TA Session of Econometrics 2 (Oct 2020 - Jan 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 $x_i = (x_{i1}, ... x_{ij})$ 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 $(x_{11}, ... x_{nn})$, and other elements are zero. That is,

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

Especially, a matrix $I = diag(1, 1, ..., 1)$ 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 $(c_1, c_2, ..., c_k)$. 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 $i \in \{1, ..., n\}$. 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, $a'x = a_1x_1 + \cdots + a_nx_n$. 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 $g : \mathbb{R}^n \to \mathbb{R}$ is maximized (minimized) at the point $x^0 = (x_1^0, \ldots, x_n^0)$, 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 $A_1, A_2, ...$ 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 $X_1, \ldots, X_n$ are random variables and $a_1, \ldots, a_n$ are constants. $E(\sum_i a_i X_i) = \sum_i a_i E(X_i)$;
- Suppose that $X_1, \ldots, X_n$ 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 $X_1, \ldots, X_n$ are independent random variables and $a_1, \ldots, a_n$ 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 $X_1, ..., X_n$ 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 $X_1, X_2, ...$ 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 $X_1, X_2, ...$ 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 $i = 1, \ldots, n$, 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 $n = 1, \ldots, 20000$, and show line plot with logged number of trial on x-axis and the proportion of heads on y-axis.

```
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")
```

2.2.3.2 Central Limit Theorem The second important theorm is the **central limit theorem**. Suppose that $X_i, \ldots, X_n$ 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 $X_1, \ldots, X_n$ be IID with mean μ and variance σ^2. Let $\bar{X}_n = n^{-1}\sum_{i=1}^n X_i$. Then,
>
> $$Z_n \equiv \frac{\bar{X}_n - \mu}{\sqrt{V(\bar{X}_n)}} = \sqrt{n}\frac{\bar{X}_n - \mu}{\sigma} \xrightarrow{d} Z,$$
>
> 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 $\sigma^2 =$

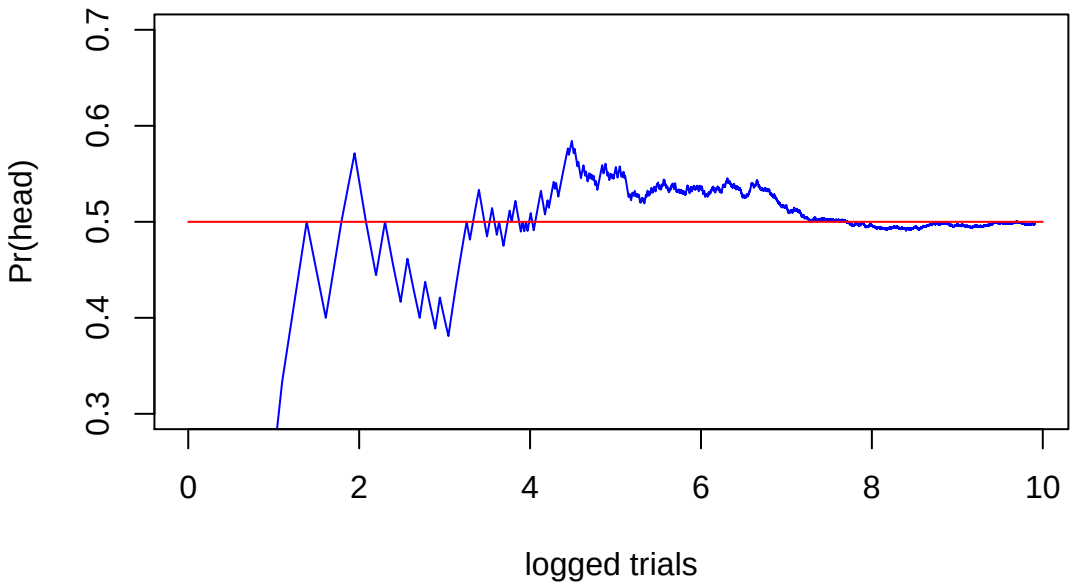

Figure 1: Simulation Result of WLLN

$0.5(1-0.5) = 0.25$. 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.

```
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 = "")
}
```

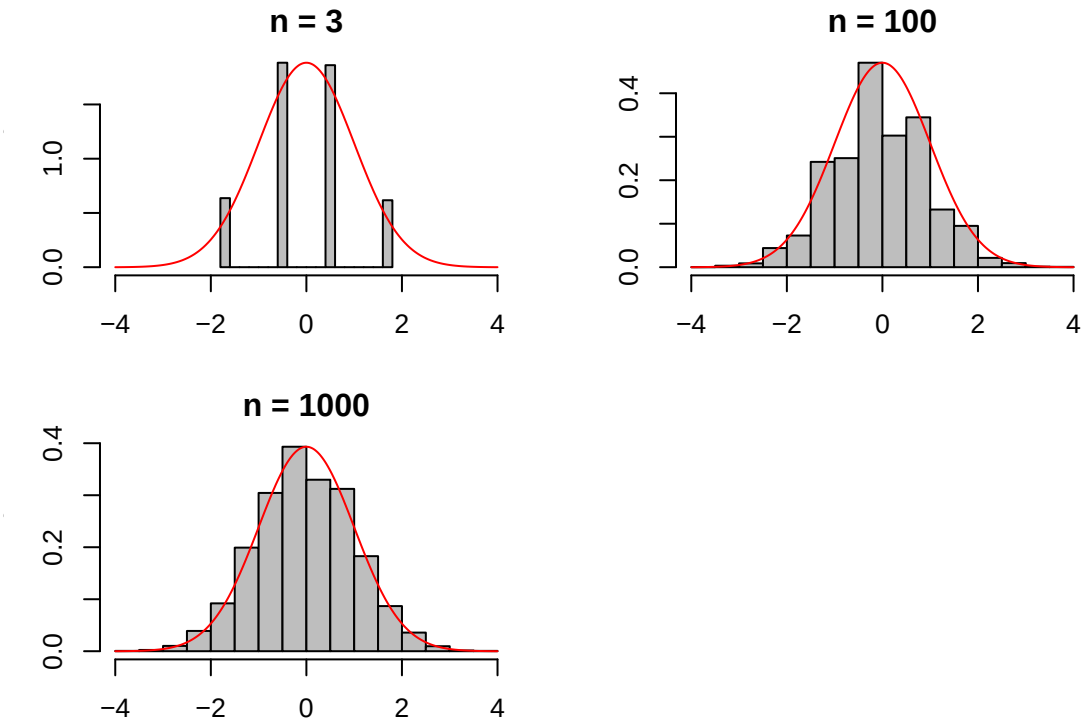

Figure 2: Simulation Result of CLT

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,

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 $Y = (y_1, \ldots, y_n)'$ is a $n \times 1$ vector, and $X = (x_1, \ldots, x_n)'$ 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 $\hat{u}=Y-X\hat{\beta}=(I_n-X(X'X)^{-1}X')u=M_Xu$ and M_X is a symmetric and idempotent matrix. Note that $\hat{\sigma}^2$ is an unbiased estimator of σ^2 since

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

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 $V(R\hat{\beta})=\sigma^2R(X'X)^{-1}R'$. This leads to

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

If the null hypothesis is true, then

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

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

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

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 $R = (0, 1, 0, ..., 0)$ 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.
- `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.

```
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,]
```

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

```r
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.

```
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.

```
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"
)
```

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

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

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.

```
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")) {

  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)
```

[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.

```r
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
```

```r
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
```

```r
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
```

```r
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
```

```r
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
```

```r
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.

```r
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"
)
```

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

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 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:

```
#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
```

```
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 $G(\mathbf{x}_i\widehat{\beta}) > 0.5$, and takes zero if $G(\mathbf{x}_i\widehat{\beta}) \leq 0.5$. We calculate this index, using the training data and the test data.

```
# 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),
  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
)
```

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

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

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

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 ($N = 538$).

```
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
```

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.

```
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 $\mathbf{x}_i\widehat{\beta}$. 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.

```
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.

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

library(stargazer)
stargazer(
  oprobit, ologit,
  report = "vcs", keep.stat = c("n"),
```

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

```
  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
)
```

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`.

```
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`.

```
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.

```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

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	

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
```

[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.

```
##  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:

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

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(
    "\\#.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

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

observations with `lfp = 0` to estimate the tobit model.

```
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:

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

We estimate $\log p_{\beta,\sigma^2}((y_i|x_i), i = 1, \ldots, n)$, 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`.

```
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)
}

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`.

```
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.

```
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"
  ),
  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.

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

- `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?

```
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.

```
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.

```
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", ""),
```

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

```
    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
)
```

5 Generalized Least Squares Method

5.1 OLS Estimation with HAC Estimator

5.1.1 Background and Data

The data we use today originates from Harrison and Rubinfeld (1978) which investigates the methodological problems associated with the use of housing market data to measure the willingness to pay for clean air. It drew a conclusion that marginal air pollution damages are found to increase with the level of air pollution and with household income. But here we

just use some variables of the original data to construct a multiple linear regression model focusing on how the attributes of communities affect housing prices(value) of Boston city.

We use an open data which is called as "Boston Neighboorhood Housing Prices Dataset[5]". Although there exist many variables, we just take 4 of them: `value`, `crime`, `industrial`, `distance`. And here come the descriptions.

- `value`: a continuous variable meaning value of owner-occupied homes in $1000's.
- `crime`: a continuous variable representing per capita crime rate by town.
- `industrial`: a continuous variable showing proportion of non-retail business acres per town.
- `distance`: a continuous variable revealing weighted distances to five Boston employment centres.

As I mentioned before, `value` is the dependent variable and the other three are independent variables. Firstly, let's bulid the dataset in R.

```
library(AER) # Including lmtest and sandwich
library(nlme) # Do gls

# Preparations and preliminary tests

dt = read.csv(
file = "./boston.csv",
header = TRUE, sep = ",", row.names = NULL, stringsAsFactors = FALSE)
# complete.cases returns TURE if one row doesn't contain NA
dt = dt[complete.cases(dt),]
dt = dt[,c("value", "crime", "industrial", "distance")]

n = nrow(dt)
head(dt)
```

```
##    value   crime     industrial distance
## 1  24.0  0.00632       2.31      4.0900
## 2  21.6  0.02731       7.07      4.9671
## 3  34.7  0.02729       7.07      4.9671
## 4  33.4  0.03237       2.18       6.0622
## 5  36.2  0.06905       2.18      6.0622
## 6  28.7  0.02985       2.18      6.0622
```

5.1.2 Model

The multiple regression model is easily written as below.

$$value_i = \beta_0 + \beta_1 crime_i + \beta_2 industrial_i + \beta_3 distance_i + u_i \quad \forall i = 1, \dots, n.$$

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

And don't forget the assumptions.

Assumptions of GLS

A linear regression model with heteroscedastic and autocorrelative error terms is defined as follows.

$$\underline{Y} = \underline{\mathbf{X}} \ + \underline{u}$$

with assumptions:
GH1 $\mathbb{E}[\underline{u}|\underline{\mathbf{X}}] = 0$.
GH2 $Var(\underline{u}|\underline{\mathbf{X}}) = \Omega = \Sigma(\underline{\mathbf{X}}, \theta) \succ 0$.
GH3 $\underline{\mathbf{X}}^T\underline{\mathbf{X}} \succ 0$.

5.1.3 Checking heteroskedasticity and autocorrelation

It is very common for us to apply ols to linear regressions. But with the previous assumptions GH1-GH3, OLS estimators do not own the BLUE property any more. To deal with the relaxations of Guass-Markov assumptions, one can use OLS method with a heteroscedasticity and autocorrelation consistent (HAC) covariance matrix estimator. Before we talk about HAC estimator, let me show you 2 tests which are helpful in checking heteroskedasticity and autocorrelation, respectively. Firstly, the Breusch=Pagan=Godfrey(BPG) test for heteroskedasticity.

The BPG Test

Assume that in a linear regression model, the followings hold.

$$\begin{aligned} y_i &= \mathbf{X}_i \ + u_i \quad u_i|\mathbf{X}_i \sim \mathcal{N}_{\mathbb{R}}(0, \sigma_i^2) \\ \sigma_i^2 &= \mathbb{E}[u_i^2|\mathbf{X}_i] = \alpha_0 + \alpha_1 X_{1i} + \cdots + \alpha_p X_{pi} + v_i \quad \forall i = 1, \ldots, n. \end{aligned}$$

where v_i is a 0-meand and homoscedastic error term which is not correlated with $\mathbf{X}_i$, for all i. And then, the null hypothesis is $H_0 : \alpha_1 = \alpha_2 = \cdots = \alpha_p = 0$(means homoscedasticity). To implement the test, one can use the three-step procedure.
1. Apply OLS in the model

$$y_i = \mathbf{X}_i \ + u_i$$

2. Compute the regression residuals, $\hat{u}_i$, square them, and estimate the auxiliary regression

$$\hat{u}_i^2 = \alpha_0 + \alpha_1 X_{1i} + \cdots + \alpha_p X_{pi} + v_i.$$

It is possible to use different covariates instead of $\mathbf{X}_i$. 3. Multiply the coefficient of determination(R squared) derieved from the auxiliary regression in step 2 by sample size n to obtain the test satistic

$$nR^2 \sim \chi_p^2$$

For more details, please check Breusch and Pagan (1979). Performing `bptest()` function from `lmtest` package. We have the followings.

```
# OLS
model = value ~ crime + industrial + distance
ols = lm(model, data = dt)
ols_summary = summary(ols)

# Breusch=Pagan=Godfrey test against heteroskedasticity.

(bpgtest = bptest(ols, data = dt, studentize = F)) # from lmtest
# If studentize is set to TRUE
# Koenker's studentized version of the test statistic will be used.
```

And the result is returned as:

```
##          Breusch-Pagan test
## data:  ols
## BP = 28.757, df = 3, p-value = 2.519e-06
```

Since the p-value is very small, it's known that the null hypothesis is rejected, there exists heteroscedasticity in this model.

Next, let me introduce the Durbin-Watson test (Durbin and Watson 1950, 1951) to you.

The Durbin-Watson Test

In statistics, the Durbin - Watson statistic is a test statistic used to detect the presence of autocorrelation at lag 1 in the residuals (prediction errors) from a regression analysis. It is named after James Durbin and Geoffrey Watson. Assume that

$$\begin{aligned} y_i &= \mathbf{X}_i \ + u_i \\ u_i &= \rho u_{i-1} + \epsilon_i \quad \forall i = 1, \dots, N. \end{aligned}$$

where ϵ_i is a 0-meand and homoscedastic error term which is not correlated with u_i, for all i. Durbin-Watson statistic states that null hypothesis: $\rho = 0$, alternative hypothesis $\rho \neq 0$, then if $\hat{u}_i$ is the OLS residual, the test statistic is

$$dw = \frac{\sum_{i=2}^{N}(\hat{u}_i - \hat{u}_{i-1})^2}{\sum_{i=1}^{N} \hat{u}_i^2}$$

where N is the number of observations. And there is a useful approximatate equation, $dw = 2(1-\hat{\rho})$ where $\hat{\rho}$ is the sample autocorrelation of the residuals. Besides, dw statistic can be interpreted as follows. With lower and upper critical values given as $dw_{L,\alpha}$ and $dw_{U,\alpha}$, to test for **positive autocorrelation** at significance α,

- If dw $< dw_{L,\alpha}$, there is statistical evidence that the error terms are positively autocorrelated.
- If dw $> dw_{U,\alpha}$, there is **no** statistical evidence that the error terms are positively autocorrelated.
- If $dw_{L,\alpha} <$ dw $< dw_{U,\alpha}$, the test is inconclusive.

