Applied Statistics

Robin Hankin

Contents

Overview

This is the handbook for MATPMDB (statistics). Details of the course structure, including assessment and timetables, may be found on Canvas. The whole course is based around R and RStudio which may be downloaded from

[https://cran.r-project.org/]

and

[https://www.rstudio.com/]

respectively.

This course lectures have been recorded and the entire playlist is available on YouTube at:

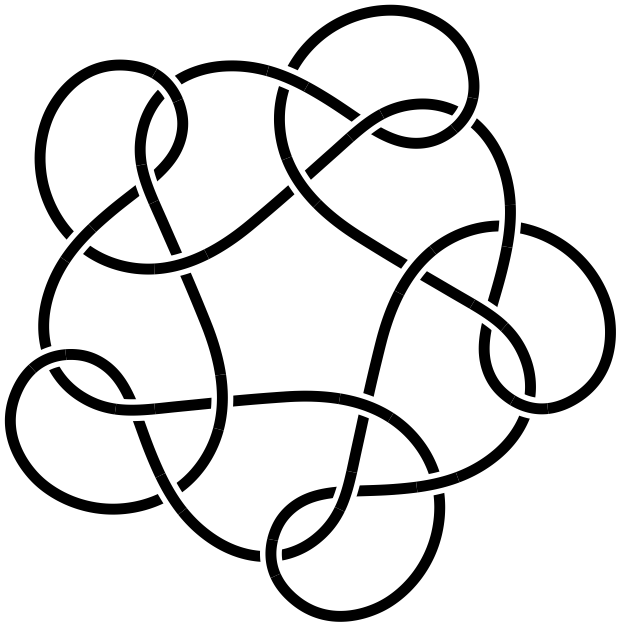
https://www.youtube.com/playlist?list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

which you may view at your leisure. The URL should be clickable in this and similar boxes throughought the manual. Note that the course as delivered will vary slightly from the YouTube videos and in particular the material might be presented in a different order. Some of the links take you to a video which starts midway through a lecture; if this is a problem, look at the playlist at the link above and this will show all lectures in order. Note, however, that each class is a reasonably self-contained unit, split into three or four parts. This manual is distributed under the GNU GPL and is available at

[https://github.com/RobinHankin/matpmdb]

Please feel free to send corrections, bug reports, fanmail, etc to the issues page

[https://github.com/RobinHankin/matpmdb/issues]



[https://github.com/RobinHankin/knotR]

Chapter 1

The binomial distribution

1.1 Bernoulli trials

https://en.wikipedia.org/wiki/Binomial_distribution

A *Bernoulli trial* is an experiment with two possible outcomes. Examples would include tossing a coin (Heads or Tails) but many other examples exist. Observe that any continuous variable may be transformed into a Bernoulli trial by reporting whether it exceeds a particular value. Thus if you are measuring the height of people, the observation "height is less than 1.8m" is a Bernoulli trial as this statement may be true or false.

The standard terminology is "success" or "failure" but remember that these words carry no value judgement. A Bernoulli trial is completely characterised by p, the probability of success.

1.2 Total number of successes: the binomial distribution

YouTube https://www.youtube.com/watch?v=4q9zDgIm1H4&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&t=0s&index=2

YouTube https://www.youtube.com/watch?v=JfYPwu3qWmk&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=2

Given `n` independent Bernoulli trials[1] we are interested in the random variable `r`, the total number of successes. We can observe straightaway that $0 \leq r \leq n$ as it is impossible to have more than n successes, or less than zero.

If we observe r successes, there must be $n - r$ failures; if the probability of success is p then the probability of failure is $1 - p$. The probability of observing r successes followed by $n - r$ failures will be

$$\underbrace{p \times p \ldots \times p}_{r \text{ times}} \times \underbrace{(1-p) \times (1-p) \ldots \times (1-p)}_{n-r \text{ times}} = p^r (1-p)^{n-r} \tag{1.1}$$

But there are many ways to arrange for r successes and $n-r$ failures. Elementary combinatorics shows that there are $\frac{n!}{r!(n-r)!}$ ways. Here $n!$ denotes the factorial function (`?factorial`)[2]. Thus, if the random variable X denotes the number of successes out of n trials each of probability p, we can say

$$P(X = r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r} \tag{1.2}$$

This is more easily expressed using the "choose notation":

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} \tag{1.3}$$

Thus

$$P(X = r) = \binom{n}{r} p^r (1-p)^{n-r}$$

The R programming language has many builtin functions to deal with the binomial distribution.

We will start with the `rbinom()` function[3] which samples from the binomial distribution. From the help page, this function takes three arguments: n, size, prob.

Examples:

```
rbinom(100,5,0.5)
```

[1] Independence to be defined formally later in the course; here we say that the trials do not affect one another.

[2] A question mark is R idiom for accessing the online help system, which can provide further information. For example, to get help on the factorial function, you type `?factorial` at the R prompt.

[3] Remember to type `?rbinom` at the R prompt to get help on this function

```
##   [1] 5 1 2 2 4 2 2 4 3 4 3 0 0 2 3 3 4 3 2 3 3 4 2 3 2 4 2 2 3 0 4 2 4 2 3 2 1
##  [38] 1 3 3 1 2 2 2 1 3 3 3 5 2 2 1 4 2 2 4 2 3 2 2 2 2 3 3 2 3 5 4 5 1 5 5 4 2
##  [75] 2 5 2 2 5 3 3 3 2 3 1 2 3 3 4 2 3 3 2 5 1 4 4 1 4 0
```

You can do this too! The only way to learn is to execute these commands yourself.

In the above, we have 100 samples of size 5, each with a probability of success of 0.5. It is as though I give a fair coin to each of 100 students and tell each one to toss the coin 5 times and record the number of heads (successes).

Observe that if I type the same commands a second time, I get different results:

```
rbinom(100,5,0.5)
```

```
##   [1] 5 2 2 3 5 2 3 3 2 3 3 4 2 4 5 3 5 3 2 2 3 2 4 2 3 3 3 3 2 2 1 1 3 3 3 3 1
##  [38] 2 4 2 1 4 4 1 3 1 5 3 2 3 1 2 2 1 4 3 2 2 1 3 3 4 2 3 3 1 2 0 1 1 1 3 2 2
##  [75] 3 3 3 2 3 1 4 0 3 3 3 3 1 1 3 4 2 2 3 2 3 4 5 3 1 2
```

Now we can change some of the numbers. Let's vary each one in turn. First, change the number of observations from 100 to 50:

```
rbinom(50,5,0.5)
```

```
##  [1] 4 3 2 3 3 4 3 2 3 2 1 2 1 3 3 0 3 2 0 1 3 3 4 4 2 2 0 1 5 3 2 2 1 3 1 2 5 0
## [39] 3 2 3 2 3 1 2 2 4 3 1 0
```

(we have only 50 results now). Now change the number of coin tosses from 5 to 20:

```
rbinom(50,20,0.5)
```

```
##  [1] 14  7 10  7 11 10 11 13  4  8 12 10  7  7 13 10 10 13  7 13  7  6  9  9  9
## [26]  6  6 13  8 11  9 11  7  9  8 11 12 11  9 10  9 10  7  8 10 11  9 10 14 10
```

(see how the numbers have increased). Now change the probability of success from 0.5 to 0.9:

```
rbinom(50,20,0.9)
```

```
##  [1] 18 18 19 17 18 20 18 17 18 17 19 17 18 19 18 14 20 17 16 19 19 19 19 16 19
## [26] 20 16 18 18 16 20 17 18 17 17 19 19 18 20 20 19 15 18 18 17 19 19 18 19 17
```

(see how the number of successes is now higher, as 90% of the coin tosses land heads). Now change the probability to 0.01:

```
rbinom(50,20,0.01)
```

```
##  [1] 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 1 0 0
## [39] 1 0 0 0 0 0 0 0 0 0 1 0
```

(most students get zero heads and 20 tails, a few get 1 or 2 heads).

1.2.1 Summary statistics

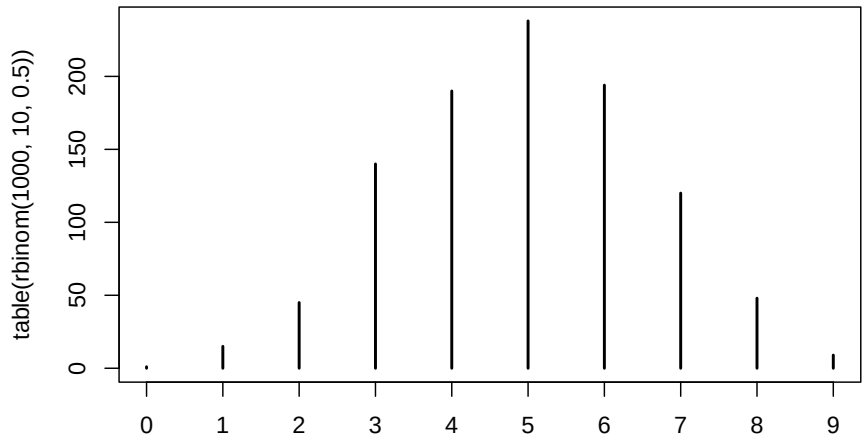
Consider again random sampling from the binomial distribution but this time we would like to summarise a large number of observations, say 1000. We can use the `table()` function:

```
set.seed(0)
table(rbinom(1000,10,0.5))
```

```
##
##   1   2   3   4   5   6   7   8   9  10
##   7  43 121 204 246 203 118  43  13   2
```

See how easy this is to understand. There are 7 observations with 1 success, 43 with 2, and so on up to 2 observations with 10 successes. We can even plot the output:

```
plot(table(rbinom(1000,10,0.5)))
```



Above, we simply wrap the `table()` function inside a `plot()` function). This is a

powerful visualization method. We can use the binomial probability distribution, equation (1.2), in an R session using the `dbinom()` function. Suppose we have a binomial distribution with size 12 and probability 1/3, and we want to calculate the probability of observing 4 successes. We can use the mathematical formula explicitly:

```
n <- 12
p <- 1/3
r <- 4
factorial(n)/(factorial(r)*factorial(n-r))*p^r*(1-p)^(n-r)
```

```
## [1] 0.238446
```

but in this case it is much better to use the built-in functionality `dbinom()`:

```
dbinom(4,12,1/3)
```

```
## [1] 0.238446
```

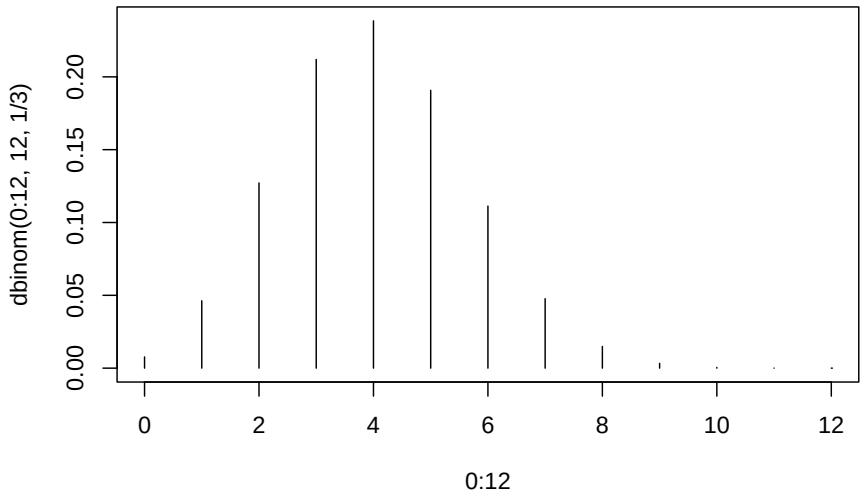
See how the two answers above agree with one another. Note that it is always better to use the builtin functionality in R, rather than to use a written function (the builtin version is faster and more accurate). One further advantage of using the builtin is that it is vectorized:

```
dbinom(0:12,12,1/3)
```

```
##  [1] 7.707347e-03 4.624408e-02 1.271712e-01 2.119520e-01 2.384460e-01
##  [6] 1.907568e-01 1.112748e-01 4.768921e-02 1.490288e-02 3.311751e-03
## [11] 4.967626e-04 4.516023e-05 1.881676e-06
```

This shows the probability of observing $0, 1, 2, \ldots, 12$ successes out of 12 trials. It is easy to visualise this using a plot:

```
plot(0:12, dbinom(0:12,12,1/3),type='h')
```



1.2.2 Numerical verification

It is important when learning new theoretical formulas to verify that they are correct. We can do this easily with R, as it includes a large number of numerical sampling routines.

Consider, for example, the binomial distribution with size 9 and probability 0.4. We seek the probability that exactly 3 successes are observed. First, calculate the theoretical value:

```
dbinom(3,9,0.4)
```

```
## [1] 0.2508227
```

Now, we will take a random sample and *count* how many are equal to 3:

```
set.seed(0)
table(rbinom(1e6,9,0.4) == 3)
```

```
##
##  FALSE   TRUE
## 749782 250218
```

(here we take a sample of size one million (`1e6`); the `==` is R idiom for asking the question "is the left hand side equal to the right hand side?", the answer to

which is either `TRUE` or `FALSE`). In this case, The `TRUE` count of 250218 shows how many observations from our random sample were indeed equal to 3.

The probability of a random observation drawn from this binomial distribution being equal to three is thus

```
250218/1e6
```

```
## [1] 0.250218
```

Note that this closely matches the theoretical value of 0.2508227 given above. It is possible to vectorize this reasoning and streamline the idiom. Say we have size 7 and probability 0.33:

```
dbinom(0:7,7,0.33)
```

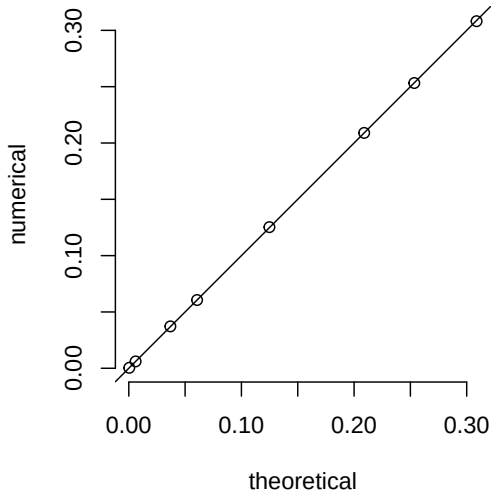
```
## [1] 0.0606071161 0.2089588628 0.3087601107 0.2534597924 0.1248384052
## [6] 0.0368925436 0.0060569848 0.0004261844
```

```
table(rbinom(1e6,7,0.33))/1e6
```

```
##
##        0        1        2        3        4        5        6        7
## 0.060808 0.209343 0.309419 0.252704 0.124347 0.036973 0.005965 0.000441
```

And visualization is possible:

```
theoretical <- dbinom(0:7,7,0.33)
numerical <- table(rbinom(1e6,7,0.33))/1e6
par(pty='s')
plot(theoretical,numerical,asp=1,axes=FALSE)
axis(side=1,at=seq(from=0,to=0.3,by=0.05))
axis(side=2,at=seq(from=0,to=0.3,by=0.05))
abline(0,1)
```



In the plot above, we plot theoretical value on the horizontal axis and numerical values on the vertical axis. Exact agreement would mean the points are exactly on the diagonal line; see how close the agreement is.

Chapter 2

The normal or Gaussian distribution

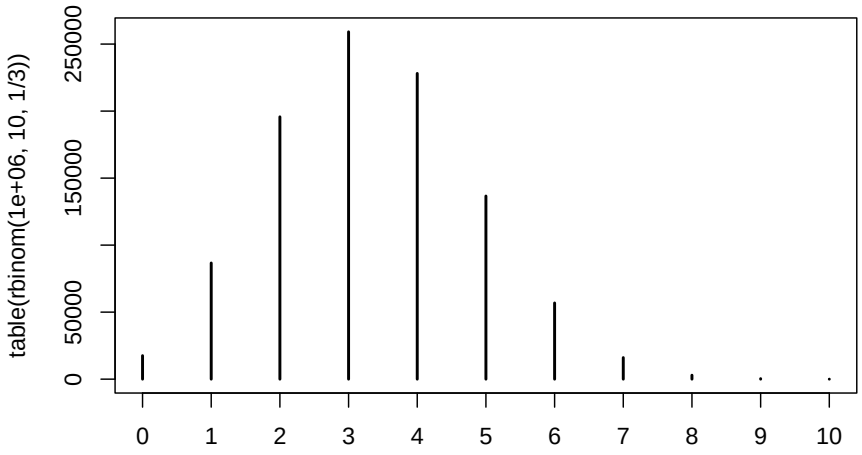
https://www.youtube.com/watch?v=2I30Ju4we10&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=6

https://www.youtube.com/watch?v=TVd2p-L5e7g&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=7

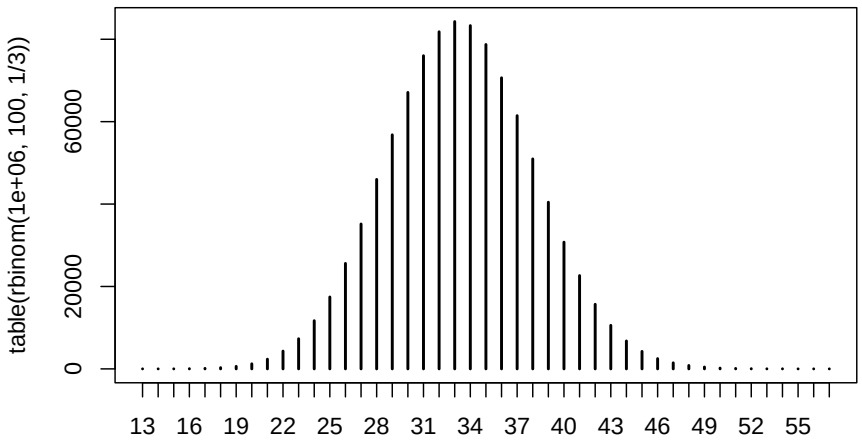
2.1 Binomial distribution in the limit of large n

Suppose we consider the binomial distribution but with larger and larger values of n. The following three figures show table plots of binomial distributions with p=1/3 and n=10,100,1000,10000:

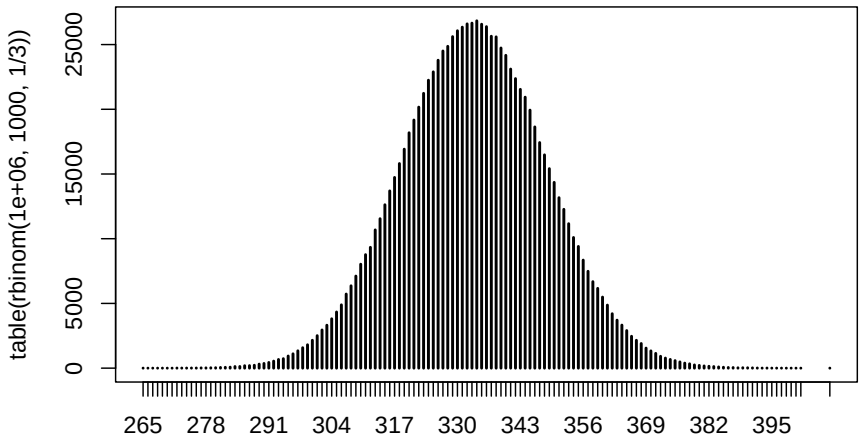
```
plot(table(rbinom(1e6,10,1/3)))
```



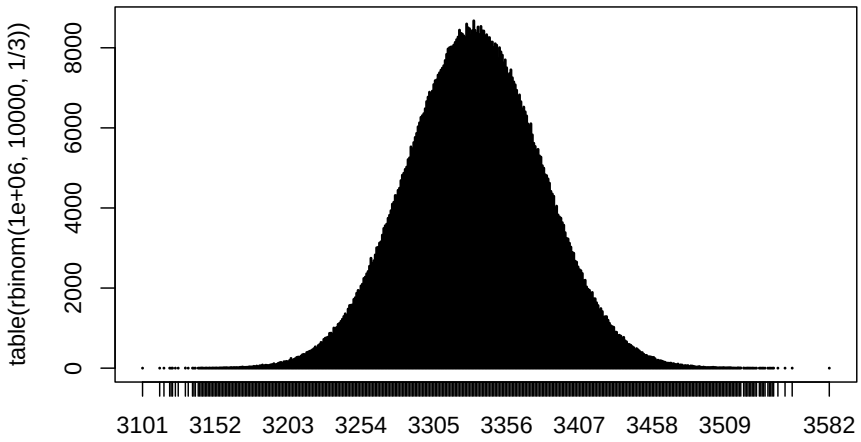

```
plot(table(rbinom(1e6,100,1/3)))
```



```
plot(table(rbinom(1e6,1000,1/3)))
```




```
plot(table(rbinom(1e6,10000,1/3)))
```



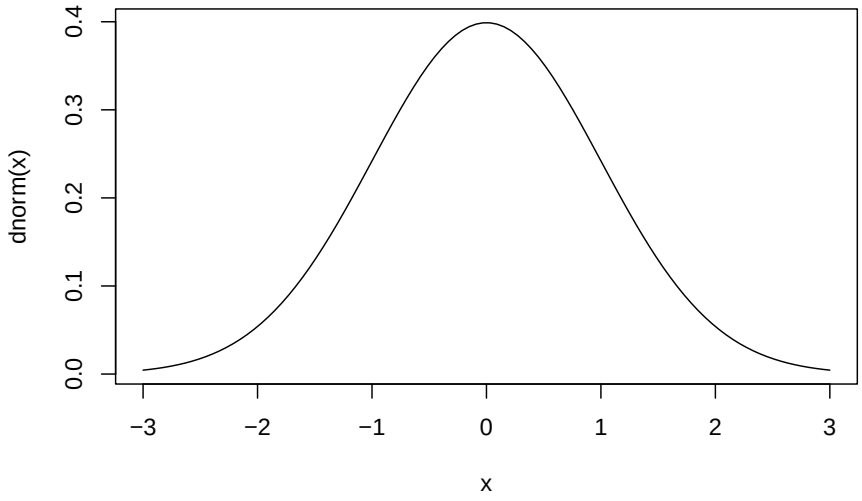
Apart from some small-scale irregularities, it is clear that the distributions are approaching a bell-shaped distribution. This is known as the *Normal*, or the *Gaussian* distribution.

2.2 The Gaussian distribution

https://en.wikipedia.org/wiki/Normal_distribution

The Gaussian distribution is the most commonly encountered distribution in the whole of statistics, and has a characteristic shape which we can plot using the `dnorm()` function.

```
x <- seq(from= -3,to=3,len=100)
plot(x,dnorm(x),type='l')
```



(the `seq()` function gives a sequence; see `?seq` for details). In the above, we see the Gaussian from -3 to 3 but the distribution is infinitely wide. The probability density function for the Gaussian is:

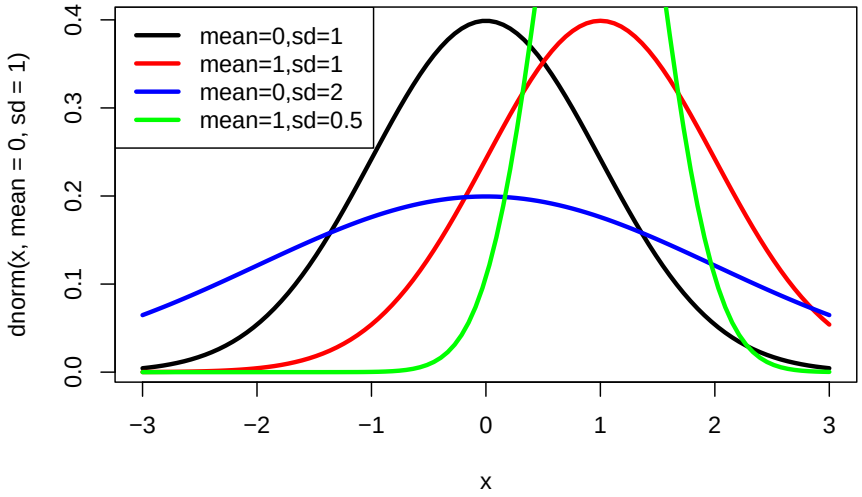
$$\frac{1}{\sqrt{2\pi}} \exp\left(\frac{-x^2}{2}\right) \tag{2.1}$$

But R provides a builtin function, `dnorm()`, which is easier to use, faster, and more accurate. Remember you can type "?dnorm" at the R prompt to get more help.

The Gaussian distribution has two adjustable parameters, the *mean* (written "μ" in mathematics) and the *standard deviation* (written "σ"). These govern the position and width of the distribution. We can change the mean and standard deviation by supplying the appropriate arguments to `dnorm()`:

```
x <- seq(from= -3,to=3,len=100)
  plot(x,dnorm(x,mean=0,sd=1  ),type="l",lwd=3,col="black")
points(x,dnorm(x,mean=1,sd=1  ),type="l",lwd=3,col="red")
points(x,dnorm(x,mean=0,sd=2  ),type="l",lwd=3,col="blue")
points(x,dnorm(x,mean=1,sd=0.5),type="l",lwd=3,col="green")
legend("topleft",
    legend=c("mean=0,sd=1",
                 "mean=1,sd=1",
```

```
                  "mean=0,sd=2",
                  "mean=1,sd=0.5"),
lwd=3,col=c("black","red","blue","green"))
```



In the above figure, see how the different mean and sd arguments are passed to `dnorm()`. The green curve is higher than the others and is truncated at the top, but all the curves have the same area. From the R help page `?dnorm` we can see that the mean and standard deviation have default values of 0 and 1 respectively, which is what was plotted in black above. This is known as the *standard Normal distribution* and is denoted $N(0, 1)$.

2.3 Cumulative distribution function

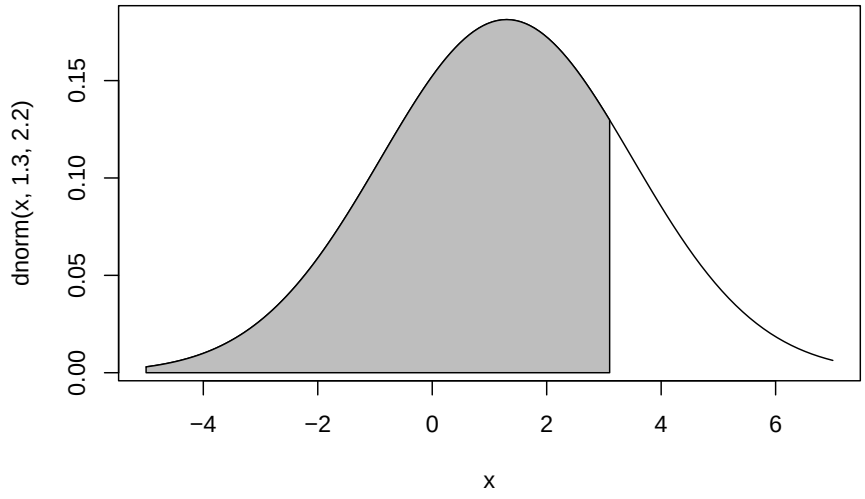
If we want the probability that a Gaussian random variable is less than a particular value we can use the `pnorm()` function. Suppose we have X drawn from a Gaussian distribution with a mean of 1.3 and a standard deviation of 2.2. What is the probability that $X \leq 3.1$?

```
pnorm(3.1, mean=1.3, sd=2.2)
```

```
## [1] 0.7933733
```

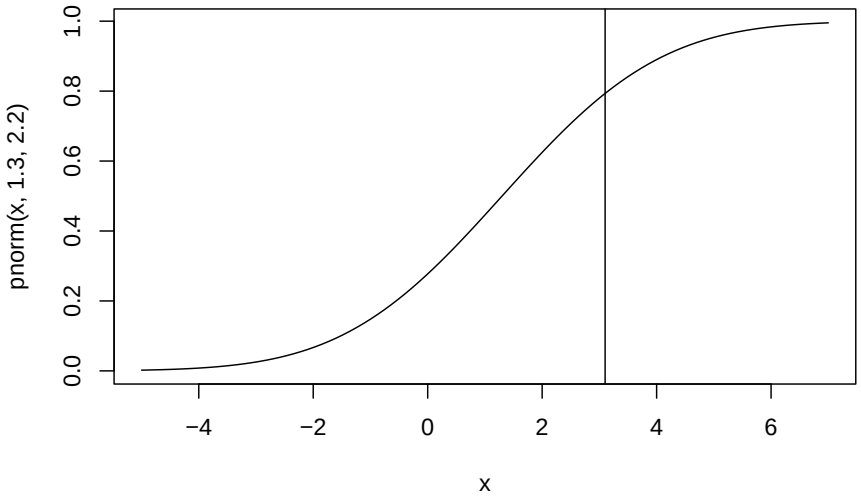
that is, a probability of about 79%. We can get two different pictorial versions of this as follows:

```
x <- seq(from=-5,to=7,len=100)
plot(x,dnorm(x,1.3,2.2),type="l")
x1 <- seq(from=-5,to=3.1,len=100)
polygon(c(x1,rev(x1)),c(x1*0,rev(dnorm(x1,1.3,2.2))),col='gray')
```



(see how we pass the mean and standard deviation directly to `dnorm()`). The 79% figure is the grey area in the above figure. The other way to do this would be to plot the *cumulative* distribution function `pnorm()` which gives the probability that X is *less than or equal to* a particular value. Thus:

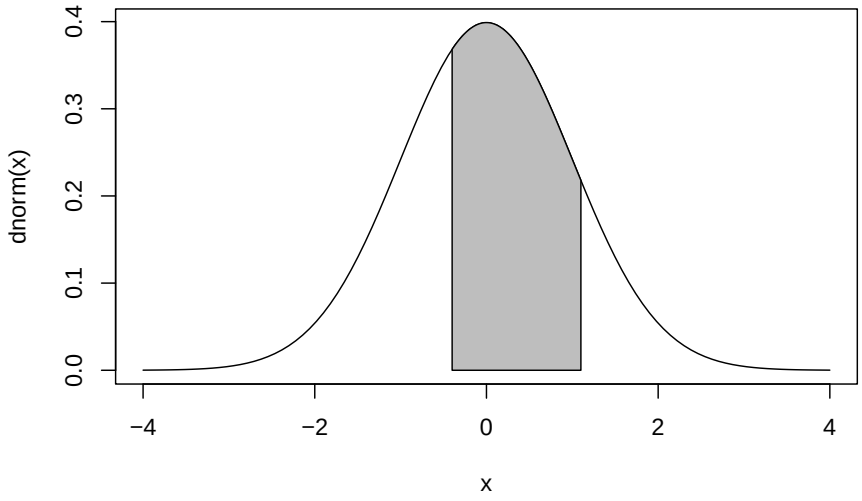
```
x <- seq(from=-5,to=7,len=100)
plot(x,pnorm(x,1.3,2.2),type="l")
abline(v=3.1)
```



In the above figure, we have drawn a line at $y = 3.1$ and this intersects the CDF at about 0.79. If we want to find the probability that X lies between two values a and b, then this would be represented as the compound statement $(X > a) \quad \& \quad (X < b)$ as there are two conditions present[1]. This is more usually written $a < X < b$. We can visualise this easily. Suppose we have a standard Gaussian $N(0, 1)$ and want the probability that $-0.4 < X < 1.1$:

```
x <- seq(from=-4,to=4,len=100)
plot(x,dnorm(x),type="l")
x1 <- seq(from=-0.4,to=1.1,len=100)
polygon(c(x1,rev(x1)),c(x1*0,rev(dnorm(x1))),col='gray')
```

[1] for simplicity I am ignoring the difference between "$<$" and "$\leq$"



The R idiom to calculate the area is

```
pnorm(1.1) - pnorm(-0.4)
```

```
## [1] 0.5197557
```

which corresponds to a probability of about 52%, which looks about right.

