

Chapter 1

Introduction

1.1 This course

This course requires a good understanding of the Part IB course *Statistics*. (Don't worry if you are rusty; we will review all we need from IB *Statistics* in the first two lectures.) This course complements the Part II courses *Principles of Statistics* and *Mathematics of Machine Learning* by providing a more applied and computational perspective. You will learn to write some R code, use statistical models to analyze some real datasets, prove a few results that are not too technically challenging but very insightful, and, most importantly, think like a statistician.

The Michaelmas 2026 course has 24 sessions on Mondays, Wednesdays and Fridays, 10–11am in MR9, from 9 October to 2 December. On the course website you will also find the lecture notes from 2024 before the schedule was last updated. Additionally, you might find the following books useful:

- A. Agresti. *Foundations of Linear and Generalized Linear Models*. Wiley 2015. (Especially Chapters 2, 3, 4, 7.)
- D. Freedman. *Statistical Models: Theory and Practice*. Cambridge University Press 2009. (Provides perspectives from causal inference and social science.)
- G. James, D. Witten, T. Hastie, R. Tibshirani. *An Introduction to Statistical Learning (with Applications in R)*. Springer 2013. (Provides perspectives from machine learning.)
- B. Efron, T. Hastie. *Computer Age Statistical Inference: Algorithms, Evidence and Data Science*. Cambridge University Press 2016. (Not in the schedules but expand course materials in many ways.)

We will make the distinction between lectures and practical sessions (in which you will need to bring in a laptop with R installed) less pronounced.

For Cambridge mathematics students, it might not be obvious that *statistics is not a branch of mathematics*.¹ There is no consensus on the definition of statistics (especially with the rise of machine learning and data science), but the following definition in Wikipedia cannot be too wrong:

- *Statistics is the discipline that concerns the collection, organization, analysis, interpretation, and presentation of data.*

Compare this with the following definition of mathematical statistics:

- *Mathematical statistics is the study of statistics from a mathematical standpoint, using probability theory as well as other branches of mathematics such as linear algebra and analysis.*

This is similar in many ways to physics versus theoretical physics. Another way to think about it is that mathematics is mostly about deductive reasoning from a set of axioms and assumptions, while statistics is mostly concerned with inductive reasoning from empirical data.² Through exploring different statistical models and learning R, a great programming language for statistical computing, you will be exposed to both the mathematical and non-mathematical elements of statistics.

To understand how statistics are used in practice, the following quote by George Box³ may be illuminating:

Scientific research is usually an iterative process. The cycle: conjecture–design–experiment–analysis leads to a new cycle of conjecture–design–experiment–analysis and so on.... The experimental environment ... and techniques appropriate for design and analysis tend to change as the investigation proceeds.

At one point, the dominant view was that statistical modelling is a critical step of “analysis” and the model is built after data are collected. However, modern statisticians (and in fact, many pioneers like Box and Fisher) view statistical model as an essential component of the scientific process that guides all steps of the cycle and is being continuously updated. And others like John Tukey envisioned a more dynamic/less rigorous process. My personal take is a tree—“statistical theory” is the root, “statistical methodology” is the trunk, “statistical principles” are the branches, and “statistical practices” are the leaves and fruits. In any case, the title of this course is of great taste: we will learn about “statistical modelling” instead of just “statistical models”.

In the broad sense, a statistical model is a mathematical representation of some real-world process that generate data. With the rise of data mining/machine learning/big data/data science/artificial intelligence, you will notice a divide between “two cultures” of statistical modelling: the “data modelling” culture and the “algorithmic modelling” culture in the language used by Leo Breiman a quarter of a century ago.⁴ This is about two aspects of any data analyses:

Computing Numbers (or other types of data) go into a computer, and some numbers (or other types of data) come out.

Inference What the input means, and what the output tells us about our problem.

While statistics are mainly concerned with inference in the past, computing is playing a bigger and bigger role. We will try to maintain a neutral position and introduce both perspectives in this course.

We will primary use “statistical model” in the narrow sense to refer to a collection of probability distributions $\mathbb{P} = \{P_\theta : \theta \in \Theta\}$.⁵ Statistical inference in the narrow sense is about learning the unknown parameter θ or the distribution P_θ using a realization from P_θ (this is why statistics was sometimes called “inverse probability” in very old times). Actually most of the time we will specify statistical models by making restrictions/assumptions on (the probability distribution of) the random variables. Although we often do not write down the statistical model explicitly, you should always be prepared to accept that challenge.

A slightly more technical point is that statistical models can be defined at different levels:

- (i) Models for conditional moments. For example, a *linear model for conditional expectation* assumes $E[Y | X = x] = x^T \beta$.
- (ii) Models for joint or conditional distributions. For example, the classical *normal linear model* assumes $Y = X^T \beta + \epsilon$ where the noise variable $\epsilon \perp\!\!\!\perp X$ and $\epsilon \sim N(0, \sigma^2)$.
- (iii) Structural or causal models that not only describe (associational) relationship for the data at hand but also (causal) relationship under counterfactual interventions. For example, the *linear structural equation model* assumes $Y^{(x)} = x^T \beta + \epsilon$, where $Y^{(x)}$ is the counterfactual value of Y under the intervention that sets X to x and ϵ is an independent noise variable.

This course used to be mainly about models of the second kind, but we will try to cover other kinds of models as part of the modernization effort.

1.2 Basic probability and statistics

1.2.1 Notation

Upper-case letters indicate matrices or random variables. Lower-case letters indicate fixed quantities. We use I_p to denote the $p \times p$ identity matrix, 1_p to denote the p -vector of ones, and 0_p the p -vector of zeros. Independent random variables (or vectors) X and Y are denoted as $X \perp\!\!\!\perp Y$. As a convention, we usually use subscript $i \in \{1, \dots, n\}$ to index observations and $j \in \{1, \dots, p\}$ to index variables. “Independent and identically distributed” is abbreviated as “i.i.d.”. The Euclidean norm of a vector Y is denoted as $\|Y\|$. Convergence in probability is denoted as $\xrightarrow{P}$. Convergence in distribution (weak convergence) is denoted as $\xrightarrow{d}$. We use sans serif font $\mathbf{P}$ to denote a probability distribution and $\mathbb{P}$ to denote a statistical model—a collection of probability distributions.

1.2.2 Basic probability

A *probability space* $(\Omega, \mathcal{F}, \mathbf{P})$ is a triple where Ω is a set of outcomes (the *sample space*), $\mathcal{F}$ is a collection of subsets of Ω that is a σ -algebra (the *event space*), and $\mathbf{P} : \mathcal{F} \rightarrow [0, 1]$ is a function that satisfies the standard Kolmogorov axioms (non-negativity, unit measure, and countable additivity). A (*real-valued*) *random variable* X is a (Borel measurable) function from the probability space to $\mathbb{R}$. A *random vector* X is a (Borel measurable) function from the probability space to $\mathbb{R}^d$, $d \geq 2$. These abstract definitions will never be used and best forgotten in this course. Instead, what is important for us is the probability measure on $\mathbb{R}$, $\mathbb{R}^d$, or a more general space that the data live in that is *induced by the random variable*. So the basic objects for us are the data—modelled as a random variable/vector—and how they are generated—modelled by their probability distribution, which we also denote by $\mathbf{P}$.

There are many ways to characterize the probability distribution of a real-valued random variable X . The *cumulative distribution function* (CDF) of X is defined as⁶

$$F(x) = \mathbf{P}(X \leq x).$$

The *quantile function* is the inverse of the CDF:

$$F^{-1}(u) = \inf\{x \in \mathbb{R} : F(x) \geq u\}.$$

If X is (absolutely) continuous, we can describe the distribution of X using its *probability density function*

$$f(x) = \frac{d}{dx} F(x).$$

If X is discrete, we can use its probability mass function

$$f(x) = \mathbf{P}(X = x).$$

If you know the Radon-Nikodym theorem, you will know these are basically the same thing (so we use the same notation). The *moment generating function* is defined as

$$M(t) = \mathbf{E}(e^{tX}), \quad t \in \mathbb{R}.$$

We know $M(0) = 1$, but it can be infinite elsewhere. The expectation $\mathbf{E}$ here is over the probability distribution $\mathbf{P}$ and you should know how that is defined.⁷ All these definitions, besides the quantile function, can be naturally extended to multivariate X .

At this point, you should check if you can confidently answer the following questions:

- (i) What does it mean for random variables $X_1, \dots, X_n$ to be *independent*? What is the implication of independence on the *joint density function* of $X = (X_1, \dots, X_n)$?
- (ii) How do you compute the expectation and variance of a random variable? If $X = (X_1, \dots, X_n)$ is a random vector, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ are fixed, what is the expectation and covariance matrix of $AX + b$?
- (iii) What is the law of large numbers? What is the central limit theorem?
- (iv) What is conditional expectation? What is the law of total expectation?

If you are unsure about anything above, take a look at the notes for IB Statistics lecture 1.

We now review the normal distribution and related distributions. A d -dimensional random vector Z is said to follow the *multivariate normal* distribution with mean $\mu \in \mathbb{R}^d$ and positive definite⁸ covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$, written as $Z \sim N_d(\mu, \Sigma)$, if its probability density function is given by

$$f(z) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-(z-\mu)^T \Sigma^{-1} (z-\mu)/2}.$$

The multivariate normal distribution has two important properties:

- (i) If $Z \sim N_d(\mu, \Sigma)$, then for any fixed matrix $A \in \mathbb{R}^{k \times d}$ and vector $b \in \mathbb{R}^k$, $AZ + b \sim N_k(A\mu + b, A\Sigma A^T)$.
- (ii) If Z_1 and Z_2 are two random vectors and $\begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix}$ follows a multivariate normal distribution, then $Z_1 \perp Z_2$ if and only if $\text{Cov}(Z_1, Z_2) = 0$.

We often omit the subscript d in $N_d(\mu, \Sigma)$ if the dimension is clear from the context.

