

Quantitative Methods in Linguistics – Lecture 5

Adrian Brasoveanu*

April 29, 2014

Contents

1	Introducing least squares models	1
1.1	Data generation	1
1.2	First attempt: the mean of y	4
1.3	Quantifying the error	7
1.4	From sample to population: basic statistical (inductive) inference	12
1.4.1	Sample size	13
1.4.2	Sample variance	15
1.4.3	Putting the two together: the standard error (SE) of the mean, the Central Limit Theorem (CLT), and 95% confidence intervals (CIs)	18
1.5	Understanding the <code>lm</code> output a little bit better: t-distributions	22
2	Applying the Central Limit Theorem (CLT) to Bernoulli distributed data	26
2.1	The Bernoulli distribution	26
2.2	The binomial distribution	30
2.3	The binomial distribution and the Central Limit Theorem	32

1 Introducing least squares models

1.1 Data generation

Let's simulate some data:

- we generate values for two predictors, a continuous one (x_1) and a binary categorical one (x_2)
- we generate the response values y by combining the predictors to obtain a set of mean y values and adding some normally-distributed noise around those values

We first simulate the continuous predictor. You can think about it as drawing 100 heights (in inches) centered at 64 inches and with a standard deviation of 3. This is a rough approximation of the distribution of height for females in the US.

*These notes have been generated with the 'knitr' package (Xie 2013) and are based on many sources, including but not limited to: Abelson (1995), Miles and Shevlin (2001), Faraway (2004), De Veaux et al. (2005), Braun and Murdoch (2007), Gelman and Hill (2007), Baayen (2008), Johnson (2008), Wright and London (2009), Gries (2009), Kruschke (2011), Diez et al. (2013), Gries (2013).

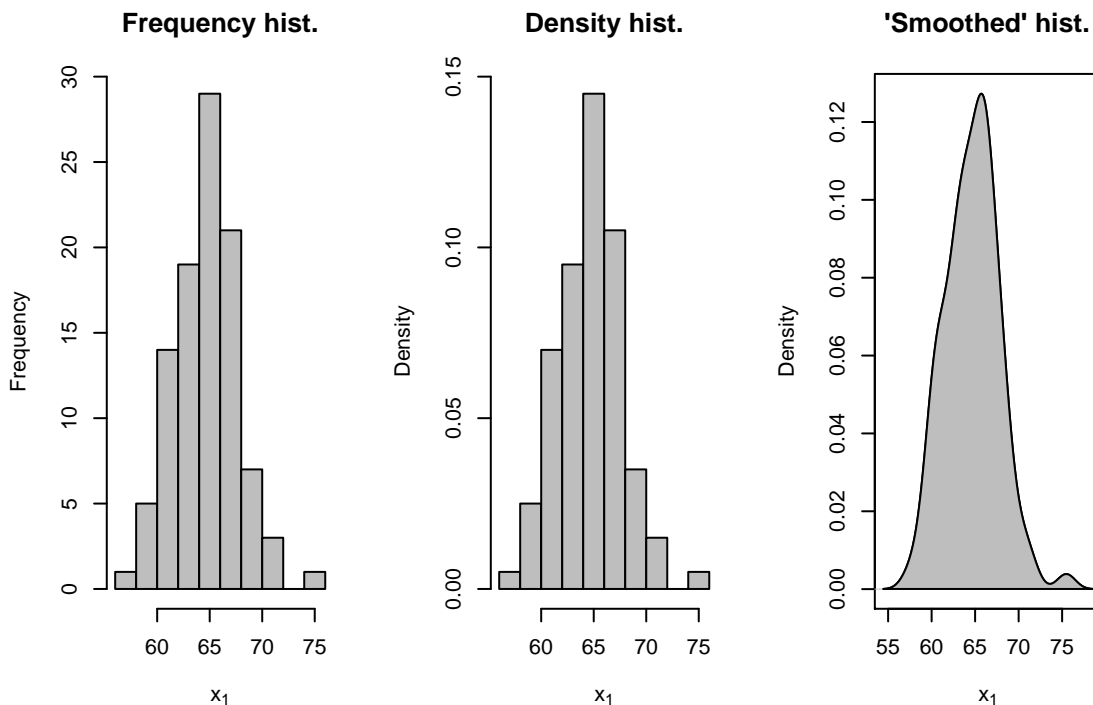
```

> x1 <- rnorm(100, 64, 3)
> summary(x1)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  57.5   62.7   64.8   64.7   66.5   75.5

> par(mfrow = c(1, 3))
> hist(x1, col = "gray", main = "Frequency hist.", xlab = expression(x[1]))
> hist(x1, freq = FALSE, col = "gray", main = "Density hist.", xlab = expression(x[1]))
> plot(density(x1), main = "'Smoothed' hist.", xlab = expression(x[1]))
> polygon(density(x1), col = "gray", border = "black")

```



```

> par(mfrow = c(1, 1))

```

We now simulate the categorical predictor, which will measure / encode the gender of the person we observe. We will let 1 stand for male and 0 for female. This numerical recoding of gender is called a *contrast*: it is a way to take the levels of a factor (i.e., the possible outcomes of a categorical variable) and recode them in terms of vectors of integers – vectors of 1s and 0s in the simplest case. These vectors are called ‘dummy variables’.

There is nothing special about any particular contrast: we set it up in the way that is the most appropriate for our analytical / simulation goals. We’ll come back to this, but for now you should be aware that R will always internally set up a particular contrast for all your categorical variables, and the interpretation of all estimates, tests etc. are based on those contrasts.

We simulate the gender of the person by flipping a fair coin ($\theta = 0.5$) 100 times. A more realistic θ would actually be slightly lower than 0.5 since there are more females than males in the US population – see here, for example: http://en.wikipedia.org/wiki/List_of_countries_by_sex_ratio.

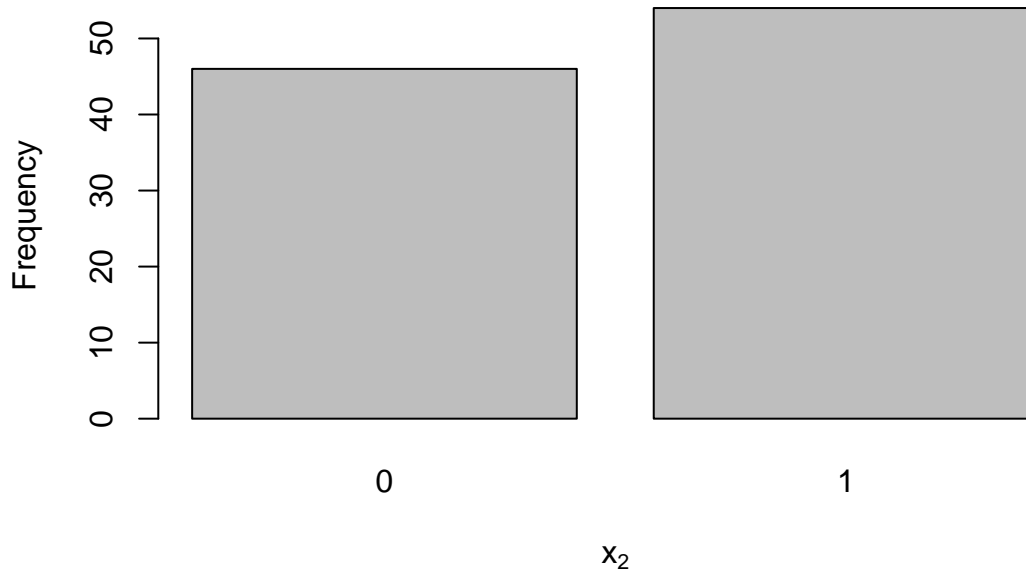
```

> x2 <- rbinom(100, 1, 0.5)
> sum(x2)

[1] 54

> plot(as.factor(x2), col = "gray", xlab = expression(x[2]), ylab = "Frequency")

```



Finally, we simulate the response variable y , which is intended to measure the weight (in lbs) of the person we observe:

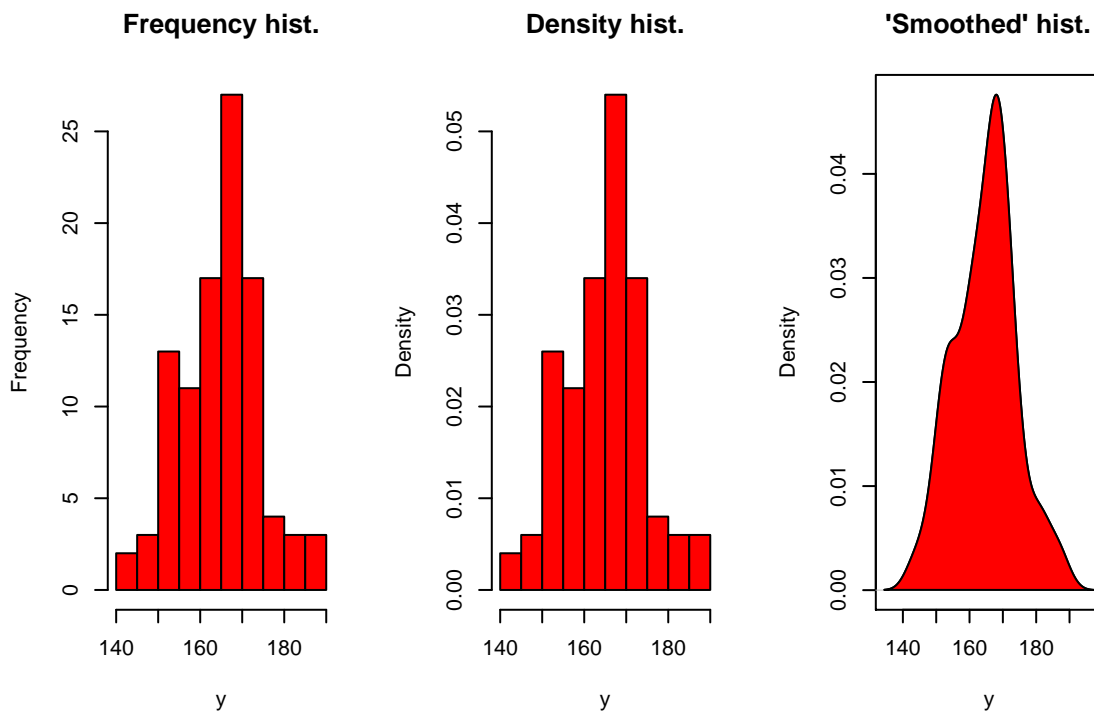
```

> y <- x1 * 2.5 + x2 * 6 + rnorm(100, 0, 4)
> summary(y)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   144    159    166    165    170    188

> par(mfrow = c(1, 3))
> hist(y, col = "red", main = "Frequency hist.")
> hist(y, freq = FALSE, col = "red", main = "Density hist.")
> plot(density(y), main = "'Smoothed' hist.", xlab = "y")
> polygon(density(y), col = "red", border = "black")

```



```
> par(mfrow = c(1, 1))
```

What is the (informal) interpretation of the coefficients for x_1 and x_2 , a.k.a. the β s? That is, what is the interpretation of $\beta_{x_1} = 2.5$ and $\beta_{x_2} = 6$?

What is the (informal) interpretation of the standard deviation of 4 for the 'noise' / 'error' / random variation in y values?

One basic goal of statistical analysis: **prediction**. Another goal – even more relevant to us – is inferring (causal) relationships, but let's ignore that for the moment.

So, assume y is the weight (in lbs), x_1 is the height (in inches) and x_2 is the gender (male=1 vs. female=0) of a randomly selected sample of 100 people from a population of interest (the US human population in this case).

Now, we have a new person from the same population. We determined their height and gender and we want to estimate their weight. What do we do?

1.2 First attempt: the mean of y

We can take the mean of the vector of responses y to be our prediction for the weight of the new person. If we didn't know the height and gender of the new person, this would actually be the best we could do. And even if we know the height and gender of the new person, we still have to check whatever purported predictive edge these predictors give us against this very basic model.

```
> sum(y)/length(y)
[1] 164.8

> mean(y)
[1] 164.8
```

The mean is a very simple model of weight. In general:

(1) $DATA = MODEL + ERROR$

- a. the **MODEL** is the **deterministic** (rule/law-like) part of our theory of the data; it gives us the (possibly very complex) rule to obtain the average / mean / central tendencies in our DATA
- b. the **ERROR**, a.k.a. the noise, is the **non-deterministic** (random/probabilistic) part of our theory; it gives us the probability distribution of the *actual* observations in the data, i.e., their probabilistic pattern of variation around the central tendency described by the deterministic component

I will often use **model** more generally for the entire theory, i.e., for the deterministic and non-deterministic parts put together, instead of simply for the deterministic part. But for now, we'll stick to the narrow interpretation.

The mean is the simplest **least squares** model, where:

- **squares** refers to a way of measuring the error, and
- **least** indicates that we want to minimize the error (as measured in terms of squares).

Note that *data*, *model* and *error* are not used here in the sense they are (more or less implicitly) used in generative linguistics (syntax, formal semantics etc.):

- what counts as data for the purposes of generative linguistics is *generalizations*
- what counts as a model is the proposed *analysis* that accounts for the target data / generalizations, and the background theoretical framework, e.g., first-order logic, context-free grammars etc.
- what counts as error is the *unaccounted-for generalizations*

The **model in the present context** consists of what generative linguists call generalizations, i.e., the *patterns* in the data that we try to ascertain.

Theoretical linguists (some of them) think that they have an uncanny talent for 'seeing' or 'intuiting' patterns in the data. As trained, gifted linguists, we do have a lot of experience with language and language analysis, and often the patterns we see are real patterns – but not always: what seems 'intuitively' really obvious to us might be incorrect.

The main idea that underlies all statistical **arguments** (yes, arguments, not mindlessly applied 'tests' / 'procedures' – see Abelson 1995) is that we never know if what we see are patterns or just random noise. Random noise should be the default assumption ('null hypothesis'): what we think we see is not really there, we're just spotting shapes in the clouds, we're not hearing a loud-and-clear message from 'nature'. The burden of proof is on the theory that posits patterns / generalizations, not the other way around.

The **data in the present context** is whatever primary measurements you have. Which means that yeah, you have to go out there and take some behavioral measurements first. Introspective judgments – and the verbal protocols associated with them – are a good starting point, but are not the only source of information about language structure, and are not privileged in any way that can withstand careful scrutiny.

