# Discrete Choice Model Estimation with R

Dan Yavorsky

# Table of contents

# Welcome

⚠ Warning

This is an early draft. The book is incomplete. Every aspect of it is subject to change. Many things might be incorrect. Are you sure you want to read this yet?

> You don't understand it, until you have coded it.

It's a mantra that my mentors instilled in me in graduate school, and one that I propagate on my students today. I believe it in deeply. If you want to understand a statistical model, it is insufficient to interact with the model only through pen and paper. No matter how well you organize the subscripts or how masterfully you interweave elements of the Greek and Roman alphabets, you won't really "get it." To deeply understand a statistical model, you need to be able to simulate data from the model, and then take that simulated data to an estimation routine that enables you to recover the parameters of interest.

This is something you learn to do in graduate school. But, unless you have one of the most caring and pedagogical advisers, no one teaches it to you! Instead, it's a skill that students develop independently and inefficiently between classes and assignment due dates or, in my case, after I was knee-deep in my dissertation research. There is a tremendous imbalance between how immensely important this skill is and the time and instruction dedicated to it.[1] That imbalance is the motivation for this book.

I address this gap in the specific domain of Discrete Choice Modeling. I do so alongside both Kenneth Train's masterful text *Discrete Choice Methods with Simulation* (Second Edition, 2009) freely available online at https://eml.berkeley.edu/books/choice2.html and the underappreciated tome *Applied Choice Analysis* by David Hensher, John Rose, and William Greene (Second Edition, 2015). These books are properly cited as Train (2009) and Hensher, Rose, and Greene (2015) but will hereafter be referred to as DCMS and ACA. I will reference them throughout.

Train and I align in our thinking:

[1] In this context, a *closed-form expression* means a way of writing the integral so that the anti-derivative sign is not part of solution. For example, the integral $\int x\,dx$ has the closed for expression $x^2/2$ plus some constant. We will see later that the Extreme Value distribution is often chosen for $f$ predominantly because it leads to a closed for expression for the choice probability $p(y|x)$.

> "…the true value of [] choice modeling is the ability to create tailor-made models. The computation and programming steps that are needed to implement a new model are usually not difficult. An important goal [] is to teach these skills as an integral part of the exposition of the models themselves."[2]

However, what is "usually not difficult" for an experienced researcher can be immensely difficult for a typical graduate student. The aim of this book is to ease that difficulty. I demonstrate R code alongside the math. You will see the mathematical derivations and pseudo-code from DCMS and ACA transformed into working R code. No mysteries will remain. To borrow a quote from Quintilian, Cobbett, Cooke, and many others: my goal is to take you through the process of programming the simulation and estimation of common discrete choice models *not so that you can understand, but so that you cannot possibly misunderstand.*

## How to read this book

The active reader will benefit most from securing copies of DCMS and ACA alongside this book. While I introduce each model before diving into the associated programming, DCMS and ACA provide substantially more context regarding their development and use in economics and psychology. They also offer more thorough descriptions and derivations of the properties of the models. My recommendation is that you should read — or most likely, re-read — the referenced chapters from DCMS and ACA as we begin to explore each specific model, and then read and work through the corresponding chapter here, alternating between the texts. Do not simply read this book from start to finish without frequently returning to DCMS and/or ACA. Doing so only deprives you of the intended experience and likely dramatically reduces the amount you will learn from the process.

In addition, I strongly encourage you write or copy the code as you work though this book. I would mandate this if I could, but I'm not sitting there next to you and so I must settle for simply sharing my encouragement. Write the code. Play with the code. Break the code. Make it your own. That behavior is where deep understanding comes from, not from highlighting the occasional sentence or equation you find important.

## Intended Audience

This book is intended for a narrow audience, predominantly current or former graduate students with an interest in discrete choice modelling who will find value from seeing and interacting with the programmatic implementation of the multinomial logit and its extended family of related models. In other words, someone who read DCMS and thought *how would I code that?*

[2] More precisely, $p(y=2|x) = \Pr(x+\varepsilon > 1|x=0.5) = \Pr(\varepsilon > 0.5) = \int_{0.5}^{1} f(\varepsilon)d\varepsilon = (0.5\varepsilon)|_{0.5}^{1} = 0.25$.