Let $Z \sim N_d(0, I)$. Then we say

$$\|Z\|^2 = \sum_{i=1}^d Z_i^2 \sim \chi_d^2$$

follows the *chi-square distribution* with d degrees of freedom. Suppose $Z \sim N(0, 1)$, $S \sim \chi_d^2$, and $Z \perp S$. Then we say

$$\frac{Z}{\sqrt{S/d}} \sim t_d$$

follows the (*Student's*) *t-distribution*⁹ with d degrees of freedom. Suppose $S_1 \sim \chi_{d_1}^2$, $S_2 \sim \chi_{d_2}^2$, and $S_1 \perp S_2$. Then we say

$$\frac{S_1/d_1}{S_2/d_2} \sim F_{d_1, d_2}$$

follows the *F-distribution* with degrees of freedom d_1 and d_2 .

As a memory aid, the above definitions can be summarized as follows:

$$\chi_d^2 = \underbrace{N(0, 1)^2 + \cdots + N(0, 1)^2}_{d \text{ times}}, \quad t_d = \frac{N(0, 1)}{\sqrt{\chi_d^2/d}}, \quad F_{d_1, d_2} = \frac{\chi_{d_1}^2/d_1}{\chi_{d_2}^2/d_2}.$$

In this informal description, two random variables (as indicated by their distributions) are independent whenever they appear in the same expression. It is obvious that $t_d^2 = F_{1, d}$.

1.2.3 Basic statistics

A *statistical model* is a collection of probability distributions $\mathbb{P} = \{\mathbb{P}_\theta : \theta \in \Theta\}$ indexed by a unknown *parameter* θ . This notation comes with an implicit bias that the interesting object is the parameter θ , but a true statistician is really interested in knowing the probability distribution $\mathbb{P}_\theta$. This motivates the concept of *identifiability*: we say a statistical model is *identifiable* if $\mathbb{P}_{\theta_1} = \mathbb{P}_{\theta_2}$ implies $\theta_1 = \theta_2$. Most of the models in this course are *parametric*, meaning Θ is an (usually open) subset of $\mathbb{R}^p, p \geq 1$. However, nonparametric and semiparametric models, in which Θ is infinite dimensional, do creep in. The properties of any statistical procedure (how we interpret the “numbers” coming out of a computer) depend on the statistical model $\mathbb{P}$. When doing math we assume the statistical model $\mathbb{P}$ is given, but that is usually not the case when we analyze data.

We learned in IB *Statistics* about ways to obtain point/interval estimators of θ and test a hypothesis like $\theta \in \Theta_0$. In parametric models, the center pillar for this the *likelihood function*:

$$L(\theta) = f(X; \theta),$$

where the data $X \sim \mathbb{P}_\theta$ with density function $f(\cdot; \theta)$. In words, the likelihood function is the density function evaluated at the data and viewed as a function of the parameter. The *maximum likelihood estimator* (MLE) is defined as

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\theta} L(\theta).$$

It is often easier to do this maximization using the log-likelihood function, $l(\theta) = \log L(\theta)$. The MLE is a *statistic* (a function of the data), because the likelihood function depends on the data.

We can characterize how good a point estimator $\hat{\theta}$ is using its *bias*:

$$\text{Bias}_\theta(\hat{\theta}) = \mathbb{E}_\theta(\hat{\theta}) - \theta$$

and its *mean squared error* (MSE):

$$\text{MSE}_\theta(\hat{\theta}) = \mathbb{E}_\theta\{(\hat{\theta} - \theta)^2\}.$$

These are functions of the unknown parameter θ , which is important because we care about how good $\hat{\theta}$ is over the entire statistical model. We say $\hat{\theta}$ is *unbiased* if $\text{Bias}(\hat{\theta}; \theta) = 0$ for all θ . We say a statistic $T = T(X)$ is *sufficient* if the conditional distribution of X given T does not depend on θ . The *factorization theorem* says that T is sufficient if and only if $f(x; \theta) = g(T(x), \theta)h(x)$ for some suitable functions g and h . The *Rao-Blackwell theorem* says that if T is a sufficient statistic, then the MSE of $\mathbf{E}(\tilde{\theta} | T)$ is no larger than that of $\tilde{\theta}$ for any estimator $\tilde{\theta}$ at any θ . The mean squared error admits a *bias-variance decomposition*:

$$\text{MSE}_\theta(\hat{\theta}) = \text{Bias}_\theta(\hat{\theta})^2 + \text{Var}_\theta(\hat{\theta}).$$

So it is possible for biased estimators to have smaller MSE than unbiased estimators.

We say $S(X) \subseteq \Theta$ is a $(1 - \alpha)$ -*confidence set* of θ if

$$\mathbf{P}_\theta\{\theta \in S(X)\} = 1 - \alpha, \text{ for all } \theta \in \Theta.$$

In IB *Statistics* we only looked at the case where $S(X) = [L(X), U(X)]$ is an interval. The key to construct confidence intervals/sets is usually a *pivotal quantity* $R(X, \theta)$ whose distribution under $\mathbf{P}_\theta$ does not depend on θ .

Hypothesis testing is often formulated as deciding between $H_0 : \theta \in \Theta_0$ and $H_1 : \theta \in \Theta_1$ (usually $\Theta_1 = \Theta \setminus \Theta_0$), and the decision can be represented by a test function $T(X) \in \{0, 1\}$. The key concept is the *power function* $\beta(T; \theta) = \mathbf{P}_\theta(T(X) = 1)$. The *size* of the test is $\sup_{\theta \in \Theta_0} \beta(T, \theta)$. The Neyman-Pearson theory formulates hypothesis testing as maximizing $\beta(T; \theta)$ for $\theta \in \Theta_1$ while controlling its size at some prespecified level α . For simple versus simple ($\Theta_0 = \{\theta_0\}$ and $\Theta_1 = \{\theta_1\}$), the most powerful test rejects H_0 if the following *likelihood ratio statistic* is larger than some threshold

$$\Lambda(X) = \frac{L(\theta_1)}{L(\theta_0)} = \frac{f(X; \theta_1)}{f(X; \theta_0)}.$$

This is a special case of the *generalized likelihood ratio statistic*

$$\Lambda(X) = \frac{\sup_{\theta \in \Theta_0 \cup \Theta_1} L(\theta)}{\sup_{\theta \in \Theta_0} L(\theta)}, \quad (1.1)$$

which can be used for composite hypotheses. It is often easier to work with them on the logarithmic scale. For the multinomial model $(N_1, \dots, N_k) \sim \text{Multi}(n; p_1, \dots, p_k)$ where n is given and $\theta = (p_1, \dots, p_k)$ is any vector in the standard $(k - 1)$ -simple, we saw in IB *Statistics* that the generalized log-likelihood ratio statistic for testing $H_0 : p_i = p_{0i}, i = 1, \dots, k$ for some given $\theta_0 = (p_{01}, \dots, p_{0k})$ is given by

$$2 \log \Lambda = 2 \sum_{i=1}^k O_i \log \left(\frac{O_i}{E_i} \right) \approx \sum_{i=1}^k \frac{(O_i - E_i)^2}{E_i},$$

where $O_i = N_i$ is the “observed count”, $E_i = np_{0i}$ is the “expected count”, and the approximation on the right hand side is called Pearson’s χ^2 -statistic because it (and also the other expression) converge in distribution to χ_{k-1}^2 as $n \rightarrow \infty$ by Wilks’ theorem.

There is a useful duality between simple hypothesis tests and confidence sets. Consider

$$T(\theta, X) = 1_{\{\theta \notin I(X)\}} \text{ and } S(X) = \{\theta : T(\theta, X) = 0\}.$$

Then $T(\theta_0, X)$ is a size- α test of $H_0 : \theta = \theta_0$ for all $\theta_0 \in \Theta$ if and only if $S(X)$ is a $(1 - \alpha)$ -confidence set for θ .

For hypothesis tests, it is often easier to report the *P-value*:

$$P = \sup_{\theta \in \Theta_0} P(\theta, X), \text{ where } P(\theta, X) = \mathbb{P}_\theta(\Lambda(\tilde{X}) \geq \Lambda(X) \mid X),$$

and $\tilde{X}$ is an i.i.d. copy of X ($\tilde{X} \sim \mathbb{P}_\theta$ and $\tilde{X}$ is independent of X). So $P(\theta, X)$ measures how “extreme” $\Lambda(X)$ is if the data X are generated from $\mathbb{P}_\theta$. Rejecting H_0 when $P \leq \alpha$ (so $T = 1_{\{P \leq \alpha\}}$) ensures that the test has size α . This corresponds to the $(1 - \alpha)$ -confidence set

$$S(X) = \{\theta \in \Theta : P(\theta, X) > \alpha\},$$

which contains all θ not rejected at level α .

Thus far we have discussed frequentist inference that treats probability as long-term frequency and the parameter θ as a fixed quantity describing a natural law. An alternative viewpoint is Bayesian inference that treats probability as subjective plausibility and the parameter θ as a random variable, with *prior distribution* π . By using the Bayes formula, the *posterior distribution* of θ is given by

$$\pi(\theta \mid X) = \frac{\pi(\theta)f(x \mid \theta)}{f(x)},$$

where $f(x) = \int f(x \mid \theta)\pi(\theta)d\theta$ is the marginal density of X . We often ignore the normalizing constant and write $\pi(\theta \mid X) \propto \pi(\theta)f(x \mid \theta)$, or in other words,

$$\text{posterior} \propto \text{prior} \cdot \text{likelihood}.$$

So we use information in the likelihood to update our belief about θ .

In Bayesian statistics, all we know about the parameter is contained in the posterior distribution $\pi(\theta \mid X)$. We can use the *Bayes risk* to evaluate an estimator $\hat{\theta} = \hat{\theta}(X)$:

$$R(\hat{\theta}) = \mathbb{E}_\pi(L(\theta, \hat{\theta}) \mid X) = \int_{\Theta} L(\theta, \hat{\theta}(X))\pi(\theta \mid X)d\theta,$$

where $L : \Theta \times \Theta \rightarrow \mathbb{R}$ is a *loss function*. The *Bayes estimator* is given by $\hat{\theta} = \arg \min_{\tilde{\theta}} R(\tilde{\theta})$. We saw in IB *Statistics* that the Bayes estimator under the quadratic loss $L(\theta, \tilde{\theta}) = (\theta - \tilde{\theta})^2$ is the posterior mean $\mathbb{E}_\pi(\theta \mid X)$.