Psychology started out in the late 1800s as a science with one of its main goals to settle empirically the issue of the origins of knowledge. The field had a great deal of difficulty initially in making any headway on the issue. The problem is that scientists need to have agreed-upon data and it took psychology a while to establish consensus about what its data were. The introspectionists [...] took as their data self-observations of the content of thought. The problem was that different researchers with different theories would observe different things about their internal thoughts that confirmed their different theories. For instance, some introspectionists claim they could have thought devoid of sensory content while others claimed they could not. Because of irresolvable controversies like this, it became clear that a more objective data source was required.

This was one of the stimuli for the behaviorist movement that began around 1920, a movement much misunderstood, particularly by many of the behaviorists who practiced it. The behaviorist movement began with the observation that the prime source of objective data was recordings of the behavior of people. There were other possible sources of objective data such as physiological recordings but these turn out to be much harder to obtain than behavior. There are two essential features that distinguish behavioral data from introspective data. The first is that it is equally available to all scientists and not the private domain of the scientist who is having the introspection. Second, the psychologist is not constrained as to how to

theoretically interpret the data. Thus, if a subject of a psychological experiment says 'I have a visual image of a cat', the scientist is free to propose any theory that will produce that verbal protocol and is not required to propose visual images as part of the theory. Because of the similarity of verbal protocols to introspective reports, many behaviorists have refused to admit verbal protocol data. However, [...] verbal protocols are very appropriate and powerful sorts of behavioral data, when treated as behavioral data and not as introspective data.

However, there was a second point of motivation in stressing behavior as the measure by which a theory of knowledge will be assessed. This arose out of an emphasis on the functional nature of human knowledge. There was no point in making distinctions about knowledge that did not have consequences for behavior. If the person behaved the same whether he possessed knowledge X or not, in what sense does he really know X? If two pieces of knowledge result in the same behavior in what sense are they really different? Such arguments should be quite familiar to the AI community where we commonly talk about equivalence among different knowledge representation schemes. This point of view also anticipates the Turing-type tests for deciding if a system is intelligent.

The behaviorists frequently argued that there was no such thing as knowledge in the abstract; when we speak of someone having certain knowledge we mean that the person has certain behavioral potentials. This led to prohibitions against discussing mental structures and a claim that an objective science should only talk about behavior. Here we see a basically correct observation being carried to unfortunate extremes. Behaviorism can be separated into a methodological and a theoretical position:

(1) *Methodological*. Behavioral data is the major data for deciding among theories of knowledge. Different theories that imply no difference for the behavior (including verbal) of the system might as well be regarded as notational variants.

(2) *Theoretical*. The terms of a theory should be behavioral. Since only external behavior counts, a theory should not make reference to underlying mental structures.

The fundamental error in behaviorism comes from extending the methodological prescription to the theoretical prohibition. Given that they were prohibited from theorizing about mental structure it is not surprising that the behaviorist theories tended to take strong empiricist stands and claim all knowledge arose directly from experience. A methodology that denied the existence of a mind denied the possibility of a contribution of the mind to knowledge.

[...]

The behaviorist learning theories of the first half-century, while they were rigorous in their choice of data, were fundamentally sloppy in their theorizing. They never really showed that the mechanisms of their theories could be put together to account for the complexities of human behavior. The advent of more computationally oriented theories in the 1950s provide the basis for exploring the issues of what could actually be computed in these frameworks. It became clear that there were serious inadequacies. Chomsky's criticism of Skinner's account of language was the most dramatic instance of such analysis, but there were many others. The learning theorists had insisted that they could account for behavior with theories whose only terms were objectively observable stimuli and responses. It became apparent that one needed to make reference to non-observable mechanisms to account for human behavior.

(Anderson 1989, pp. 315-316)

The **error in the present context** is the noise in the data brought in by measurement error, uncontrolled-for variables, or the probabilistic nature of the phenomenon itself (if applicable).

The goal is to establish the best generalization / model, i.e., get the data/measurements with the minimum amount of error (empirically optimal account) and the minimum number of predictors (Occam's razor; theoretically optimal account). Minimizing error and minimizing predictors are usually opposite criteria of evaluation – having more predictors leads to a better data fit / less error.

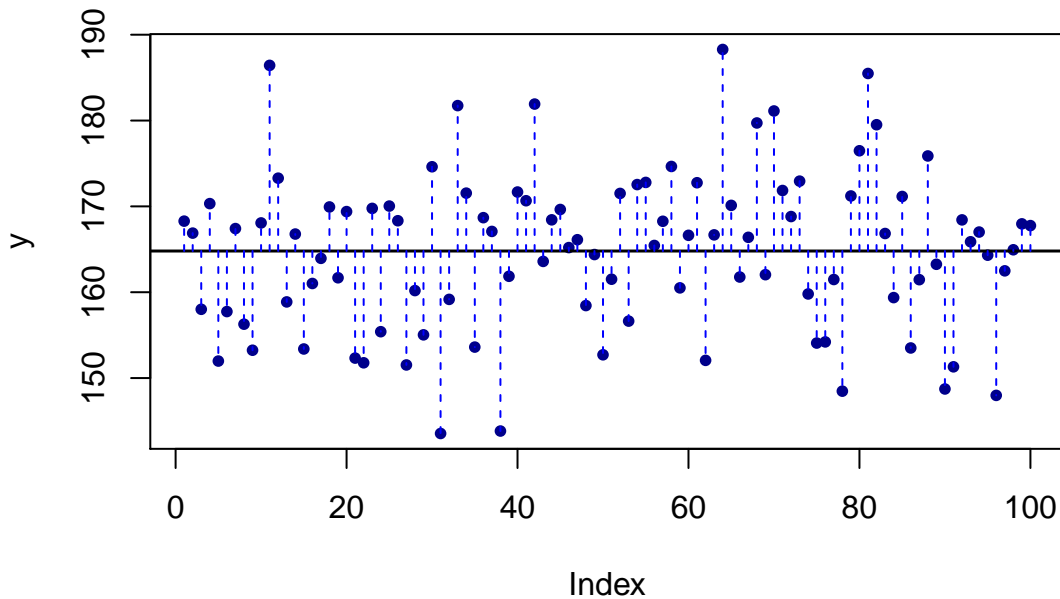
One way to think about regression models in general (and I'm using 'model' here in the broad, not narrow sense), and least-squares models in particular is as way of **summarizing the data in a theoretically informed way**, which is what generalizations are. They do not explain patterns, they only identify them. But this process of identification is theoretically informed, it is not theoretically agnostic: we include particular predictors in a particular combination in the deterministic part of our models.

Ideally, this distinction between what counts as quantitative / statistical models and what counts as theoretical / linguistic analyses ends up being pretty blurry: as statistical models become more and more sophisticated, i.e., as they incorporate more and more of our prior theoretical knowledge, they become more like explanatory accounts rather than simply descriptive generalizations.

```

> plot(y, pch = 20, col = "darkblue")
> abline(mean(y), 0, lwd = 1.5)
> errors <- (y - mean(y))
> segments(seq(1, 100), rep(mean(y), 100), seq(1, 100), y, lty = 2,
+          col = "blue")

```



```

> round(summary(errors), 2)

```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-21.30	-6.03	1.46	0.00	5.37	23.50

The sum of the errors for individual responses, a.k.a. the sum of the residuals (the 'residual' part of the data not accounted for by the deterministic component), is always 0. Why?

```

> sum(errors)
[1] 1.137e-12

> round(sum(errors), 4)
[1] 0

```

1.3 Quantifying the error

A standard way of quantifying the error associated with a model is to:

- i.* square the errors (this ensures that all the errors are positive),
- ii.* sum the squares, and

iii. divide the sum by the number of observations to obtain the Mean Squared Error (MSE).

Actually, we divide by the number of observations minus 1 because we already spent 1 degree of freedom (dof) on the mean of the observations y when we used that mean to compute the errors. Please read Walker (1940) (available on eCommons in the same folder as these lecture notes) for a leisurely introduction to the concept of degrees of freedom.

We need to divide by the number of observations (minus 1) because otherwise calculating the error for the same model but relative to 200 observations rather than 100 would roughly double the error. This is not right, since it's the same model and it's should be just as good / bad for 100 or 200 observations.

Thus, we are after the mean error of the model with respect to our data. This mean squared error (MSE) of a vector of observations is known as the **variance** of that vector. We can calculate by hand or with a quick command:

```
> squared.errors <- (y - mean(y))^2
> data.frame(errors = round(errors, 2), squared.errors = round(squared.errors,
+ 2))
```

	errors	squared.errors
1	3.48	12.13
2	2.09	4.36
3	-6.79	46.07
4	5.53	30.53
5	-12.82	164.27
6	-7.05	49.72
7	2.62	6.85
8	-8.52	72.60
9	-11.56	133.54
10	3.27	10.71
11	21.63	467.83
12	8.49	72.04
13	-5.93	35.14
14	1.98	3.91
15	-11.41	130.19
16	-3.78	14.30
17	-0.85	0.72
18	5.14	26.38
19	-3.11	9.70
20	4.59	21.10
21	-12.47	155.59
22	-13.02	169.48
23	4.97	24.68
24	-9.40	88.29
25	5.24	27.45
26	3.53	12.46
27	-13.27	176.11
28	-4.61	21.23
29	-9.76	95.32
30	9.82	96.39
31	-21.26	451.81
32	-5.63	31.73
33	16.94	286.96
34	6.75	45.50
35	-11.19	125.19
36	3.89	15.10

37	2.28	5.22
38	-20.97	439.68
39	-2.94	8.65
40	6.88	47.28
41	5.86	34.32
42	17.12	293.16
43	-1.21	1.46
44	3.64	13.22
45	4.84	23.44
46	0.40	0.16
47	1.32	1.74
48	-6.35	40.34
49	-0.41	0.17
50	-12.09	146.10
51	-3.28	10.79
52	6.71	45.04
53	-8.16	66.64
54	7.74	59.95
55	7.99	63.88
56	0.65	0.42
57	3.47	12.02
58	9.85	97.08
59	-4.29	18.43
60	1.83	3.34
61	7.96	63.28
62	-12.74	162.21
63	1.88	3.54
64	23.48	551.34
65	5.32	28.30
66	-3.03	9.18
67	1.60	2.55
68	14.92	222.56
69	-2.75	7.59
70	16.32	266.21
71	7.05	49.75
72	4.02	16.14
73	8.14	66.31
74	-5.00	25.01
75	-10.72	114.94
76	-10.59	112.07
77	-3.32	11.00
78	-16.32	266.47
79	6.41	41.14
80	11.68	136.45
81	20.68	427.86
82	14.71	216.53
83	2.04	4.17
84	-5.42	29.42
85	6.36	40.42
86	-11.29	127.54
87	-3.34	11.13
88	11.08	122.69
89	-1.54	2.39

```

90  -16.07      258.22
91  -13.49      181.96
92   3.63       13.17
93   1.08        1.16
94   2.21        4.89
95  -0.49        0.24
96 -16.82      282.83
97  -2.31        5.33
98   0.15        0.02
99   3.16       10.02
100  2.96        8.79

> variance <- sum(squared.errors)/(length(y) - 1)
> variance

[1] 85.58

> var(y)

[1] 85.58

```

To revert back to the original units of measurement, we take the square root of the variance / MSE. The result is called Root Mean Squared Error (RMSE), or the **standard deviation** of a vector of observations.

```

> standard.deviation <- sqrt(variance)
> standard.deviation

[1] 9.251

> sd(y)

[1] 9.251

```

The mean is a **least squares model** because it minimizes the error as measured by variance / standard deviation.

```

> mean(y)

[1] 164.8

> sd(y)

[1] 9.251

> (alternative.middle.points <- seq(mean(y) - 10, mean(y) + 10, length.out = 49))

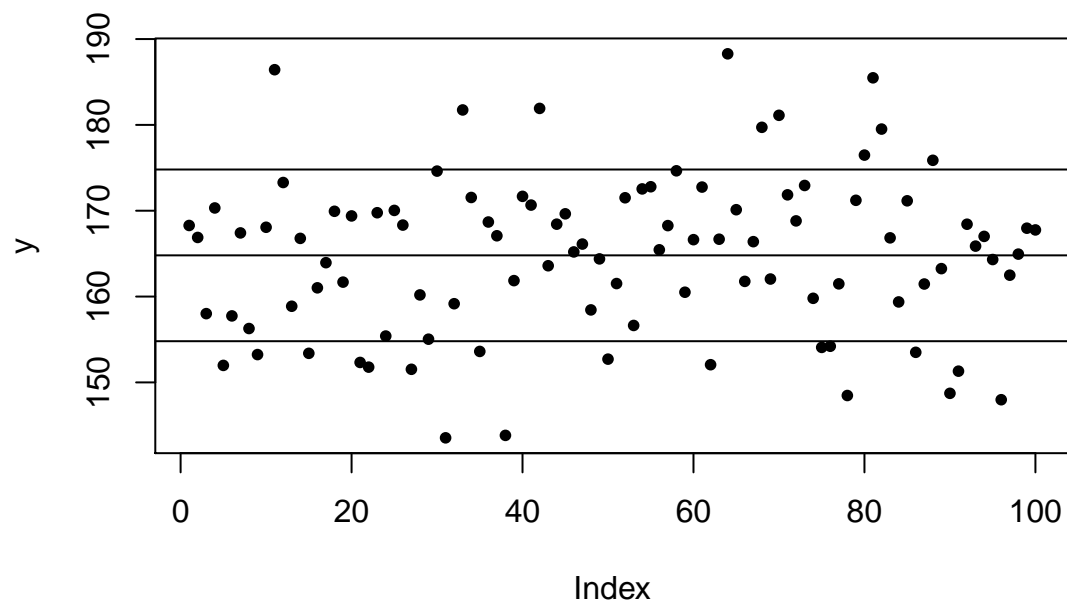
[1] 154.8 155.2 155.6 156.1 156.5 156.9 157.3 157.7 158.1 158.6 159.0
[12] 159.4 159.8 160.2 160.6 161.1 161.5 161.9 162.3 162.7 163.1 163.6
[23] 164.0 164.4 164.8 165.2 165.6 166.1 166.5 166.9 167.3 167.7 168.1
[34] 168.6 169.0 169.4 169.8 170.2 170.6 171.1 171.5 171.9 172.3 172.7
[45] 173.1 173.6 174.0 174.4 174.8

> alternative.middle.points[25] == mean(y)

[1] TRUE

> plot(y, pch = 20)
> abline(mean(y), 0)
> abline(alternative.middle.points[1], 0)
> abline(alternative.middle.points[49], 0)