2.3.1 Quantile function

https://en.wikipedia.org/wiki/Quantile_function

The remaining R function for the Gaussian distribution is the quantile function `qnorm()`. In R, `qnorm(p)` gives the value of x for which

$$\text{Prob}\,(X \leq x) = p \tag{2.2}$$

For example, suppose we have $X \sim N(0, 1)$ and want to find the value of X that is exceeded with a probability of 0.05, or 5%. This means that $\text{Prob}\,(X \leq x)$ is 0.95 so the R idiom would be:

```
qnorm(0.95)
```

```
## [1] 1.644854
```

2.4 Numerical verification

All of the above results using `pnorm()` etc should be verified by numerical sampling. We can use the function `rnorm()` to generate random numbers from the Gaussian.

```
rnorm(10)
```

```
##  [1]  1.61343760 -0.35957784 -1.13884847  0.12927467 -0.85528622 -0.07700082
##  [7] -0.16091911  0.01607006  0.10376938  0.21195451
```

(in the above, the argument 10 specifies the number of observations to sample). Function `rnorm()` takes arguments to specify the mean and standard deviation if necessary:

```
rnorm(10,mean=100)
```

```
##  [1] 100.56922  99.71601 100.75246  99.45836 100.55185  98.93561 100.55889
##  [8]  98.48356  99.53361  99.97497
```

```
rnorm(10,mean=100,sd=0.01)
```

```
##  [1]  99.99725 100.03030 100.00749 100.00033  99.98434  99.98341  99.98431
##  [8]  99.99468  99.99552 100.00754
```

We can use R to "ask a question" with the "<" and ">" symbols. If we sample from a standard Gaussian and are interested in whether the observation is greater than, say, 1.3, then the idiom would be:

```
rnorm(10) > 1.3
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
```

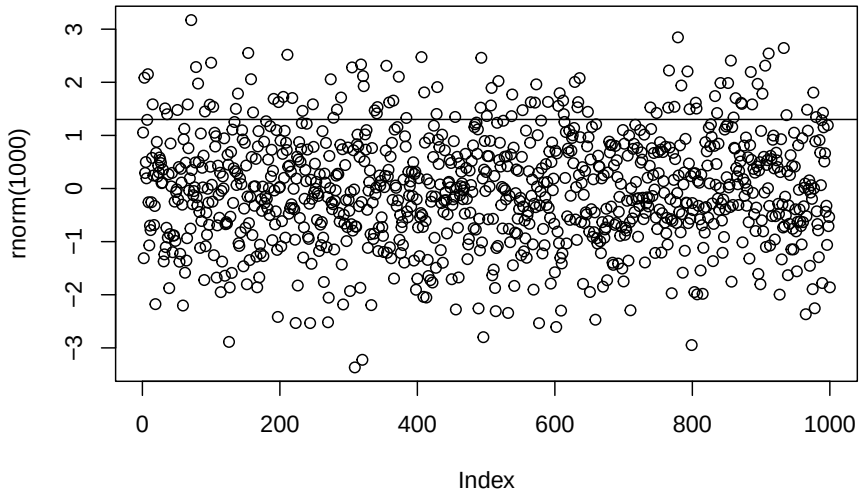
and if we wanted to, we could use the `table()` function to summarize:

```
set.seed(0)
table(rnorm(1e6) > 1.3)
```

```
##
##  FALSE   TRUE
## 903503  96497
```

[do not type the `set.seed(0)` command; this is here to ensure reproducibility in the manual]. In the above, we had `1e6` (that is, 10^6 or one million) samples, of which 96497 were over 1.3. This corresponds to a probability of 96497/1000000, or a little over 0.096. We can make a different visualization of the same situation as follows:

```
plot(rnorm(1000))
abline(h=1.3)
```



and in the above plot we can see that about 10% of the points are above the line as we would expect.

2.5 Relationship between pnorm() and qnorm().

Functions `pnorm()` and `qnorm()` are in an inverse relationship with one another. Thus

```
pnorm(qnorm(0.1))  # should return 0.1
```

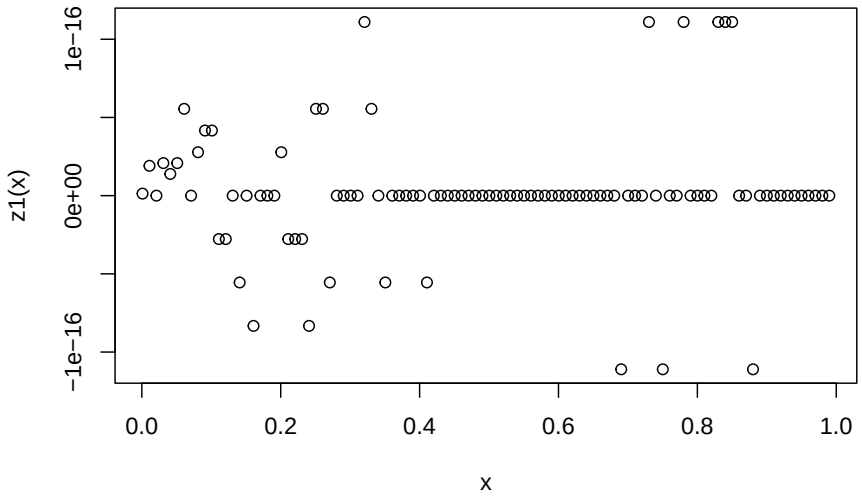
```
## [1] 0.1
```

```
qnorm(pnorm(1.1))  # should return 1.1
```

```
## [1] 1.1
```

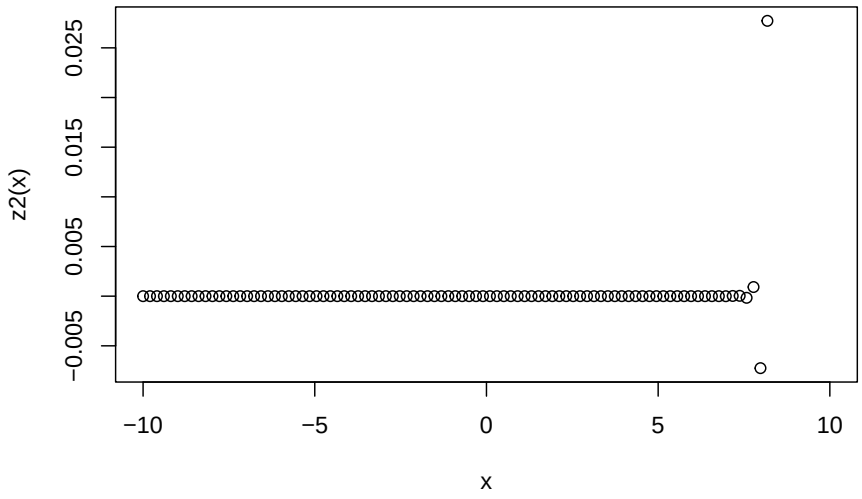
In the above, note carefully the difference between the two examples. In the first line, observe that `qnorm()` requires its argument to be in the range 0 to 1 as it takes a probability; while in the second line, `pnorm()` takes any value on the real line. We can conduct a more exacting test as follows:

```
z1 <- function(p){pnorm(qnorm(p))-p}   # should return zero
x <- seq(from=0.001,to=0.99,len=100)
plot(x,z1(x))
```



In the above, note the vertical scale. For completeness we should also plot the other test:

```
z2 <- function(v){qnorm(pnorm(v))-v}   # should return zero
x <- seq(from=-10,to=10,len=100)
plot(x,z2(x))
```



In the above, note how the most serious errors are for large values of x, and there are no points at all above about $x = 8$. Is this what you would expect? Does this graph verify that `pnorm()` and `qnorm()` are operating correctly? Or does it reveal a problem either with our expectations or the behaviour of an R function? Is the test a good one? If not (or indeed if it is a good test), could you make a better one?

2.6 Examples

Here are some simple examples of the `pnorm()` function in use.

2.6.1 Example 1

"If X is drawn from a standard Gaussian, find the probability that $X < 1.1$"

Theoretical answer

```
pnorm(1.1)
```

```
## [1] 0.8643339
```

Numerical verification

```
table(rnorm(1e6) < 1.1)/1e6
```

```
##
##    FALSE     TRUE
## 0.135657 0.864343
```

In the above I have divided the table results by the number of observations so the numbers represent probabilities.

2.6.2 Example 2

"If X is drawn from a Gaussian distribution with mean 5 and standard deviation 1.4, find the probability that: (i), $X < 6$; (ii) $X > 7$. Verify your findings numerically."

2.6.2.1 part (i)

```
pnorm(6, mean=5, sd=1.4)
```

```
## [1] 0.7624747
```

```
table(rnorm(1e6,5,1.4) < 6)/1e6
```

```
##
##    FALSE     TRUE
## 0.237079 0.762921
```

2.6.2.2 part (ii)

To get the probability that X is *greater than* 7, we need one minus the probability that X is *less than* 7:

```
1-pnorm(7, mean=5, sd=1.4)
```

```
## [1] 0.07656373
```

```
table(rnorm(1e6,5,1.4)  > 7)/1e6
```

```
##
##   FALSE    TRUE
## 0.92342 0.07658
```

Another way to do this is to use the following construction:

```
pnorm(7, mean=5, sd=1.4, lower.tail=FALSE)
```

```
## [1] 0.07656373
```

which would be more accurate.

2.6.3 Example 3

"If X is drawn from a Gaussian distribution with mean 10 and standard deviation 0.4, find the probability that X lies in the range 10.1-10.3" and verify your findings numerically."

Theoretical answer

```
pnorm(10.3,10,0.4)-pnorm(10.1,10,0.4)
```

```
## [1] 0.1746663
```

Numerical verification

```
x <- rnorm(1e6,10,0.4)
table((x>10.1) & ( x<10.3))/1e6
```

```
##
##    FALSE     TRUE
## 0.825562 0.174438
```

So the theoretical and numerical values agree approximately.

2.7 Meaning of population mean and standard deviation

I have been using the terms mean and standard deviation rather loosely. Here I will give a more precise definition of them.

2.7.1 Mean of a distribution

https://en.wikipedia.org/wiki/Mean#Mean_of_a_probability_distribution https://en.wikipedia.org/wiki/Expected_value

The mean of a distribution [sometimes "population mean", sometimes "expectation"] has a technical meaning but here we can say that it is defined as the long-run average of observations drawn from that distribution. We sometimes

write $\mathbb{E}(X)$ for the expectation. The expected value of the Gaussian distribution is μ, and this is easy to demonstrate in R:

```
mean(rnorm(100,10,2))
```

```
## [1] 10.16566
```

```
mean(rnorm(100,10,2))
```

```
## [1] 9.950975
```

```
mean(rnorm(100,10,2))
```

```
## [1] 9.645864
```

In each of the three lines above, we have one hundred random samples from a $N(10, 2)$ distribution. The arithmetic mean of these—that is, the sample mean—is roughly equal to the population mean, which is 10. The result is not exact, due to the finite sample size. Observe carefully that we cannot tell from a sample what the population mean is.

If we wanted a more precise verification, we would have to use a larger sample size:

```
mean(rnorm(1e6,10,2))
```

```
## [1] 9.995938
```

```
mean(rnorm(1e6,10,2))
```

```
## [1] 9.998527
```

```
mean(rnorm(1e6,10,2))
```

```
## [1] 9.998053
```

The mean of the Gaussian distribution is identical to its mode and median, as it is symmetric and unimodal. Note that some distributions do not have a mean (for example, the Cauchy distribution).

2.7.2 Variance and standard deviation of a distribution

The standard deviation measures the "width" of a distribution, again with a specific technical meaning.

It is easier to start with "deviance" $\mathbb{D}$, which is defined as the difference between a random variable and its expectation, $\mathbb{D} = X - \mathbb{E}(X)$. The variance is the expectation of $\mathbb{D}^2$. Standard deviation is just the square root of variance. In R, we can estimate the standard deviation of a sample using the `sd()` command:

```
sd(rnorm(100,10,2))
```

```
## [1] 1.891868
```

```
sd(rnorm(100,10,2))
```

```
## [1] 2.083996
```

```
sd(rnorm(100,10,2))
```

```
## [1] 1.717998
```

See the higher variability than the mean. We can again use a larger sample size:

```
sd(rnorm(1e6,10,2))
```

```
## [1] 2.002242
```

```
sd(rnorm(1e6,10,2))
```

```
## [1] 2.000522
```

```
sd(rnorm(1e6,10,2))
```

```
## [1] 2.001964
```

and we see that the estimated standard deviation is approximately the true value of 2.

The *variance* is just the square of the standard deviation (alternatively, the standard deviation is the square root of the variance).

The mathematical notation for mean and standard deviation of a random variable S is $\mathbb{E}(X)$ and $\mathbb{V}(X)$ respectively. The mean is usually denoted with Greek letter μ and standard deviation is Greek sigma, σ. Variance is usually σ^2.

2.8 Binomial distribution: mean and standard deviation

As we saw at the beginning of this chapter, if we consider a binomial distribution with fixed p but allow n to grow very large, we have approximately a normal distribution.

Even if n is small, the binomial distribution $B(n,p)$ has a well-defined mean μ and variance σ^2 given by

$$\mu = np \qquad \sigma^2 = np(1-p) \tag{2.3}$$

Thus, for example, if $n = 6$ and $p = \frac{1}{3}$ we would have a mean of $np = 6 \times \frac{1}{3} = 2$ and a variance of $np(1-p) = 6 \times \frac{1}{3} \times \frac{2}{3} = \frac{4}{3}$. Thus the standard deviation will be $\sqrt{\frac{4}{3}}$, or about 1.154. Verifying this numerically is straightforward:

```
mean(rbinom(1e6,6,1/3))
```

```
## [1] 2.001557
```

```
mean(rbinom(1e6,6,1/3))
```

```
## [1] 2.000584
```

```
mean(rbinom(1e6,6,1/3))
```

```
## [1] 2.001221
```

and

```
sd(rbinom(1e6,6,1/3))
```

```
## [1] 1.154097
```

```
sd(rbinom(1e6,6,1/3))
```

```
## [1] 1.154407
```

```
sd(rbinom(1e6,6,1/3))
```

```
## [1] 1.153437
```

Thus the numerical results closely match the theoretical values. We will see the same technique used for other distributions later in the course.

Chapter 3

Hypothesis testing

https://en.wikipedia.org/wiki/Statistical_hypothesis_testing

https://www.youtube.com/watch?v=yc_SKWZwPBw&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=3

It is a very frequent occurence in science that we are investigating an effect and seek evidence that it has occurred. The general terms discussed in this chapter are

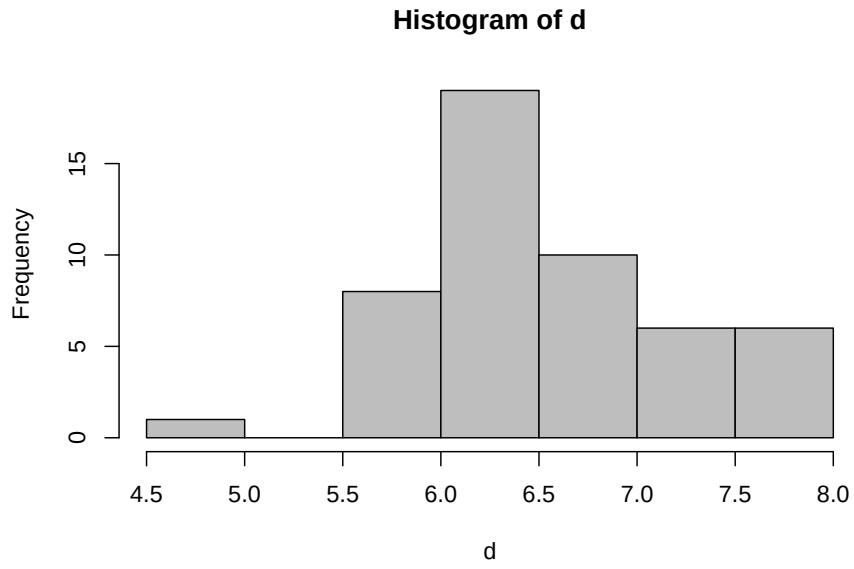
- hypothesis testing
- standard error of the mean
- null hypothesis
- p-value
- critical region
- confidence interval

We will use a standard dataset drawn from the `iris` to illustrate the general theory. First we will take our data, drawn from a classic dataset describing sepal lengths of iris flowers:

```
d <- iris3[,1,"Virginica"]
d
```

```
##  [1] 6.3 5.8 7.1 6.3 6.5 7.6 4.9 7.3 6.7 7.2 6.5 6.4 6.8 5.7 5.8 6.4 6.5 7.7 7.7
## [20] 6.0 6.9 5.6 7.7 6.3 6.7 7.2 6.2 6.1 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7 6.3 6.4
## [39] 6.0 6.9 6.7 6.9 5.8 6.8 6.7 6.7 6.3 6.5 6.2 5.9
```

```
hist(d,col='gray')
```



3.1 Standard error of the mean

https://en.wikipedia.org/wiki/Standard_error

What we want to do is to calculate the *population* mean for sepal lengths. That is, we assume that sepal lengths are a random variable drawn from $N(\mu, \sigma)$ [Gaussian distribution] but with unknown mean μ. Of course we can calculate the *sample* mean of our data:

```
mean(d)
```

```
## [1] 6.588
```

But what is the *uncertainty* of this estimate? Observe that the true population mean of sepal lengths may well be 6.589 or 5.787; but we would be uncomfortable stating that the true population mean is as 114.2 or -232. Observe carefully that we are using the *sample mean* to estimate the *population mean*.

3.1.1 Sampling distribution of the mean

https://en.wikipedia.org/wiki/Sampling_distribution

It is a mathematical fact that if $x_1, x_2, \ldots, x_n$ are independent and drawn from $N(\mu, \sigma)$, then

$$\overline{x} \sim N\left(\mu, \frac{\sigma}{\sqrt{n}}\right) \tag{3.1}$$

where $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the sample mean. The standard deviation of the sample mean is known as the *standard error of the mean*. We can test that equation (3.1) is correct by numerical sampling; we use a standard Gaussian $N(0, 1)$ on the grounds that we know the true value of the mean and standard deviation. Supposing we have a sample of size 40:

```
mean(rnorm(40))
```

```
## [1] 0.1798706
```

```
mean(rnorm(40))
```

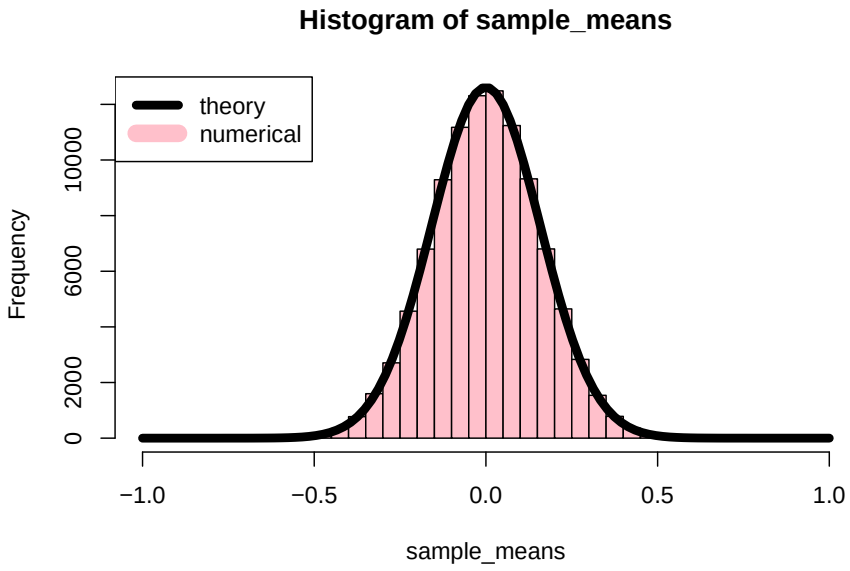
```
## [1] 0.02822533
```

```
mean(rnorm(40))
```

```
## [1] 0.06263054
```

the above shows some of the variability of the sample mean. Because each of the observations is $N(0, 1)$, equation (3.1) shows that the sample mean is distributed as $N\left(0, \frac{1}{\sqrt{40}}\right)$. We can verify this using `replicate()`:

```
width <- 0.05
n <- 100000
sample_means <- replicate(n,mean(rnorm(40)))
hist(sample_means,breaks=seq(from=-1,to=1,by=width),col='pink')
x <- seq(from=-1,to=1,len=100)
points(x,n*width*dnorm(x,sd=1/sqrt(40)),type="l",lwd=6)
legend("topleft",lwd=c(6,12),col=c("black","pink"),legend=c("theory","numerical"))
```



In the above, the black line shows the distribution function for the theoretical Gaussian, closely matching the observations.

3.2 Null Hypothesis

https://en.wikipedia.org/wiki/Null_hypothesis

Returning to the sepal lengths, suppose we know that last year the population mean was exactly 6.4. Is there evidence that this year the sepals are longer?

Well, the sample mean certainly exceeds 6.4:

```
mean(d)
```

```
## [1] 6.588
```

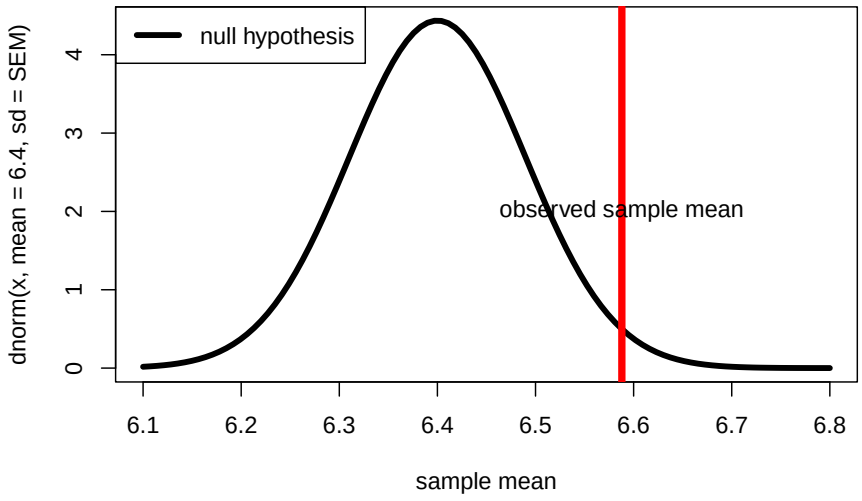
But as we saw above, there is sampling uncertainty on the mean (the standard deviation of the sample mean is the standard error of the mean), so it is not clear whether the sample mean is actually evidence of a real change, or the result of random variability.

To proceed, we define a *null hypothesis* H_0, which is the statement that there is no change. The concept of null hypothesis is difficult and not altogether consistent so I will not define it formally here. It usually means something

along the lines of "the effect we are investigating is not present". The idea is that we give the null hypothesis a fair go and if it does not account for our observations we infer that the null is incorrect and that the effect is real. We then consider what the null predicts.

If the null is true, we have a standard error of $\frac{\sigma}{\sqrt{n}}$:

```
x <- seq(from=6.1,to=6.8,len=100)
n <- length(d)
SEM <- sd(d)/sqrt(n)
plot(x,dnorm(x,mean=6.4,sd=SEM),xlab="sample mean",type="l",lwd=4)
legend("topleft",lwd=c(4),col=c("black"),legend=c("null hypothesis"))
abline(v=mean(d),lwd=5,col='red')
text(mean(d),2,"observed sample mean")
```



3.3 p-value

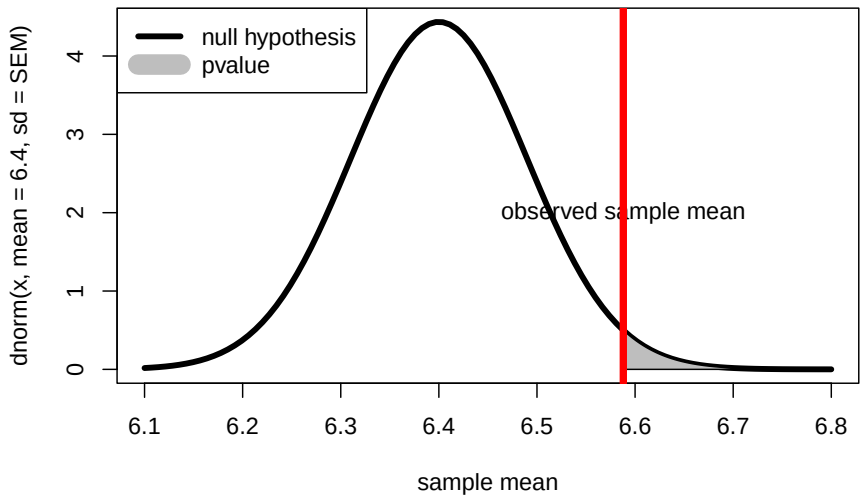
https://en.wikipedia.org/wiki/P-value

https://www.youtube.com/watch?v=-q6KG_ZurcU&index=8&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

We can assess whether an observation is consistent with a particular null by calculating the p-value. We measure consistency by calculating the p-value:

The p-value is defined as the probability, if the null is true, of obtaining the observation or an observation more extreme. Here, "more extreme" means "larger than the observed sample mean", although other interpretations are possible and considered later.

```
x <- seq(from=6.1,to=6.8,len=100)
n <- length(d)
SEM <- sd(d)/sqrt(n)
plot(x,dnorm(x,mean=6.4,sd=SEM),xlab="sample mean",type="l",lwd=4)
legend("topleft",lwd=c(4,14),col=c("black","gray"),legend=c("null hypothesis","pvalue")
text(mean(d),2,"observed sample mean")
jj <- seq(from=mean(d),to=6.8,len=100)
jj1 <- c(jj,rev(jj))
polygon(jj1,c(jj*0,dnorm(rev(jj),mean=6.4,sd=SEM)),col='gray')
abline(v=mean(d),lwd=5,col='red')
```



It is easy to calculate the pvalue in R:

```
1-pnorm(mean(d),6.4,SEM)
```

```
## [1] 0.01828261
```

We compare the pvalue with 0.05 (that is, 5%), and if it is smaller than 0.05 we *reject* the null and infer that the null is incorrect. If, on the other hand, p is greater than 0.05, we fail to reject the null, and come to no conclusion.

In this case the p-value is about 1.8%, which is less than 0.05 so we reject the null and conclude that this year the sepals are in fact longer.

3.4 Student t-test

https://en.wikipedia.org/wiki/Student's_t-test

In the above, we have used the estimated value of the standard deviation given by R idiom `sd()`. However, the estimated value is uncertain, as we can see by numerical sampling, here with sample size 100:

```
replicate(6,sd(rnorm(100)))
```

```
## [1] 0.9914902 0.9973961 1.0809776 1.0280012 1.1108637 1.0113929
```

and if the number of observations is smaller, say 4, then the uncertainty is higher:

```
replicate(6,sd(rnorm(4)))
```

```
## [1] 1.3291036 0.6201184 1.0744435 0.5614508 0.5205831 0.9685620
```

This does not cause a problem if the number of observations is greater than about 30 but for small sample sizes the error becomes serious. It is possible to correct the error using a technique called the *Student t test*.

The concepts are the same but the sampling distribution changes to reflect the fact that we are conditioning on an estimated value of the variance. In any event, the R idiom is straightforward:

```
t.test(d,mu=5.6,alternative="greater")
```

```
##
##  One Sample t-test
##
## data:  d
## t = 10.987, df = 49, p-value = 4.039e-15
## alternative hypothesis: true mean is greater than 5.6
## 95 percent confidence interval:
##  6.437233      Inf
```

```
## sample estimates:
## mean of x
##     6.588
```

Giving a similar p-value to the Z test above.