Or someone who used nlogit while working through the second half of ACA and wants to "see under the hood" of its estimation routines.

We will simulate data from the statistical models and then estimate the parameters of those models from the simulated data. Doing so will require us to generate random draws from distributions, approximate integrals with monte carlo simulation, maximize likelihood functions with optimization routines, and (when taking a Bayesian approach) explore posterior distributions with Markov Chain Monte Carlo methods.

We will do all of this in R, a freely available software environment for statistical computing and graphics. As a result, I assume you are reasonably familiar with R. If not, there is a tremendous set of free online R resources collected and organized at https://www.bigbookofr.com/. Popular books include *R for Data Science* properly cited as Wickham, Cetinka-Rundel, and Grolemund (2023) and *Advanced R* properly cited as Wickham (2019). If you are new to R or your familiarity with it has waned, you should consult these resources before proceeding here.

I also assume you have taken introductory statistics or econometrics courses, as the concepts and techniques taught there are foundational for understanding and estimating discrete choice models. In particular, you should have no uncertainty about the difference between a model, an estimator, and an estimate. To briefly review:

- a *model* is the set of mathematical assumptions about how data are generated,
- an *estimator* (or equivalently, the estimation routine) is an algorithm or function of the data, and
- an *estimate* is the result of applying the estimator to a particular dataset.

In my experience, students are often given a model and an estimator, after which much time in the classroom is spent deriving properties of the estimator for that particular model. Often a homework assignment follows which asks students to implement the estimator on a dataset to find an estimate. While this approach succeeds in developing the student's skills and understanding of the particular estimator for the particular model, it does so with blinders on, obscurring consideration about the choice or specification of the model as well as the choice of estimator.

I'm sure you can recall your introduction to linear regression where you were handed a model with a linear conditional mean and a Normally distributed "error" and told that you would estimate the parameters of the model via least squares. Hopefully this left you wondering about less-particular situations such as those with a non-continous outcome variable, a non-Normally distributed error term, a non-linear conditional mean, correlation between the observed covariates and the error, a causal interpretation, or a different estimation approach.

In this book, we will focus on models with non-continous outcome variables and we will use Maximum Likelihood and Bayesian methods to estimate the parameters of those models. If you would like to review key ideas from statistics and econometrics in preparation for tackling our discrete choice models with likelihood and Bayeian approaches, I highly recommend Abramovich and Ritov (2023, link) on the topic of mathematical statistics, Goldberger (1991, link) and

Kennedy (2008, link) for econometrics, and McElreath (2018, link) and Gelman et al. (2013, link) for Bayesian statistics. Bruce Hansen's recently released two-volume series Hansen (2022b, link) and Hansen (2022a, link) on probability, statistics, and econometrics are an excellent set of alternative resources.

## Acknowledgments

I gratefully aknowledge the academic giants who pioneered and developed the field of Discrete Choice Analysis. The models offer flexible forms to study human behavior, the estimation methods are fascinating, their implementation usually proceeds from vexing to fun, and their application has been immensely useful in my consulting practice where I seek to help firms better understand consumers in order to improve consumer well-being and firm profitability.

I am also immensely grateful to the teams that work on the R-Project, those at Posit who provide RStudio, Positron, and Quarto, and the related communities of developers, academics, and R users. The free tools they provide are exceptional and I use them daily. The welcoming communities they have established provide the necessary support humans need, be it technical troubleshooting or emotional encouragement.

More personally, I would like to thank my academic mentors Ella Honka, Peter Rossi, and Eric Bradlow, as well as the folks that have or continue to tolerate me through my consulting work, including Prachi Bhalerao, June Wu, Chris Diener, Keith Chrzan, and the hosts and attendees of Sawtooth Software's annual Analytics and Insights Summit. I have learned so much from these researcher's guidance and often their work.

Special thanks go to generous people who reviewed and assisted with drafts of this book, including Darren Aeillo, Kalyan Rallabandi, and Geoff Zheng.

Unquestionably, my deepest thanks go to Alison, Zach, and Ben for their patience and support.