Exercise 1.1. Suppose X is a random variable with a continuous cumulative distribution function $F(x)$. Without assuming $F(x)$ is strictly increasing, show that $F(X) \sim \text{Unif}[0, 1]$. Use this to show that $P(\theta, X)$ is a pivotal quantity if the CDF of $\Lambda(X)$ is continuous for all $\theta \in \Theta$.

1.3 Practical 1: R basics

R is a free and open-source programming language for statistical computing and data visualization. Please download and install the appropriate distribution of R from <https://cran.r-project.org/>. Many students will find it easier to use an integrated development environment (IDE) such as RStudio, which can be downloaded from <https://posit.co/download/rstudio-desktop/>.

1.3.1 Arithmetics

R can be used as a calculator:

```
> (9.1^3) * sqrt(14) * exp(-5) / log(4)
```

Help on any R function can be found by typing a question mark followed by the function.

```
> ?exp
> help(exp)
> ??exp
```

You will need to use this **help** facility extensively (and get used to skim-reading to find the relevant bit!). Note that R is case-sensitive.

The `<-` symbol is the usual assignment operator in R (the `=` symbol can also be used, but it has slightly different purposes so you are recommended to stick to `<-`). For instance, we can assign the value 3 to the variable `x`, and then perform operations on `x`. Anything which appears after the hash symbol `#` is a comment and need not be typed.

```
> x <- 3
> round(x^2 + log10(x), 3) # try ?round to see what it does
[1] 9.477
> 37 %/% 3 # try ?'%/%'
[1] 12
> 37 %% 3
[1] 1
```

1.3.2 Vectors, matrices, lists

Creating vectors

The `c` function (for “concatenate”) combines values into a vector.

```
> x <- c(3, 6, 4, 2)
> x
[1] 3 6 4 2
> length(x)
[1] 4
```

There is no such thing as a scalar in R; what one might think of as a scalar is treated as a vector of length 1. Note that, unlike MATLAB, R does not distinguish between row and column vectors.

A sequence of equally spaced numbers can be created using the `seq` function. The `rep` function provides different ways of repeating vectors.

The base R does many things in a “smart” way. This is often convenient but can sometimes be perilous. For example, the default `seq` method takes the following input:

```
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)),
    length.out = NULL, along.with = NULL, ...)
```

```
> seq(1, 10)
[1] 1 2 3 4 5 6 7 8 9 10
> seq(10)
[1] 1 2 3 4 5 6 7 8 9 10
> seq(1, 10, 2)
[1] 1 3 5 7 9
> seq(1, by = 2, length = 5)
[1] 1 3 5 7 9
> seq(1, 10, length = 5)
[1] 1.00 3.25 5.50 7.75 10.00
> seq(10, length = 5)
[1] 10 11 12 13 14
> seq(to = 10, length = 5)
[1] 6 7 8 9 10
```

The `seq` function is a bit extreme, but it is generally a good idea to try things out when you are not sure.

Operations on vectors

Operations on vectors in R are performed component by component. For example

```
> x + x
[1] 6 12 8 4
> x*x
[1] 9 36 16 4
> exp(x)
[1] 20.085537 403.428793 54.598150 7.389056
```

Another “smart” thing in R: when operations are performed on vectors of different lengths, the shorter vector is cycled until it is the same length as the longer vector.

```
> x <- c(3, 6, 4, 2)
> y <- c(1, 2)
> x + y
[1] 4 8 5 4
```

```

> x*y
[1] 3 12 4 4
> x^y
[1] 3 36 4 4
> y <- 1:3 # same as y <- c(1, 2, 3) or y <- seq(1, 3)
> x + y
[1] 4 8 7 3
Warning message:
In x + y : longer object length is not a multiple of shorter object length

```

What are the values of $x + 2$, $3*x$ and $(2 + x)^3$?

Indexing vectors

```

> x <- c(3, 6, 4, 2)
> x[2] # 2nd component of x
[1] 6
> x[c(1, 3)] # 1st and 3rd components of x
[1] 3 4
> x[-1] # All of x except the 1st component
[1] 6 4 2
> x[-(1:2)] # All of x except the 1st two components
[1] 4 2
> x[1:2] <- c(7.1, 3.4) # We can assign values to components
> x
[1] 7.1 3.4 4.0 2.0

```

Note that after the final command, x has automatically transformed from a vector of integers to a vector of *floating point numbers* (these are a way of representing real numbers on computers, though of course only to a certain degree of accuracy).

We can also index components of a vector using a **TRUE / FALSE** (logical) vector.

```

> index_vec <- c(TRUE, TRUE, FALSE, TRUE)
> x[index_vec]
[1] 7.1 3.4 2.0

```

Logical vectors can also be created using the binary operator $<$ which performs componentwise comparisons.

```

> x > 3.6
[1] TRUE FALSE TRUE FALSE
> x[x > 3.6]
[1] 7.1 4.0

```

Matrices

We can create a matrix using the `matrix` function.

```
> A <- matrix(1:8, 2, 4)
> A
      [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

Can you enter the terms by row instead?

Rows and columns of matrices can be extracted in the following way:

```
> A[1, ]
[1] 1 3 5 7
> A[, 3]
[1] 5 6
```

Note that the rows and columns thus formed are now vectors. We can check this using the very helpful `str` (for ‘structure’) function.

```
> str(A[1, ])
int [1:4] 1 3 5 7
```

Here we see that `A[1, ]` is an integer vector of length 4. To keep the 2-by-1-matrix structure-type, we use

```
> A[, 2, drop = FALSE]
      [,1]
[1,]     3
[2,]     4
```

An alternative is to do

```
> matrix(A[, 2])
      [,1]
[1,]     3
[2,]     4
```

Submatrices can be formed by e.g. `A[, 1:3]`. The diagonal can be extracted using `diag`. We can perform many standard operations on matrices.

```
> A %*% x # matrix vector multiplication
      [,1]
[1,] 51.3
[2,] 67.8
> A*A # componentwise multiplication
      [,1] [,2] [,3] [,4]
[1,]    1    9   25   49
```

```

[2,]    4   16   36   64
> t(A)
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
> A %*% t(A) # matrix matrix multiplication
      [,1] [,2]
[1,]   84  100
[2,]  100  120

```

The `solve` function can be used to solve linear systems like $Ax = b$. If b is missing in the input, it will invert the matrix A .

```

> solve(A %*% t(A))
      [,1] [,2]
[1,]  1.50 -1.25
[2,] -1.25  1.05

```

Lists

Lists collect together items of different types, e.g.

```

> Empl <- list(employee="Eve", spouse = "Adam",
               children=2, child.ages = c(4, 7))

```

Elements of a list need not be of the same length, but its components are numbered. Thus `Empl` is a list of length 4, and its components are referred to as `Empl[[1]]` etc.. Notice that `Empl[[4]]` is a vector, so `Empl[[4]][1]` is its first entry. Names of components can also be used:

```

> Empl$employee
> Empl[["employee"]]
> Empl$child.ages[2]

```

The different components of a list can be almost anything, even functions or other lists.

1.3.3 Functions

A few important functions

```

> x <- c(3, 6, 4, 2)
> sum(x)
[1] 15
> sum(x > 3) # TRUE is treated as 1 and FALSE, 0
[1] 2
> mean(x)

```

```
[1] 3.75
> sort(x)
[1] 2 3 4 6
> sd(x) # standard deviation
[1] 1.707825
```

How is the standard deviation being calculated? Functions for matrices:

```
> mean(A) # mean treats A as a vector
[1] 4.5
> colMeans(A)
[1] 1.5 3.5 5.5 7.5
> rowSums(A)
[1] 16 20
```

The function `cbind` ‘glues’ columns of matrices together.

```
> cbind(1, A)
      [,1] [,2] [,3] [,4] [,5]
[1,]    1    1    3    5    7
[2,]    1    2    4    6    8
```

Writing your own functions

When writing anything more than a few lines, it is useful to edit the commands from a script file that ends with `.R`. Create a script file (in Rstudio you can use `Ctrl+Shift+N`), then copy and paste the following code:

```
f <- function (x, y) {
  z <- x^2 + y^2
  return(c(cos(z), sin(z)))
}
```

Once you have written the algorithm and saved the file as `Rubbish.R`, say, in the current working directory, you can execute the commands by typing

```
> source("Rubbish.R")
```

in the console. Typing `f` in the console will now echo the code of your function, and you can run your function by giving it the right arguments e.g. `f(2, 3)`.

Generating (pseudo) random numbers

(Pseudo) independent and identically distributed sequences of random numbers are generated with commands like `rnorm`, `runif`, `rchisq` etc. (normal, uniform, χ^2). The corresponding density, cumulative distribution and quantile functions are, e.g. `dnorm`, `pnorm`, `qnorm`.

```

> x <- rnorm(1000)
> hist(x, freq = FALSE)
> x_vec <- seq(-3, 3, by = 0.1)
> lines(x_vec, dnorm(x_vec), col = "red") # adds lines to an existing plot

```

What does the following code do?

```

> X <- matrix(runif(50*1000, min=-1, max=1), 50, 1000)
> hist(sqrt(50) * colMeans(X) / sd(X), freq = FALSE) # sd treats X as a vector
> lines(x_vec, dnorm(x_vec), col = "red")

```

`lines` plots on top of the current plot, but if you wish to use superposition to histograms (or plots) you can use the following instruction (equivalent to to “hold on” in MATLAB).

```

> par(new=TRUE)

```

Experiment with other distributions and other sample sizes.

1.3.4 Loops and vectorization

for Loops and while Loops

Often we will like to run the same code many times, for instance if we using simulated data to examine a statistical methodology. The prime tools to do this are `for` and `while` loops in the code. These loops repeat the code inside the `{ }` blocks then iterates until the for loop runs out of indices or the while loop condition is broken. These have similar syntax:

```

for (i in 1:B) { doSomething() }

counter <- 0
while (counter < B) {
  doSomething()
  counter <- counter + 1
}

```

both of which accomplish the same goal of calling the `doSomething` function B times.

This approach applies the code blocks in sequence, which is useful if accessing the same memory block and adjusting it multiple times (which is what happened to the memory storing i in the while loop). However often the code will access new memory for each iteration, and so we could instead run these in parallel. For further reading on this, search for the documentation on the `parallel` package in R.

Vectorization of Code

Consider the computing the column means of the elements in a large matrix X using a for loop:

```
n <- 100
d <- 3000

X <- matrix(rnorm(n * d), nrow = n, ncol = d)
X_means <- rep(0, d)
for (i in 1:d) {
  X_means[i] <- mean(X[,i])
}
```

we can use the `system.time` function to test how long this computation takes:

```
system.time(for (i in X) {X_means[i] <- mean(X[,i])})
      user  system elapsed 
43.518    7.317   51.124
```

In contrast, we can compute this much more quickly by using the `apply` function to compute these column means in parallel.