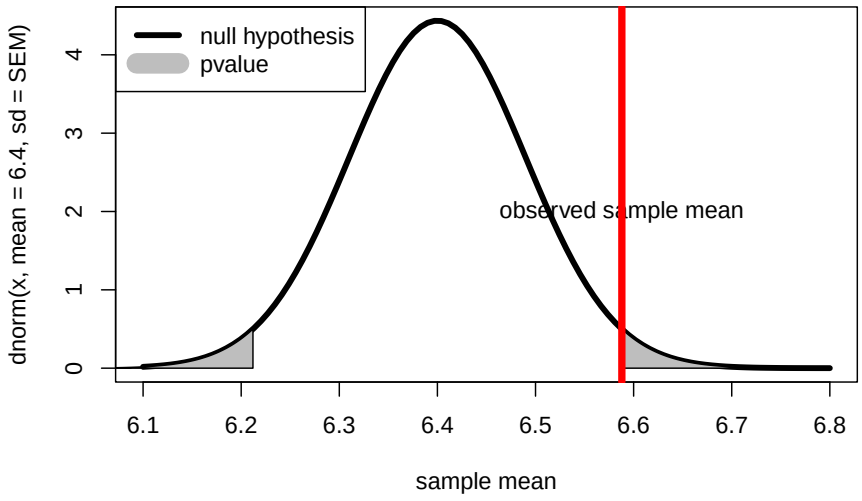
3.5 One-sided and two-sided tests

https://en.wikipedia.org/wiki/One-_and_two-tailed_tests

Recall the definition of p-value: "the probability, if the null is true, of obtaining the observation or an observation *more extreme*". Here, we will investigate what "more extreme" means.

Up to now, "more extreme" has meant "larger" but it could easily mean "further away from the null mean" and we would get an image like the following:

```
x <- seq(from=6.1,to=6.8,len=100)
n <- length(d)
SEM <- sd(d)/sqrt(n)
plot(x,dnorm(x,mean=6.4,sd=SEM),xlab="sample mean",type="l",lwd=4)
legend("topleft",lwd=c(4,14),col=c("black","gray"),legend=c("null hypothesis","pvalue")
text(mean(d),2,"observed sample mean")
f <- function(x){
  jj <- c(x,rev(x))
  polygon(jj,c(x*0,dnorm(rev(x),mean=6.4,sd=SEM)),col='gray')
}
y <- seq(from=mean(d),to=6.8,len=100)
f(y)
f(6.4+(6.4-y))
abline(v=mean(d),lwd=5,col='red')
```



In this case the p-value will be double the one-sided value:

```
2*(1-pnorm(mean(d),6.4,SEM))
```

```
## [1] 0.03656522
```

In general one uses a one-sided (right tail) test when we are testing for the population mean *exceeding* the null mean, a one sided (left tail) test when we are testing for the population mean being *less than* the the null mean; and a two-sided test when we are testing for the population mean *differing* from the null mean.

The decision to use a one-sided or two-sided test is often subtle. In this course, there will always be a linguistic clue such as "it is not clear whether we expect an increase or a decrease" (indicating a two-sided test), or "the experiment is designed to detect an increase in mean" or some similar wording. See previous exam papers for examples.

3.6 Two-sample hypothesis testing

Although the one-sample test considered above is the simplest case, it is more common to consider two samples and ask whether they have a different mean.

If $x_1, \ldots, x_n \sim N(\mu_x, \sigma_x^2)$ and $y_1, \ldots, y_m \sim N(\mu_y, \sigma_y^2)$ then a sensible null hypothesis might be $\mu_x = \mu_y$. Note that the actual values of μ_x and μ_y are not

interesting; we are interested in whether they differ. Also observe that we are not interested in the values of the standard deviations σ_x, σ_y. Again, one-sided or two-sided tests may be used.

The details are rather messy but the Student t-test described above may be modified to cope. Observe that we are comparing two population means, neither of which is known with certainty. They have different uncertainties which makes life difficut.

The t-test uses $\delta = \overline{x} - \overline{y}$ to test the null. If the null is true, then δ has a particular distribution (it is again a modified student t distribution). The details are complicated but R makes life easy.

3.6.1 Example of two-sample hypothesis testing.

Suppose we collect weights kiwi birds from two forests, forest A and forest B, and want to know whether the kiwi from A and B differ. The data is

```
kiwi_a <- c(0.9, 1.02, 1.07, 0.92, 0.84, 0.9, 1.06, 1.01, 1.14, 0.96)
kiwi_b <- c(1.09, 1.14, 1.15, 1.03, 1.18, 1.04, 1.18, 1.17, 1.08, 1.02,
0.99, 1.06, 1.03)
```

We can calculate the *sample* means easily:

```
mean(kiwi_a)
```

```
## [1] 0.982
```

```
mean(kiwi_b)
```

```
## [1] 1.089231
```

But to assess whether this is evidence for the *population* means differing we need a t-test:

```
t.test(kiwi_a,kiwi_b)
```

```
##
##  Welch Two Sample t-test
##
## data:  kiwi_a and kiwi_b
## t = -3.0633, df = 15.703, p-value = 0.007558
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.18155249 -0.03290904
```

```
## sample estimates:
## mean of x mean of y
##  0.982000  1.089231
```

and we can see from the small p-value (0.007558) that we may reject the null hypothesis, and infer that the kiwi weights do in fact differ between the forests.

3.6.2 Paired test

Sometimes we have a slightly different structure in a dataset, in which the elements of the two datasets may be paired. This often occurs when the two datasets represent the same object but before and after some treatment or the passage of time. For example, suppose we collect exam scores for students before and after some instruction, and want to investigate whether the instruction was effective. The data is:

```
before <- c(smith=112, jones=199, robinson=120, taylor=89, williams=100, brown=110)
after  <- c(smith=119, jones=201, robinson=137, taylor=91, williams=104, brown=109)
before
```

```
##    smith    jones robinson   taylor williams    brown
##      112      199      120       89      100      110
```

```
after
```

```
##    smith    jones robinson   taylor williams    brown
##      119      201      137       91      104      109
```

Observe how the different students have very different score, but each one changes a small amount. Because the datasets have a natural pairing with each other (we would compare smith *before* with smith *after*).

There are two natural approaches. The simplest is to consider the difference in scores and perform a t-test on that:

```
t.test(before-after,alternative="less")
```

```
##
##  One Sample t-test
##
## data:  before - after
## t = -1.987, df = 5, p-value = 0.05182
## alternative hypothesis: true mean is less than 0
## 95 percent confidence interval:
##        -Inf 0.07288965
## sample estimates:
```

```
## mean of x
## -5.166667
```

(also, observe that we use a one-sided test: noone expects the instruction to *decrease* exam scores). But this approach effectively assumes that the variance is unaltered by the instruction, which might be incorrect (why?). We can adjust for this by using the *paired* t-test:

```
t.test(before, after, paired=TRUE,alternative="less")
```

```
##
##  Paired t-test
##
## data:  before and after
## t = -1.987, df = 5, p-value = 0.05182
## alternative hypothesis: true mean difference is less than 0
## 95 percent confidence interval:
##        -Inf 0.07288965
## sample estimates:
## mean difference
##       -5.166667
```

3.7 Binomial distribution and testing

The definition of p-value: "the probability, if the null is true, of obtaining the observation or an observation more extreme" applies to a wide variety of null hypotheses and distributions. Consider, for example, bernoulli trials of a coin landing heads or tails. We suspect that the coin is biased towards heads, that is, the probability of success $p > \frac{1}{2}$. Our data is 58 heads out of 100 trials.

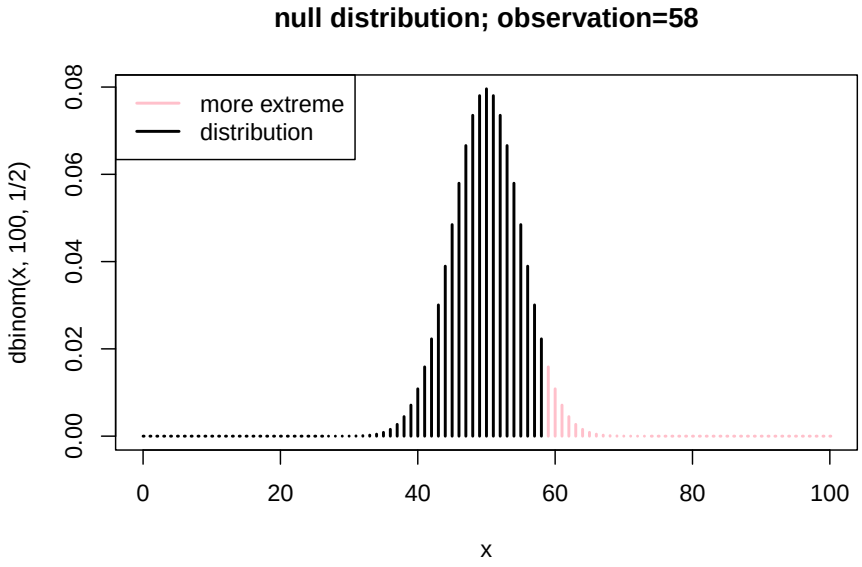
To proceed, we specify a null hypothesis $H_0 : p = \frac{1}{2}$ and try to obtain a p-value. The p-value is directly computable in R:

```
sum(dbinom(58:100,100,1/2))
```

```
## [1] 0.06660531
```

See how the above is a *direct* implementation of the definition of p-value. In this case, the p-value is 6.6% so we fail to reject the null. Visually:

```
x <- 0:100
plot(x,dbinom(x,100,1/2),col=c(rep("black",59),rep("pink",42)),type="h",lwd=2,main="nul
legend("topleft",lty=1,lwd=2,col=c("pink","black"),
        legend=c("more extreme","distribution"))
```



In the above, the red shows the probability, if the null is true, of obtaining the observation (58 in this case), or an observation more extreme (> 58). This is precisely the definition of p-value. Incidentally, the professional would use `pbinom()`:

```
pbinom(57,100,1/2,lower.tail=FALSE)
```

```
## [1] 0.06660531
```

3.7.1 Two-sided binomial tests

The above reasoning applies to two-sided tests. Suppose we have a null of $p = 1/2$ but are not sure whether the true value of p is greater than, or less than, $1/2$. Then a two-sided test would be appropriate because "more extreme" could be interpreted as "further away from half"

Suppose we observed 69 heads. Then "more extreme" would mean either > 69 heads or < 31 heads, these values being equally distant from 50.

The p-value would then be

```
sum(dbinom(c(0:31,69:100),100,1/2))
```

```
## [1] 0.0001831432
```

and the p-value is less than 5% so we reject the null. Observe that in this case the two-sided p-value is exactly twice the one-sided p-value.

3.8 Critical region

There is another view of statistical hypothesis testing, associated with statistician Karl Pearson. In his view we consider the null distribution of our test statistic and regard the tail region of this as a "rejection region". The main difference between this approach and the p-value approach is that we decide *in advance* on a p-value, then report a yes/no finding: did we reject the null, or not? In practice the two approaches are used interchangeably although to be absolutely consistent one should use one or the other and not mix the notations (but everyone does).

3.9 Confidence intervals

https://en.wikipedia.org/wiki/Confidence_interval

The concept of *confidence interval* is a common one in statistics. The formal definition is quite hard, and the interpretation difficult.

First of all: "Interval" is another word for "range". The basic idea is that we are trying to estimate a parameter (for example, the population mean μ). We want to give a *range* of values, and we hope that the true value of our parameter is within this range. We ensure that the probability of this range including the true value of the parameter is 95% (other values such as 99% or 99.5% are sometimes used, but we will stick to 95% here).

To fix ideas, consider the following dataset, representing weights of kiwi birds:

```
kiwi <- c(0.95, 0.56, 0.86, 0.83, 0.88, 0.77, 1.09)
mean(kiwi)
```

```
## [1] 0.8485714
```

Thus the mean value of the kiwi weights is about 0.85kg. We might hypothesise that the population mean μ is 0.9kg, and test this hypothesis using `t.test()`:

```
t.test(kiwi,mu=0.9)$p.value
```

```
## [1] 0.4359164
```

where the p-value has been extracted (from the full output calculated by `t.test()`) using a dollar construction. We can see that we fail to reject the

null that $\mu = 0.9$, on the grounds that the p-value exceeds 0.05. This says that the population mean being 0.9 is somehow consistent with our data. We can now test the hypothesis that $\mu = 1.2$:

```
t.test(kiwi,mu=1.2)$p.value
```

```
## [1] 0.001256824
```

and in this case we may reject the hypothesis that $\mu = 1.2$. It is clear that we reject values for the mean that are far removed from the bulk of the data, and we fail to reject values for the mean that are close to the bulk of the data. The *confidence interval* for the mean is all values x for which we would fail to reject the null hypothesis that $\mu = x$. From the above, we would conclude that 0.9 is *inside* the confidence interval [because we fail to reject the null that $\mu = 0.9$] and 1.2 is *outside* the confidence interval [because we rejected the null that $\mu = 1.2$].

All of this complicated reasoning is carried out by `t.test()`:

```
t.test(kiwi)
```

```
##
##  One Sample t-test
##
## data:  kiwi
## t = 13.771, df = 6, p-value = 9.121e-06
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.6977877 0.9993552
## sample estimates:
## mean of x
## 0.8485714
```

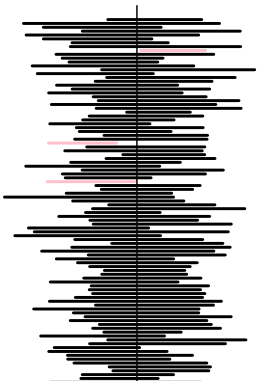
which includes the confidence interval (it doesn't matter here, but the reported p-value refers to a null of $\mu = 0$).

The confidence interval is constructed so that it contains the true value of the population mean with probability 95%. Note carefully that this is not the same as saying "the true value of the mean is contained within this confidence interval with probability 95%". The confidence interval is a random variable, the result of a randomly observed dataset. It is constructed so that it is rare [i.e. probability 5%] for the true value of the mean to be outside the confidence interval.

```
f <- function(n){ as.vector(t.test(rnorm(n))$conf.int) }

set.seed(0)
plot(c(-4,4),c(1,10),axes=FALSE,xlab='',ylab='',type='n')
```

```
for(i in 1:100){
    jj <- f(10)
    if(prod(jj)<0){col <- 'black' } else {col <- 'pink'}
    segments(x0=jj[1],y0=i/10,x1=jj[2],y1=i/10,col=col,lwd=2)
}
abline(v=0)
```



The figure above shows 100 confidence intervals generated from a distribution where the true mean $\mu = 0$. See how the majority of the intervals contain the true value but occasionally (with probability 5%) the interval lies completely to one side of the true value, and such intervals are shown in pink. See how much of the R idiom you can understand.

Note that although confidence interval has a univeral definition, it is not without problems and many workers [including your lecturer!] question the validity of this form of reasoning. Actually, your lecturer harbours serious doubts as to the validity of the entire concept of p-values and indeed that of probability theory itself.

Chapter 4

Type I and type II errors

https://en.wikipedia.org/wiki/Type_I_and_type_II_errors

YouTube https://www.youtube.com/watch?v=-q6KG_ZurcU&index=8&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

In the previous chapter we considered the p-value as an inferential tool. The rule was to reject the null if $p < 0.05$. Here we introduce a different approach which allows us to consider different types of error. We will make heavy use of random sampling from known distributions.

The basic approach is to define a *critical region* and reject the null hypothesis if our observation falls in this critical region. The standard critical region is defined by the observation exceeding a particular critical value. Thus for example we reject H_0 if $\overline{x} > V$ for some V that we choose.

The general idea is that it is *rare* for the observation to land in the critical region if the null is true. So if it does land in the critical region we have a dichotomy: either something rare has occurred, or the null is false.

It is possible to define the critical region as we wish, and in this chapter we discuss desiderata for assessing different critical values. For simplicity we will consider only one-sided tests. If we have observations from $N(\mu, 1)$ [that is, Gaussian with unknown mean μ and standard deviation 1], we may wish to test $H_0 \colon \mu = 0$.

Our test statistics will be $\overline{x}$, the sample mean. We know from previous chapters that, if H_0 is true we will have $\overline{x} \sim N\left(0, \frac{1}{\sqrt{n}}\right)$. Suppose for concreteness that $n = 10$: we have 10 observations. Then we reject H_0 if $\overline{x}$ exceeds a certain

critical value (a one-sided test). We want to ensure that, if the null is true, the null is rejected only 5% of the time, so the critical value is given by:

```
qnorm(0.95,0,1/sqrt(10))
```

```
## [1] 0.5201484
```

we will call this 0.52 for simplicity. Below is the R idiom for testing $H_0 : \mu = 0$; the dollar construction extracts just the p-value:

```
set.seed(0)
t.test(rnorm(10),alternative="greater")$p.value
```

```
## [1] 0.1854769
```

In the above, the null was true (see the help page for `rnorm`). The p-value exceeds the critical value of 0.05, so we fail to reject the null as we should. However, we can repeatedly try the same test with the `replicate()` command:

```
f <- function(n,mean=0){
 t.test(rnorm(n,mean=mean),alternative="greater")$p.value
}
set.seed(0)
replicate(7,f(n=10))
```

```
## [1] 0.1854769 0.9372070 0.3800349 0.2719640 0.6083597 0.6279446 0.8323123
```

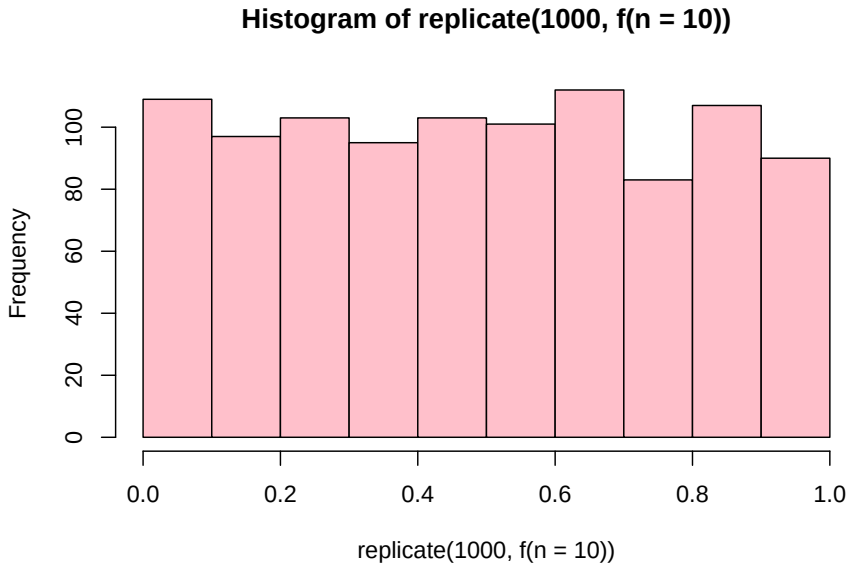
Study the R idiom above carefully. We are repeatedly carrying out a t-test on random data for which the null is *known* to be true. In this case, none of the tests rejected the null. However, we may carry out the test a large number of times:

```
set.seed(0)
table(replicate(1000,f(n=10)) < 0.05)
```

```
##
## FALSE  TRUE
##   952    48
```

In the above, the null is known to be true (the population mean is zero), but we incorrectly reject the null hypothesis 48 times (the `set.seed(0)` command ensures that results are repeatable). Rejecting the null hypothesis when it is true is known as a *type I error*. We want this to be rare. Studying the diagram below:

```
set.seed(0)
hist(replicate(1000,f(n=10)),col='pink')
```



The above demonstration shows that the p-value is uniformly distributed from 0 to 1[1]. We can deduce that the probability of rejecting the null hypothesis—that is, the probability of committing a type I error—is 0.05. We say that the *size* of the test is the probability of committing a type I error and in this case the test is of size 0.05 (because that is the p-value that we selected). In statistics, one usually denotes the size of a test as α.

Observe that we can make the size of the test α any probability we like (by choosing the critical p-value) but it is usually required that the size of any test be less than 5%. For example, we could have a test of size 0.01 by rejecting the null if the p-value is less than 0.01.

4.1 Critical region

The above reasoning suggests that we define a "critical region" for a test statistic. This is usually the tail region of the null distribution of the test statistic, and has a small probability (usually 5%). The idea is that if the test statistic falls in the critical region, we reject the null; observe that if the null is true we reject it only with a small probability. Study the following diagram:

[1]We will see many examples of statistical tests and they all should have a uniform distribution of p-values

```
x <- seq(from=-5,to=5,len=100)
plot(x,dnorm(x),type='n',xlab='sample mean',ylab='probability density')
abline(v=qnorm(0.95))

xx <- seq(from=qnorm(0.95),to=5,len=100)
jj <- c(xx,rev(xx))
polygon(x=jj,y=c(dnorm(xx),xx*0),border=NA,col='gray')
points(x,dnorm(x),type='l',lwd=2,col='black')
legend("topright",col=c("black","gray"),lwd=c(1,10),legend=c("null","critical region"))
```

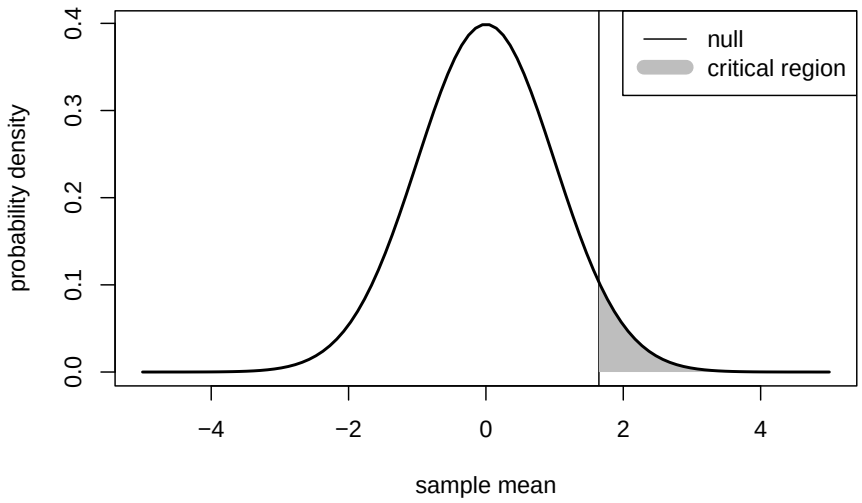


Figure 4.1: A one-sided critical region

In Figure 4.1, the critical region is indicated in grey. We reject the null if our test statistic (the sample mean) falls in the grey region. If the null is true, the probability of rejecting it is 0.05: in other words, the size of the test is 5%

4.2 Type II errors

The type I error has a converse: the failure to reject the null hypothesis when it is false. If the null is false, the correct course of action is to reject it, and failure to do so is an error: we call this a "type II error". A type II error is a sort of inverted version of a type I error.

4.2.1 Power of a test

We usually denote the probability of committing a type II error as β. Note carefully that the value of β depends on the alternative hypothesis we are considering.

The *power* of a test is defined as $1 - \beta$. The power is thus the probability of correctly detecting that the null is incorrect.

4.2.2 Visual representation

The following diagram shows the different types of errors visually. Study the R idiom carefully.

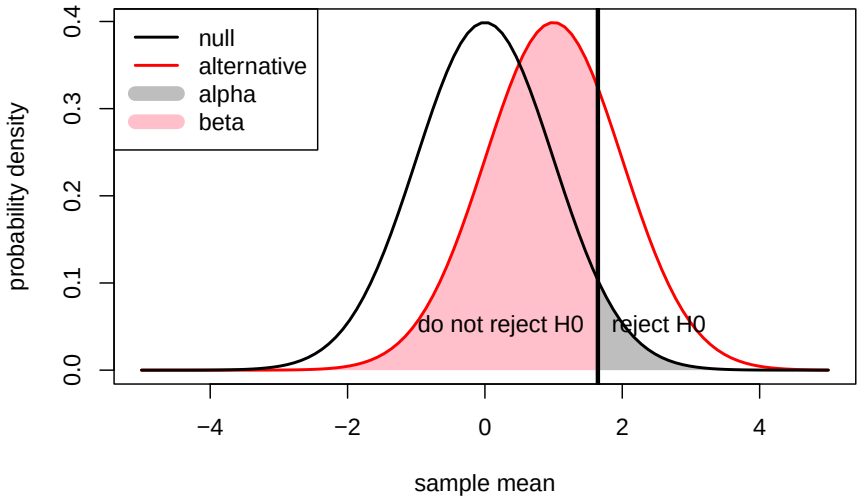
```
x <- seq(from=-5,to=5,len=100)
plot(x,dnorm(x),type='n',lwd=2,xlab='sample mean',ylab='probability density')

xx <- seq(from=-5,to=qnorm(0.95),len=100)
jj <- c(xx,rev(xx))
polygon(x=jj,y=c(dnorm(xx,mean=1),xx*0),border=NA,col='pink')
points(x,dnorm(x,mean=1),type='l',lwd=2,col='red')

xx <- seq(from=qnorm(0.95),to=5,len=100)
jj <- c(xx,rev(xx))
polygon(x=jj,y=c(dnorm(xx),xx*0),border=NA,col='gray')
points(x,dnorm(x),type='l',lwd=2)


text(qnorm(0.95),0.05,'do not reject H0',pos=2)
text(qnorm(0.95),0.05,'reject H0',pos=4)
abline(v=qnorm(0.95),lwd=3)

legend("topleft",
     lwd=c(2,2,10,10),
     col=c("black","red","gray","pink"),
     legend=c("null","alternative","alpha","beta")
     )
```



Type I errors and type II errors have a mutual relationship in that making α smaller forces β to be larger, although the relationship is not simple.

In the limit, one may achieve $\alpha = 0$ by never rejecting the null; but this has the effect of making $\beta = 1$. Similarly, one may have $\beta = 0$ but this will entail $\alpha = 1$.

Again, one usually insists that $\alpha \leq 0.05$ and hope that β is not too big; one standard requirement is that $\beta \leq 0.2$ or equivalently that the power should exceed 0.8.

4.3 Some numerical simulations

In the following we will keep $\alpha = 0.05$ and $H_0 : \mu = 0$ unless stated otherwise. For convenience I will show some earlier work here. First a helper function:

```
f <- function(n,mean=0){
    t.test(rnorm(n,mean=mean),alternative="greater")$p.value
}
```

In the above, argument n is the number of observations to take. Then, if the null is true (that is, $\mu = 0$, or in R idiom, `mean=0`), we check that the probability of committing a type I error is indeed 0.05:

```
set.seed(0)
table(replicate(1000,f(n=6)) < 0.05)
```

```
##
## FALSE  TRUE
##   933    67
```

In the above, the `TRUE` count shows the type I errors: in this case 67 out of 1000, or a little above 5%. Now, we can force the null to be incorrect and assess β:

```
set.seed(0)
table(replicate(1000,f(n=6,mean=1)) < 0.05)
```

```
##
## FALSE  TRUE
##   333   667
```

In the above, the `TRUE` count *correctly* rejects the null, and the `FALSE` count commits a type II error. So β is about 33% and the power is about 67%. It is possible to improve upon the power by taking more observations. Above, we had $n = 6$ observations but now we try $n = 10$:

```
set.seed(0)
table(replicate(1000,f(n=10)) < 0.05)
```

```
##
## FALSE  TRUE
##   952    48
```

```
table(replicate(1000,f(n=10,mean=1)) < 0.05)
```

```
##
## FALSE  TRUE
##    97   903
```

In the above, the size (α) of the test is fixed at 5% but the power is increased to over 90% as this is the proportion of tests which correctly reject the null. This is what more observations buys you.

4.3.1 Changing the alternative

If we keep $n = 10$ we can investigate the effect of changing the alternative. In the previous section we had an alternative $H_A : \mu = 1$ but suppose we try $H_A : \mu = 0.5$. We might expect that the power would decrease on the grounds

that there is less difference between the null and alternative than previously, and this is indeed the case:

```
set.seed(0)
table(replicate(1000,f(n=10)) < 0.05)
```

```
##
## FALSE  TRUE
##   952    48
```

```
table(replicate(1000,f(n=10,mean=0.5)) < 0.05)
```

```
##
## FALSE  TRUE
##   582   418
```

In the above, we can see from the first table that that size α is indeed about 0.05; the second table shows that the power—that is, the probability of correctly rejecting the null when incorrect—is about 42%.

4.3.2 Changing the size of the test.

Earlier I said that type I and type II errors were exchangeable in the sense that increasing α decreases β. Here I show that this is true using numerical simulation. We will return to $H_A : \mu = 1$.

```
set.seed(0)
table(replicate(1000,f(n=10)) < 0.05)
```

```
##
## FALSE  TRUE
##   952    48
```

```
table(replicate(1000,f(n=10,mean=1)) < 0.05)
```

```
##
## FALSE  TRUE
##    97   903
```

```
table(replicate(1000,f(n=10)) < 0.01)
```

```
##
## FALSE  TRUE
##   987    13
```

```
table(replicate(1000,f(n=10,mean=1)) < 0.01)
```

```
##
## FALSE  TRUE
##   363   637
```

In the above, I have changed the size of the test from 0.05 in the first two lines to 0.01 in the second. See how α changes from about 5% to 1%, while β increases from about 10% to about 36%.

4.4 Exact analysis of type I and type II errors.

Suppose we draw n observations from $N(\mu, 1)$, which is to say that they are Gaussian with unknown mean μ and standard deviation 1. We consider a one-sided test $H_0 \colon \mu = 0$ with size α against an alternative hypothesis $H_a \colon \mu = x$, and want to investigate how the power depends on n and α. The power function would be given by

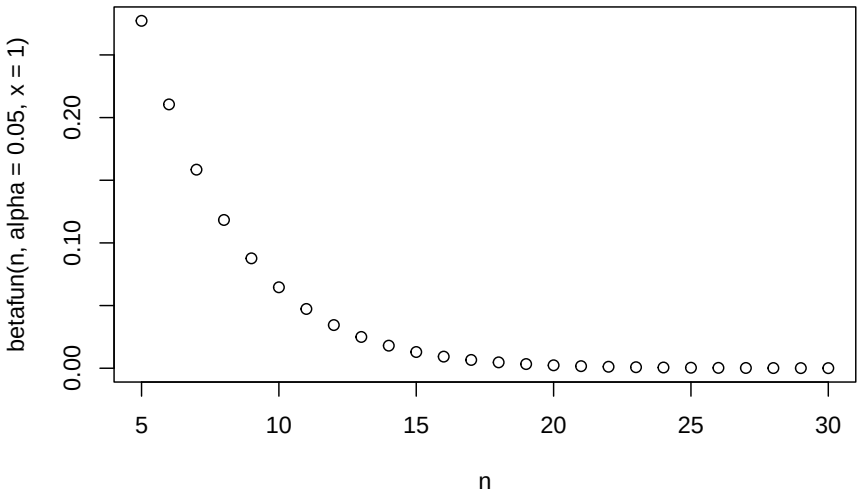
```
betafun <- function(n,alpha,x){
pnorm(qnorm(1-alpha,mean=0,sd=1/sqrt(n)),mean=x,sd=1/sqrt(n))}
```

Study the R idiom above carefully. See how β is given by `pnorm()`, which is the area to the left of its first argument: the probablity of correctly rejecting the null. The first argument to `pnorm()` is just the critical point of the test. The critical point is the value at which we reject the test with probabilty α [and therefore fail to reject with probability $1-\alpha$], which is given by `qnorm()`. Both functions use a standard deviation of $\frac{1}{\sqrt{n}}$, as this is the standard error of the mean; but observe that the null distribution has `mean=0` while the alternative has `mean=x`, as this may be altered.