```



```
> (errors.for.alternatives <- numeric(length = 49))

[1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[36] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

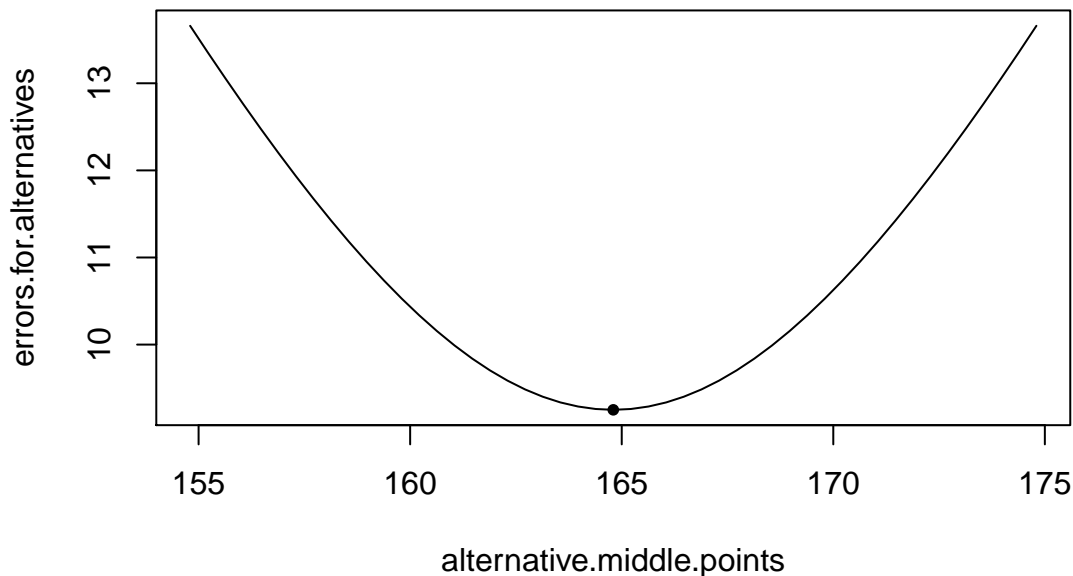
> for (i in 1:49) {
+   errors.for.alternatives[i] <- sqrt(sum((y - alternative.middle.points[i])^2)/(length(y) -
+     1))
+ }
> errors.for.alternatives

[1] 13.660 13.355 13.056 12.764 12.479 12.202 11.933 11.673 11.423 11.182
[11] 10.952 10.734 10.528 10.335 10.155 9.989 9.839 9.704 9.586 9.485
[21] 9.402 9.336 9.289 9.261 9.251 9.261 9.289 9.336 9.402 9.485
[31] 9.586 9.704 9.839 9.989 10.155 10.335 10.528 10.734 10.952 11.182
[41] 11.423 11.673 11.933 12.202 12.479 12.764 13.056 13.355 13.660

> errors.for.alternatives[25] == sd(y)

[1] TRUE

> plot(alternative.middle.points, errors.for.alternatives, type = "l")
> points(mean(y), sd(y), pch = 20)
```



Why square the errors, average them and take the square root of the result instead of simply taking the absolute values of the errors, average them, and try to minimize that measure of error (for example)? The absolute value error is also known as Laplace error because it follows a Laplace distribution, see for example here http://en.wikipedia.org/wiki/Least_absolute_deviations – just as squared errors are also known as Gaussian / normal errors.

It turns out that the **median** minimizes the absolute-value error; see, for example, chapter 2 of Gonick and Smith (1993) if you're not familiar with the notion of median – go read about it now. The study guide for Gonick and Smith (1993) available on eCommons also discusses the **mode** as yet a third measure of central tendency.

We use least-squares, i.e., normal / Gaussian errors, because we often assume that the error / noise is normally distributed. We'll soon talk about the Central Limit Theorem as one very important result showing that this assumption about the distribution of the errors is true in the limit when the errors are the result of summing over many small and unrelated random effects (see, for example, the 'Fuzzy Central Limit Theorem' in chapter 5 of Gonick and Smith 1993). But this assumption is not always supported by the data, in which case it is better to use alternative ways of quantifying model error, e.g., absolute-value errors or other methods (see, for example, Wang and Bovik 2009).

Another reason is that the closed-form math and also the numerical computations needed to minimize squared errors are nicer / easier than for absolute value errors.

1.4 From sample to population: basic statistical (inductive) inference

While working through this subsection and the remainder of the present set of lecture notes, it is highly recommended that you also read chapters 5, 6, and 7 of Gonick and Smith (1993) and the corresponding parts of the study guide on eCommons.

The mean is the simplest least squares model for our sample of 100 observations / people. How well does it generalize to the entire population?

That is, how well is the **mean of the entire population** approximated by our **sample mean**?

To measure the sampling error, i.e., the fact that our random sample is only a partial, imperfect image of the entire population, we look at:

- i. the size of the sample – the bigger the better
- ii. the standard deviation of our sample – the smaller the better

And the Central Limit Theorem says that looking at these 2 things is actually all we need to do.

Note that this way of quantifying sampling error is correct (in the limit, and approximately correct for a reasonably large sample) **only if** our sample is random. Ensuring random sampling (or something as close to it as practically possible) is a very important component of experimental design.

For our simulated data set, we made sure we had a random sample from the population of US adults by:

- randomly choosing 100 different heights – distributed according to a probability distribution that is close to the empirical distribution of height for US adults; i.e., implicitly, we took a uniform random sample of people from the population of US adults and measured their heights, which are distributed according to a normal distribution
- randomly choosing the gender of the 100 people whose height we ‘measured’

Note that conceptually, the above two bullet points are true, but we actually implemented things a bit differently: x_1 is a random sample from the distribution of heights for US adult females; we then added a ‘deflection’ of +6 pounds for all the people / observations (randomly) identified by x_2 as being male. An exercise in hw assignment 3 will ask you to simulate the same data in a different manner so that you understand exactly what the implementation does.

Thus, the random draws we did for x_1 (height) and x_2 (gender) when we simulated the data are conceptually very different from the random draws we did when we generated the responses y (the weights of the ‘people’ in our sample): those draws added Gaussian error / noise to the mean weights, i.e., they added the inherent variation observable in weights (after factoring out height and gender).

Random sampling is under our experimental control, and we use it to choose a sample in such a way that the sampling error for the mean is really only what the Central Limit Theorem assumes it is. This enables us to assume that the sampling error for the mean (the imperfect image of the population mean as reflected in our sample mean) is just a function of (i) the sample size, and (ii) the inherent variation in our response variable.

Random error / noise is this inherent variation in the response variable, and we use the data we extract from our random sample (in particular, the standard deviation of the sample responses y) to estimate it.

1.4.1 Sample size

The bigger the sample, the better it will reflect the structure of the whole population.

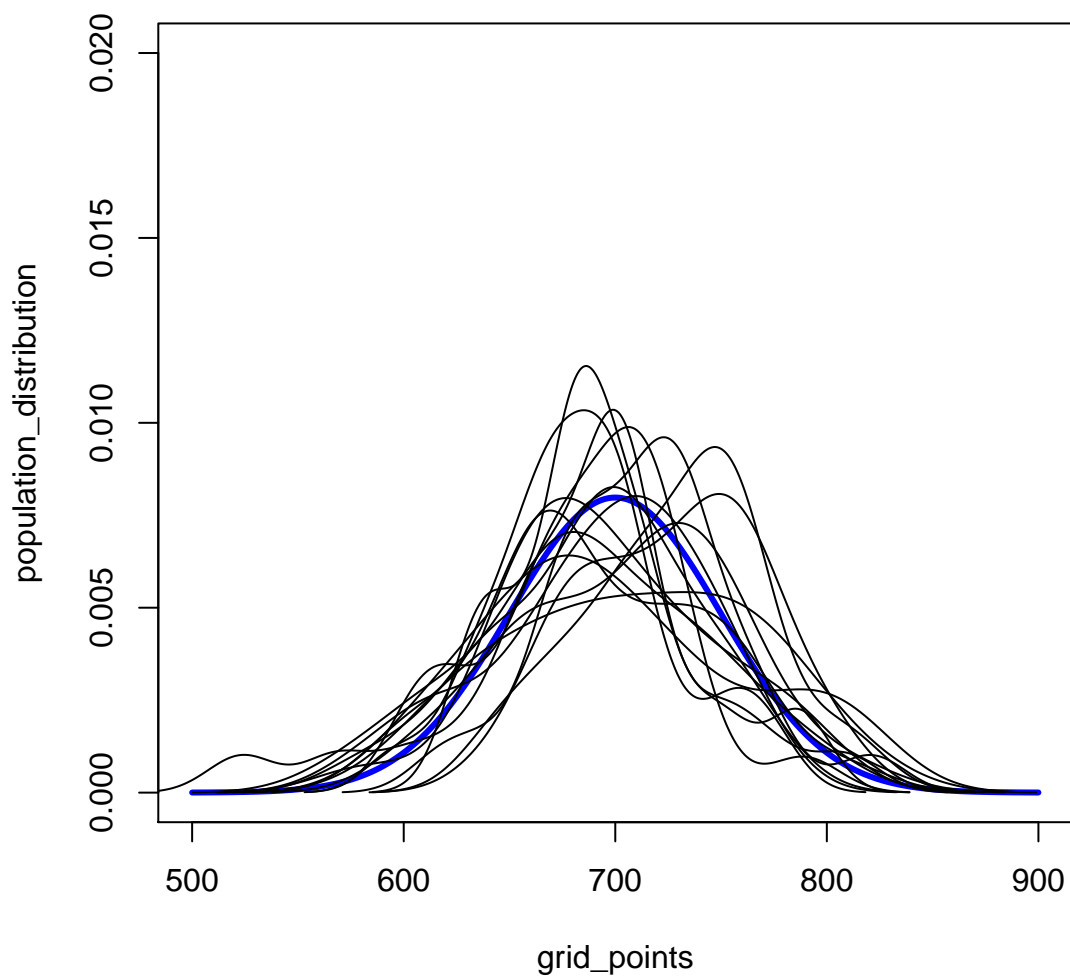
We’ll use a different example than our running example of height, gender and weight, to illustrate the effects of sample size and sample standard deviation.

Assume that the mean time to recognize that a particular sequence of characters (henceforth, the stimulus) is an English word in a lexical decision task is 700 ms. Furthermore, the decision times are normally distributed around this mean, and the normal distribution has a standard deviation of 50 ms.

```
> grid_points <- seq(500, 900, length.out = 2000)
> population_distribution <- dnorm(grid_points, 700, 50)
```

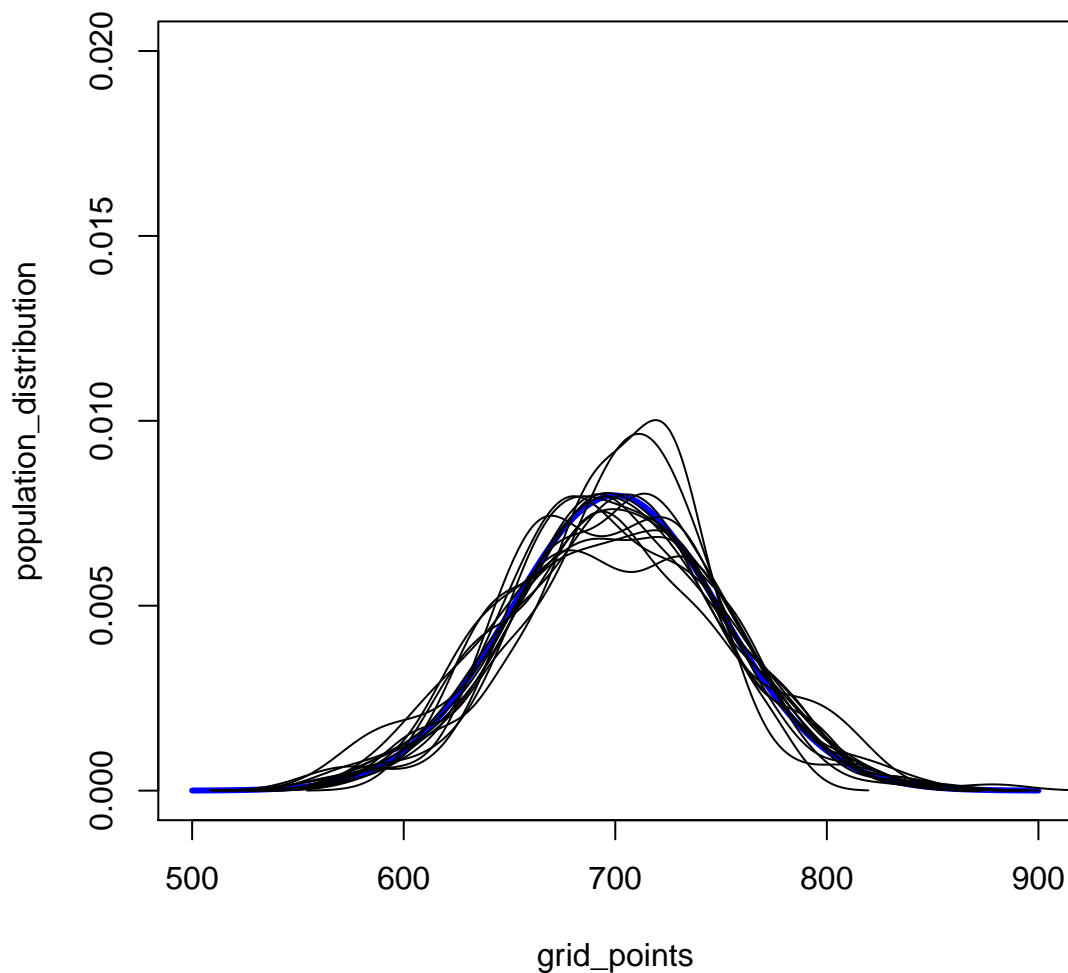
Let’s see the distribution of 15 samples from this population / theoretical distribution, each consisting 25 decisions times:

```
> plot(grid_points, population_distribution, type = "l", lwd = 3, col = "blue",
+      xlim = c(500, 900), ylim = c(0, 0.02))
> for (i in 1:15) {
+   z <- rnorm(25, 700, 50)
+   lines(density(z), col = "black")
+ }
```



Now let's look at 15 samples of 150 observations each:

```
> plot(grid_points, population_distribution, type = "l", lwd = 3, col = "blue",  
+       xlim = c(500, 900), ylim = c(0, 0.02))  
> for (i in 1:15) {  
+   z <- rnorm(150, 700, 50)  
+   lines(density(z), col = "black")  
+ }
```



1.4.2 Sample variance

A smaller variance in the sample indicates that the variance in the population is itself smaller, so we have a higher chance of sampling near the mean.