## Colophon

An online version of this book is available at https://dcme-r.danyavorsky.com. The source of the book is available at https://github.com/dyavorsky/dcme-r. The book is authored using Quarto, an open-source scientific and technical publishing system that makes it easy to create articles, presentations, websites, books, and other publications that combine text and executable code.

# Part I

# Introduction

# 1 Introduction

It is instructive to start with the first sentence of Train (2009) Section 1.3, “Discrete choice analysis consists of two interrelated tasks: **specification** of the behavioral model and **estimation** of the parameters of that model” (emphasis added). In particular, chapters 1–7 of Train (2009) only specify models; there is not even a hint of model estimation. And since I follow Train (2009), we too begin with a focus on model specification.

Let me be clear: many students will see $y$ and $x$ later in this section and immediate think about “fitting” a model; that is, they assume they have data that they will put into an estimation routine to find estimates of the parameters of the model. We are not there yet! At this early stage in the book, we are only specifying models; that is, listing sets of assumptions about data generating processes and exploring the implications of those assumptions. There are no data yet. There will be parameters introduced in our choice of model specification, but we are not yet estimating those parameters. Have patience, we will eventually do these things.

## 1.1 Recap of Train Ch. 1

Train denotes the outcome in any given choice situation as $y$, determined by some observable factors collected in the vector $\mathbf{x}$ and some unobservable factors collected in the vector $\boldsymbol{\varepsilon}$. The factors ($\mathbf{x}$ and $\boldsymbol{\varepsilon}$) relate to the agent’s choice ($y$) through a function $y = h(\mathbf{x}, \boldsymbol{\varepsilon})$. We assume for the moment that we know $h(\cdot)$ and that $\mathbf{x}$ and $\boldsymbol{\varepsilon}$ are length-one vectors (i.e., scalars) denoted $x$ and $\varepsilon$.

Since we do not observe $\varepsilon$, we can’t predict $y$ exactly. Instead, we focus on the probability of $y$, that is:

$$\begin{aligned} p(y|x) &= \Pr\left(\varepsilon \text{ such that } h(x,\varepsilon) = y\right) \\ &= \Pr\left(I\left[h(x,\varepsilon) = y\right] = 1\right) \\ &= \int I\left[h(x,\varepsilon) = y\right] f(\varepsilon)\, d\varepsilon \end{aligned} \tag{1.1}$$

For certain special choices of $h$ and $f$, a closed-form expression[1] for the integral is available. But more generally, for almost any choice of $h$ and $f$, we can approximate the integral through simulation. Train provides pseudo code on how to do so:

1. Repeat the following two steps many ($r = 1, \dots, R$) times:
   - Draw $\varepsilon^{(r)}$ from $f(\varepsilon)$.
   - Determine whether $h(x, \varepsilon^{(r)}) = y$. If so, set $I^{(r)} = 1$; else set $I^{(r)} = 0$.
2. Average the $R$ values of $I$

Next we look at two examples where we use this procedure to approximate the $p(y|x)$ integral. The first example I made up. The second example is the binary logit model discussed by Train, but for which I'll show you how to simulate the $p(y|x)$ integral.

## 1.2 A Simple Example

Let's first set up a toy example to demonstrate how simulation can approximate the $p(y|x)$ integral. Suppose $x = 0.5$ and $\varepsilon$ is uniformly distributed between -1 and 1. Define $h(x, \varepsilon)$ to be:

$$h(x,\varepsilon) = \begin{cases} 0 & \text{if } x + \varepsilon < 0 \\ 1 & \text{if } x + \varepsilon \in [0, 1] \\ 2 & \text{if } x + \varepsilon > 1 \end{cases} \tag{1.2}$$

We'll focus on the outcome $y = 2$. You can probably intuit that the $p(y = 2|x) = 0.25$ since only one quarter of the time will $\varepsilon$ be sufficiently positive to make $x + \varepsilon > 1$.[2] Nevertheless, let's approximate the integral representation of $p(y = 2|x)$ through simulation to ensure we understand the process.