```
system.time(apply(X, 1, mean))
      user  system elapsed 
0.002    0.000    0.002
```

This is much faster because vectorized computation in R directly calls functions written in C. Vectorizing the for loops also helps you focus on the statistical operations.

1.3.5 Exercises

You are recommended to discuss computing exercises in this course with a learning partner during and/or after the practical sessions.

Exercise 1.2. Use R to solve the following linear equations:

$$\begin{aligned}3a + 4b - 2c + d &= 9 \\2a - b + 7c - 2d &= 13 \\6a + 2b - c + d &= 11 \\a + 6b - 2c + 5d &= 27.\end{aligned}$$

Exercise 1.3. Estimate the upper 5% quantile of a χ_6^2 distribution using the `dchisq` and `quantile` functions, then validate your answer using `qchisq`.

Exercise 1.4. Consider a real valued random variable X with distribution function given by:

$$F_X(x) = \frac{1}{\pi} \arctan(x) + \frac{1}{2}.$$

Write a function `rmydist(n)` that generates n independent samples of X . Use this function to estimate $\text{Var}(X)$. [*Hint*: Theorem 1.1]

Exercise 1.5. Two lecturers mark the same Tripos question for two randomly selected, disjoint sets of students. To make sure that neither of them is more lenient than the other, they want to test whether their average marks are equal. But they are afraid the sample size is too small to apply the Central Limit Theorem, so one of them writes the following code.

```
grades1 <- c(10,11,14.5,15,15,18,12,19,18.5,19,20,13)
grades2 <- c(12,11,14.5,13,12,11,12,14.5,20,17)
(diff <- mean(grades1) - mean(grades2))
grades.all <- c(grades1,grades2)
n1 <- length(grades1); n2 <- length(grades2);

one.sim <- function(iter) {
  grades.perm <- sample(grades.all);
  mean(grades.perm[1:n1]) - mean(grades.perm[n1 + 1:n2])
}

system.time(diff.perms <- replicate(100000, one.sim(0)))
typeof(diff.perms)
mean(abs(diff.perms) > diff)

system.time(diff.perms <-
  parallel::mclapply(1:100000, one.sim, mc.cores = 2))
typeof(diff.perms)
diff.perms <- unlist(diff.perms)
mean(abs(diff.perms) > diff)