Suppose the standard deviation of our decision-time distribution is 25, not 50:

```
> grid_points <- seq(500, 900, length.out = 2000)
> population_distribution <- dnorm(grid_points, 700, 25)
```

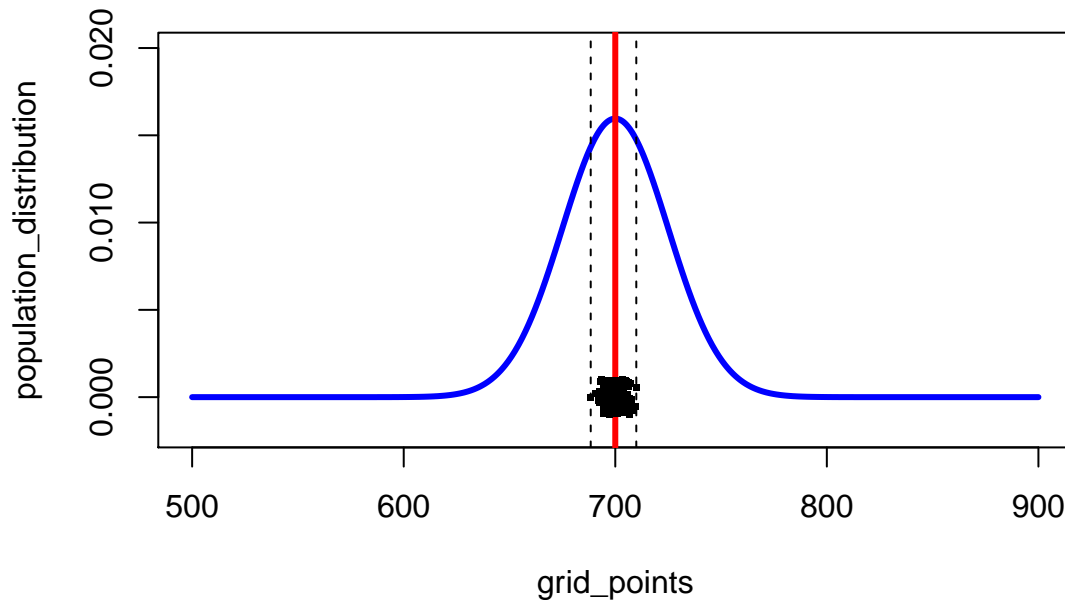
Now let's look at the sample means for 200 samples of 50 observations each from this distribution:

```
> plot(grid_points, population_distribution, type = "l", lwd = 3, col = "blue",
+       xlim = c(500, 900), ylim = c(-0.002, 0.02))
> abline(v = 700, lwd = 3, col = "red")
> sample_means <- numeric(length = 200)
> for (i in 1:200) {
+   z <- rnorm(50, 700, 25)
```

```

+   sample.means[i] <- mean(z)
+ }
> points(sample.means, jitter(rep(0, 200), amount = 0.001), pch = ".",
+   cex = 3.2)
> abline(v = min(sample.means), lty = 2)
> abline(v = max(sample.means), lty = 2)

```

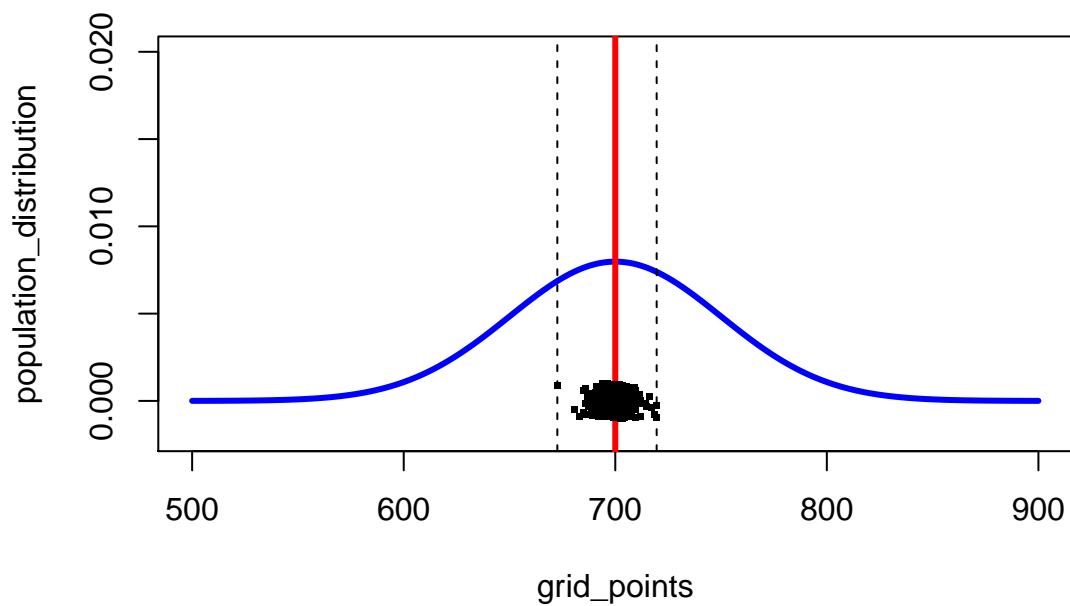


Now assume that the standard deviation of our population is 50:

```

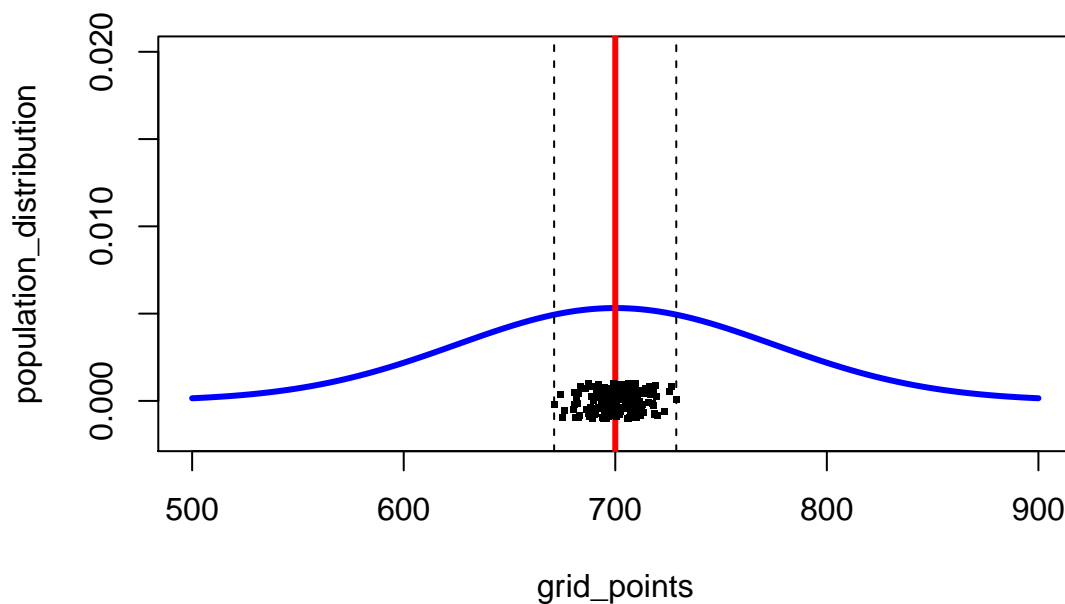
> grid_points <- seq(500, 900, length.out = 2000)
> population_distribution <- dnorm(grid_points, 700, 50)
> plot(grid_points, population_distribution, type = "l", lwd = 3, col = "blue",
+   xlim = c(500, 900), ylim = c(-0.002, 0.02))
> abline(v = 700, lwd = 3, col = "red")
> sample.means <- numeric(length = 200)
> for (i in 1:200) {
+   z <- rnorm(50, 700, 50)
+   sample.means[i] <- mean(z)
+ }
> points(sample.means, jitter(rep(0, 200), amount = 0.001), pch = ".",
+   cex = 3.2)
> abline(v = min(sample.means), lty = 2)
> abline(v = max(sample.means), lty = 2)

```

Finally, look what happens if the standard deviation of the population is 75:

```
> grid_points <- seq(500, 900, length.out = 2000)
> population_distribution <- dnorm(grid_points, 700, 75)
> plot(grid_points, population_distribution, type = "l", lwd = 3, col = "blue",
+       xlim = c(500, 900), ylim = c(-0.002, 0.02))
> abline(v = 700, lwd = 3, col = "red")
> sample.means <- numeric(length = 200)
> for (i in 1:200) {
+   z <- rnorm(50, 700, 75)
+   sample.means[i] <- mean(z)
+ }
> points(sample.means, jitter(rep(0, 200), amount = 0.001), pch = ".",
+       cex = 3.2)
> abline(v = min(sample.means), lty = 2)
> abline(v = max(sample.means), lty = 2)
```



1.4.3 Putting the two together: the standard error (SE) of the mean, the Central Limit Theorem (CLT), and 95% confidence intervals (CIs)

- (2) Standard error (SE) of the mean = $\frac{\text{standard deviation of the sample}}{\sqrt{\text{sample size}}}$

```
> mean(y)
[1] 164.8

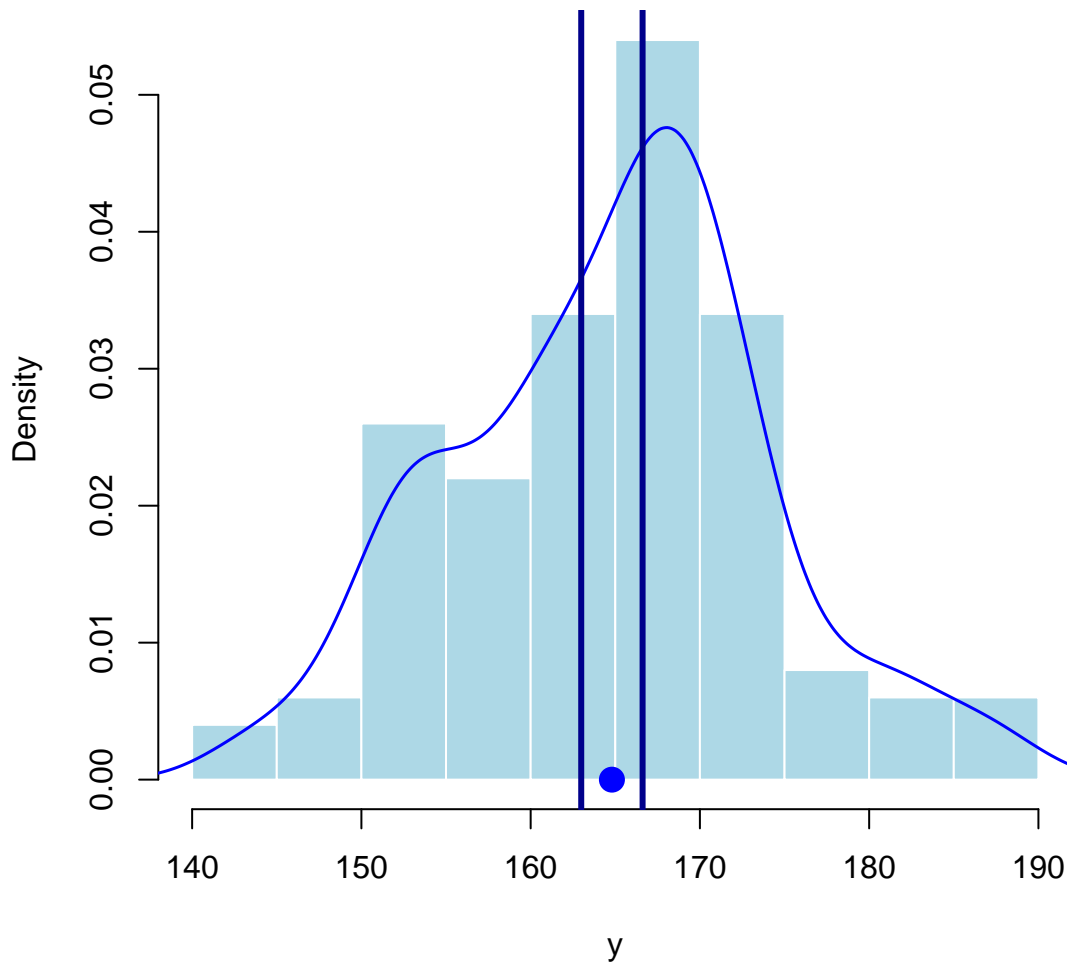
> (se_of_mean <- sd(y)/sqrt(length(y)))
[1] 0.9251
```

Now that we have the SE of the sample mean, and the sample mean itself, we can construct an (approximate) interval in which we can be fairly confident the population mean can be found (basically). The usual (minimum) level of confidence is 95%. This is how we define the 95% confidence interval (CI):

- (3) The 95% confidence interval (CI) for the population mean is approximately:
(sample mean $- 1.96 \cdot \text{SE of the mean}$, sample mean $+ 1.96 \cdot \text{SE of the mean}$)

```
> hist(y, freq = F, col = "lightblue", border = "white", main = "Histogram of y, mean of y, and 95% CI")
> lines(density(y), col = "blue", lwd = 1.5)
> points(mean(y), 0, pch = 20, col = "blue", cex = 2.5)
> abline(v = mean(y) - (1.96 * se_of_mean), lwd = 3, col = "darkblue")
> abline(v = mean(y) + (1.96 * se_of_mean), lwd = 3, col = "darkblue")
```

Histogram of y, mean of y, and 95% CI for the mean



An incorrect, but intuitive way of putting it: 95% of the time, the population mean falls roughly within 1.96 SEs of the sample mean, where SE is the standard error of the mean computed as shown above. See Gonick and Smith (1993), chapter 7, for the correct (but not really intuitive) way of interpreting the 95% interval; we'll discuss this in class too. The above intuitive paraphrase is actually correct in the Bayesian paradigm.