4.4.1 Type II errors: β as a function of n, the number of observations.

We can plot the power as a function of n, the number of observations. In the R idiom below, we use the standard value of $\alpha = 0.05$ which is here held constant.

```
n <- 5:30
plot(n,betafun(n,alpha=0.05,x=1))
```

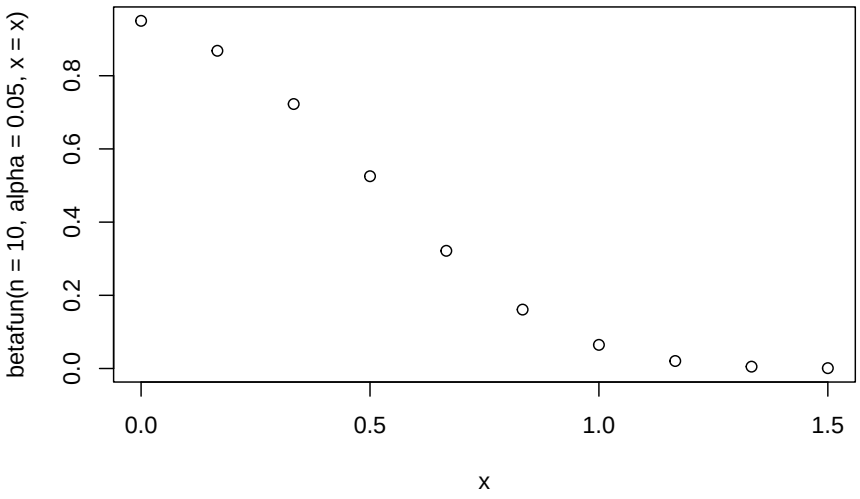


In the above figure, see how β decreases with n for fixed $\alpha = 0.05$ and $H_A : \mu = 1$: the probability of committing a type II eror becomes smaller with as n increases.

4.4.2 Type II errors: β as a function of x, the mean of the alternative distribution

Here we fix $n = 10$ and $\alpha = 0.05$, and show how β varies with the mean of the alternative hypothesis.

```
x <- seq(from=0,to=1.5,len=10)
plot(x,betafun(n=10,alpha=0.05,x=x))
```

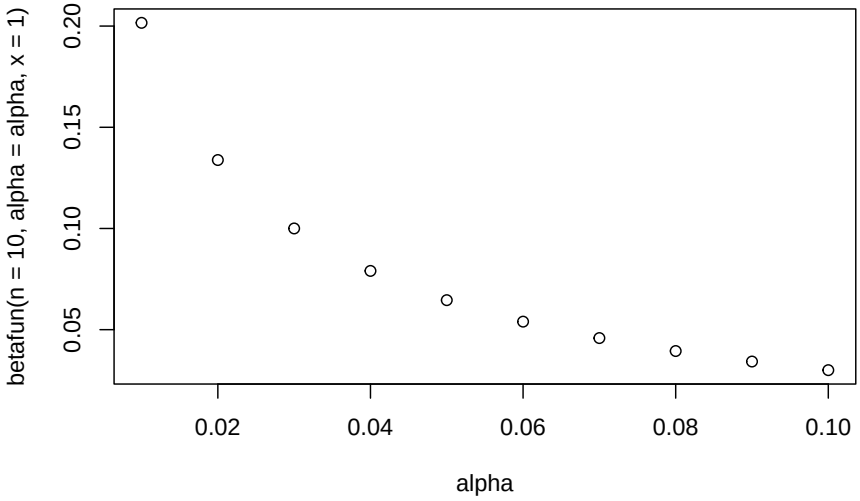


In the above figure, see how β decreases with x (the alternative mean) for fixed $\alpha = 0.05$ and $n = 10$: the probability of committing a type II eror becomes smaller with as x increases.

4.4.3 Type II errors: β as a function of α, the size of the test

Here we fix $n = 10$ and $\alpha = 0.05$, and show how β varies with the mean of the alternative hypothesis.

```
alpha <- seq(from=0.01,by=0.01,to=0.1)
plot(alpha,betafun(n=10,alpha=alpha,x=1))
```

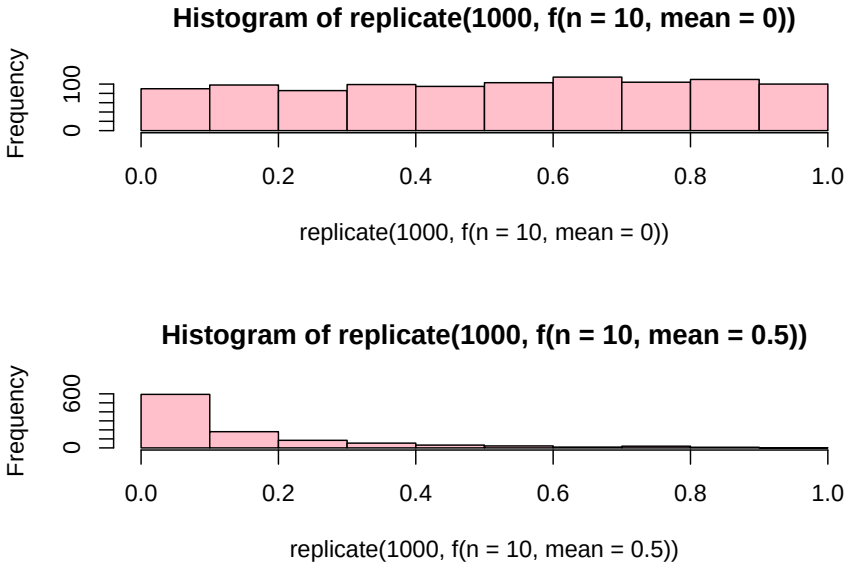


In the above figure, see how β decreases with α (the size of the test) for fixed $n = 10$ and $x = 1$: the probability of committing a type II eror becomes smaller with as α increases.

4.5 The distribution of the p-value

We will now show how the distribution of p-values changes depending on the null and alternative hypothesis.

```
f <- function(n,mean=0){ t.test(rnorm(n,mean=mean),alternative="greater")$p.value}
par(mfrow=c(2,1))     # makes R plot two histograms in the same figure
hist(replicate(1000,f(n=10,mean=0)),col='pink')    # null is true
hist(replicate(1000,f(n=10,mean=0.5)),col='pink') # null is false
```



In the above diagram, the top histogram shows the distribution of p-values when the null is true: a uniform distribution between 0 and 1. The lower histogram shows the distribution of p-values when the null is incorrect and the mean is 0.5. See how the p-values are shifted towards zero: there is—as there should be—a higher probability of rejecting the null when it is incorrect.

Chapter 5

Point estimation

So far we have been considering distributions with *known* parameters. In the case of the Gaussian distribution we have the mean μ and standard deviation σ. However, in practice we very frequently do not know the true values of a parameter. Consider the binomial distribution $B(n, p)$ with known n but *unknown* p. We wish to estimate p from observational data.

Suppose $n = 100$ and we observe $r = 35$ successes. We might estimate that $p = 0.35$, on the basis that 35% of the 100 trials were successes. However, observe carefully that, if p really was 0.35, then we would not observe exactly 35 successes:

```
rbinom(20, 100, 0.35)
```

```
##  [1] 37 38 30 41 31 28 34 42 30 32 35 41 38 36 31 40 30 28 38 30
```

Recall from the previous chapter that the observations have a mean of $np = 100 \times 0.35 = 35$, and a variance of $100 \times 0.35 \times (1 - 0.35) = 22.75$.

However, observe that if the true value of p was 0.36 then we might well observe 35 successes. Thus the observation of 35 successes does not allow us to distinguish between the two possibilities that $p = 0.35$ and $p = 0.36$.

However, the observation of 35 successes means that we would be very confident in rejecting the suggestion that $p = 0.99$. It is not at all obvious why $p = 0.99$ is "unacceptable" in this sense while $p = 0.35$ is OK. What we want to do is to find a "best estimate" for the value of p on the basis of our observation; but also to indicate a range of uncertainty for our estimate.

5.1 Likelihood

https://www.youtube.com/watch?v=yeduCyob7DY&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=23

Suppose three people, A, B, and C are discussing the observation of 35 successes out of 100 trials, and wish to make inferences about p, the unknown probability of success. Person A says that $p = 0.2$, person B says that $p = 0.3$, and person C says that $p = 0.5$.

To compare these three estimates for p, it is natural (after a while) to calculate the probability of making the observation [of 35 successes out of 100 trials], given that p takes in turn each of the values specified by the three people. This is easy in R:

```
dbinom(35,100,p=0.2)
```

```
## [1] 0.0001889469
```

```
dbinom(35,100,p=0.3)
```

```
## [1] 0.04677968
```

```
dbinom(35,100,p=0.5)
```

```
## [1] 0.0008638557
```

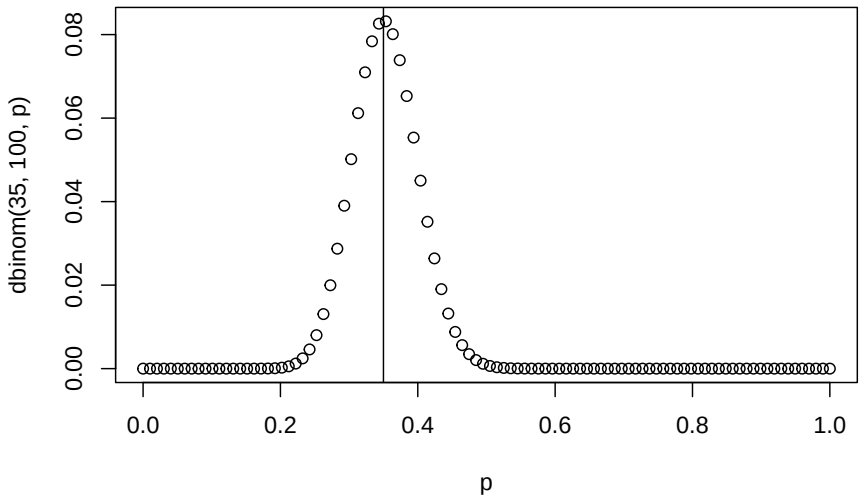
Think about what this means. Take the first line: if the probability truly was 0.2, then the probability of actually seeing the data [35 out of 100 trials] is 0.00019. If it truly was 0.3, the probability of seeing the data is about 0.047, and if it truly is 0.5, the probability would be 0.00086. Of these three possibilities, 0.3 is preferable because this has the highest probability. We say that $p = 0.3$ is more *likely* than the other two guesses.

5.2 The likelihood function

https://en.wikipedia.org/wiki/Likelihood_function

The above line of reasoning can be streamlined and extended by allowing p, the supposed probability of success, to be *any* value between zero and one. It makes sense to plot the probability of observing the data [35 successes out of 100 trials] as a function of the supposed probability of success. This is easy in R:

```
p <- seq(from=0, to=1, len=100)
plot(p,dbinom(35,100,p))
abline(v=0.35)
```



The above plot shows a *likelihood* function: the probability of seeing the data, as a function of the assumed probability of success. We can see that the likelihood function is maximized at $p = 0.35$[1]. The graph also shows why $p = 0.9$ is a poor estimate: the likelihood is much smaller than the likelihood for $p = 0.35$.

The *likelihood function* is a general-purpose tool used in many branches of statistical inference. It is used when we are considering a statistical distribution that has one or more unknown parameters. In this example, the unknown parameter is the probability p of success. The likelihood function is a function $\mathcal{L}(\cdot)$ of the unknown parameter, with $\mathcal{L}(p) = \text{Prob}(D|p)$. In English we say "the probability of seeing the data D, given the true value of the parameter is p". Sometimes we call the parameter the "hypothesized value", or 'hypothesis" for short and write $\text{Prob}(D|H)$.

The formal definition of likelihood function is a little harder: We may multiply the likelihood function as described above by an arbitrary positive value C and come to the same conclusions: it is only *relative* changes in likelihood that matter, not the absolute value. For the example of 35 successes out of 100 trials the likelihood function is

[1]This fact can be verified by using differential calculus but this course is not using calculus

$$\mathcal{L}(p) = C.\binom{100}{35} p^{35}(1-p)^{65} \tag{5.1}$$

The combinatorical term $\binom{100}{35}$ is just a constant, and such constants can be very difficult to evaluate. But we can choose the constant C to be any positive number, and if we choose $C = \frac{1}{\binom{100}{35}}$, then the combinatorial term in the binomial cancels out and we are left with

$$\mathcal{L}(p) = \frac{1}{\binom{100}{35}} \binom{100}{35} p^{35}(1-p)^{65} = p^{35}(1-p)^{65} \tag{5.2}$$

which is easier to deal with[2]. In R it is just

```
function(p){p^35*(1-p)^65}
```

We will see many examples where the arbitrary constant C saves us a huge amount of mathematical difficulty. Later when we consider Bayesian analysis we will see how this line of reasoning allows us to pursue a different form of statistical inference.

5.3 Likelihood functions for the Gaussian

Another nice example of likelihood functions is given by the Gaussian distribution. Suppose we have observations drawn from a Gaussian distribution $N(\mu, 1)$: so we know that the standard deviation is 1, but we do not know what the mean μ is, and want to make inferences about its true value.

Our data is:

```
d <- c(6.1, 6.7, 6.3, 5.7)
```

Of course we could calculate the *sample* mean:

```
mean(d)
```

```
## [1] 6.2
```

but what is the uncertainty on this? The likelihood function is, in R idiom, easy to calculate. The idea is that the *data* stays fixed but allow the putative mean μ to vary. Suppose we start with $\mu = 5.8$. Then the likelihood would be

[2]Actually, "*the* likelihood function" is a slight misnomer, because it is defined only up to a multiplicative constant. We really should say "*a* likelihood function", by which we mean any function of H that is proportional to $\text{Prob}(D|H)$.

```
dnorm(6.1,mean=5.8)*
dnorm(6.7,mean=5.8)*
dnorm(6.3,mean=5.8)*
dnorm(5.7,mean=5.8)
```

```
## [1] 0.01418239
```

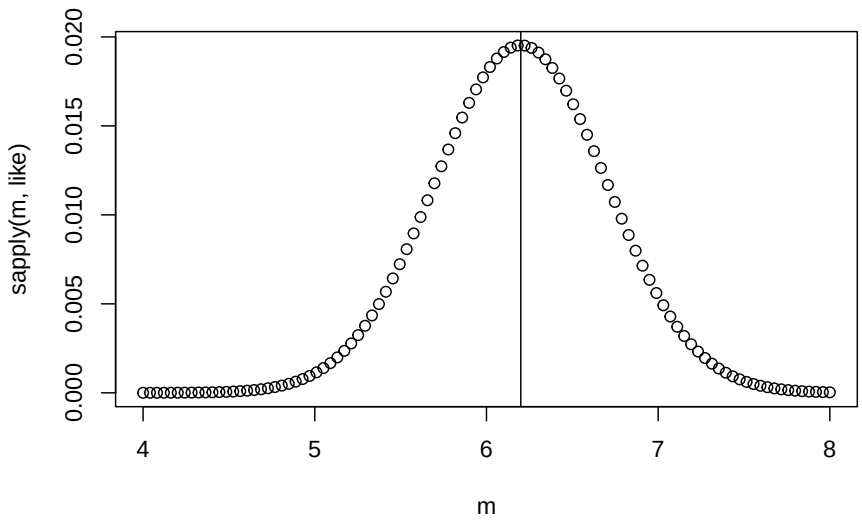
and we might compare this with the likelihood for $\mu = 5.9$:

```
dnorm(6.1,mean=5.9)*
dnorm(6.7,mean=5.9)*
dnorm(6.3,mean=5.9)*
dnorm(5.7,mean=5.9)
```

```
## [1] 0.01631363
```

Comparing these two, we see that $\mu = 5.9$ is marginally more likely than $\mu = 5.8$ on the basis that the likelihood is (slightly) higher. We can plot a likelihood function:

```
like <- function(m){prod(dnorm(d,mean=m))}
m <- seq(from=4,to=8,len=100) # range of plausible population means
plot(m,sapply(m,like))        # see below for why sapply() is used
abline(v=mean(d))             # sample man is the maximum likelihood estimator
```



Study the above carefully: it might look simple, but there is a lot going on. Make sure you understand exactly what is happening before moving on. Note the use of the `sapply()` construction here, needed to vectorize the `like()` function. See how the likelihood is maximized at the *sample* mean; this is *why* the sample mean is calculated.

5.4 The support function

The likelihood function $\mathcal{L}(\cdot)$ is defined as the probability of obtaining the data, given the hypothesis H, multiplied by an arbitrary constant C.

The *support* function $\mathcal{S}(\cdot)$ is defined as the logarithm of the likelihood:

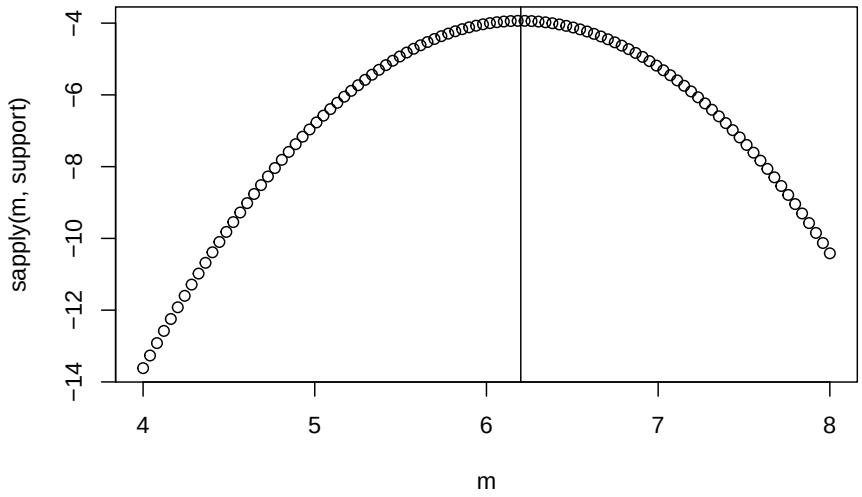
$$\mathcal{S}(H) = \log(\mathcal{L}(H)) \tag{5.3}$$

But because the likelihood has an arbitrary multiplicative constant C, the formula is

$$\mathcal{S}(H) = \log(\text{Prob}(D|H)) + \log(C) \tag{5.4}$$

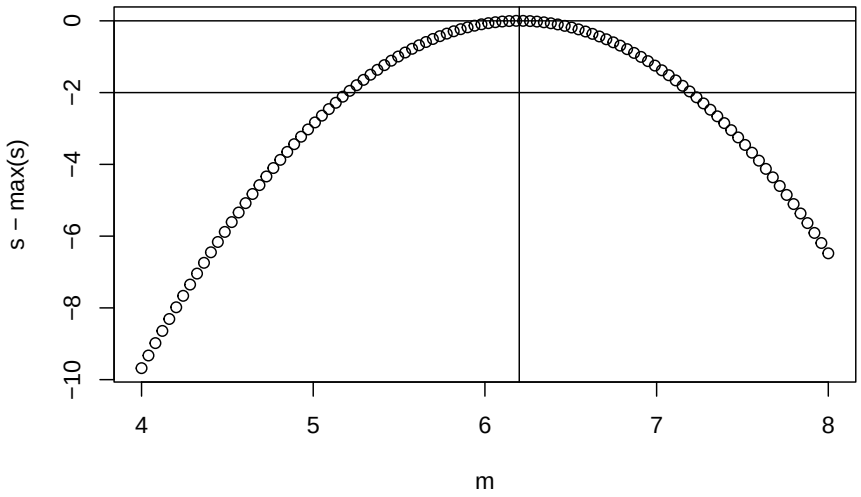
Thus support is a function defined up to an *additive* constant (all logs are natural logs). Using the Gaussian example in the previous section we can plot a support function:

```
support <- function(m){sum(dnorm(d,mean=m,log=TRUE))}  # support function
m <- seq(from=4,to=8,len=100)     # range of plausible population means
plot(m,sapply(m,support))         # plot the support
abline(v=mean(d))                 # sample mean is the best-supported value
```



Remembering that we can add or subtract an arbitrary constant, we may as well ensure that our curve has a maximum at zero:

```
support <- function(m){sum(dnorm(d,mean=m,log=TRUE))}
m <- seq(from=4,to=8,len=100)  # range of plausible population means
s <- sapply(m,support)         # calculate support
plot(m,s-max(s))               # subtract maximum value so support=0 at max
abline(v=mean(d))              # best-supported value is the sample mean
abline(h=0)                    # maximum support = 0
abline(h=-2)                   # two units of support gives credible interval
```



5.4.1 Credible interval

The two horizontal lines correspond to the maximum likelihood estimate for the mean, and the line of $\mathcal{S} = -2$. Two units of support is the standard measure of "a lot of support", so this gives a region of reasonably well-supported values of the mean that are above the line $\mathcal{S} = -2$; this is known as a *credible interval*. From the graph, the credible interval is from about 5.2 to 7.1.

The way to think about this is to ask how much "extra" support one can achieve from any given starting point. Suppose one started at $\mu = 4$. On the graph above, we can see that this has a support of about -10, which would correspond to a likelihood of $e^{-10} \simeq 4.54 \times 10^{-5}$. Then by moving from $\mu = 4$ to $\mu = 6.2$ we can slide up the support curve to the maximum point, and thereby achieve an *extra* 10 units of support. This is more than the 2-units-of-support criterion, so we can be reasonably sure that $\mu = 4$ is not the true value; but $\mu = 6$ is acceptable.

The point at which the support curve is maximized is the "maximum likelihood estimate", also sometimes called the "evaluate".

5.5 Note on likelihood and support as relative measures of credibility

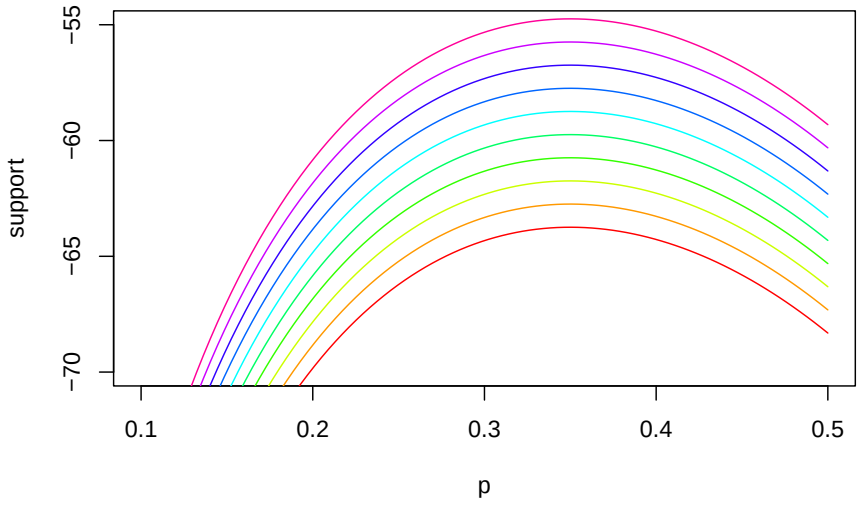
The arbitrary constant C in the definitions of support and likelihood reminds us that only *relative* values of support and likelihood are meaningful. You cannot

calculate "the support" of any single hypothesis; the concept is only meaningful when comparing two hypotheses.

Going back to our example of 35 successes out of 100 trials, we have been calculating the likelihood as $p^{35}(1-p)^{65}$, and the support will be $35\log p + 65\log(1-p)$. However, as stated above we are only interested in relative changes of support, and relative support support is unchanged by addition of an arbitrary constant C. The diagram below shows a number of support curves:

```
p <- seq(from=0.1,to=0.5,len=100)  # set up horizontal axis
plot(p,p*0,type='n',ylab='support',ylim=c(-70,-55))  # set up axes

for(i in 1:10){  # loop over i
  points(p,35*log(p)+65*log(1-p)+i,col=rainbow(10)[i],type='l')  # plot curves
}
```



Observe how the actual value of the y-axis is immaterial. One would make the same inferences from any of the support curves shown, which is why the arbitrary constant $\log C$ may be added to the support function.

5.6 Bias

https://en.wikipedia.org/wiki/Bias_of_an_estimator

Suppose we want to estimate the standard deviation σ of a population from $n = 5$ observations from a standard Gaussian $N(0, 1)$, and we use R idiom `sd()` to do so:

```
sd(rnorm(5))
```

```
## [1] 0.6065003
```

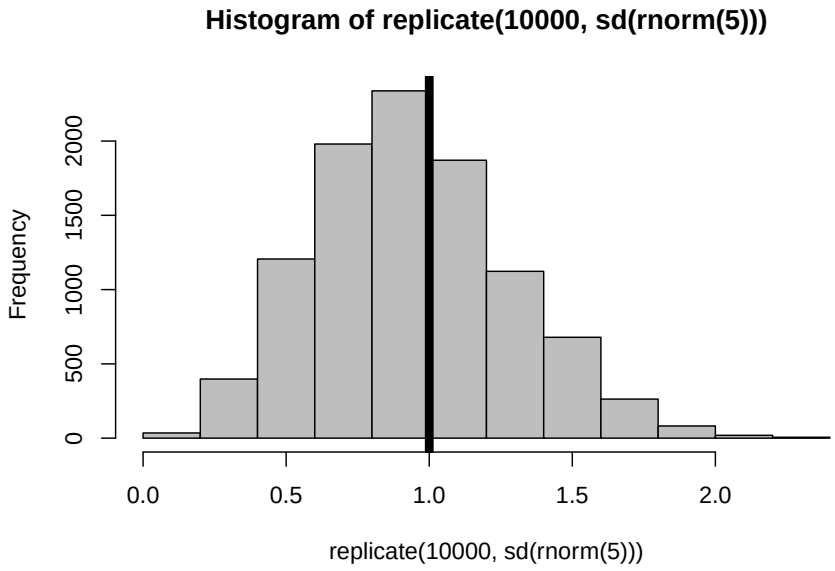
Observe that in the above, we actually know that $\sigma = 1$ but the estimate is not exactly correct, due to sampling error. We might ask what happens if we repeatedly estimate σ:

```
replicate(30,var(rnorm(5)))
```

```
##  [1] 0.2636332 0.6244827 2.2733764 0.7039248 0.7667110 2.7024787 1.9502045
##  [8] 0.9122648 0.2518953 0.4264528 0.5123239 0.7174054 0.9720719 0.4239559
## [15] 1.3792717 0.7113114 0.4252085 0.2275198 0.5699558 0.4307749 1.2283147
## [22] 1.4652956 1.3626886 0.1868831 1.0105077 1.5244497 1.7727549 0.7138863
## [29] 3.1563287 1.2159171
```

In the above, observe that some figures are overestimates and some are underestimates. Of course, in practice we only have a single number to work with, and in general we will not know whether it is an overestimate or an underestimate. Statisticians use a "hat" over a symbol to denote an estimate; thus "$\hat{\sigma}$" means an estimated value of σ. Using R, we can draw a histogram of estimates $\hat{\sigma}$:

```
hist(replicate(10000,sd(rnorm(5))),col='gray')
abline(v=1,lwd=6)
```



in the above we have drawn the correct value of the standard deviation $\sigma = 1$ for convenience. See how the distribution of estimates includes severe overestimates and severe underestimates, both caused by the finite sample size (5 in this case). We might ask what the *expected* value is of $\hat{\sigma}$:

```
mean(replicate(10000,sd(rnorm(5))),col='gray')
```

```
## [1] 0.9379363
```

So the mean of the estimates $\overline{\hat{\sigma}} = 0.93$ is quite far from the true value of $\sigma = 1$. The *bias* of an estimator is defined as the difference between the expected value of the estimator and the true value of the parameter. The bias of `sd()` is difficult to calculate exactly but in this case its numerical value is about $0.93 - 1.00 = -0.07$.

5.6.1 Bias for variance

(this is harder than the preceding material and should be viewed as optional reading). We might estimate the variance of a distribution by calculating the mean value of the squared deviance:

```
myvar <- function(x){
  deviance <- x-mean(x)
  return(mean(deviance^2))
}
```

This would correspond to

$$\frac{1}{n}\sum_{i=1}^{n}(x_i-\overline{x})^2$$

and we can see the bias of this estimator:

```
mean(replicate(10000,myvar(rnorm(5))))
```

```
## [1] 0.7930985
```

If we denote the value of `myvar(d)` by M, it can be shown that $\mathbb{E}(M)=\frac{n-1}{n}\sigma^2$. We can correct for this by defining $\tilde{M}=\frac{n}{n-1}M$ which is why you see $n-1$ (instead of n) as a denominator in the literature. Your lecturer has spent his entire life thinking about bias of estimates. He does not think that bias is a useful or informative thing to calculate and believes that pursuit of unbiased (that is, a bias of zero) estimators is pointless, counter-productive and indeed actively misleading.

