is basically a bell shaped distribution with very fat tails.[a] What are reasonable values for the location and scale? What's the difference between having a Cauchy or a normal distribution as priors?

Continues in the next page.

[a]You can read about this distribution in https://en.wikipedia.org/wiki/Cauchy_distribution.

3. For the models in Section 3.2.
   (a) Edit the `generated quantities` block in `noreading_log.stan` to estimate the slowdown in milliseconds (mean and 90% credible interval) for the last time the participant pressed the space bar in the experiment. In addition, predict the slowdown if the experiment would have had 500 observations.
   (b) Optional but very recommended: Simulate data that assumes a value of α of 400 and σ of 125 and a practice effect of (i) 0.00001 and (ii) 0.1 (both in log-scale). Fit these data with `noreading_log.stan`. Is the true value of the effect in the 95% credible interval in both cases?
   (c) Add word length as a covariate in `noreading_log.stan`, summarize the posterior. How does the length of the words affect RTs?

3.5 Appendix - Troubleshooting problems of convergence

1. Rhat > 1.1 First of all check that there are no silly mistakes in the model. Forgetting to put parenthesis, multiplying the wrong parameters, using the wrong operation, etc. can create a model that can't converge. As our models grow in complexity there are more places where to make mistakes. Start simple, see if the model works, add complexity slowly, checking if the model converges at every step. In very rare occasions, when Rhat >1.1 and the model is correct, it may help to increase the number of iterations, but then it's usually a better idea to re-parametrize the model, see 3.

2. Stan gives a warning. The solution may also be point 1. But if the model is correctly specified, you should check Stan's website, there is a very good guide to solve problems in: http://mc-stan.org/misc/warnings.html. If this doesn't work, you may need to re-parametrize the model, see 3.

3. Some models have convergence issues because the sampler struggles to explore the parameters space. This is specially relevant in complex hierarchical models. In this case, the solution might be to re-parametrize the model. This is by no means trivial. However, the simplest parametrization trick to try is to have all the priors on the same rough scale, that is priors shouldn't have different orders of magnitude. You can find some suggestions in the chapter 21 of Stan manual (Stan Development Team 2017), and the following case study: http://mc-stan.org/users/documentation/case-studies/qr_regression.html.

4 Hierarchical linear modeling

References

Blitzstein, Joseph K, and Jessica Hwang. 2014. *Introduction to Probability*. Chapman; Hall/CRC.

Bürkner, Paul-Christian. 2017. "brms: An R Package for Bayesian Multilevel Models Using Stan." *Journal of Statistical Software* 80 (1): 1–28. doi:10.18637/jss.v080.i01.

Gelman, Andrew, John B Carlin, Hal S Stern, and Donald B Rubin. 2014. *Bayesian Data Analysis*. Third Edition. Taylor & Francis.

Hoekstra, Rink, Richard D Morey, Jeffrey N. Rouder, and Eric-Jan Wagenmakers. 2014. "Robust Misinterpretation of Confidence Intervals." *Psychonomic Bulletin & Review* 21 (5). Springer: 1157–64. doi:10.3758/s13423-013-0572-3.

Jäger, Lena Ann, Felix Engelmann, and Shravan Vasishth. 2017. "Similarity-Based Interference in Sentence Comprehension: Literature Review and Bayesian Meta-Analysis." *Journal of Memory and Language* 94: 316–39.

Kerns, G. Jay. 2018. *Introduction to Probability and Statistics Using R*. https://www.nongnu.org/ipsur/.

Malsburg, Titus von der, and Bernhard Angele. 2017. "False Positives and Other Statistical Errors in Standard Analyses of Eye Movements in Reading." *Journal of Memory and Language* 94. Elsevier: 119–33.

Morey, Richard D, Rink Hoekstra, Jeffrey N. Rouder, Michael D Lee, and Eric-Jan Wagenmakers. 2016. "The Fallacy of Placing Confidence in Confidence Intervals" 23 (1): 103–23. doi:10.3758/s13423-015-0947-8.

Neal, Radford M, and others. 2011. "MCMC using Hamiltonian dynamics." *Handbook of Markov Chain Monte Carlo* 2 (11): 2.

Nicenboim, Bruno, and Shravan Vasishth. 2016. "Statistical methods for linguistic research: Foundational Ideas - Part II." *Language and Linguistics Compass* 10 (11): 591–613. doi:10.1111/lnc3.12207.

Nicenboim, Bruno, Felix Engelmann, Katja Suckow, and Shravan Vasishth. Submitted. "Number Interference in German: Evidence for Cue-Based Retrieval." Open Science Framework. doi:10.17605/OSF.IO/MMR7S.

Nicenboim, Bruno, Timo B. Roettger, and Shravan Vasishth. 2018. "Using Meta-Analysis for Evidence Synthesis: The case of incomplete neutralization in German." *Journal of Phonetics* 70: 39–55. doi:https://doi.org/10.1016/j.wocn.2018.06.001.

Nicenboim, Bruno, Shravan Vasishth, Felix Engelmann, and Katja Suckow. 2018. "Exploratory and Confirmatory Analyses in Sentence Processing: A case study of number interference in German." *Cognitive Science* 42 (S4). doi:10.1111/cogs.12589.

Ross, Sheldon. 2002. *A First Course in Probability*. Pearson Education.

Shiffrin, Richard M., Michael Lee, Woojae Kim, and Eric-Jan Wagenmakers. 2008. "A Survey of Model Evaluation Approaches with a Tutorial on Hierarchical Bayesian Methods." *Cognitive Science* 32 (8). Wiley-Blackwell: 1248–84. doi:10.1080/03640210802414826.

Stan Development Team. 2017. "Stan: A C++ Library for Probability and Sampling, Version 2.16.0." http://mc-stan.org/.

Vasishth, Shravan. 2003. *Working Memory in Sentence Comprehension: Processing Hindi Center Embeddings*. New York: Garland Press.

Vasishth, Shravan, and Richard L. Lewis. 2006. "Argument-Head Distance and Processing Complexity: Explaining Both Locality and Antilocality Effects." *Language* 82 (4): 767–94.

Vasishth, Shravan, and Bruno Nicenboim. 2016. "Statistical Methods for Linguistic Research: Foundational Ideas – Part I." *Language and Linguistics Compass* 10 (8): 349–69.

Vasishth, Shravan, Zhong Chen, Qiang Li, and Gueilan Guo. 2013. "Processing Chinese Relative Clauses: Evidence for the Subject-Relative Advantage." *PLoS ONE* 8 (10). Public Library of Science: 1–14.

Vasishth, Shravan, Nicolas Chopin, Robin Ryder, and Bruno Nicenboim. 2017. "Modelling Dependency Completion in Sentence Comprehension as a Bayesian Hierarchical Mixture Process: A Case Study Involving Chinese Relative Clauses." In *Proceedings of Cognitive Science Conference*. London, UK. https://arxiv.org/abs/1702.00564v2.

Vasishth, Shravan, Daniela Mertzen, Lena A. Jäger, and Andrew Gelman. 2018. "The Statistical Significance Filter Leads to Overoptimistic Expectations of Replicability." *Journal of Memory and Language* 103: 151–75. doi:https://doi.org/10.1016/j.jml.2018.07.004.