We can perform this type of plausible / inductive inference from the sample mean to the population mean because of the Central Limit Theorem, which basically says:

- (4) The Central Limit Theorem (CLT): as the sample size approaches ∞ , the distribution of the sample mean (a.k.a. the sampling distribution of the mean) approaches a normal distribution whose mean is the population mean and whose standard deviation is the standard error of the mean.

Given the CLT, we can understand why we use the 'magic number' 1.96 in our computation of the 95% CI: the CLT tells us that the distribution of the sample mean is normal, and 95% of the probability mass (i.e., area) under the standard normal curve is between -1.96 and 1.96 .

```
> qnorm(0.025)
```

```
[1] -1.96
```

```

> qnorm(0.975)
[1] 1.96

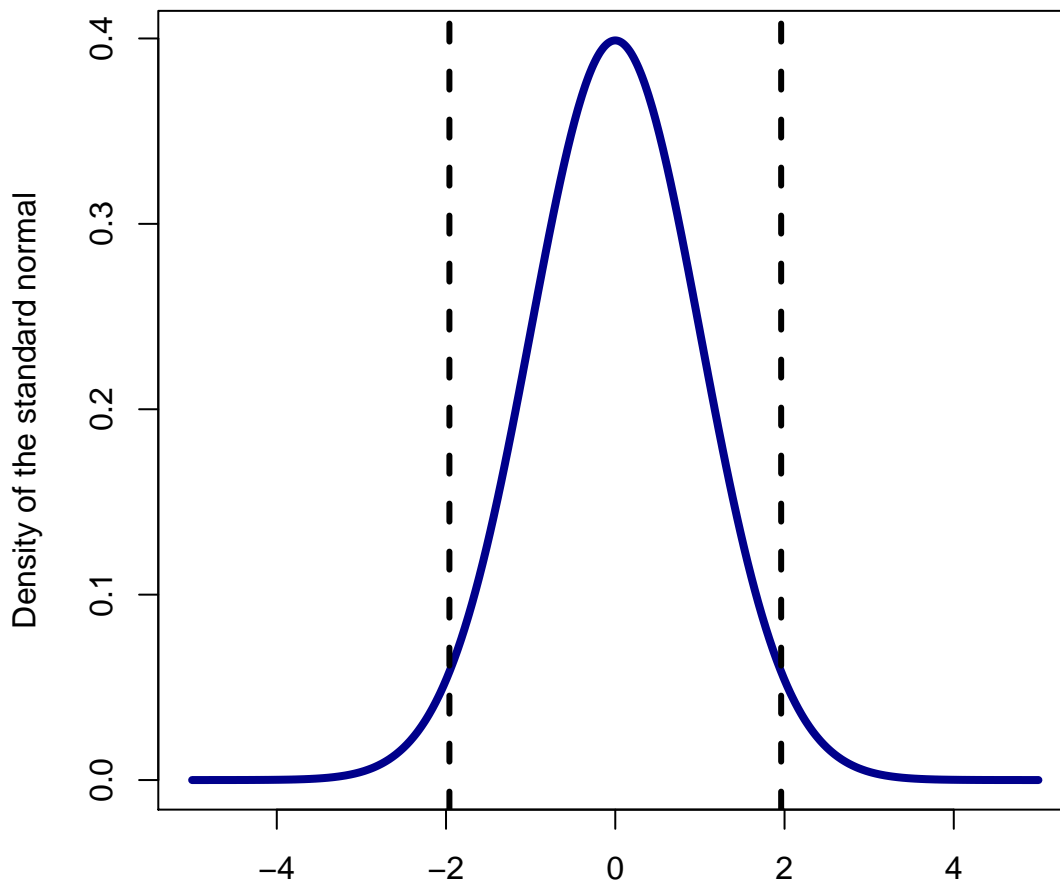
> integrate(dnorm, -1.96, 1.96)

0.95 with absolute error < 1e-11

> grid_points <- seq(-5, 5, length.out = 1000)
> plot(grid_points, dnorm(grid_points), type = "l", lwd = 4, main = "Central 95% interval for the standard normal",
+       xlab = "", ylab = "Density of the standard normal", col = "darkblue")
> abline(v = qnorm(0.025), lty = 2, lwd = 3)
> abline(v = qnorm(0.975), lty = 2, lwd = 3)

```

Central 95% interval for the standard normal



We can ask R to do all these computations in one go by using the `lm` (linear model) function as follows:

```

> the_mean_model <- lm(y ~ 1)
> summary(the_mean_model)

```

```

Call:
lm(formula = y ~ 1)

Residuals:
    Min       1Q   Median       3Q      Max
-21.26  -6.03   1.46   5.37  23.48

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  164.800      0.925    178   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.25 on 99 degrees of freedom

```

Note that the intercept and the SE for the intercept – the intercept is symbolized by 1 in the `lm` formula $y \sim 1$ – are very close to the mean and the SE for the mean we calculated above. ‘Intercept’ is just the linear-model term for the mean of all the responses (in general, it will not be the mean of *all* the responses, but it will always be a mean).

```

> print(summary(the_mean_model)$coef[, 1:2], dig = 3)

      Estimate Std. Error
      164.800      0.925

> print(mean(y), dig = 5)

[1] 164.8

> print(se_of_mean, dig = 3)

[1] 0.925

```

Note also how the residual standard error is the standard deviation of the vector of errors relative to the mean, i.e., of $y - \text{mean}_y$:

```

> print(summary(the_mean_model)$sigma, dig = 3)

[1] 9.25

> print(sd(y - mean(y)), dig = 3)

[1] 9.25

```

We will discuss, and work on understanding, the output of the `lm` function in detail over the next couple of lectures. For now, note that the p value for the intercept is so small, it is actually 0 given R’s level of machine precision:

```

> .Machine$double.eps

[1] 2.22e-16

> .Machine$double.neg.eps

[1] 1.11e-16

```

If you look at the help file `?Machine`, you'll see:

- `double.eps`: the smallest positive floating-point number x such that $1 + x \neq 1$
- `double.neg.eps`: a small positive floating-point number x such that $1 - x \neq 1$

Summary:

- the deterministic part of our model, i.e., the underlying pattern present in the population and imperfectly reflected in the sample: the mean of the responses y
- the error / non-deterministic part of our model: a normal distribution centered at the mean of y , whose standard deviation is (approximated by) the standard deviation of the sample of responses y
- the 95% CI that we can obtain based on the CLT tells us how the deterministic part of our model should be 'scaled up' / generalized from the sample to the entire population

1.5 Understanding the `lm` output a little bit better: t-distributions

Let's take one more step towards understanding the `lm` output: why do we have a t-value for the intercept / mean?

```
> summary(the_mean_model)

Call:
lm(formula = y ~ 1)

Residuals:
    Min       1Q   Median       3Q      Max
-21.26  -6.03   1.46   5.37  23.48

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   164.800      0.925     178   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

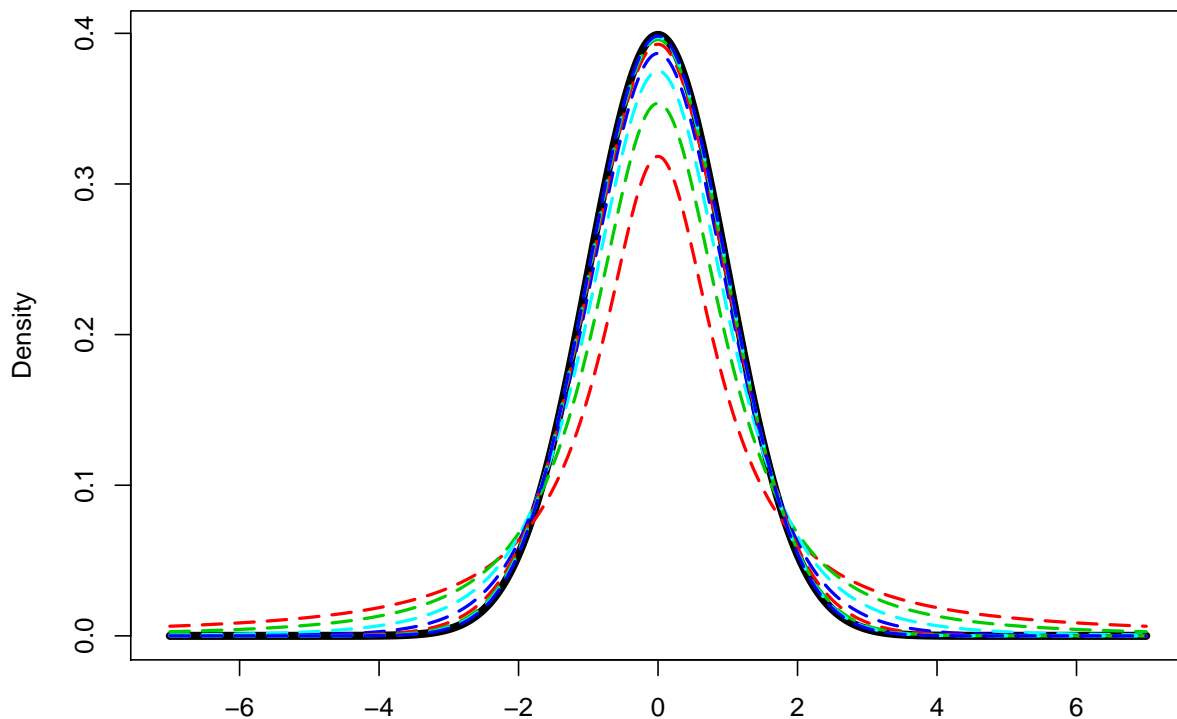
Residual standard error: 9.25 on 99 degrees of freedom
```

The t-value has this name because it is used in connection to a family of probability distributions called t-distributions. We won't compute the t-value and associated p-value in this set of lecture notes, but we will introduce t-distributions here as a prelude to that, and also because they help us compute better CIs for the population mean.

T-distributions are like standard normal distributions, except they have heavier tails, i.e., they're lower and wider. They have one degree-of-freedom (dof) parameter, and the smaller the dof parameter, the thicker the tails are. As the dof parameter approaches ∞ , the tails become thinner and thinner and the t-distribution approaches the standard normal distribution.

```
> grid_points <- seq(-7, 7, length.out = 1000)
> plot(grid_points, dnorm(grid_points), type = "l", lwd = 5, xlab = "",
+       ylab = "Density", main = "T-distributions (dashed) and the normal distribution")
> for (i in c(1, 2, 4, 8, 16, 32, 64, 128)) {
+   lines(grid_points, dt(grid_points, df = i), col = i%5 + 1, lty = 5,
+         lwd = 2)
+ }
```

T-distributions (dashed) and the normal distribution



We use t-distributions to get more accurate CIs for population means when samples are small and we do not know the standard deviation of the population – which we usually don't if we didn't simulate the data ourselves. Note: you should probably also read the discussion of t-distributions in chapters 6 and 7 of Gonick and Smith [1993](#)).

```
> qt(c(0.025, 0.975), df = 1)
[1] -12.71 12.71
> qt(c(0.025, 0.975), df = 2)
[1] -4.303 4.303
> qt(c(0.025, 0.975), df = 4)
[1] -2.776 2.776
> qt(c(0.025, 0.975), df = 8)
[1] -2.306 2.306
> qt(c(0.025, 0.975), df = 16)
[1] -2.12 2.12
> qt(c(0.025, 0.975), df = 32)
```

```
[1] -2.037  2.037

> qt(c(0.025, 0.975), df = 64)

[1] -1.998  1.998

> qt(c(0.025, 0.975), df = 128)

[1] -1.979  1.979

> qnorm(c(0.025, 0.975))

[1] -1.96  1.96
```

The number of dof.s we use for the t-distribution is the number of observations minus the number of parameters we estimate. So, for our 1-mean model, we have 99 dof.s because we have 100 observations in the vector y and we're estimating 1 mean.