Chapter 6

The Poisson distribution

https://en.wikipedia.org/wiki/Poisson_distribution

https://www.youtube.com/watch?v=KeN73vSsbDg&index=12&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

https://www.youtube.com/watch?v=VrEGI6Om3Sg&index=13&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

We have considered what happens to the binomial distribution in the limit of large n and constant p: the distribution approaches a Gaussian with mean np and variance $np(1-p)$. In this chapter we will consider a different limiting process, in which we allow n to grow very large, but require p to become small in such a way as to hold the product np constant. Consider table 6.1, showing values of n becoming larger and larger, and p adjusted so that the product np is always equal to 4. See how the variance rapidly approaches 4 as n approaches infinity. We can get a visual impression of the limiting process by examining the following diagram which shows a range of n:

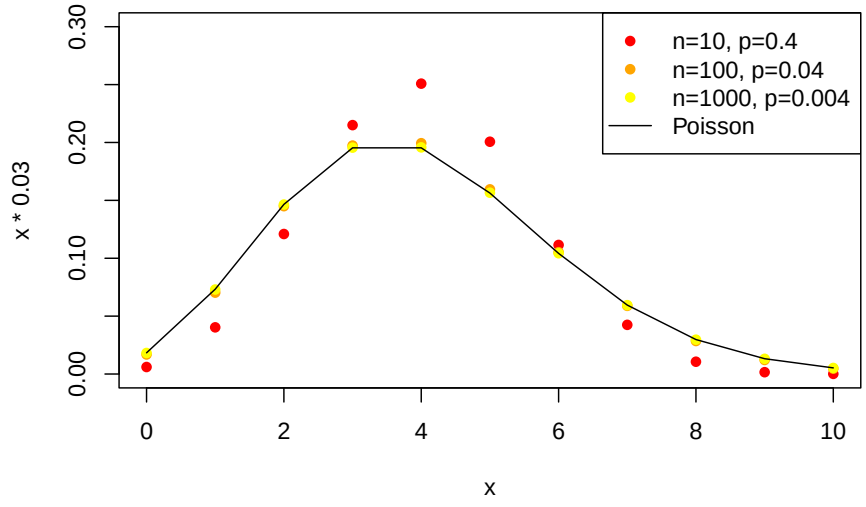
```
e <- 1:5
jj <- data.frame(n=10^e, p=0.4/10^e, np=4, var=4*(1-1/10^e))
knitr::kable(jj,caption="limiting process for the Poisson distribution")
```

```
n <- 10^(1:3)
x <- 0:10
```

Table 6.1: limiting process for the Poisson distribution

n	p	np	var
1e+01	4e-02	4	3.60000
1e+02	4e-03	4	3.96000
1e+03	4e-04	4	3.99600
1e+04	4e-05	4	3.99960
1e+05	4e-06	4	3.99996

```
plot(x,x*0.03,type='n')
cols <- c("red","orange", "yellow", "green", "blue", "purple")
for(i in seq_along(n)){
    points(x,dbinom(x,n[i],4/n[i]),pch=16,col=cols[i])
}
points(x,dpois(x,4),type='l')
legend("topright",
       legend=c("n=10, p=0.4","n=100, p=0.04", "n=1000, p=0.004","Poisson"),
       pch=c(16,16,16,NA),col=c(cols[seq_along(n)],"black"),lty=c(NA,NA,NA,1))
```



The diagram above includes "Poisson" as the result of the limiting process. The Poisson distribution arises naturally whenever we consider a binomial distribution with large n, small p, and moderate np. Examples would include the number of car accidents on any given day (here n would be the number of drivers

and p the (small) probability of having an accident); number of goals scored in a football match (here n would be the number of minutes in a match and p the small probability of scoring a goal in any given minute); or the number of calls received in one hour at a call center (n the number of potential callers, and p the small probability of any given caller calling the center in the hour in question).

The Poisson distribution has only one parameter, λ, which in our limiting case is the value of the product np. The probability mass function of the Poisson is

$$\text{Prob}\,(X = n) = e^{-\lambda}\frac{\lambda^n}{n!}, \qquad n = 0, 1, 2, ... \tag{6.1}$$

where $\lambda > 0$ is the parameter of the distribution.

6.1 The Poisson distribution in R

In the following, we will investigate the Poisson distribution with $\lambda = 4.5$ in R. The relevant functions are `dpois()`, `rpois()`, etc; remember to consult the R help pages for reference. First of all, we will sample from the distribution:

```
rpois(40,lambda=4.5)
```

```
##  [1]  5  5  4  5  2  1  7  3  4  6  9  4  7  4  2  2  5  6 12  7  5  4  3  3  2
## [26]  4  6  2  5  4  4  3  4  5  4  4  3  6  4  4
```

See how random observations from the Poisson are non-negative integers. We can verify that the probabilities of the Poisson add up to 1:

```
sum(dpois(0:100,lambda=4.5))
```

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

(mathematically, the above will be slightly less than 1 because we are not including numbers over 100; but the difference will be small). The mean of the Poisson distribution is known to be λ (this is inherited from the binomial), and we can verify this numerically:

```
mean(rpois(1e6,lambda=4.5))
```

```
## [1] 4.498977
```

```
mean(rpois(1e6,lambda=4.5))
```

```
## [1] 4.500239
```

```
mean(rpois(1e6,lambda=4.5))
```

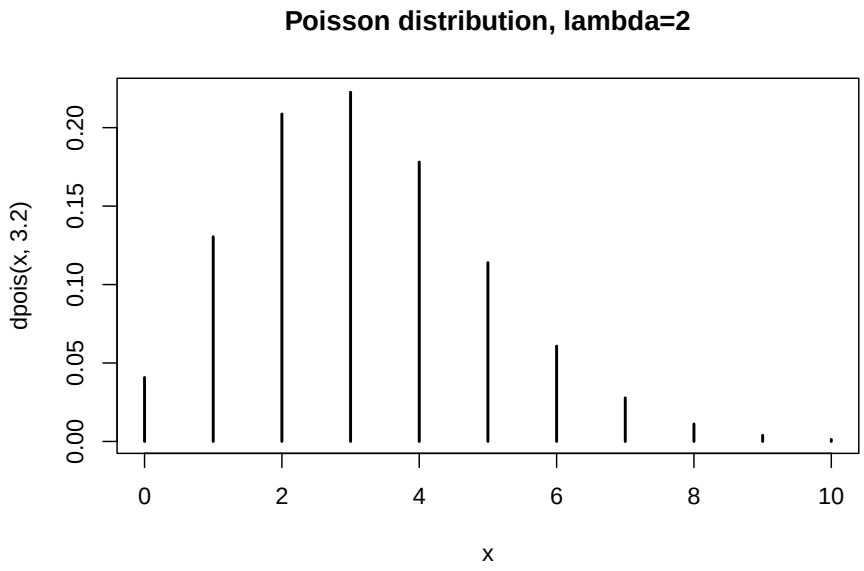
```
## [1] 4.50157
```

In the above, see how the sample mean is very close to the value of $\lambda = 4.5$; the small differences are due to random variability.

6.2 Some examples

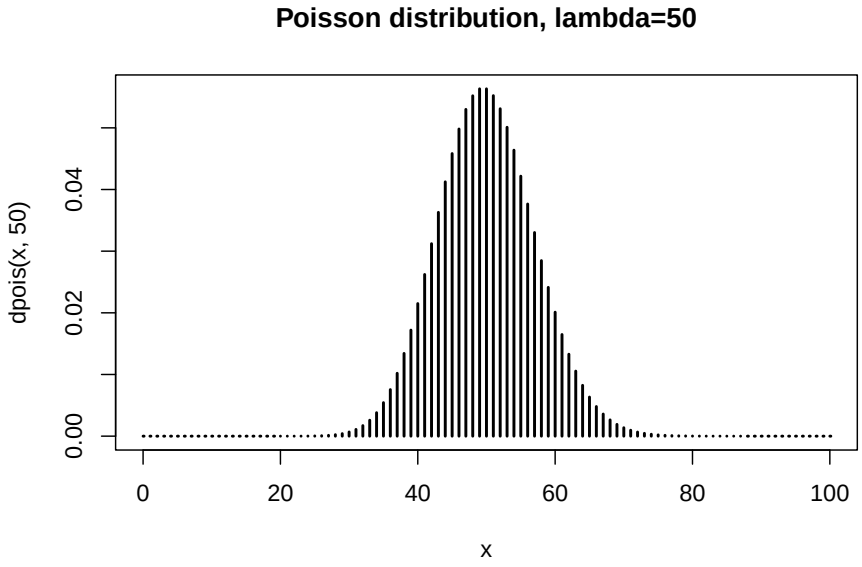
Here we plot a couple of histograms of Poisson distributions.

```
x <- 0:10
plot(x,dpois(x,3.2),type='h',lwd=2,main='Poisson distribution, lambda=2')
```



The Poisson distribution has a characteristic shape, slightly left skewed.

```
x <- 0:100
plot(x,dpois(x,50),type='h',lwd=2,main='Poisson distribution, lambda=50')
```



The Poisson distribution is approximately Gaussian for large values of λ, with Pois(λ) being close to a Gaussian with mean λ and standard deviation $\sqrt{\lambda}$.

6.3 Estimation of the parameter λ

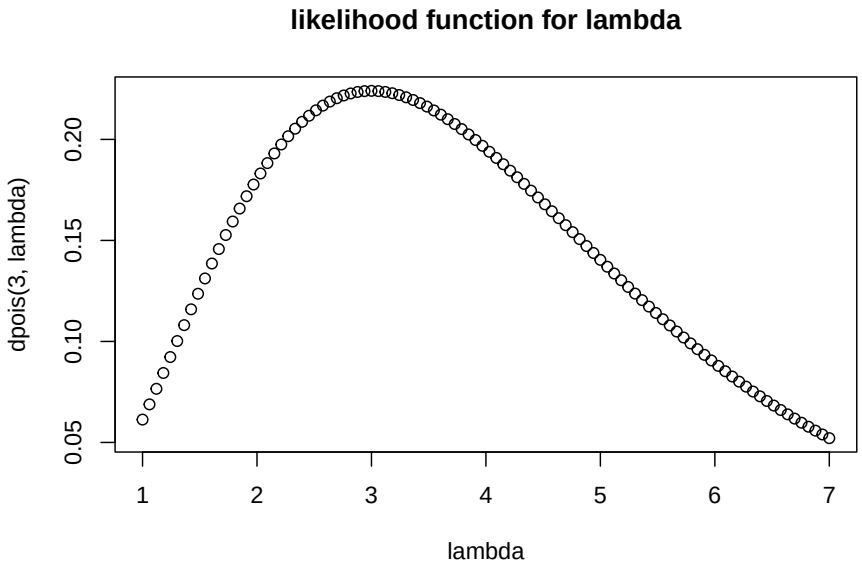
In the above, we have tacitly assumed that the value of λ is known. However, in practice we have to estimate it and likelihood is an easy technique to use.

Recall that likelihood is defined as the probability of seeing the data, given the hypothesis (multiplied by an arbitrary constant). Suppose we observe $n = 3$ counts. Then a likelihood function for λ is just `dpois(3,lambda)`.

Think about that for a moment. It is anodyne, boringly self-evident, and almost trivial; yet it it is simultaneously the most profound statement in the whole of this course.

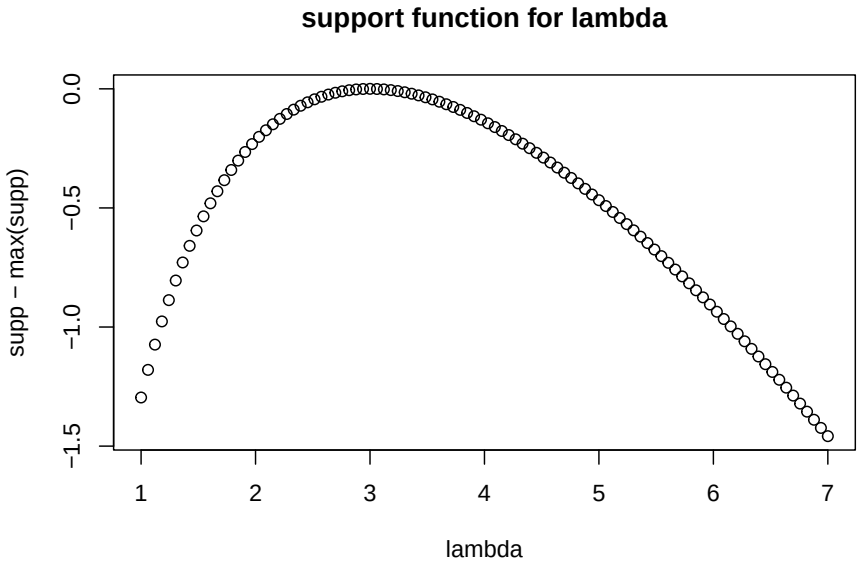
It is easy to plot a likelihood curve:

```
lambda <- seq(from=1,to=7,len=100)
plot(lambda,dpois(3,lambda),main='likelihood function for lambda')
```



and a support curve is easy too:

```
lambda <- seq(from=1,to=7,len=100)
supp <- dpois(3,lambda,log=TRUE)
plot(lambda,supp-max(supp),main='support function for lambda')
```



(in the above, I subtract `max(supp)` so the curve maxes out at zero). See how the maximum likelihood estimate for λ is just 3, the observation (in general, the MLE is equal to the observation).

Chapter 7

Bayes's theorem

https://en.wikipedia.org/wiki/Bayes%27_theorem

IMPORTANT:

- "Bayes theorem": incorrect; no possession marker
- "Bayes' theorem": incorrect (American usage)
- "Bayes's theorem": correct British English

Do not get this wrong.

https://www.youtube.com/watch?v=-q6KG_ZurcU&index=8&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

https://www.youtube.com/watch?v=AK3h_4LPl_Q&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=9

https://www.youtube.com/watch?v=ib7juP-ZBNk&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=10

https://www.youtube.com/watch?v=CxORosyXS5A&index=11&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

In this chapter I will discuss a totally different philosophy for statistics, called the *Bayesian* approach. This is very different from the hypothesis testing material presented above; it is arguably more consistent and coherent than the frequentist approach (null hypotheses and type I/type II errors; confidence intervals).

One feature of modern statistics is that statisticians are often expected to declare themselves as either "Bayesians" or "frequentists", depending on whether or not they buy into the Bayesian philosophy. My estimate would be that 30% of statisticians are Bayesians.

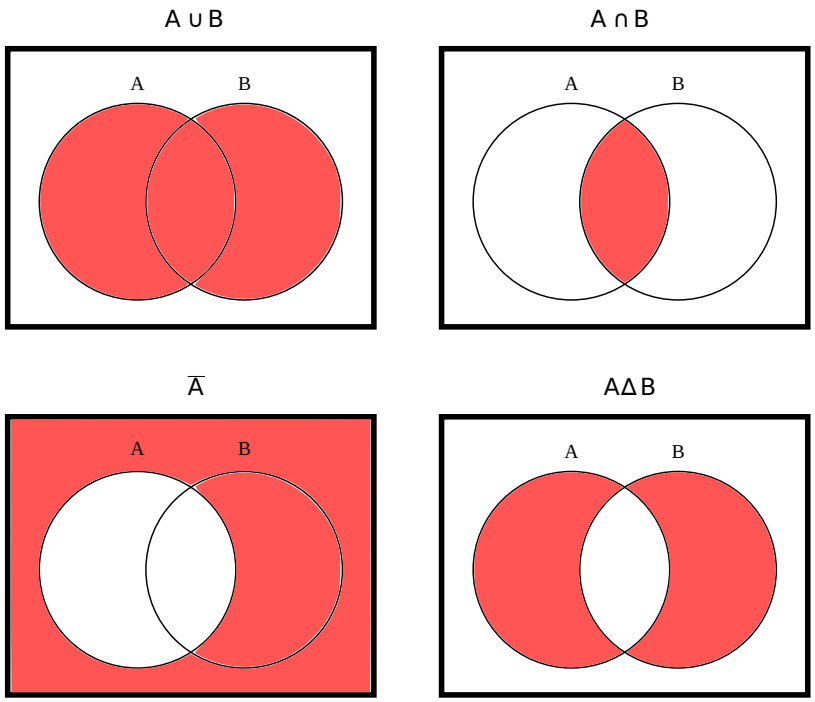


Figure 7.1: Two sets, A and B on a Venn diagram together with red areas marking set union $(A \cup B)$, set intersection $(A \cap B)$, complement of A $(\overline{A})$ and symmetric difference $(A \Delta B)$

In figure 7.1 we see four set diagrams. Top left, set union $A \cup B$; top right; set intersection $A \cap B$; lower left, the complement of A, $\overline{A}$, that is, everything not in A; lower right, symmetric difference $A \triangle B$. Think of A and B being events with specific probabilities, a good example might be A="it is raining" and B="Robin cycles to work". Note that Robin very rarely cycles to work if it is raining (he gets the bus instead). We can define probabilities Prob (A) and Prob (B). We can also define Prob $(A|B)$ as the probability of A given B. This would be the probability of it raining, given that Robin cycles to work. Observe that this differs from Prob $(B|A)$, the probability of B given A, which would be the probability of cycling to work given that it is raining.

We can see that Prob $(A|B) = \frac{\text{Prob}(A \cap B)}{\text{Prob}(B)}$, and also that Prob $(B|A) = \frac{\text{Prob}(B \cap A)}{\text{Prob}(A)}$.

These relations are called the *law of conditional probability*. Noting that $A \cap B \equiv B \cap A$, and rearranging, we arrive at *Bayes's theorem*:

$$\frac{\text{Prob}\,(A|B)}{\text{Prob}\,(A)} = \frac{\text{Prob}\,(B|A)}{\text{Prob}\,(B)} \tag{7.1}$$

In practice, we re-write $\text{Prob}\,(B)$ and rearrange. It should be clear that $\text{Prob}\,(B) = \text{Prob}\,(B \cap A) + \text{Prob}\left(B \cap \overline{A}\right)$. Using the law of conditional probability once more, we obtain $\text{Prob}\,(B) = \text{Prob}\,(A)\,\text{Prob}\,(B|A) + \text{Prob}\left(\bar{(}A)\right)\text{Prob}\left(B|\overline{A}\right)$. Rearranging:

$$\text{Prob}\,(A|B) = \text{Prob}\,(A)\,\frac{\text{Prob}\,(B|A)}{\text{Prob}\,(A)\,\text{Prob}\,(B|A) + \text{Prob}\left(\overline{A}\right)\text{Prob}\left(B|\overline{A}\right)} \tag{7.2}$$

With our example, the left hand side of the above equation is "the probability that it is raining, given that Robin cycles to work". The value of the equation is that all the terms on the right hand side are available. Note carefully how the causation operates: the fine weather *causes* Robin to cycle, and rain *causes* Robin to get the bus to work, and not cycle. But we can use the observation that Robin cycles to *infer* that it was a fine day, on the grounds that if it was raining he would have caught the bus.

It is easier to appreciate the import of Bayes's theorem if it is re-stated with different letters:

$$\text{Prob}\,(H|D) = \text{Prob}\,(H)\,\frac{\text{Prob}\,(D|H)}{\text{Prob}\,(H)\,\text{Prob}\,(D|H) + \text{Prob}\left(\overline{H}\right)\text{Prob}\left(D|\overline{H}\right)} \tag{7.3}$$

Here, H stands for "hypothesis" which explains data D. Here the hypothesis H would be "it is raining" and of course $\overline{H}$ would mean "it is not raining". Note that the two hypotheses H and $\overline{H}$ are exclusive and exhaustive: exactly one of these two alternatives is true. The data D would be "Robin cycled to work today". In Bayes's theorem we would have the following interpretations for the various terms:

- $\text{Prob}\,(H)$ This is the *prior* probability that it is raining; that is, the overall probability of raining before observing the data
- $\text{Prob}\,(H|D)$ This is the *posterior* probability that it is raining; that is, the probability of rain, given the data which in this case is the observation that Robin cycled to work that day.
- $\text{Prob}\,(D|H)$. This is the probability of observing the data, given that hypothesis H is correct; in this case it is the probability that Robin cycles, given that it is raining

- $\text{Prob}\left(D|\overline{H}\right)$. This is the probability of observing the data, given that hypothesis H is incorrect (i.e. that is is fine); in this case it is the probability that Robin cycles, given that it is fine weather.

We can put numbers to this:

- $\text{Prob}\left(H\right) = 0.2$ a prior probability for rain, from previous experience. Observe that this imples $\text{Prob}\left(\overline{H}\right) = 0.8$.
- $\text{Prob}\left(D|H\right) = 0.1$ the probability of cycling given rain. I generally can't be bothered to cycle if it is wet and get the bus instead.
- $\text{Prob}\left(D|\overline{H}\right) = 0.5$ the probability of cycling given fine weather. If it is fine weather, I cycle about half of the time, and half the time I am too tired and get the bus anyway.

We can plug these figures directly in to Bayes's theorem:

```
(0.2*0.1)/(0.2*0.1 + 0.8*0.5)
```

```
## [1] 0.04761905
```

Thus the posterior probability of rain is about 4.7%. Note carefully that the prior probability of rain was 20% but this has decreased to 4.7% due to the observation that Robin cycled in to work. Intuitively, if you see me cycling to work you know it is unlikely that it is raining.

7.1 Independence

If events A, B satisfy the relationhip $\text{Prob}\left(A\right) = \text{Prob}\left(A|B\right)$, then this tells us that observation of B does not change the probability of A. We can recast this equation as

$$\text{Prob}\left(A\right)\text{Prob}\left(B\right) = \text{Prob}\left(A \cap B\right)$$

that is, to find the probability of both A and B occurring (the right hand side in the above equation), we simply multiply the probabilities of A and B occurring separately. If this is the case, we say that A and B are *independent*.

7.2 Bayes's theorem as a formal mechanism for updating beliefs

Observe the role of *knowledge* in the preceding discussion. The prior probability of rain was my estimate of the probability of rain, drawing on background meteorology of Auckland. We update this probability in the light of data, using Bayes's theorem, to give a posterior probability that incorporates the observation that Robin cycled to work.

In the literature you will often see the prior probability called a *subjective prior* as it reflects features such as hunches, opinion and other informal reasoning. Observe carefully that the posterior contains traces of the subjective prior but is modified by the data.

7.3 Bayes and more than two hypotheses

In the cycling example we had two hypotheses but it is straightforward to generalize to three. Suppose we have H_1, H_2 and H_3 being three hypotheses one of which is true. Then

$$\text{Prob}\,(H_i|D) = \text{Prob}\,(H_i)\,\frac{\text{Prob}\,(D|H_i)}{\text{Prob}\,(H_1)\,\text{Prob}\,(D|H_1) + \text{Prob}\,(H_2)\,\text{Prob}\,(D|H_2) + \text{Prob}\,(H_3)\,\text{Prob}\,(D|H_3)}$$

where i is equal to 1,2 or 3 depending on which of H_1, H_2, H_3 we are interested in. This may be extended further. If we have hypotheses $H_1, H_2, \ldots, H_n$ we would have

$$\text{Prob}\,\big(H_j|D\big) = \text{Prob}\,\big(H_j\big)\,\frac{\text{Prob}\,\big(D|H_j\big)}{\sum_{i=1}^{n}\text{Prob}\,(H_i)\,\text{Prob}\,(D|H_i)} \tag{7.4}$$

and (although the formal mathematics gets a little gnarly) we can have an infinite number of hypotheses indexed by a parameter $x \in X$:

$$\text{Prob}\,\big(H_y|D\big) = \text{Prob}\,\big(H_y\big)\,\frac{\text{Prob}\,\big(D|H_y\big)}{\int_{x\in X} P(x)\,\text{Prob}\,(D|H_x)\,dx}$$

where H_y is some particular hypothesis we are interested in, and $P(x)$ is the prior probability density function of H_x. We will see examples of this later in this chapter but for now we can use the observation that the denominator is a constant which does not depend on y and restate Bayes as:

$$\text{Prob}\,\big(H_y|D\big) \propto \text{Prob}\,\big(H_y\big)\,\text{Prob}\,\big(D|H_y\big) \tag{7.5}$$

7.3.1 Independence and Bayes

If D and H are independent, we would have $\text{Prob}\,(H) = \text{Prob}\,(H|D)$. Informally, this means that the prior probability of H is equal to its posterior probability. Thus observation of D does not change the probability of H occurring. We say that D is *uninformative* about H.

7.4 Bayes and the beta distribution

https://en.wikipedia.org/wiki/Beta_distribution

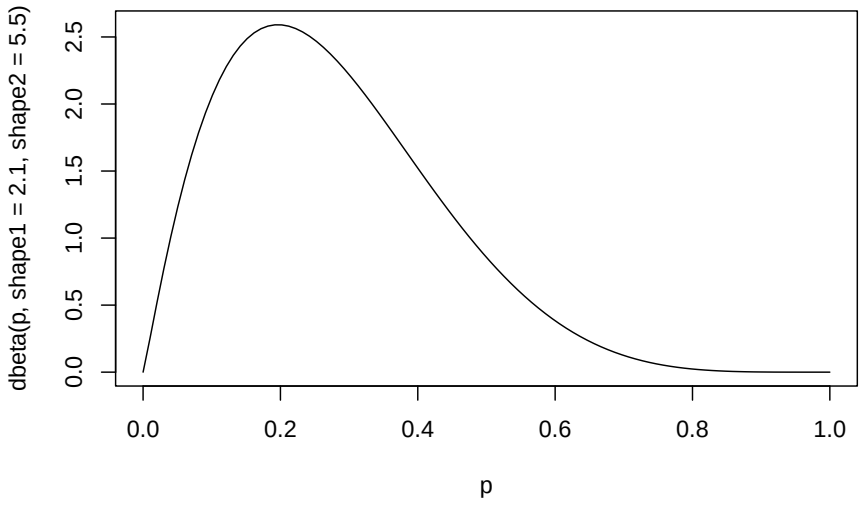
The *beta* distribution is often used when we are discussing a sequence of Bernoulli trials and wish to make inferences about p, the probability of success. One difficulty in dealing with probabilities is that probability must be non-negative, and cannot exceed 1; we have $0 \leq p \leq 1$, and this fact means that we cannot use distributions such as the Gaussian which is infinitely wide.

The only reasonable distribution for probabilities is the beta distribution; see `?rbeta` for details on R's functionality. The probability density function for the beta is

$$P(p) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1}(1-p)^{\beta-1} \tag{7.6}$$

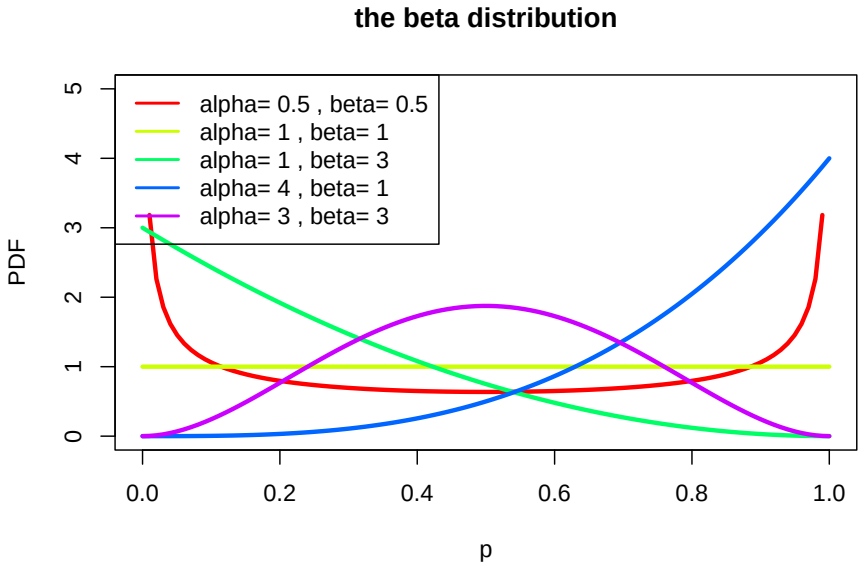
In the above equation, α and β are parameters [cf the Gaussian, which has parameters μ, σ] and p is a probability, and we understand $0 \leq p \leq 1$. We also introduce the gamma function $\Gamma(\cdot)$ which is defined as $\Gamma(x) = (x-1)!$; get help on this by typing `?gamma` at the R prompt. We can choose different values for α and β to change the shape of the distribution in much the same way as we choose the mean and variance in the Gaussian.

```
p <- seq(from=0,to=1,len=100)
plot(p,dbeta(p,shape1=2.1, shape2=5.5),type='l')
```



(note that R uses shape1 and shape2 in place of the more common notation α, β). In general, high values of α and β coresond to sharply-peaked distributions, and low values correspond to wider distributions. We require that $\alpha, \beta > 0$ but are otherwise arbitrary. The figure below shows a variety of beta distributions; study the R idiom carefully.

```
M <- matrix(c(0.5,0.5, 1,1, 1,3, 4,1, 3,3),ncol=2,byrow=TRUE) #each row=params
p <- seq(from=0,to=1,len=100)  # probability for x-axis
plot(p,p*5,type='n',ylab='PDF',main='the beta distribution')  # setup plot
for(i in seq_len(nrow(M))){  # iterate through rows of M
points(p,dbeta(p, M[i,1], M[i,2]),type='l',lwd=3,col=rainbow(nrow(M))[i]) #draw
}
legend(
"topleft",
legend = apply(M,1,function(x){paste("alpha=",x[1],", beta=",x[2])}),
lwd=2, col=rainbow(nrow(M))
)
```



See how we can represent a wide variety of curves by varying the parameters; not in particular that the red curve is bimodal.