hist(abs(diff.perms))
abline(v = abs(diff), col = "red", lty = "dashed")
```

Execute the code in your R session and explain what it is doing. Search for the documentation of any function you have not encountered before.

1.4 Normal linear models

1.4.1 Model and likelihood

We now review the normal linear model introduced in IB *Statistics*. This model assumes that $(X_1, Y_1), (X_2, Y_2), \dots, (X_n, Y_n) \in \mathbb{R}^p \times \mathbb{R}^1$ are independent and the conditional distribution of Y_i given X_i satisfies¹⁰

$$Y_i = \sum_{j=1}^p X_{ij}\beta_j + \epsilon_i, \quad \epsilon_i \perp\!\!\!\perp X_i, \quad \epsilon_i \sim N(0, \sigma^2), \quad i = 1, \dots, n. \quad (1.2)$$

Equation (1.2) is rather cumbersome and can be simplified by introducing the following vector/matrix notation:

$$Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}, \quad X = \begin{pmatrix} X_1^T \\ \vdots \\ X_n^T \end{pmatrix}, \quad \beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \text{and } \epsilon = \begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{pmatrix}.$$

Now we can rewrite (1.2) as

$$Y = X\beta + \epsilon, \quad \epsilon \mid X \sim N(0, \sigma^2 I_p). \quad (1.3)$$

This is also known as *linear regression*, due to early applications of such models by Francis Galton that identified a “regression toward the mean” phenomenon in the inheritance of human traits (such as height).

Example 1.6 (Normal measurements). This model assumes $Y_i \stackrel{i.i.d.}{\sim} N(\mu, \sigma^2)$, $i = 1, \dots, n$. The model matrix $X = 1_n$ is a matrix with just one column and the regression coefficient $\beta = \mu$ is one-dimensional.

Example 1.7 (ANOVA). Let $F_i \in \{1, \dots, l\}$ be a *categorical variable* with l levels (also called a *factor*). The classical ANalysis Of VAriance (ANOVA) assumes $Y_i = \beta_{F_i} + \epsilon_i$, $\epsilon_i \stackrel{i.i.d.}{\sim} N(0, \sigma^2)$ where β is a l -dimensional parameter vector. The i th row X_i of the corresponding model matrix X is an indicator vector whose F_i th entry is 1 and all other entries are 0.

The matrix X is known as the *design matrix* or *model matrix*. The former terminology was derived from the classical setting in experimental design in which X is chosen by the experimenter. This is rarely the case in modern applications. For this reason, we will refer to X as the model matrix in this course.

To distinguish (1.2) and (1.3) with models for the conditional expectation, let $\mu_i = E[Y_i \mid X_i]$. Then (1.3) contains three different types of assumptions:

(i) The conditional expectation satisfies

$$\mu = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix} = X\beta;$$

(ii) The noise $\epsilon = Y - \mu$ satisfies $\epsilon \perp\!\!\!\perp X$;

(iii) The noise ϵ is distributed as $N(0, \sigma^2 I_n)$.

Note that (1.2) does not make any assumptions on the distribution of the regressors X . This is dangerous, because the distribution of X should really be regarded as a parameter in the model and be included as an argument of the likelihood function in (1.4). In other words, although the normal linear model is often regarded as the most basic parametric model, it is actually semiparametric (involving infinite-dimensional parameters) in general. Of course, this remark no longer applies if one assumes X is fixed or the distribution of X is unknown, but in most applications the distribution of X is unknown.

In any case, the distribution of X does not really matter in the classical normal linear model because the assumption $\epsilon \perp\!\!\!\perp X$ allows us to factorize the likelihood function as

$$L(\beta, \sigma^2) = f(x_1, \dots, x_n, y_1, \dots, y_n; \beta, \sigma^2) = f(x_1, \dots, x_n) \cdot \prod_{i=1}^n f(y_i | x_i; \beta, \sigma^2), \quad (1.4)$$

where f is a generic symbol for density functions and $f(y_i | x_i; \beta, \sigma^2)$ is the density function of a normal random variable:

$$f(y | x; \beta, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y - x^T \beta)^2 / (2\sigma^2)}.$$

Because the marginal density function $f(x_1, \dots, x_n)$ of X does not depend on β , whether the distribution of X is known or not does not affect the inference for β following the likelihood principle.¹¹

1.4.2 Ordinary least squares and its geometry

Derivation of ordinary least squares

Following (1.4), the log-likelihood function is given by

$$\begin{aligned} l(\beta, \sigma^2) &= \log \prod_{i=1}^n f(Y_i | X_i; \beta) + \text{constant} \\ &= -\frac{n}{2} \log \sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^n (Y_i - X_i^T \beta)^2 + \text{constant} \\ &= -\frac{n}{2} \log \sigma^2 - \frac{1}{2\sigma^2} \|Y - X\beta\|^2 + \text{constant} \end{aligned}$$

Therefore, the maximum likelihood estimator (MLE) of β is given by the solution of the *ordinary least squares* (OLS) problem

$$\hat{\beta} = \arg \min_{\beta} \|Y - X\beta\|^2. \quad (1.5)$$

Notice that this holds regardless of whether σ^2 is known or not.

We may obtain a closed-form solution to (1.5) by using the following identities for matrix calculus:

$$\frac{\partial}{\partial \beta}(a^T \beta) = a, \text{ and } \frac{\partial}{\partial \beta}(\beta^T A \beta) = (A + A^T)\beta.$$

Therefore the OLS estimator satisfies

$$X^T(Y - X\hat{\beta}) = 0. \quad (1.6)$$

Equation (1.6) is called the *normal equations* because it requires the vector of *residuals* $R = Y - X\hat{\beta}$ to be orthogonal to X .

The linear equations (1.6) have a unique solution if $X^T X$ is invertible (or equivalently if X has full column rank, which requires $n \geq p$). In this case, we have

$$\hat{\beta} = (X^T X)^{-1} X^T Y.$$

We maintain the assumption that X has rank p in this Chapter.

The maximum likelihood estimator of σ^2 can be obtained by differentiating $l(\beta, \sigma^2)$ with respect to σ^2 :

$$\frac{\partial}{\partial \sigma^2} l(\beta, \sigma^2) = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4} \|Y - X\beta\|^2.$$

By solving $l(\hat{\beta}, \sigma^2) = 0$, we obtain

$$\hat{\sigma}_{\text{MLE}}^2 = \frac{1}{n} \|Y - X\hat{\beta}\|^2 = \frac{1}{n} \|R\|^2.$$

The quantity $\|R\|^2$ is often referred to as the *residual sum of squares* (RSS). Because $\hat{\sigma}_{\text{MLE}}^2$ is biased (see Section 1.4.3), it is more common to use the following unbiased estimator of σ^2 :

$$\hat{\sigma}^2 = \frac{n}{n-p} \hat{\sigma}_{\text{MLE}}^2 = \frac{1}{n-p} \|Y - X\hat{\beta}\|^2 = \frac{1}{n-p} \|R\|^2.$$

Orthogonal projections

Before discussing the statistical properties of the OLS estimator, it is useful to get a geometric understanding of what it does. By definition, the *fitted values* in the linear model are given by

$$\hat{\mu} = X\hat{\beta} = X(X^T X)^{-1} X^T Y,$$

which is a linear transformation of the original response vector Y . Let $P = X(X^T X)^{-1} X^T$, which is sometimes called the *hat matrix* in statistics literature for the obvious reason. Geometrically, the least squares problem (1.5) implies that the vector of fitted values $\hat{\mu} = PY$ is the projection of the response vector Y onto the column space of X .

The matrix P is an *orthogonal projection matrix*, meaning it is symmetric ($P^T = P$) and idempotent ($P^2 = P$). Some useful properties include:

- (i) $P = UU^T$, where columns of $U \in \mathbb{R}^{n \times p}$ form an orthonormal basis for the column space of X .
- (ii) Eigen-values of P are either 0 or 1. In consequence, $\text{tr}(P) = \text{rank}(P) = p$.
- (iii) $I - P$ is also an orthogonal projection matrix, and

$$\|Y\|^2 = \|PY\|^2 + \|(I - P)Y\|^2 = \|\hat{\mu}\|^2 + \|R\|^2.$$

The following result will be useful.

Lemma 1.8. *Let $Z \sim N(0, I_n)$ be a standard normal vector. Then $\|PZ\|^2 \sim \chi_p^2$.*

Exercise 1.9. Prove the last result.

Projection onto nested models

Consider a partition of the regressors:

$$X = (X_0 \ X_1), \ \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix},$$

where $X_0 \in \mathbb{R}^{n \times p_0}$, $X_1 \in \mathbb{R}^{n \times (p-p_0)}$, $\beta_0 \in \mathbb{R}^{p_0 \times 1}$, and $\beta_1 \in \mathbb{R}^{(p-p_0) \times 1}$. Recall that $\mu_i = \mathbb{E}(Y_i | X_i)$ and $\mu = (\mu_1, \dots, \mu_n)^T$. It is useful to think about this in terms of defining different statistical models:

- (i) The *saturated model* $\mu \in \mathbb{R}^n$;
- (ii) The *full model* $\mu \in \{X\beta : \beta \in \mathbb{R}^p\}$;
- (iii) The *submodel* $\mu \in \{X_0\beta_0 : \beta_0 \in \mathbb{R}^{p_0}\}$.

To formally define the model we have to further specify the distribution of Y . In the normal linear model, this is $Y \sim N(\mu, \sigma^2 I_n)$.

Let P (P_0) denote the projection matrix onto the column space of X (X_0). They satisfy two important properties:

- (i) $PP_0 = P_0P = P_0$; see Figure 1.1.
- (ii) $P - P_0$ is also a projection matrix.

Exercise 1.10. Prove the second property. Which subspace does $P - P_0$ project onto?

These geometric properties imply the following useful result that generalizes the Gram-Schmidt process in linear algebra.¹²

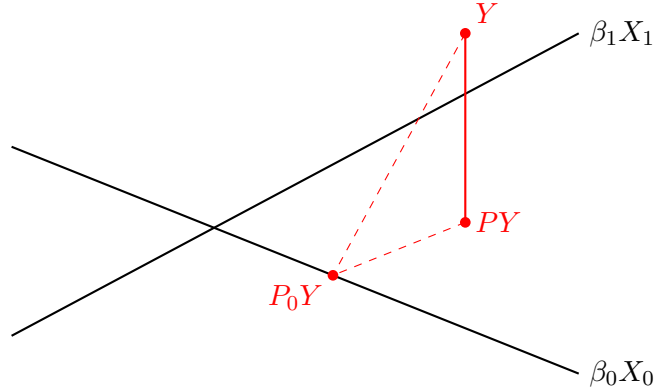


Figure 1.1 Nested model projections.

Proposition 1.11 (Partial regression characterization). *Consider the partition of X as described above and let*

$$\hat{\beta} = \begin{pmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \end{pmatrix}$$

be the corresponding partition of the OLS estimator $\hat{\beta}$. Let $\tilde{X}_1 = (I - P_0)X_1$ and $\tilde{Y} = (I - P_0)Y$ be the residuals of X_1 and Y after projecting onto the column space of X_0 . Then $\hat{\beta}_1$ is equal to the OLS estimator for a linear regression of $\tilde{Y}$ on $\tilde{X}_1$:

$$\hat{\beta}_1 = (\tilde{X}_1^T \tilde{X}_1)^{-1} \tilde{X}_1^T \tilde{Y}. \quad (1.7)$$

To prove this, consider any $X_2 \in \mathbb{R}^{n \times (n-p)}$ such that $(X_0 \ X_1 \ X_2)$ is a full-rank $n \times n$ matrix. By applying Gram-Schmidt, we obtain matrices X_0 , $\tilde{X}_1 = (P - P_0)X_1$, and $\tilde{X}_2 = (I - P)X_2$ that are orthogonal to each other. In consequence, $P - P_0 = P_{\tilde{X}_1} = \tilde{X}_1(\tilde{X}_1^T \tilde{X}_1)^{-1} \tilde{X}_1^T$ is the projection matrix onto the column space of $\tilde{X}_1$. Correspondingly, Y can be decomposed as

$$\begin{aligned} Y &= X_0 \hat{\beta}_0 + X_1 \hat{\beta}_1 + R \\ &= \underbrace{(X_0 \hat{\beta}_0 + P_0 X_1 \hat{\beta}_1)}_{P_0 Y} + \underbrace{(I - P_0) X_1 \hat{\beta}_1}_{(P - P_0) Y} + \underbrace{R}_{(I - P) Y}. \end{aligned}$$

Therefore,

$$\tilde{X}_1 \hat{\beta}_1 = (P - P_0)Y = P_{\tilde{X}_1} Y.$$

Because $\tilde{X}_1$ has full column rank, this establishes (1.7).

An important special case is $p_0 = p - 1$, where X_1 is a single regressor. In this case, some authors refer to $\hat{\beta}_1$ as the *partial regression coefficient* to distinguish from the *marginal regression coefficient* in a regression of Y on just X_1 .

Example 1.12 (Simple linear regression). When $p = 1$, the OLS estimator is given by the simple formula:

$$\hat{\beta} = \frac{X^T Y}{X^T X}.$$

When $p = 2$ and the model matrix is

$$X = \begin{pmatrix} 1 & X_1 \\ \vdots & \vdots \\ 1 & X_n \end{pmatrix},$$

the coefficient β_1 is called the *intercept* and β_2 is called the *slope*. By treating the first column of X as X_0 in the above partition, we obtain

$$\hat{\beta}_2 = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^n (X_i - \bar{X})^2},$$

where $\bar{X} = \sum_{i=1}^n X_i/n$ and $\bar{Y} = \sum_{i=1}^n Y_i/n$.

1.4.3 Exact inference for the normal linear model

Besides motivating the OLS problem (1.5) as finding the MLE of β in the normal linear model, the rest of Section 1.4.2 was entirely algebraic. Next, we discuss statistical properties of the OLS estimator and how to use it to make exact inference under the normal linear model (1.2).

Distribution of $\hat{\beta}$ and $\hat{\sigma}^2$

Since $\hat{\beta} = (X^T X)^{-1} X^T Y$ is a linear transformation of Y , it has a multivariate normal distribution conditional on X :

$$\begin{aligned} \hat{\beta} \mid X &\sim N((X^T X)^{-1} X^T \mu, (X^T X)^{-1} X^T \sigma^2 I_n X (X^T X)^{-1}) \\ &= N(\beta, \sigma^2 (X^T X)^{-1}). \end{aligned}$$

This shows that $\hat{\beta}$ is unbiased. Unbiasedness depends on linearity of μ but not the assumption of normal distribution. The Gauss-Markov theorem says that $\hat{\beta}$ is the *best linear unbiased estimator* (BLUE), that is, it has the smallest variance among all unbiased estimators of β that is linear in Y .

The estimator $\hat{\sigma}^2$ of the noise variance σ^2 can be written as

$$\hat{\sigma}^2 = \frac{1}{n-p} \|Y - X\hat{\beta}\|^2 = \frac{1}{n-p} \|(I_n - P)Y\|^2.$$

Because $I_n - P$ is also a projection matrix, this implies that

$$\hat{\sigma}^2 \mid X \sim \sigma^2 \chi_{n-p}^2 / (n-p).$$

This shows that $E(\hat{\sigma}^2) = E[E(\hat{\sigma}^2 \mid X)] = \sigma^2$ and thus $\hat{\sigma}^2$ is unbiased.¹³

Exercise 1.13. Show $\hat{\beta}$ and $\hat{\sigma}^2$ are still unbiased without the normality assumption, that is, by only assuming $E(\epsilon | X) = 0$ and $\text{Var}(\epsilon | X) = \sigma^2 I_n$.

Finally, under the normal linear model $\hat{\beta}$ and $\hat{\sigma}^2$ are independent (more precisely, conditionally independent given X) because $(X^T X)^{-1} X^T Y$ and $(I_n - P)Y$ are linear transforms of Y that is multivariate normal and

$$\text{Cov}((I_n - P)Y, (X^T X)^{-1} X^T Y | X) = (I_n - P)\sigma^2 X(X^T X)^{-1} = 0.$$

Confidence sets

We have been very careful about when X is conditioned on. This makes the statements precise but clumsy. For the rest of this section, we view X as fixed. You should know how to translate the statements if X is not fixed but conditioned on.

The key to exact inference is to find *pivotal quantities* whose distribution does not depend on unknown parameters. For example,

$$\frac{(n-p)\hat{\sigma}^2}{\sigma^2} \sim \chi_{n-p}^2 \quad (1.8)$$

is pivotal, but

$$\hat{\beta} - \beta \sim N(0, \sigma^2(X^T X)^{-1}) \quad (1.9)$$

is not pivotal because the distribution depends on σ^2 . Instead, we can use the following pivotal quantity

$$\frac{\hat{\beta} - \beta}{\hat{\sigma}} \sim \frac{N(0, (X^T X)^{-1})}{\sqrt{\chi_{n-p}^2/(n-p)}}.$$

Element-wise, we have

$$\frac{\hat{\beta}_j - \beta_j}{\hat{\sigma}} \sim \frac{N(0, (X^T X)_{jj}^{-1})}{\sqrt{\chi_{n-p}^2/(n-p)}} = \sqrt{(X^T X)_{jj}^{-1}} \cdot t_{n-p}, \quad j = 1, \dots, p. \quad (1.10)$$

By using (1.10), we can immediately construct a $(1 - \alpha)$ -confidence interval for β_j :

$$\mathcal{CI}_j(\alpha) = \left[\hat{\beta}_j - \hat{\sigma} \sqrt{(X^T X)_{jj}^{-1}} t_{n-p}(\alpha/2), \hat{\beta}_j + \hat{\sigma} \sqrt{(X^T X)_{jj}^{-1}} t_{n-p}(\alpha/2) \right],$$

where $t_{n-p}(\alpha/2)$ is the upper $(\alpha/2)$ -quantile of t_{n-p} . By $(1 - \alpha)$ -confidence interval, we mean the following probabilistic statement is true:

$$P(\beta_j \in \mathcal{CI}_j(\alpha)) = 1 - \alpha.$$

Indeed, this is true given any realization of X , and conditional coverage is stronger than unconditional coverage because $P(\beta_j \in \mathcal{CI}_j(\alpha)) = E\{P(\beta_j \in \mathcal{CI}_j(\alpha) | X)\}$ by the law of total probability.

To construct a confidence region for the p -dimensional vector β , a simple approach is to take the product of univariate confidence intervals $\prod_{j=1}^p \mathcal{CI}_j(\alpha/p)$. (*Exercise:* Show

that this set covers β with probability at least $1 - \alpha$.) However, this product set is usually quite conservative because it does not take into account the dependence between the entries of $\hat{\beta}$. A better solution is to use the following pivotal quantity

$$\frac{(\hat{\beta} - \beta)^T (X^T X) (\hat{\beta} - \beta)}{p \hat{\sigma}^2} \sim F_{p, n-p}. \quad (1.11)$$

So the following ellipsoid is a $(1 - \alpha)$ -confidence set of β :

$$\mathcal{CS}(\alpha) = \left\{ \beta \in \mathbb{R}^p \left| \frac{(\hat{\beta} - \beta)^T (X^T X) (\hat{\beta} - \beta)}{p \hat{\sigma}^2} \leq F_{p, n-p}(\alpha) \right. \right\},$$

where $F_{p, n-p}(\alpha)$ is the upper α -quantile of $F_{p, n-p}$.

Exercise 1.14. Use (1.8) to construct a $(1 - \alpha)$ -confidence interval for σ^2 .

Exercise 1.15. Let $(X^*, Y^*) \in \mathbb{R}^p \times \mathbb{R}$ be a new observation of the normal linear model. That is, suppose $Y^* = (X^*)^T \beta + \epsilon^*$ where $\epsilon^* \perp (X, \epsilon, X^*)$ and $\epsilon^* \sim N(0, \sigma^2)$. Construct a $(1 - \alpha)$ -confidence interval for $(X^*)^T \beta$ and Y^* . (The latter is called a $(1 - \alpha)$ -*prediction interval*.)

Hypothesis tests and analysis of variance

By using the duality between hypothesis testing and confidence interval, we can easily construct level- α tests for

$$H_0 : \beta_j = 0 \text{ vs. } H_1 : \beta_j \neq 0 \quad \text{and} \quad H_0 : \beta = 0 \text{ vs. } H_1 : \beta \neq 0.$$

That is, we reject $\beta_j = 0$ if $0 \notin \mathcal{CI}_j(\alpha)$ and reject $\beta = 0$ if $0 \notin \mathcal{CI}(\alpha)$.

More generally, we may be interested in comparing nested linear models. As before, consider the following partition

$$X = (X_0 \ X_1) \text{ and } \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix},$$

where $X_0 \in \mathbb{R}^{n \times p_0}$ and $\beta_0 \in \mathbb{R}^{p_0}$. We are interested in comparing the full model $\mu = X\beta$ with the submodel $\mu = X_0\beta_0$, which amounts to testing $H_0 : \beta_1 = 0$ vs. $H_1 : \beta_1 \neq 0$. The (*generalized*) *likelihood ratio statistic* is given by

$$\frac{\sup_{\beta \in \mathbb{R}^p} L(\beta, \sigma^2)}{\sup_{\beta_0 \in \mathbb{R}^{p_0}, \beta_1 = 0} L(\beta, \sigma^2)} = \left\{ 1 + \frac{\|(P - P_0)Y\|^2}{\|(I - P)Y\|^2} \right\}^{n/2}$$

Exercise 1.16. Prove the above equality.

Thus, the likelihood ratio test rejects $H_0 : \beta_1 = 0$ if $\|(P - P_0)Y\|^2 / \|(I - P)Y\|^2$ is large. To determine the critical value, we need to derive the distribution of the test statistic under $H_0 : \beta_1 = 0$, which is given by

$$F = \frac{\|(P - P_0)Y\|^2 / (p - p_0)}{\|(I - P)Y\|^2 / (n - p)} \sim F_{p-p_0, n-p} \text{ under } H_0.$$

So the level- α likelihood ratio test rejects $H_0 : \beta_1 = 0$ when $F > F_{p-p_0, n-p}(\alpha)$.

Exercise 1.17. Show that the null distribution of the F -statistic above in the normal linear model is $F_{p-p_0, n-p}$. [Hint: Show that $\|(P - P_0)Y\|^2 = \|(P - P_0)\epsilon\|^2$, then use Theorem 1.8.]

Note that $\|(I - P)Y\|^2$ is the residual sum of squares (RSS) of the full model, while $\|(P - P_0)Y\|^2$ is the reduction of RSS when we enlarge the submodel to the full model. This ratio has obvious geometric interpretations; see Figure 1.1.

Exercise 1.18. Show that the t -test and F -test for $H_0 : \beta_j = 0$ vs. $H_0 : \beta_j = 0$ are equivalent.

1.5 Practical 2: Data and normal linear models

1.5.1 *Data manipulation

Many functions in R expect that the dataset is stored in a `data.frame` object, which is essentially a list of equal-length vectors (possibly of heterogeneous types). We will learn about a package called `data.table` which provides a fast and modern implementation of the legacy `data.frame` data structure in R. To this end, you will first need to install and load the `data.table` package.