Therefore, the numbers we should use as a 'multipliers' for the SE of the mean when we compute the 95% CI are not the ones that give us the 95% interval for standard normal, but the slightly larger ones that give us the 95% interval for a t-distribution with 99 dof.s.

```
> qnorm(c(0.025, 0.975))

[1] -1.96  1.96

> qt(c(0.025, 0.975), df = 99)

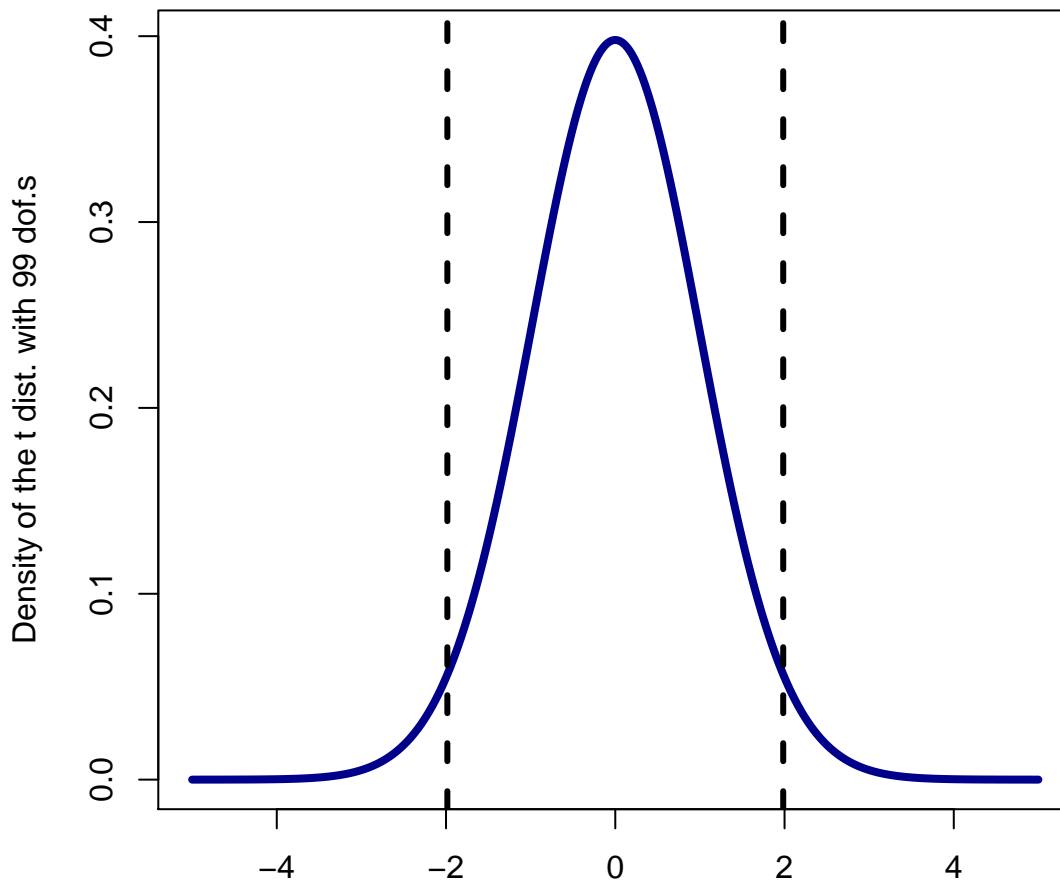
[1] -1.984  1.984

> t_99_density <- function(x) {
+   dt(x, df = 99)
+ }
> integrate(t_99_density, -1.984, 1.984)

0.95 with absolute error < 4.7e-11

> grid_points <- seq(-5, 5, length.out = 1000)
> plot(grid_points, dt(grid_points, df = 99), type = "l", lwd = 4, main = "Central 95% interval for the
+   xlab = "", ylab = "Density of the t dist. with 99 dof.s", col = "darkblue")
> abline(v = qt(0.025, df = 99), lty = 2, lwd = 3)
> abline(v = qt(0.975, df = 99), lty = 2, lwd = 3)
```


Central 95% interval for the t dist. with 99 dof.s



Because the number of dof.s (99) is pretty big, the ‘multipliers’ will be very close, and our CI will not increase that much (note that a larger CI indicates bigger uncertainty about the population estimate).

So let’s compare:

- the 95% CI computed according to the standard normal ‘multipliers’ – this is the CI we computed above – and
- the more accurate 95% CI that is computed according to the larger, t-distribution based ‘multipliers’ – these larger multipliers better reflect our uncertainty / ignorance about the variance of the population

```
> mean(y)
[1] 164.8
> se_of_mean
[1] 0.9251
> c(mean(y) + qnorm(0.025) * se_of_mean, mean(y) + qnorm(0.975) * se_of_mean)
```

```
[1] 163.0 166.6

> c(mean(y) + qt(0.025, df = 99) * se_of_mean, mean(y) + qt(0.975, df = 99) *
+     se_of_mean)

[1] 163.0 166.6
```

They are really close in this case because we have many observations (100), hence many dof.s for our t-distribution. But they are more clearly different for small samples, for example, for a sample of 15 observations from – let’s say – a normal probability distribution of reading times centered at 1800 ms and with a standard deviation of 250 ms:

```
> y2 <- rnorm(15, 1800, 250)
> summary(y2)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1260   1760   1850   1830   2060   2300

> (se_of_mean_y2 <- sd(y2)/sqrt(length(y2)))

[1] 81.42

> c(mean(y2) + qnorm(0.025) * se_of_mean_y2, mean(y2) + qnorm(0.975) *
+     se_of_mean_y2)

[1] 1667 1986

> c(mean(y2) + qt(0.025, df = 14) * se_of_mean_y2, mean(y2) + qt(0.975,
+     df = 14) * se_of_mean_y2)

[1] 1652 2001
```

The t-value and p-value reported by the `lm` function are computed based on t-distributions, we’ll see how in the next set of lecture notes.

2 Applying the Central Limit Theorem (CLT) to Bernoulli distributed data

This section provides another example of this type of inference from the sample mean to the population mean, the basis of which is the Central Limit Theorem.

We will consider the Bernoulli distribution in this section and we will see how the standard normal distribution plays a central role in making inferences about the mean of a Bernoulli-distributed population (the inherent bias of a coin across the entire population of coin flips) based on the mean of a sample (a particular finite set of coins flips).

This section is partly based on Lesigne (2005).

If you don’t follow all the details in this section, don’t worry. You can think of this section as an excuse to introduce the Bernoulli and binomial distributions and to think about the normal distribution some more.

2.1 The Bernoulli distribution

We flip a coin. For (mathematical) convenience, we designate H (eads) as 1 and T (ails) as 0. I.e., we flip a coin and it comes up H , and instead of saying that the value of the observation y is H , i.e., $y = H$, we say that $y = 1$. Similarly, instead of $y = T$, we say $y = 0$.

If the coin has a bias θ for heads, then the probability of heads, i.e., of a single observation $y = 1$, is θ . We express this more concisely as a formula:

(5) $p(y = 1|\theta) = \theta$, or even shorter: $p(1|\theta) = \theta$

Assuming the same bias θ for heads, the probability of tails, i.e., of a single observation $y = 0$, is $1 - \theta$. We express this as follows:

(6) $p(y = 0|\theta) = 1 - \theta$, or even shorter: $p(0|\theta) = 1 - \theta$

Because of our convention $H = 1, T = 0$, we can very succinctly express the entire Bernoulli probability distribution (i.e., the probability of both heads and tails) as follows:

(7) The Bernoulli probability distribution (a.k.a. probability mass function, or pmf for short):

$$p(y|\theta) = \theta^y \cdot (1 - \theta)^{1-y}$$

a. if $y = 1$, we have $p(1|\theta) = \theta^1 \cdot (1 - \theta)^{1-1} = \theta$

b. if $y = 0$, we have $p(0|\theta) = \theta^0 \cdot (1 - \theta)^{1-0} = 1 - \theta$

If we have a fixed θ , e.g., $\theta = 0.45$, we can plot the Bernoulli probability distribution as follows:

```
> theta <- 0.45
> Bern <- function(y, theta) {
+   theta^y * (1 - theta)^(1 - y)
+ }
> outcomes <- c(1, 0)
> outcomeProbs <- Bern(outcomes, theta)
> outcomeProbs

[1] 0.45 0.55

> outcomes

[1] 1 0

> as.matrix(outcomes)

      [,1]
[1,]    1
[2,]    0

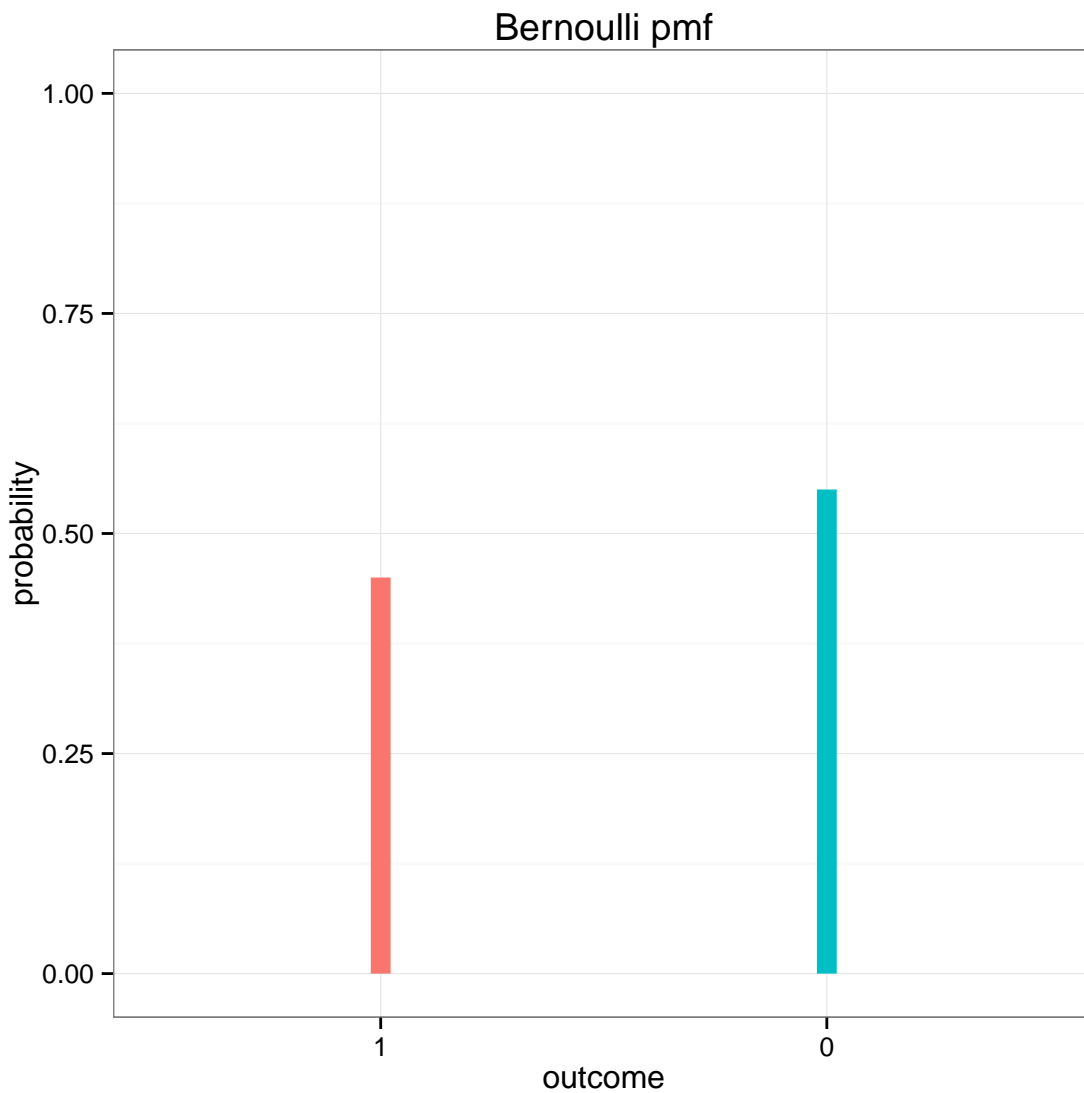
> outcomeProbs <- apply(as.matrix(outcomes), 1, Bern, theta)
> outcomeProbs

[1] 0.45 0.55

> library("ggplot2")
> outcomesToPlot <- factor(as.character(outcomes), levels = as.character(outcomes))
> outcomesToPlot

[1] 1 0
Levels: 1 0

> qplot(outcomesToPlot, ymin = 0, ymax = outcomeProbs, type = "h", col = outcomesToPlot,
+   lwd = 2.5, ylim = range(0, 1), xlab = "outcome", ylab = "probability",
+   main = "Bernoulli pmf", geom = "linerange") + theme_bw() + theme(legend.position = "none")
```



Note that if we sum the probabilities of all outcomes, we get 1:

```
> outcomeProbs
[1] 0.45 0.55
> sum(outcomeProbs)
[1] 1
```

Suppose now the data set consists of multiple coin flips, e.g., 3 coin flips. We can easily generalize the Bern. prob. dist. function to data sets consisting of multiple observations if we assume they are independent: since they are independent, the joint probability of the 3 flips is the product of the probabilities of the individual flips:

```
> theta <- 0.45
> Bern <- function(y, theta) {
+   prod(theta^y * (1 - theta)^(1 - y))
+ }
```

```

> y <- 1
> Bern(y, theta)

[1] 0.45

> y <- 0
> Bern(y, theta)

[1] 0.55

> y <- c(1, 1)
> Bern(y, theta)

[1] 0.2025

> theta^2

[1] 0.2025

> y <- c(1, 1, 0)
> Bern(y, theta)

[1] 0.1114

> theta^2 * (1 - theta)

[1] 0.1114

```

The probability space for 3-coin-flip samples consists of 8 outcomes (2^3 – why?), shown below:

```

> outcomes <- matrix(c(c(1, 1, 1), c(1, 1, 0), c(1, 0, 1), c(0, 1, 1),
+   c(1, 0, 0), c(0, 1, 0), c(0, 0, 1), c(0, 0, 0)), byrow = TRUE,
+   nrow = 8)
> outcomes

      [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    0
[3,]    1    0    1
[4,]    0    1    1
[5,]    1    0    0
[6,]    0    1    0
[7,]    0    0    1
[8,]    0    0    0

> outcomeProbs <- apply(outcomes, 1, Bern, theta)
> outcomeProbs

[1] 0.09113 0.11138 0.11138 0.11138 0.13613 0.13613 0.13613 0.16638

> sum(outcomeProbs)

[1] 1

> library("ggplot2")
> almostOutcomesToPlot <- vector(length = nrow(outcomes))
> for (i in 1:nrow(outcomes)) {

```

```

+   almostOutcomesToPlot[i] <- paste(as.character(outcomes[i, ]),
+   sep = "", collapse = "")
+ }
> almostOutcomesToPlot

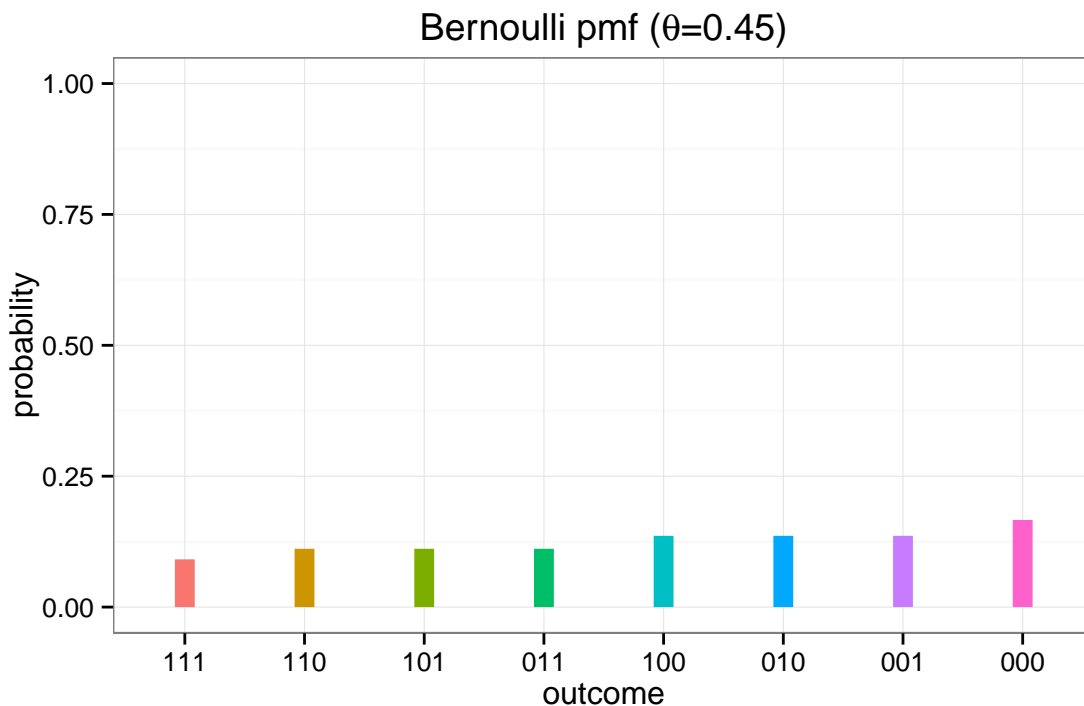
[1] "111" "110" "101" "011" "100" "010" "001" "000"

> outcomesToPlot <- factor(almostOutcomesToPlot, levels = almostOutcomesToPlot)
> outcomesToPlot

[1] 111 110 101 011 100 010 001 000
Levels: 111 110 101 011 100 010 001 000

> qplot(outcomesToPlot, ymin = 0, ymax = outcomeProbs, type = "h", col = outcomesToPlot,
+   lwd = 2.5, ylim = range(0, 1), xlab = "outcome", ylab = "probability",
+   main = expression(paste("Bernoulli pmf (", theta, "=0.45)", sep = "")),
+   geom = "linerange") + theme_bw() + theme(legend.position = "none")

```



2.2 The binomial distribution

Let's focus on the number of heads in a sample of n Bernoulli-distributed observations. This is a random variable whose values (possible outcomes) are all the positive integers $\leq n$ and 0.