7.5 Beta distributions as priors

Suppose we wish to make inferences about the probability of success for a Bernoulli trial. We are not sure about the value of p but have some beliefs about what it might be. We then carry out some experiments and observe that the experiment succeeds a times and fails b times out of $n = a + b$ trials; this is our data D. The Bayesian approach is to use equation (7.5) to modify our beliefs using the data D. We may represent our prior beliefs with a beta distribution with parameters α, β. This distribution represents our uncertainty about the value of p. To use the observational data D [a successes and b failures out of $a + b$ trials] we use equation (7.5) with a binomial probability distribution:

$$\text{Prob}\,(p) \propto \binom{a+b}{a\ b} p^a (1-p)^b \cdot \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1}(1-p)^{\beta-1} \tag{7.7}$$

Clearing the equation of constants we get that the probability density of p is

$$K p^{\alpha+a-1}(1-p)^{\beta+b-1} \tag{7.8}$$

(here K is a constant of proportionality set so the area under the curve is one). This is recognisable as another beta distribution with different parameters.

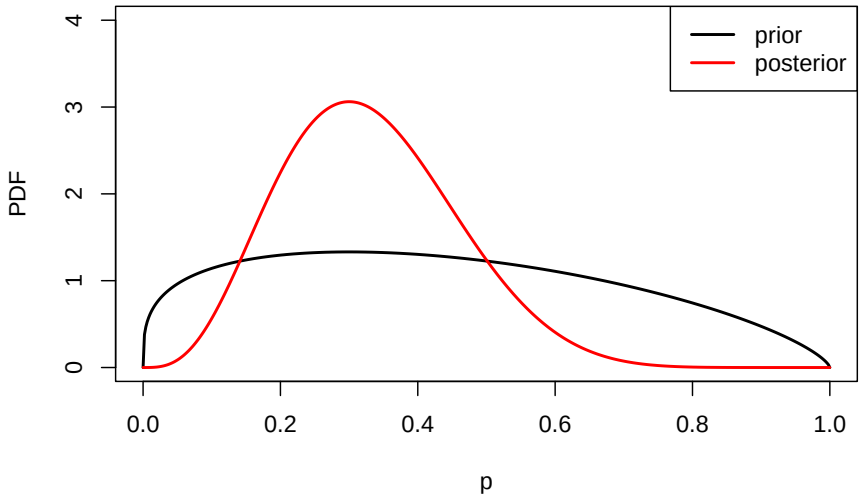
By comparing this with the definition of the beta distribution we find that $K = \frac{\Gamma(a+\alpha)\Gamma(b+\beta)}{\Gamma(a+b+\alpha+\beta)}$ is the value which ensures that the area under the curve is 1.

Note carefully that in this case both the prior and the posterior are beta distributions, but with different parameters ($\alpha \longrightarrow \alpha + a$, $\beta \longrightarrow \beta + b$). This is a very desirable feature of the beta distribution, as the algebra is straightforward. In general, the phenomenon of a prior distribution having the same form as the posterior is known as "conjugate prior".

7.5.1 Example.

Suppose we have a prior of $\alpha = 1.3, \beta = 1.7$ as a prior for p. This represents our opinion about the likely values of p. We conduct 10 Bernoulli trials and observe 3 successes and 7 failures. Then the posterior distribution for p will be beta with parameters $\alpha + a = 1.3 + 3 = 4.3$ and $\beta + 1.7 + 7 = 8.7$:

```
p <- seq(from=0,to=1,len=500)   # probability on horizontal axis
plot(p,p*0,ylim=c(0,4),ylab="PDF",type="n") # set up plot axes
points(p,dbeta(p,1.3,1.7),lwd=2,type="l",col='black') # prior
points(p,dbeta(p,1.3+3, 1.7+7),lwd=2,type="l",col='red') # posterior
legend("topright",lwd=2,col=c("black","red"),legend=c("prior","posterior"))
```



In the above diagram, see how the prior (black) distribution is wide, reflecting the large uncertainty. The posterior (red) distribution is narrow, showing that our data is informative about the value of p.

Observe that the Bayesian approach does not provide "the" value of p. What we have done is to determine a posterior distribution that reflects our knowledge about it. We can find the value of p that maximizes the posterior, or find the mean of the posterior distribution [see the wikipedia page for how to do this]. We can also calculate more abstract results; for example, we can find the posterior probability that $p < \frac{1}{2}$:

```
pbeta(1/2, 4.3, 8.7)
```

```
## [1] 0.898819
```

7.6 Further examples of Bayes's theorem.

Bayes's theorem in the form of equation (7.4) is very general. Here are some other examples of Bayesian reasoning.

7.6.1 Poisson distribution

Suppose we have a sample of a radioactive source. It is either substance A or substance B but we do not know which. The hypothesis that it is in fact substance A is written H_A and the hypothesis that it is in fact substance B is written H_B. Our priors are $p(H_A) = 0.9$ and $p(H_B) = 0.1$. Our experiment is to observe the number of counts on a Geiger counter. Counts are known to be Poisson and substance A has $\lambda = 5.4$ while substance B has $\lambda = 7.7$. Our data is the observed number of counts which is $n = 5$. Bayes's theorem:

$$p(H_A|D) = \frac{p(H_A)p(D|H_A)}{p(H_A)p(D|H_A) + p(H_B)p(D|H_B)} \tag{7.9}$$

We can use R to calculate the likelihoods:

```
(LA <- dpois(5, 5.4))
```

```
## [1] 0.1728213
```

```
(LB <- dpois(5, 7.7))
```

```
## [1] 0.1021421
```

Then Bayes gives us the posterior probability $p(H_A|D)$ as

```
prior_A <- 0.9
prior_B <- 0.1
posterior_A <- prior_A*LA/(prior_A*LA + prior_B*LB)
posterior_A
```

```
## [1] 0.9383771
```

So in this case our prior probability $p(H_A)$ was 0.9 and our posterior $p(H_A|D)$ has increased to about 0.94, on account of the low count ($n = 5$) being more consistent with H_A ($\lambda = 5.4$) than H_B ($\lambda = 7.7$)

7.6.2 Dice

Men of a certain age will be familiar with different dice used in board games. Suppose someone has three dice, d4,d6,d8. In standard terminology, "d4" is a four-sided die in the shape of a tetrahedron with results $\{1, 2, 3, 4\}$, "d6" is a cube that can give $\{1, 2, 3, 4, 5, 6\}$, and "d8" is an octahedron with $\{1, 2 \ldots, 8\}$. A person takes one of these dice and throws it, obtaining a 5. We do not know which of the dice they threw but we have three hypotheses to consider: H_4, H_6, and H_8. The observation of "5" is informative about these via Bayes's theorem. Suppose our priors are $p(H_4) = 0.1, p(H_6) = 0.8, p(H_8) = 0.1$. Then the likelihoods are $p(D|H_4) = 0$ (think about it), $p(D|H_6) = \frac{1}{6}$ and $p(D|H_8) = \frac{1}{8}$. Bayes gives us

$$p(H_4|D) = \frac{p(H_4)p(D|H_4)}{p(H_4)p(D|H_4) + p(H_6)p(D|H_6) + p(H_8)p(D|H_8)} = \frac{0.1 \cdot 0}{0.1 \cdot 0 + 0.8 \cdot \frac{1}{6} + 0.1 \cdot \frac{1}{8}} = 0$$

$$p(H_6|D) = \frac{p(H_6)p(D|H_6)}{p(H_4)p(D|H_4) + p(H_6)p(D|H_6) + p(H_8)p(D|H_8)} = \frac{0.8 \cdot \frac{1}{6}}{0.1 \cdot 0 + 0.8 \cdot \frac{1}{6} + 0.1 \cdot \frac{1}{8}} = 91\%$$

$$p(H_8|D) = \frac{p(H_4)p(D|H_4)}{p(H_4)p(D|H_4) + p(H_6)p(D|H_6) + p(H_8)p(D|H_8)} = \frac{0.1 \cdot \frac{1}{8}}{0.1 \cdot 0 + 0.8 \cdot \frac{1}{6} + 0.1 \cdot \frac{1}{8}} = 9\%$$

and we that Bayes has updated the priors for our three hypotheses and given three posterior probabilities.

Chapter 8

Fisher's exact test

https://en.wikipedia.org/wiki/Fisher%27s_exact_test

https://www.youtube.com/watch?v=a7ESQKI7nao&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84&index=16

https://www.youtube.com/watch?v=O9LUvIFSlRo&index=17&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

https://www.youtube.com/watch?v=IL-2Su9YQ60&index=18&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

https://www.youtube.com/watch?v=HM0oNfNZQlY&index=19&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

Suppose I am interested in the prevalence of left handedness and wish to determine whether there is a difference between boys and girls. I might survey some students and end up with a dataset like the following:

```
a <- matrix(c(5,2,6,14),2,2,byrow=TRUE)   # define matrix
dimnames(a) <- list(gender=c("M","F"),lefthanded=c(T,F))  # display only
a
```

```
##        lefthanded
```

```
## gender TRUE FALSE
##      M    5     2
##      F    6    14
```

Such a dataset is known as a *contingency table* and is surprisingly common in statistics. I would say that over half of the statistical analyses I perform for clients is on contingency tables like the one above.

Is there any evidence that handedness differs by gender? Just looking at the table, we can see that the majority of males are left handed and the majority of females are right handed. We need to provide statistical analysis of this question which means providing a p-value.

The starting point would be to provide a null hypothesis, which is that handedness is independent of gender. We would thus have H_0: $\text{Prob}\,(L|M) = \text{Prob}\,(L)$. This is mathematically identical to the assertion that $\text{Prob}\,(L|M) = \text{Prob}\,(L|F)$. It certainly looks as though there is a difference, but to quantify this we need a more careful analysis.

8.1 The hypergeometric distribution

https://en.wikipedia.org/wiki/Hypergeometric_distribution

(note that the wikipedia page has different parameterizations from the R help page. Here, I will follow the notation used in R, but be aware that other conventions exist). Suppose we have an urn with m white and n black balls, and we draw out k balls and write down their colour. Note carefully that we are drawing *without replacement*: we draw a ball from the urn and then do not put it back in the urn (if we were to replace the ball in the urn, the number of white balls would be binomial with size k and probability $\frac{m}{m+n}$).

What is the distribution of the number of white balls drawn without replacement? Well, it can be shown that

$$\text{Prob}\,(X = x) = \frac{\binom{m}{x}\binom{n}{k-x}}{\binom{m+n}{k}} \tag{8.1}$$

where X is the random variable corresponding to the number of white balls drawn and x is an integer. The R idiom for this is `dhyper()` which calculates the density function. We have the standard quartet of functions: `rhyper()`, `dhyper()`, `qhyper()` and `phyper()` which are documented together. The `dhyper()` function takes a number of arguments.

Consult the R help page for details, but the basic command is `dhyper(x,m,n,k)` where x is the number of white balls drawn, m number of white balls in the urn, n number of black balls in the urn, and k is the number of balls drawn.

We can make the following mapping between drawing balls out of a urn and left/right male/female dataset as follows:

- white ball ⟶ left handed
- black ball ⟶ right handed
- drawn from urn ⟶ male
- left in urn ⟶ female

Alternatively we might say

- white ball ⟶ male
- black ball ⟶ female
- drawn from urn ⟶ left handed
- left in urn ⟶ right handed

Taking the second of the two identifications, we have

- number of white balls = number of males =5+2=7
- number of black balls = number of females = 6+14=20
- drawn from urn = left handed = 5+6=11
- left in urn = right handed = 2+14=16

With this identification, the *observation* is the number of white balls drawn from the urn (= the number of left handed males) which is 5.

The probability of this occurring if the null is true is

```
dhyper(5,7,20,11)
```

```
## [1] 0.06243032
```

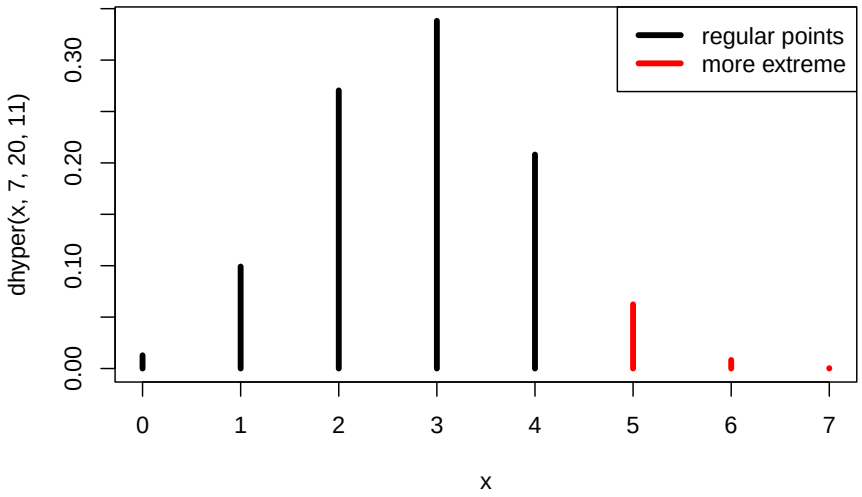
Recall the definition of p-value: "the probability, if the null is true, of obtaining the observation or an observation more extreme". In this case (if we use a one-sided test), "more extreme" means greater than the observation of 5; and we know that the number of left handed males (=white balls drawn) cannot be larger than 7. So the p-value is

```
sum(dhyper(5:7,7,20,11))
```

```
## [1] 0.07112598
```

This shows that the p-value of about 7% is not significant (the professional uses `phyper(4,7,20,11,lower.tail=FALSE)`). We can get a visual impression easily, using R; study the idiom carefully:

```
x <- 0:7  # define x-axis
plot(x,dhyper(x,7,20,11),type='h',lwd=4,col=c(rep("black",5),rep("red",3)))
legend("topright",lwd=4, col=c("black","red"),legend=c("regular points","more extreme"))
```



Fortunately, R can deal with all this with the `fisher.test()` function:

```
fisher.test(a, alternative="greater")
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  a
## p-value = 0.07113
## alternative hypothesis: true odds ratio is greater than 1
## 95 percent confidence interval:
##  0.8614627       Inf
## sample estimates:
## odds ratio
##   5.421131
```

(recall that R variable `a` was defined at the beginning of the chapter).

Chapter 9

Pearson's chi-square test

https://en.wikipedia.org/wiki/Chi-squared_test

https://www.youtube.com/watch?v=OkatmisQs8E&index=14&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

https://www.youtube.com/watch?v=al-2-RRzMek&index=15&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

Suppose I poll a class of 40 students and ask each one for their favourite colour out of red, green, blue, and yellow. My observations are:

```
o <- c(red=8, green=10, blue=17, yellow= 5)
o
```

```
##    red  green   blue yellow
##      8     10     17      5
```

("o" for "observation"). Thus 8 students chose red, 10 chose green, etc. Is there any evidence that there is a systematic bias towards some colours? Well, the first step is to formulate a null hypothesis $H_0 : p_{\text{red}} = p_{\text{green}} = p_{\text{blue}} = p_{\text{yellow}} = \frac{1}{4}$, where p_{colour} is the probability of a student choosing that colour. If the null is true, we would expect to observe 10 students choosing each colour:

```
e <- c(red=10, green=10, blue=10, yellow= 10)
e
```

```
##    red  green   blue yellow
##     10     10     10     10
```

("e" for "expectation"). We need to quantify the difference between observation and expectation[1] as a single number. What we want is to calculate a number which is *small* if the observations are close to expectations, and *large* if the observations are far away from expectations. I usually call this number B, for "badness of fit".

It turns out that one particular definition of B has nice properties that we can work with:

$$B = \sum_i \frac{(e_i - o_i)^2}{e_i} \tag{9.1}$$

where o_i is the i^{th} observation and e_i is the i^{th} expectation. This is easy to calculate:

```
(8-10)^2/10 + (10-10)^2/10 + (17-10)^2/10 + (5-10)^2/10
```

```
## [1] 7.8
```

but of course there is an easier way:

```
B <- sum((o-e)^2/e)
B
```

```
## [1] 7.8
```

See how the definition of B ensures that if the observations were exactly equal to the expectations, B would be zero. We can also see that B cannot be negative (because each term is a squared number divided by a positive number), and also that B is large if the expectations are very different from the observations.

Recall the definition of p-value: "the probability, if the null is true, of obtaining the observation or an observation more extreme". Here, our observation is the value of $B = 7.8$. Our job is to figure out what the distribution of B is, given that the null is true. If we assume that the expected number of observations is not too small (> 5 is the usual heuristic), it turns out that the null distribution of B has a pleasing mathematical form, called the *chi-squared distribution*.

[1] Remember: the whole of science reduces to a comparison between observation and expectation

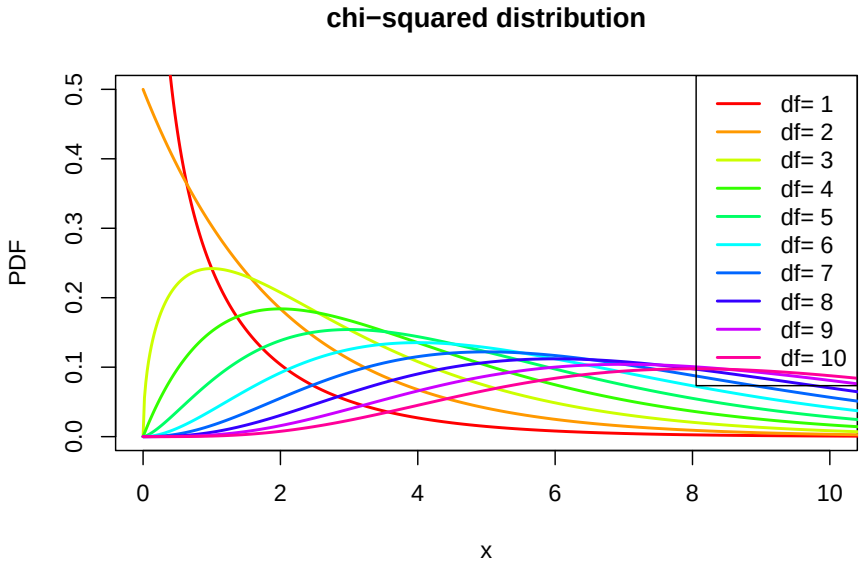
9.1 The chi squared distribution

https://en.wikipedia.org/wiki/Chi-squared_distribution

The chi squared distribution, sometimes denoted χ^2, is useful for many reasons but here we need it because it tells us about the distribution of B if the null is true.

The χ^2 distribution does not have parameters in the same way that the Gaussian distribution has μ and σ. But it does have "degrees of freedom" which is a strictly positive integer usually written as a subscript: χ^2_n. The degrees of freedom varies from problem to problem but is easy to calculate. The R idiom below shows some chi-square distributions.

```
x <- seq(from=0,to=20,len=1000)  # values for horizontal axis
plot(1:10,1:10,type='n',xlim=c(0,10),ylim=c(0,0.5),xlab='x',ylab='PDF',
  main='chi-squared distribution')  # setup plot axes and title
for(i in 1:10){  # loop over i
    points(x,dchisq(x,df=i),type='l',lwd=2,col=rainbow(10)[i]) #plot one curve
}
legend("topright",lwd=2,col=rainbow(10),legend=paste("df= " ,1:10,sep=""))
```

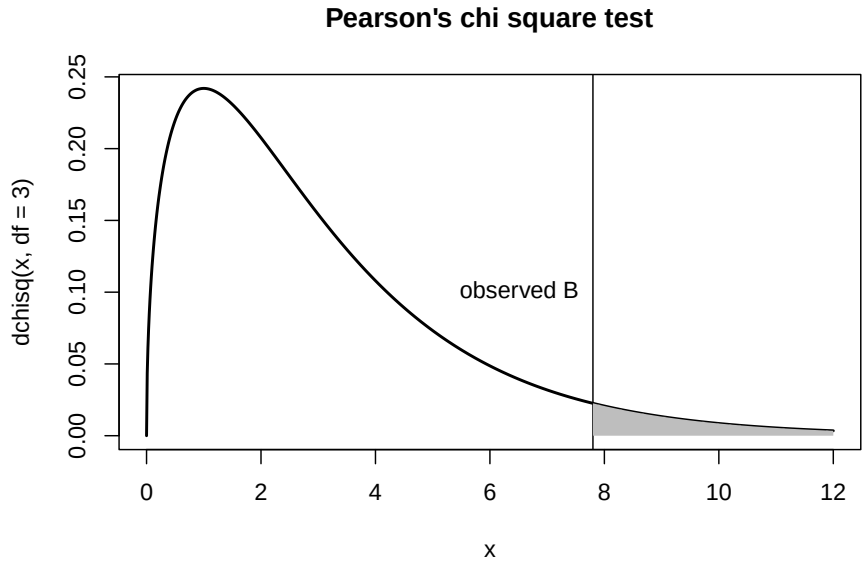


Study the above R idiom carefully, and note how the distributions move to the right with increasing degrees of freedom.

9.2 The chi squared distribution and Pearson's chi-squared test

It turns out that if the null is true then B, the badness of fit measure defined above, follows a chi-squared distribution. The number of degrees of freedom is given by the number of cells (in the students choosing colours example above, this would be 4) minus one, which would be $4 - 1 = 3$ degrees of freedom. The reason you subtract one from the number of cells is that knowing three cells is enough to calculate the fourth, because we know how many students are in the class.

```
x <- seq(from=0,to=12,len=1000)  # set up x-axis
plot(x,dchisq(x,df=3),type='l',lwd=2,main="Pearson's chi square test")  # setup axes
B <- 7.8  # value of Badness from above
abline(v=B) # draw vertical line
jj <- seq(from=B,to=12,len=100)  # temporary variable
polygon(c(jj,rev(jj)),c(jj*0,dchisq(rev(jj),df=3)),col='gray',border=NA)  #shade pvalu
text(7.8,0.1,"observed B",pos=2)
```



In the above figure, the p-value is shown in gray: it is the probability, if the null is true, of obtaining the observation or an observation more extreme. In this case, the observation is $B = 7.8$ and its null distribution is χ^2_3. Calculating the p-value is straightforward:

```
pchisq(7.8,df=3,lower.tail=FALSE)
```

```
## [1] 0.0503311
```

just short of the 5% critical value, so we fail to reject the null.

9.3 Numerical verification

Above, I stated that B has a chi-square distribution with 3 degrees of freedom. Here I demonstrate that this is true. We can generate synthetic observations using the `sample()` function:

```
sample(1:4,40,replace=T)  # census of the class
```

```
##  [1] 2 4 2 2 1 3 2 1 4 4 3 1 1 3 2 2 2 1 3 2 4 4 3 4 4 3 4 2 4 1 3 1 3 4 4 3 3 1
## [39] 1 4
```

```
tabulate(sample(1:4,40,replace=TRUE))  # how many students prefer each colour
```

```
## [1] 13  7  6 14
```

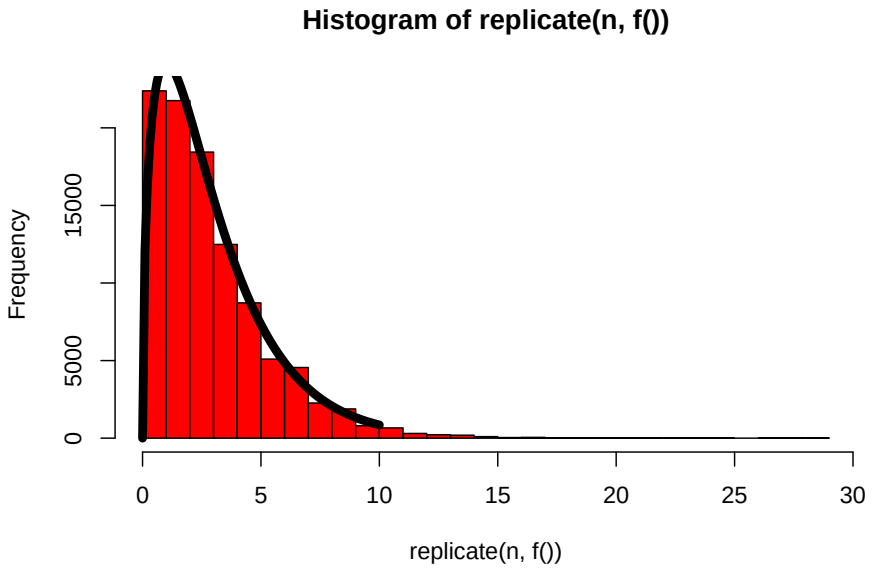
(we identify red=1, green=2, etc). We can calculate B for such synthetic data:

```
o <- tabulate(sample(1:4,40,replace=TRUE))  # synthetic observations
e <- 10                                     # expectation under the null
B <- sum((o-e)^2/e)                         # Badness-of-fit
B
```

```
## [1] 1.4
```

And I assert that B is drawn from χ^2_3. This is straightforward to verify in R:

```
f <- function(...){  # define single-use function f()
   o <- tabulate(sample(1:4,40,replace=TRUE))  # synthetic observations
   e <- 10  # expectations
   return(sum((o-e)^2/e))  # return badness B
}
n <- 1e5   # number of synthetic observations to take
hist(replicate(n,f()),col='red',nclass=30)  # plot the histogram of synth.obs
x <- seq(from=0,to=10,len=100)   # x-values for theoretical distribution
points(x,n*dchisq(x,df=3),type='l',lwd=6)  # plot chi-squared, df=3 points
```



9.4 Another example

Here I will give an example along the same lines as above but with a different null. Suppose we give 100 students a coin and tell them to toss it 4 times. We can then ask each student to write down the number of heads they get. First, our observations:

```
o <-c("0"=5, "1"=27, "2" = 47, "3"=15, "4"=6)  # this is a named vector
o
```

```
##  0  1  2  3  4
##  5 27 47 15  6
```

The expectations are a little harder. Our null is that the number of heads is $\text{Bin}(4, 1/2)$, and we have 100 observations, so our expectations are given by

```
e <- 100*dbinom(0:4,4,1/2)
e
```

```
## [1]  6.25 25.00 37.50 25.00  6.25
```

(note that the expected values are not necessarily integer-valued). Now the badness of fit:

```
B <- sum((o-e)^2/e)
B
```

```
## [1] 6.826667
```

And the p-value is

```
pchisq(B,df=4,lower.tail=FALSE)
```

```
## [1] 0.1453366
```

So the p-value exceeds 5% and the result is not significant.

9.5 Pearson chi-square test with estimated parameters

Sometimes we might have a null hypothesis that includes an unknown parameter. Suppose we are looking at Robin's photograph album with 23 photos in it. Robin has 3 children and we know he likes taking photos of his kids. We count how many photos have 0,1,2,3 children in them and observe the following dataset:

```
( o <- c("0"=7,"1"=3,"2"=4,"3"=9))
```

```
## 0 1 2 3
## 7 3 4 9
```

We might hypothesize that the number of children in each photo is binomial $\text{Bin}(3, p)$ but we do not know the value of p. To estimate p, we figure out how many photos of the three children there are and divide by how many there could possibly be:

```
phat <- (0*7 + 1*3 + 2*4 + 3*9)/(3*(7+3+4+9))
```

or

```
(phat <- sum((0:3)*o)/(3*sum(o)))
```

```
## [1] 0.5507246
```

Armed with this, we can calculate expectations:

```
(e <- 23*dbinom(0:3,3,phat))
```

```
## [1] 2.085766 7.670237 9.402226 3.841770
```

and the p-value is easy to calculate

```
pval <- pchisq(sum((o-e)^2/e),df=2,lower.tail=FALSE)
pval
```

```
## [1] 4.902114e-06
```

showing that we may reject the null: we have strong evidence that the number of children in a photo is not binomially distributed.

9.5.1 Note on number of degrees of freedom

In the photo example above, note that the number of degrees of freedom is 2, not 3 as might be expected. This is because we have to subtract an additional degree of freedom to compensate for the fact that `phat` was calculated from the data. Because `phat` is a single parameter, this corresponds to a single degree of freedom which has to be subtracted from the `pchisq()` calculation. Some more sophisticated tests involve distributions with multiple parameters; one has to subtract one degree of freedom per parameter estimated from the data.

9.6 Chi-square test and the Poisson distribution

The chi-square test works well with the Poisson, but we have to be careful with small tail probabilities. Here we will consider the number of goals scored by Manchester City Football club in the 2016-2017 season, data courtesy of wikipedia.

https://en.wikipedia.org/wiki/2016%E2%80%9317_Manchester_City_F.C._season

The dataset is

```
manC <-
    c(2,4,3,1,4,3,0,1,1,4,1,2,2,
      1,2,2,2,3,0,2,0,2,4,2,2,0,
      1,2,1,3,3,5,3,0,5,2,2,0)
```

Thus in the first game, they scored 2 goals, in the second 4, and so on up to game number 38 which scored zero goals. We wish to test whether the number of goals is Poisson. This would be a reasonable supposition on the grounds that there are a large number of minutes in a football match, each one of which would have a small probability of scoring a goal. The first step would be to tabulate the data:

```
o <- table(manC)    #'o' for 'observations'
o
```

```
## manC
##  0  1  2  3  4  5
##  6  7 13  6  4  2
```

Thus, we have six matches with zero goals, 7 matches with 1 goal, 13 matches with 2 goals, and so on. Now we need to calculate an expectation to compare these observations against. The mean number of goals is:

```
mean(manC)
```

```
## [1] 2.026316
```

that is, a little over two goals per match. We can model the number of goals scored in each match as a Poisson distribution with $\lambda = 2.026$. We will classify games as having 0,1,2,3,4, or ≥ 5 goals (this device prevents expected numbers being too small for the Chi square test to operate). The probabilities of 0-4 goals is

```
dpois(0:4, lambda=mean(manC))
```

```
## [1] 0.13182028 0.26710952 0.27062412 0.18278997 0.09259755
```

Then the probability of scoring 5 goals will be the remaining probability, viz

```
1-sum(dpois(0:4, lambda=mean(manC)))
```

```
## [1] 0.05505856
```

So the probabilities of $0, 1, 2, 3, 4, \geq 5$ goals is

```
probs <- c(dpois(0:4, lambda=mean(manC)), ppois(4,lambda=mean(manC),lower.tail=FALSE))
probs
```

```
## [1] 0.13182028 0.26710952 0.27062412 0.18278997 0.09259755 0.05505856
```

check:

```
sum(probs)
```

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

Then the expected number of games with $0, 1, 2, 3, 4, \geq 5$ goals is:

```
e <- length(manC)*probs
e
```

```
## [1]  5.009171 10.150162 10.283716  6.946019  3.518707  2.092225
```

Thus we expect to see about 5 games with zero goals, 10 with one goal, etc. The chi-square test is straightforward:

```
B <- sum((o-e)^2/e)
pchisq(B,df=4,lower.tail=FALSE)
```

```
## [1] 0.7192346
```

and we can see that we fail to reject the null: the observations are consistent with a Poisson distribution. For the degrees of freeedom, note that we subtract (from 6, the number of entries in the table) one because we know the total number of games played, and another one because we needed to use the data to estimate λ which was used to generate the expectations. Thus we use 6-1-1=4.