To walk you through the code, we first set a seed so that the pseudo-random numbers generated by `runif()` can be replicated exactly each time the code is run (even on different computers). We then specify that we will use 1,000 draws in the simulation and we create a vector `I` to hold our results. The simulation occurs via a `for()` loop where each time through the loop we take a draw of $\varepsilon$, calculate $0.5 + \varepsilon$ and check whether that sum is greater than one. If so, then $h(x, \varepsilon) = 2$ matching the value of $y$ for the choice probability we want to assess — i.e., $p(y = 2|x)$ — and thus we store a 1 in the $r^{\text{th}}$ position of `I`; otherwise we store a 0. We then average the values in `I` to get our approximation of $p(y = 2|x)$.

[1] In this context, a *closed-form expression* means a way of writing the integral so that the anti-derivative sign is not part of solution. For example, the integral $\int x \, dx$ has the closed for expression $x^2/2$ plus some constant. We will see later that the Extreme Value distribution is often chosen for $f$ predominantly because it leads to a closed for expression for the choice probability $p(y|x)$.

[2] More precisely, $p(y = 2|x) = \Pr(x + \varepsilon > 1|x = 0.5) = \Pr(\varepsilon > 0.5) = \int_{0.5}^{1} f(\varepsilon)d\varepsilon = (0.5\varepsilon)|_{0.5}^{1} = 0.25$.

```
set.seed(1234)

R <- 1000
I <- vector(length=R)

for(r in 1:R) {
    eps  <- runif(1, min=-1, max=1)
    h    <- 0.5 + eps
    I[r] <- as.integer(h > 1)
}
mean(I)
```

```
[1] 0.258
```

The simulated value 0.258 approximates the exact value 0.25 and can be made closer by increasing the number of draws used in the simulation.

Notice that we took draws of $\varepsilon$ to empirical approximate the integral of $p(y|x)$, but we should not think of these draws as "data" in the sense of a dataset. They are ancillary numbers generated from the distribution of $\varepsilon$ that we use to approximate the integral.

R users will recognize that we can shorten the code by taking advantage of R's vectorized functions and its conversion of boolean values to 0/1 when used in mathematical operations. Here is a shorter implementation of the simulation; whether it's "better" code is a matter of preference.

```
set.seed(1234)
R <- 1000
mean( runif(R, min=-1, max=1) + 0.5 > 1 )
```

```
[1] 0.258
```

That's it. If you can generate pseudo-random draws from the density $f$ and you know $h$, approximating a choice probability by simulation might only require a handful of lines of code.

## 1.3 A Binary Logit Model Example

As an example of a model with a complete closed-form solution, Train provides the binary logit model. In this example, we'll let $\mathbf{x}$ and $\boldsymbol{\varepsilon}$ be vectors of length two, and we'll specify $h$ and $f$ as follows:

The "binary" part refers to the aspect of the model whereby the decision maker does one of two things; they either take an action ($y = 1$) or not ($y = 0$). To tie this model into a framework of behavior, we start with a utility function $U$. In Train's specific example, utility is specified as

$$U(\mathbf{x}, \boldsymbol{\beta}, \varepsilon) = \mathbf{x}'\boldsymbol{\beta} + \varepsilon \tag{1.3}$$

where $\mathbf{x}$ is a vector of observable explanatory variables, $\boldsymbol{\beta}$ is a vector of parameters that through the functional form $\mathbf{x}'\boldsymbol{\beta}$ effectively serve as weights on the covariates, and $\varepsilon$ is a scalar index collecting the value of information used by the decision maker but unobserved to the researcher. Notice that we're allowing $\mathbf{x}$ and $\boldsymbol{\varepsilon}$ to be vectors in this example, whereas they were scalars (or equivalently length-one vectors) in the previous example.

In this model, the threshold for action is 0 because the decision maker takes action ($y = 1$) when utility is positive; conversely, when utility is negative, the decision makes elects not to take the action (i.e., $y = 0$). Therefore we can specify $h$ as:

$$h(\mathbf{x}, \boldsymbol{\beta}, \varepsilon) = \begin{cases} 0 & \text{if } \ U(\mathbf{x}, \boldsymbol{\beta}, \varepsilon) \leq 0 \\ 1 & \text{if } \ U(\mathbf{x}, \boldsymbol{\beta}, \varepsilon) > 0 \end{cases} \tag{1.4}$$

The "logit" part of the model's name refers to the choice of $f$. The binary logit model assumes $f$ is the logistic distribution:

$$f(\varepsilon) = \frac{e^{-\varepsilon}}{(1 + e^{-\varepsilon})^2} \tag{1.5}$$

Having specified $h$ and $f$, let's choose some values for $\mathbf{x}$ and $\boldsymbol{\beta}$ and use simulation to approximate the integral for $p(y|\mathbf{x}, \boldsymbol{\beta})$.[3] Let's pick $\mathbf{x} = (0.5, 2)$ and $\boldsymbol{\beta} = (3, -1)$ such that $\mathbf{x}'\boldsymbol{\beta} = (0.5)(3) + (2)(-1) = 1.5 - 2 = -0.5$. We know from the closed-form solution to this model provided by Train that, with these values of $\mathbf{x}$ and $\boldsymbol{\beta}$, the probability the decision maker takes action is:

$$p(y = 1|\mathbf{x}, \boldsymbol{\beta}) = \frac{e^{\mathbf{x}'\boldsymbol{\beta}}}{1 + e^{\mathbf{x}'\boldsymbol{\beta}}} = \frac{e^{-0.5}}{1 + e^{-0.5}} = 0.3775407 \tag{1.6}$$

We can approximate this integral as before. Below I use the function `rlogis()` to take `R=1000` draws from the binary logistic distribution, and I approximate the integral with the proportion of times $\mathbf{x}'\boldsymbol{\beta} + \varepsilon$ is greater than the threshold for action (0):

[3] Only at the early stage of simulating data from a model to better understand the model do we pick values for $\mathbf{x}$ and $\boldsymbol{\beta}$. Typically, $\mathbf{x}$ will part of the data you collect and $\boldsymbol{\beta}$ will be parameters whose values you seek to estimate.

```
set.seed(2345)
R <- 1000

x    <- c(0.5, 2)
beta <- c(3, -1)

U <- as.vector(x %*% beta) + rlogis(R)
mean(U > 0)
```

```
[1] 0.364
```

Our simulated value 0.364 approximates the exact value of the integral 0.378.

## 1.4 Key Learnings

The key learning from this chapter is that with discrete choice models our focus is on the *probability* of the choie outcome $y$ – that is, $p(y|x)$. This choice outcome $y$ results from the joint distribution $f$ of unobserved factors $\boldsymbol{\varepsilon}$ and the behavioral model $h$ that relates $y$ to $\mathbf{x}$ (and $\boldsymbol{\varepsilon}$). The probability of the choice outcome $p(y|x)$ can be written in closed form for only very special choices of $f$ and $h$, but for almost any choice of $f$ and $h$ we can simulate $p(y|x)$, as the two examples in this chapter demonstrate.

# Part II

# Logit

# 2 Drawing from Densities

## 2.1 Pseudo-Random Numbers from Uniform Distribution

## 2.2 Inverse Transform Method

$$
\begin{aligned}
F_X(x) &= \Pr\left(X \leq x\right) \\
&= \Pr\left(T(U) \leq x\right) \\
&= \Pr\left(U \leq T^{-1}(x)\right) \\
&= T^{-1}(x)
\end{aligned}
$$

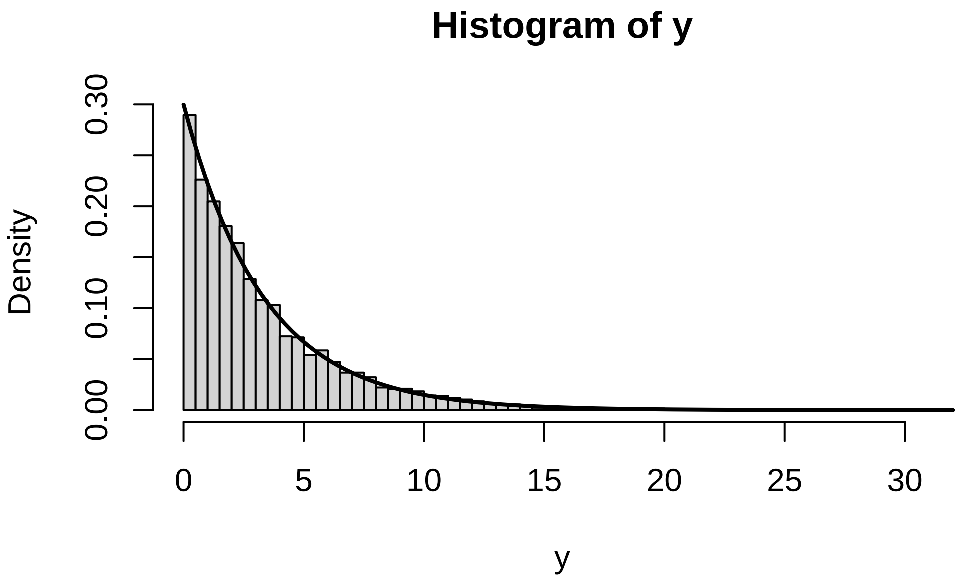
## 2.3 Example: Exponential

$$F(y, \lambda) = 1 - \exp\left(-\lambda y\right) \quad \text{for } y \geq 0$$

$$F^{-1}(u) = -\log(1 - u)/\lambda$$

```r
n <- 1e4
lambda <- 0.3
u <- runif(n)
y <- -log(1-u)/lambda
```

```r
x <- seq(from=0, to=ceiling(max(y)), length.out=100)
hist(y, freq=F, breaks=100)
lines(x=x, y=dexp(x,lambda), lwd=2)
```



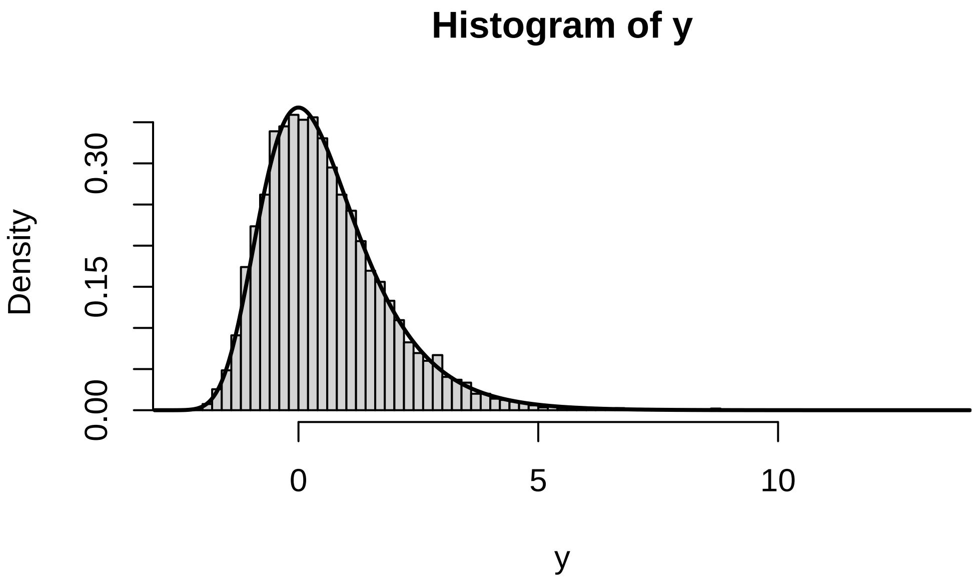
## 2.4 Example: Gumbel

$$F(y) = \exp\left(-\exp\left(-y\right)\right)$$

$$F^{-1}(u) = -\log(-log(u))$$

```
n <- 1e4
u <- runif(n)
y <- -log(-log(u))
```

```
x <- seq(from=-3, to=ceiling(max(y)), length.out=500)
hist(y, freq=F, breaks=100)
lines(x=x, y=ordinal::dgumbel(x), lwd=2)
```



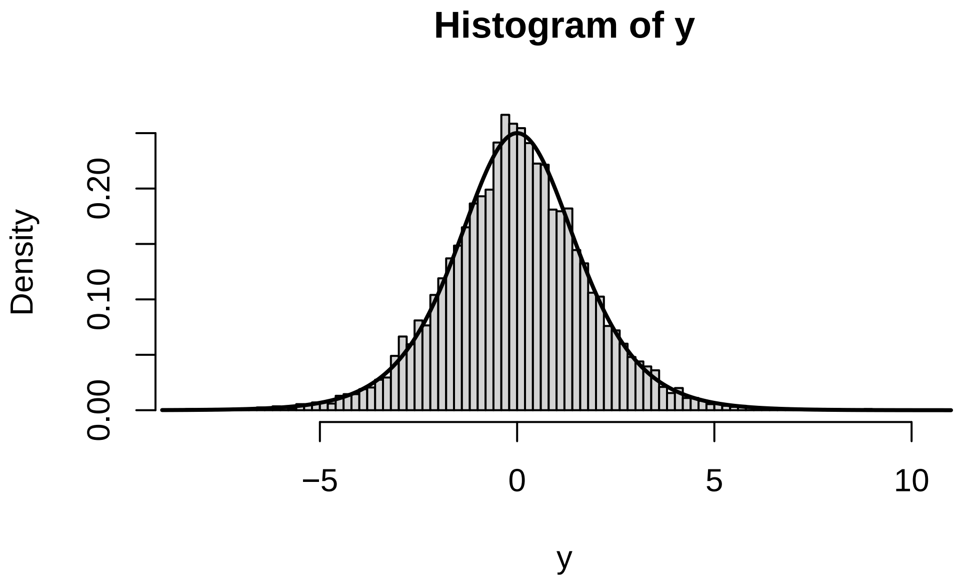
## 2.5 Example: Logistic

$$F(y) = \frac{1}{1 + \exp\left(\text{-}x\right)}$$

$$F^{\text{-}1}(u) = \log(u) \text{ - } \log(1 \text{ - } u)$$

```r
n <- 1e4
u <- runif(n)
y <- log(u) - log(1-u)
```

```r
x <- seq(from=floor(min(y)), to=ceiling(max(y)), length.out=500)
hist(y, freq=F, breaks=100)
lines(x=x, y=dlogis(x), lwd=2)
```



## 2.6 Example: Normal

```r
n <- 1e4
y <- rnorm(n)

u <- runif(n)
y <- qnorm(u)
```

## 2.7 Extensions: Multivariate Normal

# 3 Simulating from a Model

add

# 4 Maximum Likelihood Estimation

add

# 5 Optimization

add

# 6 Estimation

add

# 7 Code Performance

add

~~~
`<!-- quarto-file-metadata: eyJyZXNvdXJjZURpciI6ImNoYXB0ZXJzIn0= -->`{=html}

```{=html}
<!-- quarto-file-metadata: eyJyZXNvdXJjZURpciI6ImNoYXB0ZXJzIiwiYm9va0l0ZW1UeXBlIjoiY2hhcHRlc
~~~

# 8 Bayesian Approach

add

# Part III

# Probit

# 9 Multinomial Probit

add

# 10 Nested Logit Model

add

# 11 GEV Logit

add

# Part IV

# Mixed

# 12 Simulation-Assisted Estimation

add

# 13 Latent Class MNL Model

add

# 14 Mixed Logit Model

add

# References

Abramovich, Felix, and Ya'acov Ritov. 2023. *Bayesian Statistics.* CRC press.

Gelman, Andrew, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. 2013. *Bayesian Data Analysis.* CRC press.

Goldberger, Arthur S. 1991. *A Course in Econometrics.* Harvard University Press.

Hansen, Bruce. 2022a. *Econometrics.* Princeton University Press.

———. 2022b. *Probability & Statistics for Economists.* Princeton University Press.

Hensher, David A., John M. Rose, and William H. Greene. 2015. *Applied Choice Analysis.* 2nd ed. Cambridge University Press.

Kennedy, Peter. 2008. *A Guide to Econometrics.* John Wiley & Sons.

McElreath, Richard. 2018. *Statistical Rethinking: A Bayesian Course with Examples in r and Stan.* Chapman; Hall/CRC.

Train, Kenneth E. 2009. *Discrete Choice Methods with Simulation.* 2nd ed. Cambridge University Press. https://eml.berkeley.edu/books/choice2.html.

Wickham, Hadley. 2019. *Advanced r.* 2nd ed. Chapman; Hall/CRC.

Wickham, Hadley, Mine Cetinka-Rundel, and Garrett Grolemund. 2023. *R for Data Science.* 2nd ed. O'Reilly Media.

# A Appendix

add