```
install.packages("data.table")
library(data.table)
```

One important advantage of `data.table` is that it is much faster.

```
> path <- "https://raw.githubusercontent.com/Rdatatable/data.table/"
> file <- "master/vignettes/flights14.csv"
> input <- paste0(path, file)
> system.time(flights <- read.csv(input))
   user  system elapsed 
0.657   0.023   1.143 
> class(flights)
[1] "data.frame"
> system.time(flights <- fread(input))
[100%] Downloaded 2193882 bytes...
   user  system elapsed 
0.127   0.028   0.340 
> class(flights)
[1] "data.table" "data.frame"
```

The last line shows that `flights` is an object of class “data.table”, which inherits the “data.frame” class.

The `flights` dataset contains all flights that departed from New York City airports in January to October, 2014. You can get a sense about this dataset in various ways.

```
> dim(flights)
[1] 253316    11
> print(flights, 2)
   year month day dep_delay arr_delay carrier origin dest air_time
1: 2014     1   1         14         13      AA   JFK  LAX       359
2: 2014     1   1         -3         13      AA   JFK  LAX       363
---
253315: 2014    10  31         -4         15      MQ   LGA  DTW        75
253316: 2014    10  31         -5          1      MQ   LGA  SDF       110
   distance hour
1:      2475    9
```

```

      2:      2475    11
----
253315:      502    11
253316:      659     8
> summary(flights)
      year      month      day      dep_delay
Min.   :2014   Min.   : 1.000   Min.   : 1.00   Min.   : -112.00
1st Qu.:2014   1st Qu.: 3.000   1st Qu.: 8.00   1st Qu.:  -5.00
Median :2014   Median : 6.000   Median :16.00   Median :  -1.00
Mean   :2014   Mean   : 5.639   Mean   :15.89   Mean    : 12.47
3rd Qu.:2014   3rd Qu.: 8.000   3rd Qu.:23.00   3rd Qu.: 11.00
Max.   :2014   Max.   :10.000   Max.   :31.00   Max.   :1498.00
      arr_delay      carrier      origin      dest
Min.   : -112.000   Length:253316   Length:253316   Length:253316
1st Qu.: -15.000   Class :character   Class :character   Class :character
Median :  -4.000   Mode  :character   Mode  :character   Mode  :character
Mean    :   8.147
3rd Qu.: 15.000
Max.    :1494.000
      air_time      distance      hour
Min.   : 20.0   Min.   : 80   Min.   : 0.00
1st Qu.: 86.0   1st Qu.: 533   1st Qu.: 9.00
Median :134.0   Median : 944   Median :13.00
Mean   :156.7   Mean   :1099   Mean   :13.06
3rd Qu.:199.0   3rd Qu.:1416   3rd Qu.:17.00
Max.   :706.0   Max.   :4983   Max.   :24.00

```

The `data.table` class implements an elegant syntax of data manipulation. The general form is `DT[i, j, by]`, which means “take `DT`, subset/reorder rows using `i`, then calculate `j`, grouped by `by`”. For example, we can easily count the number of departure delays longer than 2 hours for each carrier in June, 2014 by

```

> flights[dep_delay > 120 & month == 6, table(carrier)]
carrier
AA  AS  B6  DL  EV  F9  FL  MQ  UA  US  VX  WN
76   2  80 109 213   2   4  24 139  26   4  63

```

We can find the five most delayed carrier-destination pairs that have at least 30 flights in total by

```

> delay_summary <- flights[, .(total = .N,
                                delayed = sum(dep_delay > 120),
                                percent = mean(dep_delay > 120),
                                mean = mean(dep_delay)),
                              by = .(carrier, dest)]
> delay_summary[total > 30][order(-percent)][1:5]

```

	carrier	dest	total	delayed	percent	mean
1:	AA	EGE	85	11	0.12941176	43.32941
2:	WN	MSY	284	28	0.09859155	25.68662
3:	EV	CHO	124	12	0.09677419	26.16129
4:	WN	CAK	365	33	0.09041096	31.77808
5:	EV	BGR	274	22	0.08029197	22.54380

This should be enough for us for now, but you are encouraged to follow the official introduction to `data.table` (<https://cran.r-project.org/web/packages/data.table/vignettes/datatable-intro.html>) to learn more about it.

1.5.2 Data visualization

For simple and quick visualization, you can use the `plot`, `hist`, and `boxplot` functions in base R. Try the following