Chapter 10

Linear Regression

https://en.wikipedia.org/wiki/Linear_regression

YouTube https://www.youtube.com/watch?v=QVFz4idnd6o&index=30&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

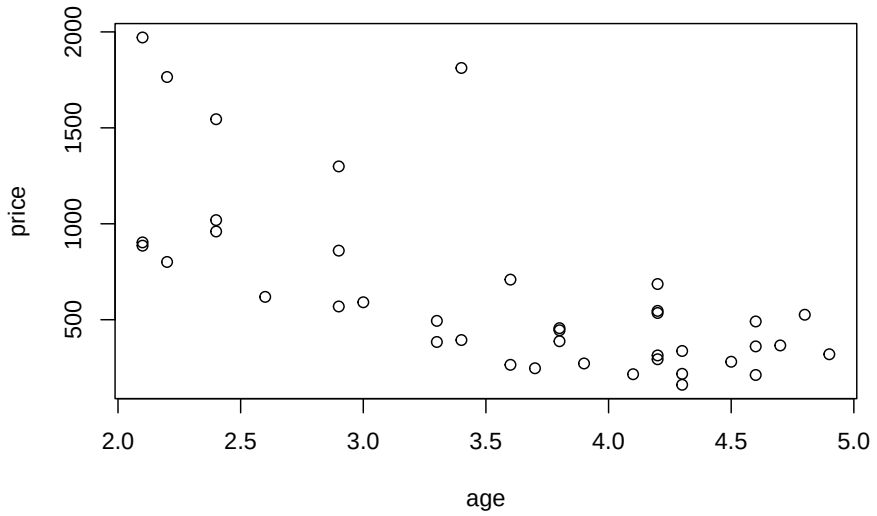
YouTube https://www.youtube.com/watch?v=ynNEEB3UfO8&index=31&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

In this chapter we investigate linear dependencies between two datasets. The technique used will be *linear regression* which is a very widely used technique in many fields of quantitative analysis. We will consider a number of examples drawn from different disciplines. First we will consider second hand cars, with the price of a car being a function of age. The dataset is:

```
carprices <- data.frame(
age =
      c(4.2, 2.1, 4.8, 2.1, 2.9, 4.3, 4.3, 3.3, 3.8, 3.4, 4.2,
        3.3, 4.1, 2.1, 4.2, 2.9, 2.4, 3.9, 4.9, 3.6, 4.5, 4.6,
    2.4, 4.2, 4.3, 4.7, 2.4, 4.6, 3.7, 3.8, 3.0, 2.6, 4.2,
    3.4, 3.6, 2.2, 4.6, 2.2, 2.9, 3.8),
price =
  c(686, 886, 526, 1971, 860, 161, 218, 494, 445, 1812, 535, 384, 216, 903, 314, 569,
1019, 272, 320, 265, 281, 361, 960, 546, 337, 366, 1545, 212,
247, 456, 591, 619, 294, 394, 709, 1765, 491, 801, 1299, 388))
```

THE FIRST AND NON-NEGOTIABLE STEP TO INVESTIGATING DATA OF THIS NATURE IS TO DRAW A SCATTERGRAPH. THE TECHNIQUES PRESENTED HERE ARE TOTALLY WORTHLESS IF YOU DO NOT DRAW A SCATTERGRAPH. IF YOU CONDUCT A LINEAR ANALYSIS WITHOUT A SCATTERGRAPH YOU WILL GET ZERO COURSE CREDIT.

```
plot(price~age, data=carprices)
```



In the above R idiom, the "~" symbol is read "is a function of". See how the two variables `age` and `price` appear to be related, with older cars having a smaller price. The scattergraph shows directly that a linear fit might be a good idea: the relationship is linear, there are no serious outliers, and the variability is roughly constant along the x-axis. Having established that a linear model is a good starting point, we will use R to quantify the relationship between the two variables.

Observe that the vertical axis (price) depends on the horizontal axis (age). Older cars are worth less. In this example, it is obvious which variable appears as the horizontal and which on the vertical axis. However, sometimes it is less clear.

10.1 Linear regression in R

In general, we have variables $x_1, x_2, \ldots, x_n$ which are drawn on the horizontal axis. These are the *independent* variables: they may be specified as we wish.

We also have variables $y_1, y_2, \ldots, y_n$ which are drawn on the vertical axis. These variables respond in some way to x's and we call these the *dependent* variables, because they depend on the x's.

The relationship we usually use is as follows:

$$y_i = \alpha + \beta x_i + \epsilon_i, \qquad i = 1, 2, \ldots n$$

Here,

- x_i are the independent variables
- y_i are the dependent variables
- α, β are the intercept and slope of the straight line which expresses the relationship between x_i and y_i
- $\epsilon_i \sim N(0, \lambda^2)$ is an error term. Observe that the ϵ_i are independent and identically distributed.

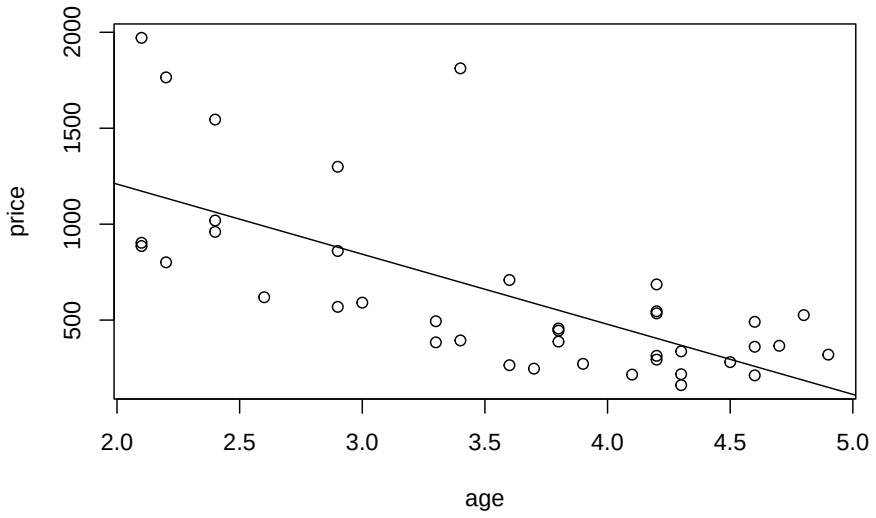
We do not know the values of α, β, so have to estimate them. The standard approach is to find the maximum likelihood estimate, $\hat{\alpha}, \hat{\beta}$. The R idiom to do this is to use the `lm()` function:

```
lm(price~age,data=carprices)
```

```
##
## Call:
## lm(formula = price ~ age, data = carprices)
##
## Coefficients:
## (Intercept)          age
##      1939.3       -365.3
```

This shows that the line of best fit is $y = 1939.3 - 365.3 * x$. It is straightforward to plot this on the graph:

```
plot(price~age,data=carprices)
abline(lm(price~age,data=carprices))
```



However, at this point we are not sure how accurately the fit parameters are known, or in particular whether the slope is statistically significant. To answer such questions we specify $H_0 : \beta = 0$ and seek a p-value. R provides functionality to do this:

```
summary(lm(price~age,data=carprices))
```

```
##
## Call:
## lm(formula = price ~ age, data = carprices)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -370.54 -245.11  -93.14  141.58 1114.69
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1939.28     227.51   8.524 2.37e-10 ***
## age          -365.29      62.06  -5.886 8.16e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 339.4 on 38 degrees of freedom
## Multiple R-squared:  0.4769, Adjusted R-squared:  0.4631
```

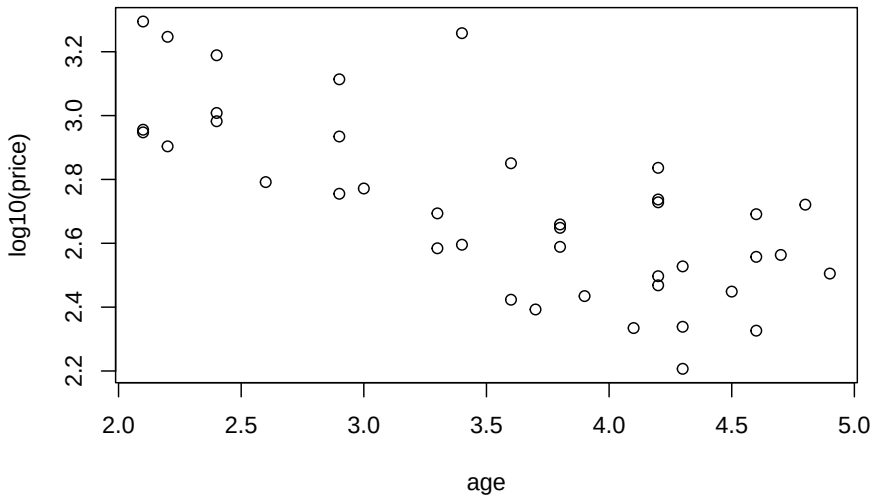
```
## F-statistic: 34.65 on 1 and 38 DF,  p-value: 8.16e-07
```

The p-value reported is two-sided; to get the one-sided equivalent, simply divide by 2.

10.1.1 Critical evaluation and logarithmic transform

The above analysis is not perfect. For example, the structure of the model has two defects: firstly, the price of the car becomes negative after a certain amount of time, which is unrealistic. We can mitigate this by considering a logarithmic transform:

```
plot(log10(price)~age,data=carprices)
```



Here we use logs to the base 10. This looks suitable for linear regression:

```
summary(lm(log10(price)~age,data=carprices))
```

```
##
## Call:
## lm(formula = log10(price) ~ age, data = carprices)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.33395 -0.13129 -0.02928  0.15103  0.50749
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.54362    0.12835  27.610  < 2e-16 ***
## age         -0.23322    0.03501  -6.661 7.09e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1915 on 38 degrees of freedom
## Multiple R-squared:  0.5387, Adjusted R-squared:  0.5266
## F-statistic: 44.38 on 1 and 38 DF,  p-value: 7.086e-08
```

See how the p-value is about the same in this case. One way to objectively compare the two approaches is to look at the correlation coefficient, often called R^2. According to the linear model above, these two models are about the same.

The transformed model would be

$$\log(\text{price}) = 3.54 - 0.233 * \text{age}$$

which would translate to about $3500 * 1.71^{-\text{age}}$. Note that this model cannot predict negative prices as exponenentials are always strictly positive.

10.2 Multiple and restricted regression

The tilde device for formulae (that is, expressions such as "`y~x`") is a very powerful and flexible notation.

Consider the `swiss` dataset, built in to R. Type `?swiss` at the prompt to get more details. Here, we are interested in infant mortality as a function of fertility and prevalence of Catholicism. Symbolically we would write

$$I_i = \beta_0 + \beta_1 F_i + \beta_2 C_i + \epsilon_i$$

where β_0 is the intercept and β_1, β_2 are slopes. The R idiom is:

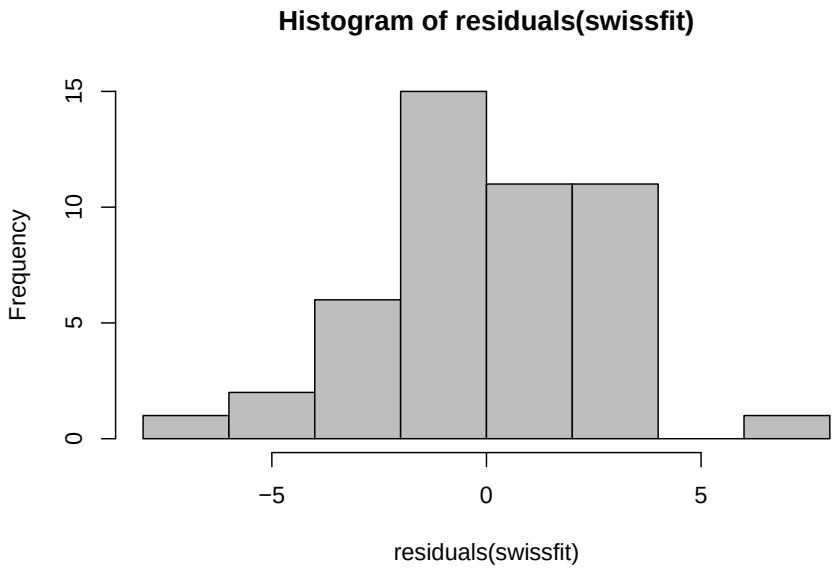
```
summary(lm(Infant.Mortality ~ Fertility + Catholic, data=swiss))
```

```
##
## Call:
## lm(formula = Infant.Mortality ~ Fertility + Catholic, data = swiss)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6265 -1.5382 -0.0718  1.8753  6.1420
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 13.023771   2.389259   5.451 2.15e-06 ***
## Fertility    0.099560   0.036060   2.761  0.00837 **
## Catholic    -0.001571   0.010801  -0.145  0.88504
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.707 on 44 degrees of freedom
## Multiple R-squared:  0.1739, Adjusted R-squared:  0.1364
## F-statistic: 4.632 on 2 and 44 DF,  p-value: 0.01495
```

In the above, note how the "+" symbol is used to regress infant mortality against fertility *and* percentage of Catholics. In this case, see that catholicism is not a significant factor as the p-value exceeds 5%. Note that it is difficult to draw a scattergraph with more than one dependent variable (it is possible but difficult with two, and pretty much impossible to do with more than two). However, we can plot the *residuals* which are defined as the difference between observation and prediction; ϵ_i the formula above.

```
swissfit <- lm(Infant.Mortality ~ Fertility + Catholic, data=swiss)
hist(residuals(swissfit),col='gray')
```



The above histogram shows that the residuals are roughly Gaussian, in line with

the assumptions of linear modelling. Note in passing that `fit` is a perfectly legit R object and we can examine and manipulate it easily. For example:

```
coefficients(swissfit)
```

```
##  (Intercept)    Fertility     Catholic
## 13.023771428  0.099560216 -0.001570734
```

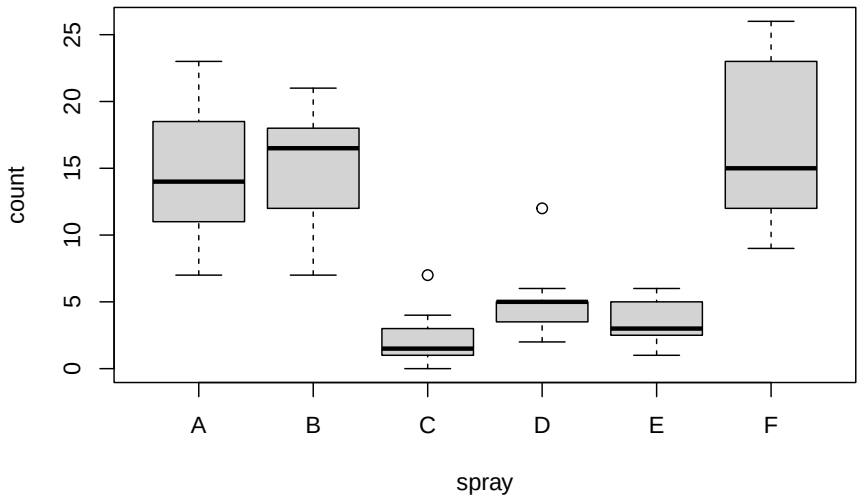
shows the three coefficients in the model fit.

Multiple regression is a difficult and technical branch of statistical theory and it can be a difficult and fraught matter to choose the appropriate linear model from similar alternatives.

10.3 Categorical regression

Consider the `InsectSpray` dataset, in which different pesticides were applied to agricultural units and the number of insects counted. First step is to make a plot:

```
boxplot(count~spray,data=InsectSprays)
```



It certainly looks as though the different sprays have different effects. To quantify this we can use the `lm()` idiom:

```
summary(lm(count~spray,data=InsectSprays))
```

```
##
## Call:
## lm(formula = count ~ spray, data = InsectSprays)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.333 -1.958 -0.500  1.667  9.333
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.5000     1.1322  12.807  < 2e-16 ***
## sprayB        0.8333     1.6011   0.520    0.604
## sprayC      -12.4167     1.6011  -7.755 7.27e-11 ***
## sprayD       -9.5833     1.6011  -5.985 9.82e-08 ***
## sprayE      -11.0000     1.6011  -6.870 2.75e-09 ***
## sprayF        2.1667     1.6011   1.353    0.181
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.922 on 66 degrees of freedom
## Multiple R-squared:  0.7244, Adjusted R-squared:  0.7036
## F-statistic:  34.7 on 5 and 66 DF,  p-value: < 2.2e-16
```

In the above, we can see the various estimates for the difference between spray A and the other sprays (by default, R takes the "base case" to be the first label alphabetically). Note that `spray` is a *categorical* variable. The bottom line gives an overall p-value for the null that the sprays are identical.

The tilde notation is a powerful tool and we do not have time to go in to all its functionality. For further details, consult the help page at `?formula`.

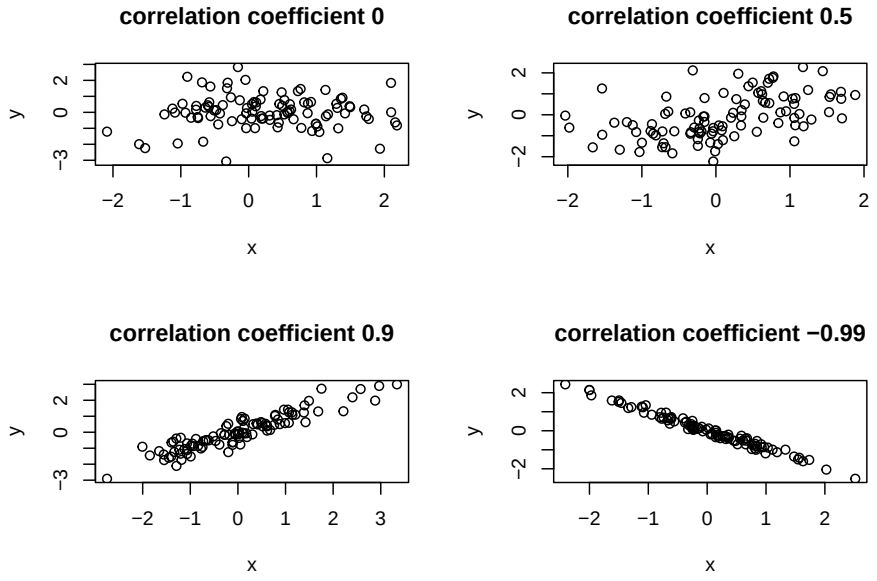
10.3.1 Correlation coefficient

The p-value measures the strength of evidence against the null, but this is not informative about the "closeness of fit" of the linear regression. To measure this, we use the correlation coefficient r where $r = 0$ corresponds to no relation r close to $+1$ corresponds to close positive association, and r close to -1 corresponds to close negative association. Intermediate values correspond to weak correlation. See the following diagram for visualization.

```
library("mvtnorm")
f <- function(n,rho){
    d <- rmvnorm(n,sigma=matrix(c(1,rho,rho,1),2,2))
```

```
 plot(d,xlab='x',ylab='y',main=paste("correlation coefficient ",rho,sep=""))
}

par(mfrow=c(2,2))
f(100,0)
f(100,0.5)
f(100,0.9)
f(100,-0.99)
```



It is common to consider r^2 which is non-negative and treats positive and negative slopes equally. in R, use `summary()` whih reports an estimate for r^2.

Chapter 11

Logistic regression

https://www.youtube.com/watch?v=VlMLL5vaLlo&index=35&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

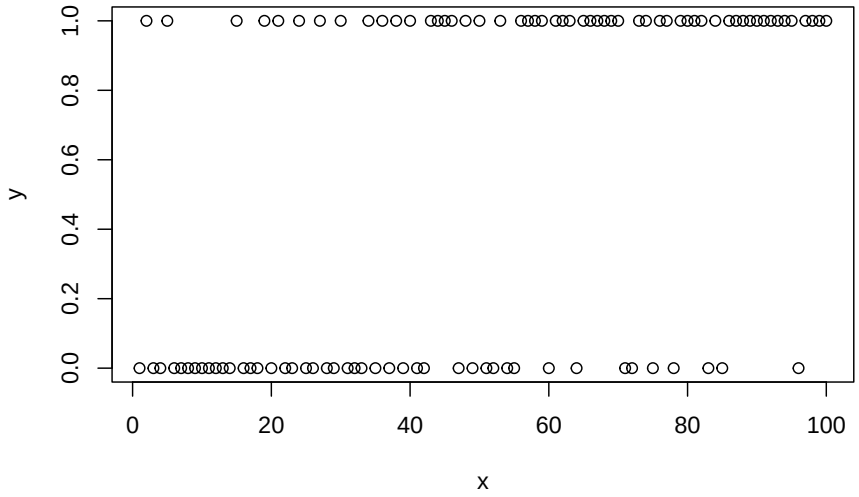
Here we continue to use the formula structure of R to deal with Bernoulli response variables. Suppose we are studying how performance improves with practice. We ask a subject to perform a task repeatedly. We expect the person to become more adept with practice, and we seek to estimate the probability of success as a function of the number of attempts. Consider the following dataset:

```
y <-
c(0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,
0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0,
1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1,
0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1,
1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 1, 1, 1, 1)
```

Here, "0" represents failure and "1" success. Thus the first trial was a fail, the second a success, and so on. Bernoulli responses such as this cannot be handled with conventional regression techniques because any straight line will either be constant, or extend beyond the $(0, 1)$ allowable range for a probability.

The first step is, as with any regression, to PLOT the data:

```
x <- 1:100
plot(y~x)
```



It is a little difficult to see, but we can perhaps detect a slight improvement in performance from left to right; the success points are a little thicker at the right of the graph compared with the left.

In order to make sense of this dataset, we do not work with probability on the vertical axis; we consider instead the *odds*. Odds are defined as the probability of success divided by the probability of failure. Thus an event which has probability p has odds $\frac{p}{1-p}$.

Regression then operates on the *log odds*, $LO = \log\left(\frac{p}{1-p}\right)$. Inverting this formula gives $p = \frac{e^{LO}}{1+e^{LO}}$; observe that this correctly ensures that p is always between 0 and 1, while log-odds can be any real value, positive or negative. The regression is then

$$LO = \alpha + \beta x$$

where α and β have the same meanings (intercept and slope) as they do in conventional regression. Inverting this formula gives

$$p = \frac{e^{\alpha+\beta x}}{1 + e^{\alpha+\beta x}}$$

These functions are easy to translate into R idiom:

```
LO <- function(p){log(p/(1-p))}  # log odds
pr <- function(LO){exp(LO)/(1+exp(LO))}
```

It is relatively straightforward to calculate a support function for α, β:

```
f <- function(params){
    alpha <- params[1]
    beta  <- params[2]
    success <- x[which(y==1)]
    failure <- x[which(y==0)]
    return(
     sum(log(  pr(alpha + beta * success))) +
     sum(log(1-pr(alpha + beta * failure)))
    )
}
```

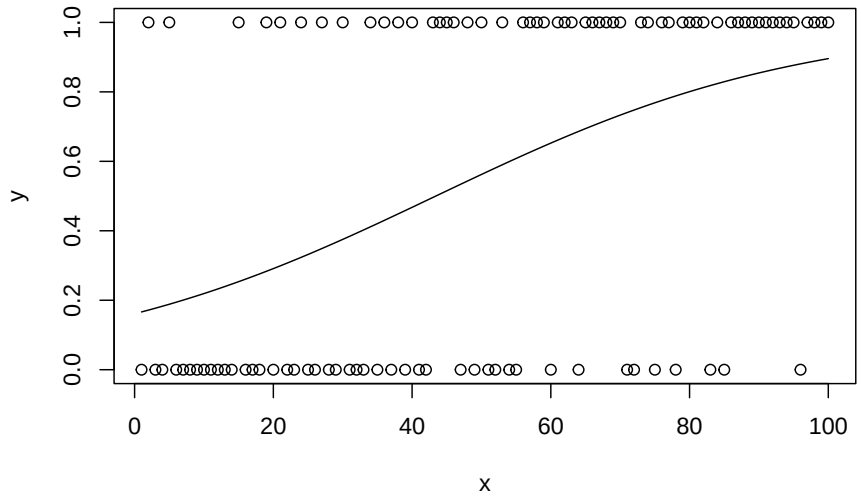
Thus `f(alpha,beta)` takes a sum of log-probabilities for the successes and a sum of logs of probabilities of failures. See how the data remains fixed while alpha and beta change. We can maximize the support numerically, using `optim()`, which is a general-purpose suite of optimization routines:

```
optim(c(-0.5, 0.01),f,control=list(fnscale= -1))  # fnscale<0 means maximize
```

```
## $par
## [1] -1.65451678  0.03784904
##
## $value
## [1] -57.14481
##
## $counts
## function gradient
##       83       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

In the above, the `$par` line gives the maximized values of `alpha, beta` as about $(-1.65, 0.038)$. This may be plotted:

```
plot(y~x)
points(x,pr(-1.65 + 0.038*x),type="l")
```



See how the logistic fit takes an "S" shape (the correct term is "ogive"). Of course, R can make things easier:

```
summary(glm(y~x,family="binomial"))
```

```
##
## Call:
## glm(formula = y ~ x, family = "binomial")
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.654403   0.477339  -3.466 0.000528 ***
## x            0.037851   0.008803   4.300 1.71e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.63  on 99  degrees of freedom
## Residual deviance: 114.29  on 98  degrees of freedom
```

```
## AIC: 118.29
##
## Number of Fisher Scoring iterations: 4
```

which gives more details. See how the coefficients in the two methods match (which one is more accurate?). Observe that the given p-value is a two sided test. We might argue that a one-sided test is more appropriate (why?) and, if so, we would simply halve it to give a one-sided value.

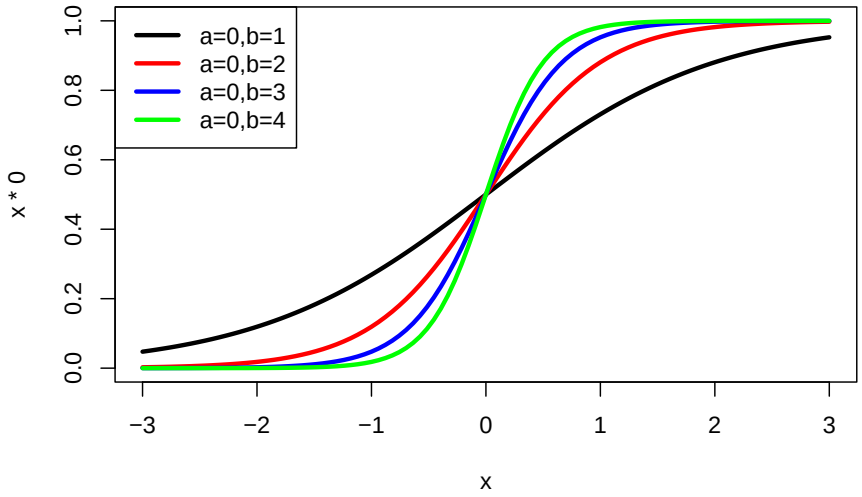
11.1 Interpretation of the coefficients

We have that

$$LO = a + bx$$

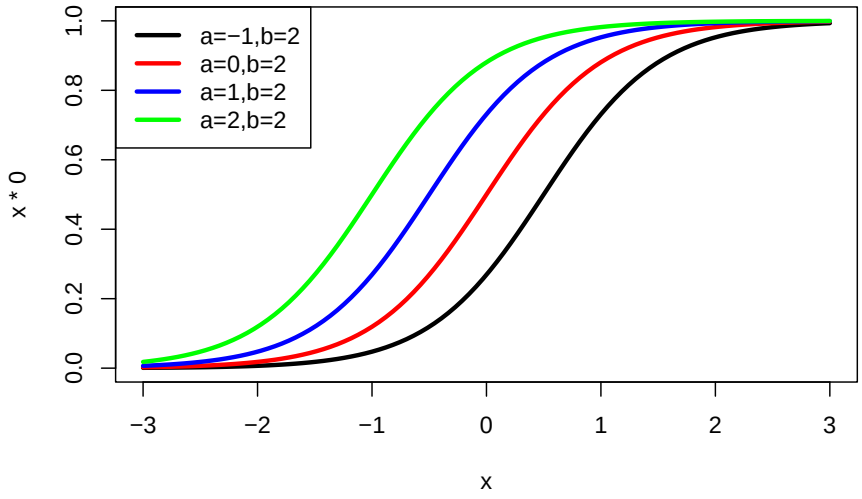
where LO is the log-odds and we have $LO = \log\left(\frac{p}{1-p}\right)$ and $p = \frac{e^{LO}}{1+e^{LO}}$; here we are using a, b rather than α, β for convenience. We are going to plot this relation for different values of α and β.