For example, given the bias $\theta = 0.45$ for our coin and samples of 3 coin flips (as above), the probability distribution for the possible number of heads is as follows:

```

> n_heads <- apply(outcomes, 1, sum)
> data.frame(outcomesToPlot, n_heads, outcomeProbs)

  outcomesToPlot n_heads outcomeProbs

```

```

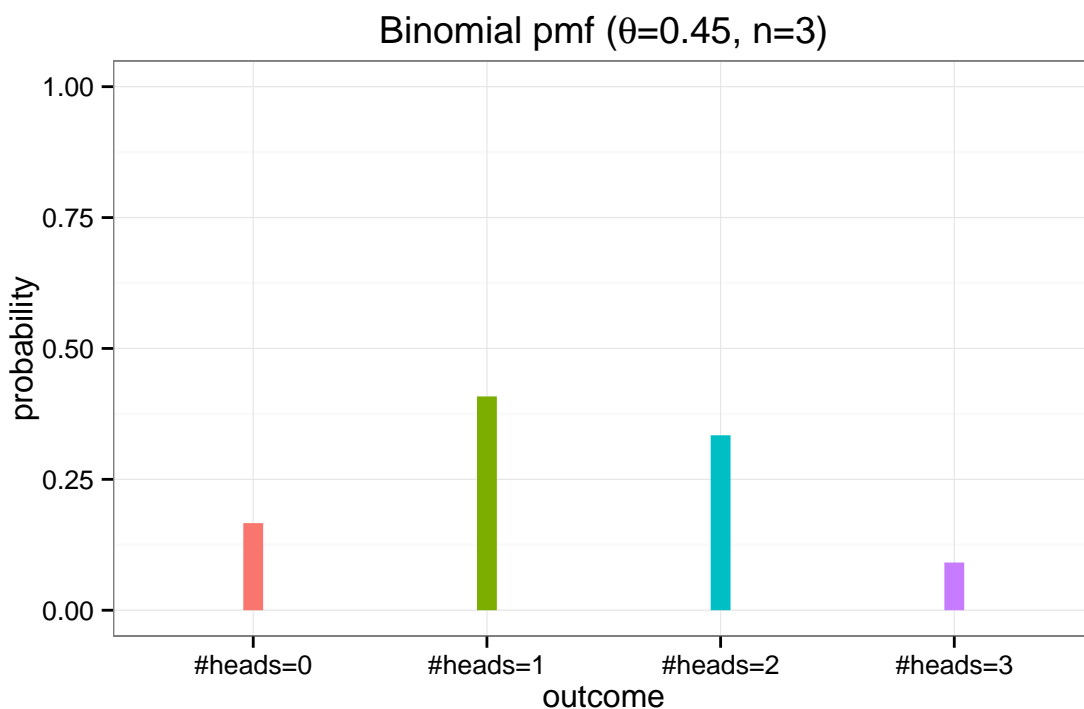
1      111      3      0.09113
2      110      2      0.11138
3      101      2      0.11138
4      011      2      0.11138
5      100      1      0.13613
6      010      1      0.13613
7      001      1      0.13613
8      000      0      0.16638

> n_headsProbs <- by(outcomeProbs, n_heads, sum)
> data.frame(outcomes = rev(names(n_headsProbs)), probs = rev(as.vector(n_headsProbs)))

  outcomes  probs
1        3 0.09113
2        2 0.33413
3        1 0.40838
4        0 0.16638

> outcomesToPlot <- as.factor(paste("#heads=", names(n_headsProbs),
+   sep = ""))
> outcomeProbs <- as.vector(n_headsProbs)
> qplot(outcomesToPlot, ymin = 0, ymax = outcomeProbs, type = "h", col = outcomesToPlot,
+   lwd = 2.5, ylim = range(0, 1), xlab = "outcome", ylab = "probability",
+   main = expression(paste("Binomial pmf (", theta, "=0.45, n=3)",
+   sep = "")), geom = "linerrange") + theme_bw() + theme(legend.position = "none")

```



That is, for each number of heads, we sum the probabilities of all the outcomes (i.e., all Bernoulli-sample configurations) that contain that number of heads:

- how many outcomes in 3-coin-flip samples with 3 heads? 1

- how many outcomes in 3-coin-flip samples with 2 heads? 3
- how many outcomes in 3-coin-flip samples with 1 heads? 3
- how many outcomes in 3-coin-flip samples with 0 heads? 1

In general, for n -coin-flip samples, there are $\binom{n}{k}$ outcomes with k heads:

$$(8) \quad \binom{n}{k} = \frac{n!}{(n-k)!k!}$$

```
> choose(3, 3)
[1] 1
> choose(3, 2)
[1] 3
> choose(3, 1)
[1] 3
> choose(3, 0)
[1] 1
```

So, the general probability distribution function for the number of heads in a sample of n coin flips with bias θ is as follows:

$$(9) \quad \text{The binomial pmf: } p(k|\theta, n) = \binom{n}{k} \theta^k (1 - \theta)^{n-k}.$$

2.3 The binomial distribution and the Central Limit Theorem

We care about the binomial distribution because this distribution gives us the probability of the number of heads k for Bernoulli-distributed samples of size n , which is very closely related to the sampling distribution of the Bernoulli mean $\frac{k}{n}$ we are after. But it's easier to think/talk about sampling distribution of the number of heads k in a sample with known/fixed sample size n than about the sample mean.

The Central Limit Theorem applied to Bernoulli-distributed data basically tells us that the distribution of the number of heads k approaches a normal distribution as n gets larger and larger.¹ This normal distribution is centered at the population mean, which is $n \cdot \theta$ heads out of n flips.

- this is the (theoretical) mean of the binomial distribution because the binomial is the sum of n independent variables (i.e., n coin flips) that are all Bernoulli-distributed with mean θ
- and the mean of the sum is the sum of the means, i.e., $\theta + \theta + \dots + \theta(n \text{ times}) = n\theta$

Let's see graphically how the binomial gets closer and closer to a Gaussian. We first have to define our probability distribution functions.

We start with the binomial pmf:

¹Here's the exact statement; for more discussion and connection to the version outlined in the main text, see Lesigne (2005), p. 30 et seqq.:

- (1) The Central Limit Theorem (for Bernoulli-distributed data with mean θ). Let a and b be two elements in $\mathbb{R} \cup \{+\infty\} \cup \{-\infty\}$ such that $a < b$. Then
- $$p\left(a \leq \frac{S_n - n\theta}{\sqrt{n\theta(1-\theta)}} \leq b\right) \rightarrow \frac{1}{\sqrt{2\pi}} \int_a^b e^{-\frac{x^2}{2}} dx$$
- as $n \rightarrow \infty$, where S_n is the sum of n independent identically-distributed variables with distribution $Bern(\theta)$.


```

> p_binom <- function(k, n, theta) {
+   choose(n, k) * theta^k * (1 - theta)^(n - k)
+ }
> p_binom(3, 3, 0.45)

[1] 0.09113

> p_binom(2, 3, 0.45)

[1] 0.3341

> p_binom(1, 3, 0.45)

[1] 0.4084

> p_binom(0, 3, 0.45)

[1] 0.1664

```

In R, the pmf of the binomial distribution is given by the function `dbinom`:

```

> dbinom(3, 3, 0.45)

[1] 0.09113

> dbinom(2, 3, 0.45)

[1] 0.3341

> dbinom(1, 3, 0.45)

[1] 0.4084

> dbinom(0, 3, 0.45)

[1] 0.1664

```

We now turn to the pdf of the normal distribution of interest. As $n \rightarrow \infty$, the binomial distribution $p(k|\theta, n)$ approaches a normal distribution with mean $n \cdot \theta$ (which is the mean of the binomial distribution) and variance $n \cdot \theta \cdot (1 - \theta)$ (which is the variance of the binomial distribution).

Let's define this Gaussian pdf:

$$(10) \quad \text{In general: } p(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

$$(11) \quad \text{For } \mu = n\theta \text{ and } \sigma^2 = n\theta(1 - \theta), \text{ we have: } p(x|n\theta, n\theta(1 - \theta)) = \frac{1}{\sqrt{2\pi n\theta(1 - \theta)}} e^{-\frac{(x - n\theta)^2}{2n\theta(1 - \theta)}}.$$

```

> p_norm <- function(x, n, theta) {
+   1/sqrt(2 * pi * n * theta * (1 - theta)) * exp(-(x - n * theta)^2 / (2 *
+     n * theta * (1 - theta)))
+ }

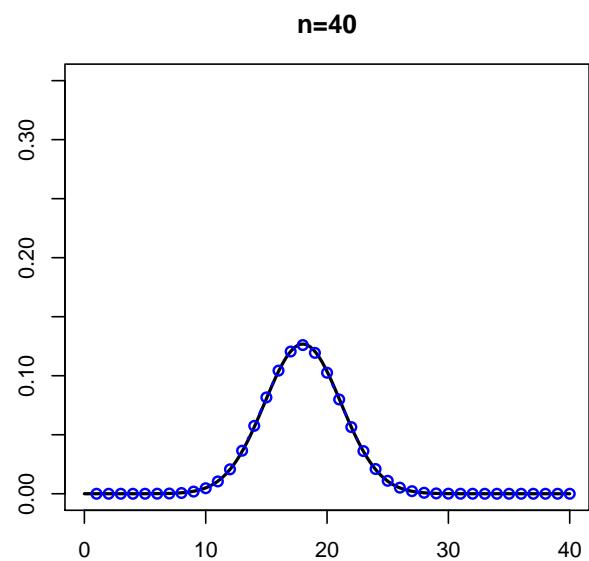
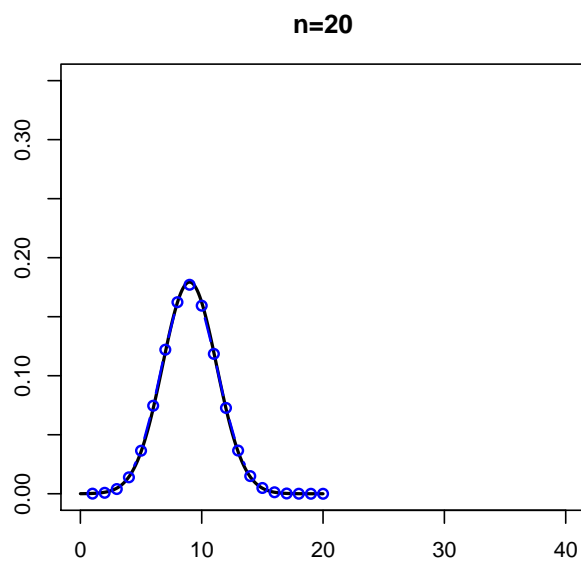
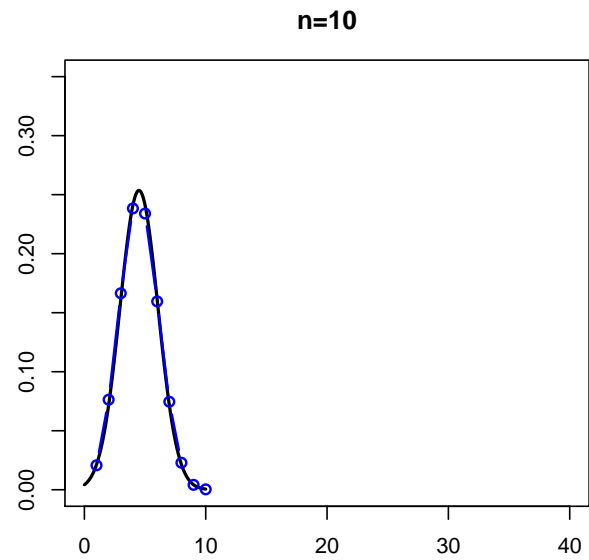
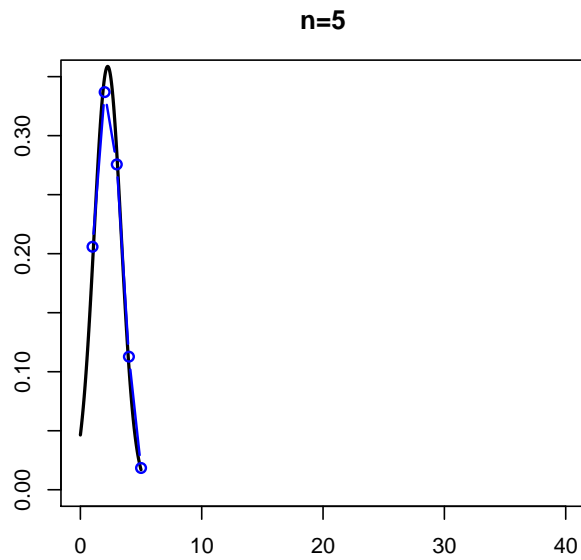
```