```
x <- rnorm(100); y <- x + rnorm(100); plot(x, y)
hist(flights$air_time)
boxplot(air_time ~ carrier, flights)
```

The last command requires a bit of explanation. The `boxplot` function can take in a vector or a “formula” (such as `y ~ grp` where the numeric vector `y` is split into groups according to `grp`). Boxplot is a powerful way to visualize the distribution of a continuous variable, see Figure 1.2.¹⁴

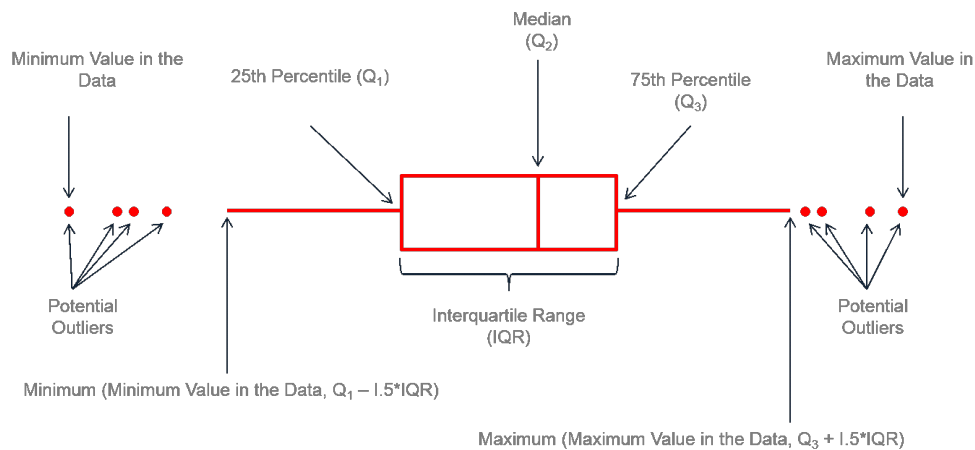


Figure 1.2 A boxplot.

The `ggplot2` package and extensions provide tools to generate production-quality figures using human-friendly syntax. We provide one example here to give you a taste; there are many amazing tutorials online and you can also easily get help from Google/Stack Overflow/LLMs.

```
> # install.packages("ggplot2") # uncomment if ggplot2 is not installed yet
```

```

> library(ggplot2)
> pl <- ggplot(delay_summary[total > 30],
               aes(x = percent, y = mean, col = carrier, size = total))
> pl <- pl + geom_point(alpha = 0.5) + theme_minimal(base_size = 15)
> pl <- pl + scale_x_continuous(labels = scales::percent)
> pl <- pl + xlab("Delay > 2 hours") + ylab("Average delay (in minutes)")
> pl

```

The output can be found in Figure 1.3.

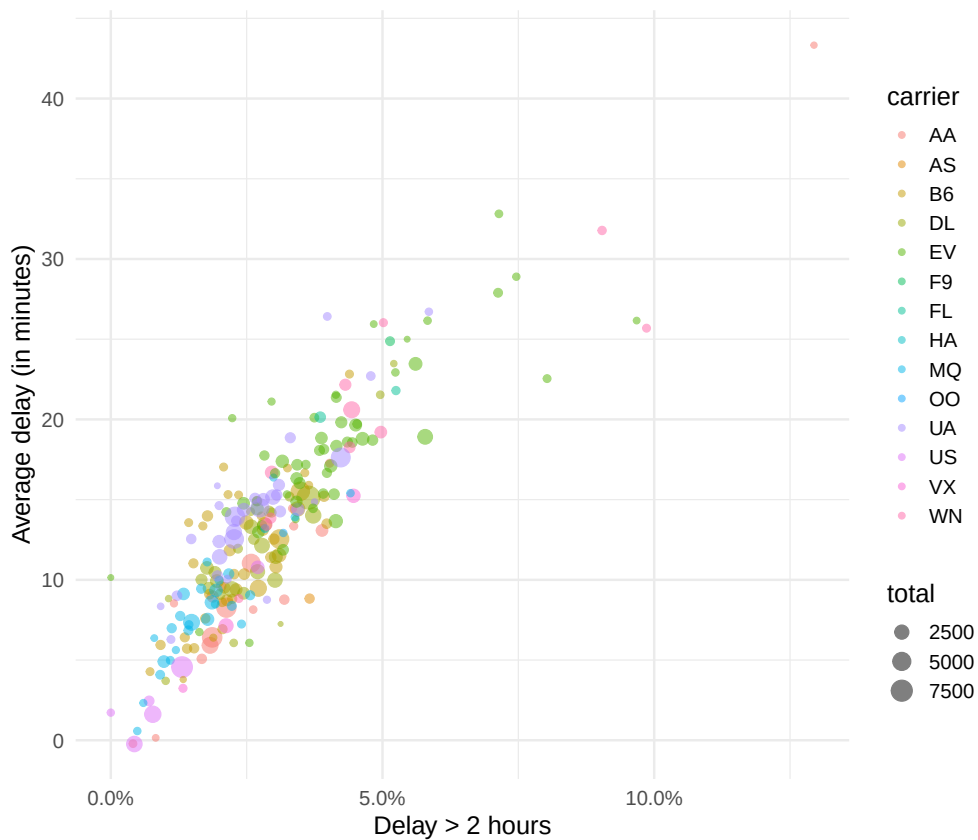


Figure 1.3 Flight delay summary.

1.5.3 Normal linear models

Let us learn about fitting normal linear models in R using some simulated data.

```

set.seed(42)
n <- 50
a <- rnorm(n)
b <- rnorm(n)
y <- a - b + 2 * rnorm(n)

```

```
data <- data.frame(y = y, x1 = a, x2 = b)
```

The main function for normal linear model in R is `lm`.

```
lm(formula, data, subset, weights, na.action,
   method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE,
   singular.ok = TRUE, contrasts = NULL, offset, ...)
```

You can directly use `lm` on the global variables `a`, `b`, `y`, but you are recommended to use the `data` argument which needs to be a `data.frame` object. The following function fits a normal linear model and outputs some important summary statistics.

```
> fit1 <- lm(y ~ x1 + x2, data)
> summary(fit1)
```

Call:

```
lm(formula = y ~ x1 + x2, data = data)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-3.4578	-0.8958	-0.1260	0.5328	6.1281

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.3005	0.2680	-1.122	0.2677
x1	1.1764	0.2406	4.888	1.23e-05 ***
x2	-0.9570	0.2996	-3.194	0.0025 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.883 on 47 degrees of freedom

Multiple R-squared: 0.484, Adjusted R-squared: 0.462

F-statistic: 22.04 on 2 and 47 DF, p-value: 1.767e-07

The `summary` function is very useful and adapts to the object class. (When applying the `summary` method to `fit1` which is a `lm` object, R automatically calls `summary.lm`; try `?summary.lm`.)

Here are what the numbers in the output mean:

- Below **Residuals**: are the five-number summary of the residuals R .
- The **Estimate** column contains the OLS estimator $\hat{\beta}_j, j = 1, \dots, p$.
- The **Std. Error** column contains the estimated standard errors $\hat{\sigma} \sqrt{(X^T X)^{-1}_{jj}}$.
- The **t value** column contains the t-statistics, which is simply the ratio between the previous columns.

- The `Pr(>|t|)` column contains the p-values for the two-sided t-tests.
- The `Residual standard error` is simply $\hat{\sigma}$, with degrees of freedom $n - p$.
- The `Multiple R-squared` refers to the so-called “coefficient of determination”, which is defined as the variance explained by the regressors

$$R^2 = \frac{\|\hat{\mu} - \bar{Y}1\|^2}{\|Y - \bar{Y}1\|^2} = 1 - \frac{\|Y - \hat{\mu}\|^2}{\|Y - \bar{Y}1\|^2},$$

where $\bar{Y} = \sum_{i=1}^n Y_i/n$. The `Adjusted R-squared` is a less biased estimator of the population R^2 and is defined as (you don’t need to remember this)

$$R_{\text{adj}}^2 = 1 - (1 - R^2) \frac{n - 1}{n - p - 1}.$$

- The `F-statistic` is the F -statistic with respect to the submodel in which the coefficients of all non-intercept terms are 0, which has degrees of freedom $p - 1$ and $n - p$. The `p-value` is the p-value for the corresponding F -test.