```
x <- seq(from=-3,to=3,len=100)
dolines <- function(a,b,...){points(x,pr(a+b*x),type='l',lwd=3, ...)}
plot(x,x*0,ylim=c(0,1),type="n")
dolines(0,1,col="black")
dolines(0,2,col="red")
dolines(0,3,col="blue")
dolines(0,4,col="green")
legend("topleft",col=c("black","red","blue","green"),lty=1,lwd=3,
       legend=c("a=0,b=1","a=0,b=2","a=0,b=3","a=0,b=4"))
```



In the diagram above, see how the value of b governs how abrupt the change from zero to one is: larger values of b correspond to sharper changes. Indeed, if $b < 0$ the curve slopes the other way. We are now going to change the value of a, keeping b fixed at 2:

```
x <- seq(from=-3,to=3,len=100)
dolines <- function(a,b,...){points(x,pr(a+b*x),type='l',lwd=3, ...)}
plot(x,x*0,ylim=c(0,1),type="n")
dolines(-1,2,col="black")
dolines(0,2,col="red")
dolines(1,2,col="blue")
dolines(2,2,col="green")
legend("topleft",col=c("black","red","blue","green"),lty=1,lwd=3,
       legend=c("a=-1,b=2","a=0,b=2","a=1,b=2","a=2,b=2"))
```



In the diagram above, see how the value of a governs where the steepest part of the curve occurs: changing a moves the curve left and right.

If we ask instead what value of x corresponds to $p = 0.5$, then we see that the log-odds will be $LO = \log\left(\frac{0.5}{1-0.5}\right) = \log 1 = 0$. Thus, we can find the value of x at which $p = 0.5$ by solving $a + bx = LO$ and setting $LO = 0$; that is, $a + bx = 0$, or $x = -a/b$.

Chapter 12

Quantile methods

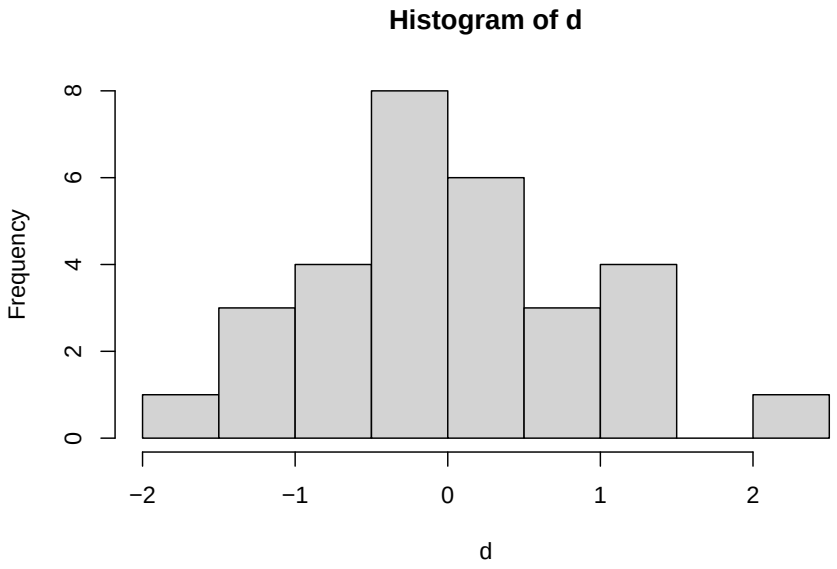
https://en.wikipedia.org/wiki/Quantile_function

YouTube

https://www.youtube.com/watch?v=wJQATecNZHk&index=36&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

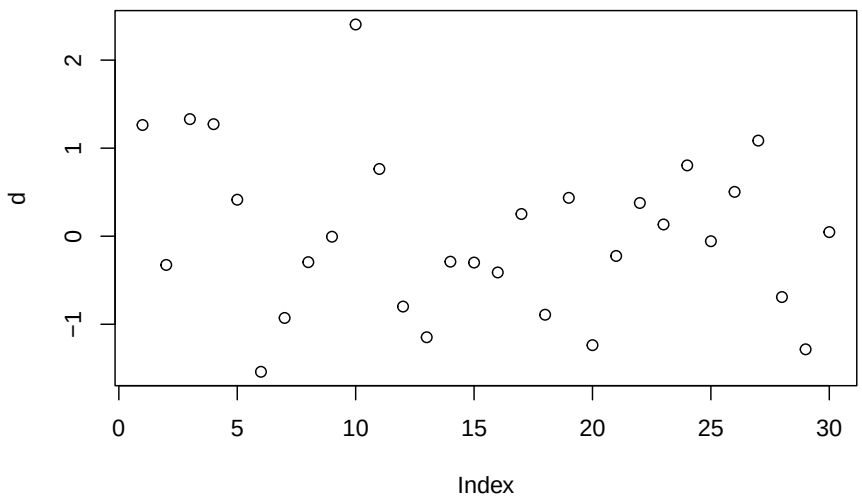
You will recall the deficiencies of the histogram for independent observations. Consider, for example, the following diagram:

```
set.seed(0)
d <- rnorm(30)
hist(d)
```



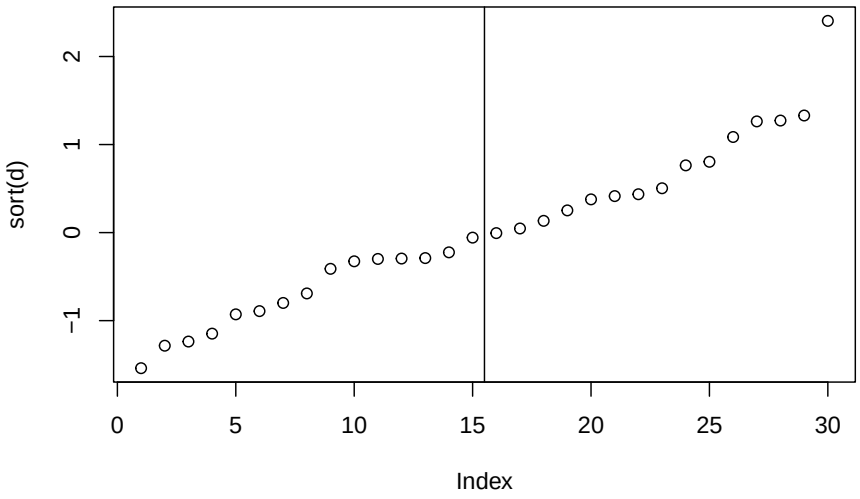
In the above, we see that the precise values of the observations are hidden by the bins of the histogram. We can of course plot the entire dataset:

```
plot(d)
```



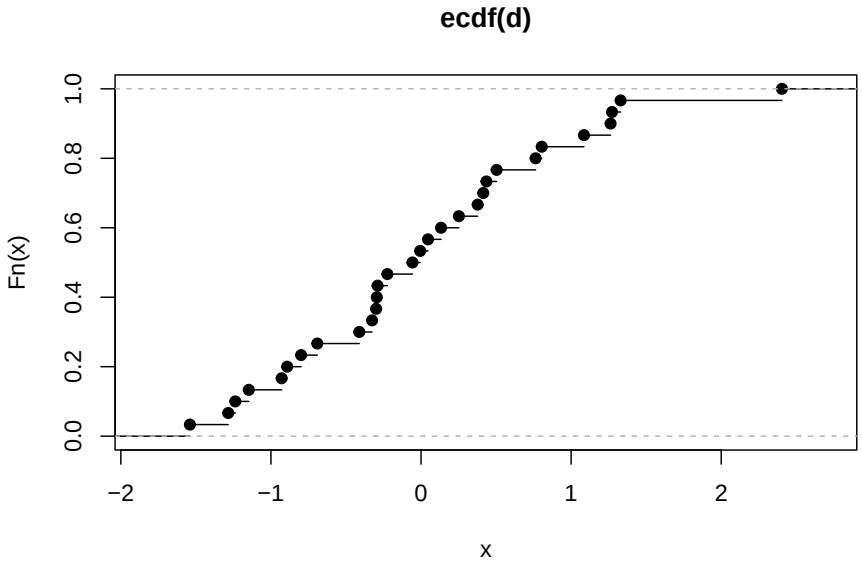
but this introduces a spurious and possibly misleading aspect to the plot, namely the order in which the datapoints are plotted. Because the observations are independent, the order in which they appear can should have no effect on any inferences we make. We can exploit this fact by plotting the *order statistic*, which is the dataset we observe but sorted from smallest to largest:

```
plot(sort(d))
abline(v=15.5)
```



In the figure above, the median has been shown as a vertical line. However, it is more convenient to transpose the axes, and the R idiom for this is `ecdf()` [the letters stand for empirical cumulative distribution function]:

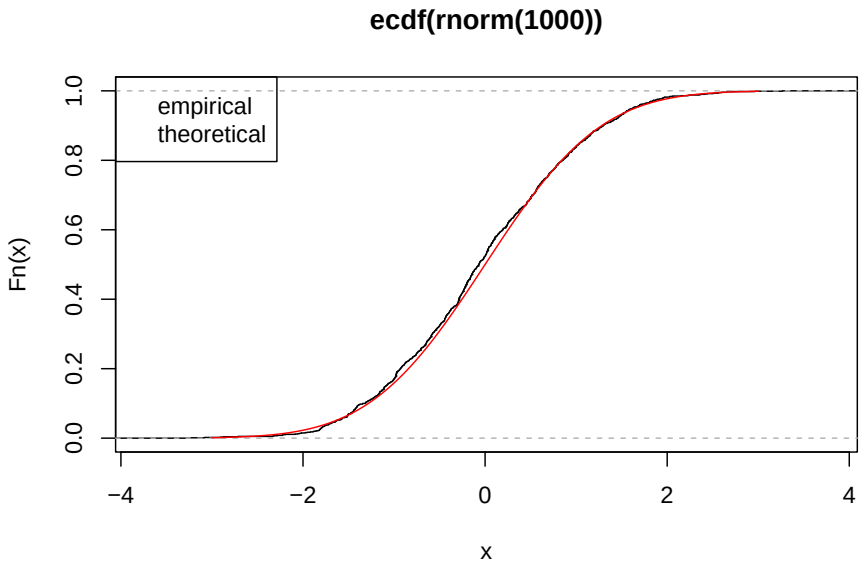
```
plot(ecdf(d))
```



In the figure above, the vertical axis has been scaled to represent a probability rather than a count, and horizontal lines have been added for convenience. The function shown is an empirical [data-driven] approximation to the cumulative distribution function, $F(x) = \text{Prob}\,(X \leq x)$. Note that the points appear at the *left* of the horizontal lines, showing that the function approximated is $\text{Prob}\,(X \leq x)$ and not $\text{Prob}\,(X < x)$.

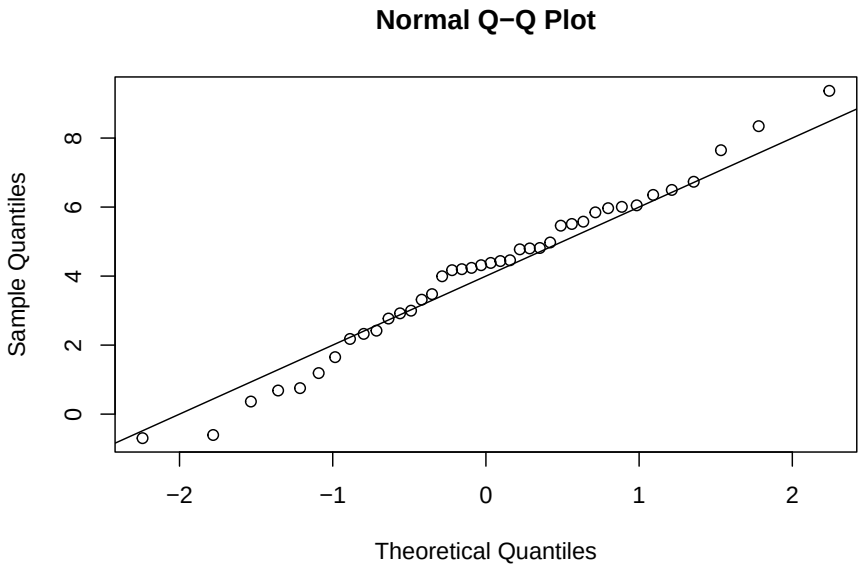
We can use R to investigate the properties of the empirical cumulative distribution function for Gaussian data with a large number of observations:

```
plot(ecdf(rnorm(1000)))
x <- seq(from=-3,to=3,len=100)
points(x,pnorm(x),col="red",type="l")
legend("topleft",col=c("black","red"),legend=c("empirical","theoretical"))
```



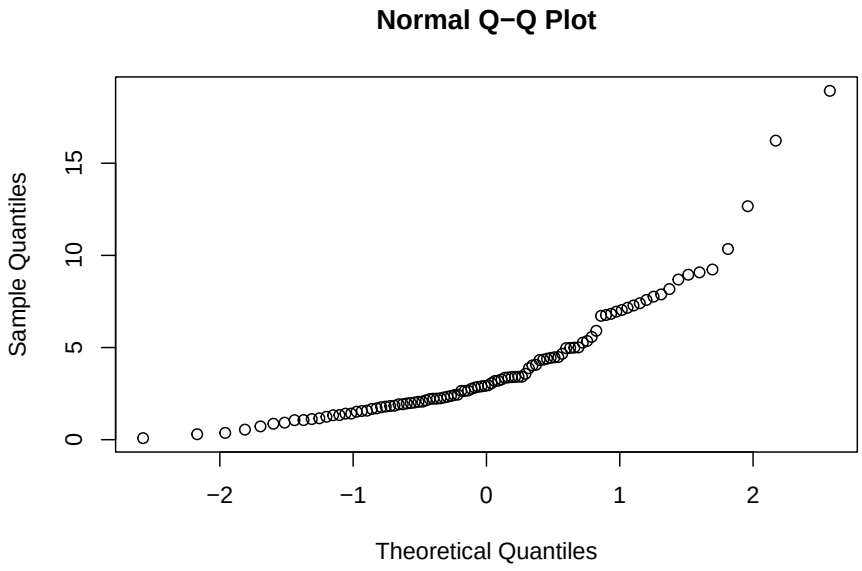
In the figure above, see how the distribution function `pnorm()` of the Gaussian is approximated. However, the theoretical (red) line is not straight, and this makes visual assessment harder. It would be a nice if we could distort the horizontal axis so that the theoretical line is straight, and this would make visual comparison easier. The appropriate way to do this is to transform the horizontal axis with the `pnorm()` function. The technique works nicely even if the mean and standard deviation are modified; the appropriate R idiom for this is `qqnorm()`:

```
qqnorm(rnorm(40,mean=4,sd=2))
abline(4,2)
```



In the figure above, see how the data fall approximately on a straight line; it turns out that the mean is approximated by the intercept and the standard deviation by the slope. The exact values are drawn on the diagram as the diagonal line. The `qqnorm()` function may be used to detect non-normality. Suppose we sample from a chi-squared distribution:

```
qqnorm(rchisq(100,df=4))
```

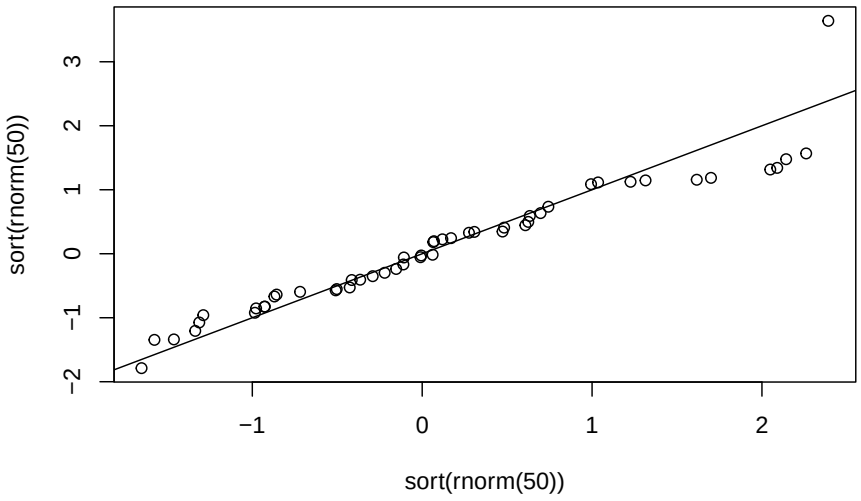


In the figure above, we see a marked curve on the line indicating that the distribution is not Gaussian.

12.1 Quantile-quantile plots

We may apply similar techniques to compare two datasets without losing any data as would be the case if we simply drew two histograms. By plotting the two datasets' order statistics against each other, (one on the x axis and one on the y axis), we have a powerful visual tool to compare two datasets:

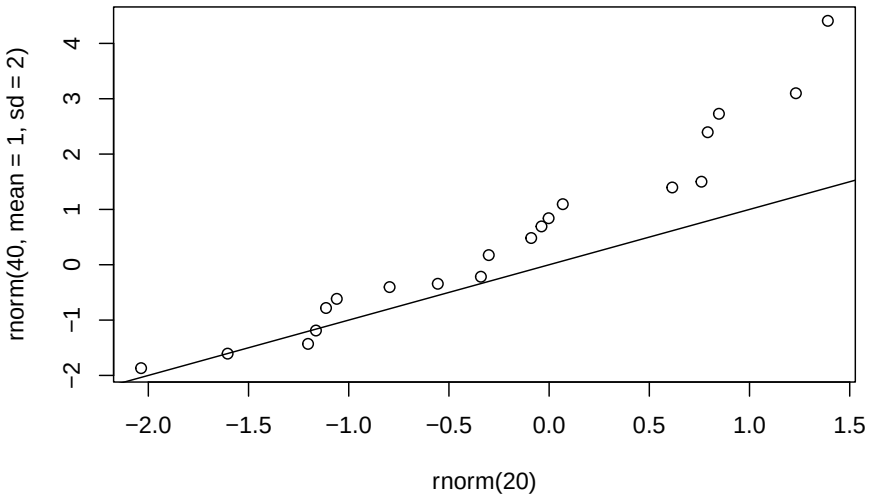
```
plot(sort(rnorm(50)),sort(rnorm(50)))
abline(0,1)
```



In the above, we are plotting the smallest observation against the smallest, the second smallest against the second smallest, and so on, up to the largest observation. If the two datasets are drawn from the same distribution then the points will be close to the 45° line, as they are in this case.

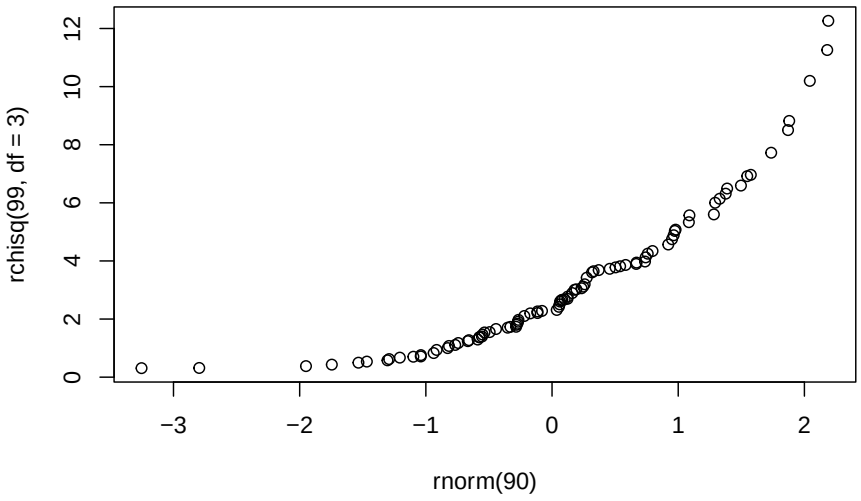
If we have different numbers of observations then the R idiom gets trickier and we can use the builtin function `qqplot()`:

```
qqplot(rnorm(20),rnorm(40,mean=1,sd=2))
abline(0,1)
```



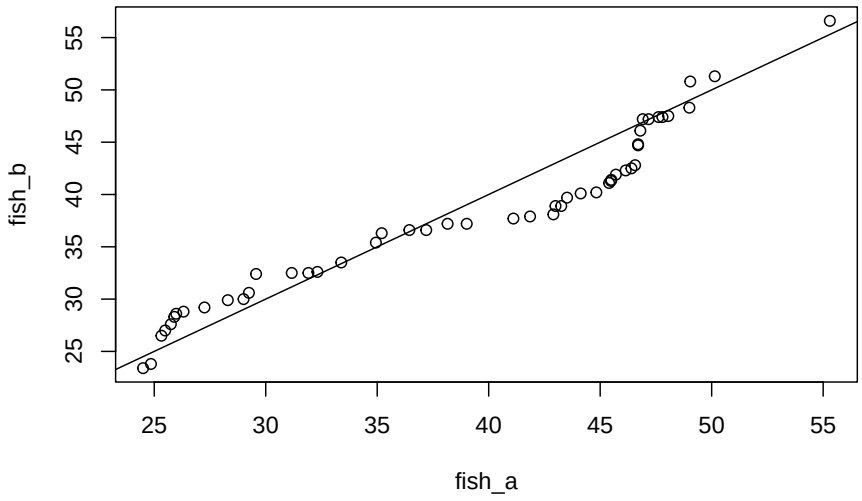
In the above figure, the points are not close to the 45° line showing that the two distributions are different (note that the function deals with different sample sizes: 20 and 40). However, the points do fall on a *straight* line which shows that the two distributions are related by a linear transform, in this case we can see that the means and standard deviations differ. If the distributions are of different shapes, then this shows up as a curved line:

```
qqplot(rnorm(90),rchisq(99,df=3))
```



In the above, we plot quantiles of normal random data against chi-squared random data, and because these are different distributions we see a curved line. We can use these techniques in a more practical setting. Suppose we have two lakes and we sample fish in the lakes, measuring their lengths:

```
fish_a <-
c(31.2, 26.3, 24.5, 25.9, 29.5, 29.1, 36.7, 28.3, 32.2, 29.0, 25.4,
  25.2, 37.9, 35.1, 24.7, 34.9, 29.6, 31.8, 26.0, 25.9, 25.5, 35.3,
  25.7, 28.3, 32.4, 39.2, 33.5, 26.4, 41.3, 37.1, 44.2, 49.0, 46.2,
  41.1, 55.3, 47.6, 44.7, 45.5, 45.4, 46.7, 47.0, 46.7, 46.4, 43.6,
  43.0, 47.9, 45.5, 46.0, 42.9, 43.4, 48.1, 38.5, 46.5, 45.4, 46.8,
  46.8, 50.6, 43.2, 42.9, 45.5, 49.1, 47.7, 49, 46.7, 47.2)
fish_b <-
c(44.7, 42.5, 38.9, 38.9, 28.8, 27.6, 44.8, 56.6, 51.3, 40.2, 35.4,
  39.7, 47.2, 40.1, 36.6, 37.2, 33.5, 42.8, 42.3, 26.5, 37.7, 28.3,
  47.2, 37.2, 41.3, 29.2, 32.4, 28.6, 30.0, 38.1, 41.4, 36.6, 27.0,
  41.1, 50.8, 36.3, 47.5, 47.4, 32.6, 41.9, 32.5, 32.5, 48.3, 37.9,
  47.4, 23.4, 30.6, 23.8, 46.1, 29.9)
qqplot(fish_a, fish_b)
abline(0,1)
```



The above figure gives evidence that the two populations are different. To provide quantitative evidence for a difference, we need techniques discussed in the next chapter.

Chapter 13

Nonparametric statistics

https://en.wikipedia.org/wiki/Nonparametric_statistics

https://www.youtube.com/watch?v=6vuf753mo64&index=37&list=PL018X5Hlr4RkgE65Pg93TFY-32KCVpW84

Up to now, all the techniques that have been used assume that the underlying distribution has got a particular form. The Student t-test, for example, assumes that the observations are drawn from a Gaussian distribution. Calculation of p-values in general requires knowledge of the null distribution, whether this is binomial, hypergeometric or another parametrically determined distribution. In this chapter, we show some techniques that do not require any such assumptions and operate correctly independently of the underlying distributions. The general word for such techniques is *nonparametric* methods.

13.1 Kolmogorov Smirnov test

https://en.wikipedia.org/wiki/Kolmogorov=Smirnov_test

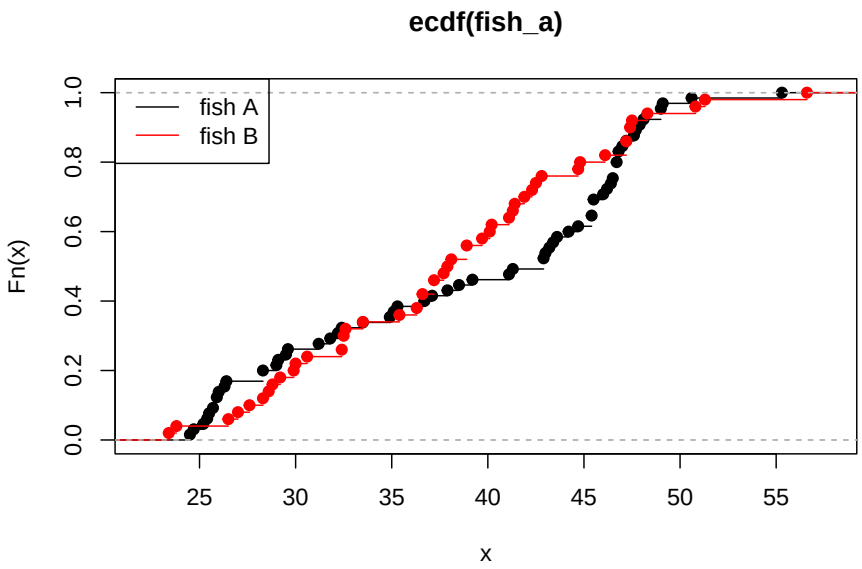
Consider the example in the previous chapter, with the two fish populations, reproduced here for convenience:

```
fish_a <-
c(31.2, 26.3, 24.5, 25.9, 29.5, 29.1, 36.7, 28.3, 32.2, 29.0, 25.4,
  25.2, 37.9, 35.1, 24.7, 34.9, 29.6, 31.8, 26.0, 25.9, 25.5, 35.3,
  25.7, 28.3, 32.4, 39.2, 33.5, 26.4, 41.3, 37.1, 44.2, 49.0, 46.2,
```

```
  41.1, 55.3, 47.6, 44.7, 45.5, 45.4, 46.7, 47.0, 46.7, 46.4, 43.6,
  43.0, 47.9, 45.5, 46.0, 42.9, 43.4, 48.1, 38.5, 46.5, 45.4, 46.8,
  46.8, 50.6, 43.2, 42.9, 45.5, 49.1, 47.7, 49, 46.7, 47.2)
fish_b <-
c(44.7, 42.5, 38.9, 38.9, 28.8, 27.6, 44.8, 56.6, 51.3, 40.2, 35.4,
  39.7, 47.2, 40.1, 36.6, 37.2, 33.5, 42.8, 42.3, 26.5, 37.7, 28.3,
  47.2, 37.2, 41.3, 29.2, 32.4, 28.6, 30.0, 38.1, 41.4, 36.6, 27.0,
  41.1, 50.8, 36.3, 47.5, 47.4, 32.6, 41.9, 32.5, 32.5, 48.3, 37.9,
  47.4, 23.4, 30.6, 23.8, 46.1, 29.9)
```

We observed that the `qqplot()` technique suggested a difference. One way to refine this is to plot their empirical cumulative distribution functions on the same axes:

```
plot(ecdf(fish_a))
plot(ecdf(fish_b),add=TRUE,col='red')
legend("topleft",col=c("black","red"),legend=c("fish A", "fish B"),lty=1)
```



The two ECDFs show a difference, but is it significant? To answer this, we need to quantify the difference between the two curves. One way to do this is to find the maximum (vertical) distance between the curves. From the graph, it looks like this is at about $x = 42$; at this point, the black line is at about 0.49 and the red line is at about 0.76, giving a difference D of about $0.76 - 0.49 = 0.33$.

It turns out that D has a certain probability distribution—a Kolmogorov distribution—if the two datasets are indeed drawn from the same distribution. The details are messy (see the wikipedia page for details) but R has a builtin function:

```
ks.test(fish_a,fish_b)
```

```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  fish_a and fish_b
## D = 0.26769, p-value = 0.0258
## alternative hypothesis: two-sided
```

showing, by virtue of the p-value of 3.5%, that the difference is indeed statistically significant.

Note carefully that the Student t-test does not reveal a difference:

```
t.test(fish_a,fish_b)
```

```
##
##  Welch Two Sample t-test
##
## data:  fish_a and fish_b
## t = 0.54913, df = 110.61, p-value = 0.584
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.216193  3.915270
## sample estimates:
## mean of x mean of y
##  38.86154  38.01200
```

This is because the t-test tests a null of identical means under the assumption that the observations are Gaussian, which is not the case here.

In general, the KS test is not very powerful when compared with more specialist tests (with specific alternatives) such as the Student t-test. But the KS test makes very few assumptions about the distribution from which the observations are drawn, and it is a popular and robust test in many scientific disciplines.

13.2 Mann-Whitney-Wilcoxon test

https://en.wikipedia.org/wiki/Mann%E2%80%93Whitney_U_test

The Mann-Whitney-Wilcoxon test is an even more general test than the Kolmogorov test discussed above. It tests a rather peculiar null that is more general than that of the Kolmogorov test, specifically that it is equally likely that a randomly selected value from one sample will be less than or greater than a randomly selected value from a second sample. The test is often used for races in which the actual times are not important, and only the order of finishing matters.

The test statistic is the total, over observations a in dataset A, of observations dataset B beaten by a. As an example, suppose we have two cycling teams, red and blue. We stand at the finishing line and observe the colour of the cyclists as they cross the line:

```
Red <- 0
Blue <- 1
d <- c(Red, Blue, Blue, Blue, Blue, Blue, Red, Red, Red, Red, Red, Blue)
```

Thus the first across the line was Red, the second Blue, and so on. We take each Red in turn, and count the number of Blues he beats, getting 6, 1, 1, 1, 1, 1. We add these to get $U = 11$ (alternatively, we could take each Blue in turn, and count the number of Red he beats).

If the null is true, then U has approximately a Gaussian distribution with mean $n_1 n_2/2$ and variance $\frac{n_1 n_2 (n_1+n_2+1)}{12}$.

All of this is calculated in R using `wilcoxon.test()`:

```
wilcox.test(which(d==0),which(d==1))
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  which(d == 0) and which(d == 1)
## W = 25, p-value = 0.3095
## alternative hypothesis: true location shift is not equal to 0
```

showing no signficant difference. The Wilcoxon test has many variants, documented under `?wilcox.test`, and the R function has many arguments; only the simplest case is presented here.