And to test for **negative autocorrelation** at significance α,

- If 4-dw $< dw_{L,\alpha}$, there is statistical evidence that the error terms are negatively autocorrelated.
- If 4-dw $> dw_{U,\alpha}$, there is **no** statistical evidence that the error terms are negatively autocorrelated.
- If $dw_{L,\alpha} <$ 4-dw $< dw_{U,\alpha}$, the test is inconclusive.

Performing `dwtest()` function from `lmtest` package. We have the followings.

```
# Durbin-Watson test for autocorrelation of disturbances.
# alternative = c("greater", "two.sided", "less")

(dwtest = dwtest(ols, data = dt, alternative = "two.sided"))
```

And the result is:

```
##          Durbin-Watson test
##  data:  ols
##  DW = 0.77642, p-value < 2.2e-16
##  alternative hypothesis: true autocorrelation is not 0
```

Obviously, autocorrelation exists.

Let's learn how to read the statistic table in case you may only have the DW statistic sometime. When you are faced with a table of DW test statistics, please pay attention to whether there is an intercept and the number of covariates in the original linear model $y_i = \mathbf{X}_i + u_i$. For example, in our case, we should refer to the table which clearly tells that there are 3 covaritates and an intercept. However, a statistic table marked by 3 covariates but no intercept is totally different from the previous one.

Table 9: Durbin-Watson Statistic: 1 Percent Significance Points of dL and dU(with an intercept)

	k' = 3		k' = 5		k' = 10	
n	dL	dU	dL	dU	dL	dU
10	0.340	1.733	0.150	2.690	NA	NA
25	0.906	1.408	0.756	1.645	0.409	2.362
50	1.245	1.491	1.206	1.537	0.955	1.864
100	1.482	1.604	1.441	1.647	1.335	1.765
200	1.643	1.704	1.633	1.715	1.571	1.779

Table 10: Durbin-Watson Statistic: 1 Percent Significance Points of dL and dU(with no intercept)

	K = 3		K = 5		K = 10	
n	dL	dU	dL	dU	dL	dU
10	0.223	2.121	0.070	1.224	NA	NA
25	0.839	2.518	0.693	2.282	0.361	1.572
50	1.208	2.471	1.128	2.374	0.921	2.098
100	1.463	2.377	1.422	2.333	1.317	2.215
200	1.634	2.286	1.613	2.265	1.561	2.211

In the previous 2 tables, both k' and K represent the number of covariates. And in our case, we choose table 9. Because lower bound is monotonically increasing and DW = 0.77642 $<$ 1.643, it's clear that there exists positive autocorrelation at 1% significane level.

5.1.4 OLS method with HAC covariance matrix estirmator

When it comes to autocorrelation, we can't simply apply ols method with the White estimator. An estimator, overcoming heteroscedasticity and autocorrelation at the same time, is called as HAC(Heteroscedasticity and Autocorrelation Consistent) covariance matrix estirmator. Today, I show you the most famous one, Newey–West estimator. The Newey–West estimator was devised by Newey and West (1987), although there are a number of later variants.

Newey - West estimator

The estimator is used to try to overcome autocorrelation (also called serial correlation), and heteroskedasticity in the error terms in the models, often for regressions applied to time series data. The general approach is to use $\underline{\mathbf{X}}$ and $\mathbf{e}$ to devise an estimator of Q^*, a matrix of sums of squares and cross products that involves σ_{ij} and the rows of $\underline{\mathbf{X}}$. The least squares estimator $\mathbf{b}$ is a consistent estimator of , which implies that the least squares residuals $\mathbf{e_i}$ are "point-wise" consistent estimators of their population counterparts.

$$Q^* = \frac{1}{T}\sum_{t=1}^{T} e_t^2 x_t x_t' + \frac{1}{T}\sum_{\ell=1}^{L}\sum_{t=\ell+1}^{T} \omega_\ell e_t e_{t-\ell}(x_t x_{t-\ell}' + x_{t-\ell}' x_t)$$

$$\omega_\ell = 1 - \frac{\ell}{L+1}$$

ω_ℓ can be thought of as a "weight". Disturbances that are farther apart from each other are given lower weight, while those with equal subscripts are given a weight of 1. This ensures that second term converges (in some appropriate sense) to a finite matrix. This weighting scheme also ensures that the resulting covariance matrix is positive semi-definite.

Because numerical caculation is a little bit difficult, we directly use `sandwich::NeweyWest}()` function to obtain Newey-West covariance estimate. Following are the commands and results.

```
# OLS method with HAC covariance matrix estirmator

cov_hac = NeweyWest(ols)
se_hac = sqrt(diag(cov_hac))

t_hac = coef(ols)/se_hac
p_hac = pt(abs(t_hac), df = nrow(dt) - 4, lower.tail = FALSE)*2
```

```
print("NeweyWest covariance matrix estimate:"); cov_hac
```

```
##      [1] "NeweyWest covariance matrix estimate:"
##               (Intercept)   crime       industrial   distance
## (Intercept)  8.90329265 -0.025862361 -0.327823433 -1.012079887
## crime       -0.02586236  0.003067065 -0.002275065  0.005972089
## industrial  -0.32782343 -0.002275065  0.020075383  0.027409192
## distance    -1.01207989  0.005972089  0.027409192  0.140507641
```

```
print("NeweyWest se estimates:"); se_hac
```

```
## [1] "NeweyWest se estimates:"
## (Intercept)   crime    industrial    distance
```

```
## 2.98383858  0.05538109  0.14168763  0.37484349
```

```
print("T statistics by NeweyWest covariance:"); t_hac
```

```
## [1] "T statistics by NeweyWest covariance:"
## (Intercept)   crime    industrial    distance
##  11.899262   -4.926366   -5.153366   -2.709985
```

```
print("P values:"); p_hac
```

```
## [1] "P values:"
## (Intercept)      crime       industrial    distance
##  6.200717e-29 1.139249e-06 3.684529e-07 6.958822e-03
```

The previous steps can be easily realized by using `lmtest::coeftest}()` function. Here, I give the commands and display the results.

```
# Comparing with lmtest::coeftest
print("Test using NeweyWest estimator:"); coeftest(ols, vcov. = NeweyWest)
```

```
## [1] "Test using NeweyWest estimator:"

## t test of coefficients:
##
##      Estimate  Std. Error t value  Pr(>|t|)
## (Intercept) 35.505478   2.983839 11.8993 < 2.2e-16 ***
## crime       -0.272828   0.055381 -4.9264 1.139e-06 ***
## industrial  -0.730168   0.141688 -5.1534 3.685e-07 ***
## distance    -1.015820   0.374843 -2.7100  0.006959 **
## ---
## Signif. codes:  0  '***'  0.001  '**'  0.01  '*'  0.05  '.'  0.1  ' '  1
```

5.2 Numerical FGLS

In statistics, generalized least squares (GLS) is a technique for estimating the unknown parameters in a linear regression model when there are a certain degree of correlation between the residuals and(or) heteroscedasticity in a regression model. The estimators are summarized as follows.

Esitmators of GLS Method

With assumptions GH1-GH3, the GLS estimators for the parameters and covariance matrix are denoted like follows.

$$\hat{\beta}_{GLS} = \left(\underline{\mathbf{X}}^T \Omega^{-1} \underline{\mathbf{X}}\right)^{-1} \underline{\mathbf{X}}^T \Omega^{-1} \underline{Y}$$
$$Var[\hat{\beta}_{GLS}|\underline{\mathbf{X}}] = \left(\underline{\mathbf{X}}^T \Omega^{-1} \underline{\mathbf{X}}\right)^{-1}$$

5.2.1 Simple FGLS

If the covariance of the errors Ω is unknown, one can get a consistent estimate of Ω. say $\hat{\Omega}$, using an implementable version of GLS known as the feasible generalized least squares (FGLS) estimator. In FGLS, modeling proceeds in two stages:

Step 1 the model is estimated by OLS or another consistent (but inefficient) estimator, and the residuals are used to build a consistent estimator of the errors covariance matrix (to do so, one often needs to add additional constraints).

Step 2 using the consistent estimator of the covariance matrix of the errors, one can implement GLS ideas.

To perfrom the previous steps, the main problem is that how we construct a consisten estimator $\hat{\Omega}$. For simplicity, we assume that Ω is a diagonal matrix which is $\begin{bmatrix} \sigma_1^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n^2 \end{bmatrix}$, and replace each diagonal element by squared OLS residuals($\hat{u}_i^2$), which means that $\hat{\Omega} = \begin{bmatrix} \hat{u}_1^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \hat{u}_n^2 \end{bmatrix}$.

Here are commands.

```
# FGLS method by numerical calculation

X = as.matrix(cbind(rep(1, n), dt$crime, dt$industrial, dt$distance))
label = c("(Intercept)", "crime", "industrial", "distance")
colnames(X) = label

Y = as.vector(dt[, "value"])

# Using HCCME(HC_0)

cov_hat = diag(resid(ols)^2)


# Deriving estimates by closed-form estimators
b_fgls =  solve(t(X) %*% solve(cov_hat) %*% X) %*% (t(X) %*% solve(cov_hat) %*% Y)
cov_fgls = solve(t(X) %*% solve(cov_hat) %*% X)

se_fgls = sqrt(diag(cov_fgls))
t_fgls = b_fgls/se_fgls
p_fgls = pt(abs(t_fgls), df = n - ncol(X), lower.tail = FALSE)*2
```

And we can use the following commands to check the results.

```
print("FGLS covariance matrix estimate:"); cov_fgls
```

```
##      [1] "FGLS covariance matrix estimate:"
##       (Intercept)        crime     industrial     distance
## (Intercept)  4.211767e-03 -8.010524e-05 -1.199272e-04 -8.757138e-04
## crime       -8.010524e-05  5.611015e-05 -1.415095e-06  1.594712e-05
## industrial  -1.199272e-04 -1.415095e-06  6.642517e-06  1.865562e-05
## distance    -8.757138e-04  1.594712e-05  1.865562e-05  2.076158e-04
```

```
print("FGLS se estimates:"); se_fgls
```

```
##    [1] "FGLS se estimates:"
## (Intercept)       crime  industrial    distance
## 0.064898124 0.007490671 0.002577308 0.014408881
```

```
print("T statistics by FGLS covariance:"); t_fgls
```

```
## [1] "T statistics by FGLS covariance:"
##      [,1]
## (Intercept)  547.28832
## crime        -36.71753
## industrial  -283.84251
## distance     -70.21385
```

```
print("P values:"); p_fgls
```

```
## [1] "P values:"
##      [,1]
## (Intercept)  0.000000e+00
## crime       2.666154e-144
## industrial   0.000000e+00
## distance    9.426943e-262
```

5.2.2 Iterative FGLS Method

Sometimes, in order to improve the accuracy of the estimators in finite samples, we use iteration, i.e. taking the residuals from FGLS to update the errors covariance estimator, and then updating the FGLS estimation, applying the same idea iteratively until the estimators vary less than some tolerance. But this method does not necessarily improve the efficiency of the estimator very much if the original sample was small.

Now let me briefly introduce the key steps to you. The OLS estimator is calculated as usual by

$$\hat{\beta}_{OLS} = \left(\underline{\mathbf{X}}^T \underline{\mathbf{X}}\right)^{-1} \underline{\mathbf{X}}^T \underline{Y}$$

and estimates of the residuals $\hat{u}_j = (\underline{Y} - \underline{\mathbf{X}}\hat{}_{OLS})_j$ are constructed.

Assuming diagonal covariance matrix Ω, we build $\hat{\Omega}_{OLS} = diag(\hat{\sigma}_1^2, \hat{\sigma}_2^2, \dots, \hat{\sigma}_n^2)$.

Then we estimate $_{FGLS1}$ using $\hat{\Omega}_{OLS}$. With $_{FGLS1}$, performing the following steps.

1. $\hat{\beta}_{FGLS1} = \left(\underline{\mathbf{X}}^T \hat{\Omega}_{OLS}^{-1} \underline{\mathbf{X}}\right)^{-1} \underline{\mathbf{X}}^T \hat{\Omega}_{OLS}^{-1} \underline{Y}$
2. $\hat{\mathbf{u}}_{FGLS1} = \underline{Y} - \underline{\mathbf{X}} \hat{\beta}_{FGLS1}$
3. $\hat{\Omega}_{FGLS1} = diag(\hat{\sigma}^2_{FGLS1,1}, \hat{\sigma}^2_{FGLS1,2}, \dots, \hat{\sigma}^2_{FGLS1,n})$
4. $\hat{\beta}_{FGLS2} = \left(\underline{\mathbf{X}}^T \hat{\Omega}_{FGLS1}^{-1} \underline{\mathbf{X}}\right)^{-1} \underline{\mathbf{X}}^T \hat{\Omega}_{FGLS1}^{-1} \underline{Y}$

Repeating step2 to step4 until the convergence of $\hat{\Omega}$.

Let's check the commands.

```
# Iterative numerical method

termination = function(b, omega){
    grad = -2 * t(X) %*% solve(omega) %*% (Y - X %*% b)
    Inner = t(grad) %*% grad
    Inner_sqrt = sqrt(Inner)
    return(Inner_sqrt)
}

b_loop = b_fgls # step1
cov_u = cov_hat
n_loop = 0

while (termination(b_loop, cov_u) > 10^-12) { # judge
    u_hat = as.vector(Y - X %*% b_loop) # step2
    cov_u = diag(u_hat^2)   # step3
    cov_b_loop = solve(t(X) %*% solve(cov_u) %*% X)
    b_loop =  solve(t(X) %*% solve(cov_u) %*% X) %*%
        (t(X) %*% solve(cov_u) %*% Y) # step4
    n_loop = n_loop + 1
}


se_loop = sqrt(diag(cov_b_loop))
t_loop = b_loop/se_loop
p_loop = pt(abs(t_loop), df = n - ncol(X), lower.tail = FALSE)*2
```

And the results[6]

```
print("Number of iterations:"); n_loop
```

[6]Indeed, we didn't reach convergence because $\hat{\Omega}_{FGLS}$ turned out to be computationally sigular after 2 times of iterations.

```
## [1] "Number of iterations:"
## [1] 2
```

```
print("FGLS covariance matrix estimate:"); cov_b_loop
```

```
## [1] "FGLS covariance matrix estimate:"
##      (Intercept)         crime    industrial      distance
## (Intercept)  1.036975e-04 -6.721951e-05 -3.911968e-06 -1.146902e-05
## crime       -6.721951e-05  4.795889e-05  1.298768e-06  8.942851e-06
## industrial  -3.911968e-06  1.298768e-06  5.092477e-07 -1.687944e-08
## distance    -1.146902e-05  8.942851e-06 -1.687944e-08  1.832846e-06
```

```
print("FGLS se estimates:"); se_loop
```

```
## [1] "FGLS se estimates:"
## (Intercept)        crime   industrial     distance
## 0.0101831954 0.0069252360 0.0007136159 0.0013538264
```

```
print("T statistics by FGLS covariance:"); t_loop
```

```
## [1] "T statistics by FGLS covariance:"
##      [,1]
## (Intercept)  3488.42343
## crime         -41.94157
## industrial  -1027.18612
## distance     -744.37635
```

```
print("P values:"); p_loop
```

```
## [1] "P values:"
##      [,1]
## (Intercept)  0.000000e+00
## crime       3.533637e-166
## industrial   0.000000e+00
## distance     0.000000e+00
```

5.3 Built-in R Function

5.3.1 nlme::gls Function

In fact, we don't need to repeat the previous section when estimating because we have `nlme::gls()` function. And to use this function, please pay attention to the following 2 arguments.

- **correlation**: We should assign an argument describing the correlation structure of the error terms. In this case, we use `corAR1(rho)` which provides a AR(1) DGP and rho is derieved from dw test statistic.
- **weights**: We should assign an argument describing the heteroscedasticity structure of

the error terms. In this case, just use `varPower()`[7].

