Зміст

навчальних відео із використання R та Rstudio для статистичного аналізу та базової економетрики. Поточний документ містить короткий опис кожного окремого уроку присвяченого R із змістом усіх відео і програмним кодом, який міститься у додатках. Загальна тривалість усіх приблизно 15 годин. Виконавець – Букін Едуард.

1. Встановлення R та R Studio

В першому уроку ми ознайомимося із тим як саме встановлювати R та R Studio на комп'ютері із операційною системою Windows. Усі інструкції із встановлення надані у відео. Таблиця нижче (для цього і всіх наступних уроків) пояснює структуру уроку, і місце знаходження відео файлів, папок і скриптів пов'язаних із уроком.

	Папка із файлами теми	\urok_1\			
	Файл із кодом теми	\urok_1\1 - Vstanovlennia R.R			
	Додаток із змістом і кодом теми	Додаток 1			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
1	1. Встановлення R та R Studio	0.00	\urok_1\urok-1.mp4	5	28

2. Перше знайомство із R

В уроці розкриваються основи використання R та основні базові елементи інтерфейсу ядра R. Цей матеріал викладається для того, щоб ознайомити користувачів із базовим функціоналом. У наступних відео ми застосовуватимемо інтерфейс R Studio для роботи в R.

	Папка із файлами теми	\urok_2\			
	Файл із кодом теми	\urok_2\2 - Pershe znaiomstvo is R.R			
	Додаток із змістом і кодом теми	Додаток 2			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
2	2. Перше знайомство із R	0.00	\urok_2\urok-2-Part-2.1-2.3 .mp4	6	28
	2.1 командний рядок - консоль R	5.00			
	2.2 виконання команд в R консолі	5.30			
	2.3 основи синтаксису - базові команди та арифметичні операції	0.00	\urok_2\urok-2-Part-2.3 .mp4	15	45
	2.4 приклад візуалізації і графіки	0.00	\urok_2\urok-2-Part-2.4-2.8 .mp4	21	40
	2.5 довідка і допомога в R	5.48			

	2.6 робоча директорія і чому вона важлива	9.39			
	2.7 відкриття та збереження скриптів	18.40			

3. Інтерфейс 'R Studio' для базових операцій

Розкриваються основні елементи інтерфейсу R Studio для роботи із мовою програмування R. Особлива увага має бути приділена роботі із скриптами і робочою директорією.

	Папка із файлами теми	\urok_3\			
	Файл із кодом теми	\urok_3\3 - Interface R Studio.R			
	Додаток із змістом і кодом теми	Додаток 3			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
3	3. Інтерфейс 'R Studio' для базових операцій	0.00	\urok_3\urok-3-Part-3.1-3.6.mp4	17	15
	3.1 командний рядок - консоль R в інтерфейсі 'R Studio'	1.00			
	3.2 скрипти та коментарі у скрипті	5.20			
	3.3 розділи інтерфейсу 'R Studio'	8.20			
	3.4 пересування між вікнами	10.30			
	3.5 робоча директорія	13.04			
	3.6 використання проектів в R Studio	00.00	\urok_3\urok-3-Part-3.6-3.7.mp4	12	10
	3.7 додаткові ресурси для навчання	10.00			

4. Основи роботи із R та основи синтаксису

Особлива увага в цьому уроці приділяється синтаксису мові програмування R та основним командам, які необхідні для подальшої роботи із R.

	Папка із файлами теми	\urok_4\			
	Файл із кодом теми	\urok_4\R\4 - Osnovi roboti is R ta sintaksis.R			
	Додаток із змістом і кодом теми	Додаток 4			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
4	4. Основи роботи із R та основи синтаксису.	00.00	\urok_4\urok-4-Part-4.0-4.2.mp4	21	12
	4.1 Основи синтаксису	4.15			
	4.2 об'єкти в R та призначення об'єктів	0.00	\urok_4\urok-4-Part-4.2.mp4	45	50
	4.3 базові арифметичні функції в R	0.00		55	40

	4.4 типові помилки, підчас роботи із R та як їх розуміти	17.20	\urok_4\urok-4-Part-4.3-4.5.mp4		
	4.5 пакети в R	38.10			

5. Основи завантаження, маніпуляції та збереження даних за допомогою R

Цей урок базується на досить розповсюдженій проблемі пов'язаній із аналізом даних в R – завантаження даних. Особлива увага приділяється тому як завантажити дані із різних джерел і файлів різного формату.

	Папка із файлами теми	\urok_5\			
	Файл із кодом теми	\urok_5\R\5 - Zavantazhennia danih.R			
	Додаток із змістом і кодом теми	Додаток 5			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
5	5. Основи завантаження, маніпуляції та збереження даних за допомогою R	0.00	\urok_5\urok_5_Part_5.0-5.1.mp4	29	21
	5.0 Встановлення коректної робочої директорії та завантаження пакетів	3.00			
	5.1 Основи роботи із об'єктом `data.frame` та/або `data_frame`	9.13			
	5.2 Завантаження/відкриття табличних даних використовуючи базовий функціонал R	0.00	\urok_5\urok_5_Part_5.2.mp4	42	55
	5.3 Те саме, що і в 5.2 але із більш зручним пакетом 'readr'	0.00	\urok_5\urok_5_Part_5.3.mp4	35	55
	5.4 Відкриття табличних даних у форматі Excel (readxl)	0.00	\urok_5\urok_5_Part_5.4-5.5.mp4	41	9
	5.5 Перейменування назв колонок в табличних даних	0.00	\urok_5\urok_5_Part_5.5.mp4	2	46
	5.6 Змінна/колонка таблиці/`data_frame` як вектор.	0.00	\urok_5\urok_5_Part_5.6.mp4	7	25
	5.7 Виправлення типів змінних у табличних даних	0.00	\urok_5\urok_5_Part_5.7-5.9.mp4	26	20
	5.8 Базові операції із вектором простими методами	12.13			
	5.9 Збереження таблиць	16.14			

6. Базові операції із табличними даними

Будь-який аналіз базується на даних, тому вкрай важливо вміти маніпулювати даними із метою прискорення і поліпшення проведення аналізу. В поточному уроці основна увага приділяється саме тому, як працювати із табличними даними в R.

	Папка із файлами теми	\urok_6\			
	Файл із кодом теми	\urok_6\R\6 - Manipulazia danih.R			
	Додаток із змістом і кодом теми	Додаток 6			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
6	6. Базові операції із табличними даними	0.00	\urok_6\urok_6_Part_6.0-6.2.mp4	35	55
	6.0 Підготовка середовища	1.23			
	6.1 Перейменування деяких змінних	11.15			
	6.2 Вибір/виокремлення змінних/колонок із однієї таблиці	17.02			
	6.3 Перегляд змісту змінних	0.00	\urok_6\urok_6_Part_6.3.mp4	6	30
	6.4 Фільтрація таблиць по одній або декількох змінних	0.00	\urok_6\urok_6_Part_6.4.mp4	17	30
	6.5 Сортування таблиць	0.00	\urok_6\urok_6_Part_6.5-6.7.mp4	36	37
	6.6 Створення нових змінних	4.30			
	6.7 Редагування змісту змінних	20.00			
	6.8 Підбиття підсумків по колонкам/змінним	0.00	\urok_6\urok_6_Part_6.0-6.2.mp4	21	54

7. Описова статистика та базовий графічний аналіз

Перед будь-яким економетричним аналізом важливо провести повноцінний візуальний аналіз даних, які аналізуються. Цей урок фокусується на методах візуального аналізу даних в R.

	Папка із файлами теми	\urok_7\			
	Файл із кодом теми	\urok_7\R\7-descriptive-statistic.R			
	Додаток із змістом і кодом теми	Додаток 7			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
7	7. Описова статистика (Descriptive statistics) та базовий графічний аналіз	0.00	\urok_7\urok_7_Part_7.0-7.5.mp4	84	20
	7.0 завантаження тренувальних даних та підготовка робочого середовища	0.00			
	7.1 Основна описова статистика окремої змінної	14.00			
	7.2 Scatter Plot	21.00			

	7.3 Box-plot	33.00			
	7.4 Distribution histogram та Density Plot	44.20			
	7.5 Кореляція між змінними та статистична значимість кореляції	54.00			

8. Звичайна лінійна регресія в R

Один із центральних інструментів економетричного аналізу це лінійна регресія, чому і присвячений цей урок.

	Папка із файлами теми	\urok_8\			
	Файл із кодом теми	\urok_8\R\8-Linear-regression.R \urok_8\R\8-Linear-regression-bonus.R			
	Додаток із змістом і кодом теми	Додаток 8			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
8	8. Звичайна лінійна регресія в R	0.00	\urok_8\urok_8_zart_8.0.mp4	15	8
	8.0 Підготовка середовища та завантаження тренувальних даних	0.00			
	8.1 Побудова базової лінійної регресії	0.00	\urok_8\urok_8_zart_8.1.1.mp4 \urok_8\urok_8_zart_8.1.2.mp4	36	48
	8.2 Перегляд результатів регресії	0.00	\urok_8\urok-8-part-8.2-8.3.mp4	27	15
	8.3 Побудова діагностичних графіків	0.00	\urok_8\urok-8-part-8.3.mp4	12	52
	8.4 Лінійність та нормальність розподілу залишків	0.00	\urok_8\urok-8-part-8.4.mp4	14	36
	8.5 Однорідність розподілу залишків або постійна варіація залишків	0.00	\urok_8\urok-8-part-8.5.mp4	22	36
	8.6 Мультиколінеарність	0.00	\urok_8\urok-8-part-8.6.mp4	5	21
	8.7 Не лінійність	0.00	\urok_8\urok-8-part-8.7.mp4	2	5
	8.8 Незалежність залишків	0.00	\urok_8\urok-8-part-8.8-8.9.mp4	15	23
	8.9 Ідентифікація аутлаєрів та впливових величин	5.00			
	8.10 «Бонус» вправа із побудови лінійної регресії	0.00	\urok_8\urok-8-part-8.10-bonus.mp4	58	0

9. Панельна регресія в R

Останній урок присвячено одному із досить важливих інструментів економетричного аналізу – панельній регресії.

	Папка із файлами теми	\urok_9\			
	Файл із кодом теми	\urok_9\R\9-Panel data.R			
	Додаток із змістом і кодом теми	Додаток 9			
№	Тема	Час теми у файлі	Шлях до відео файлу	Тривалість відео	
				Хвилини	Секунди
9	9. Панельна регресія в R	0.10	\urok_9\urok-9-part-9.0-9.2.mp4	13	45
	9.0 Підготовка середовища та завантаження тренувальних даних	1.00			
	9.1 Візуальний аналіз панельних даних	5.23			
	9.2 Побудова базової лінійної регресії	0.00	\urok_9\urok-9-part-9.3-9.5.mp4	41	23
	9.3 "Панельна регресія" із фіксованим ефектом метод "dummy variables"	15.36			
	9.4 Панельна регресія із фіксованим ефектом	20.25			
	9.5 Перегляд результатів панельної регресії та коефіцієнтів	32.46			
	9.6 Визначення чи є доцільність у панельній регресії	0.30	\urok_9\urok-9-part-9.6-9.12.mp4	42	30
	9.7 Побудова панельної регресії із випадковим ефектом	6.00			
	9.8 Визначення, що краще: фіксований чи випадковий ефект	10.25			
	9.9 Тестування крос-секційної залежності та кореляції у часі	15.30			
	9.10 Breusch-Godfrey/Wooldridge тест серійної кореляції	16.21			
	9.11 Dickey-fuller тест про стохастичний тренд та стаціонарність	18.30			
	9.12 Перевірка припущень	23.20			

Додатки

Додаток 1. Встановлення R та R Studio

```
# Крок 1.
# Завантажуємо R за посиланням:
#       https://cran.r-project.org/bin/windows/base/

# Крок 2.
# Завантажуємо R Studio за посиланням:
#       https://www.rstudio.com/products/rstudio/download/#download

# Крок 3.
# Встановлюємо R, запускаючи файл, який ми щойно завантажили, не змінюючи ніяких параметрів.

# Крок 4.
# Встановлюємо R Studio, запускаючи файл, який ми щойно завантажили, не змінюючи ніяких параме
трів.
```

Додаток 2. Перше знайомство із R

```
# 2.1 командний рядок - консоль R; --------------------------------------

# 2.2 виконання команд в R консолі; --------------------------------------

# 2.3 основи синтаксису - базові команди та арифметичні операції; ---------

# Команди введення
1
2
12312312003123019312
0.2
"abc"
TRUE

# Команди призначення
a <- 1
b <- 2
A <- 10
B <- 20

# Прості арифметичні операції
1 + 2
a + b
A+B

#   збереження примітивних об'єктів;
x <- "abc    abc     abc"
y <- "ABC"

# Чому наступне дає помилку?
x + y

# Функції

# Друку об'єкту у консолі
print(     25    )          # Друку об'єкту у консолі
print(x)
print(     x     )
print(y)

# функція створення вектору
xy <- c(x, y)
xy

c(1, 2, 5, 10)
print(xy)

# Чому наступне дає помилку?
print(x, y)

# 2.4 приклад візуалізації і графіки; ------------------------------------

# візуалізація даних введених вручну
xx <- c(1,2,3,4,5,6,7,8,9,2,5,3,7,5,8,9,2)
yy <- c(2,5,7,1,6,2,3,4,7,9,3,3,5,6,8,9,3)

plot(xx, yy)
```

```r
# візуалізація випадкових даних
xxx <- rnorm(100)
yyy <- rpois(100, 2)

plot(xxx, yyy)

# гістограма випадкової величини
hist(xxx)
hist(yyy)

# 2.5 довідка і допомога в R; ----------------------------------------------

help(print)
?print
print

# 2.6 робоча директорія і чому вона важлива; -----------------------------

?getwd
getwd()

?setwd
setwd("~")

dir()
list.files()

# 2.7 відкриття та збереження скриптів; ----------------------------------
```

Додаток 3. Інтерфейс ‘R Studio’ для базових операцій

```
# 3.1 командний рядок - консоль R в інтерфейсі 'R Studio'; ----------------

# 3.2 скрипти та коментарі у скрипті; -------------------------------------
# відкриття, виконання та збереження скриптів
# як використовувати скрипти надані разом із цими відео

# 3.3 розділи інтерфейсу 'R Studio'; --------------------------------------
# файли,
# довідка,
# панель візуалізації,
# середовище R,

# 3.4 пересування між вікнами; --------------------------------------------

# 3.5 робоча директорія; --------------------------------------------------

# 3.6 використання проектів в R Studio; -----------------------------------

# Рекомендована практика для організації папок у проекті
# =="project_folder" - Проектна папка - будь-яка назва проекту на Ваш вибір латинськими літера
ми
#   |== "R" - папка, де зберігається виключно програмний код (стандартна назва)
#   |== "data" - папка, де зберігаються дані
#   |== "documents" - папка для документації, інструкцій та пояснень
#   |== "output" - папка для збереження результатів аналізу

# 3.7 додаткові ресурси для навчання; -------------------------------------

# https://www.rstudio.com/resources/webinars/rstudio-essentials-webinar-series-part-1/
```

Додаток 4. Основи роботи із R та основи синтаксису

```r
# 4.1 Основи синтаксису ------------------------------------

# Змінні та як їх назвати;

wheatUkrData
wheatUkr <- 2
wheat_ukr <- 5
ukraine.wheat
ukr.123.wheat

ukr.wheat <- 1

# Пробіли та розбірливість коду;

ukr 123 wheat
ukr.wheat
123.ukr.wheat

# ukr.wheat.007 <- 100

# Довжина одного рядка коду;

# Скрипт та його структура;

# 4.2 об'єкти в R та призначення об'єктів; -------------------------------

# створення (призначення) об'єктів

a <- 1
a
b <- 2
b
D <- a + b
D

# перевизначення об'єктів

a <- -2
b <- 3

D <- a + b
D

var_1 <- 2
var_2 <- -5

var_3 <- var_1 + var_2

var_4 = var_1 + var_2

# Не варто використовувати знак рівності `=` для призначення об'єктів
# Цей знак потрібно використовувати виключно під час специфікації функцій

# виключення у назвах об'єктів
# 1_var <- 1  # назва не має починатися із цифри
# привіт <- 1 # не варто використовувати кирилицю у назві об'єктів
```

```r
# краще уникати перевизначення системних об'єктів:
c <- 1                          # тому, що c() - це системна функція і
# в подальшому можуть виникнути помилки

# типи об'єктів

na_value <- NA # невизначене число
integer_number <- 3L            # ціле число
double_number <- 0.2            # дробове число
text_string <- "abc - are the first three letters of the alphabet"
text_string_2 <-
  'abc - are the first three letters of the alphabet'  # рядок
logical_value <- TRUE           # логічне значення
logical_value_1 <- T           # логічне значення
logical_value_2 <- FALSE           # логічне значення
logical_value_3 <- F           # логічне значення

vector_object_of_numbers <-
  c(1, 2, 3, 4, 5, 6)           # простий вектор складений із чисел

vector_object_of_strings <-
  c("a", "b", "c", "d")         # простий вектор складений із текстових рядків

vector_of_numbers_named <-
  c(
    "a" = 1,
    "b" = 2,
    "c" = 3,
    "d" = 4
  )           # простий вектор із поіменованими елементами

vector_object_of_logical_values <-
  c(TRUE, FALSE, TRUE, FALSE) # простий вектор складений із логічних значень

matrix_object_of_numbers <-
  matrix(c(1, 2, 3, 4),
         nrow = 2,
         ncol = 2)              # матриця складена із чисел

matrix_object_of_other_values <-
  matrix(vector_object_of_strings,
         nrow = 2,
         ncol = 2)              # матриця складена із текстових рядків

list_of_objects <-
  list(vector_object_of_numbers,
       matrix_object_of_numbers,
       integer_number)          # список складений із різних об'єктів

# Перевірка типу об'єкта

? class
help(class)

class(matrix_object_of_other_values)
class(logical_value)

? str
str(double_number)
str(vector_object_of_numbers)
```

```r
str(vector_of_numbers_named)

str(list_of_objects)

str(vector_of_numbers_named)

? attr
attr(vector_of_numbers_named, "names")

# Чому краще не змішувати об'єкти в одному векторі
vector_object_of_mixed_values <-
  c(vector_object_of_numbers,
    vector_object_of_strings,
    integer_number,
    logical_value)

# порівняння об'єктів (величин)

a > b
a < b
a >= D
a <= D
integer_number == double_number
integer_number != double_number

vector_object_of_numbers == integer_number

vector_object_of_numbers[vector_object_of_numbers != integer_number]
vector_object_of_numbers[c(T, T, F, T, T, T)]

# Особливості роботи із об'єктами, які мають значення NA та пошук таких значень
abc <- c(1, 2, 3, NA, NA, 6)
DEF <- c("a", "b", "some text", NA, NA, "NA")

abc == 1
DEF == "b"

abc == NA     # Некоректно використовувати тому, що порівняння не відбувається
DEF == NA

? is.na
is.na(abc)    # Замість попереднього, варто використовувати це.
is.na(DEF)

# Як перевірити чи є у векторі хоча б одне значення NA.
? any
any(is.na(abc))
any(is.na(DEF))

# 4.4 базові арифметичні функції в R (sum(), mean(), sqrt(), log()) -------

1 +
  2 -
  5 * 3

# сумування
? sum
```

```r
sum(1, 2)

vect_to_sum <- c(1, 3, 4, 5, 7, 10, 0.5)
sum(vect_to_sum)

sum(c(4, 5, 6, NA))
sum(c(4, 5, 6, NA), na.rm = TRUE)

# середнє
? mean
mean(vector_object_of_numbers)

mean(c(vector_object_of_numbers, NA))
mean(c(vector_object_of_numbers, NA), na.rm = TRUE)
mean(c(vector_object_of_numbers, 0))

# Квадратний корінь
? sqrt

sqrt(4)
sqrt(vector_object_of_numbers)
sqrt(NA)

# Число Ейлера
? exp
exp(1)

# натуральний логарифм
? log

log(exp(1))
log(exp(2))
log(100)

log(vector_object_of_numbers)

# Розрахуємо точки параболи

x_axis <- seq(from = -2, to = 3, by = 0.1)

x_axis <- seq(-2, 3, 0.1)

y <- x_axis ^ 2 - 2 * x_axis - 5
plot(x_axis, y)


y_2 <- -3 * x_axis ^ 2 + 2 * x_axis - 5
plot(x_axis, y_2)

# 4.5 типові помилки, підчас роботи із R та як їх розуміти; ---------------

# помилка: об'єкт не існує - найчастіша.
boom <- 1
boo
boom

# помилка: функція не існує - найчастіша.
sul(1, 2, 3)
sum()
```

```r
# Помилка: функція була визначена кириличною літерою
с(-1, 2) # іваф
c(-1, 2)

# помилка:  Операції із несумісними об'єктами
integer_number + text_string

# попередження:  Математичні розрахунки, що не мають розв'язання
log(-1)

log(c(-1, 2))
log(c(c(-1, 2), c(3, exp(1))))


###### 4. Основи роботи із R та основи синтаксису. ######

# 4.1 Основи синтаксису ------------------------------------

# Змінні та як їх назвати;

wheatUkrData
wheatUkr <- 2
wheat_ukr <- 5
ukraine.wheat
ukr.123.wheat

ukr.wheat <- 1

# Пробіли та розбірливість коду;

ukr 123 wheat
ukr.wheat
123.ukr.wheat

# ukr.wheat.007 <- 100

# Довжина одного рядка коду;

# Скрипт та його структура;

# 4.2 об'єкти в R та призначення об'єктів; -------------------------------

# створення (призначення) об'єктів

a <- 1
a
b <- 2
b
D <- a + b
D

# перевизначення об'єктів

a <- -2
b <- 3

D <- a + b
```

```
D

var_1 <- 2
var_2 <- -5

var_3 <- var_1 + var_2

var_4 = var_1 + var_2

# Не варто використовувати знак рівності `=` для призначення об'єктів
# Цей знак потрібно використовувати виключно під час специфікації функцій

# виключення у назвах об'єктів
# 1_var <- 1  # назва не має починатися із цифри
# привіт <- 1 # не варто використовувати кирилицю у назві об'єктів

# краще уникати перевизначення системних об'єктів:
c <- 1                          # тому, що c() - це системна функція і
                                # в подальшому можуть виникнути помилки

# типи об'єктів

na_value <- NA # невизначенне число
integer_number <- 3L            # ціле число
double_number <- 0.2            # дробове число
text_string <- "abc - are the first three letters of the alphabet"
text_string_2 <-
 'abc - are the first three letters of the alphabet'  # рядок
logical_value <- TRUE           # логічне значення
logical_value_1 <- T           # логічне значення
logical_value_2 <- FALSE           # логічне значення
logical_value_3 <- F           # логічне значення

vector_object_of_numbers <-
  c(1, 2, 3, 4, 5, 6)           # простий вектор складений із чисел

vector_object_of_strings <-
  c("a", "b", "c", "d")         # простий вектор складений із текстових рядків

vector_of_numbers_named <-
  c(
"a" = 1,
 "b" = 2,
    "c" = 3,
 "d" = 4
)          # простий вектор із поіменованими елементами

vector_object_of_logical_values <-
  c(TRUE, FALSE, TRUE, FALSE) # простий вектор складений із логічних значень

matrix_object_of_numbers <-
  matrix(c(1, 2, 3, 4),
         nrow = 2,
         ncol = 2)              # матриця складена із чисел

matrix_object_of_other_values <-
  matrix(vector_object_of_strings,
         nrow = 2,
         ncol = 2)              # матриця складена із текстових рядків
```

```r
list_of_objects <-
  list(vector_object_of_numbers,
       matrix_object_of_numbers,
       integer_number)        # список складений із різних об'єктів

# Перевірка типу об'єкта

? class
help(class)

class(matrix_object_of_other_values)
class(logical_value)

? str
str(double_number)
str(vector_object_of_numbers)

str(vector_of_numbers_named)

str(list_of_objects)

str(vector_of_numbers_named)

? attr
attr(vector_of_numbers_named, "names")

# Чому краще не змішувати об'єкти в одному векторі
vector_object_of_mixed_values <-
  c(vector_object_of_numbers,
    vector_object_of_strings,
    integer_number,
    logical_value)

# порівняння об'єктів (величин)

a > b
a < b
a >= D
a <= D
integer_number == double_number
integer_number != double_number

vector_object_of_numbers == integer_number

vector_object_of_numbers[vector_object_of_numbers != integer_number]
vector_object_of_numbers[c(T, T, F, T, T, T)]

# Особливості роботи із об'єктами, які мають значення NA та пошук таких значень
abc <- c(1, 2, 3, NA, NA, 6)
DEF <- c("a", "b", "some text", NA, NA, "NA")

abc == 1
DEF == "b"

abc == NA     # Некоректно використовувати тому, що порівняння не відбувається
DEF == NA

? is.na
```

```r
is.na(abc)    # Замість попереднього, варто використовувати це.
is.na(DEF)

# Як перевірити чи є у векторі хоча б одне значення NA.
? any
any(is.na(abc))
any(is.na(DEF))

# 4.4 базові арифметичні функції в R (sum(), mean(), sqrt(), log()) -------

1 +
  2 -
  5 * 3

# сумування
? sum

sum(1, 2)

vect_to_sum <- c(1, 3, 4, 5, 7, 10, 0.5)
sum(vect_to_sum)

sum(c(4, 5, 6, NA))
sum(c(4, 5, 6, NA), na.rm = TRUE)

# середнє
? mean
mean(vector_object_of_numbers)

mean(c(vector_object_of_numbers, NA))
mean(c(vector_object_of_numbers, NA), na.rm = TRUE)
mean(c(vector_object_of_numbers, 0))

# Квадратний корінь
? sqrt

sqrt(4)
sqrt(vector_object_of_numbers)
sqrt(NA)

# Число Ейлера
? exp
exp(1)

# натуральний логарифм
? log

log(exp(1))
log(exp(2))
log(100)

log(vector_object_of_numbers)

# Розрахуємо точки параболи

x_axis <- seq(from = -2, to = 3, by = 0.1)

x_axis <- seq(-2, 3, 0.1)
```

```r
y <- x_axis ^ 2 - 2 * x_axis - 5
plot(x_axis, y)


y_2 <- -3 * x_axis ^ 2 + 2 * x_axis - 5
plot(x_axis, y_2)

# 4.5 типові помилки, підчас роботи із R та як їх розуміти; ---------------

# помилка: об'єкт не існує - найчастіша.
boom <- 1
boo
boom

# помилка: функція не існує - найчастіша.
sul(1, 2, 3)
sum()


# Помилка: функція була визначена кириличною літерою
c(-1, 2) # іваф
c(-1, 2)

# помилка:  Операції із несумісними об'єктами
integer_number + text_string

# попередження:  Математичні розрахунки, що не мають розв'язання
log(-1)

log(c(-1, 2))
log(c(c(-1, 2), c(3, exp(1))))

# 4.6 пакети в R; ----------------------------------------------------------

# для чого потрібні та як ми їх використовуємо;

# встановлення
? install.packages
install.packages()

# завантаження пакету у робоче середовище
? library
library()

# як встановити відсутній пакет
install.packages("dplyr")

# Після встановлення обов'язково завантажуйте пакет для використання.
library(dplyr)

# помилки, що виникають, коли пакет не завантажено.
library(tidyverse)

install.packages("tidyverse")

# 4.6 пакети в R; ----------------------------------------------------------

# для чого потрібні та як ми їх використовуємо;
```

```r
# встановлення
? install.packages
install.packages()

# завантаження пакету у робоче середовище
? library
library()

# як встановити відсутній пакет
install.packages("dplyr")

# Після встановлення обов'язково завантажуйте пакет для використання.
library(dplyr)

# помилки, що виникають, коли пакет не завантажено.
library(tidyverse)

install.packages("tidyverse")
```

Додаток 5. Основи завантаження, маніпуляції та збереження даних за допомогою R

```r
# 5.0 Встановлення коректної робочої директорії та завантаження пакетів----------
---

# Перевірка, чи є сесія R Studio частиною проекту.
# Вихід або зміна проекту, якщо це необхідно.
# Перевірка поточної директорії (де саме R буде шукати файли)
getwd()

# Зміна робочої директорії в разі необхідності
# setwd()

# Завантаження (встановлення) пакетів
# library(readr)
# library(readxl)
# library(dplyr)
library(tidyverse)

# 5.1 Основи роботи із об'єктом `data.frame` та/або `data_frame` -----------------

# Завантажимо дані (приклад, який ми детальніше розглянемо, далі)
exampleData <- read.delim("data/USDA_wheat_yields_data.txt")

# Перегляд структури даних class(), str(), head(), tail(), View()

class(exampleData)  # Перевірити тип об'єкту
str(exampleData)    # Перевірити структуру об'єкту

head(exampleData)   # Переглянути перші шість спостережень у таблиці
tail(exampleData)   # Переглянути останні спостереження у таблиці

View(exampleData)   # Переглянути усю таблицю

# Приклад: Чому звичайний об'єкт data.frame є незручним для роботи із даними.
exampleData         # друк таблиці у консолі є проблемним

# Пакет `tibble` для роботи із об'єктом `tbl_df` (не `data.frame`)
# install.packages("tibble")
library(tibble)

# Перетворення звичайного об'єкту `data.frame` на `tbl_df`.
exmUpgrade <- as_tibble(exampleData)

# exmUpgrade <- tibble::as_tibble(exampleData)
# exmUpgrade <- dplyr::as_tibble(exampleData)

exmUpgrade          # друк таблиці у консолі є значно зручнішим
                    # немає необхідності використовувати функції
                    # head(), tail(), str(), View()

glimpse(exmUpgrade) # Зручний перегляд структури таблиці

# Додаткові матеріали:
# https://github.com/rstudio/cheatsheets/raw/master/data-import.pdf
# http://tibble.tidyverse.org/

# 5.2 Завантаження/відкриття табличних даних з використанням базового функціоналу R --------
```

```r
# "data/USDA_wheat_yields_data.txt"
# read.delim()
fileToOpen_1 <- "./data/USDA_wheat_yields_data.txt"
usdaWheatYields <- read.delim(file = fileToOpen_1)

class(usdaWheatYields)
str(usdaWheatYields)

# Типові помилки
# Файл не знайдений
usdaWheatYields <-
  read.delim(file = "./data/USDA_wheat_yilds_data.txt")

usdaWheatYields <-
  read.delim(file = "./data/USDA_wheat_yields_data")

usdaWheatYields <-
  read.delim(file = "./data/USDA_wheat_yields data")

usdaWheatYields <-
  read.delim(file = "./dat/USDA_wheat_yields_data.txt")

# Вирішення проблеми:
#   Перевведення назви файлу:
usdaWheatYields <-
  read.delim(file = "./data/USDA_wheat_yields_data.txt")

# "data/corn_trial_example.csv"
# read.csv()
cornTrialData <-
  read.csv(file = "./data/corn_trial_example.csv")

class(cornTrialData)
str(cornTrialData)
head(cornTrialData)

# Типова під час неправильного вказування параметру.
# випадок параметру row names
cornTrialData <-
  read.csv(file = "./data/corn_trial_example.csv", row.names = TRUE)

# "data/USDA_corn_yields_data.csv"
# read.csv2()
usdaCornYields <-
  read.csv2(file = "./data/USDA_corn_yields_data.csv")

class(usdaCornYields)
str(usdaCornYields)
head(usdaCornYields)
View(usdaCornYields)

# 5.3 Те саме, що і в 5.2, але із більш зручним пакетом 'readr' ----------------

library(readr)
# install.packages("readr")

# read_delim()
usdaWheat <-
```

```r
  read_delim(file = "./data/USDA_wheat_yields_data.txt",
             delim = "\t")

# read_csv()
cornTrial <-
  read_csv(file = "./data/corn_trial_example.csv")

cornTrial_2 <-
  read_delim(file = "./data/corn_trial_example.csv",
             delim = ",")

# read_csv2()
usdaCorn <-
  read_csv2(file = "./data/USDA_corn_yields_data.csv")

# Приклад складного файлу.
usdaCorn_2 <-
  read_delim(file = "./data/USDA_corn_yields_data.csv",
             delim = ";")

# Порівняння фреймів із даними

# Порівняти два ідентичних фрейми, завантажених із різними функціями в R.

# Хибний підхід
cornTrialData == cornTrial

# Правильний підхід
all.equal(cornTrial, cornTrialData)
all.equal(usdaWheat, usdaWheatYields)
# Для того, щоб зрозуміти його результати, необхідно уважно прочитати текст, який
# з'явиться у консолі. Після порівняння цих об'єктів бачимо, що
# дані у об'єктах однакові, відмінності лише полягають у тому, що додаткові
# параметри об'єктів не співпадають. Це не впливає на нашу роботу із даним.

# Порівняння заздалегідь різних об'єктів дає нам відповідний результат.
all.equal(usdaWheat, cornTrial)

# Додаткові матеріали про зручний імпорт даних в R
# https://github.com/rstudio/cheatsheets/raw/master/data-import.pdf
# http://readr.tidyverse.org/

# 5.4 Відкриття табличних даних у форматі Excel (readxl) -----------------

# install.packages("readxl")
library(readxl)
# excel_sheets()
# read_excel()

# Відкриття файлів Excel із незручною структурою.
# Загальні приклади документів із Держкомстату

# "data/roslinnitstvo.xls"
filePath <- "./data/roslinnitstvo.xls"

excel_sheets(filePath)

read_excel(filePath)
read_excel(filePath, sheet = 1)
```

```r
read_excel(filePath, sheet = 1, skip = 4)
roslData <- read_excel(filePath, sheet = 1, skip = 4, n_max = 27)

# приклад типової помилки
View(read_excel(filePath, sheet = 1, skip = 5, n_max = 25))

# "data/prod_prazi.xlsx"
filePath_2 <- "data/prod_prazi.xlsx"

excel_sheets(filePath_2)

read_excel(filePath_2)
read_excel(filePath_2, sheet = 1, skip = 3)
View(read_excel(filePath_2, sheet = 1, skip = 3))

prodPrazi <- read_excel(filePath_2, sheet = 1, skip = 3, n_max = 27)

prodPraziUpg <- read_excel("./data/prod_prazi_eng.xlsx")

# 5.5 Перейменування назв колонок в табличних даних -----------------

names(prodPrazi)

names(prodPrazi)[1]
names(prodPrazi)[5]

names(prodPrazi)[1] <- "rik"
names(prodPrazi)[1]

names(prodPrazi)[c(2,3,4)] <- c("prod_prazi_sg_virob",
                                "prod_prazi_rost",
                                "prod_prazi_tvar")
prodPrazi

# Приклад даних по рослинництву
names(roslData)
names(roslData)[1]
names(roslData)[2]
names(roslData)[c(3,4,5)]

names(roslData)[1] <- "year"
names(roslData)[2] <- "zern_zernobob"
names(roslData)[c(3,4,5)] <- c("tsukr_bur", "soniashnik", "kartoplia")
names(roslData)[6] <- "ovochi"
names(roslData)[7] <- "kormi"

# Приклад даних по продуктивності праці
names(prodPrazi) <-
  c("year", "cg_virobn", "rosl", "tvar",
    "cg_virobn_2", "rosl_2", "tvar_2")

# 5.6 Змінна/колонка таблиці/`data_frame` як вектор. -----------------

# Оператор $
prodPraziUpg$rik

length(prodPraziUpg$rik)

mean(prodPraziUpg$rik)
```

```r
sum(prodPraziUpg$rik)
log(prodPraziUpg$rik)

zminna_1 <-  prodPraziUpg$rik

# Приклад створення нової колонки
prodPraziUpg$rik2 <- log(prodPraziUpg$rik)

# 5.7 Виправлення типів змінних у табличних даних -------------------

class(prodPraziUpg$prod_prazi_na_1_tvar)

# as.numeric()
as.numeric(prodPraziUpg$prod_prazi_na_1_tvar)

prodPraziUpg$prod_prazi_na_1_tvar <-
  as.numeric(prodPraziUpg$prod_prazi_na_1_tvar)


prodPraziUpg$prod_prazi_na_1_rosl <-
  as.double(prodPraziUpg$prod_prazi_na_1_rosl)


prodPraziUpg$prod_prazi_do_pop_roku_sg_virob <-
  as.double(prodPraziUpg$prod_prazi_do_pop_roku_sg_virob)

prodPraziUpg$prod_prazi_do_pop_roku_rosl <-
  as.double(prodPraziUpg$prod_prazi_do_pop_roku_rosl)

prodPraziUpg$prod_prazi_do_pop_roku_tvar <-
  as.double(prodPraziUpg$prod_prazi_do_pop_roku_tvar)

# Перевірка правильності перетворення
str(prodPraziUpg)

as.numeric(zminna_1)
as.numeric(prodPrazi$tvar)

prodPrazi$tvar <- as.numeric(prodPrazi$tvar)

# as.character()
# as.factor()

# as.integer()
prodPraziUpg$rik <- as.integer(prodPraziUpg$rik)

# 5.8 Базові операції із вектором простими методами -----------------

# sum()
# mean()
mean(prodPraziUpg$prod_prazi_na_1_rosl)

any(is.na(prodPraziUpg$prod_prazi_na_1_rosl))

mean(prodPraziUpg$prod_prazi_na_1_rosl, na.rm = TRUE)

# length()
length(prodPraziUpg$prod_prazi_na_1_rosl)
```

```r
# 5.9 Збереження таблиць ----------------------------------------------

# write.csv()
prodPraziUpg
write.csv(x = prodPraziUpg,
          file = "./output/prod_prazi_clean.csv")
write.csv(x = prodPraziUpg,
          file = "./output/prod_prazi_clean.csv",
          row.names = FALSE)

# write_csv()
library(readr)
write_csv(x = prodPraziUpg,
          path = "./output/prod_prazi_clean_other_function.csv")

# write_excel_csv()
write_excel_csv(x = prodPraziUpg,
                path = "./output/prod_prazi_clean_other_function_2.csv")

# 'R Cheatsheet' із оглядом основних функцій для маніпуляції даних
# https://www.rstudio.com/resources/cheatsheets/
```

Додаток 6. Базові операції із табличними даними із використанням зручнішого пакету (пакет dplyr)

```
# 6.0 Підготовка середовища -----------------

# Завантаження пакетів
# install.packages("tidyverse")        # Встановлення пакетів за необхідності
library(tidyverse)

# Додаткова інформація про наступні функції http://dplyr.tidyverse.org/
# Основні команди:
#   arrange()
#   n()
#   filter()
#   select()
#   mutate()
#   summarise()
#   group_by()

# Пакет для завантаження даних
library(readxl)

# Завантаження великого файлу ексель
excel_sheets("./data/data-livestock.xlsx")
ls_full <- read_excel(path = "./data/data-livestock.xlsx",
                      sheet = "data livestock")
glimpse(ls_full)

# Завантаження меншого файлу ексель
excel_sheets("./data/wht-brl-crn-rye_data.xlsx")
grain_full <- read_excel(path = "./data/wht-brl-crn-rye_data.xlsx",
                      sheet = "Data")
grain_full
glimpse(grain_full)

# 6.1 Перейменування деяких змінних  ----------------

# Перейменуємо змінні "X__1" і "X__2"
names(grain_full)[1] <- "mark_year"
names(grain_full)[1]
names(grain_full)[names(grain_full) == "X__2"] <- "year"
names(grain_full)[2]
names(grain_full)

# 6.2 Вибір/виокремлення змінних/колонок із однієї таблиці  -----------------------------

# dplyr::select()
# Специфікація необхідних змінних
grain_subset <-
  select(grain_full,
         year, wht_prd_1000t, wht_yield_100kgha, wht_arhrvstd_1000ha,
         `wht_prcrl_prd_uah/100kg`, gdp_rl_UAH, `exrt_uah/usd`)

# Зміна порядку змінних у таблиці
select(grain_subset,
       `wht_prcrl_prd_uah/100kg`,
       year, wht_prd_1000t, wht_yield_100kgha, wht_arhrvstd_1000ha)

# Специфікація непотрібних змінних
select(grain_subset,
```

```r
       - `exrt_uah/usd`,
       - gdp_rl_UAH)

# Із перевизначенням змінної
grain_subset <-
  select(grain_subset,
         - `exrt_uah/usd`,
         - gdp_rl_UAH)

# Очищення великої таблиці від зайвих змінних
ls_full
glimpse(ls_full)

# Вибірка обмеженої кількості змінних із однієї таблиці
ls_clean <-
  select(ls_full,
         - X__1,
         - oblast__1,
         - region__1)

# Типові помилки
select(ls_clean, X__1)
select(ls_clean, - X__1)

# 6.3 Перегляд змісту змінних ------------------

# dplyr::distinct()
distinct(ls_clean, year)
distinct(ls_clean, year, region)
distinct(ls_clean, ID)
distinct(grain_subset, year)

# 6.4 Фільтрація таблиць по одній або декількох змінних  -----------------------------

# dplyr::filter()
ls_clean_short <-
  select(ls_clean,
         ID, year, oblast, region, ctl_lvstck, pg_lvstck,
         shp_lvstck, pltr_lvstck, mlk_lvstck)

# Фільтр із 1 аргументом
filter(ls_clean_short, year == 2010)
filter(ls_clean_short, oblast == "Odesa")

# Фільтр із 2 і більше аргументами
filter(ls_clean_short, year == 2009, oblast == "Odesa")

temp_df <- filter(ls_clean_short, year == 2009)
filter(temp_df, oblast == "Odesa")

filter(ls_clean_short, year == 2009 & oblast == "Odesa")

# Фільтр із знаками порівняння
filter(ls_clean_short, year >= 2010)
filter(ls_clean_short, year >= 2010, ID > 2000, ID < 3000)

# Фільтрація 'NA' значень
filter(ls_clean_short, is.na(shp_lvstck))
filter(ls_clean_short, !is.na(shp_lvstck))
```

```r
filter(ls_clean_short, !is.na(shp_lvstck), !is.na(pltr_lvstck))
filter(ls_clean_short, !is.na(shp_lvstck), is.na(pltr_lvstck))

# 6.5 Сортування таблиць ----------------------------

# dplyr::arrange()
arrange(ls_clean_short, ID)
arrange(ls_clean_short, desc(ID))
arrange(ls_clean_short, oblast, desc(ID))
arrange(ls_clean_short, oblast, desc(ID), desc(year))

# 6.6 Створення нових змінних  ----------------------------

# як результат арифметичних операцій
grain_subset

# df$newVar <- df$newVar1 + df$newVar2
grain_subset$wht_prd_2 <-
  grain_subset$wht_yield_100kgha *
  grain_subset$wht_arhrvstd_1000ha / 10

grain_subset$wht_prd_2 <-
  grain_subset$wht_yield_100kgha *
  grain_subset$wht_arhrvstd_1000ha / 10 -
  grain_subset$wht_prd_1000t

# dplyr::mutate()
mutate(grain_subset,
       wht_prd_3 = wht_yield_100kgha * wht_arhrvstd_1000ha / 10 - wht_prd_1000t)

mutate(grain_subset,
       wht_yield_lag_1 = lag(wht_yield_100kgha))

mutate(grain_subset,
       average_wht_prd = mean(wht_prd_1000t),
       sum_prod_1000t = sum(wht_prd_1000t))

# 6.7 Редагування змісту змінних  ----------------------------

# Боротьба із 'NA' значеннями у змінних
test <- filter(ls_clean_short, ID == 1345)

mutate(test,
       totl_animal = ctl_lvstck + pg_lvstck +
         shp_lvstck + pltr_lvstck + mlk_lvstck)

# Фільтрація (dplyr::filter())

# Умовне перевизначення 'NA' значень (ifelse())
mutate(test,
       ctl_lvstck = ifelse(is.na(ctl_lvstck), 0, ctl_lvstck),
       shp_lvstck = ifelse(is.na(shp_lvstck), 0, shp_lvstck),
       pltr_lvstck = ifelse(is.na(pltr_lvstck), 0, pltr_lvstck),
       mlk_lvstck = ifelse(is.na(mlk_lvstck), 0, mlk_lvstck),
       pg_lvstck = ifelse(is.na(pg_lvstck), 0, pg_lvstck),
       totl_animal = ctl_lvstck + pg_lvstck +
         shp_lvstck + pltr_lvstck + mlk_lvstck)

# tidyr::replace_na()
```

```r
library(tidyr)

test_2 <-
  replace_na(test,
             list(ctl_lvstck = 0,
                  pg_lvstck = 0,
                  shp_lvstck = 0,
                  pltr_lvstck = 0,
                  mlk_lvstck = 0))
mutate(test_2,
       totl_animal = ctl_lvstck + pg_lvstck +
         shp_lvstck + pltr_lvstck + mlk_lvstck)

# Альтернативний запис:
test %>%
  replace_na(list(ctl_lvstck = 0,
                  pg_lvstck = 0,
                  shp_lvstck = 0,
                  pltr_lvstck = 0,
                  mlk_lvstck = 0)) %>%
  mutate(totl_animal = ctl_lvstck + pg_lvstck +
           shp_lvstck + pltr_lvstck + mlk_lvstck)

# 6.8 Підбиття підсумків по колонкам/змінним ----------------------------

# dplyr::summarise()
ls_test <- filter(ls_clean_short,
                  oblast %in% c("Kharkiv", "Odesa", "Lviv"))

ls_test

# Базове резюме таблиці
summarise(ls_test,
          ctl_lvstck = sum(ctl_lvstck, na.rm = TRUE))

# dplyr::group_by()
group_by(ls_test, year, oblast)
summarise(group_by(ls_test, year, oblast),
          ctl_lvstck = sum(ctl_lvstck, na.rm = TRUE))


mutate(group_by(ls_test, year, oblast),
          ctl_lvstck_1 = sum(ctl_lvstck, na.rm = TRUE),
       share = ctl_lvstck / ctl_lvstck_1 * 100) %>%
  arrange(desc(share)) %>%
  filter(oblast == "Lviv", year == 2011)
```

Додаток 7. Описова статистика (Descriptive statistics) та базовий графічний аналіз

```r
# 7.0 завантаження тренувальних даних та підготовка робочого середовища -------------------

library(tidyverse)
library(readxl)

###### завантаження першої таблиці
ag_full <- read_excel("data/wht-brl-crn-rye_data.xlsx")
names(ag_full)[2] <- "year"

# Очистка даних та формування вибірки
```

```r
ag_clean <- select(ag_full,
                   year,
                   wht_yield_100kgha,
                   `wht_prcrl_prd_uah/100kg`,
                   wht_arhrvstd_1000ha)

# Те саме за допомогою оператора %>%
ag_clean <-
  ag_full %>%
  rename(year = year) %>%
  select(year, wht_yield_100kgha,
         `wht_prcrl_prd_uah/100kg`, wht_arhrvstd_1000ha)

####### Завантаження другої таблиці та її резюмування
# Із попередніх відео ми знаємо, що ця таблиця містить данні по кожному
# окремому підприємству. Нам необхідно мати дані згруповані по областях, а не по
# окремих підприємствах. Тому ми проводимо наступні операції по групуванню даних.
ls_full <- read_excel("data/data-livestock.xlsx", sheet = 1)

# Очистка даних та формування вибірки
ls_clean <- group_by(ls_full, year, oblast, region)
ls_clean <- summarise(ls_clean,
                      egg_lvstck = sum(egg_lvstck, na.rm = TRUE),
                      egg_prd = sum(egg_prd, na.rm = TRUE),
                      pg_lvstck = sum(pg_lvstck, na.rm = TRUE),
                      pg_prd = sum(pg_prd, na.rm = TRUE))
ls_clean <- ungroup(ls_clean)
ls_clean <- mutate(ls_clean,
                   egg_per_lvstk = egg_prd / egg_lvstck,
                   pg_per_lvstk = pg_prd / pg_lvstck)

# Те саме за допомогою оператора %>%
ls_clean <-
  ls_full %>%
  group_by(year, oblast, region) %>%
  summarise(egg_lvstck = sum(egg_lvstck, na.rm = TRUE),
            egg_prd = sum(egg_prd, na.rm = TRUE),
            pg_lvstck = sum(pg_lvstck, na.rm = TRUE),
            pg_prd = sum(pg_prd, na.rm = TRUE)) %>%
  ungroup() %>%
  mutate(egg_per_lvstk = egg_prd / egg_lvstck,
         pg_per_lvstk = pg_prd / pg_lvstck)

# 7.1 Основна описова статистика окремої змінної ------------------------

ag_clean
ag_clean$wht_yield_100kgha

mean(ag_clean$wht_yield_100kgha)
min(ag_clean$wht_yield_100kgha)
max(ag_clean$wht_yield_100kgha)
sd(ag_clean$wht_yield_100kgha)

psych::kurtosi(ag_clean$wht_yield_100kgha)
psych::skew(ag_clean$wht_yield_100kgha)

median(ag_clean$wht_yield_100kgha)

# summary()
summary(ag_clean)
```

```r
# psych::describe()
library(psych)
describe(ag_clean)

summary(ls_clean)
describe(ls_clean)

# 7.2 Scatter Plot -------------------------------

# plot()
plot(x = rnorm(10000, 25, 5),
     y = rnorm(10000, 10, 2 ),
     main = "Центральна назва",
     sub = "Підзаголовок",
     xlab = "Вісь X",
     ylab = "Вісь Y")

# plot(x = ..., y = ...)
plot(ag_clean$year, ag_clean$wht_yield_100kgha)

plot(ag_clean$`wht_prcrl_prd_uah/100kg`, ag_clean$wht_yield_100kgha)

plot(ag_clean$year, ag_clean$`wht_prcrl_prd_uah/100kg`, type = "b")

plot(ag_clean$wht_arhrvstd_1000ha, ag_clean$wht_yield_100kgha)

plot(ag_clean$wht_arhrvstd_1000ha,
     ag_clean$wht_yield_100kgha,
     main = "Назва",
     xlab = "Вісь X",
     ylab = "Вісь Y")


plot(ls_clean$egg_lvstck, ls_clean$egg_prd)

plot(log(ls_clean$egg_lvstck), log(ls_clean$egg_prd))

plot(ls_clean$egg_lvstck, ls_clean$pg_lvstck)


# 7.3 Box-plot ------------------

# boxplot()

boxplot(ls_clean$egg_lvstck)
boxplot(ls_clean$pg_lvstck)
boxplot(ag_clean$wht_yield_100kgha)

boxplot(egg_per_lvstk ~ oblast, data = ls_clean)
boxplot(egg_per_lvstk ~ year, data = ls_clean)

boxplot(pg_per_lvstk ~ oblast, data = ls_clean)
boxplot(pg_per_lvstk ~ year, data = ls_clean)

# 7.4 Distribution histogram ma Density Plot -------------------

# hist()
```

```r
hist(ls_clean$pg_lvstck)
hist(log(ls_clean$pg_lvstck))

hist(ls_clean$egg_prd)
hist(log(ls_clean$egg_prd))

# density(),
plot(density(ls_clean$egg_prd))

# CDF
plot.ecdf(ls_clean$pg_prd)

a <- ecdf(ls_clean$pg_prd)
str(a)

plot.ecdf(ls_clean$egg_prd)
plot.ecdf(ls_clean$pg_per_lvstk)

# 7.5 Кореляція між змінними та статистична значимість кореляції -------------

# cor()
ag_clean
cor(x = ag_clean$wht_yield_100kgha, y = ag_clean$year)
cor(x = ag_clean$wht_yield_100kgha, y = ag_clean$wht_arhrvstd_1000ha)
cor(x = ag_clean$wht_yield_100kgha, y = ag_clean$`wht_prcrl_prd_uah/100kg`)

# cor.test()
cor.test(x = ag_clean$wht_yield_100kgha, y = ag_clean$year)
cor.test(x = ag_clean$wht_yield_100kgha, y = ag_clean$wht_arhrvstd_1000ha)
cor.test(x = ag_clean$wht_yield_100kgha, y = ag_clean$`wht_prcrl_prd_uah/100kg`)

ag_clean_2 <-
  mutate(ag_clean,
         price = lag(`wht_prcrl_prd_uah/100kg`),
         ah = lag(wht_arhrvstd_1000ha))

cor.test(x = ag_clean_2$wht_yield_100kgha, y = ag_clean_2$price)
cor.test(x = ag_clean_2$wht_yield_100kgha, y = ag_clean_2$ah)

# Autocorrelation
b <- acf(ag_clean$wht_arhrvstd_1000ha)
str(b)
b$acf

ccfResults <- ccf(ag_clean_2$wht_yield_100kgha,
                  ag_clean_2$price,
                  lag.max = 5,
                  na.action = na.pass)


ccfResults <- ccf(ag_clean$wht_yield_100kgha,
                  ag_clean$`wht_prcrl_prd_uah/100kg`,
                  lag.max = 5,
                  na.action = na.pass)

str(ccfResults)
ccfResults$acf
ccfResults$lag
```

```r
get_values(ccfResults)

# функція для витягування коефіцієнтів кореляції та
# розрахунку значимостей коефіцієнтів
get_values <- function(ccf_results) {
  lags <- as.vector(ccf_results$lag)
  corValues <- as.vector(ccf_results$acf)
  tVector <-
    corValues * sqrt(ccf_results$n.used - 2) / sqrt(1 - corValues ^ 2)
  pValue <- pt(-abs(tVector), df = ccf_results$n.used - 2) * 2
  tibble(lag = lags,
         cor = corValues,
         `p-value` = pValue)
}
```

Додаток 8. Звичайна лінійна регресія в R

```
##### Постановка задачі релевантної для моделі AGMEMOD #########

# Змоделюємо врожайність пшениці (змінна "wht_yield_100kgha") виражену у центнерах
# на один гектар посівних площ, як функцію:
#   * Реальної ціни на пшеницю у попередньому періоді (змінна "wht_prcrl_prd_uah/100kg")
#   * Очікуваних посівних площ пшениці у поточному періоді (змінна "wht_arhrvstd_1000ha")
#   * Лінійного тренду
#   * Додаткові змінні
# на основі даних "wht-brl-crn-rye_data.xlsx".
# Спробуємо варіації такої моделі без врахування лінійного тренду або враховуючи
# минулорічні посівні площі.

# 8.0 Підготовка середовища та завантаження тренувальних даних ------------

library(tidyverse)
library(readxl)

###### завантаження першої таблиці
ag_full <- read_excel("./data/wht-brl-crn-rye_data.xlsx")

names(ag_full)[2] <- "year"

# Очистка даних та формування вибірки
ag_clean2 <- select(ag_full,
                    year,
                    wht_yield_100kgha,
                    `wht_prcrl_prd_uah/100kg`,
                    wht_arhrvstd_1000ha)

ag_clean2 <- arrange(ag_clean2, year)

ag_clean2 <- mutate(ag_clean2,
                    lin_trend = year - 2004,
                    lagged_price = lag(`wht_prcrl_prd_uah/100kg`),
                    lagged_ah = lag(wht_arhrvstd_1000ha))

# Те саме за допомогою оператора %>%
ag_clean <-
  ag_full %>%
  rename(year = year) %>%
  select(year, wht_yield_100kgha,
         `wht_prcrl_prd_uah/100kg`, wht_arhrvstd_1000ha) %>%
  mutate(lin_trend = year - 2004,
         lagged_price = lag(`wht_prcrl_prd_uah/100kg`),
         lagged_ah = lag(wht_arhrvstd_1000ha))

# Перевіримо, чи обидві таблиці однакові
all.equal(ag_clean, ag_clean2)

# Видалимо зайву таблицю
rm(ag_clean2)


# 8.1 Побудова базової лінійної регресії --------------------------

# lm()
lm()
?lm
```

```r
help(lm)

# Data generation
set.seed(100)
exampleData <-
  tibble(x1 = runif(100, 10, 25),
         x2 = rgamma(100, 2, 1),
         x3 = rnorm(100, 34, 7)) %>%
  mutate(y = 18 + 0.3 * x1 + 2.6 * x2 - 5 * x3 + 1.7 * x1 * x2 + rnorm(100, 50, 10))

# lm()
?lm

# Formula
# y ~ x1 + x2 + x3
# y ~ x1 * x2 + x3

exfit1 <- lm(formula = y ~ x1 + x2, data = exampleData)
exfit1
str(exfit1)
class(exfit1)
summary(exfit1)

exfit2 <- lm(formula = y ~ x1 + x2 + x3, data = exampleData)
summary(exfit2)

exfit3 <- lm(formula = y ~ x1 + x2 + x3 + x1:x2, data = exampleData)
summary(exfit3)

exfit4 <- lm(formula = y ~ x3 + x1:x2, data = exampleData)
summary(exfit4)

# Візуальна інстпекція
plot(exampleData)
plot(mutate(exampleData, x1_x2 = x1 * x2))

exfit5 <- lm(formula = y ~ x1 + x3 + x1:x2, data = exampleData)
summary(exfit5)

# Рівняння
# y ~ x1 + x2
# y ~ x1 * x2  це те саме, що і рівняння x1 + x2 + x1 * x2

fit <- lm(wht_yield_100kgha ~ lagged_price + wht_arhrvstd_1000ha + lin_trend,
          data = ag_clean,
          na.action = na.omit)  # Параметр в якому визначається як, що саме робити із NA
                                # na.omit - значення за замовчанням.
fit

# Статистичні модифікації із змінними можна проводити у самому рівнянні
fit2 <- lm(log(wht_yield_100kgha) ~
             log(lagged_price) +
             log(wht_arhrvstd_1000ha) +
             log(lin_trend),
          data = ag_clean)
fit2

#
class(fit)
str(fit)
```

```r
fit$coefficients
fit$residuals
fit$effects
fit$qr$qr


# 8.2 Перегляд результатів регресії -----------------------------

# summary()
summary(fit)
summary(fit2)

# Dummy variable example
ag_clean <- mutate(ag_clean, dummy_year = ifelse(year %in% c(2008, 2014), 1L, 0L))

# Регресія покращена за допомогою dummy variable
fit3 <- lm(wht_yield_100kgha ~ lagged_price + wht_arhrvstd_1000ha + lin_trend + dummy_year,
           data = ag_clean)
fit3

# Summar
summary(fit3)

# coefficients()
coef(fit3)
coefficients(fit3)

str(coefficients(fit))

coef(fit3)[1]
coef(fit3)[2]
coef(fit3)[4]

# confint()
confint(fit3)
summary(fit3)

# fitted()
fitted(fit3)

comparison <-
  ag_clean %>%
  select(wht_yield_100kgha, year) %>%
  mutate(modelled_yields = c(NA, fitted(fit3)))
comparison

plot(comparison$year, comparison$wht_yield_100kgha)
lines(comparison$year, comparison$modelled_yields)

# residuals()
residuals(fit3)

# anova()
anova(fit3)

# vcov(fit)
vcov(fit3)
```

```r
# influence(fit)
influence(fit3)


# 8.3 Побудова діагностичних графіків  ----------------------------

fit3
plot(fit3)

# Комбінована візуалізація
par(mfrow=c(2,2))
plot(fit3)
par(mfrow=c(1,1))

# par(mfrow=c(2,2)); plot(fit); par(mfrow=c(1,1)))
par(mfrow=c(2,2)); plot(fit3) ; par(mfrow=c(1,1))


######## Чому важливо йти далі і перевіряти припущення #########


# Завантажуємо данні
anscombe_data <- read_csv("data/anscombe_quartet.csv")
plot(anscombe_data)

par(mfrow=c(2,2))
plot(anscombe_data$x1, anscombe_data$y1, main = "X1",  xlim=c(3,20),  ylim=c(0,20))
plot(anscombe_data$x2, anscombe_data$y2, main = "X2",  xlim=c(3,20),  ylim=c(0,20))
plot(anscombe_data$x3, anscombe_data$y3, main = "X3",  xlim=c(3,20),  ylim=c(0,20))
plot(anscombe_data$x4, anscombe_data$y4, main = "X4",  xlim=c(3,20),  ylim=c(0,20))
par(mfrow=c(1,1))

ansFit_1 <- lm(y1~x1, data = anscombe_data)
ansFit_2 <- lm(y2~x2, data = anscombe_data)
ansFit_3 <- lm(y3~x3, data = anscombe_data)
ansFit_4 <- lm(y4~x4, data = anscombe_data)

all_regs <- list(ansFit_1, ansFit_2, ansFit_3, ansFit_4)

all_regs %>%
  map(coef) %>%
  map(round, digits = 2)

# Summary
summary(ansFit_1)
summary(ansFit_2)
summary(ansFit_3)
summary(ansFit_4)

all_regs %>%
  map(.f = anova)

par(mfrow=c(2,2))
plot(anscombe_data$x1, anscombe_data$y1, main = "X1",  xlim=c(3,20),  ylim=c(0,20))
abline(ansFit_1)
plot(anscombe_data$x2, anscombe_data$y2, main = "X2",  xlim=c(3,20),  ylim=c(0,20))
abline(ansFit_2)
plot(anscombe_data$x3, anscombe_data$y3, main = "X3",  xlim=c(3,20),  ylim=c(0,20))
abline(ansFit_3)
```

```r
plot(anscombe_data$x4, anscombe_data$y4, main = "X4",  xlim=c(3,20),  ylim=c(0,20))
abline(ansFit_4)
par(mfrow=c(1,1))

par(mfcol=c(4,4), mfrow = c(4,4), mar = rep(2, 4))
all_regs %>%
  map(.f = plot)
par(mfrow=c(1,1));

########## Основні припущення ###############

# 8.4 Лінійність та нормальність розподілу залишків ---------------------------

# Лінійність
plot(anscombe_data$x2, anscombe_data$y2, main = "X2")
abline(ansFit_2)

# QQ-plot - qqPlot();
par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit3)
par(mfrow=c(1,1));

# Histogram and box plot of residualts
hist(resid(fit3))
boxplot(resid(fit3))

# Додатково встановлюємо пакет car
# install.packages("car")
library(car)
qqPlot(fit3, main="QQ Plot")

# Приклад не нормального розподілу
qqPlot(ansFit_3)

# Приклад не нормального розподілу
qqPlot(exfit3)

# Shapiro-Wilk test of normality - shapiro.test();

# H0: Нульова гіпотезі: залишки мають нормальний розподіл
shapiro.test(residuals(fit3))
shapiro.test(resid(fit3))

shapiro.test(residuals(ansFit_3))

# На що впливає ненормальність розподілу залишків? на p-Value
# Central Limit Theorem; n > 50, n > 100, n > 200, n > 40...


# 8.5 Однорідність розподілу залишків або постійна варіація залишків -----------

par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit3)
par(mfrow=c(1,1))


par(mfcol=c(2,2), mfrow = c(2,2))
plot(exfit4)
par(mfrow=c(1,1))
```

```r
# Breusch-Pagan test
# install.packages("lmtest")
library(lmtest)

# H0: Нульова гіпотезі: Постійна варіація залишків
bptest(fit3)
bptest(exfit4)

# Non-constant variance test - ncvTest();
# H0: Нульова гіпотезі: Постійна варіація залишків
library(car)
ncvTest(fit3)
# Щоб прийняти гіпотезу 0 необхідно отримати p-value > 0.05 (рівень значущості)
ncvTest(exfit4)

# Plot: studentized residuals vs. fitted values;
spreadLevelPlot(fit3)


plot(ansFit_4)
ncvTest(ansFit_4)
bptest(ansFit_4)

# В разі, якщо ми виявили проблему із рівномірністю розподілу залишків
# то це не впливає на коефіцієнти але впливає на їх значимість.
# тому нам необхідно застосувати white's heteroscedasticity-consistent estimator

# Отримання звичайної таблиці із коефіцієнтами
summary(fit)

# Функція для корегування параметрів значимості регресії
summaryR <- function(model, type=c("hc3", "hc0", "hc1", "hc2", "hc4"), ...){

  if (!require(car)) stop("Required car package is missing.")

  type <- match.arg(type)
  V <- hccm(model, type=type)
  sumry <- summary(model)
  table <- coef(sumry)
  table[,2] <- sqrt(diag(V))
  table[,3] <- table[,1]/table[,2]
  table[,4] <- 2*pt(abs(table[,3]), df.residual(model), lower.tail=FALSE)

  sumry$coefficients <- table
  p <- nrow(table)
  hyp <- cbind(0, diag(p - 1))
  sumry$fstatistic[1] <- linearHypothesis(model, hyp,white.adjust=type)[2,"F"]
  sumry$type <- type
  return(sumry)
}

summary(ansFit_4)

# Скориговані параметри регресії
summaryR(ansFit_4)

# Порівняння із звичайними результатами
summary(fit3)
```

```r
summaryR(fit3)

# 8.6 Мультиколінеарність ------------------------------------

fit3

# Variance Inflation Factor - vif()
library(car)
fit3
vif(fit3)
vif(fit3) ^ 0.5
vif(fit3) ^ 0.5 > 2

# 8.7 Не лінійність ------------------------------------------

par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit3)
plot(fit2)
plot(ansFit_2)
par(mfrow=c(1,1));

# 8.8 Незалежність залишків ---------------------------------

# Durbin-Watson Test for Autocorrelated Errors - durbinWatsonTest()
# H0: Автокореляція дорівнює нулю
library(car)
durbinWatsonTest(fit3)

# пакет lmtest, функція dwtest())
# H0: Автокореляція дорівнює нулю
library(lmtest)
dwtest(fit3)

# Автокореляція залишків із іншими лагами
acf(resid(fit))
ccf(resid(fit), resid(fit))

get_values <- function(ccf_results) {
  lags <- as.vector(ccf_results$lag)
  corValues <- as.vector(ccf_results$acf)
  tVector <-
    corValues * sqrt(ccf_results$n.used - 2) / sqrt(1 - corValues ^ 2)
  pValue <- pt(-abs(tVector), df = ccf_results$n.used - 2) * 2
  tibble(lag = lags,
         cor = corValues,
         `p-value` = pValue)
}

get_values(ccf(resid(fit), resid(fit)))

# Breusch-Godfrey test for serial correlation: пакет lmtest, bgtest());
# H0: Автокореляція дорівнює нулю
bgtest(fit, order = 5)

# 8.9 Ідентифікація аутлаєрів та впливових величин ----------------

par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit)
par(mfrow=c(1,1), mfrow = c(1,1));
```

```r
comparison <-
  ag_clean %>%
  select(wht_yield_100kgha, year) %>%
  mutate(modelled_yields = c(NA, fitted(fit)))
plot(comparison$year, comparison$wht_yield_100kgha)
lines(comparison$year, comparison$modelled_yields)
summary(fit)


par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit)
par(mfrow=c(1,1), mfrow = c(1,1));


summary(fit3)
comparison <-
  ag_clean %>%
  select(wht_yield_100kgha, year) %>%
  mutate(modelled_yields = c(NA, fitted(fit3)))
plot(comparison$year, comparison$wht_yield_100kgha)
lines(comparison$year, comparison$modelled_yields)
summary(fit3)

par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit3)
par(mfrow=c(1,1), mfrow = c(1,1));

######### Приклад регресійних моделей із більшою вибіркою ######################

library(tidyverse)
library(readxl)
library(car)
library(lmtest)

# Завантажимо та очистимо альтернативні дані
oecd_fao_data <-
  read_csv("data/outlook_data.csv") %>%
  filter(COMMODITY != "CG") %>%
  select(COUNTRY, COMMODITY, VARIABLE, TIME, Value) %>%
  unite(CODE, COMMODITY, VARIABLE, sep = "_") %>%
  spread(CODE, Value) %>%
  left_join(
    read_csv("data/outlook_data_pse.csv") %>%
      filter(Unit == "Ratio") %>%
      select(LOCATION, PSECSE_INDICATOR, TIME, Value) %>%
      spread(PSECSE_INDICATOR, Value),
    by = c("COUNTRY" = "LOCATION", "TIME")) %>%
  rename(MA_PNPC = `MA-PNPC`,
         WT_PNPC = `WT-PNPC`,
         year = TIME) %>%
  filter(year < 2016) %>%
  group_by(COUNTRY) %>%
  mutate(lag_WT_PNPC = lag(WT_PNPC),
         lag_WT_AH = lag(WT_AH),
         lag_WT_PP = lag(WT_PP),
         lag_WT_YLD = lag(WT_YLD)) %>%
  ungroup()

# Виділимо Канаду у окремий об`єкт
```

```r
can_oecd <-
  oecd_fao_data %>%
  filter(COUNTRY == "CAN",
         year > 1990) %>%
  select(WT_YLD, lag_WT_PP, lag_WT_AH, year, WT_AH, lag_WT_PNPC, lag_WT_YLD)

# Step 1: Visual inspection
plot(can_oecd)

# Step 2: Model building
fit1 <- lm(WT_YLD ~ lag_WT_PP + lag_WT_AH + year + lag_WT_PNPC,
           data = can_oecd)

summary(fit1)

par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit1)
par(mfrow=c(1,1));

# Linearity
# Visually assumed
# Normality
shapiro.test(resid(fit1))

# Homoschedasticity
bptest(fit1)

# Multicollinearity
sqrt(vif(fit1)) > 2

# Residuals independence
durbinWatsonTest(fit1)
acf(resid(fit1))
ccf(resid(fit1), resid(fit1))

get_values <- function(ccf_results) {
  lags <- as.vector(ccf_results$lag)
  corValues <- as.vector(ccf_results$acf)
  tVector <-
    corValues * sqrt(ccf_results$n.used - 2) / sqrt(1 - corValues ^ 2)
  pValue <- pt(-abs(tVector), df = ccf_results$n.used - 2) * 2
  tibble(lag = lags,
         cor = corValues,
         `p-value` = pValue)
}

get_values(acf(resid(fit1)))

# Існує певна автокореляція залишків із лагом 1.

# Step 4. Regreession and data improvement
boxplot(can_oecd$WT_YLD)
can_oecd1 <-
  can_oecd %>%
  mutate(bad_wthr_2002 = ifelse(year == 2002, 1, 0),
         good_wthr_2013 = ifelse(year == 2013, 1, 0),
         lnt = year - 1990)

fit2 <- lm(
  WT_YLD ~ lag_WT_PP + lag_WT_AH + year + lag_WT_PNPC + bad_wthr_2002 + good_wthr_2013,
```

```r
  data = can_oecd1
)
summary(fit2)

# Зміна складу незалежних змінних
fit3 <- lm(
  WT_YLD ~ lag_WT_PP + lnt + lag_WT_PNPC + bad_wthr_2002 + good_wthr_2013,
  data = can_oecd1
)
summary(fit3)

par(mfcol=c(2,2), mfrow = c(2,2))
plot(fit3)
par(mfrow=c(1,1));
shapiro.test(resid(fit3)) # Normality assumption is violated
bptest(fit3)
sqrt(vif(fit3)) > 2
durbinWatsonTest(fit3)

round(coefficients(fit3), 6)

plot(can_oecd1$year, can_oecd1$WT_YLD,
     main = "Predicted yields vs data",
     xlab = "Year",
     ylab = "Wheat yields")

lines(x = can_oecd1$year, y = fitted(fit3))
```

Додаток 9. Панельна регресія в R

```r
# 9.0 Підготовка середовища та завантаження тренувальних даних ------------

library(tidyverse)
library(readxl)
library(car)

# install.packages("plm")
library(plm)

# Завантаження та очистка тренувальних даних;
dairy <-
  read_csv("data/dairy.csv") %>%
  select(FARM:FEED, YEAR93:YEAR98) %>%
  arrange(FARM, YEAR)

# 9.1 Візуальний аналіз панельних даних ---------

# Що таке панельні дані
dairy
View(dairy)

# Балансовані і не балансовані панельні дані
dairy %>%
  group_by(FARM) %>%
  summarise(n = n()) %>%
  ungroup() %>%
  summarise(min_n = min(n), max_n = max(n))

dairy %>%
  group_by(YEAR) %>%
  summarise(n = n()) %>%
  ungroup() %>%
  summarise(min_n = min(n), max_n = max(n))

# Візуальна інспекція
plot(select(dairy, FARM:FEED))

# Сенс панельних даних
fewFarms <- filter(dairy, FARM %in% sample(unique(dairy$FARM), 5))
scatterplot(MILK~YEAR|FARM, boxplots=F, smooth=TRUE, reg.line=FALSE, data = fewFarms)


# 9.2 Побудова базової лінійної регресії ---------------

linear_form <- MILK ~ COWS + LABOR + FEED + LAND
linear_production <- lm(linear_form, data = dairy)

summary(linear_production)

par(mfrow = c(2,2))
plot(linear_production)
par(mfrow = c(1,1))

# Візуальна інспекція припущень свідчить про ознаки не лінійності
# та нерівномірності варіації. Тому спробуємо виконати теоретично-обґрунтоване
# лінійне перетворення даних (використаємо виробничу функцію Коба-Дугласа).

# Вдосконалення лінійної регресії функція Коба-Дугласа
```

```r
cd_form <- log(MILK) ~ log(COWS) + log(LABOR) + log(FEED) + log(LAND)
cd_production <- lm(cd_form, data = dairy)

summary(cd_production)

par(mfrow = c(2,4))
plot(linear_production)
plot(cd_production)
par(mfrow = c(1,1))

# Приклад із функцією ‘транслог’
tl_form <- log(MILK) ~
  log(COWS) + log(LABOR) + log(FEED) + log(LAND) +
  I(0.5 * log(COWS) ^ 2) +
  I(0.5 * log(LABOR) ^ 2) +
  I(0.5 * log(FEED) ^ 2) +
  I(0.5 * log(LAND) ^ 2) +
  I(log(COWS) * log(LABOR)) +
  I(log(COWS) * log(FEED)) +
  I(log(COWS) * log(LAND)) +
  I(log(LABOR) * log(LAND)) +
  I(log(FEED) * log(LAND))+
  I(log(FEED) * log(LABOR))
tl_production <- lm(tl_form, data = dairy)

summary(tl_production)

# Порівняння трьох функціональних форм

par(mfrow = c(3,4))
plot(linear_production)
plot(cd_production)
plot(tl_production)
par(mfrow = c(1,1))

# Зважаючи на результат порівняння, оберемо функціональну форму Коба-Дугласа
cd_form <- log(MILK) ~ log(COWS) + log(LABOR) + log(FEED) + log(LAND)
cd_production <- lm(cd_form, data = dairy)

summary(cd_production)

par(mfrow = c(2,2))
plot(cd_production)
par(mfrow = c(1,1))

# те саме, що із lm але за допомогою пакету plm
library("plm")
cd_production_plm <-
  plm(cd_form, data = dairy, model = "pooling")

summary(cd_production_plm)

# 9.3 "Панельна регресія" із фіксованим ефектом метод "dummy variables" -----------

View(dairy)
# Припустимо, що виробництво молока може різнитися із року в рік, що пов'язано із
# захворюваннями корів. Для того, щоб компенсувати індивідуальний ефект кожного
# року, ми використаємо dummy-variable для кожного окремого року.
cd_form_2 <-
  log(MILK) ~ log(COWS) + log(LABOR) + log(FEED) + log(LAND) +
```

```r
  YEAR93 + YEAR94 + YEAR95 + YEAR96 + YEAR97

# Ні в якому разі не можна використовувати усі бінарні змінні через те,
# що виникне проблема абсолютної колінеарності і регресія буде не достовірною і не правильною.
# За погодний ефект у 98 році (або у році, який відсутній у переліку бінарних змінних)
# відповідатиме Intercept.
cd_production_2 <- lm(cd_form_2, data = dairy)

summary(cd_production_2)

par(mfrow = c(2,2))
plot(cd_production_2)
par(mfrow = c(1,1))

# те саме із іншим записом:
cd_form_3 <-
  log(MILK) ~ log(COWS) + log(LABOR) + log(FEED) + log(LAND) + factor(YEAR)
cd_production_3 <- lm(cd_form_3, data = dairy)

summary(cd_production_2)

# 9.4 Панельна регресія із фіксованим ефектом ------------------------

library(plm)

fixef_time_plm <-
  plm(formula = cd_form,
      data = dairy,
      index = c("FARM", "YEAR"),
      effect = "time",
      model = "within")

# General statistics
summary(fixef_time_plm)

fixef(fixef_time_plm)

# 9.5 Перегляд результатів панельної регресії та коефіцієнтів -----------

# General statistics
summary(fixef_time_plm)

# регресійні коефіцієнти
coefficients(fixef_time_plm)

# Фіксовані ефекти
fixef(fixef_time_plm)
fixef(fixef_time_plm, effect = "individual")
fixef(fixef_time_plm, effect = "time")

# Порівняння фіксованих ефектів із результатами звичайної регресії:
coefficients(cd_production_2)[6:10] + coefficients(cd_production_2)[1]
coefficients(cd_production_2)[1]

c(coefficients(cd_production_2)[6:10] +
    coefficients(cd_production_2)[1],
  coefficients(cd_production_2)[1])

# Residuals
```

```r
resid(fixef_time_plm)
residuals(fixef_time_plm)

#### Альтернативна панельна регресія із фіксованим ефектом ферми (не часу)
fixef_farm_plm <-
  plm(formula = cd_form,
      data = dairy,
      index = c("FARM", "YEAR"),
      effect = "individual",
      model = "within")

summary(fixef_farm_plm)
summary(fixef_time_plm)

coefficients(fixef_farm_plm)
coefficients(fixef_time_plm)

fixef(fixef_farm_plm)

#### Альтернативна панельна регресія із фіксованим ефектом ферми і часу
fixef_farm_time_plm <-
  plm(formula = cd_form,
      data = dairy,
      index = c("FARM", "YEAR"),
      effect = "twoways",
      model = "within")

coefficients(fixef_farm_plm)
coefficients(fixef_time_plm)
coefficients(fixef_farm_time_plm)

summary(fixef_farm_time_plm)

# Для того, щоб побачити фіксований ефект часу
fixef(fixef_farm_time_plm, effect = "time")

# Для того, щоб побачити фіксований ефект фірми
fixef(fixef_farm_time_plm, effect = "individual")

# 9.6 Визначення чи є доцільність у панельній регресії ------------------

# Для цього порівняння нам необхідно використати pFtest()
library(plm)
# Для тесту, нам необхідно мати декілька регресій побудованими із однаковою формулою:
# * Звичайну лінійну регресію: cd_production або cd_production_plm
# * панельну регресію із ефектом ферми: fixef_farm_plm
# * панельну регресію із ефектом часу: fixef_time_plm
# * панельну регресію із ефектом ферми і часу: fixef_farm_time_plm

# Ф-тест
# Чи є доцільність фіксованого ефекту часу
pFtest(fixef_time_plm, cd_production)
# або
pFtest(fixef_time_plm, cd_production_plm)
# або
pFtest(cd_form, data = dairy, index = c("FARM", "YEAR"), effect = "time")
# Нульова гіпотеза: вплив будь-якого ефекту порівняно із лінійною регресією
# дорівнює нулю. p-value >= 0.05 Приймаємо нульову гіпотезу
# p-value < 0.05 відхиляємо нульову гіпотезу
```

```r
# Чи є доцільність фіксованого ефекту ферми
pFtest(fixef_farm_plm, cd_production)
# або
pFtest(fixef_farm_plm, cd_production_plm)
# або
pFtest(cd_form, data = dairy, index = c("FARM", "YEAR"), effect = "individual")
# Нульова гіпотеза: вплив будь-якого ефекту порівняно із лінійною регресією
# дорівнює нулю. p-value >= 0.05 Приймаємо нульову гіпотезу
# p-value < 0.05 відхиляємо нульову гіпотезу

# Чи є доцільність фіксованого ефекту ферми і часу
pFtest(fixef_farm_time_plm, cd_production)
# або
pFtest(fixef_farm_time_plm, cd_production_plm)
# або
pFtest(cd_form, data = dairy, index = c("FARM", "YEAR"), effect = "twoways")
# Нульова гіпотеза: вплив будь-якого ефекту порівняно із лінійною регресією
# дорівнює нулю. p-value >= 0.05 Приймаємо нульову гіпотезу
# p-value < 0.05 відхиляємо нульову гіпотезу

# 9.7 Побудова панельної регресії із випадковим ефектом ----------------------

rand_farm_plm <-
  plm(formula = cd_form,
      data = dairy,
      index = c("FARM", "YEAR"),
      model = "random")

# Перегляд результатів панельної регресії із випадковим ефектом
rand_farm_plm
summary(rand_farm_plm)
coefficients(rand_farm_plm)
resid(rand_farm_plm)
residuals(rand_farm_plm)

# Порівняння результатів пан. рег. із випадковим ефектом і фіксованим ефектом
tibble(farm_fixed = coefficients(fixef_farm_plm),
       time_fixed = coefficients(fixef_time_plm),
       farm_time_fixed = coefficients(fixef_farm_time_plm),
       random_ef = coefficients(rand_farm_plm)[2:5])

# 9.8 Визначення, що краще: фіксований чи випадковий ефект ------------------

# Для цього порівняння нам необхідно використати phtest() Hausman Test
# Нульова гіпотеза: модель із випадковим ефектом надає кращі оцінки регресії

# Для тесту, нам необхідно мати декілька регресій побудованих за однаковою формулою:
# * панельну регресію із ефектом ферми або часу або ферми і часу:
# fixef_farm_plm, fixef_farm_time_plm, fixef_time_plm
# * панельну регресію із випадковим ефектом: rand_farm_plm

# Hausman Test
# Нульова гіпотеза: модель із випадковим ефектом надає кращі оцінки регресії

# Чи є випадковий ефект кращим за фіксований ефект ферми?
phtest(fixef_farm_plm, rand_farm_plm)
phtest(rand_farm_plm, fixef_farm_plm)
# або
phtest(cd_form, data = dairy)
```

```r
# те саме для ефекту часу і ферми та часу?
phtest(fixef_time_plm, rand_farm_plm)
phtest(fixef_farm_time_plm, rand_farm_plm)
# p-value > 0.5 - YES; p-value <= 0.5 - NO

# 9.9 Тестування крос-секційної залежності та кореляції у часі -------------------

# Тест перевіряє взаємозалежність залишків між частинами панельних даних
# Наявність такої залежності може призвести до неправильних коефіцієнтів

# Нульова гіпотеза: Взаємозалежність існує
# Для тесту необхідна звичайна панельна регресія із індивідуальним ефектом (ферми)
# або ефектами часу або ферми і часу

# Метод Breusch and Pagan's original Lagrange Multiplier
pcdtest(fixef_farm_plm, test = c("lm"))
pcdtest(fixef_time_plm, test = c("lm"))
pcdtest(fixef_farm_time_plm, test = c("lm"));
# p-value > 0.5 - Взаємозалежність існує
# p-value <= 0.5 - Взаємозалежність не існує

# 9.10 Breusch-Godfrey/Wooldridge тест серійної кореляції ---------------------

# Тест перевіряє серійну кореляцію залишків
# Э актуальним виключно для макро-моделей із тривалими часовими рядами

# Нульова гіпотеза: Серійна кореляція дорівнює нулю
# Для тесту необхідна звичайна панельна регресія із індивідуальним ефектом (ферми)
# або ефектами часу або ферми і часу
pbgtest(cd_production_plm)
pbgtest(fixef_time_plm)
pbgtest(fixef_farm_plm)
pbgtest(fixef_farm_time_plm)

# 9.11 Dickey-fuller тест про стохастичний тренд та стаціонарність ---------------------

# install.packages("tseries")
library(tseries)

# Конвертації даних у панельні дані
panel_data_converted <-
  pdata.frame(dairy, index = c("FARM", "YEAR"))

# Нульова гіпотеза: Часовий ряд не має стаціонарності (unit root присутній)
adf.test(panel_data_converted$MILK, k = 1)
adf.test(panel_data_converted$MILK, k = 2)
adf.test(panel_data_converted$MILK, k = 3)

# 9.12 Перевірка припущень ---------------------------------

# Припущення

# Лінійність
# Аналізуємо спираючись на результати лінійної регресії без
#   панельної складовою і теоретично обґрунтовуємо лінійність

# Нормальність - припускаємо спираючись на центральну граничну теорему
qqnorm(residuals(fixef_farm_plm), ylab='Residuals')
```

```r
qqline(residuals(fixef_farm_plm))
hist(residuals(fixef_time_plm), xlab='Residuals')

# Постійність розподілу варіації
plot(fitted(fixef_time_plm), resid(fixef_time_plm))

# Нульова гіпотеза: постійна варіація
library(lmtest)
bptest(
  log(MILK) ~ log(COWS) + log(LABOR) +
    log(FEED) + log(LAND) + factor(FARM) + factor(YEAR),
  data = dairy,
  studentize = F)

# Візуальна інспекція
resid_data <-
  dairy %>%
  mutate(resid = resid(fixef_farm_time_plm)) %>%
  select(FARM, YEAR, resid) %>%
  filter(FARM %in% sample(unique(dairy$FARM), 50))

boxplot(resid_data$resid ~ resid_data$FARM)
boxplot(resid_data$resid ~ resid_data$YEAR)

# Мультиколінеарність - припускаємо теоретично її відсутність

# В разі існування проблеми із постійністю розподілу варіації необхідно:
# провести коректовані оцінки коефіцієнтів значимості регресії.

# Для фіксованого ефекту:

# Звичайні коефіцієнти
summary(fixef_farm_plm)
# Коректовані значущості
fixeed_ef_vcov <- vcovHC(fixef_farm_plm, method = "arellano", type = "HC1")
coeftest(fixef_farm_plm, vcov. = fixeed_ef_vcov)

# Звичайні коефіцієнти
summary(fixef_time_plm)
# Коректовані значущості
coeftest(fixef_time_plm,
         vcov. = vcovHC(fixef_time_plm, method = "arellano", type = "HC1"))

# Звичайні коефіцієнти
summary(fixef_farm_time_plm)
# Коректовані значущості
coeftest(fixef_farm_time_plm,
         vcov. = vcovHC(fixef_farm_time_plm, method = "arellano", type = "HC1"))

# Для випадкового ефекту:
# Звичайні коефіцієнти
summary(rand_farm_plm)

# Коректовані значущості
# white1 - коли присутня нерівномірність розподілу залишків, але не має серійної кореляції
coeftest(rand_farm_plm,
         vcov. = vcovHC(rand_farm_plm, method = "white1", type = "HC1"))

# white2 - коли присутня нерівномірність розподілу залишків та серійна кореляція
```

```r
coeftest(rand_farm_plm,
         vcov. = vcovHC(rand_farm_plm, method = "white2", type = "HC1"))
```