You can use `anova` for analysis of variance. Here is a simple example.

```
> fit2 <- lm(y ~ x1, data)
> anova(fit2, fit1)
Analysis of Variance Table

Model 1: y ~ x1
Model 2: y ~ x1 + x2
  Res.Df    RSS Df Sum of Sq    F    Pr(>F)
1      48 202.93
2      47 166.73  1      36.2 10.205  0.0025 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

1.5.4 Exercises

Exercise 1.19. Explain why the p-value in the `anova` output (0.002502) is exactly the same as the p-value in the `summary.lm` output for `x2`.

Exercise 1.20. You can fit an intercept-only model using `fit3 <- lm(y ~ 1, data)`. Without executing `anova(fit3, fit1)`, can you guess what its output will be based on the output `summary.lm`?

Exercise 1.21. Use R to validate Theorem 1.11.

Exercise 1.22. Without calling `lm` or using an LLM,¹⁵ write a function called `mylm` with arguments `X` ($n \times p$ model matrix) and `Y` (n -vector) that outputs all the numbers reported by `summary.lm`. Test your code using the example above.

Exercise 1.23. Modify your `mylm` function so that it also accepts a logical vector `S0` of length p and performs the analysis of variance test for the sub-model that only uses the regressors `X0 <- X[, S0]`. Compare your output with `anova`.

1.6 Basic asymptotic statistics

The theory of normal linear model is beautiful but relies quite heavily on properties of the multivariate normal distribution. We will introduce many relaxations of the normal linear model in this course. In general, we cannot obtain exact inference for parameters in those models, but it is often possible to obtain approximate inference when the sample size n is large. As the focus of this course is on statistical modelling instead of statistical theory, we will only state the relevant results in asymptotic statistics (the proofs are covered in *Principles of Statistics*) and discuss how to use them in statistical modelling.

Asymptotic statistics (or large-sample theory) is primarily based on two fundamental results in probability theory: the law of large numbers and the central limit theorem. We will state the versions in IA *Probability*. Let $X_1, X_2, \dots, X_n, \dots$ be a sequence of i.i.d. random variables and let $\bar{X}_n = (X_1 + \dots + X_n)/n$. The *weak law of large numbers* says that if $\mu = E(X_1)$ exists, then as $n \rightarrow \infty$,

$$\bar{X}_n \xrightarrow{p} \mu,$$

where $\xrightarrow{p}$ means “convergence in probability”, that is, $P(|\bar{X}_n - \mu| > \epsilon) \rightarrow 0$ for all $\epsilon > 0$. The *central limit theorem* says that if in addition $\sigma^2 = \text{Var}(X_1)$ exists, then as $n \rightarrow \infty$,

$$\frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma} \xrightarrow{d} N(0, 1),$$

where $\xrightarrow{d}$ means “convergence in distribution”, that is, $P(\sqrt{n}(\bar{X}_n - \mu)/\sigma \leq z) \rightarrow \Phi(z)$ for all $z \in \mathbb{R}$ where Φ is the CDF of $N(0, 1)$. A less precise but more intuitive way to state this is $\bar{X}_n \sim N(\mu, \sigma^2/n)$ where $\sim$ means “approximate distribution”.

One of the most important results in asymptotic statistics is the following.

Theorem 1.24 (Informal). *If $\mathbb{P}$ is a “regular” parametric model and the data $X_1, \dots, X_n$ are generated i.i.d. from P_θ , then the maximum likelihood estimator*

(i) exists;

(ii) is consistent in the sense that $\hat{\theta}_{MLE} \rightarrow \theta$ in probability as $n \rightarrow \infty$;

(iii) is asymptotically normal in the sense that

$$\sqrt{n}(\hat{\theta}_{MLE} - \theta) \xrightarrow{d} N(0, \{I(\theta)\}^{-1}),$$

where $I(\theta) = \text{Var}_\theta(\nabla_\theta l(\theta))$ is called the Fisher information matrix and $l(\theta) = \log L(\theta)$ is the log-likelihood function of a single observation X_1 . Moreover, $I(\theta)^{-1}$ is the “best possible asymptotic variance” for any “regular” estimator of θ .

There is a fascinating history about how this result was conceived, formalized, and proved.¹⁶ We will not attempt to prove it, but our investigation of exponential family in Chapter 3 will provide some insights about this deep result.

The next two lemmas about convergence of random variables are often useful.

Lemma 1.25 (Slutsky's lemma). *Suppose $A_n \xrightarrow{d} A$ and $B_n \xrightarrow{p} b$ where b is a constant. Then*

- (i) $A_n + B_n \xrightarrow{d} A + b$;
- (ii) $A_n B_n \xrightarrow{d} Ab$;
- (iii) $A_n/B_n \xrightarrow{d} A/b$ if $b \neq 0$.

This can be easily extended to multivariate settings. This result is a corollary of the *continuous mapping theorem*, which says if g is a continuous function, then $X_n \rightarrow X$ implies $g(X_n) \rightarrow g(X)$ where $\rightarrow$ can be almost sure convergence, convergence in probability, or convergence in distribution (but the two $\rightarrow$ need to be the same).

Lemma 1.26 (Delta method). *Suppose*

$$\sqrt{n}(\hat{\eta} - \eta) \xrightarrow{d} N(0, \tau^2),$$

where τ^2 may depend on η and $g(\eta)$ is continuously differentiable at η . Then

$$\sqrt{n}\{g(\hat{\eta}) - g(\eta)\} \xrightarrow{d} N(0, \tau^2 g'(\eta)^2).$$

This result can be proved by considering the Taylor expansion of $g(\hat{\eta})$ at η and applying Slutsky's lemma. It can be easily extended to multivariate settings.

For this course, you only need to know how to apply Slutsky's lemma and the delta method. An immediate application of Slutsky's lemma at this point is that

$$\sqrt{n}I(\hat{\theta})^{1/2}(\hat{\theta}_{\text{MLE}} - \theta) \xrightarrow{d} N(0, I).$$

This allows us to use

$$\hat{\theta}_{\text{MLE}} \sim N(\theta, \{I(\hat{\theta})\}^{-1}/n)$$

to construct asymptotic tests and confidence intervals. For example, a (pointwise¹⁷) asymptotic $(1 - \alpha)$ -confidence interval for θ_j is

$$\mathcal{CI}_j(\alpha) = \left[\hat{\theta}_j - \hat{\sigma} \sqrt{(X^T X)_{jj}^{-1}} z(\alpha/2), \hat{\theta}_j + \hat{\sigma} \sqrt{(X^T X)_{jj}^{-1}} z(\alpha/2) \right],$$

where $z(\alpha/2)$ is the upper $(\alpha/2)$ -quantile of $N(0, 1)$, that is,

$$\lim_{n \rightarrow \infty} P_{\theta}(\theta_j \in \mathcal{CI}_j(\alpha)) \rightarrow 1 - \alpha \quad \text{as } n \rightarrow \infty \text{ for all } \theta \in \Theta. \quad (1.12)$$

The last useful asymptotic result for us is Wilks' theorem.

Theorem 1.27 (Informal). *Consider testing $H_0 : \theta \in \Theta_0$ vs. $H_1 : \theta \in \Theta \setminus \Theta_0$ in parametric models using the generalized likelihood ratio statistic Λ in (1.1). Under regularity conditions (smoothness of the likelihood function), if $\Theta_0 \subset \Theta$ are nested linear spaces, then*

$$2 \log \Lambda \xrightarrow{d} \chi^2_{\dim(\Theta) - \dim(\Theta_0)} \quad \text{as } n \rightarrow \infty,$$

under P_θ for any $\theta \in \Theta_0$.

Notes

¹Perhaps this is so obvious because for most people mathematics and statistics are both about “numbers”. When I told my neighbours that I am a statistician, most of their first reaction is that I do mathematics.

²Mathematics also involves induction, see G. Pólya’s book *Mathematics and Plausible Reasoning*, but mathematical induction is a deductive method. Statistics also involves deductive reasoning (which is basically mathematical statistics).

³Box, G. E. P. (1957). Abstracts. *Biometrics*, 13(2), 238–246.

⁴Breiman, L. (2001). Statistical modeling: The two cultures. *Statistical Science*, 16(3), 199–231.

⁵The notation here has the risk of being eccentric because many authors use $\mathbb{P}$ to denote a probability measure. The problem with that is it encourages one to think the probability measure as something fixed, which is very dangerous for statisticians. The blackboard bold font is commonly used to denote “special sets”, such as $\mathbb{R}$, $\mathbb{C}$, etc. In statistics, the special set is often the statistical model—a collection of probability distributions. So in this sense it is appropriate to denote a statistical model by $\mathbb{P}$, and it is even better when someone is confused about what $\mathbb{P}$ means because that forces them to think about what model is being used.

⁶I received complaints about also using P to denote CDF in IB *Statistics*, so I am switching to the conventional notation in this course.

⁷In IA *Probability*, expectation is defined as a unique linear functional on random variables on a probability space that satisfies certain properties. Some people actually prefer to define probability via expectation, with good reasons; see Whittle, P. (2000). *Probability via expectation*. Springer. De Finetti would even use P to also denote the expectation operator. We will not be a zealot about this.

⁸It is possible to define multivariate normal with a degenerate covariance matrix; see Rao, C. R. (1973). *Linear statistical inference and its applications* (2nd ed.). Wiley, Chapter 8.

⁹Named after the statistician W. S. Gosset who used the pseudonym “Student” to publish his method.

¹⁰Equation (1.2) does not necessarily describe the causal relationship between X_i and Y_i . That is, if an external force sets X_i to x_i (instead of its “natural” value), (1.2) does not make any assumptions on what the resulting Y_i would become. In contrast, a linear structural equation model assumes that the counterfactual value of Y_i , often denoted as $Y_i^{(x_i)}$, is $x_i^T \beta + \epsilon_i$. What has confused generations of statisticians and scientists is that many people also use (1.2) to indicate a linear structural equation model. For more discussion on the distinction between regression and causation, see Section 6.4 of Freedman’s book.

¹¹This is why some texts assume X is “fixed”. A better way to think about this is that the inference for the normal linear model (1.2) is conditional on the model matrix X . This is an instance of the *conditionality principle*, which says that the unconditional distribution and the conditional distribution given an ancillary statistic (X in this case) carry the same information for statistical inference.

¹²In econometrics, this result is known as the Frisch–Waugh–Lovell theorem.

¹³If we are more rigorous, we should write the estimator of σ^2 as $\widehat{\sigma^2}$ instead of $\hat{\sigma}^2$. But it is widely understood that we estimate σ by first estimating σ^2 and $\hat{\sigma}^2$ does not mean “ $\hat{\sigma}$ square”. Notice that unbiasedness of $\hat{\sigma}^2$ does not translate to unbiasedness of $\sqrt{\hat{\sigma}^2}$ as an estimator of σ .

¹⁴Image source: <https://leansigmacorporation.com/box-plot-with-minitab/>.

¹⁵Do we still need to learn programming with the rise of LLMs? I asked the same question to an AI and got the following code <https://g.co/gemini/share/add2a2a9cad6>. There are at least two substantial mistakes in the examples that accompany the function written by the AI. Can you find them?

¹⁶Stigler, S. M. (2007). The epic story of maximum likelihood. *Statistical Science*, 22, 598–620.

¹⁷Results like (1.12) are sometimes unsatisfactory in practice because it does not tell us how large the sample size n needs to be for the coverage probability to be sufficiently close to $1 - \alpha$. Moreover, (1.12) does not exclude the possibility that the confidence interval may have bad coverage probability (e.g. less than $1 - 2\alpha$) at some θ for any sample size n . This is why some people prefer *uniform* asymptotic confidence sets S that satisfy

$$\liminf_{n \rightarrow \infty} \inf_{\theta \in \Theta} \mathbb{P}_\theta(\theta \in S) \geq 1 - \alpha \quad \text{as } n \rightarrow \infty.$$