Commands and results are displayed as follows.

```
# Estimating by nlme::gls

# Caculating AR(1) estimate from dwtest
rho = 1 - 0.5 * as.numeric(dwtest$statistic)

# Estimating
gls_esti = gls(model, data = dt, correlation = corAR1(rho), weights = varPower())

(gls_summary = summary(gls_esti))
```

```
## Generalized least squares fit by REML
## Model: model
## Data: dt
## AIC      BIC      logLik
## 3259.125 3288.655 -1622.562
##
## Correlation Structure: AR(1)
## Formula: ~1
## Parameter estimate(s):
## Phi
## 0.6622399
## Variance function:
## Structure: Power of variance covariate
## Formula: ~fitted(.)
## Parameter estimates:
## power
## 0.4333236
##
## Coefficients:
##      Value Std.Error   t-value p-value
## (Intercept) 31.310565 2.5770092 12.149962  0.0000
## crime       -0.107077 0.0298874 -3.582697  0.0004
## industrial  -0.588257 0.1110842 -5.295590  0.0000
## distance    -0.514168 0.3808652 -1.349999  0.1776
##
## Correlation:
##     (Intr) crime  indstr
## crime      -0.105
## industrial -0.843 -0.053
## distance   -0.851  0.119  0.585
```

[7] an varFunc object in R and for more details, use help(varPower) and help(varFunc)

```
##
## Standardized residuals:
## Min         Q1        Med         Q3        Max
## -1.6892209 -0.6198956 -0.1628794  0.3536828  4.2750494
##
## Residual standard error: 2.078919
## Degrees of freedom: 506 total; 502 residual
```

5.3.2 Results of all methods

Use `stargazer` function to make a table displaying the results of each model.

```
# Making a table to show each model
library(stargazer)

R2_fgls = 1 - as.numeric(crossprod(Y - X %*% b_fgls)) / sum((Y - mean(Y))^2)
R2_loop = 1 - as.numeric(crossprod(Y - X %*% b_loop)) / sum((Y - mean(Y))^2)
R2_byR  = 1 - as.numeric(crossprod(Y - X %*% coef(gls_esti))) / sum((Y - mean(Y))^2)

se_byR  = coef(gls_summary)[, "Std.Error"]
t_byR   = coef(gls_summary)[, "t-value"]
p_byR   = coef(gls_summary)[, "p-value"]

stargazer(
ols, gls_esti, gls_esti, gls_esti,
coef = list(coef(ols), b_fgls, b_loop, coef(gls_esti)),
se = list(se_hac, se_fgls, se_loop, se_byR),
t = list(t_hac, t_fgls, t_loop, t_byR),
p = list(p_hac, p_fgls, p_loop, p_byR),
t.auto = FALSE, p.auto = FALSE,
report = "vcstp", keep.stat = c("n"),
add.lines = list(
    c("Type", "HA-Roubusted OLS", "fgls", "fgls_loop", "Built-in R gls"),
    c("R-Squared", round(ols_summary$r.squared, 4), round(R2_fgls, 4),
        round(R2_loop, 4), round(R2_byR, 4))
),
title = "Results of linear model estimations",
label = "LS",
type = "latex", header = FALSE, font.size = "small",
table.placement = "htb", omit.table.layout = "n"
)
```

For results, check table 11.

5.3.3 Summary and Which Method to Choose

Looking at table 11, we know that compared with OLS using HAC estimates, numerical gls (2) and (3) obviously return more efficient estimates. And by iteration, accuracy did increase to some extent. But with a more correctly specified structure of errors covariance matrix, `gls()` in R didn't give optimal estiamtes(slightly better than OLSe).

Whereas GLS is more efficient than OLS under heteroscedasticity or autocorrelation, this is not true for FGLS. The feasible estimator is, provided the errors covariance matrix is consistently estimated, asymptotically more efficient, but for a small or medium size sample, it can be actually less efficient than OLS. This is why, some authors prefer to use OLS, and reformulate their inferences by simply considering an alternative estimator for the variance of the estimator robust to heteroscedasticity or serial autocorrelation(HCCME or HACE). But for large samples FGLS is preferred over OLS under heteroskedasticity or serial correlation[8].

6 Panel Data Model

6.1 Backgruond and Data

A researcher wants to estimate the effect of full-time work experience on wages. He uses a *balanced* panel of 595 individuals from 1976 to 1982, taken from the Panel Study of Income Dynamics (PSID). The *balanced* panel data means that we can observe all individuals every year.

```
dt <- read.csv("./data/wages.csv")
head(dt, 14)
```

```
##    exp wks bluecol ind south smsa married  sex union ed black   lwage id time
## 1    3  32      no   0   yes   no     yes male    no  9    no 5.56068  1    1
## 2    4  43      no   0   yes   no     yes male    no  9    no 5.72031  1    2
## 3    5  40      no   0   yes   no     yes male    no  9    no 5.99645  1    3
## 4    6  39      no   0   yes   no     yes male    no  9    no 5.99645  1    4
## 5    7  42      no   1   yes   no     yes male    no  9    no 6.06146  1    5
## 6    8  35      no   1   yes   no     yes male    no  9    no 6.17379  1    6
## 7    9  32      no   1   yes   no     yes male    no  9    no 6.24417  1    7
## 8   30  34     yes   0    no   no     yes male    no 11    no 6.16331  2    1
## 9   31  27     yes   0    no   no     yes male    no 11    no 6.21461  2    2
## 10  32  33     yes   1    no   no     yes male   yes 11    no 6.26340  2    3
## 11  33  30     yes   1    no   no     yes male    no 11    no 6.54391  2    4
## 12  34  30     yes   1    no   no     yes male    no 11    no 6.69703  2    5
## 13  35  37     yes   1    no   no     yes male    no 11    no 6.79122  2    6
## 14  36  30     yes   1    no   no     yes male    no 11    no 6.81564  2    7
```

The variable `id` and `time` indicate individual and time indexs. We use these two variables

[8] A cautionary note is that the FGLS estimator is not always consistent . One case in which FGLS might be inconsistent is if there are individual specific fixed effects (Hansen 2007).

Table 11: Results of linear model estimations

	Dependent variable:			
	value			
	OLS		*generalized least squares*	
	(1)	(2)	(3)	(4)
crime	−0.273	−0.275	−0.290	−0.107
	(0.055)	(0.007)	(0.007)	(0.030)
	t = −4.926	t = −36.718	t = −41.942	t = −3.583
	p = 0.00001	p = 0.000	p = 0.000	p = 0.0004
industrial	−0.730	−0.732	−0.733	−0.588
	(0.142)	(0.003)	(0.001)	(0.111)
	t = −5.153	t = −283.843	t = −1,027.186	t = −5.296
	p = 0.00000	p = 0.000	p = 0.000	p = 0.00000
distance	−1.016	−1.012	−1.008	−0.514
	(0.375)	(0.014)	(0.001)	(0.381)
	t = −2.710	t = −70.214	t = −744.376	t = −1.350
	p = 0.007	p = 0.000	p = 0.000	p = 0.178
Constant	35.505	35.518	35.523	31.311
	(2.984)	(0.065)	(0.010)	(2.577)
	t = 11.899	t = 547.288	t = 3,488.423	t = 12.150
	p = 0.000	p = 0.000	p = 0.000	p = 0.000
Type	HA-Roubusted OLS	fgls	fgls_loop	Built-in R gls
R-Squared	0.3044	0.3044	0.3041	0.2732
Observations	506	506	506	506

to apply panel data models. Additionally, we use the following variables:

- `exp`: years of full-time work experience
- `sqexp`: squared value of `exp`
- `lwage`: logarithm of wage

```
dt <- dt[,c("id", "time", "exp", "lwage")]
dt$sqexp <- dt$exp^2
summary(dt)
```

```
##        id             time         exp              lwage           sqexp
##  Min.   :  1   Min.   :1   Min.   : 1.00   Min.   :4.605   Min.   :   1.0
##  1st Qu.:149   1st Qu.:2   1st Qu.:11.00   1st Qu.:6.395   1st Qu.: 121.0
##  Median :298   Median :4   Median :18.00   Median :6.685   Median : 324.0
```

```
##  Mean   :298   Mean   :4   Mean   :19.85   Mean   :6.676   Mean   : 514.4
##  3rd Qu.:447   3rd Qu.:6   3rd Qu.:29.00   3rd Qu.:6.953   3rd Qu.: 841.0
##  Max.   :595   Max.   :7   Max.   :51.00   Max.   :8.537   Max.   :2601.0
```

To examine the effect of labor experience on wages, we want to estimate the following linear panel data model:

$$\text{lwage}_{it} = \beta_1 \cdot \text{exp}_{it} + \beta_2 \cdot \text{sqexp}_{it} + u_{it}.$$

We can define the regression equation as the `formula` object in `R`. To exclude the intercept, we must specify `-1` in the rhs of regression equation. Thus, in `R`, we define the linear panel data model as follows:

```
model <- lwage ~ -1 + exp + sqexp
```

6.2 Pooled OLSE

We want to estimate the above regression equation by the OLS method. We will discuss assumptions for implementation. Let $\mathbf{X}_{it}$ be a $1 \times K$ (stochastic) explanatory vector. This vector contains `exp` and `sqexp`. Let Y_{it} be a random variable of outcome, that is, `lwage`. Then, the linear panel data model can be rewritten as follows:

$$Y_{it} = \mathbf{X}_{it}\beta + u_{it}, \quad t = 1, \dots, T, \quad i = 1, \dots, n.$$

Using notations $\underline{\mathbf{X}}_i = (\mathbf{X}'_{i1}, \dots, \mathbf{X}'_{iT})'$ and $\underline{Y}_i = (Y_{i1}, \dots, Y_{iT})'$, and $\underline{u}_i = (u_{i1}, \dots, u_{iT})'$, we can reformulate this model as follows:

$$\underline{Y}_i = \underline{\mathbf{X}}_i\beta + \underline{u}_i, \quad \forall i.$$

Now, we assume

1. (contempraneous) exogneity assumption: $E[\mathbf{X}'_{it}u_{it}] = 0$, $\forall i, t$.
   - This assumption,implies that u_{it} and $\mathbf{X}_{it}$ are orthogonal in the conditional mean sence, $E[u_{it}|\mathbf{X}_{it}] = 0$. However, this assumption does not imply u_{it} is uncorrelated with the explanatory variables in all time periods (strictly exogeneity), that is, $E[u_{it}|\mathbf{X}_{i1}, \dots, \mathbf{X}_{iT}] = 0$. This assumption palces no restriction on the relationship between $\mathbf{X}_{is}$ and u_{it} for $s \neq t$.
2. $E[\underline{\mathbf{X}}'_i\underline{\mathbf{X}}_i] \succ 0$.

Under these two assumptions, the true parameter is given by

$$\beta = E[\underline{\mathbf{X}}'_i\underline{\mathbf{X}}_i]^{-1}E[\underline{\mathbf{X}}'_i\underline{Y}_i].$$

Hence, the OLSE (pooled OLSE) is given by

$$\hat{\beta} = \left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}'_i\underline{\mathbf{X}}_i\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}'_i\underline{Y}_i\right) = \left(\frac{1}{n}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbf{X}'_{it}\mathbf{X}_{it}\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\sum_{t=1}^{T}\mathbf{X}'_{it}Y_{it}\right).$$

Using the full matrix notation, the OLS estimator is

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

where $\mathbf{X} = (\underline{\mathbf{X}}_1, \dots, \underline{\mathbf{X}}_n)'$ and $Y = (\underline{Y}_1, \dots, \underline{Y}_n)'$.

In `R` programming, the `lm` function provides the pooled OLSE in the context of panel data model. Another way is the `plm` function in the package `plm`. When you want to estimate pooled OLS by the `plm` function, you need to specify `model = "pooling"`. Moreover, you should specify individual and time index using `index` augment. This augment passes `index = c("individual index", "time index")`.

```
bols1 <- lm(model, data = dt)

library(plm)
bols2 <- plm(model, data = dt, model = "pooling", index = c("id", "time"))
```

The pooled OLS estimator is consistent and asymptotically normally distributed.

$$\sqrt{n}(\hat{\beta} - \beta) \sim N_{\mathbb{R}^K}(0, A^{-1}BA^{-1}),$$

where $A = E[\underline{\mathbf{X}}_i'\underline{\mathbf{X}}_i]$ and $B = E[\underline{\mathbf{X}}_i'\underline{u}_i\underline{u}_i'\underline{\mathbf{X}}_i]$. The consistent estimator of A and B is given by

$$\hat{A} = \frac{1}{n}\sum_{i=1}^{n} \underline{\mathbf{X}}_i'\underline{\mathbf{X}}_i,$$

$$\hat{B} = \frac{1}{n}\sum_{i=1}^{n} \underline{\mathbf{X}}_i'\hat{\underline{u}}_i\hat{\underline{u}}_i'\underline{\mathbf{X}}_i.$$

Thus, estimator of asymptotic variance of the pooled OLSE is

$$\widehat{Asyvar}(\hat{\beta}) = \left(\sum_{i=1}^{n} \underline{\mathbf{X}}_i'\underline{\mathbf{X}}_i\right)^{-1} \left(\sum_{i=1}^{n} \underline{\mathbf{X}}_i'\underline{u}_i\underline{u}_i'\underline{\mathbf{X}}_i\right) \left(\sum_{i=1}^{n} \underline{\mathbf{X}}_i'\underline{\mathbf{X}}_i\right)^{-1}.$$

Using the full matrix notations, we can reformulate

$$\widehat{Asyvar}(\hat{\beta}) = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'U\mathbf{X})(\mathbf{X}'\mathbf{X})^{-1},$$

where

$$U = \begin{pmatrix} \hat{\underline{u}}_1\hat{\underline{u}}_1' & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \hat{\underline{u}}_2\hat{\underline{u}}_2' & \cdots & \mathbf{0} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \hat{\underline{u}}_n\hat{\underline{u}}_n' \end{pmatrix}.$$

The standard errors calculated by this matrix is called *robust standard errors clustered by individuals*.

In `R`, the `lm` and `plm` function provide the standard errors based on $\widehat{Asyvar}(\hat{\beta}) = \hat{\sigma}^2(X'X)^{-1}$, where $\hat{\sigma}^2 = \hat{u}\hat{u}'/(nT - K)$ and $\hat{u} = Y - X\hat{\beta}$. There are two ways to obtain

cluster robust standard errors. The first way is to caclulate by yourself. The second way is to use the `coeftest` function in the package `lmtest`. When you use this function, we should use the `plm` function to estimate the pooled OLSE, and the `vcovHC` function (the package `sandwich`) in the `vcov` augment of `coeftest` fucntion.

```
# Setup
N <- length(unique(dt$id)); T <- length(unique(dt$time))
X <- model.matrix(bols1); k <- ncol(X)

# Inference
uhat <- bols1$residuals
uhatset <- matrix(0, nrow = nrow(X), ncol = nrow(X))

i_from <- 1; j_from <- 1
for (i in 1:max(dt$id)) {
  x <- as.numeric(rownames(dt))[dt$id == i]
  usq <- uhat[x] %*% t(uhat[x])
  i_to <- i_from + nrow(usq) - 1
  j_to <- j_from + ncol(usq) - 1
  uhatset[i_from:i_to, j_from:j_to] <- usq
  i_from <- i_to + 1; j_from <- j_to + 1
}

Ahat <- t(X) %*% X
Bhat <- t(X) %*% uhatset %*% X
vcovols <- solve(Ahat) %*% Bhat %*% solve(Ahat)
seols <- sqrt(diag(vcovols))

# Easy way
library(lmtest)
library(sandwich)
easy_cluster <- coeftest(
  bols2, vcov = vcovHC(bols2, type = "HC0", cluster = "group"))
```

The result is shown in the first column of Table 12. The partial effect of experience represents the percent change of wages. Thus,

$$(\% \text{ Change of Wage}) = 64.6 - 2 \cdot 1.3 \cdot \exp.$$

For example, wages increase by 12.99% at a mathematical mean of labor experience (`exp`). Moreover, this result implies diminishing marginal returns of labor experience.

6.3 Feasible GLSE

Adding and assumption of the conditional variance of $\underline{u}_i$ allows for using the Generalized Ordinary Squares method. To implement, we assume

1. $E[\underline{X}_i \otimes \underline{u}_i] = 0$. A sufficient condition to satisfy this relationship is $E[\underline{u}_i|\underline{X}_i] = 0$. This assumption implies $E[\underline{X}_i'\Omega^{-1}\underline{u}_i] = 0$ where $\Omega = E[\underline{u}_i\underline{u}_i']$ is $T \times T$ matrix.
2. $\Omega \succ 0$ and $E[\underline{X}_i'\Omega^{-1}\underline{X}_i'] \succ 0$.

The GLS estimator is given by

$$\hat{\beta}_{GLS} = \left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\Omega^{-1}\underline{\mathbf{X}}_i\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\Omega^{-1}\underline{Y}_i\right).$$

Under two assumptions, this estimator is weakly consistent.

In the feasible GLS method, we replace the unknown Ω with a consistent estimator. Here, we consider the two-step FGLS: obtain the OLS estimator and residuals; replace Ω by it. Then, the unknown Ω is replaced by

$$\hat{\Omega} = \frac{1}{n}\sum_{i=1}^{n}\underline{\hat{u}}_i\underline{\hat{u}}_i',$$

where $\underline{\hat{u}}_i = \underline{Y}_i - \underline{\mathbf{X}}_i\hat{\beta}_{OLS}$.

Thus, the FGLS estimator is

$$\hat{\beta}_{FGLS} = \left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\hat{\Omega}^{-1}\underline{\mathbf{X}}_i\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\hat{\Omega}^{-1}\underline{Y}_i\right).$$

Using the full matrix notations,

$$\hat{\beta}_{FGLS} = \{\mathbf{X}'(I_n \otimes \hat{\Omega}^{-1})\mathbf{X}\}^{-1}\{\mathbf{X}'(I_n \otimes \hat{\Omega}^{-1})Y\}.$$

In the `R` programming, there are two ways to obtain the FGLS estimator. The first way is to caclulate by yourself. The second way is to use the `pggls` function in the package `plm`. When you use the `pggls` function, you should specify individual and time indexs using `index` augment, and type in `model = "pooling"`.

```
# Setup
X <- model.matrix(model, dt); k <- ncol(X)
y <- dt$lwage
N <- length(unique(dt$id)); T <- length(unique(dt$time))

# Estimator of Omega
uhat <- bols1$residuals

Omega_sum <- matrix(0, ncol = T, nrow = T)
for (i in 1:N) {
  x <- as.numeric(rownames(dt))[dt$id == i]
  Omega_sum <- uhat[x] %*% t(uhat[x]) + Omega_sum
```

```
}
Omega <- Omega_sum/N

# FGLS estimator
kroOmega <- diag(N) %x% solve(Omega)
bfgls <- solve(t(X) %*% kroOmega %*% X) %*% (t(X) %*% kroOmega %*% y)

# Easy way!!!
easy_fgls <- pggls(
  model, data = dt, index = c("id", "time"), model = "pooling")
```

The asymptotic distribution of the FGLS estimator is given by

$$\sqrt{n}(\hat{\beta}_{FGLS} - \beta) \sim N_{\mathbb{R}^K}(0, A^{-1}BA^{-1}),$$

where $A = E[\underline{\mathbf{X}}_i'\Omega^{-1}\underline{\mathbf{X}}_i]$ and $B = E[\underline{\mathbf{X}}_i'\Omega^{-1}\underline{u}_i\underline{u}_i'\Omega^{-1}\underline{\mathbf{X}}_i]$. The consistent estimator of A and B is

$$\hat{A} = \frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\hat{\Omega}^{-1}\underline{\mathbf{X}}_i,$$

$$\hat{B} = \frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\hat{\Omega}^{-1}\hat{\underline{u}}_i^{FLGS}\hat{\underline{u}}_i^{FGLS'}\hat{\Omega}^{-1}\underline{\mathbf{X}}_i,$$

where $\underline{u}_i^{FLGS} = \underline{Y}_i - \underline{\mathbf{X}}_i\hat{\beta}_{FGLS}$. Thus, estimator of asymptotic variance of the FGLS estimator is

$$\widehat{Asyvar}(\hat{\beta}_{FGLS}) = \left(\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\hat{\Omega}^{-1}\underline{\mathbf{X}}_i\right)^{-1}\left(\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\hat{\Omega}^{-1}\underline{u}_i^{FLGS}\underline{u}_i^{FGLS'}\hat{\Omega}^{-1}\underline{\mathbf{X}}_i\right)\left(\sum_{i=1}^{n}\underline{\mathbf{X}}_i'\hat{\Omega}^{-1}\underline{\mathbf{X}}_i\right)^{-1}.$$

Using the full matrix notations,

$$\widehat{Asyvar}(\hat{\beta}_{FGLS}) = \{\mathbf{X}'(I_n\otimes\hat{\Omega}^{-1})\mathbf{X}\}^{-1}\{\mathbf{X}'(I_n\otimes\hat{\Omega}^{-1})U(I_n\otimes\hat{\Omega}^{-1})\mathbf{X}\}^{-1}\{\mathbf{X}'(I_n\otimes\hat{\Omega}^{-1})\mathbf{X}\}^{-1},$$

where

$$U = \begin{pmatrix} \hat{\underline{u}}_1^{FLGS}\hat{\underline{u}}_1^{FGLS'} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \hat{\underline{u}}_2^{FLGS}\hat{\underline{u}}_2^{FGLS'} & \cdots & \mathbf{0} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \hat{\underline{u}}_n^{FLGS}\hat{\underline{u}}_n^{FGLS'} \end{pmatrix}.$$

In the `R` programming, you need to caclculate by yourself. The `pggls` function provides the FGLS estimator. However, this function calculates standard errors, assuming *system homoskedasticity*, that is, $E[\underline{\mathbf{X}}_i'\Omega^{-1}\underline{u}_i\underline{u}_i'\Omega^{-1}\underline{\mathbf{X}}_i] = E[\underline{\mathbf{X}}_i'\Omega^{-1}\underline{\mathbf{X}}_i]$. If you can rationale this assumption, the `bggls` fucntion is the easiest way to carry out statistical inference.

```
ufgls <- y - X %*% bfgls
uhatset <- matrix(0, nrow = nrow(X), ncol = nrow(X))
i_from <- 1; j_from <- 1
for (i in 1:max(dt$id)) {
  x <- as.numeric(rownames(dt))[dt$id == i]
  usq <- uhat[x] %*% t(uhat[x])
  i_to <- i_from + nrow(usq) - 1
  j_to <- j_from + ncol(usq) - 1
  uhatset[i_from:i_to, j_from:j_to] <- usq
  i_from <- i_to + 1; j_from <- j_to + 1
}

Ahat <- t(X) %*% kroOmega %*% X
Bhat <- t(X) %*% kroOmega %*% uhatset %*% kroOmega %*% X
vcovfgls <- solve(Ahat) %*% Bhat %*% solve(Ahat)
sefgls <- sqrt(diag(vcovfgls))
```

The result is shown in the second column of Table 12. The partial effect of experience represents the percent change of wages. Thus,

$$(\% \text{ Change of Wage}) = 52.9 - 2 \cdot 0.9 \cdot \exp.$$

For example, wages increase by 17.17% at a mathematical mean of labor experience (`exp`).

6.4 Fixed Effect Model

To examine the effect of labor experience on wages, we introduce unobserved heterogeneity such as ability. The unobserved effects model is given by

$$\text{lwage}_{it} = \beta_1 \cdot \exp_{it} + \beta_2 \cdot \text{sqexp}_{it} + c_i + u_{it},$$

where c_i is unobserved component which is constant over time, u_{it} is the idiosyncratic error term. The fixed effect model treats c_i as a parameter to be estimated for each cross section unit i.

We generalize the unobserved effects model as follows:

$$Y_{it} = \mathbf{X}_{it}\beta + c_i + u_{it}, \quad t = 1, \dots, T, \quad i = 1, \dots, n.$$

Using notations $\underline{\mathbf{X}}_i = (\mathbf{X}'_{i1}, \dots, \mathbf{X}'_{iT})'$ and $\underline{Y}_i = (Y_{i1}, \dots, Y_{iT})'$, and $\underline{u}_i = (u_{i1}, \dots, u_{iT})'$, we can reformulate this model as follows:

$$\underline{Y}_i = \underline{\mathbf{X}}_i\beta + \iota c_i + \underline{u}_i, \quad \forall i,$$

where $\iota = (1, \dots, 1)'$ is $T \times 1$ vector.

To implement the fixed effect model, we assume the following three assumptions:

1. Strict exogeneity: $E[u_{it}|\mathbf{X}_{i1}, \dots, \mathbf{X}_{iT}, c_i] = 0$.
2. Full rank: $rank(\sum_t E[\ddot{\mathbf{X}}_{it}'\ddot{\mathbf{X}}_{it}]) = rank(E[\underline{\ddot{\mathbf{X}}}_i'\underline{\ddot{\mathbf{X}}}_i]) = K$ where $\ddot{\mathbf{X}}_{it} = \mathbf{X}_{it} - T^{-1}\sum_t \mathbf{X}_{it}$.
3. homoskedasticity: $E[\underline{u}_i\underline{u}_i'|\mathbf{X}_{i1}, \dots, \mathbf{X}_{iT}, c_i] = \sigma_u^2 I_T$.

To obtain the FE estimator, we consider the within transformation first. Averaging the unobserved effects model for individual i and time t over time yields

$$\bar{Y}_i = \bar{\mathbf{X}}_i\beta + c_i + \bar{u}_i,$$

where $\bar{Y}_i = T^{-1}\sum_t Y_{it}$, $\bar{\mathbf{X}}_i = T^{-1}\sum_t \mathbf{X}_{it}$, and $\bar{u}_i = T^{-1}\sum_t u_{it}$. Subtracting this equation from the original one for each t yields

$$Y_{it} - \bar{Y}_i = (\mathbf{X}_{it} - \bar{\mathbf{X}}_i)\beta + (u_{it} - \bar{u}_i) \Leftrightarrow \ddot{Y}_{it} = \ddot{\mathbf{X}}_{it}\beta + \ddot{u}_{it}.$$

Note that $E[\ddot{u}_{it}|\ddot{\mathbf{X}}_{i1}, \dots, \ddot{\mathbf{X}}_{iT}] = 0$ under the first assumption. Using the T system of equation, the within transformation is

$$Q_T\underline{Y}_i = Q_T\underline{\mathbf{X}}_i\beta + Q_T\underline{u}_i \Leftrightarrow \underline{\ddot{Y}}_i = \underline{\ddot{\mathbf{X}}}_i\beta + \underline{\ddot{u}}_i.$$

where $Q_T = I_T - \iota(\iota'\iota)^{-1}\iota$ is *time-demeaning matrix*, and $Q_T\iota = 0$. Using the matrix notations, the within transformation is

$$(I_n \otimes Q_t)Y = (I_n \otimes Q_t)X\beta + (I_n \otimes Q_t)u \Leftrightarrow \ddot{Y} = \ddot{X}\beta + \ddot{u}.$$

Before showing the FE estimator, I will show $Q_T\underline{Y}_i = Y_{it} - T^{-1}\sum_t Y_{it}$, using `R`. As an illustration, we caclulate time-demeaned outcome variable for $i = 1$, $\ddot{Y}_{1t}$. R snippet is as follows:

```
# extract outcome variables for i = 1
i <- as.numeric(rownames(dt))[dt$id == 1]
y1 <- dt$lwage[i]

# deviation from mean
Ydev1 <- y1 - mean(y1)
print("Deviation from mean across time"); Ydev1
```

```
## [1] "Deviation from mean across time"
```

```
## [1] -0.40407857 -0.24444857  0.03169143  0.03169143  0.09670143  0.20903143
## [7]  0.27941143
```

```
# time demean-matrix
T <- length(y1)
vec1 <- rep(1, T)
Qt <- diag(T) - vec1 %*% solve(t(vec1) %*% vec1) %*% t(vec1)
Ydev2 <- Qt %*% y1
print("Time-demeaning matrix"); Ydev2
```

```
## [1] "Time-demeaning matrix"
```

```
##              [,1]
## [1,] -0.40407857
## [2,] -0.24444857
## [3,]  0.03169143
## [4,]  0.03169143
## [5,]  0.09670143
## [6,]  0.20903143
## [7,]  0.27941143
```

The FE estimator is given by

$$\hat{\beta}_{FE} = \left(\frac{1}{n} \sum_{i=1}^{n} \ddot{\underline{\mathbf{X}}}_i' \ddot{\underline{\mathbf{X}}}_i \right)^{-1} \left(\frac{1}{n} \sum_{i=1}^{n} \ddot{\underline{\mathbf{X}}}_i' \ddot{\underline{Y}}_i \right) = (\ddot{X}'\ddot{X})^{-1}(\ddot{X}'\ddot{Y}).$$

In the `R` programming, there are two ways to obtain the FE estimator. The first way is to caclulate by yourself. The second way is to use the `plm` function. When you use the `plm` function, you need to specify `model = "within"` to implement the FE model.

```
# Setup
X <- model.matrix(model, dt); k <- ncol(X)
y <- dt$lwage
N <- length(unique(dt$id)); T <- length(unique(dt$time))

# FE estimator
i <- rep(1, T)
Qt <- diag(T) - i %*% solve(t(i) %*% i) %*% t(i)
Ydev <- diag(N) %x% Qt %*% y
Xdev <- diag(N) %x% Qt %*% X
bfe <- solve(t(Xdev) %*% Xdev) %*% t(Xdev) %*% Ydev

# Awesome way !!!
plmfe <- plm(model, data = dt, index = c("id", "time"), model = "within")
```

Under the third assumption, asymptotic distribution of the FE estimator is given by

$$\sqrt{n}(\hat{\beta}_{FE} - \beta) \sim N_{\mathbb{R}^K}(0, \sigma_u^2 E[\ddot{\underline{\mathbf{X}}}_i' \ddot{\underline{\mathbf{X}}}_i]).$$

The consistent estimator of the asymptotic variance of the FE estimator is

$$\widehat{Asyvar}(\hat{\beta}_{FE}) = \hat{\sigma}_u^2 \left(\sum_{i=1}^{n} \ddot{\underline{\mathbf{X}}}_i' \ddot{\underline{\mathbf{X}}}_i \right)^{-1} = \hat{\sigma}_u^2 (\ddot{X}'\ddot{X})^{-1},$$

where $\hat{\sigma}_u^2 = \frac{1}{n(T-1)-K} \sum_i \sum_t \hat{\ddot{u}}_{it}$, and $\hat{\ddot{u}}_{it} = \ddot{Y}_{it} - \ddot{\mathbf{X}}_{it}\hat{\beta}_{FE}$.

In the `R` programming, the `plm` function also returns standard errors, $\hat{\sigma}_u^2(\ddot{X}'\ddot{X})^{-1}$. Of course, you can compute the standard errors manually. The sample code is as follows:

```
uhat <- Ydev - Xdev %*% bfe
sigmahat <- sum(uhat^2)/(N*(T-1)-k)
vcovfe <- sigmahat * solve(t(Xdev) %*% Xdev)
sefe <- sqrt(diag(vcovfe))
```

The result is shown in the third column in Table 12. The partial effect of experience represents the percent change of wages. Thus,

$$(\% \text{ Change of Wage}) = 11.4 - 2 \cdot 0.04 \cdot \text{exp}.$$

For example, wages increase by 9.812% at a mathematical mean of labor experience (`exp`).

6.5 Random Effect Model

Again, consider the unobserved effects model:

$$Y_{it} = \mathbf{X}_{it}\beta + c_i + u_{it}, \quad t = 1, \dots, T, \quad i = 1, \dots, n.$$

The random effect model treats c_i as a random variable. Thus, the variable c_i is put into the error term. We reformulate the model as follows:

$$Y_{it} = \mathbf{X}_{it}\beta + v_{it},$$

where $v_{it} = c_i + u_{it}$. Using notations $\underline{\mathbf{X}}_i = (\mathbf{X}'_{i1}, \dots, \mathbf{X}'_{iT})'$ and $\underline{Y}_i = (Y_{i1}, \dots, Y_{iT})'$, and $\underline{u}_i = (u_{i1}, \dots, u_{iT})'$, we can reformulate this model as follows:

$$\underline{Y}_i = \underline{\mathbf{X}}_i\beta + \underline{v}_i,$$

where $\underline{v}_i = \iota c_i + \underline{u}_i$, and $\iota = (1, \dots, 1)'$ is $T \times 1$ vector.

To implement the RE model, we assume

1. Strict exogeneity: $E[u_{it}|\mathbf{X}_{i1}, \dots, \mathbf{X}_{iT}, c_i] = 0$.
2. Orthogonality between c_i and $\mathbf{X}_{it}$: $E[c_i|\mathbf{X}_{i1}, \dots, \mathbf{X}_{iT}] = 0$.
3. Full rank: $rank(E[\underline{\mathbf{X}}'_i\Omega^{-1}\underline{\mathbf{X}}_i]) = K$.
4. $E[\underline{u}_i\underline{u}'_i|\mathbf{X}_{i1}, \dots, \mathbf{X}_{iT}, c_i] = \sigma_u^2 I_T$, and $E[c_i^2|\mathbf{X}_{i1}, \dots, \mathbf{X}_{iT}] = \sigma_c^2$.

Using the FGLS method through the introduction of Σ, we can obtain the FGLS-type RE estimator as follows:

$$\hat{\beta}_{RE} = \left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}'_i\hat{\Omega}^{-1}\underline{\mathbf{X}}_i\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\underline{\mathbf{X}}'_i\hat{\Omega}^{-1}\underline{Y}_i\right),$$

where

$$\hat{\Omega} = \hat{\sigma}_u^2 I_T + \hat{\sigma}_c^2 \iota\iota' = \begin{pmatrix} \hat{\sigma}_c^2 + \hat{\sigma}_u^2 & \hat{\sigma}_c^2 & \cdots & \hat{\sigma}_c^2 \\ \hat{\sigma}_c^2 & \hat{\sigma}_c^2 + \hat{\sigma}_u^2 & \cdots & \hat{\sigma}_c^2 \\ \vdots & \vdots & \cdots & \vdots \\ \hat{\sigma}_c^2 & \hat{\sigma}_c^2 & \cdots & \hat{\sigma}_c^2 + \hat{\sigma}_u^2 \end{pmatrix}.$$

The estimator $\hat{\sigma}_u^2$ and $\hat{\sigma}_c^2$ can be obtained by

$$\hat{\sigma}_u^2 = \hat{\sigma}_v^2 - \hat{\sigma}_c^2,$$

$$\hat{\sigma}_v^2 = \frac{1}{nT-K}\sum_{i=1}^{n}\sum_{t=1}^{T}\hat{v}_{it}^2,$$

$$\hat{\sigma}_c^2 = \frac{1}{nT(T-1)/2-K}\sum_{i=1}^{n}\sum_{t=1}^{T-1}\sum_{s=t+1}^{T}\hat{v}_{it}\hat{v}_{is},$$

$$\hat{v}_{it} = Y_{it} - X_{it}\hat{\beta}_{OLS}.$$

In the `R` programming, the `plm` function provides the random effect model. However, the procedure is not the feasible GLS method, but the OLS method on a dataset in which all variables are subject to quasi-demeaning.[9] The two procedures generate the same RE estimator. Moreover, the idiosyncratic error and the unobserved component are obtained by differenct approach. To implement the RE model described above, we compute manually.

```
# Setup
X <- model.matrix(model, dt)
y <- dt$lwage
k <- ncol(X)
N <- length(unique(dt$id))
T <- length(unique(dt$time))

# estimator of Omega
pols <- lm(model, dt)
vhat <- pols$residuals
sigmav <- sum(vhat^2)/(N*T - k)

sumuc <- matrix(0, nrow = N, ncol = T-1)
for (i in 1:N) {
  for (t in 1:T-1) {
    it <- as.numeric(rownames(dt))[dt$id == i & dt$time == t]
    is <- as.numeric(rownames(dt))[dt$id == i & dt$time > t]
    sumuc[i,t] <- vhat[it] * sum(vhat[is])
  }
}
sigmac <- sum(colSums(sumuc))/((N*T*(T-1))/2-k)
sigmau <- sigmav - sigmac
```

[9]The RE estimator by the quasi-demeaning method is simple. First, we calculate quasi-demeaned variables as in $\tilde{Y}_{it} = Y_{it} - \theta\bar{Y}_i$ where $\theta = 1-(\sigma_u^2/(\sigma_u^2+T\sigma_c^2))^{1/2}$. Using the matrix notations, $\underline{\tilde{Y}}_i = \tilde{Q}_T\underline{Y}_i$ where $\tilde{Q}_T$ is the quasi-demeaning matrix, which is given by $\tilde{Q}_T = I_T - \theta\iota(\iota'\iota)^{-1}\iota$. After transforming all variables, we estimate $\underline{\tilde{Y}}_i = \underline{\tilde{X}}_i\beta + \underline{\tilde{u}}_i$ by OLS method. The variance-covariance matrix is $\hat{\sigma}(\sum_i \underline{\tilde{X}}_i\underline{\tilde{X}}_i')^{-1}$ where $\hat{\sigma} = \underline{\tilde{Y}}_i - \underline{\tilde{X}}_i\hat{\beta}$. See http://ricardo.ecn.wfu.edu/~cottrell/gretl/random-effects.pdf in detail.

Table 12: Effect of Experience on Wages (Standard errors are in parentheses)

	Dependent variable:			
	lwage			
	Pooled OLS	FGLS	Fixed Effect	Random Effect
	(1)	(2)	(3)	(4)
exp	0.646	0.529	0.114	0.395
	(0.011)	(0.010)	(0.002)	(0.006)
sqexp	−0.013	−0.009	−0.0004	−0.006
	(0.0004)	(0.0004)	(0.0001)	(0.0002)
Observations	4,165	4,165	4,165	4,165

```
i <- rep(1, T)
Omega <- sigmau * diag(T) + sigmac * i %*% t(i)
kroOmega <- diag(N) %x% solve(Omega)

# Random effect
bre <- solve(t(X) %*% kroOmega %*% X) %*% t(X) %*% kroOmega %*% y
```

A consistent estimator of asymptotic variance of the RE estimator is given by

$$\widehat{Asyvar}(\hat{\beta}_{RE}) = \left(\underline{\mathbf{X}}_i' \hat{\Omega}^{-1} \underline{\mathbf{X}}_i\right)^{-1}.$$

In the `R` programming, the `plm` function returns standard errors calculated by variance-covariance matrix of OLS on a quasi-demeaned data. To obtain the FGLS-type standard errors, we compuate manually. The sample code is as follows:

```
vcovre <- solve(t(X) %*% kroOmega %*% X)
sere <- sqrt(diag(vcovre))
```

The result is shown in the fourth column in Table 12. The partial effect of experience represents the percent change of wages. Thus,

$$(\%\text{ Change of Wage}) = 39.5 - 2 \cdot 0.6 \cdot \text{exp}.$$

For example, wages increase by 15.68% at a mathematical mean of labor experience (`exp`).

6.6 Hausman Test

The Hausman test provides empirical evidence on choosing between FE and RE model. The null hypothesis of this test is $\mathbf{X}_{it}$ and c_i are independent. If we can reject the null hypothesis,

then the FE model is preferred. If we cannot reject the null hypothesis, then the RE model should be used.

The test statistics is

$$\hat{H} = (\hat{\beta}_{RE} - \hat{\beta}_{FE})' \{\widehat{Var}(\hat{\beta}_{RE}) - \widehat{Var}(\hat{\beta}_{FE})\}^{-1} (\hat{\beta}_{RE} - \hat{\beta}_{FE}).$$

The limiting distribution of this test statistics is $\hat{H} \to \chi^2(K)$.

In the `R` programming, the manual computation is very easy. Alternatively, the `phtest` function in the package `plm` provides the Hausman test. To use the `phtest`, we need to estimate the FE and RE model by the `plm` function.

```
delta <- bre - bfe
diffv <- vcovre - vcovfe
H <- t(delta) %*% solve(diffv) %*% delta
qtchi <- qchisq(0.99, nrow(delta))
paste("The test statistics of Hausman test is ", round(H, 3))
```

```
## [1] "The test statistics of Hausman test is  3999.537"
```

```
paste("The 1% quantile value of chi-sq dist is", round(qtchi, 3))
```

```
## [1] "The 1% quantile value of chi-sq dist is 9.21"
```

In this empirical application, we can reject the null hypothesis at 1% significance level. This implies that we should use the FE model in this application beucase observed covariates and unobserved component are not independent.

7 Generalized Method of Moments

7.1 Before Estimating

7.1.1 Background and Data

The data we use today originates from Harrison and Rubinfeld (1978) which investigates the methodological problems associated with the use of housing market data to measure the willingness to pay for clean air. It drew a conclusion that marginal air pollution damages are found to increase with the level of air pollution and with household income. But here we just use some variables of the original data to construct a multiple linear regression model focusing on how the attributes of communities affect housing prices(value) of Boston city. And this time, we will concentrate on the endogenous problem.

We use an open data which is called as "Boston Neighboorhood Housing Prices Dataset[10]". Although there exist many variables, this time, we take 6 of them: `value`, `crime`, `industrial`, `distance`, `black` and `ptratio`. And here come the descriptions.

[10]data source: http://biostat.mc.vanderbilt.edu/DataSets. This linkage seems not to work very well so I have uploaded the csv file on CLE.

- `value`: a continuous variable meaning value of owner-occupied homes in $1000's.
- `crime`: a continuous variable representing per capita crime rate by town.
- `industrial`: a continuous variable showing proportion of non-retail business acres per town.
- `distance`: a continuous variable revealing weighted distances to five Boston employment centres.
- `black`: a continuous variable which is the black proportion of population, used as an instrument for `crime`.
- `ptratio`: a continuous variable defined as pupil-teacher ratio by town school district, measuring public sector benefits and used as an instrument for `crime` as well.

As same as the last time, `value` is the dependent variable. Now, let's quickly build the dataset to use today.

```
library(AER) # Including lmtest, sandwich and ivreg(2SLS)
library(gmm) # Including gmm function
library(MASS) # Using ginv: generalized inverse operation
library(stargazer)

# Preparations

dt = read.csv(
file = "~/boston.csv",
header = TRUE, sep = ",", row.names = NULL, stringsAsFactors = FALSE)
dt = dt[complete.cases(dt),]
# complete.cases returns TURE if one row doesn't contain NA

n = nrow(dt)
head(dt[, c("value", "crime", "industrial", "distance", "black",  "ptratio")])
```

And the first few rows are shown as below.

```
  value  crime    industrial distance black    ptratio
1  24.0 0.00632       2.31   4.0900 0.0000000    15.3
2  21.6 0.02731       7.07   4.9671 0.0000000    17.8
3  34.7 0.02729       7.07   4.9671 0.3238495    17.8
4  33.4 0.03237       2.18   6.0622 0.1804159    18.7
5  36.2 0.06905       2.18   6.0622 0.0000000    18.7
6  28.7 0.02985       2.18   6.0622 0.2210220    18.7
```

7.1.2 The idea of finding an instrumental variable

We say that when exogeneity condition is no longer satisfied(relaxization of Gauss Markov basic assumption), the ols estimator will lose consistency. To deal with such problem, IV(instrumental variable) method is a good solution. But how do we find an instrumental variable? Here are two main criteria for defining an IV:

1. The instrument must be correlated with the endogenous explanatory variables, conditionally on the other covariates[11].
2. The instrument cannot be correlated with the error term in the explanatory equation, conditionally on the other covariates. The explanatory equation is that, taking a linear example, an equation which is given by $\underline{\mathbf{Y}} = \underline{\mathbf{X}}\theta + \underline{\mathbf{U}}$[12].

To initialize today's empirical application, we think that `crime` may be an endogenous variable and want to deal with the problems resulting from endogeneity. Following the previous 2 criteria, `black` and `ptratio` are, in a way, correlated with crime rate(the original paper is published in 1978 and I mean no offense here). However, these two variables remain a probability of not functioning very well and that's called weak instruments problem which won't be talked today. If you are interested in such stuffs, please check Nichols (2006) on your own.

Then, let's prepare the explanatory variables and instrumental variables in a matrix form with R for further estimations. Here, we denote the dimensions that $dim(g) = L$ and $dim(\theta) = d$. In this case, L = d + 1 = 5(containing constant). Remember that we detected heteroskedasticity and autocorrelation in TA session GLS with this dataset. So it provides us a good example to learn about 2SGMM(2 step GMM) method.

```
# Extract varibales and instruments
name_x = c("constant", "crime", "industrial", "distance")
name_z = c(
    "constant", "black",  "ptratio", "industrial", "distance"
) # means L = d + 1 = 5
constant = rep(1, n)

X = cbind(constant, dt[, c("crime", "industrial", "distance")])
colnames(X) = name_x

Z = cbind(constant, dt[, c("black", "ptratio", "industrial", "distance")])
colnames(Z) = name_z

Y = dt[, "value"]
```

7.2 Numerical 2SGMM Method

7.2.1 Formulations and Recalling HA-Robusted Ols

Firstly, let's declare the linear model. The multiple regression model is easily written as below.

$$value_i = \beta_0 + \beta_1 crime_i + \beta_2 industrial_i + \beta_3 distance_i + u_i \quad \forall i = 1, \dots, n.$$

[11] But a weak correlation may provide misleading inferences about parameter estimates and standard errors (Nichols 2006) .

[12] "explanatory" is used because we want to know the causal effects between $\underline{\mathbf{Y}}$ and $\underline{\mathbf{X}}$

Remember that when there exist heteroskedasticity and autocorrelation in the error term, we need to apply a NeweyWest(HAC) variance covariance estimator. Here, I just show you with the commands and don't display the results. The results will be summarized in a table as well as the 2 step estimates for comparision.

```
# Numerical 2SGMM Method to A Simple Linear Over-identified case

# Since we tested out the heteroskedasticity,
# this dataset provides an appropriate example for studying 2SGMM.


# Firstly, recall HA-robust ols estimates for later comparison
model = value ~ crime + industrial + distance

ols = lm(model, data = dt)
#summary(ols)
coeftest(ols, vcov. = NeweyWest)

cov_hac = NeweyWest(ols)
se_hac = sqrt(diag(cov_hac))

t_hac = coef(ols)/se_hac
p_hac = pt(abs(t_hac), df = nrow(dt) - ncol(X), lower.tail = FALSE)*2
```

7.2.2 Numerical 2SGMM

7.2.2.1 Step 1 of the 2SGMM Method Firstly, let's review the procedures of the 2 step estimation[13].

Step 1 of the 2SGMM Method

Choose the metric in the first step. The optimal metric is the inverse of the variance of the orthogonality conditions which are $g(K_i, \theta) = (\mathbf{Z}_i^{IV})^T(Y_i - \mathbf{X}_i\theta) = (\mathbf{Z}_i^{IV})^T u_i$ for all i. And the population orthogonality condition(or so-called moment condition) is given by $\mathbb{E}\left(g(K_i, \theta)\right) = \mathbf{0}$.

Thus it is better to choose a metric in the first step, which is close to the optimal metric. To do so, we assume that homoskedasticity assumption is satisfied. For example,

$$\mathbb{E}[u_i^2|\mathbf{Z}_i^{IV}] = \mathbb{E}[u_i^2] = \sigma^2.$$

Then, the first step metric would be

[13]For a better understanding of the codes I provided, I use the notations from class and modified them a little bit. Of course, stacked form is used as well.

$$\begin{aligned}
\widehat{Var}(g(K_i,\hat{\theta}))_1 &= \frac{1}{n}\sum_{i=1}^{n}(\mathbf{Z}_i^{IV})^T(\mathbf{Z}_i^{IV}),\\
&= \frac{1}{n}(\underline{\mathbf{Z}}^{IV})^T(\underline{\mathbf{Z}}^{IV}),\\
\widehat{W}_1 &= \left(\widehat{Var}(g(K_i,\hat{\theta}))_1\right)^{-1}.
\end{aligned}$$

When this metric is applied, the first step estimator is then given by

$$\begin{aligned}
\hat{\theta}_1 =& \left\{\left(\frac{1}{n}\sum_{i=1}^{n}((\mathbf{Z}_i^{IV})^T\mathbf{X}_i)\right)^T\widehat{W}_1\left(\frac{1}{n}\sum_{i=1}^{n}((\mathbf{Z}_i^{IV})^T\mathbf{X}_i)\right)\right\}^{-1}\\
&\times\left(\frac{1}{n}\sum_{i=1}^{n}((\mathbf{Z}_i^{IV})^T\mathbf{X}_i)\right)^T\widehat{W}_1\left(\frac{1}{n}\sum_{i=1}^{n}((\mathbf{Z}_i^{IV})^T\mathbf{Y}_i)\right),\\
=&\left\{\left((\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right)^T\widehat{W}_1(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right\}^{-1}\left((\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right)^T\widehat{W}_1(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{Y}},
\end{aligned}$$

where $\underline{\mathbf{Y}} = (Y_1,\dots,Y_n)^T$, $\underline{\mathbf{X}} = (\mathbf{X}_1^T,\dots,\mathbf{X}_n^T)^T$ and $\underline{\mathbf{Z}}^{IV} = ((\mathbf{Z}_1^{IV})^T,\dots,(\mathbf{Z}_n^{IV})^T)^T$. Such an estimator with the first step metric $\widehat{W}_1$ is called as the 2 step instrumental variable (2S-IV) estimator or the 2 step least square (2SLS) estimator. After being calculated, this 2S-IV is then plugged in the second step of GMM.

And the asmptotic variance covariance matrix can, according to the properties of GMM estimator (section 7.4), be written as

$$\begin{aligned}
\mathbb{V}(\hat{\theta}_1) =& \left\{\mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right]^T W_0\mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right]\right\}^{-1}\mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right]^T W_0 Var\left(g(K_i,\theta_0)\right)\\
&\times W_0\mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right]\left\{\mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right]^T W_0\mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T\underline{\mathbf{X}}\right]\right\}^{-1},
\end{aligned}$$

where θ_0 is the true parameter with $Var\left(g(K_i,\theta_0)\right) = \mathbb{E}\left[g(K_i,\theta_0)g(K_i,\theta_0)^T\right]$ and $\widehat{W}_1 \xrightarrow{p} W_0$.

And if we just take $\widehat{W}_1$ and $\widehat{Var}(g(K_i,\theta))_1$ as the empirical counterparts of W_0 and $Var\left(g(K_i,\theta_0)\right)$, respectively. We will have the covariance estimator as

$$
\begin{aligned}
\hat{\mathbb{V}}(\hat{\theta}_1) = & \left\{ \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right)^T \widehat{W}_1 (\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right\}^{-1} \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right)^T \widehat{W}_1 \widehat{W}_1^{-1} \\
& \times \widehat{W}_1 \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right) \left\{ \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right)^T \widehat{W}_1 (\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right\}^{-1}, \\
= & \left\{ \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right)^T \widehat{W}_1 (\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right\}^{-1}.
\end{aligned}
$$

Now, let's check the commands and runing results.

```
# Secondly, perform 2SGMM and
# the 1st step is like follows (looks like 2SLS).

Z = as.matrix(Z)
X = as.matrix(X)

ZX = t(Z) %*% X/n # or ZX = crossprod(Z, X)/n
ZY = t(Z) %*% Y/n

var_g_1 = t(Z) %*% Z/n

theta_1 = solve(t(ZX) %*% solve(var_g_1) %*% ZX) %*%
t(ZX) %*% solve(var_g_1) %*% ZY

cov_1   = solve(t(ZX) %*% solve(var_g_1) %*% ZX)


# Caculate the estimates and check
se_1  = sqrt(diag(cov_1))
t_1   = theta_1/se_1
p_1   = pt(abs(t_1), df = nrow(dt) - ncol(X), lower.tail = FALSE)*2

print("1st step estimates:"); theta_1

print("1st step se estimates:"); se_1

print("T statistics :"); t_1

print("P values:"); p_1
```

And the results are listed as

```
## [1] "1st step estimates:"
##                [,1]
## constant   37.7720297
```

```
## crime      -1.1413414
## industrial -0.4293433
## distance   -1.6688765
##
## [1] "1st step se estimates:"
## constant    crime     industrial distance
## 4.7137367  0.3971785  0.2482251  0.7364254
##
## [1] "T statistics :"
##                [,1]
## constant    8.013182
## crime      -2.873623
## industrial -1.729653
## distance   -2.266185
##
##[1] "P values:"
##                [,1]
## constant   7.881782e-15
## crime      4.229911e-03
## industrial 8.430688e-02
## distance   2.386493e-02
```

7.2.3 Step 2 of 2SGMM Method

Let's review the remaing part of the 2SGMM method,

Step 2 of the 2SGMM Method

From step 1, we have the estimator $\hat{\theta}_1$ in hand and if moments are assumed to be IID distributed, we have

$$\begin{aligned}
\widehat{Var}(g(K_i, \hat{\theta}_1))_2 &= \frac{1}{n}\sum_{i=1}^{n}(\mathbf{Z}_i^{IV})^T(Y_i - \mathbf{X}_i\hat{\theta}_1)^2(\mathbf{Z}_i^{IV}), \\
&= \frac{1}{n}(\underline{\mathbf{Z}}^{IV})^T(\underline{\mathbf{Y}} - \underline{\mathbf{X}}\hat{\theta}_1)(\underline{\mathbf{Y}} - \underline{\mathbf{X}}\hat{\theta}_1)^T(\underline{\mathbf{Z}}^{IV}), \\
\widehat{W}_2 &= \left(\widehat{Var}(g(K_i, \hat{\theta}_1))_2\right)^{-1}.
\end{aligned}$$

However, we already know that there exist heteroskedasticity and autocorrelation in this case, it's suggested by Chaussé (2010) that we should use a HAC variance covariance estimator. In this situation, denote $\hat{\underline{\mathbf{u}}} = \underline{\mathbf{Y}} - \underline{\mathbf{X}}\hat{\theta}_1$, we have

$$\begin{aligned}
Var(g(\underline{\mathbf{K}}, \hat{\theta}))_2 &= Var\left((\underline{\mathbf{Z}}^{IV})^T\hat{\underline{\mathbf{u}}}\right), \\
&= (\underline{\mathbf{Z}}^{IV})^T Var(\hat{\underline{\mathbf{u}}})\underline{\mathbf{Z}}^{IV}.
\end{aligned}$$

Denote that $\Omega = Var(\underline{\hat{\mathbf{u}}})$, if we have the NeweyWest covariance estimate of $\hat{\theta}_1$ as Q(check section 7.5 for definition), then

$$Q = \underline{\mathbf{X}}^T \widehat{\Omega} \underline{\mathbf{X}}.$$

Therefore with the generalized inverse operation, $\widehat{\Omega}$ can be calculated numerically by

$$\widehat{\Omega} = (\underline{\mathbf{X}}^T)^{-1} Q (\underline{\mathbf{X}})^{-1}.$$

Or to make it more comfortable to see[14],

$$\widehat{\Omega} = \left(\underline{\mathbf{X}}\underline{\mathbf{X}}^T\right)^{-1} \underline{\mathbf{X}} Q \underline{\mathbf{X}}^T \left(\underline{\mathbf{X}}\underline{\mathbf{X}}^T\right)^{-1}.$$

With these conditions above, it's able to get

$$\begin{aligned}
\widehat{Var}(g(K_i, \hat{\theta}_1))_2^{HAC} &= \frac{1}{n} (\underline{\mathbf{Z}}^{IV})^T \widehat{\Omega} \underline{\mathbf{Z}}^{IV}, \\
\widehat{W}_2^{HAC} &= \left(\widehat{Var}(g(K_i, \hat{\theta}_1))_2^{HAC}\right)^{-1}.
\end{aligned}$$

Here, $\widehat{W}_2$ is just the metric in sildes P201. But we finally use $\widehat{W}_2^{HAC}$. The results of these two metrics are provided at the end of this section.

The remaining parts are similar to those in step 1.

$$\begin{aligned}
\hat{\theta}_2 &= \left\{ \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right)^T \widehat{W}_2 (\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right\}^{-1} \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right)^T \widehat{W}_2 (\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{Y}}, \\
\mathbb{V}(\hat{\theta}_2) &= \left\{ \mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right]^T W_0 \mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right] \right\}^{-1} \mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right]^T W_0 Var\left(g(K_i, \theta_0)\right) \\
&\quad \times W_0 \mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right] \left\{ \mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right]^T W_0 \mathbb{E}\left[(\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right] \right\}^{-1}, \\
\hat{\mathbb{V}}(\hat{\theta}_2) &= \left\{ \left((\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}}\right)^T \widehat{W}_2 (\underline{\mathbf{Z}}^{IV})^T \underline{\mathbf{X}} \right\}^{-1},
\end{aligned}$$

where θ_0 is the true parameter with $Var\left(g(K_i, \theta_0)\right) = \mathbb{E}\left[g(K_i, \theta_0) g(K_i, \theta_0)^T\right]$ and $\widehat{W}_2 \xrightarrow{p} W_0$.

As usual, I show you the codes and results as below.

[14] Almostly same values were returned.

```
# Do 2nd step 2SGMM

# Theoretically, we should calculate the estimate of the variance with theta_1 by

Zu_1    = t(Z) %*% (Y - X %*% theta_1)
var_g_2 = Zu_1 %*% t(Zu_1)/n
# or var_g_1 = tcrossprod(Zu_1, Zu_1)/n

# But the previous var_g_2 is derived under the iid assumption.
# It is equal to set vcov = "iid" in gmm function,
# which would slow down the calculation.

# In practice, let's build a HAC covariance estimate from 2sls residuals and X.
# For simplicity, set L = 10(<= n).

u_1 = Y - X %*% theta_1
i = 1
l = 1
L = 10
weight = rep(0, L)

Q = matrix(0, nrow = ncol(X), ncol = ncol(X))
Q_1 = matrix(0, nrow = ncol(X), ncol = ncol(X))
Q_2 = matrix(0, nrow = ncol(X), ncol = ncol(X))

for (l in 1:L) {
    weight[l] = 1 - l/(L + 1)
}

for (i in 1:n) {
    Q_1 = Q_1 + u_1[i]^2 * (X[i,] %*% t(X[i,])) # X[i,] here returns column vector
}
Q_1 = Q_1/n

for (l in 1:L) {
    for (i in (l+1):n) {
        Q_2 = Q_2 + weight[l]*u_1[i]*u_1[i-l]*
        (X[i,] %*% t(X[i-l,]) + X[i-l,] %*% t(X[i,]))
    }
}
Q_2 = Q_2/n
Q = Q_1 + Q_2

XXT = X %*% t(X)
```

```r
omega = ginv(XXT) %*% (X %*% Q %*% t(X)) %*% ginv(XXT)
# Use ginv here to suppress error warnings from traditional inverse operation.
# omega = ginv(t(X)) %*% Q %*% ginv(X) returns almostly same values.
var_g_2 = t(Z) %*% omega %*% Z /n

theta_2 = ginv(t(ZX) %*% ginv(var_g_2) %*% ZX) %*%
t(ZX) %*% ginv(var_g_2) %*% ZY

cov_2   = ginv(t(ZX) %*% ginv(var_g_2) %*% ZX)


# Calculate the estimates and check
se_2  = sqrt(diag(cov_2))
se_2  = as.vector(se_2)
t_2   = theta_2/se_2
p_2   = pt(abs(t_2), df = nrow(dt) - ncol(X), lower.tail = FALSE)*2

print("2nd step estimates:"); theta_2

print("2nd step se estimates:"); se_2

print("T statistics :"); t_2

print("P values:"); p_2

# Finally, theta_1 is not optimal
se_1 >= se_2
```

Firstly, let me show you what will happen if $\widehat{W}_2$ in the IID case is used.

```
## [1] "2nd step estimates:"
##           [,1]
## [1,]  0.1411261
## [2,] -2.0908670
## [3,] -0.1512835
## [4,]  0.9792031
##
## [1] "2nd step se estimates:"
## [1] 0.4575280 6.7785496 0.4904583 3.1745572
##
## [1] "T statistics :"
##           [,1]
## [1,]  0.3084534
## [2,] -0.3084534
## [3,] -0.3084534
```

```
## [4,]  0.3084534
##
## [1] "P values:"
##         [,1]
## [1,] 0.7578653
## [2,] 0.7578653
## [3,] 0.7578653
## [4,] 0.7578653
```

As we can see, after the second step estimation, efficiency get worse than before and due to the sigularity of variance covariance matrix, we obtain the same but bad p values after generalized inverse operation is performed. Then, let's check the results using the NeweyWest covariance estimator .

```
## [1] "2nd step estimates:"
##         [,1]
## [1,] 31.2124282
## [2,] -0.7333242
## [3,] -0.3499875
## [4,] -0.6303447
##
## [1] "2nd step se estimates:"
## [1] 3.0612074 0.3907887 0.1751742 0.4126052
##
## [1] "T statistics :"
##         [,1]
## [1,] 10.196117
## [2,] -1.876524
## [3,] -1.997940
## [4,] -1.527719
##
## [1] "P values:"
##         [,1]
## [1,] 2.589106e-22
## [2,] 6.116328e-02
## [3,] 4.626223e-02
## [4,] 1.272123e-01
```

And we can check the optimality by logical operation command as

```
## se_1 >= se_2
## constant   crime industrial   distance
## TRUE       TRUE       TRUE       TRUE
```

7.2.4 Displaying the results

All the estimation results are given in table 13 and I omit the interpretations because they are not different from those in OLS estimation.

```
# Display the results
stargazer(
ols, ols, ols,
coef = list(coef(ols), theta_1, theta_2), se = list(se_hac, se_1, se_2),
t = list(t_hac, t_1, t_2), p = list(p_hac, p_1, p_2),
t.auto = FALSE, p.auto = FALSE,
report = "vcstp", keep.stat = c("n"),
add.lines = list(
c("Type", "HA-Roubusted OLS", "1st step GMM", "2nd step GMM")),
title = "Results of rols and numerical 2 step GMM",
label = "Numeric",
type = "latex", header = FALSE, font.size = "small",
table.placement = "htb", omit.table.layout = "n"
)
```

7.3 2SLS and 2SGMM by R

7.3.1 2SLS and Hausman Tests for Endogeneity

7.3.1.1 2SLS in R The works we did in section 7.2.2.1 can be simply performed by `AER::ivreg` function in R. I just show you the codes here and the result remains to be talked about a little later.

```
# Built-in R Functions

# 2SLS and Hausman Wu Test for Endogeneity (By ivreg).
model_iv = value ~ crime + industrial + distance |
    black + ptratio + industrial + distance

twoSLS = ivreg(model_iv, data = dt)
```

7.3.1.2 Durbin-Wu-Hausman Test for Endogeneity The DWH test looks similar to the Hausman test in chapter 5, talking about how to choose between FE model and RE model. Here, let me share a simple definition with you.

Durbin-Wu-Hausman Test

Assume that

$$y_i = \mathbf{X}_i\beta + u_i \quad \forall i = 1, \dots, N,$$

where u_i is a 0-meaned error term and $\mathbf{X}_i^T \in \mathbb{R}^K$.

Table 13: Results of rols and numerical 2 step GMM

	Dependent variable:		
	value		
	(1)	(2)	(3)
crime	−0.273	−1.141	−0.733
	(0.055)	(0.397)	(0.391)
	t = −4.926	t = −2.874	t = −1.877
	p = 0.00001	p = 0.005	p = 0.062
industrial	−0.730	−0.429	−0.350
	(0.142)	(0.248)	(0.175)
	t = −5.153	t = −1.730	t = −1.998
	p = 0.00000	p = 0.085	p = 0.047
distance	−1.016	−1.669	−0.630
	(0.375)	(0.736)	(0.413)
	t = −2.710	t = −2.266	t = −1.528
	p = 0.007	p = 0.024	p = 0.128
Constant	35.505	37.772	31.212
	(2.984)	(4.714)	(3.061)
	t = 11.899	t = 8.013	t = 10.196
	p = 0.000	p = 0.000	p = 0.000
Type	HA-Roubusted OLS	1st step GMM	2nd step GMM
Observations	506	506	506

Declare the null hypothesis $H_0 : \mathbb{E}[\mathbf{X}_i^T u_i] = 0$[15] and suppose that we have the ols estimator $\hat{\beta}_{ols}$ and gmm estimator $\hat{\beta}_{GMM}$, respectively. The test statistic is given by

$$\hat{H} = \left(\hat{\beta}_{ols} - \hat{\beta}_{GMM}\right)^T \left\{\widehat{Var}(\hat{\beta}_{GMM}) - \widehat{Var}(\hat{\beta}_{ols})\right\}^{-1} \left(\hat{\beta}_{ols} - \hat{\beta}_{GMM}\right) \xrightarrow{d} \chi^2(K)$$

It is also known that

- Under the null hypothesis, the estimator $\hat{\beta}_{ols}$ is efficient and the estimator $\hat{\beta}_{GMM}$ is consistent but typically not efficient.
- Under the alternative hypothesis, the estimator $\hat{\beta}_{ols}$ is inconsistent, the estimator $\hat{\beta}_{GMM}$ is consistent.

[15] Exogeneity condition meaning that all explanatory variables are exogenous.

Apply this method in R[16], we have the follows,

```
# Durbin-Wu-Hausman Test
dwh.test = function(model.iv, model.ols){
    cf_diff = coef(model.iv) - coef(model.ols)
    vc_diff = vcovHC(model.iv, "HC0") - vcovHC(model.ols, "HC0")
    # NeweyWest() doesn't fit well to model.iv so we use the White estimator.
    x2_diff = as.vector(t(cf_diff) %*% solve(vc_diff) %*% cf_diff)
    pvalue = pchisq(q = x2_diff, df = dim(vc_diff)[1], lower.tail = F)

    result = list(x2_diff, dim(vc_diff)[1], pvalue)
    names(result) = c("DWH Statistic", "df", "P value")
    return(result)
}
dwh.test(twoSLS, ols)
```

And then, the results of applying this test to our framework are shown as

```
## $`DWH Statistic`
## [1] 10.77423
##
## $df
## [1] 4
##
## $P value
## [1] 0.02922208
```

From this result, we know that the null hypothesis is rejected and there exists at least one endogenous variable.

7.3.1.3 Wu-Hausman Test in R

Indeed, R has already encompassed the Wu-Hausman 2 step test for endogeneity. I don't clearly specify this test and anyone interested is suggested to refer to Wooldridge (2010) for details.[17]

```
# "http://klein.uk/R/myfunctions.R" from professor Thilo Klein.
dwh.test(twoSLS, ols)

# Use "diagnostics = TRUE" calls a Wu-Hausman(2 step) F test
# (available in Wooldridge(2003)).
summary(twoSLS, vcov. = NeweyWest, df = Inf, diagnostics = TRUE)
```

Let's check the result.

```
## Call:
```

[16]The original code is posted at "http://klein.uk/R/myfunctions.R" from professor Thilo Klein. And I modified it a little bit.

[17]And be careful that the second step is replaced by the Fisher test in R but not the t test.

```
## ivreg(formula = model_iv, data = dt)
##
## Residuals:
##    Min      1Q     Median    3Q     Max
## -16.515  -6.069  -1.965   2.639  84.315
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  37.7720     3.3464  11.287  < 2e-16 ***
## crime        -1.1413     0.4339  -2.630 0.008527 **
## industrial   -0.4293     0.2126  -2.019 0.043452 *
## distance     -1.6689     0.4852  -3.440 0.000582 ***
##
## Diagnostic tests:
##                   df1 df2 statistic  p-value
## Weak instruments   2  501   5.921   0.00287 **
## Wu-Hausman         1  501   15.498  9.43e-05 ***
## Sargan             1  NA    17.923  2.30e-05 ***
## ---
## Signif. codes:  0  '***'  0.001  '**'  0.01  '*'  0.05  '.'  0.1  ' '  1
##
## Residual standard error: 10.25 on Inf degrees of freedom
## Multiple R-Squared: -0.2352, Adjusted R-squared: -0.2425
## Wald test: 35.98 on 3 DF,  p-value: 7.561e-08
```

Look at the Wu-Hausman row, the null hypothesis is rejected at 0.1% level. And the endogenous variable is indeed `crime` because we specify the structure in the previous model_iv .

7.3.2 2SGMM and The Sargan-Hansen (Sargan's J) Test in R

7.3.2.1 2SGMM in R The works we did in section 2 can also be conducted by `gmm::gmm` function in R. As usual, I just display the codes here.

```r
# 2SGMM and Sargan's J Test (By gmm)
value = dt[, "value"]
crime = dt[, "crime"]
industrial = dt[, "industrial"]
distance = dt[, "distance"]

instruments = dt[, c("black", "ptratio", "industrial", "distance")]

twoSGMM_iid = gmm(g = model, x = instruments, vcov = "iid")
twoSGMM_HAC = gmm(g = model, x = instruments, vcov = "HAC")

summary(twoSGMM_iid)
```

```r
summary(twoSGMM_HAC)
```

7.3.2.2 The Sargan-Hansen Test The Sargan-Hansen test is a statistical test used for testing over-identifying restrictions in a statistical model. It was proposed by Sargan (1958), and several variants were derived by Sargan (1988). Lars Peter Hansen re-worked through the derivations and showed that it can be extended to general non-linear GMM in a time series context (Hansen 1982).

As same as the notations in the slides, we denote that $dim(g) = L$ and $dim(\theta) = d$. When we have more instruments than we need to identify an equation, we can test whether the additional instruments are valid in the sense that they are uncorrelated with the error term.

In our case, we have the maybe endogenous explanatory variables and the exogenous instrumental variables as

$$\begin{aligned}\mathbf{X}_i &= (constant_i, crime_i, industrial_i, distance_i),\\ \mathbf{Z}_i &= (constant_i, black_i, ptratio_i, industrial_i, distance_i),\end{aligned}$$

respectively. So conducting the Sargan-Hansen test means that we want to check whether `black` and `ptratio` are good instruments. By the way, you can take `industrial` and `distance` as instruments of their own. For more details, please refer to Wooldridge (2010).

And then, with the statistical criterion $\hat{\theta}_{GMM} = \arg\max_{\theta\in\Theta} \left\{-(\frac{1}{n}\sum_{i=1}^{n} g(\mathbf{K}_i,\theta))^T \widehat{W} (\frac{1}{n}\sum_{i=1}^{n} g(\mathbf{K}_i,\theta))\right\}$, we are able to build the J statistic.

The null hypothesis is that $H_0 : \exists\theta, \mathbb{E}[g(\mathbf{K}_i,\theta)] = \mathbf{0}$, and the statistic is

$$\hat{J} = (\frac{1}{\sqrt{n}}\sum_{i=1}^{n} g(\mathbf{K}_i,\hat{\theta}))^T \widehat{W} (\frac{1}{\sqrt{n}}\sum_{i=1}^{n} g(\mathbf{K}_i,\hat{\theta})) \xrightarrow{d} \chi^2(L-d)$$

The statistic is then based on the optimal GMM estimator.

This test is already included in `gmm::gmm` function, to look for the result, please use ‘base::summary function.

```
## Call:
## gmm(g = model, x = instruments, vcov = "HAC")
##
## Method:  twoStep
##
## Kernel:  Quadratic Spectral(with bw =  1.54322 )
##
## Coefficients:
##                Estimate     Std. Error   t value      Pr(>|t|)
```

```
## (Intercept)   3.8101e+01   3.2027e+00   1.1897e+01   1.2331e-32
## crime        -1.1011e+00   3.4308e-01  -3.2095e+00   1.3297e-03
## industrial   -4.6190e-01   1.8771e-01  -2.4607e+00   1.3867e-02
## distance     -1.7307e+00   4.4494e-01  -3.8898e+00   1.0032e-04
##
## J-Test: degrees of freedom is 1
##                 J-test    P-value
## Test E(g)=0:    5.698567  0.016979
##
##
## Initial values of the coefficients
## (Intercept)    crime     industrial   distance
## 37.7720297  -1.1413414  -0.4293433  -1.6688765
```

Focus on the "J-Test" part, since the p value is smaller than 0.02, we can say that the null hypothesis is rejected at 5% level and this model specification is indeed not correct. Better instruments are wanted for making a proper estimation.

```
# Display
stargazer(
twoSLS, twoSGMM_iid, twoSGMM_HAC,
report = "vcstp", keep.stat = c("n"),
add.lines = list(
c("Type", "2SLS", "2SGMM(iid)", "2SGMM(HAC)")),
title = "Results of 2SLS and 2SGMM by R",
label = "byR",
type = "latex", header = FALSE, font.size = "small",
table.placement = "htb", omit.table.layout = "n"
)
setwd("~")
```

7.3.2.3 Displaying the results Look at table 14, you may find that 2SLS and 2SGMM(iid) returned similar estimates for that IID assumption is not satisfied in this case. And the reason why 2SGMM(HAC) didn't give the optimal estimates is that, by the Sargan-Hansen test, we know that our model specification is indeed not correct, therefore such results were returned.

7.4 (Appendix) Properties of the GMM estimator

Under suitable conditions, the GMM estimator satisfies

$$\hat{\theta} \xrightarrow{p} \theta_0,$$
$$\sqrt{n}(\hat{\theta} - \theta_0) \xrightarrow{d} \mathcal{N}_{\mathbb{R}^{dim(\theta)}}(\mathbf{0}, \mathbb{V}(\hat{\theta})),$$

Table 14: Results of 2SLS and 2SGMM by R

	Dependent variable:		
	value		
variable			
	(1)	(2)	(3)
crime	−1.141	−1.141	−1.101
	(0.181)	(0.180)	(0.343)
	t = −6.305	t = −6.330	t = −3.209
	p = 0.000	p = 0.000	p = 0.002
industrial	−0.429	−0.429	−0.462
	(0.113)	(0.113)	(0.188)
	t = −3.795	t = −3.810	t = −2.461
	p = 0.0002	p = 0.0002	p = 0.014
distance	−1.669	−1.669	−1.731
	(0.336)	(0.334)	(0.445)
	t = −4.972	t = −4.992	t = −3.890
	p = 0.00000	p = 0.00000	p = 0.0002
Constant	37.772	37.772	38.101
	(2.148)	(2.140)	(3.203)
	t = 17.582	t = 17.652	t = 11.897
	p = 0.000	p = 0.000	p = 0.000
Type	2SLS	2SGMM(iid)	2SGMM(HAC)
Observations	506	506	506

with

$$\mathbb{V}(\hat{\theta}) = \left\{G^T W_0 G\right\}^{-1} G^T W_0 Var\left(g(K_i, \theta_0)\right) W_0 G \left\{G^T W_0 G\right\}^{-1},$$

and

$$Var\left(g(K_i, \theta_0)\right) = \mathbb{E}\left[g(K_i, \theta_0) g(K_i, \theta_0)^T\right], \widehat{W} \xrightarrow{p} W_0.$$

Be careful that G is the population level Jacobian

$$G = G(\theta_0) = \mathbb{E}\left[\nabla_\theta g(K_i, \theta_0)\right] \in \mathcal{M}_{dim(g) \times dim(\theta)}(\mathbb{R})$$

7.5 (Appendix) NeweyWest Variance Covariance Estimator

According to Newey and West (1987), if $\mathbf{X}_i$ is the 1×d row vector and e_i is the residual, for all i = 1, ..., n. Then NeweyWest estimator should be

$$Q = \frac{1}{n}\sum_{i=1}^{n} e_i^2 \mathbf{X}_i^T \mathbf{X}_i + \frac{1}{n}\sum_{l=1}^{L}\sum_{i=l+1}^{n} w_l e_i e_{i-l}\left(\mathbf{X}_i^T \mathbf{X}_{i-l} + \mathbf{X}_{i-l}^T \mathbf{X}_i\right),$$
$$w_l = 1 - \frac{l}{L+1},$$

where w_l can be thought of as a "weight". Disturbances that are farther apart from each other are given lower weight, while those with equal subscripts are given a weight of 1.

8 Empirical Application of Time Series Model: Nikkei 225

8.1 Background and Data

The "Nikkei225" is a stock price index published by Nihon Keizai Shimbun (hereafter, NIKKEI). NIKKEI calculates this price index based on 225 high liquid brands listed with first section of the Tokyo Stock Exchange. We use daily data of the Nikkei 225 index taken from the yahoo finance (https://stocks.finance.yahoo.co.jp/stocks/detail/?code=998407.O). The time length is from January 4th 2019 to January 22 2021. We have 498 observations. We read a csv data which recodes the Nikkei 225 dairy index, using the `read.csv` function in `R`. Since `R` recognize a time variable (e.g., `2021/01/22`) as a character string, we need to define a time variable, using `as.Date()` function. The data structure is as follows:

```
library(tidyverse)
dt <- read.csv("data/nikkei225.csv", stringsAsFactor = FALSE)
dt$date <- as.Date(dt$date, format = "%Y/%m/%d")
head(dt)
```

```
##         date open_price high_price low_price close_price
## 1 2021-01-22   28580.20   28698.18  28527.16    28631.45
## 2 2021-01-21   28710.41   28846.15  28677.61    28756.86
## 3 2021-01-20   28798.74   28801.19  28402.11    28523.26
## 4 2021-01-19   28405.49   28720.91  28373.34    28633.46
## 5 2021-01-18   28238.68   28349.97  28111.54    28242.21
## 6 2021-01-15   28777.47   28820.50  28477.03    28519.18
```

There are five variables:

- `date`: date variable
- `open_price`: open price in day t

- `high_price`: high price in day t
- `low_price`: low price in day t
- `close_price`: closed price in day t

Mainly, we use the `date` and `close_price`.

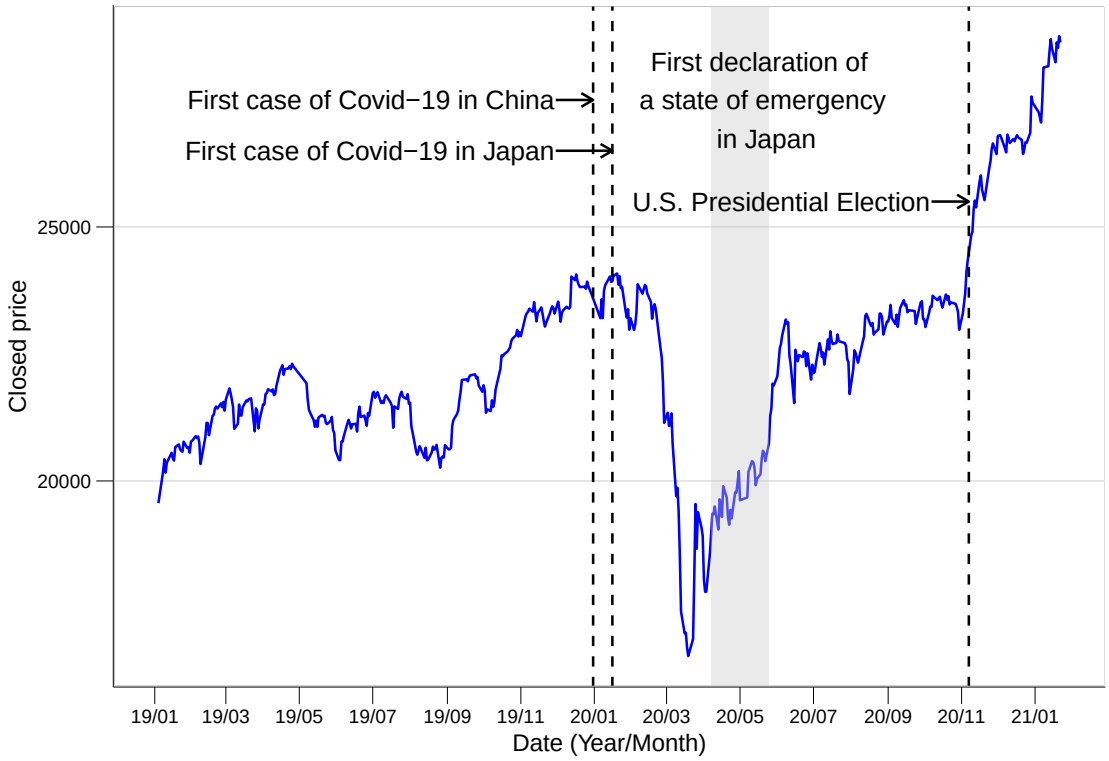

Figure 3: Time Series Data of Nikkei 225 (Closed Price)

Figure 3 shows the time series of closed price of the Nikkei225. We summarize some features as follows:

- After the COVID-19 occured in Japan and China, the Nikkei225 has drastically decreased.
- During the first declaration of a state of emergency in Japan, the Nikkei225's performance has been a V-shaped recovery.
- The Nikkei225 has sharply increased immediaterly before and after the U.S. presidential election.

Some may wonder if the negative shock of COVID-19 reflects the Nikkei225. To discuss it, we need to consider following two potential concerns.

1. unlisted companies (such as small restaurant business) may suffer heavily from the negative shock of COVID-19.
2. the Nikkei225 does not represent a variation of price index of 225 brands. In principle, the Nikkei225 is a mathematical mean of stock price of 225 brands. In fact, the the stock prices of top five brands which contribute to the Nikkei225 have increased at

70%. On the other hand, the stock price of other brands have decreased at 5% from the begging of 2020.[18]

Anyway, we test the stationarity of this time series.

8.2 Autoregressive of Order 1: AR(1) model

To check the stationarity of this time series, we consider the following AR(1) model:

$$X_t = \beta X_{t-1} + \epsilon_t,$$

where X_t is the closed price of Nikkei225 in day t. We assume (ϵ_t) is a white noise process, $(\epsilon_t) \sim WN(0, \sigma^2)$. that is, $E(\epsilon_t) = 0$, $E(\epsilon_t^2) = \sigma^2$, and $Cov(\epsilon_t, \epsilon_{t+h}) = 0$ for $h \neq 0$.

To estimate the unkown parameters $\theta = (\beta, \sigma^2)$, we use the maximum likelihood method. We assume $\epsilon_t \sim N_{\mathbb{R}}N(0, \sigma^2)$ for inference purposes. Then, the conditional log-likelihood is

$$M_T(X_1, \dots, X_T; \theta) = \frac{1}{T} \sum_{t=2}^{T} \left\{ -\log(2\pi\sigma^2) - \frac{X_t - \beta X_{t-1}}{2\sigma^2} \right\}.$$

The MLE $\hat{\theta}$ can be obtained by solving

$$\hat{\theta} = \arg\max_{\theta} M_T(X_1, \dots, X_T; \theta).$$

In `R`, the library calles `astsa` provides a function to estimate AR(p) model, named `sarima()`. Originally, this function estimates the (seasonal) ARIMA model, using the maximum likelihood method. Note that the ARIMA(p, q, d) model is

$$X_t - X_{t-d} = c + \sum_{i=1}^{p} \beta_i X_{t-i} + \sum_{i=1}^{Q} \theta_i \epsilon_{t-i}.$$

The second term corresponds to AR(p), and the third term corresponds to MA(q). This function passes five augments, a time-series data, autoregressive of order (p), moving average of order (q), d-the difference series, and whether to include constatnt term c. To estimate AR(1) model as described above, we specify `p = 1`, `q = 0`, `d = 0`, and `no.constant = TRUE`. The R snippet is as follows:

```
ar1 <- sarima(dt$close_price, p = 1, q = 0, d = 0, no.constant = TRUE)
```

```
sprintf(
  "The estimated beta is %1.5f (s.e. = %1.5f)",
  ar1$fit$coef, sqrt(ar1$fit$var.coef)
)
```

```
## [1] "The estimated beta is 0.99994 (s.e. = 0.00009)"
```

[18] See https://news.yahoo.co.jp/articles/f63a4627b298857a62ac329b1ed41a88c2721bd4.

```
sprintf("The estimated squared sigma is %1.2f", ar1$fit$sigma2)
```

```
## [1] "The estimated squared sigma is 74056.79"
```

8.3 Dickey-Fuller test

First, we derive the stationary condition in AR(1) model. The N-time iterated substition of X_{t-1} yields

$$X_t = \beta^N X_{t-N} + \sum_{k=0}^{N} \beta^k \epsilon_{t-k}.$$

If $|\beta| < 1$, then the first term of right hand side coverges to zero as $N \to \infty$. Thus, under $|\beta| < 1$, the causal stationary solution is $X_t = \sum_{k=0}^{\infty} \beta^k \epsilon_{t-k}$.

Testing for stationarity is quivalent to test for $\beta = 1$ in AR(1). The null hypothesis is that the process is **not** stationary. The alternative hypothesis is that the process is stationary, that is, $\beta < 1$. The Dickey-Fuller test provides this *one-sided* test.

To implement this test with `R`, we ues the package called `tseries`. We use the function called `adf.test()` to carry out the Dickey-Fuller test. We need to pass two augments in this function. The first augment is time-series data. The second one, named `k`, is the number of order (p). In this example, we pass `k = 1`, that is, AR(1).

```
library(tseries)
df1 <- adf.test(dt$close_price, k = 1)
sprintf("The DF stats is %1.4f (p-value = %1.4f)", df1$statistic, df1$p.value)
```

```
## [1] "The DF stats is -2.7638 (p-value = 0.2550)"
```

As a result, we cannot reject the null hypothesis. Thus, the time series of closed price of Nikkei225 is not stationary when we use data from January 4th 2020 to January 22 2021.

8.4 Stationarity of Log Return of Nikkei225

We showed that the time series of closed price of Nikkei225 is not stationary. Then, is the log return of Nikkei225 stationary?

To check it, we first construct the log return of Nikkei225, using the time series of closed price. To construct log return variable, we use the `dplyr` library. First, after reading dataset `dt`, we sort dataset by date using `arrange()` function. Second, we make the lagged closed price, using `lag()` function. This function picks up the variable contained in the previous `n`-the row. Finally, we make the log return, using `mutate()` function which makes a new variable. As a result, the data structure is as follows:

```
dt <- dt %>%
  arrange(date) %>%
  mutate(lag_close_price = dplyr::lag(close_price, n = 1)) %>%
```

```
  mutate(log_return = log(close_price) - log(lag_close_price))
head(dt[,c("date", "close_price", "lag_close_price", "log_return")])
```

```
##         date close_price lag_close_price   log_return
## 1 2019-01-04    19561.96              NA           NA
## 2 2019-01-07    20038.97        19561.96  0.024092014
## 3 2019-01-08    20204.04        20038.97  0.008203707
## 4 2019-01-09    20427.06        20204.04  0.010977908
## 5 2019-01-10    20163.80        20427.06 -0.012971575
## 6 2019-01-11    20359.70        20163.80  0.009668539
```

Figure 4 plots the time series of the log return. After the COVID-19 pandemic, the absolute value of log return is large. However, the overall time series of this index seems to be stationary.

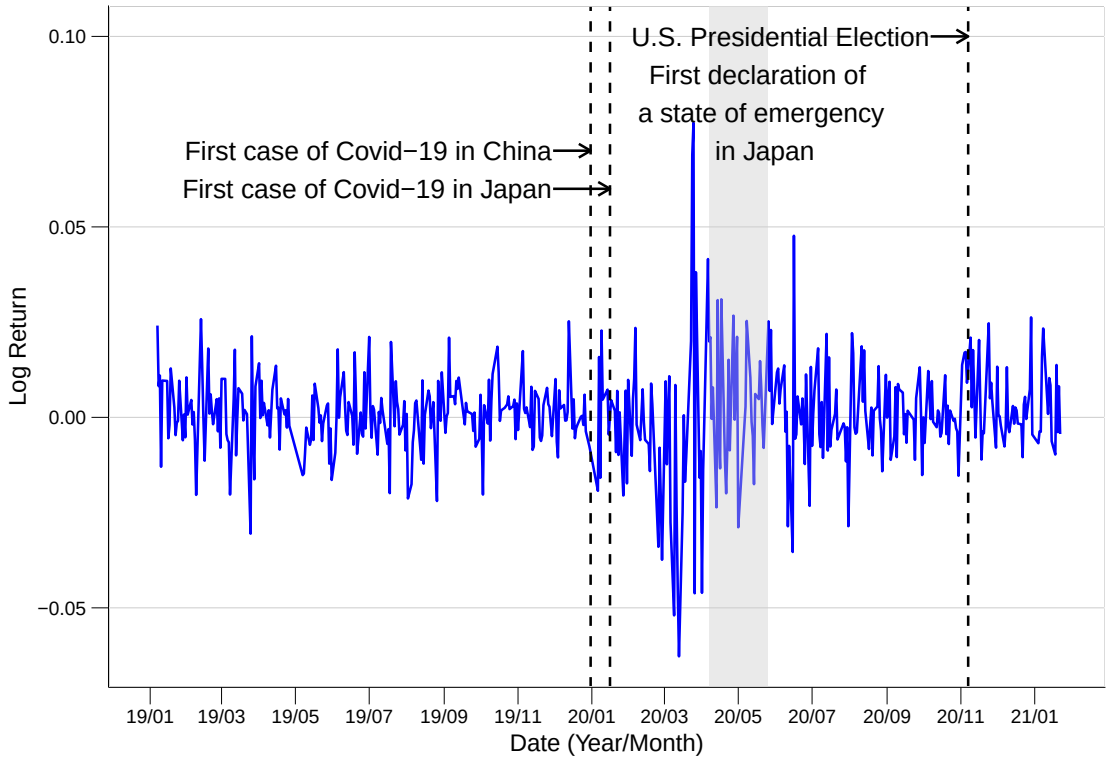

Figure 4: Time Series Data of Nikkei 225 (Log Return)

We estimate AR(1) model, using this time series data.

```
ar2 <- sarima(dt$log_return, p = 1, q = 0, d = 0, no.constant = TRUE)
```

```
sprintf(
  "The estimated beta is %1.5f (s.e. = %1.5f)",
  ar2$fit$coef, sqrt(ar2$fit$var.coef)
)
```

```
## [1] "The estimated beta is 0.06062 (s.e. = 0.04489)"
```

```
sprintf("The estimated squared sigma is %1.4f", ar2$fit$sigma2)
```

```
## [1] "The estimated squared sigma is 0.0002"
```

Using `adf.test()` function in the library `tseries`, we statistically test the stationarity of time series of log return.

```
x <- dt[dt$date != as.Date("2019/01/04"), "log_return"]
df2 <- adf.test(x, k = 1)
sprintf("The DF stats is %1.4f (p-value = %1.4f)", df2$statistic, df2$p.value)
```

```
## [1] "The DF stats is -13.8674 (p-value = 0.0100)"
```

As a result, we can reject the null hypothesis. This implies that the time series of the log return of Nikkei225 is stationary.

9 Reference

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

Breusch, T. S., and A. R. Pagan. 1979. "A Simple Test for Heteroscedasticity and Random Coefficient Variation." *Econometrica* 47 (5): 1287–94. http://www.jstor.org/stable/1911963.

Chaussé, Pierre. 2010. "Computing Generalized Method of Moments and Generalized Empirical Likelihood with R." *Journal of Statistical Software* 34 (i11). https://doi.org/http://hdl.handle.net/10.

Durbin, J., and G. S. Watson. 1950. "Testing for Serial Correlation in Least Squares Regression: I." *Biometrika* 37 (3/4): 409–28. http://www.jstor.org/stable/2332391.

———. 1951. "Testing for Serial Correlation in Least Squares Regression. II." *Biometrika* 38 (1/2): 159–77. http://www.jstor.org/stable/2332325.

Hansen, Christian B. 2007. "Generalized Least Squares Inference in Panel and Multilevel Models with Serial Correlation and Fixed Effects." *Journal of Econometrics* 140 (2): 670–94. https://doi.org/https://doi.org/10.1016/j.jeconom.2006.07.011.

Hansen, Lars Peter. 1982. "Large Sample Properties of Generalized Method of Moments Estimators." *Econometrica* 50 (4): 1029–54. http://www.jstor.org/stable/1912775.

Harrison, David, and Daniel L Rubinfeld. 1978. "Hedonic Housing Prices and the Demand for Clean Air." *Journal of Environmental Economics and Management* 5 (1): 81–102. https://doi.org/https://doi.org/10.1016/0095-0696(78)90006-2.

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

Newey, Whitney K., and Kenneth D. West. 1987. "A Simple, Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix." *Econometrica* 55 (3): 703–8. http://www.jstor.org/stable/1913610.

Nichols, Austin. 2006. "Weak Instruments: An Overview and New Techniques." North American Stata Users' Group Meetings 2006. Stata Users Group. https://EconPapers.repec.org/RePEc:boc:asug06:3.

Sargan, J. D. 1958. "The Estimation of Economic Relationships Using Instrumental Variables." *Econometrica* 26 (3): 393–415. http://www.jstor.org/stable/1907619.

———. 1988. "Testing for Misspecification After Estimating Using Instrumental Variables." In *Contributions to Econometrics*, edited by E Maasoumi. Cambridge: Cambridge University Press.

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

Wooldridge, Jeffrey M. 2010. *Econometric Analysis of Cross Section and Panel Data*. MIT Press.