Let's now plot both of these functions for different n , assuming our favorite $\theta = 0.45$.

```

> theta <- 0.45
> par(mfrow = c(2, 2), mai = c(0.62, 0.52, 0.52, 0.12))
> n <- 5
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, p_norm(x, n, theta), type = "l", xlab = "", ylab = "", main = paste("n=",
+   n, sep = ""), xlim = c(0, 40), ylim = c(0, 0.35), lwd = 2)
> points(k, p_binom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+   lty = 1)
> n <- 10
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, p_norm(x, n, theta), type = "l", xlab = "", ylab = "", main = paste("n=",
+   n, sep = ""), xlim = c(0, 40), ylim = c(0, 0.35), lwd = 2)
> points(k, p_binom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+   lty = 1)
> n <- 20
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, p_norm(x, n, theta), type = "l", xlab = "", ylab = "", main = paste("n=",
+   n, sep = ""), xlim = c(0, 40), ylim = c(0, 0.35), lwd = 2)
> points(k, p_binom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+   lty = 1)
> n <- 40
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, p_norm(x, n, theta), type = "l", xlab = "", ylab = "", main = paste("n=",
+   n, sep = ""), xlim = c(0, 40), ylim = c(0, 0.35), lwd = 2)
> points(k, p_binom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+   lty = 1)

```



```
> par(mfrow = c(1, 1), mai = c(1.02, 0.82, 0.82, 0.42))
```

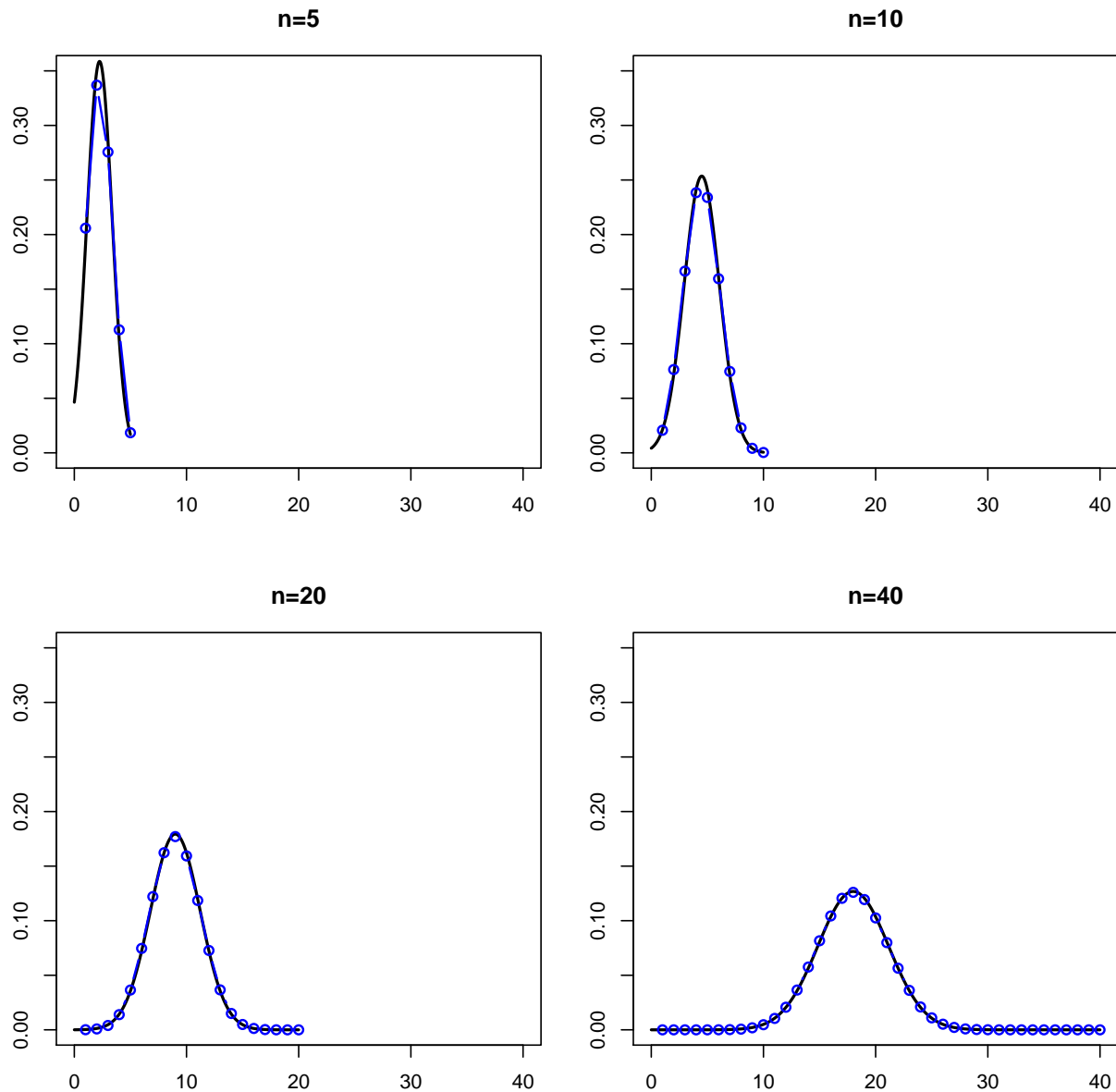
Of course, R has density functions for the binomial and normal distributions that we could use directly:

```
> theta <- 0.45
> par(mfrow = c(2, 2), mai = c(0.62, 0.52, 0.52, 0.12))
> n <- 5
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.35), lwd = 2)
> points(k, dbinom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+      lty = 1)
```

```

> n <- 10
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.35), lwd = 2)
> points(k, dbinom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+      lty = 1)
> n <- 20
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.35), lwd = 2)
> points(k, dbinom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+      lty = 1)
> n <- 40
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.35), lwd = 2)
> points(k, dbinom(k, n, theta), type = "b", col = "blue", lwd = 1.5,
+      lty = 1)

```



```
> par(mfrow = c(1, 1), mai = c(1.02, 0.82, 0.82, 0.42))
```

A simulation-based way to show the Central Limit Theorem in action for Bernoulli-distributed data is to

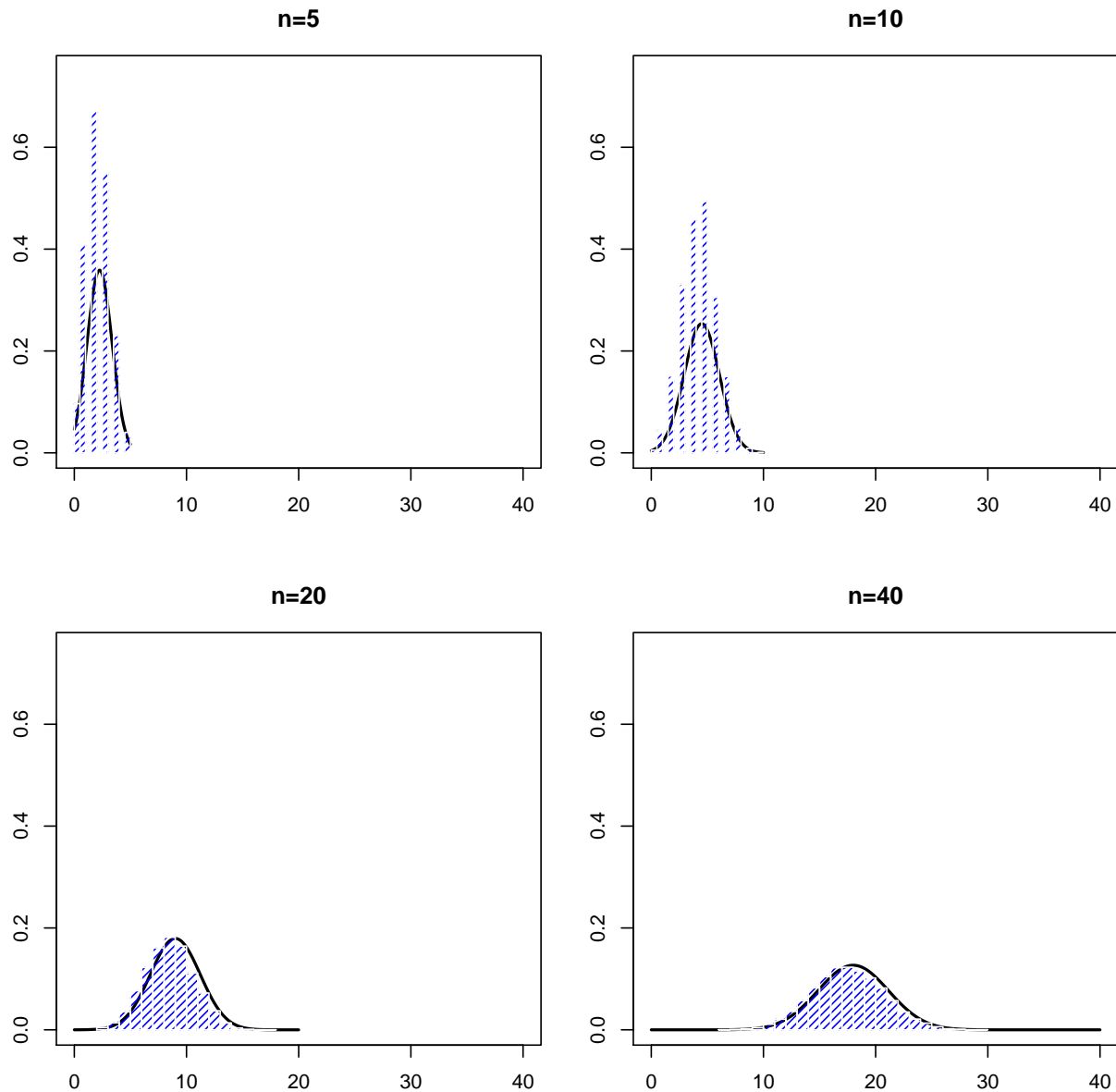
- collect many random draws from a binomial distribution – which basically means taking many samples of Bernoulli-distributed data,
- plot the histogram of the number of heads for all these samples,
- see how close that histogram is to the corresponding normal distribution.

We will take 10000 samples for each of the sample sizes $n = 5, 10, 20, 40$ and generate the corresponding plots. Note that `rbinom` is the R command to take random draws from a binomial distribution.

```

> theta <- 0.45
> par(mfrow = c(2, 2), mai = c(0.62, 0.52, 0.52, 0.12))
> n <- 5
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.75), lwd = 2)
> hist(rbinom(10000, n, theta), col = "blue", density = 25, border = "white",
+      freq = F, add = T)
> n <- 10
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.75), lwd = 2)
> hist(rbinom(10000, n, theta), col = "blue", density = 25, border = "white",
+      freq = F, add = T)
> n <- 20
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.75), lwd = 2)
> hist(rbinom(10000, n, theta), col = "blue", density = 25, border = "white",
+      breaks = 15, freq = F, add = T)
> n <- 40
> k <- 1:n
> x <- seq(0, n, length.out = 1000)
> plot(x, dnorm(x, n * theta, sqrt(n * theta * (1 - theta))), type = "l",
+      xlab = "", ylab = "", main = paste("n=", n, sep = ""), xlim = c(0,
+      40), ylim = c(0, 0.75), lwd = 2)
> hist(rbinom(10000, n, theta), col = "blue", density = 25, border = "white",
+      breaks = 25, freq = F, add = T)

```



```
> par(mfrow = c(1, 1), mai = c(1.02, 0.82, 0.82, 0.42))
```

References

- Abelson, R.P. (1995). *Statistics as Principled Argument*. L. Erlbaum Associates.
- Anderson, John R. (1989). "A Theory of the Origins of Human Knowledge". In: *Artificial Intelligence* 40, pp. 313–351.
- Baayen, R. Harald (2008). *Analyzing Linguistic Data: A Practical Introduction to Statistics Using R*. Cambridge University Press.
- Braun, J. and D.J. Murdoch (2007). *A First Course in Statistical Programming with R*. Cambridge University Press.
- De Veaux, R.D. et al. (2005). *Stats: Data and Models*. Pearson Education, Limited.

- Diez, D. et al. (2013). *OpenIntro Statistics: Second Edition*. CreateSpace Independent Publishing Platform.
URL: <http://www.openintro.org/stat/textbook.php>.
- Faraway, J.J. (2004). *Linear Models With R*. Chapman & Hall Texts in Statistical Science Series. Chapman & Hall/CRC.
- Gelman, A. and J. Hill (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Analytical Methods for Social Research. Cambridge University Press.
- Gonick, L. and W. Smith (1993). *Cartoon Guide to Statistics*. HarperCollins.
- Gries, S.T. (2009). *Quantitative Corpus Linguistics with R: A Practical Introduction*. Taylor & Francis.
- (2013). *Statistics for Linguistics with R: A Practical Introduction, 2nd Edition*. Mouton De Gruyter.
- Johnson, K. (2008). *Quantitative methods in linguistics*. Blackwell Pub.
- Kruschke, John K. (2011). *Doing Bayesian Data Analysis: A Tutorial with R and BUGS*. Academic Press/Elsevier.
- Lesigne, Emmanuel (2005). *Heads Or Tails: An Introduction to Limit Theorems in Probability*. American Mathematical Society.
- Miles, J. and M. Shevlin (2001). *Applying Regression and Correlation: A Guide for Students and Researchers*. SAGE Publications.
- Walker, Helen M. (1940). “Degrees of freedom”. In: *Journal of Educational Psychology* 31, pp. 253–269.
- Wang, Zhou and Alan C. Bovik (2009). “Mean squared error: love it or leave it? A new look at signal fidelity measures”. In: *Signal Processing Magazine, IEEE* 26, pp. 98–117.
- Wright, D.B. and K. London (2009). *Modern regression techniques using R: A practical guide for students and researchers*. SAGE.
- Xie, Yihui (2013). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC.