TA Session of Econometrics 2 (2020-2021)

Hiroki Kato Pang Kan

Contents

1 About TA Session

- Class schedule: Friday pm 13:30-15:00 via zoom.
  - You can access the meeting ID and its pascode via CLE.
- Instructor (If you have any question, please contact us via e-mail)
  1. Hiroki Kato (D2, vge008kh@student.econ.osaka-u.ac.jp)
  2. Pang Kan (D1, member_1363710747@yahoo.co.jp)

- Purpose: We will review the contents of the main class “Econometics II.” using R which is a free software environment for statistical computing.
  - We strongly recommend that you download R (https://www.r-project.org/) and its IDE called R studio (https://rstudio.com/products/rstudio/download/), and try to reproduct by yourself.

2 Reviews of Matrix Algebra and Probability

2.1 Matrix Algebra

2.1.1 Addition and Subtraction

Consider $A = (a_{ij}) \in \mathbb{R}^{m \times n}$ and $B = (b_{ij}) \in \mathbb{R}^{p \times q}$. Addition and subtraction require that the dimentions are same, that is, $m = p$ and $n = q$. Then, the sum of two matricies is

$$A + B = (a_{ij} + b_{ij}).$$

The difference of two matricies is

$$A - B = (a_{ij} - b_{ij}).$$

2.1.2 Multiplication

The standard matrix multiplication requires that the number of columns of the first matrix is equal to the number of rows of the second matrix ($n = p = l$). The product of two matricies is

$$AB = (\sum_{k=1}^{l} a_{ik} b_{kj}) \in \mathbb{R}^{m \times q}.$$

We should remark following important points about multiplication:

- it holds non-commutativity: $XY \neq YX$;
- it holds associative law: $(XY)Z = X(YZ)$;
- it holds distributive law: $X(Y + Z) = XY + XZ$;
- when $B = A$, we obtain the second power of a matrix A, that is, $A^2 = AA$. Especially, if a martix A holds $AA = A$, then the matrix is called an **idempotent matrix (べき等行列)**.

We introduce the another key product of matrix, called the **Kronecker product (クロネッカー積)**. This is defined by

$$A \otimes B = (a_{ij} B) \in \mathbb{R}^{mp \times nq}.$$

The Kronecker product has a following property:

- $X_1 X_2 \otimes Y_1 Y_2 = (X_1 \otimes Y_1)(X_2 \otimes Y_2)$

2.1.3 Transposed Matrix, Diagonal Matrix, and Inverse Matrix

Consider $X = (x_{ij}) \in \mathbb{R}^{m \times n}$ throughout this subsection.

2.1.3.1 Transposed Matrix The **transposed matrix (転置行列)** of X, denoted by X' is a $n \times m$ matrix whose element x'_{ij} holds

$$x'_{ij} = x_{ji}.$$

That is, i-th row and j-th column element of transposed matrix is j-th row and i-th column element of original matrix. We remark following important points:

- it holds $(XY)' = Y'X'$;
- it holds $(XYZ)' = Z'Y'X'$;
- $(X \otimes Y)' = X' \otimes Y'$;
- let $x_i = (x_{i1}, \dots x_{ij})$ be a row vector of matrix X. Then, we have $X'X = \sum_{n=1}^{i} x'_n x_n$;
- if a matrix X holds $X' = X$, then the matrix is called a **symmetric matrix (対称行列)**.

2.1.3.2 Diagonal Matrix and Trace Suppose a matrix X is a **square matrix (正方行列)**, that is, $n = m$. The matrix X is called a **diagonal matrix (対角行列)** whose diagonal elements (i-th row and i-th column elements) consist of $(x_{11}, \dots x_{nn})$, and other elements are zero. That is,

$$X = diag(x_{11}, x_{22}, \dots, x_{ii}) = \begin{pmatrix} x_{11} & 0 & 0 & \cdots & 0 \\ 0 & x_{22} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & x_{nn} \end{pmatrix}.$$

Especially, a matrix $I = diag(1, 1, \dots, 1)$ is called an **identity matrix (単位行列)**.

There is one important concept, called **trace (トレース)**, related with diagonal elements of matrix. The trace of matrix is derived by the sum of diagonal elements, that is,

$$tr(X) = \sum_{n=1}^{i} x_{nn}.$$

The trace has following properties:

- $tr(cX) = c \cdot tr(X)$ where c is scalar;
- $tr(X') = tr(X)$;
- $tr(X + Y) = tr(X) + tr(Y)$;
- $tr(XY) = tr(YX)$;
- $xx' = tr(x'x) = tr(xx')$ if x is a $1 \times j$ vector.

2.1.3.3 Inverse Matrix

the matrix X is **regular matrix (正則行列)** if there exists a matrix Y such that

$$XY = I,$$

where I is an identity matrix. In this case, the matrix Y is called an **inverse matrix (逆行列)**, which is denoted by X^{-1}. The inverse matrix has following important properties:

- $(X^{-1})' = (X')^{-1}$;
- $(X \otimes Y)^{-1} = X^{-1} \otimes Y^{-1}$ if the inverse exists;
- $(X \otimes Y)(X^{-1} \otimes Y^{-1}) = I$

2.1.4 Quadratic Forms

Consider a symmetric and square matrix $A \in \mathbb{R}^{n \times n}$ and a vector $x \in \mathbb{R}^{n \times 1}$. Then, the quadratic form is written as

$$Q = x'Ax.$$

For example, consider $x = (x, y)'$ and A is a 2×2 matrix whose elements is one. Then, the quadratic form is

$$Q = \begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x+y & x+y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = x^2 + 2xy + y^2 = (x+y)^2.$$

In this case, for any non-zero x and y, Q takes non-negative value. Then, we call the matrix A *positive semidefinite*. The definiteness of matrix is defined as follows:

- If $x'Ax > 0$ for all nonzero x, then A is **positive definite (正値定符号)**.
- If $x'Ax < 0$ for all nonzero x, then A is **negative definite (負値定符号)**.
- If $x'Ax \geq 0$ for all nonzero x, then A is **positive semidefinite (半正値定符号)**.
- If $x'Ax \leq 0$ for all nonzero x, then A is **negative semidefinite(半負値定符号)**.

2.1.4.1 Characteristic Roots and Characteristic Vectors

Before describing useful theorm to check definiteness easily, we have to introduce two concepts: **characteristic roots (固有根)** and **characteristic vectors (固有ベクトル)**.

> If a scalar λ and a vector $c \in \mathbb{R}^{k \times 1}$, which is normalized as $c'c = 1$, satisfy the following equation, then they are called as the **characteristic root** and the **characteristic vector**, respectively;
>
> $$Ac = \lambda c \Leftrightarrow (A - \lambda I)c = 0,$$
>
> where I is an identity matrix.

These $(\lambda_1, \lambda_2, ..., \lambda_k)$ correspond to characteristic vectors $(c_1, c_2, \dots, c_k)$. There is the following useful theorm that states the relationship between characteristic roots and definiteness:

> Let A be a symmetric matrix.

1. If all the characteristic roots of A are positive (negative), then A is positive definite (negative definite).
2. If some of roots are zero, then A is positive (negative) semidefinite if the reminder are positive (negative).
3. If A has both negative and positive roots, then A is indefinite.

2.1.4.2 Determinants Alternative way to check definiteness of matrix is to using **determinants (行列式)**, which is a scalar quantity defined by a square matrix $A = (a_{ij}) \in \mathbb{R}^{n\times n}$. The determinant of 2×2 matrix, i.e., $n = 2$, is obtained by

$$det(A) = a_{11}a_{22} - a_{12}a_{21}.$$

Using the determinant of a matrix with $n = 2$, we can calculate the determinant of 3×3 matrix. Let $det(A_{ij})$ be the determinant of the 2×2 submatrix obtained when i-th row and j-th column are removed from the original matrix, which is called **minor (小行列式)**. Furthermore, we define $C_{ij} = (-1)^{i+j}det(A_{ij})$, which is called **cofactor (余因子)**. We call a matrix in which each element a_{ij} is replaced by the corresponding cofactor C_{ij} **cofactor matrix (余因子行列)**. Then, the determinant of 3×3 matrix is

$$\begin{aligned} det(A) &= a_{11}C_{11} + a_{12}C_{12} + a_{13}C_{13} \\ &= a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}, \end{aligned}$$

or

$$det(A) = a_{11}C_{11} + a_{21}C_{21} + a_{31}C_{31}.$$

How about matrices with $n \geq 4$? Essentially, we can calculate the determinant, using cofactors. That is, for any i,

$$det(A) = \sum_{k=1}^{n} a_{ik}C_{ik},$$

or, for any j

$$det(A) = \sum_{k=1}^{n} a_{kj}C_{kj},$$

Before describing the important theorem to check definiteness, we introduce the *Cramer's rule*, which provides inverse matrices.

> Let $A \in \mathbb{R}^{n\times n}$ be a square matrix with $det(A) \neq 0$. Then, the inverse matrix of A is equal to the transposed cofactor matrix multiplied by $det(A)^{-1}$. That is,

$$A^{-1} = \frac{1}{det(A)}\begin{pmatrix} C_{11} & \cdots & C_{n1} \\ \vdots & \vdots & \vdots \\ C_{1n} & \cdots & C_{nn} \end{pmatrix}.$$

Next, we introduce the useful theorem to check definiteness of matricies. The determinant has the follwoing relation with the definiteness of matricies.

> Let $A \in \mathbb{R}^{n \times n}$ be a square matrix. Let
>
> $$det(A_i) = \begin{vmatrix} a_{11} & \cdots & a_{1i} \\ \vdots & \vdots & \vdots \\ a_{i1} & \cdots & a_{ii} \end{vmatrix}.$$
>
> A necessary and sufficient condition for a matrix A to be positive definite is that $det(A_i) > 0$ for all $i \in \{1, \dots, n\}$. Moreover, a necessary sufficient condition for a matrix A to be negative definite is that $det(A_i) < 0$ for odd i and $det(A_i) > 0$ for even i.

As an illustration, consider the following matrix:

$$A = \begin{pmatrix} 6 & 4 \\ 4 & 5 \end{pmatrix}.$$

Then, we have $det(A_1) = 6 > 0$ and $det(A_2) = 30 - 16 > 0$. Thus, this matrix is positive definite.

To show another example, we use the following diagonal matrix:

$$A = \begin{pmatrix} -3 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -1 \end{pmatrix}.$$

Since the determinant of a diagonal matrix can be computed as the product of diagonal elements, we have $det(A_1) = -3$, $det(A_2) = (-3)(-2) = 6 > 0$, and $det(A_3) = (-3)(-2)(-1) = -6 < 0$. Thus, this diagonal matrix is negative definite. Moreover, the inverse matrix of this diagonal matrix is given by

$$A^{-1} = -\frac{1}{6} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 6 \end{pmatrix} = \begin{pmatrix} -1/3 & 0 & 0 \\ 0 & -1/2 & \\ 0 & 0 & -1 \end{pmatrix}$$

Cleary, we have $AA^{-1} = I$.

2.1.5 Differentiation

Consider two vectors: $a \in \mathbb{R}^{n \times 1}$ and $x \in \mathbb{R}^{n \times 1}$. We obtain the product of transposed vector of a and x, that is, $a'x = a_1 x_1 + \cdots + a_n x_n$. Then, the differentiation of this scalar with respect to x is defined by

$$\frac{\partial a'x}{\partial x} = \begin{pmatrix} \frac{\partial a'x}{\partial x_1} \\ \vdots \\ \frac{\partial a'x}{\partial x_n} \end{pmatrix} = a.$$

Now, we expand to a symmetric and square matrix $A \in \mathbb{R}^{n\times n}$. Then, the differentiation of the quadratic form $x'Ax$ with respect to x is defined by

$$\frac{\partial x'Ax}{\partial x} = (A + A')x.$$

2.1.5.1 Optimization Consider function $y = g(x)$ where $x \in \mathbb{R}^n$, denoted as $g : \mathbb{R}^n \to \mathbb{R}$. We can obtain x^0 such that maximizing (minimizing) the function g, using the following theorem:

> If a function $g\colon \mathbb{R}^n \to \mathbb{R}$ is maximized (minimized) at the point $x^0 = (x^0_1, \dots, x^0_n)$, then the following equation holds:
>
> $$\frac{\partial g(x)}{\partial x}\Big|_{x=x^0} = \begin{pmatrix} \frac{\partial g(x^0)}{\partial x_1} \\ \vdots \\ \frac{\partial g(x^0)}{\partial x_n} \end{pmatrix} = 0.$$
>
> x^0 is maximum (minimum) point if the following **Hessian matrix (ヘッセ行列)** is negative (positive) definite:
>
> $$H = \frac{\partial g(x)}{\partial x x'} = \begin{pmatrix} \frac{\partial^2 g(x)}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 g(x)}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 g(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 g(x)}{\partial x_n \partial x_n} \end{pmatrix}.$$

2.2 Probability

This section refers to Wasserman (2013). Let Ω be a **(sample) space** which is the set of possible outcomes of an experiment. Let ω be **sample outcomes**, **reaizations**, or **elements**. Let A be **events** which are the subsets of Ω. Then, we can define the **probability** and **random variable** as follows:

> A function $\mathbb{P}$ that assigns a real number $\mathbb{P}(A)$ to each event A is a **probability** if it satisfies the following three axioms:
>
> 1. $\mathbb{P}(A) \geq 0$ for all A;
> 2. $\mathbb{P}(\Omega) = 1$;
> 3. If $A_1, A_2, \dots$ are disjoint, then $\mathbb{P}(\bigcup_{k=1}^{\infty} A_k) = \sum_{k=1}^{\infty} \mathbb{P}(A_i)$.
>
> A **random variable** is a mapping $X : \Omega \to \mathbb{R}$ that assigns a real number $X(\omega)$ to each realization ω.

For illustlation, consider the situation where you try to flip a fair coin twice. Then, the sample space $\Omega = \{TT, HH, TH, HT\}$. The probability of each outcome is $1/4$, that is, $P(\omega) = 1/4$ for all ω. Let the random variable X be the number of heads. Then, $X(TT) = 0$, $X(HH) = 2$, $X(TH) = 1$, and $X(HT) = 1$.

2.2.1 Distribution Functions

Given a random variable X, we define **probability mass function**, **probability density function** and **cumulative distribution function** as follows:

> Suppose that a random variable X is *discrete* taking countably many values $\{x_1, ...\}$. Then, the **probability mass function** for X is defined by $f_X(x) = \mathbb{P}(X = x)$.
>
> Suppose that a random variable X is *continuous*. Then, there exists a **probability density function** f_X such that (i) $f_X(x) \geq 0$ for all x, (ii) $\int_{-\infty}^{+\infty} f_X(x)dx = 1$ and (iii) for every $a \leq b$, $\mathbb{P}(a < X < b) = \int_a^b f_X(x)dx$.
>
> The **cumulative distribution function (CDF)** is the function $F_X : \mathbb{R} \to [0, 1]$ defined by $F_X(x) = \mathbb{P}(X \leq x)$.

We summarize the relationship among three distribution functions as follows:

$$F_X(x) = \begin{cases} \sum_{x_i \leq x} f_X(x_i) & X \text{ is discrete} \\ \int_{-\infty}^{x} f_X(t)dt & X \text{ is continuous} \end{cases}$$

From this, we obtain the property of CDF:

- F is non-decreasing: $x_1 < x_2$ implies $F(x_1) \leq F(x_2)$;
- F is normalized: $\lim_{x \to -\infty} F(x) = 0$ and $\lim_{x \to +\infty} F(x) = 1$;
- F is right-contious: $F(x) = F(x^+)$ for all x, where $F(x^+) = \lim_{y \downarrow x} F(y)$;
- $\mathbb{P}(X = x) = F(x) - F(x^-)$ where $F(x^-) = \lim_{y \uparrow x} F(y)$;
- $\mathbb{P}(x < X \leq y) = F(y) - F(x)$;
- $\mathbb{P}(X > x) = 1 - F(x)$.

2.2.1.1 Bivariate Distributions When there are two random varibales, you can define bivariate distributions as follows:

> Given discrete random variables X and Y, we define the **joint mass function** by $f_{X,Y}(x, y) = \mathbb{P}(X = x, Y = y)$.
>
> In the continuous case, we call a function $f(x, y)$ a **joint probability density function** for the random variables (X, Y) if (i) $f(x, y) \geq 0$, $\forall(x, y)$, (ii) $\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y)dxdy = 1$, and, (iii) for any $A \subset \mathbb{R} \times \mathbb{R}$, $\mathbb{P}[(X, Y) \in A] = \int \int_A f(x, y)dxdy$.
>
> In both cases, we define the **joint cumulative distribution function** as $F_{X,Y}(x, y) = \mathbb{P}(X \leq x, Y \leq y)$.

When you are interested to the probability of a single event occurring, there are two distributions called a **marginal distribution (周辺分布)** and a **conditional distribution (条件付き分布)**. The formar distribution is the probability of $X = x$ independent of Y. On the other hand, the latter distribution is the probability that $X = x$ occurs given the event $Y = y$ has already occured. Formally, The **marginal distribution** is defined by

$$f_X(x) = \mathbb{P}(X = x) = \begin{cases} \sum_y \mathbb{P}(X = x, Y = y) & X \text{ is discrete} \\ \int f(x, y) dy & X \text{ is continuous} \end{cases}.$$

The **conditional distribution** is defined by

$$f_{X|Y}(x|y) = \mathbb{P}(X = x|Y = y) = \begin{cases} \sum_y \frac{\mathbb{P}(X=x,Y=y)}{\mathbb{P}(Y=y)} & X \text{ is discrete} \\ \int \frac{f_{X,Y}(x,y)}{f_Y(y)} dy & X \text{ is continuous} \end{cases},$$

To define the conditional distribution function, we assume $\mathbb{P}(Y = y) > 0$ for discrete random variables and $f_Y(y) > 0$ for continuous random variables. In the case of continuous random variables, we must integrate to get a probability, that is,

$$\mathbb{P}(X \in A|Y = y) = \int_A f_{X|Y}(x|y) dx.$$

Finally, we introduce very important concept of probability, called **independence**, and define it as follows:

> Two random variables X and Y are **independent** if, for every event A and B, $\mathbb{P}(X \in A, Y \in B) = \mathbb{P}(X \in A)\mathbb{P}(Y \in B)$.

In principle, to check independence, we need to check whether this relationship for all subsets A and B. But, there is useful theorem to check independence.

> Let X and Y have joint PDF $f_{X,Y}$. Then, X and Y are independent if and only if $f_{X,Y}(x, y) = f_X(x)f_Y(y)$ for all values x and y.

Note that if two random variables are independent, then the conditional probability $\mathbb{P}(X = x|Y = y)$ reduces to $\mathbb{P}(X = x)$.

2.2.2 Expectations, Variance, and Covariance

Roughly speaking, the expectation of a random variable X is the average value of X. The variance of a random variable X measures the "spread" of a distribution. Formally,

> The **expected value (mean, first moment)** of X is defined to be
>
> $$E(X) = \mu_X = \int x dF(x) = \begin{cases} \sum_x x f(x) & X \text{ is discrete} \\ \int x f(x) dx & X \text{ is continuous} \end{cases}$$
>
> The **variance** of X is defined by $V(X) = \sigma^2 = E(X-\mu_X)^2 = \int (x-\mu_X)^2 dF(x)$.
> The **standard deviation** is $sd(X) = \sqrt{V(X)}$.

The mean of random variable, $E(X)$, exists if $\int_x |x| dF_X(x) < \infty$. Expectation and variance has some useful properties:

- Let $Y = r(X)$. Then, $E(Y) = E(r(X)) = \int r(x)dF(x)$;
- Suppose that $X_1, \dots, X_n$ are random variables and $a_1, \dots, a_n$ are constants. $E(\sum_i a_i X_i) = \sum_i a_i E(X_i)$;
- Suppose that $X_1, \dots, X_n$ are independent random variables. Then, $E(\prod_{i=1}^n X_i) = \prod_i E(X_i)$;
- $V(X) = E(X^2) - \mu^2$;
- $V(aX + b) = a^2 V(X)$ where a and b are constants;
- Suppose that $X_1, \dots, X_n$ are independent random variables and $a_1, \dots, a_n$ are constants. $V(\sum_i a_i X_i) = \sum_i a_i^2 V(X_i)$.
- If a is a vector and X is a random vector with mean μ and variance Σ, then $E(a'X) = a'\mu$ and $V(a'X) = a'\Sigma a$. If A is a matrix, then $E(AX) = A\mu$ and $V(AX) = A\Sigma A'$.

We introduce some theorems about probability inequalities which is used in the theory of convergence.

- Markov's inequality: Let X be a non-negative random variable and suppose that $E(X)$ exists. For any $t > 0$, $\mathbb{P}(X > t) \leq E(X)/t$.
- Chebyshev's inequality: Let $\mu = E(X)$ and $\sigma^2 = V(X)$. Then, $\mathbb{P}(|X - \mu| \geq t) \leq \sigma^2/t^2$ and $\mathbb{P}(|Z| \geq k) \leq 1/k^2$ where $Z = (X - \mu)/\sigma$.
- Jensen's inequality: If g is convex, then $E[g(X)] \geq g(E(X))$. If g is concave, then $E[g(X)] \leq g(E(X))$.

Next, we will introduce the definition of **covariance**. These measure how strong the linear relationship is between X and Y.

> Let X and Y be random variables with means μ_X and μ_Y, respectively. Then, the **covariance** between X and Y is defined by $\text{Cov}(X, Y) = E[(X - \mu_X)(Y - \mu_Y)]$.

Covariance has following properties:

- $\text{Cov}(X, Y) = E(XY) - E(X)E(Y)$;
- If X and Y are independent, $\text{Cov}(X, Y) = 0$. **The converse is not true.**
- $V(X + Y) = V(X) + V(Y) + 2\text{Cov}(X, Y)$
- $V(X - Y) = V(X) + V(Y) - 2\text{Cov}(X, Y)$
- $V(\sum_i a_i X_i) = \sum_i a_i^2 V(X_i) + 2 \sum_i \sum_{i<j} a_i a_j \text{Cov}(X_i, X_j)$.

2.2.2.1 Conditional Expectation and Variance

The conditional expectation of X given $Y = y$ is

$$E(X|Y = y) = \begin{cases} \sum_x x f_{X|Y}(x|y) & X \text{ is discrete} \\ \int x f_{X|Y}(x|y) dx & X \text{ is continuous} \end{cases}$$

The conditional variance is defined as

$$V(X|Y = y) = \int (x - \mu(y))^2 f(x|y) dx.$$

where $\mu(y) = E(X|Y = y)$.

Even if $r(x, y)$ is a function of x and y, we can define the conditional expectation. For the continuous random variable, $E[r(X, Y)|Y = y] = \int r(x, y) f_{X|Y}(x|y)dx$. For the discrete random variable, $E[r(X, Y)|Y = y] = \sum_x r(x, y) f_{X|Y}(x|y)$.

We have following important properties:

- For random variables X and Y, assuming the expectations exist, we have that $E_X[E(Y|X)] = E(Y)$. More generally, for any function $r(x, y)$, we have $E_X[E(r(X, Y)|X)] = E(r(X, Y))$.
- For random variables X and Y, $V(Y) = E_X[V(Y|X)] + V_X[E(Y|X)]$.

2.2.2.2 Moment The expectation of a random variable X to the k-th power, $E(X^k)$ is called k-th **moment** of X. If the k-th moment exists and if $j < k$, then j-th moment exists. Now, we define the **moment generating function** which is used for finding moments.

> The **moment generating function** of X is defined by
>
> $$\psi_X(t) = E(e^{tX}) = \int e^{tX} dF(x),$$
>
> where t varies over the real numbers.

Now, we assume that the moment generating function is well defined for all t in some open interval around $t = 0$. Then, we can interchange the operations of differentiation and taking expectation. Thus, we obtain

$$\psi'(0) = \frac{d}{dt} E(e^{tX})|_{t=0} = E\left(\frac{d}{dt} e^{tX}\right)|_{t=0} = E(Xe^{tX})|_{t=0} = E(X).$$

This implies that the mean of random variable is derived by taking first-order derivatives at $t = 0$. Thus, we can conclude that $\psi^{(k)}(0) = E(X^k)$. We should remark properties of the moment generating function.

- If $Y = aX + b$, then $\psi_Y(t) = e^{bt}\psi_X(at)$.
- If $X_1, \ldots, X_n$ are independent and $Y = \sum_i X_i$, then $\psi_Y(t) = \prod_i \psi_i(t)$ where ψ_i is the moment generating function of X_i.
- Let X and Y be random variables. If $\psi_X(t) = \psi_Y(t)$ for all t in an open interval around 0, then X and Y have the same distribution function.

2.2.3 Convergence

When we are interested in what happens as we gather more and more data, we need to concern the limiting behavior of a sequence of random variables. This part of probability is called **large sample theory** or **asymptotic theory (漸近理論)**. First, we will define two types of convergence as follows:

> Let $X_1, X_2, \ldots$ be a sequence of random variables and let X be another random variable. Let F_n denote the cumulative distribution function (CDF) of X_n and let F denote the CDF of X.

1. X_n **converges to** X **in probability**, written $X_n \xrightarrow{p} X$, if for every $\epsilon > 0$, $\mathbb{P}(|X_n - X| > \epsilon) \to 0$ as $n \to \infty$.
2. X_n **converges to** X **in distribution**, written $X_n \xrightarrow{d} X$, if $\lim_{n\to\infty} F_n(t) = F(t)$ for all t for which F is continuous.

We should remark the relationship bewteen two types of convergence and properties of each type of convergence. Especially, the property 4 and 6 are called the **Slutsky theorem**, and the property 7 and 8 are called the **continuous mapping theorem**.

1. $X_n \xrightarrow{p} X$ implies that $X_n \xrightarrow{d} X$
2. If $X_n \xrightarrow{d} X$ and $\mathbb{P}(X = c) = 1$ for some real number c, then $X_n \xrightarrow{p} X$
3. If $X_n \xrightarrow{p} X$ and $Y_n \xrightarrow{p} Y$, then $X_n + Y_n \xrightarrow{p} X + Y$
4. If $X_n \xrightarrow{d} X$ and $Y_n \xrightarrow{d} c$, then $X_n + Y_n \xrightarrow{d} X + c$
5. If $X_n \xrightarrow{p} X$ and $Y_n \xrightarrow{p} Y$, then $X_n Y_n \xrightarrow{p} XY$
6. If $X_n \xrightarrow{d} X$ and $Y_n \xrightarrow{d} c$, then $X_n Y_n \xrightarrow{d} cX$
7. If $X_n \xrightarrow{p} X$, then $g(X_n) \xrightarrow{p} g(X)$
8. If $X_n \xrightarrow{d} X$, then $g(X_n) \xrightarrow{d} g(X)$

2.2.3.1 Law of Large Numbers

The first important theorem of asymptotic theory is the **(weak) law of large numbers**. This theorem says that the mean of a large sample is close to the mean of distribution. Now, we will state more precisely.

> Let $X_1, X_2, \dots$ be an IID sample. Suppose that $\mu = E(X_i)$ for all i and $\sigma^2 = V(X_i)$ for all i. The sample mean is defined by $\bar{X}_n = n^{-1} \sum_{i=1}^n X_i$. Then, $\bar{X}_n \xrightarrow{p} \mu$.

As an illustlation, consider a situation where you flip a fair coin toss n times. The space is $\Omega = \{H, T\}$. The random variable X_i is the number of heads, that is, $X_i(H) = 1$ and $X_i(T) = 0$ for $i = 1, \dots, n$, which is binomially distrbuted with one trial and probability 0.5, $B(1, 0.5)$ (Bernoulli distribution). The sample mean of this random variable represents the proportion of heads. WLLN says that the sample mean is close to 0.5 as n gets large.

We will simulate using R. First, the random variable of Bernoulli distribution is generated by `rbinom(n, size = 1, prob)` where `n` is the number of trials, `prob` is the probability of success (head). When you specify `size` is greater than one, this random variable indicated the number of sucess when you flip coin `size` times. We calculate the proportion of heads when $n = 1, \dots, 20000$, and show line plot with logged number of trial on x-axis and the proportion of heads on y-axis.

```r
set.seed(120504)

data <- data.frame(
  trial = 1:20000,
  success = rbinom(n = 20000, size = 1, prob = .5)
)
data$sum_success <- cumsum(data$success)
data$prob <- data$sum_success/data$trial

plot(
  log(data$trial), data$prob, type = "l", col = "blue",
  ylim = c(0.3, 0.7), xlab = "logged trials", ylab = "Pr(head)")
lines(c(0, 10), c(0.5, 0.5), lwd = 1, col = "red")
```

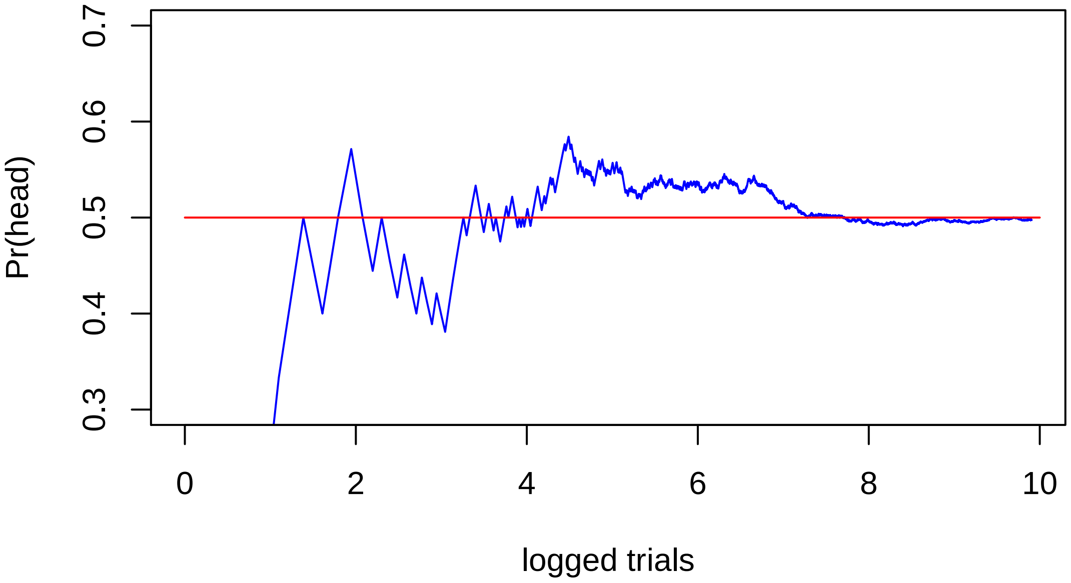

Figure 1: Simulation Result of WLLN

2.2.3.2 Central Limit Theorem The second important theorm is the **central limit theorem**. Suppose that $X_i, \dots, X_n$ are IID sample with mean μ and variance σ^2. This theorem says that the sample mean $\bar{X}_n$ has a distribution which is approximately normal with mean μ and variance σ^2/n. This theorem does not assume the distribution of X_i, except the existence of the mean and variance. Formally,

> Let $X_1, \dots, X_n$ be IID with mean μ and variance σ^2. Let $\bar{X}_n = n^{-1}\sum_{i=1}^{n} X_i$. Then,

$$Z_n \equiv \frac{\bar{X}_n - \mu}{\sqrt{V(\bar{X}_n)}} = \sqrt{n}\frac{\bar{X}_n - \mu}{\sigma} \overset{d}{\to} Z,$$

where $Z \sim N(0,1)$. In other words, $\bar{X}_n \xrightarrow{d} N(\mu, \sigma^2/n)$.

As an illustlation, consider a fair coin toss. The random variable is the number of heads. This random variable has the Bernoulli distribution with mean $\mu = 0.5$ and variance $\sigma^2 = 0.5(1-0.5) = 0.25$. Since we know μ and σ^2, we can calculate Z_n, using the sample mean $\bar{X}_n$. We work this and plot its distribution, using R programming. We generate 10,000 sample means $\bar{X}_n$ for $n = 3, 5, 100, 1000$, and transform sample means to Z_n. To calculate Z_n, we use command `sqrt()`, which returns the saquare root value. Sometimes, this procedure is called Monte-Carlo simulation.

```r
set.seed(120504)
m <- 10000; n <- c(3, 100, 1000); p <- 0.5
a <- seq(-4, 4, .01); b <- dnorm(a)

dt <- list("n = 3"=numeric(m), "n = 100"=numeric(m), "n = 1000"=numeric(m))
for (i in 1:3) {
  dt[[i]] <- rbinom(n = m, size = n[i], prob = p)
  dt[[i]] <- sqrt(n[i])*(dt[[i]]/n[i] - p)/sqrt(p*(1-p))
}

par(mfrow=c(2,2), mai = c(0.5, 0.5, 0.35, 0.35))
for (i in 1:3) {
  hist(dt[[i]], col = "grey", freq = FALSE,
    xlab = "", main = names(dt)[i], xlim = c(-4, 4))
  par(new = TRUE)
  plot(a, b, type = "l", col = "red", axes = FALSE,
    xlab = "", ylab = "", main = "")
}
```

3 Reviews of Ordinary Least Squares and Maximum Likelihood Estimation

This section refers to Johnston (1984) and Angrist and Pischke (2008). Consider the k-variables lienar regression model:

$$y_i = x_i \beta + u_i,$$

where $\beta = (\beta_0, \beta_1, \dots, \beta_k)'$ is a $k \times 1$ vector of regression coefficients, $x_i = (1, x_{i1}, \dots, x_{ik})$ is a $1 \times k$ vector of stochastic covariates, and u_i is the error term which is idependent and identically distributed (i.i.d.). Our parameter of interest is β.

3.1 Ordinary Least Squares Estimator (OLSE)

The **OLS estimators** are the value β such that minimizing the residual sums of squares, that is,

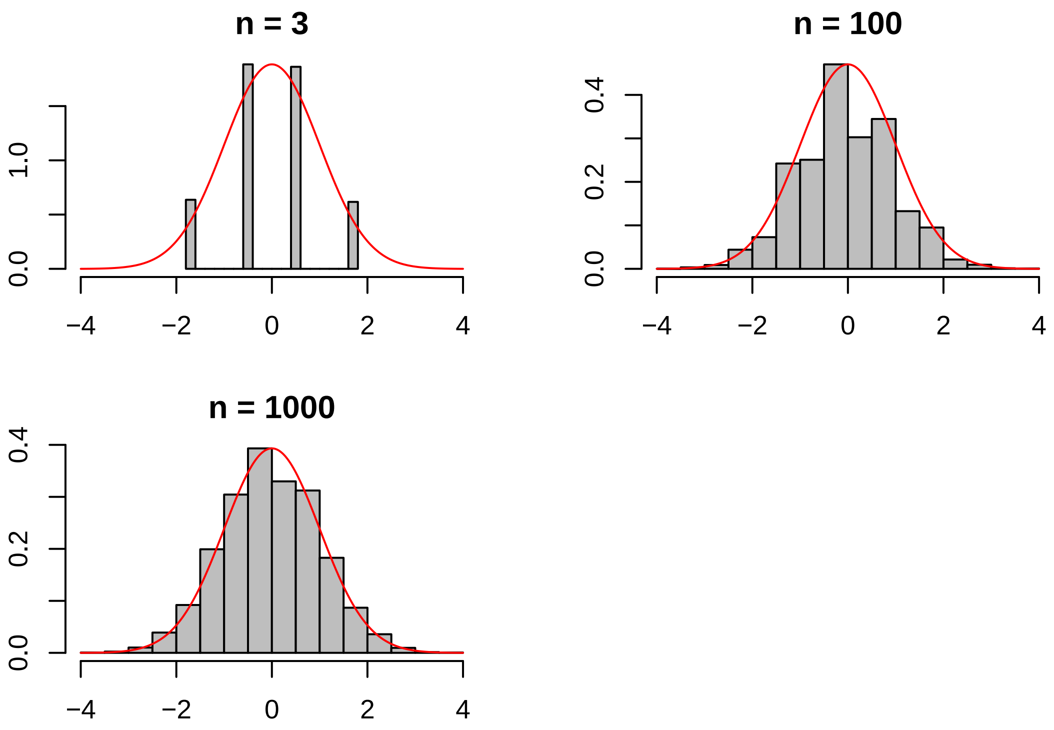

Figure 2: Simulation Result of CLT

The **OLS estimators** $\hat{\beta}$ is defined by

$$\hat{\beta} \in \arg\min_{\beta} \sum_{i=1}^{n} (y_i - x_i\beta)^2,$$

or,

$$\hat{\beta} \in \arg\min_{\beta} (Y - X\beta)'(Y - X\beta),$$

where $Y = (y_1, \dots, y_n)'$ is a $n \times 1$ vector, and $X = (x_1, \dots, x_n)'$ is a $n \times k$.

Following this definition, the OLSE is given by

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

To exist the inverse matrix, we assume that the matrix $(X'X)$ is the regular matrix (i.e., there is no perfect correlation between any two covariates).

3.1.1 Best Linear Unbiased Estimator (BLUE)

We impose assumptions about the disturbance vector u: (i) $E(u|X) = 0$ (exogenity assumption or mean-idependence), and (ii) $V(u|X) = \sigma^2 I$ (homoscedasticity and pairwise uncorrelation). Under this condition, the OLS estimator is a linear unbiased estimator, that is, $E(\hat{\beta}) = \beta$ since

$$E(\hat{\beta}|X) = E[\beta + (X'X)^{-1}(X'u)|X] = \beta + (X'X)^{-1}X'E(u|X) = \beta.$$

Furthermore, the variance-covariance matrix of OLSE is

$$\begin{aligned} V(\hat{\beta}|X) &= E[(X'X)^{-1}X'uu'X(X'X)^{-1}|X] \\ &= (X'X)^{-1}X'\sigma^2 IX(X'X)^{-1} \\ &= \sigma^2(X'X)^{-1}. \end{aligned}$$

Note that $V(\hat{\beta}) = \sigma^2 E[(X'X)^{-1}]$. The most important result is that no other linear unbiased estimator can have smaller variances than those of OLSE. In other words, the OLSE has minimum variance within the class of linear unbiased estimators. Thus, the OLSE is a best linear unbiased estimator (**BLUE**). This result is known as the *Gauss-Markov theorem* (We omit proof).

3.1.2 Asymptotic Properties

First, the OLSE is a consistent estimator, that is,

$$\operatorname{plim} \hat{\beta} = \beta + \operatorname{plim} \left(\frac{1}{n}(X'X) \right)^{-1} \operatorname{plim} \left(\frac{1}{n} X'u \right) = \beta.$$

This is bacause $\operatorname{plim} n^{-1}(X'X) = \operatorname{plim} n^{-1} \sum_i x_i' x_i = E[x_i'x_i] = \Sigma$ and $\operatorname{plim} n^{-1}(X'u) = \operatorname{plim} n^{-1} \sum_i x_i' u_i = E[x_i' u_i] = 0$ by mean-independence assumption.

Second, the OLSE is asymptotically normally distributed. To show it, we derive the asymptotic distribution of $\sqrt{n}(\hat{\beta} - \beta)$ where

$$\sqrt{n}(\hat{\beta} - \beta) = \left(\frac{1}{n} \sum_i x_i' x_i \right)^{-1} \sqrt{n} \left(\frac{1}{n} \sum_i x_i' u_i \right).$$

By the central theorem, we have

$$\sqrt{n} \left(\frac{1}{n} \sum_i x_i' u_i \right) \xrightarrow{d} N(0, \sigma^2 \Sigma).$$

Recall that $n^{-1} \sum_i x_i' x_i \xrightarrow{p} \Sigma$. By the Slutsky theorem (the 6th property of convergence), we get

$$\sqrt{n}(\hat{\beta} - \beta) \xrightarrow{d} N(0, \sigma^2 \Sigma^{-1}),$$

or,

$$\hat{\beta} \xrightarrow{d} N\left(\beta, \frac{1}{n} \sigma^2 \Sigma^{-1} \right).$$

In a practical application, the unknown Σ is replaced by the sample estimate $n^{-1}X'X$, and the unknown σ^2 is estimated by $\hat{\sigma}^2 = \hat{u}'\hat{u}/(n-k)$ where $\hat{u} = Y - X\hat{\beta} = (I_n - X(X'X)^{-1}X')u = M_X u$ and M_X is a symmetric and idempotent matrix. Note that $\hat{\sigma}^2$ is an unbiased estimator of σ^2 since

$$E[\hat{\sigma}^2] = \frac{1}{n-k}E[tr(M_X uu')] = \frac{\sigma^2}{n-k}tr(M_X I_n) = \sigma^2.$$

3.1.3 Finite-sample Distribution and Inference

Now, we add the assumption with respect to the error term, $\epsilon_i | x_i \overset{iid}{\sim} N(0, \sigma^2)$. Then, we immediately obtain

$$\hat{\beta}|X \sim N(\beta, \sigma^2(X'X)^{-1}).$$

Consider the set of linear null hypothesis embodied in $R\beta = r$ where R is a arbitrary $q \times k$ matrix and r is a known q-element vector. To develop a test procedure, we derive the exact distribution of $R\hat{\beta}$. Cleary, we see $E(R\hat{\beta}) = R\beta$ and $V(R\hat{\beta}) = \sigma^2 R(X'X)^{-1}R'$. This leads to

$$R(\hat{\beta} - \beta) \sim N(0, \sigma^2 R(X'X)^{-1}R').$$

If the null hypothesis is true, then

$$R\hat{\beta} - r \sim N(0, \sigma^2 R(X'X)^{-1}R').$$

Using it and $\hat{u}'\hat{u} = u'M_X u$, we have follwing two distributions

$$\begin{aligned}
&(R\hat{\beta} - r)'[\sigma^2 R(X'X)^{-1}R']^{-1}(R\hat{\beta} - r) \sim \chi^2(q), \\
&\frac{\hat{u}'\hat{u}}{\sigma^2} \sim \chi^2(n-k)
\end{aligned}$$

To derive these distributions, we use the following two properties about chi-squared distribution:

- If $x \sim N(0, \Sigma)$, then $x'\Sigma x \sim \chi^2(n)$ where x is n-element vector.
- If $x \sim N(0, \sigma^2 I)$ and A is idempotent matrix, then $(\sigma^2)^{-1}x'Ax \sim \chi^2(tr(A))$.

Finally, since $X_1 \sim \chi^2(d_1)$ and $X_2 \sim \chi^2(d_2)$ lead to $\frac{X_1}{d_1}/\frac{X_2}{d_2} \sim F(d_1, d_2)$, we have the distribution of test statistic, called the F-distribution,

$$\frac{(R\hat{\beta} - r)'[R(X'X)^{-1}R']^{-1}(R\hat{\beta} - r)/q}{\hat{u}'\hat{u}/(n-k)} \sim F(q, n-k).$$

The test procedure is to reject the null hypothesis $R\beta = r$ if the computed F-value exceeds a preselected cricial value.

Especially, when we test a single coefficient, we can use the t-value as an alternative test statistic. Suppose that $R = (0, 1, 0, \ldots, 0)$ and $r = 0$. The null hypothesis is $\hat{\beta}_2 = 0$. The matrix $R(X'X)^{-1}R'$ picks up the second diagonal element of $(X'X)^{-1}$ denoted by $(X'X)^{-1}_{22}$. Then, we have

$$\frac{\hat{\beta}_2^2}{\hat{\sigma}^2(X'X)^{-1}_{22}} \sim F(1, n-k).$$

Since $t \sim t(n)$ is equivalent to $t^2 \sim F(1,n)$ for any n, we finally obtain the test statistic following the Student's t-distribution

$$\frac{\hat{\beta}_2}{\hat{\sigma}\sqrt{(X'X)^{-1}_{22}}} \sim t(n-k).$$

When you use t-test of a single coefficient, you should *two-sided* t-test. If the computed t-statistic $\hat{t}$ holds $|\hat{t}| > t_{1-\alpha/2}(n-k)$ where $t_q(n-k)$ is the q-percentile t-value, then we can reject the null hyporhesis $\hat{\beta}_2 = 0$

3.2 Maximum Likelihood Estimator (MLE)

When we assume that the error term is normally distributed, we have $y_i|x_i \overset{iid}{\sim} N(x_i\beta, \sigma^2)$. Under this assumption, the estimator $\tilde{\beta}$ maximizing the log-likelihood function, called **maximum likelihood estimator**, is equivalent to the OLSE. The likelihood function is

$$\prod_{i=1}^{n} f(y_i, x_i) = \prod_{i=1}^{n} f_{Y|X}(y_i|x_i) \prod_{i=1}^{n} f_X(x_i) = \sum_{i=1}^{n} \log f_{Y|X}(y_i|x_i) + \sum_{i=1}^{n} \log f_X(x_i).$$

Since $f_X(x_i)$ does not involve the parameter vector β, the *conditional* MLE $\tilde{\beta}$ maximizes the conditional log-likelihood function $\sum_{i=1}^{n} \log f_i(y_i|x_i)$, that is,

$$\begin{aligned}\log L(\theta) &= \sum_{i=1}^{n} \log f_{Y|X}(y_i|x_i)\\ &= \sum_{i=1}^{n} \log\left((2\pi\sigma^2)^{-1/2}\exp\left(-\frac{(y_i - x_i\beta)^2}{2\sigma^2}\right)\right)\\ &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\sigma^2 - \frac{1}{2\sigma^2}(Y - X\beta)'(Y - X\beta),\end{aligned}$$

where $\theta = (\beta', \sigma^2)'$ is a $(k+1)\times 1$ vector of unknown parameters. The first-order derivatives of this function, sometimes called **score**, is given by

$$\frac{\partial \log L(\theta)}{\partial \theta} = \begin{pmatrix} -\frac{1}{2\sigma^2}(-2X'Y + 2X'X\beta) \\ -\frac{1}{2\sigma^2}\left(n - \frac{1}{\sigma^2}(Y - X\beta)'(Y - X\beta)\right)\end{pmatrix}.$$

The necessary condition of MLE is $\frac{\partial}{\partial\theta}\log L(\theta) = 0$. This leads to the following MLE:

$$\tilde{\beta} = (X'X)^{-1}(X'Y), \quad \tilde{\sigma}^2 = \frac{\hat{u}'\hat{u}}{n}.$$

The sufficient condition of MLE is the following Hessian matrix is negative definite.

$$H(\theta) = \begin{pmatrix} -\frac{1}{\sigma^2}X'X & \frac{1}{2\sigma^4}(-X'Y + X'X\beta) \\ \frac{1}{2\sigma^4}(-X'Y + X'X\beta) & \frac{n}{2\sigma^4} - \frac{1}{\sigma^6}(Y - X\beta)'(Y - X\beta)\end{pmatrix}.$$

3.3 Properties of MLE

First, we provide the *Cramer-Rao theorem* that states that ML methods gives the lower bound of variance of unbiased estimators (proof is omitted).

> Let $\tilde{\theta}$ denote an unbiased estimator of θ. Then, $V(\tilde{\theta}) - I^{-1}(\theta)$ is a positive definite where $I(\theta)$ is a **Fisher information matrix**, which is defined by
>
> $$I(\theta) = -E(H(\theta)) = -E\begin{pmatrix} \frac{\partial^2 \log L(\theta)}{\partial \theta_1^2} & \frac{\partial^2 \log L(\theta)}{\partial \theta_1 \partial \theta_2} & \cdots & \frac{\partial^2 \log L(\theta)}{\partial \theta_1 \partial \theta_k} \\ \frac{\partial^2 \log L(\theta)}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 \log L(\theta)}{\partial \theta_2^2} & \cdots & \frac{\partial^2 \log L(\theta)}{\partial \theta_2 \partial \theta_k} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial^2 \log L(\theta)}{\partial \theta_k \partial \theta_1} & \frac{\partial^2 \log L(\theta)}{\partial \theta_k \partial \theta_2} & \cdots & \frac{\partial^2 \log L(\theta)}{\partial \theta_k^2} \end{pmatrix}.$$

Note that the Fisher information matrix conditional on some random variables also provides the Cramer-Rao lower bound. In the case of linear regression, the Cramer-Rao lower bound condtional on X gives

$$I^{-1}\begin{pmatrix} \beta \\ \sigma^2 \end{pmatrix} = \begin{pmatrix} \sigma^2 X'X^{-1} & 0 \\ 0 & \frac{2\sigma^4}{n} \end{pmatrix}.$$

Although the ML estimator of β attains the Cramer-Rao lower bound, the ML estimator of σ^2 deviates.

Second, we summarize asymptotic properties of MLE as follows (proof is omitted):

> Under certain regularity conditions, (i) The ML estimator is consistent, i.e., $\tilde{\theta} \xrightarrow{p} \theta$, and (ii) The ML estimator is asymptotically normally distributed, i.e., $\tilde{\theta} \xrightarrow{d} N(\theta, I^{-1}(\theta))$

4 Qualitative Models

4.1 Empirical Application of Binary Model: Titanic Survivors

4.1.1 Background and Data

"Women and children first" is a behavioral norm, which women and children are saved first in a life-threatening situation. This code was made famous by the sinking of the Titanic in 1912. An empirical application investigates characteristics of survivors of Titanic to answer whether crews obeyed the code or not.

We use an open data about Titanic survivors.[1] Number of observations is 1,045. Although this dataset contains many variables, we use only four variables: `survived`, `age`, `fare`, and `sex`. We summarize descriptions of variables as follows:

- `survived`: a binary variable taking 1 if a passenger survived.

[1] data source: http://biostat.mc.vanderbilt.edu/DataSets.

- age: a continuous variable representing passeger's age.
- fare: a continuous variable representing how much passeger paid.
- sex: a string variable representing passenger's sex.

Using sex, we will make a binary variable, called female, taking 1 if passeger is female. Intead of sex, we use female variable in regression.

Moreover, we split data into two subsets: the *training* data and the *test* data. The training data is randomly drawn from the original data. The sample size of this data is two thirs of total observations, that is, $N = 696$. We use the training data (*in-sample*) to estimate and evaluate model fitness. The test data consists of observations which the training data does not include ($N = 349$). We use the test data (*out-of-sample*) to evaluate model prediction.

```r
dt <- read.csv(
  file = "./data/titanic.csv",
  header = TRUE,  sep = ",", row.names = NULL,  stringsAsFactors = FALSE)

dt$female <- ifelse(dt$sex == "female", 1, 0)
dt <- subset(dt, !is.na(survived)&!is.na(age)&!is.na(fare)&!is.na(female))
dt <- dt[,c("survived", "age", "fare", "female")]

set.seed(120511)
train_id <- sample(1:nrow(dt), size = (2/3)*nrow(dt), replace = FALSE)
train_dt <- dt[train_id,]
test_dt <- dt[-train_id,]

head(dt)
```

```
##   survived   age     fare female
## 1        1 29.00 211.3375      1
## 2        1  0.92 151.5500      0
## 3        0  2.00 151.5500      1
## 4        0 30.00 151.5500      0
## 5        0 25.00 151.5500      1
## 6        1 48.00  26.5500      0
```

Model. In a binary model, a dependent (outcome) variable y_i takes only two values, i.e., $y_i \in \{0, 1\}$. A binary variable is sometimes called a *dummy* variable. In this application, the outcome variable is survived. Explanatory variables are female, age, and fare. The regression function is

$$\begin{aligned} & E[survived|female, age, fare] \\ =& \mathbb{P}[survived = 1|female, age, fare] = G(\beta_0 + \beta_1 female + \beta_2 age + \beta_3 fare). \end{aligned}$$

The function $G(\cdot)$ is arbitrary function. In practice, we often use following three specifications:

- Linear probability model (LPM): $G(\mathbf{x}_i \beta) = \mathbf{x}_i \beta$.

- Probit model: $G(\mathbf{x}_i\beta) = \Phi(\mathbf{x}_i\beta)$ where $\Phi(\cdot)$ is the standard Gaussian cumulative function.
- Logit model: $G(\mathbf{x}_i\beta) = 1/(1+\exp(-\mathbf{x}_i\beta))$.

4.1.2 Linear Probability Model

The linear probability model specifys that $G(a)$ is linear in a, that is,

$$\mathbb{P}[survived = 1|female, age, fare] = \beta_0 + \beta_1 female + \beta_2 age + \beta_3 fare.$$

This model can be estimated using the OLS method. In R, we can use the OLS method, running `lm()` function.

```r
model <- survived ~ factor(female) + age + fare
LPM <- lm(model, data = train_dt)
```

The linear probability model is heteroskedastic, that is, $V(u_i|\mathbf{x}_i) = G(\mathbf{x}_i\beta)(1-G(\mathbf{x}_i\beta))$. However, `lm()` function assumes homoskedasticity. To resolve it, we need to claculate heteroskedasticity-robust standard errors using the White method.

$$\hat{V}(\hat{\beta}) = \left(\frac{1}{n}\sum_i \mathbf{x}_i'\mathbf{x}_i\right)^{-1}\left(\frac{1}{n}\sum_i \hat{u}_i^2\mathbf{x}_i'\mathbf{x}_i\right)\left(\frac{1}{n}\sum_i \mathbf{x}_i'\mathbf{x}_i\right)^{-1}$$

where $\hat{u}_i = y_i - G(\mathbf{x}_i\hat{\beta})$.

```r
# heteroskedasticity-robust standard errors
train_dt$"(Intercept)" <- 1
X <- as.matrix(train_dt[,c("(Intercept)", "female", "age", "fare")])
u <- diag(LPM$residuals^2)

XX <- t(X) %*% X
avgXX <- XX * nrow(X)^{-1}
inv_avgXX <- solve(avgXX)

uXX <- t(X) %*% u %*% X
avguXX <- uXX * nrow(X)^{-1}

vcov_b <- (inv_avgXX %*% avguXX %*% inv_avgXX) * nrow(X)^{-1}
rse_b <- sqrt(diag(vcov_b))

label <- c("(Intercept)", "factor(female)1", "age", "fare")
names(rse_b) <- label

# homoskedasticity-based standard errors
se_b <- sqrt(diag(vcov(LPM)))

print("The Variance of OLS"); vcov(LPM)
```

```
## [1] "The Variance of OLS"
```

```
##                   (Intercept) factor(female)1           age          fare
## (Intercept)      1.505787e-03   -3.905773e-04 -3.676396e-05 -5.951346e-07
## factor(female)1 -3.905773e-04    1.089299e-03  2.569835e-06 -2.154400e-06
## age             -3.676396e-05    2.569835e-06  1.264948e-06 -6.274261e-08
## fare            -5.951346e-07   -2.154400e-06 -6.274261e-08  9.167801e-08
```

```r
print("The Robust variance of OLS"); vcov_b
```

```
## [1] "The Robust variance of OLS"
```

```
##               (Intercept)        female           age          fare
## (Intercept)  1.810499e-03 -3.968956e-04 -4.601203e-05  8.979498e-07
## female      -3.968956e-04  1.239665e-03  4.975911e-06 -4.566026e-06
## age         -4.601203e-05  4.975911e-06  1.476806e-06 -7.956793e-08
## fare         8.979498e-07 -4.566026e-06 -7.956793e-08  7.846876e-08
```

```r
print("The Robust se using White method"); rse_b
```

```
## [1] "The Robust se using White method"
```

```
##     (Intercept) factor(female)1             age            fare
##    0.0425499596    0.0352088828    0.0012152389    0.0002801228
```

Using the package `lmtest` and `sandwich` is the easiest way to calculate heteroskedasticity-robust standard errors.

```r
library(lmtest) #use function `coeftest`
library(sandwich) #use function `vcovHC`
coeftest(LPM, vcov = vcovHC(LPM, type = "HC0"))[, "Std. Error"]
```

```
##     (Intercept) factor(female)1             age            fare
##    0.0425499596    0.0352088828    0.0012152389    0.0002801228
```

Finally, we summarize results of linear probability model in table 1. We will discuss interpretation of results and goodness-of-fit of LPM later.

```r
library(stargazer)
stargazer(
  LPM, LPM,
  se = list(se_b, rse_b),
  t.auto = FALSE, p.auto = FALSE,
  report = "vcs", keep.stat = c("n"),
  covariate.labels = c("Female = 1"),
  add.lines = list(
    c("Standard errors", "Homoskedasticity-based", "Heteroskedasticity-robust")),
  title = "Results of Linear Probability Model", label = "LPM",
  type = "latex", header = FALSE, font.size = "small",
  omit.table.layout = "n", table.placement = "h"
)
```

Table 1: Results of Linear Probability Model

	Dependent variable:	
	survived	
	(1)	(2)
Female = 1	0.512	0.512
	(0.033)	(0.035)
age	−0.003	−0.003
	(0.001)	(0.001)
fare	0.001	0.001
	(0.0003)	(0.0003)
Constant	0.245	0.245
	(0.039)	(0.043)
Standard errors	Homoskedasticity-based	Heteroskedasticity-robust
Observations	696	696

4.1.3 Probit and Logit Model

Unlike LPM, the probit and logit model must be estimated using the ML method. The probability of observing y_i is

$$p_\beta(y_i|\mathbf{x}_i) = \mathbb{P}(y_i = 1|x_i)^{y_i}[1 - \mathbb{P}(y_i = 1|x_i)]^{1-y_i} = G(\mathbf{x}_i\beta)^{y_i}(1 - G(\mathbf{x}_i\beta))^{1-y_i}.$$

Taking logalithm yields

$$\log p_\beta(y_i|\mathbf{x}_i) = y_i \log(G(\mathbf{x}_i\beta)) + (1 - y_i)\log(1 - G(\mathbf{x}_i\beta)).$$

The log-likelihood is

$$M_n(\beta) = \sum_{i=1}^{n} \log p_\beta(y_i|\mathbf{x}_i).$$

The MLE $\hat{\beta}$ holds that the score, which is the first-order derivatives with respect to β, is equal to 0. That is $\nabla_\beta M_n(\hat{\beta}) = 0$. For both logit and probit model, the Hessian matrix, $\nabla^2_{\beta\beta'} M_n(\beta)$, is always negative definite. This implies that log-likelihood function based on both models is grobally concave, and ensures that the MLE maximizes the log-likelihood function. The first-order condition of the probit model is

$$\nabla_\beta M_n(\hat{\beta}) = \sum_{i=1}^{n} \left(y_i - \Phi(\mathbf{x}_i\hat{\beta})\right) \frac{\phi(\mathbf{x}_i\hat{\beta})}{\Phi(\mathbf{x}_i\hat{\beta})(1 - \phi(\mathbf{x}_i\hat{\beta}))} = 0.$$

The first-order condition of the logit model is

$$\nabla_\beta M_n(\hat{\beta}) = \sum_{i=1}^{n} \left(y_i - G(\mathbf{x}_i \hat{\beta}) \right) \mathbf{x}_i' = 0.$$

Since it is hard for us to solve this condition analytically, we obtain estimators using numerical procedure.

The asymptotic distribution of $\hat{\beta}$ is $\hat{\beta} \xrightarrow{d} N(\beta, \Sigma_\beta)$ where

$$\Sigma_\beta = -\left(\sum_i E[E[\nabla^2_{\beta\beta'} \log p_\beta(y_i|\mathbf{x}_i)|\mathbf{x}_i]] \right)^{-1}.$$

In practice, we replace $E[E[\nabla^2_{\beta\beta'} \log p_\beta(y_i|\mathbf{x}_i)|\mathbf{x}_i]]$ by

$$\frac{1}{n} \sum_i \nabla^2_{\beta\beta'} \log p_{\hat{\beta}}(y_i|\mathbf{x}_i).$$

This implies that

$$\Sigma_\beta = -\left(\sum_i \frac{1}{n} \sum_i \nabla^2_{\beta\beta'} \log p_{\hat{\beta}}(y_i|\mathbf{x}_i) \right)^{-1}.$$

that is,

$$\hat{\Sigma}_\beta = -\left(\sum_i \nabla^2_{\beta\beta'} (\log p_{\hat{\beta}}(y_i|\mathbf{x}_i)) \right)^{-1}.$$

In `R`, there are two ways to estimate probit and logit model. First, the function `nlm()` provides the Newton-Raphson algorithm to minimize the function.[2] To run this function, we need to define the log-likelihood function (`LnLik`) beforehand. Moreover, we must give initial values in augments. In this application, we use OLSE as initial values because we expect to obtain same signs of coefficients as LPM. Another way is to run `glm()` function, which is widely used. Using this function, we do not need to define the log-likelihood function and initial values.

```r
Y <- train_dt$survived
female <- train_dt$female
age <- train_dt$age
fare <- train_dt$fare

# log-likelihood
LnLik <- function(b, model = c("probit", "logit")) {
```

[2] `optim()` function is an another way to minimize the function. Especially, the function `optim(method = "BFGS")` provides the Quasi-Newton algorithm which carries on the spirit of Newton method.

```
  xb <- b[1]+ b[2]*female + b[3]*age + b[4]*fare

  if (model == "probit") {
    L <- pnorm(xb)
  } else {
    L <- 1/(1 + exp(-xb))
  }

  LL_i <- Y * log(L) + (1 - Y) * log(1 - L)
  LL <- -sum(LL_i)

  return(LL)
}

#Newton-Raphson
init <- c(0.169, 0.520, -0.0002, 0.001)
probit <- nlm(LnLik, init, model = "probit", hessian = TRUE)

label <- c("(Intercept)", "factor(female)1", "age", "fare")
names(probit$estimate) <- label
colnames(probit$hessian) <- label; rownames(probit$hessian) <- label

b_probit <- probit$estimate
vcov_probit <- solve(probit$hessian); se_probit <- sqrt(diag(vcov_probit))
LL_probit <- -probit$minimum

#glm function
model <- survived ~ factor(female) + age + fare
probit_glm <- glm(model, data = train_dt, family = binomial("probit"))

#result
print("The MLE of probit model using nlm"); b_probit
```

```
## [1] "The MLE of probit model using nlm"
```

```
##     (Intercept) factor(female)1             age            fare
##    -0.740010404     1.440663450    -0.009316882     0.006302940
```

```
print("The Variance of probit model using nlm"); vcov_probit
```

```
## [1] "The Variance of probit model using nlm"
```

```
##                   (Intercept) factor(female)1           age          fare
## (Intercept)      1.764185e-02   -4.735516e-03 -4.149486e-04 -2.453847e-05
## factor(female)1 -4.735516e-03    1.255295e-02  8.495496e-06 -5.592007e-06
## age             -4.149486e-04    8.495496e-06  1.512962e-05 -9.929199e-07
## fare             -2.453847e-05   -5.592007e-06 -9.929199e-07  1.737151e-06
```

```
print("The se of probit model using nlm"); se_probit
```

```
## [1] "The se of probit model using nlm"

##     (Intercept) factor(female)1             age            fare
##     0.132822608     0.112039969     0.003889681     0.001318010
```

```
print("The coefficients of probit using glm"); coef(probit_glm)
```

```
## [1] "The coefficients of probit using glm"

##     (Intercept) factor(female)1             age            fare
##    -0.740094134     1.440662013    -0.009314690     0.006303577
```

```
print("The se of probit using glm"); sqrt(diag(vcov(probit_glm)))
```

```
## [1] "The se of probit using glm"

##     (Intercept) factor(female)1             age            fare
##     0.134738833     0.112061942     0.003966673     0.001326048
```

Using LogLik, we can also estimate logit model by Newton-Raphson algorithm. To compare result, we also use glm() function.

```
#Newton-Raphson
logit <- nlm(LnLik, init, model = "logit", hessian = TRUE)

label <- c("(Intercept)", "factor(female)1", "age", "fare")
names(logit$estimate) <- label
colnames(logit$hessian) <- label; rownames(logit$hessian) <- label

b_logit <- logit$estimate
vcov_logit <- solve(logit$hessian); se_logit <- sqrt(diag(vcov_logit))
LL_logit <- -logit$minimum

#glm function
logit_glm <- glm(model, data = train_dt, family = binomial("logit"))

#result
print("The MLE of logit model"); b_logit
```

```
## [1] "The MLE of logit model"

##     (Intercept) factor(female)1             age            fare
##     -1.19071868      2.36579523     -0.01665811      0.01049121
```

```
print("The Variance of logit model"); vcov_logit
```

```
## [1] "The Variance of logit model"
##                   (Intercept) factor(female)1           age          fare
## (Intercept)      5.347251e-02   -1.306856e-02 -1.260674e-03 -7.166131e-05
## factor(female)1 -1.306856e-02    3.678907e-02 -4.389835e-05 -2.773805e-06
## age             -1.260674e-03   -4.389835e-05  4.703086e-05 -3.343743e-06
## fare            -7.166131e-05   -2.773805e-06 -3.343743e-06  5.199195e-06
```

```
print("The se of logit model"); se_logit
```

```
## [1] "The se of logit model"
```

```
##     (Intercept) factor(female)1             age            fare
##     0.231241234     0.191804780     0.006857905     0.002280174
```

```
print("The coefficients of logit using glm"); coef(logit_glm)
```

```
## [1] "The coefficients of logit using glm"
```

```
##     (Intercept) factor(female)1             age            fare
##     -1.19080405      2.36579304     -0.01665588      0.01049185
```

```
print("The se of logit using glm"); sqrt(diag(vcov(logit_glm)))
```

```
## [1] "The se of logit using glm"
```

```
##     (Intercept) factor(female)1             age            fare
##     0.231133819     0.191810415     0.006862245     0.002272391
```

As a result, table 2 summarizes results of probit model and logit model. Standard errors are in parentheses. We will discuss interpretation of results and goodness-of-fit later.

```
stargazer(
  probit_glm, logit_glm,
  coef = list(b_probit, b_logit), se = list(se_probit, se_logit),
  t.auto = FALSE, p.auto = FALSE,
  report = "vcs", keep.stat = c("n"),
  covariate.labels = c("Female = 1"),
  add.lines = list(
    c("Log-Likelihood", round(LL_probit, 3), round(LL_logit, 3))),
  title = "Results of Probit and Logit model",
  label = "probit_logit",
  type = "latex", header = FALSE, font.size = "small",
  table.placement = "h", omit.table.layout = "n"
)
```

4.1.4 Interpretaions

In the linear probability model, interepretations of coefficients are straight-forward. The coefficient β_1 is the change in survival probability given a one-unit increase in continuous variable x. In the case of discrete variable, the coefficient β_1 is the difference in survival

Table 2: Results of Probit and Logit model

	Dependent variable:	
	survived	
	probit	*logistic*
	(1)	(2)
Female = 1	1.441	2.366
	(0.112)	(0.192)
age	−0.009	−0.017
	(0.004)	(0.007)
fare	0.006	0.010
	(0.001)	(0.002)
Constant	−0.740	−1.191
	(0.133)	(0.231)
Log-Likelihood	-351.507	-351.873
Observations	696	696

probability between two groups.

However, when we use the probit or logit model, it is hard for us to interepret results because the partial effect is not constant across other covariates. As an illustration, the partial effect of continuous variable `age` is

$$\partial_{age}\mathbb{P}[survived = 1|female, age, fare] = \begin{cases} \beta_2 & \text{if LPM} \\ \phi(\mathbf{x}_i\beta)\beta_2 & \text{if Probit} \\ \frac{\exp(-\mathbf{x}_i\beta)}{(1+\exp(-\mathbf{x}_i\beta))^2}\beta_2 & \text{if Logit} \end{cases}.$$

The partial effect of dummy variable `female` is

$$\begin{aligned} &\mathbb{P}[survived = 1|female = 1, age, fare] - \mathbb{P}[survived = 1|female = 0, age, fare] \\ &= \begin{cases} \beta_1 & \text{if LPM} \\ \Phi(\beta_0 + \beta_1 + \beta_2 age + \beta_3 fare) - \Phi(\beta_0 + \beta_2 age + \beta_3 fare) & \text{if Probit} \\ \Lambda(\beta_0 + \beta_1 + \beta_2 age + \beta_3 fare) - \Lambda(\beta_0 + \beta_2 age + \beta_3 fare) & \text{if Logit} \end{cases}, \end{aligned}$$

where $\Lambda(a) = 1/(1 + \exp(-a))$.

Table 3 shows results of linear probability model, probit model, and logit model. Qualitatively, all specifications shows same trend. The survival probability of females is greater than of male. The survival probability is decreaseing in age. Quantitatively, LPM shows that the survival probability of female is about 50% point higher than of male. Moreover, the survival probability of 0-year-old baby is about 0.3% point less than of 100-year-old elderly. This implies that the survival probability is not largely changed by age. To evaluate probit and logit model quantitatively, consider 'average' person with respect to `age` and `fare`. Average age is about 30, and average fare is about 37. Then, the survival probability of female is calculated as follows:

```r
#probit
cval_p <- b_probit[1] + 30*b_probit[3] + 37*b_probit[4]
female_p <- pnorm(cval_p + b_probit[2]) - pnorm(cval_p)
#logit
cval_l <- b_logit[1] + 30*b_logit[3] + 37*b_logit[4]
female_l <- 1/(1 + exp(-(cval_l + b_logit[2]))) - 1/(1 + exp(-cval_l))
# result
print("Probit: Diff of prob. b/w average female and male"); female_p
```

```
## [1] "Probit: Diff of prob. b/w average female and male"

## (Intercept)
##    0.527715
```

```r
print("Logit: Diff of prob. b/w average female and male"); female_l
```

```
## [1] "Logit: Diff of prob. b/w average female and male"

## (Intercept)
##     0.52958
```

As a result, in terms of the difference of survival probability between females and males the probit and logit model obtain similar result to LPM. In the same way, we can calculate the partial effect of age in the probit and logit model, but we skip this. If you have an interest, please try yourself. Overall, crews obeyed the code of "women and children first", but the survival probability of children is not largely different from of adult.

4.1.5 Model Fitness

There are two measurements of goodness-of-fit. First, the *percent correctly predicted* reports the percentage of unit whose predicted y_i matches the actual y_i. The predicted y_i takes one if $G(\mathbf{x}_i\hat{\beta}) > 0.5$, and takes zero if $G(\mathbf{x}_i\hat{\beta}) \leq 0.5$. We calculate this index, using the training data and the test data.

```r
# In-sample
in_Y <- train_dt$survived
in_X <- as.matrix(train_dt[,c("(Intercept)", "female", "age", "fare")])

in_Xb_lpm <- in_X %*% matrix(coef(LPM), ncol = 1)
in_Xb_probit <- in_X %*% matrix(b_probit, ncol = 1)
in_Xb_logit <- in_X %*% matrix(b_logit, ncol = 1)
```

```
in_hatY_lpm <- ifelse(in_Xb_lpm > 0.5, 1, 0)
in_hatY_probit <- ifelse(pnorm(in_Xb_probit) > 0.5, 1, 0)
in_hatY_logit <- ifelse(1/(1 + exp(-in_Xb_logit)) > 0.5, 1, 0)

in_pcp_lpm <- round(sum(in_Y == in_hatY_lpm)/nrow(in_X), 4)
in_pcp_probit <- round(sum(in_Y == in_hatY_probit)/nrow(in_X), 4)
in_pcp_logit <- round(sum(in_Y == in_hatY_logit)/nrow(in_X), 4)

# Out-of-sample
out_Y <- test_dt$survived
test_dt$"(Intercept)" <- 1
out_X <- as.matrix(test_dt[,c("(Intercept)", "female", "age", "fare")])

out_Xb_lpm <- out_X %*% matrix(coef(LPM), ncol = 1)
out_Xb_probit <- out_X %*% matrix(b_probit, ncol = 1)
out_Xb_logit <- out_X %*% matrix(b_logit, ncol = 1)

out_hatY_lpm <- ifelse(out_Xb_lpm > 0.5, 1, 0)
out_hatY_probit <- ifelse(pnorm(out_Xb_probit) > 0.5, 1, 0)
out_hatY_logit <- ifelse(1/(1 + exp(-out_Xb_logit)) > 0.5, 1, 0)

out_pcp_lpm <- round(sum(out_Y == out_hatY_lpm)/nrow(out_X), 4)
out_pcp_probit <- round(sum(out_Y == out_hatY_probit)/nrow(out_X), 4)
out_pcp_logit <- round(sum(out_Y == out_hatY_logit)/nrow(out_X), 4)
```

Second measurement is the *pseudo R-squared*. The pseudo R-squared is obtained by $1 - \sum_i \hat{u}_i^2 / \sum_i y_i^2$, where $\hat{u}_i = y_i - G(\mathbf{x}_i \hat{\beta})$.

```
Y2 <- in_Y^2

hatu_lpm <- (in_Y - in_Xb_lpm)^2
hatu_probit <- (in_Y - pnorm(in_Xb_probit))^2
hatu_logit <- (in_Y - 1/(1 + exp(-in_Xb_logit)))^2

pr2_lpm <- round(1 - sum(hatu_lpm)/sum(Y2), 4)
pr2_probit <- round(1 - sum(hatu_probit)/sum(Y2), 4)
pr2_logit <- round(1 - sum(hatu_logit)/sum(Y2), 4)
```

Table 3 summarizes two measurements of model fitness. There is little difference among LPM, probit model, and logit model.

```
stargazer(
  LPM, probit_glm, logit_glm,
  coef = list(coef(LPM), b_probit, b_logit),
  se = list(rse_b, se_probit, se_logit),
```

Table 3: Titanic Survivors: LPM, Probit, and Logit

	Dependent variable:		
	survived		
	OLS	*probit*	*logistic*
	(1)	(2)	(3)
Female = 1	0.512	1.441	2.366
	(0.035)	(0.112)	(0.192)
age	−0.003	−0.009	−0.017
	(0.001)	(0.004)	(0.007)
fare	0.001	0.006	0.010
	(0.0003)	(0.001)	(0.002)
Percent correctly predicted (in-sample)	0.7802	0.7744	0.7744
Percent correctly predicted (out-of-sample)	0.7794	0.7765	0.7765
Pseudo R-squared	0.5869	0.5873	0.5869
Observations	696	696	696

```
  t.auto = FALSE, p.auto = FALSE,
  omit = c("Constant"), covariate.labels = c("Female = 1"),
  report = "vcs", keep.stat = c("n"),
  add.lines = list(
    c("Percent correctly predicted (in-sample)",
      in_pcp_lpm, in_pcp_probit, in_pcp_logit),
    c("Percent correctly predicted (out-of-sample)",
      out_pcp_lpm, out_pcp_probit, out_pcp_logit),
    c("Pseudo R-squared", pr2_lpm, pr2_probit, pr2_logit)
  ),
  omit.table.layout = "n", table.placement = "t",
  title = "Titanic Survivors: LPM, Probit, and Logit",
  label = "titanic",
  type = "latex", header = FALSE
)
```

4.2 Empirical Application of Ordered Probit and Logit Model: Housing as Status Goods

4.2.1 Background and Data

A desire to signal high income or wealth may cause consumers to purchase status goods such as luxury cars. In this application, we explore whether housing serves as status goods, using the case of apartment building. We investigate the relationship between living in a high floor and income, controlling the quality of housing. Our hypothesis is that high-earners are more likely to live on the upper floor.

We use the housing data originally coming from the American Housing Survey conducted in 2013.[3] This dataset (hereafter `housing`) contains the following variables:

- `Level`: ordered value of floor where respondent lives (1:Low - 4:High)
- `lnPrice`: logged price of housing (proxy for quality of house)
- `Top25`: a dummy variable taking one if household income is in the top 25 percentile in sample.

We split data into two subsets: the *training* data and the *test* data. The training data, which is used for estimation and model fitness, is randoly drawn from the original data. The sample size of this subset is two thirds of total observations of the original one ($N = 1,074$). The test data, which is used for model prediction, consists of observations which the training data does not include ($N = 538$).

```r
dt <- read.csv(file = "./data/housing.csv", header = TRUE,  sep = ",")
dt <- dt[,c("Level", "lnPrice", "Top25")]
dt$Level <- factor(dt$Level)

set.seed(120511)
train_id <- sample(1:nrow(dt), size = (2/3)*nrow(dt), replace = FALSE)
train_dt <- dt[train_id,]; test_dt <- dt[-train_id,]

summary(train_dt)
```

```
##  Level       lnPrice            Top25
##  1:404   Min.   : 0.6931   Min.   :0.0000
##  2:165   1st Qu.:11.2898   1st Qu.:0.0000
##  3:269   Median :11.8494   Median :0.0000
##  4:236   Mean   :11.6353   Mean   :0.2393
##          3rd Qu.:12.4292   3rd Qu.:0.0000
##          Max.   :14.0931   Max.   :1.0000
```

[3] https://www.census.gov/programs-surveys/ahs.html. This is a repeated cross-section survey. We use the data at one time.

4.2.2 Model

The outcome variable is `Level` taking $\{1, 2, 3, 4\}$. Consider the following regression equation of a latent variable:

$$y_i^* = \mathbf{x}_i\beta + u_i,$$

where a vector of explanatory variables are `lnPrice` and `Top25`, and u_i is an error term. The relationship between the latent variable y_i^* and the observed outcome variable is

$$Level = \begin{cases} 1 & \text{if} \quad -\infty < y_i^* \leq a_1 \\ 2 & \text{if} \quad a_1 < y_i^* \leq a_2 \\ 3 & \text{if} \quad a_2 < y_i^* \leq a_3 \\ 4 & \text{if} \quad a_3 < y_i^* < +\infty \end{cases} .$$

Consider the probability of realization of y_i, that is,

$$\begin{aligned} \mathbb{P}(y_i = k|\mathbf{x}_i) &= \mathbb{P}(a_{k-1} - \mathbf{x}_i\beta < u_i \leq a_k - \mathbf{x}_i\beta|\mathbf{x}_i) \\ &= G(a_k - \mathbf{x}_i\beta) - G(a_{k-1} - \mathbf{x}_i\beta), \end{aligned}$$

where $a_4 = +\infty$ and $a_0 = -\infty$. Then, the likelihood function is defined by

$$p((y_i|\mathbf{x}_i), i = 1, \dots, n; \beta, a_1, \dots, a_3) = \prod_{i=1}^{n}\prod_{k=1}^{4}(G(a_k - \mathbf{x}_i\beta) - G(a_{k-1} - \mathbf{x}_i\beta))^{I_{ik}}.$$

where I_{ik} is a indicator variable taking 1 if $y_i = k$. Finally, the log-likelihood function is

$$M(\beta, a_1, a_2, a_3) = \sum_{i=1}^{n}\sum_{k=1}^{4} I_{ik}\log(G(a_k - \mathbf{x}_i\beta) - G(a_{k-1} - \mathbf{x}_i\beta)).$$

Usually, $G(a)$ assumes the standard normal distribution, $\Phi(a)$, or the logistic distribution, $1/(1 + \exp(-a))$. We estimate $\theta = (\beta, a_1, a_2, a_3)'$ to minimize the log-likelihood function, that is,

$$\hat{\theta} \in \underset{(\beta, a_1, a_2, a_3)}{\arg\min}\ M(\beta, a_1, a_2, a_3).$$

In `R`, the library (package) `MASS` provides the `polr` function which estimates the ordered probit and logit model. Although we can use the `nlm` function when we define the log-likelihood function, we do not report this method.

```r
library(MASS)

model <- Level ~ lnPrice + Top25
oprobit <- polr(model, data = train_dt, method = "probit")
ologit <- polr(model, data = train_dt, method = "logistic")
```

4.2.3 Interepretation and Model Fitness

Table 4 shows results. In both models, the latent variable y_i^* is increasing in `Top25`. This means that high-earners have higer value of latent variable y_i^*. Since the cutoff values are increasing in the observed y_i, we can conclude that high-earners are more likely to live on the upper floor.

To evaluate model fitness, we use the *percent correctly predicted*, which is the percentage of unit whose predicted y_i matches the actual y_i. First, we calculate $\mathbf{x}_i\hat{\beta}$. If this value is in $(-\infty, \hat{a}_1]$, $(\hat{a}_1, a_2]$, $(\hat{a}_2, \hat{a}_3]$, and $(\hat{a}_3, +\infty)$, then we take $\hat{y}_i = 1$, $\hat{y}_i = 2$, $\hat{y}_i = 3$ and $\hat{y}_i = 4$, respectively. Using the training data (in-sample) and the test data (out-of-sample), we calculate this index.

```r
library(tidyverse) #use case_when()
# coefficients
bp <- matrix(coef(oprobit), nrow = 2); bl <- matrix(coef(ologit), nrow = 2)
# cutoff value
ap <- oprobit$zeta; al <- ologit$zeta
seap <- sqrt(diag(vcov(oprobit)))[3:5]; seal <- sqrt(diag(vcov(ologit)))[3:5]
# in-sample prediction
indt <- as.matrix(train_dt[,c("lnPrice", "Top25")])
in_xbp <- indt %*% bp; in_xbl <- indt %*% bl

in_hatYp <- case_when(
  in_xbp <= ap[1] ~ 1,
  in_xbp <= ap[2] ~ 2,
  in_xbp <= ap[3] ~ 3,
  TRUE ~ 4
)

in_hatYl <- case_when(
  in_xbl <= al[1] ~ 1,
  in_xbl <= al[2] ~ 2,
  in_xbl <= al[3] ~ 3,
  TRUE ~ 4
)

inpred_p <- round(sum(train_dt$Level == in_hatYp)/nrow(train_dt), 3)
inpred_l <- round(sum(train_dt$Level == in_hatYl)/nrow(train_dt), 3)

# out-of-sample prediction
outdt <- as.matrix(test_dt[,c("lnPrice", "Top25")])
out_xbp <- outdt %*% bp; out_xbl <- outdt %*% bl

out_hatYp <- case_when(
  out_xbp <= ap[1] ~ 1,
  out_xbp <= ap[2] ~ 2,
  out_xbp <= ap[3] ~ 3,
  TRUE ~ 4
)

out_hatYl <- case_when(
  out_xbl <= al[1] ~ 1,
  out_xbl <= al[2] ~ 2,
  out_xbl <= al[3] ~ 3,
  TRUE ~ 4
)

outpred_p <- round(sum(test_dt$Level == out_hatYp)/nrow(test_dt), 3)
outpred_l <- round(sum(test_dt$Level == out_hatYl)/nrow(test_dt), 3)
```

As a result, the percent correctly predicted is almost 16% when we use the in-sample data. When we use the test data, this index slightly increases. Overall, out model seems not to be good because the percent correctly predicted is low.

```r
seap <- sprintf("(%1.3f)", seap); seal <- sprintf("(%1.3f)", seal)

library(stargazer)
stargazer(
  oprobit, ologit,
  report = "vcs", keep.stat = c("n"),
  omit = c("Constant"),
  add.lines = list(
    c("Cutoff value at 1|2", round(ap[1], 3), round(al[1], 3)),
    c("", seap[1], seal[1]),
    c("Cutoff value at 2|3", round(ap[2], 3), round(al[2], 3)),
    c("", seap[2], seal[2]),
    c("Cutoff value at 3|4", round(ap[3], 3), round(al[3], 3)),
    c("", seap[3], seal[3]),
    c("Percent correctly predicted (in-sample)", inpred_p, inpred_l),
    c("Percent correctly predicted (out-of-sample)", outpred_p, outpred_l)
  ),
  omit.table.layout = "n", table.placement = "t",
  title = "Floor Level of House: Ordered Probit and Logit Model",
  label = "housing",
  type = "latex", header = FALSE
)
```

Table 4: Floor Level of House: Ordered Probit and Logit Model

	Dependent variable:	
	Level	
	ordered probit	*ordered logistic*
	(1)	(2)
lnPrice	−0.007	−0.013
	(0.019)	(0.031)
Top25	0.133	0.202
	(0.080)	(0.132)
Cutoff value at 1|2	-0.371	-0.611
	(0.227)	(0.363)
Cutoff value at 2|3	0.02	0.014
	(0.226)	(0.362)
Cutoff value at 3|4	0.719	1.163
	(0.227)	(0.365)
Percent correctly predicted (in-sample)	0.161	0.161
Percent correctly predicted (out-of-sample)	0.175	0.175
Observations	1,074	1,074

4.3 Empirical Application of Multinomial Model: Gender Discremination in Job Position

4.3.1 Background and Data

Recently, many developed countries move toward women's social advancement, for example, an increase of number of board member. In this application, we explore whether the gender discremination existed in the U.S. bank industry. Our hypothesis is that women are less likely to be given a higher position than male.

We use a built-in dataset called `BankWages` in the library `AER`. This datase contains the following variables:

- `job`: three job position. The rank of position is `custodial < admin < manage`.
- `education`: years of education
- `gender`: a dummy variable of female

Again, we split data into two subsets: the *training* data and the *test* data. The training data, which is used for estimation and model fitness, is randoly drawn from the original data. The sample size of this subset is two thirds of total observations of the original one ($N = 316$). The test data, which is used for model prediction, consists of observations which the training data does not include ($N = 158$).

To use the multinomial logit model in R, we need to transform outcome variable into the form factor, which is special variable form in R. The variable form factor is similar to dummy variables. For example, factor(dt$job, levels = c("admin", "custodial", "manage")) transforms the variable form job from the form character into the form factor. Moreover, when we use job as explanatory variables, R automatically makes two dummy variables of custodial and manage.

```r
library(AER)
data(BankWages)
dt <- BankWages
dt$job <- as.character(dt$job)
dt$job <- factor(dt$job, levels = c("admin", "custodial", "manage"))
dt <- dt[,c("job", "education", "gender")]

set.seed(120511)
train_id <- sample(1:nrow(dt), size = (2/3)*nrow(dt), replace = FALSE)
train_dt <- dt[train_id,]; test_dt <- dt[-train_id,]

summary(train_dt)
```

```
##          job        education        gender   
##  admin    :240   Min.   : 8.00   male  :178  
##  custodial: 18   1st Qu.:12.00   female:138  
##  manage   : 58   Median :12.00               
##                  Mean   :13.52               
##                  3rd Qu.:15.00               
##                  Max.   :21.00               
```

4.3.2 Model

The outcome variable y_i takes three values $\{0, 1, 2\}$. Note that the labelling of the choices is arbitrary. Then, the multinomial logit model has the following response probabilities

$$P_{ij} = \mathbb{P}(y_i = j|\mathbf{x}_i) = \begin{cases} \frac{\exp(\mathbf{x}_i \beta_j)}{1+\sum_{k=1}^{2} \exp(\mathbf{x}_i \beta_k)} & \text{if} \quad j = 1, 2 \\ \frac{1}{1+\sum_{k=1}^{2} \exp(\mathbf{x}_i \beta_k)} & \text{if} \quad j = 0 \end{cases}.$$

The log-likelihood function is

$$M_n(\beta_1, \beta_2) = \sum_{i=1}^{n} \sum_{j=0}^{3} d_{ij} \log(P_{ij}),$$

where d_{ij} is a dummy variable taking 1 if $y_i = j$.

In R, some packages provide the multinomial logit model. In this application, we use the multinom function in the library nnet.

```r
library(nnet)
est_mlogit <- multinom(job ~ education + gender, data = train_dt)
```

4.3.3 Interpretations and Model Fitness

Table 5 summarizes the result of multinomial logit model. The coefficient represents the change of $\log(P_{ij}/P_{i0})$ in corresponding covariate beucase the response probabilities yields

$$\frac{P_{ij}}{P_{i0}} = \exp(\mathbf{x}_i\beta_j) \Leftrightarrow \log\left(\frac{P_{ij}}{P_{i0}}\right) = \mathbf{x}_i\beta_j.$$

For example, eduction decreases the log-odds between `custodial` and `admin` by -0.562. This implies that those who received higher education are more likely to obtain the position `admin`. Highly-educated workers are also more likely to obtain the position `manage`. Moreover, a female dummy decrease the log-odds between `manage` and `admin` by -0.748, which implies that females are less likely to obtain higher position `manage`. From this result, we conclude that the U.S. bank disencouraged females to assign higher job position.

To evalue model fitness and prediction, we use two indices: the *pseudo R-squared* and *percent correctly predicted*. The *preudo R-sqaured* is calculated by $1 - L_1/L_0$ where L_1 is the value of log-likelihood for estimated model and L_0 is the value of log-likelihood in the model with only an intercept. `R` snippet for calculation of pseudo R-sqaured is as follows: Note that `nnet:::logLik.multinom()` returns the value of log-likelihood.

```r
loglik1 <- as.numeric(nnet:::logLik.multinom(est_mlogit))
est_mlogit0 <- multinom(job ~ 1, data = train_dt)
loglik0 <- as.numeric(nnet:::logLik.multinom(est_mlogit0))
pr2 <- round(1 - loglik1/loglik0, 3)
```

The second index is the *precent correctly predicted*. The predicted outcome is the outcome with the highest estimated probability. Using the training data (in-sample) and the test data (out-of-sample), we calculate this index. `R` snippet for calculation of this index is as follows.

```r
# in-sample prediction
inpred <- predict(est_mlogit, newdata = train_dt, "probs")
inpred <- colnames(inpred)[apply(inpred, 1, which.max)]
inpcp <- round(sum(inpred == train_dt$job)/length(inpred), 3)
# out-of-sample prediction
outpred <- predict(est_mlogit, newdata = test_dt, "probs")
outpred <- colnames(outpred)[apply(outpred, 1, which.max)]
outpcp <- round(sum(outpred == test_dt$job)/length(outpred), 3)
```

As a result, our model is good in terms of fitness and prediction because the percent correctly predicted is high (83.9% of in-sample data and 88.0% of out-of-sample data), and the pseudo R-sqaured is 0.523.

Table 5: Multinomial Logit Model of Job Position

	Dependent variable:	
	custodial	manage
	(1)	(2)
Education	−0.547	1.322
	(0.116)	(0.229)
Female = 1	−10.507	−0.891
	(31.352)	(0.524)
Constant	4.634	−21.448
	(1.269)	(3.605)
Observations	316	
Percent correctly predicted (in-sample)	0.839	
Percent correctly predicted (out-of-sample)	0.88	
Log-likelihood	-102.964	
Pseudo R-sq	0.523	

```r
stargazer(
  est_mlogit,
  covariate.labels = c("Education", "Female = 1"),
  report = "vcs", omit.stat = c("aic"),
  add.lines = list(
    c("Observations", length(inpred), ""),
    c("Percent correctly predicted (in-sample)", inpcp, ""),
    c("Percent correctly predicted (out-of-sample)", outpcp, ""),
    c("Log-likelihood", round(loglik1, 3), ""),
    c("Pseudo R-sq", pr2, "")
  ),
  omit.table.layout = "n", table.placement = "t",
  title = "Multinomial Logit Model of Job Position",
  label = "job",
  type = "latex", header = FALSE
)
```

4.4 Empirical Application of Truncated Regression: Labor Participation of Married Women (1)

4.4.1 Background and Data

To develop women's social advancement, we should create environment to keep a good balance between work and childcare after marriage. In this application, using the dataset of married women, we explore how much childcare prevents married women to participate in labor market.

Our dataset originally comes from Stata sample data.[4] This dataset contains the following variables:

- `whrs`: Hours of work. This outcome variable is truncated from below at zero.
- `kl6`: the number of preschool children
- `k618`: The number of school-aged children
- `wa`: age
- `we`: The number of years of education

```
dt <- read.csv(file = "./data/labor.csv", header = TRUE,  sep = ",")
summary(dt)
```

```
##       whrs            kl6              k618              wa       
##  Min.   :  12   Min.   :0.0000   Min.   :0.000   Min.   :30.00  
##  1st Qu.: 645   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:35.00  
##  Median :1406   Median :0.0000   Median :1.000   Median :43.50  
##  Mean   :1333   Mean   :0.1733   Mean   :1.313   Mean   :42.79  
##  3rd Qu.:1903   3rd Qu.:0.0000   3rd Qu.:2.000   3rd Qu.:48.75  
##  Max.   :4950   Max.   :2.0000   Max.   :8.000   Max.   :60.00  
##        we       
##  Min.   : 6.00  
##  1st Qu.:12.00  
##  Median :12.00  
##  Mean   :12.64  
##  3rd Qu.:13.75  
##  Max.   :17.00  
```

4.4.2 Model

Since we cannot observe those who could not partiapte in the labor market (`whrs = 0`), we use the truncated regression model. Thus, the selection rule is as follows:

$$\begin{cases} y_i = \mathbf{x}_i \beta + u_i & \text{if } s_i = 1 \\ s_i = 1 & \text{if } a_1 < y_i < a_2 \end{cases}.$$

[4] http://www.stata-press.com/data/r13/laborsub.dt. Because this is dta file, we need to import it, using the `read.dta` function in the library `foreign`. I intentionally remove married women who could not participate in the labor market.

where $u_i \sim N(0, \sigma^2)$. By the distributional assumption, we have $y_i|\mathbf{x}_i \sim N(\mathbf{x}_i\beta, \sigma^2)$. In this application, we set $a_1 = 0$ and $a_2 = +\infty$.

Since we are interested in estimating β, we must condition on $s_i = 1$. The probability density function of y_i conditional on $(x_i, s_i = 1)$ is

$$p_\theta(y_i|\mathbf{x}_i, s_i = 1) = \frac{f(y_i|\mathbf{x}_i)}{\mathbb{P}(s_i = 1|\mathbf{x}_i)}.$$

where $\theta = (\beta, \sigma^2)'$. By the distributional assumption, the conditional distribution of y_i is given by

$$f(y_i|\mathbf{x}_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{y_i - \mathbf{x}_i\beta}{\sigma}\right)^2\right) = \frac{1}{\sigma}\phi\left(\frac{y_i - \mathbf{x}_i\beta}{\sigma}\right),$$

where $\phi(\cdot)$ is the standard normal density function. Moreover, the probability of observation $(s_i = 1)$ is given by

$$\begin{aligned}
\mathbb{P}(s_i = 1|\mathbf{x}_i) &= \mathbb{P}(\mathbf{x}_i\beta + u_i > 0|\mathbf{x}_i) \\
&= \mathbb{P}(u_i/\sigma > -\mathbf{x}_i\beta/\sigma|\mathbf{x}_i) \\
&= 1 - \Phi\left(\frac{y_i - \mathbf{x}_i\beta}{\sigma}\right),
\end{aligned}$$

where $\Phi(\cdot)$ is the standard normal cumulative density function.

Thus, the log-likelihood function is

$$M_n(\theta) = \sum_{i=1}^{n} \log\left(\frac{1}{\sigma}\frac{\phi(\frac{y_i - x_i\beta}{\sigma})}{1 - \Phi(\frac{-x_i\beta}{\sigma})}\right).$$

We provide two ways to estimate truncated regression, using `R`. First way is to define the log-likelihood function directly and minimize its function by `nlm` function. Recall that `nlm` function provides the Newton method to minimize the function. We need to give intial values in argument of this function. To set initial values, we assume that coefficients of explanatory variables are zero. Then, we obtain $y_i|\mathbf{x}_i \sim N(\beta_1, \sigma^2)$. Thus, the initial value of σ, `b[1]` is the standard deviation of `whrs`, and the initial value of β_1, `b[2]` is the mean of `whrs`. Note that these initial values are not unbised estimator.

```
whrs <- dt$whrs
kl6 <- dt$kl6; k618 <- dt$k618
wa <- dt$wa; we <- dt$we

LnLik <- function(b) {
  sigma <- b[1]
  xb <- b[2] + b[3]*kl6 + b[4]*k618 + b[5]*wa + b[6]*we
  condp <- dnorm((whrs - xb)/sigma)/(1 - pnorm(-xb/sigma))
  LL_i <- log(condp/sigma)
  LL <- -sum(LL_i)
  return(LL)
}

init <- c(sd(whrs), mean(whrs), 0, 0, 0, 0)
est.LnLik <- nlm(LnLik, init, hessian = TRUE)
```

Second way is to use the function `truncreg` in the library `truncreg`. We must specify the trucated point, using `point` and `direction` arguments. The `point` argument indicates where the outcome variable is truncated. If `direction = "left"`, the outcome variable is truncated from below at `point`, that is, `point < y`. On the other hand, if `direction = "right"`, the outcome variable is truncated from above at `point`, that is, `y < point`.

```
library(truncreg)
model <- whrs ~ kl6 + k618 + wa + we
est.trunc <- truncreg(
  model, data = dt, point = 0, direction = "left", method = "NR")
se.trunc <- sqrt(diag(vcov(est.trunc)))
```

4.4.3 Interpretations

Table 6 shows results of truncated regression estimated by two methods. As a comparison, we also show the OLS result in column (3). All specifications show that the number of preschool and school-aged children reduces the hours of work. The size of coefficient of the number of preschool and school-aged children become stronger when we use the truncated regression. Note that the size of coeffieient of `#.Preschool Children` estimated by `truncreg` is different from the coefficient estimated by `nlm`.

```
ols <- lm(model, data = dt)
coef.LnLik <- est.LnLik$estimate
se.LnLik <- sqrt(diag(solve(est.LnLik$hessian)))
names(coef.LnLik) <- c("sigma", names(coef(ols)))
names(se.LnLik) <- c("sigma", names(coef(ols)))

library(stargazer)
stargazer(
  ols, ols, ols,
  column.labels = c("Truncated (truncreg)", "Truncated (nlm)", "OLS"),
  coef = list(coef(est.trunc), coef.LnLik[2:6]),
  se = list(se.trunc, se.LnLik[2:6]),
  report = "vcs", keep.stat = c("n"),
  covariate.labels = c(
```

Table 6: Truncated Regression: Labor Market Participation of Married Women

	Dependent variable:		
		whrs	
	Truncated (truncreg)	Truncated (nlm)	OLS
	(1)	(2)	(3)
#.Preschool Children	−803.004	−803.032	−421.482
	(321.361)	(252.803)	(167.973)
#.School-aged Children	−172.875	−172.875	−104.457
	(88.729)	(100.590)	(54.186)
Age	−8.821	−8.821	−4.785
	(14.368)	(14.646)	(9.691)
Education Years	16.529	16.529	9.353
	(46.504)	(46.430)	(31.238)
Constant	1,586.260	1,586.228	1,629.817
	(912.354)	(932.878)	(615.130)
Estimated Sigma	983.726	983.736	
Log-Likelihood	-1200.916	-1200.916	
Observations	150	150	150

```
    "\\#.Preschool Children",
    "\\#.School-aged Children",
    "Age", "Education Years"
  ),
  add.lines = list(
    c("Estimated Sigma",
      round(coef(est.trunc)[6], 3), round(coef.LnLik[1], 3)),
    c("Log-Likelihood",
      round(est.trunc$logLik, 3), round(-est.LnLik$minimum, 3))
  ),
  omit.table.layout = "n", table.placement = "t",
  title = "Truncated Regression: Labor Market Participation of Married Women",
  label = "lfp",
  type = "latex", header = FALSE
)
```

4.5 Empirical Application of Tobit Regression: Labor Participation of Married Women (2)

4.5.1 Background and Data

We continue to investigate the previous research question. We use dataset coming from same source as the previous one. Unlike the previous dataset, we now observe married woment who do not participate in the labor market (`whrs = 0`). Additionally, we introduce the new variable:

- `lfp`: a dummy variable taking 1 if observed unit works.

The previous dataset contains observations with `lfp = 1`. In this application, we use observations with `lfp = 0` to estimate the tobit model.

```r
dt <- read.csv(file = "./data/labor2.csv", header = TRUE,  sep = ",")
summary(dt)
```

```
##       lfp           whrs             kl6             k618              wa
##  Min.   :0.0   Min.   :   0.0   Min.   :0.000   Min.   :0.000   Min.   :30.00
##  1st Qu.:0.0   1st Qu.:   0.0   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:35.00
##  Median :1.0   Median : 406.5   Median :0.000   Median :1.000   Median :43.00
##  Mean   :0.6   Mean   : 799.8   Mean   :0.236   Mean   :1.364   Mean   :42.92
##  3rd Qu.:1.0   3rd Qu.:1599.8   3rd Qu.:0.000   3rd Qu.:2.000   3rd Qu.:49.00
##  Max.   :1.0   Max.   :4950.0   Max.   :3.000   Max.   :8.000   Max.   :60.00
##        we
##  Min.   : 5.00
##  1st Qu.:12.00
##  Median :12.00
##  Mean   :12.35
##  3rd Qu.:13.00
##  Max.   :17.00
```

4.5.2 Model

Our dependent variable is censored from below at zero. The censored data is caused by the corner solution problem. Married women chooses zero labor time if, without any constraint, their optimal labor time is negative. In this case, we should use the tobit model. The tobit model is

$$y_i = \begin{cases} \mathbf{x}_i \beta + u_i & \text{if } y_i > a \\ a & \text{otherwise} \end{cases},$$

where $E(u_i) = 0$ and $\text{Var}(u_i) = 0$. In this application, we set $a = 0$.

Using this model, the probability of y_i conditional on x_i is defined by

$$p_{\beta,\sigma^2}(y_i|x_i) = \mathbb{P}(y_i \leq 0)^{1[y_i=0]} f(y_i|\mathbf{x}_i)^{1-1[y_i=0]}$$

where $f(y_i|x_i)$ is the probability density function conditional on $\mathbf{x}_i$, $1[y_i = 0]$ is an indicator function returing 1 if $y_i = 0$. Now, we assume the distribution $u_i|\mathbf{x}_i \sim N(0, \sigma^2)$. Then, we can reformulate $\mathbb{P}(y_i \leq 0)$ as follows:

$$\mathbb{P}(y_i \leq 0) = \mathbb{P}(-\mathbf{x}_i\beta \leq u_i) = \Phi\left(-\frac{\mathbf{x}_i\beta}{\sigma}\right) = 1 - \Phi\left(\frac{\mathbf{x}_i\beta}{\sigma}\right),$$

where $\Phi(\cdot)$ is the cumulative distribution function of the stadnard normal distribution. Note that the last equatility comes from symmetric property of the standard normal distribution. Moreover, the density function f is reformulated as follows:

$$f(y_i|\mathbf{x}_i) = \frac{1}{\sigma}\phi\left(\frac{y_i - \mathbf{x}_i\beta}{\sigma}\right).$$

Assuming iid sample, we obtain the join probability function as follows:

$$p_{\beta,\sigma^2}((y_i|x_i), i = 1, \dots, n) = \prod_{i=1}^{n} \left(1 - \Phi\left(\frac{\mathbf{x}_i\beta}{\sigma}\right)\right)^{1[y_i=0]} \left(\frac{1}{\sigma}\phi\left(\frac{y_i - \mathbf{x}_i\beta}{\sigma}\right)\right)^{1-1[y_i=0]}.$$

We estimate $\log p_{\beta,\sigma^2}((y_i|x_i), i = 1, \dots, n)$, using the maximum likelihood method. In `R`, there are two ways to implement the tobit regression. First way is to define the log-likelihood function directly and minimize its function by `nlm` function. We need to give intial values in argument of this function. To set initial values, we assume coefficients of explanatory variables are zero. Then, we obtain $y_i|\mathbf{x}_i \sim N(\beta_1, \sigma^2)$ where β_1 is intercept of regression equation. Thus, the initial value of σ, `b[1]` is the standard deviation of `whrs`, and the initial value of β_1, `b[2]` is the mean of `whrs`.

```r
whrs <- dt$whrs
kl6 <- dt$kl6; k618 <- dt$k618
wa <- dt$wa; we <- dt$we

LnLik <- function(b) {
  sigma <- b[1]
  xb <- b[2] + b[3]*kl6 + b[4]*k618 + b[5]*wa + b[6]*we
  Ia <- ifelse(whrs == 0, 1, 0)
  F0 <- 1 - pnorm(xb/sigma)
  fa <- dnorm((whrs - xb)/sigma)/sigma
  LL_i <- Ia * log(F0) + (1 - Ia) * log(fa)
  LL <- -sum(LL_i)
  return(LL)
}
```

```r
init <- c(sd(whrs), mean(whrs), 0, 0, 0, 0)
est.LnLik <- nlm(LnLik, init, hessian = TRUE)
coef.tobitNLM <- est.LnLik$estimate
se.tobitNLM <- sqrt(diag(solve(est.LnLik$hessian)))
```

Second way is to use the function `vglm` in the library `VGAM`. First, we need to declare the tobit distribution (`tobit`), using the `family` augment. The `tobit` function needs the censored point (the value of a) in arguments `Lower` and `Upper`. When you specify `Lower`, the observed outcome is left-censored. On the other hand, when you specify `Upper`, the observed outcome is right-censored. In this application, we set `Lower = 0`.

```r
library(VGAM)
model <- whrs ~ kl6 + k618 + wa + we
tobitVGAM <- vglm(model, family = VGAM::tobit(Lower = 0), data = dt)
coef.tobitVGAM <- coef(tobitVGAM)
coef.tobitVGAM[2] <- exp(coef.tobitVGAM[2])
se.tobitVGAM <- sqrt(diag(vcov(tobitVGAM)))[-2]
```

4.5.3 Interpretations

Table 7 shows results of tobit regression estimated by two methods. As a comparison, we also show the OLS result in column (3). Although all specifications show the same sign of coefficients, size of coefficients of censored regression becomes stronger than of OLSE. As with the truncated regression, the number of preschool and school-aged children reduces the hours of work. Unlike the truncated regression, the relationship between married women's characteristics and labor participation is statistically significant. For example, high educated women increases labor time.

```r
ols <- lm(whrs ~ kl6 + k618 + wa + we, data =dt)
names(coef.tobitNLM) <- c("sigma", names(coef(ols)))
names(se.tobitNLM) <- c("sigma", names(coef(ols)))
names(coef.tobitVGAM) <- c(names(coef(ols))[1], "sigma", names(coef(ols))[-1])
names(se.tobitVGAM) <- names(coef(ols))

stargazer(
  ols, ols, ols,
  column.labels = c("Tobit (vglm)", "Tobit (nlm)", "OLS"),
  coef = list(coef.tobitVGAM[-2], coef.tobitNLM[-1]),
  se = list(se.tobitVGAM, se.tobitNLM[-1]),
  report = "vcs", keep.stat = c("n"),
  covariate.labels = c(
    "\\#.Preschool Children",
    "\\#.School-aged Children",
    "Age", "Education Years"
  ),
```

Table 7: Tobit Regression: Labor Market Participation of Married Women

	Dependent variable:		
		whrs	
	Tobit (vglm)	Tobit (nlm)	OLS
	(1)	(2)	(3)
#.Preschool Children	−827.768	−827.733	−462.123
	(218.507)	(171.275)	(124.677)
#.School-aged Children	−140.017	−140.004	−91.141
	(75.203)	(69.379)	(45.850)
Age	−24.980	−24.973	−13.158
	(13.217)	(12.528)	(8.335)
Education Years	103.694	103.707	53.262
	(41.433)	(41.780)	(26.094)
Constant	588.961	588.488	940.059
	(838.808)	(812.625)	(530.720)
Estimated Sigma	1309.928	1309.914	
Log-Likelihood	-1367.09	-1367.09	
Observations	250	250	250

```
  add.lines = list(
    c("Estimated Sigma",
      round(coef.tobitVGAM[2], 3), round(coef.tobitNLM[1], 3)),
    c("Log-Likelihood",
      round(logLik(tobitVGAM), 3), round(-est.LnLik$minimum, 3))
  ),
  omit.table.layout = "n", table.placement = "t",
  title = "Tobit Regression: Labor Market Participation of Married Women",
  label = "lfp_tobit",
  type = "latex", header = FALSE
)
```

4.6 Empirical Application of Poisson Regression: Demand of Recreation

4.6.1 Background and Data

The Poisson distribution is used for drawing purchasing behavior. Especially, the parameter λ means that preference for goods because the expectation of frequency of purchasing, $E(X)$, is equal to λ (we omit proof here). For example, Tsuyoshi Morioka, a famous marketer contributing the v-shaped recovery of Universal Studio Japan, insists that marketers try to increase the parameter λ.

In this application, using cross-section data about recreational boating trips to Lake Somerville, Texas, in 1980, we investigates who has a high preference for this area. We use the built-in dataset called `RecreationDemand` in the library `AER`. This dataset is based on a survey administered to 2,000 registered leisure boat owners in 23 counties in eastern Texas. We use following four variables:

- `trips`: Number of recreational boating trips.
- `income`: Annual household income of the respondent (in 1,000 USD).
- `ski`: Dummy variable taking 1 if the individual was engaged in water-skiing at the lake
- `userfee`: Dummy variable taking 1 if the individual payed an annual user fee at Lake Somerville?

```r
library(AER)
data("RecreationDemand")
summary(RecreationDemand)
```

```
##      trips           quality       ski          income      userfee
##  Min.   : 0.000   Min.   :0.000   no :417   Min.   :1.000   no :646
##  1st Qu.: 0.000   1st Qu.:0.000   yes:242   1st Qu.:3.000   yes: 13
##  Median : 0.000   Median :0.000             Median :3.000
##  Mean   : 2.244   Mean   :1.419             Mean   :3.853
##  3rd Qu.: 2.000   3rd Qu.:3.000             3rd Qu.:5.000
##  Max.   :88.000   Max.   :5.000             Max.   :9.000
##      costC            costS              costH
##  Min.   :  4.34   Min.   :  4.767   Min.   :  5.70
##  1st Qu.: 28.24   1st Qu.: 33.312   1st Qu.: 28.96
##  Median : 41.19   Median : 47.000   Median : 42.38
##  Mean   : 55.42   Mean   : 59.928   Mean   : 55.99
##  3rd Qu.: 69.67   3rd Qu.: 72.573   3rd Qu.: 68.56
##  Max.   :493.77   Max.   :491.547   Max.   :491.05
```

4.6.2 Model

Let y_i be the number of recreational boating trips. We assume that this variable follows the Poisson distribution conditional co covariates $\mathbf{x}_i$. That is,

$$p_\beta(y_i|\mathbf{x}_i) = \frac{\exp(-\lambda_i)\lambda_i^{y_i}}{y_i!},$$

where $\lambda_i = \exp(\mathbf{x}_i\beta)$. Importantly, λ_i represents the preference for boating trips because

$$E[y_i|\mathbf{x}_i] = \lambda_i = \exp(\mathbf{x}_i\beta).$$

Assuming iid sample, the joint density function is defined by

$$p_\beta((y_i|\mathbf{x}_i), i = 1, \dots, n) = \prod_{i=1}^{n} \frac{\exp(-\lambda_i)\lambda_i^{y_i}}{y_i!}.$$

Thus, the log-likelihood function is

$$M_n(\beta) = \sum_{i=1}^{n}(-\lambda_i + y_i \log \lambda_i - \log y_i!) = \sum_{i=1}^{n}(-\exp(\mathbf{x}_i\beta) + y_i\mathbf{x}_i\beta - \log y_i!).$$

Since the first-order condition (orthogonality condition) is non-linear with respect to β, we apply the Newton-Raphson method to obtain MLE. In `R`, there are two way to implement the Poisson regression. First way is to define the log-likelihood function directly and minimize its function by `nlm` function. We need to give intial values in argument of this function. To set initial values, we assume that coefficients of explanatory variables are zero. Then, we have $E[y_i|\mathbf{x}_i] = \exp(\beta_1) = E[y_i]$ where β_1 is intercept of regression equation. Thus, the initial value of β_1, `b[1]` is $\log E[y_i]$. We replace the expectation of y_i by the mathematical mean of y_i.

```
trips <- RecreationDemand$trips; income <- RecreationDemand$income
ski <- as.integer(RecreationDemand$ski) - 1
userfee <- as.integer(RecreationDemand$userfee) - 1

LnLik <- function(b) {
  xb <- b[1] + b[2]*income + b[3]*ski + b[4]*userfee
  LL_i <- -exp(xb) + trips*xb - log(gamma(trips+1))
  LL <- -sum(LL_i)
  return(LL)
}

init <- c(log(mean(trips)), 0, 0, 0)
poissonMLE <- nlm(LnLik, init, hessian = TRUE)
coef.poissonMLE <- poissonMLE$estimate
se.poissonMLE <- sqrt(diag(solve(poissonMLE$hessian)))
logLik.poissonMLE <- -poissonMLE$minimum
```

The second way is to use `glm` function. To implement this function, we need to specify the Poisson distribution, `poisson()` in the `family` augment. We can obtain the value of log-likelihood function, using the `logLik` function.

```r
model <- trips ~ income + ski + userfee
poissonGLM <- glm(model, family = poisson(), data = RecreationDemand)
logLik.poissonGLM <- as.numeric(logLik(poissonGLM))
```

4.6.3 Interpretations

Table 8 shows results of the Poisson regression estimated by two methods, `nlm` and `glm`. As a comparison, we also show the result of OLS estimation. Clearly, the `nlm` methods (column 1) returns quite similar results to the `glm` method (column 2). Alotough the size of OLSE is farther away from zero than coefficients of the Poisson regression, the sign of OLSE is same as coefficients of the Poisson regression. Surprisingly, we obtain the negative relationship between annual income and preference for boating trips. This implies that high-earners are less likely to go to Lake Somerville.

```r
names(coef.poissonMLE) <- names(coef(poissonGLM))
names(se.poissonMLE) <- names(coef(poissonGLM))
ols <- lm(model, data = RecreationDemand)

stargazer(
  poissonGLM, poissonGLM, ols,
  coef = list(coef.poissonMLE),
  se = list(se.poissonMLE),
  report = "vcs", keep.stat = c("n"),
  covariate.labels = c(
    "Income",
    "1 = Playing water-skiing",
    "1 = Paying annual fee"
  ),
  add.lines = list(
    c("Method", "nlm", "glm", ""),
    c("Log-Likelihood",
      round(logLik.poissonMLE, 3), round(logLik.poissonGLM, 3), "")
  ),
  omit.table.layout = "n", table.placement = "t",
  title = "Poisson Regression: Recreation Demand",
  label = "recreation",
  type = "latex", header = FALSE
)
```

Table 8: Poisson Regression: Recreation Demand

	Dependent variable:		
	trips		
	Poisson		*OLS*
	(1)	(2)	(3)
Income	−0.146	−0.146	−0.277
	(0.017)	(0.017)	(0.133)
1 = Playing water-skiing	0.547	0.547	1.243
	(0.055)	(0.055)	(0.509)
1 = Paying annual fee	1.904	1.904	12.412
	(0.078)	(0.078)	(1.688)
Constant	1.006	1.006	2.609
	(0.065)	(0.065)	(0.545)
Method	nlm	glm	
Log-Likelihood	-2529.256	-2529.256	
Observations	659	659	659

5 Reference

Angrist, Joshua D, and Jörn-Steffen Pischke. 2008. *Mostly Harmless Econometrics: An Empiricist's Companion.* Princeton university press.

Johnston, J. 1984. *Econometric Methods 3rd.* McGraw-Hill book co.

Wasserman, Larry. 2013. *All of Statistics: A Concise Course in Statistical Inference.* Springer Science & Business Media.