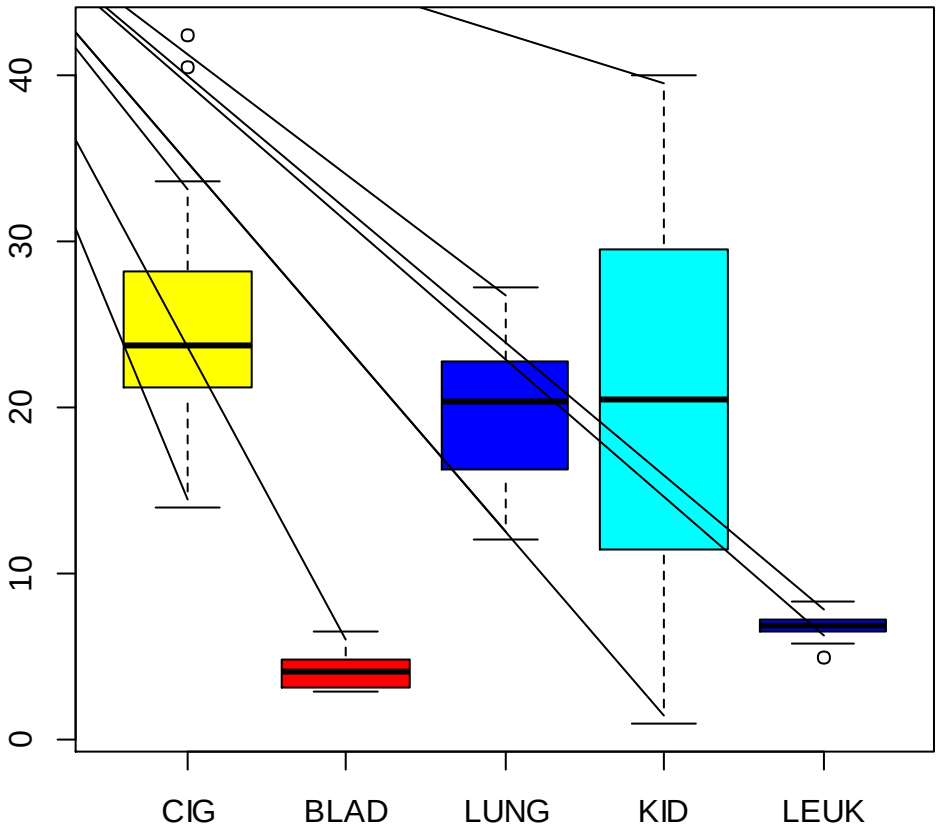
АНАЛІЗ ДАНИХ
ЗА ДОПОМОГОЮ ПАКЕТА R

Р.Є. Майборода, О.В. Сугакова



Київ - 2015

УДК 519.22.35(075.8)
ББК 22.172я73

Рецензенти:

канд. фіз.-мат. наук, доц. Василик О.І.
канд. фіз.-мат. наук, доц. Ямненко Р.Є.

Майборода Р.Є., Сугакова О.В.

Аналіз даних за допомогою пакета R: Навчальний посібник

Описано статистичний аналіз даних за допомогою програмного середовища R. Містить загальні рекомендації з користування пакетом Головну увагу приділено застосуванню R до прикладних задач математичної статистики, а також розбору прикладів застосування до реальних даних.

Для студентів кваліфікаційного рівня «бакалавр» та «магістр» всіх спеціальностей, які прослухали курс математичної статистики і спрямовані на оволодіння прикладними (комп'ютерними) аспектами статистичного аналізу даних.

Затверджено вченою радою факультету радіофізики,
електроніки та комп'ютерних систем
(протокол №10 від 20 квітня 2015 року)

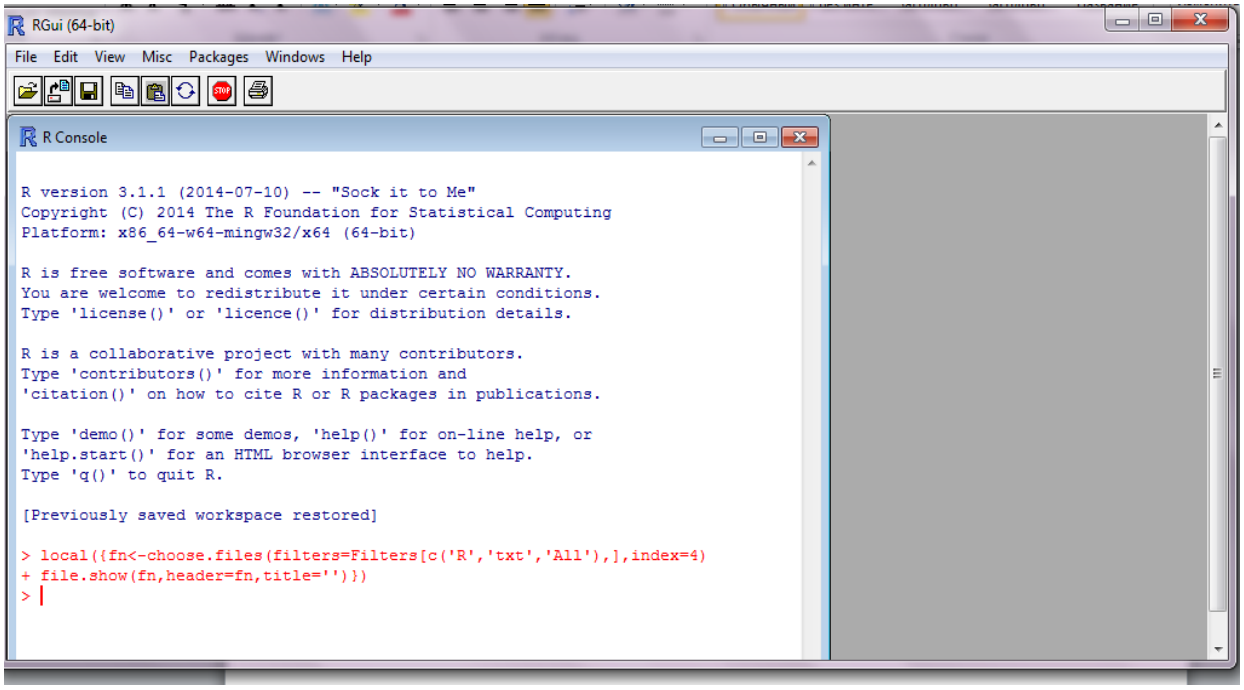
УДК 519.22.35(075.8)
ББК 22.172я73

1. Початок роботи з системою R

R - це середовище програмування для статистичного аналізу даних. Це середовище складається з базової програми R, що працює як інтерпретатор мови статистичного програмування S, та окремих пакетів, які реалізують спеціальні методи та технології статистичної обробки даних. Базова програма створена у рамках проекту GNU, як альтернативна програмна реалізація мови S (ця мова та комерційний пакет S+ для її реалізації були розроблені у Bell Laboratories під керівництвом Дж. Чемберса). На відміну від S+, програма R є некомерційною і вільно розповсюджується за умови дотримання вимог GNU General Public License. Офіційна сторінка проекту R: http://www.r-project.org/ .

Отримати останню версію інсталятора базової програми R для операційної системи Windows можна за адресою: http://cran.r-project.org/bin/windows/base/ . (На 1 січня 2015 це була версія R-3.1.2). Інсталятор завантажується у вигляді exe-файлу. Для інсталяції програми досить запустити цей файл і відповідати на його запитання. При першій спробі роботи з R рекомендовано погоджуватись з усіма пропозиціями, які робить інсталятор. Проблеми можуть виникнути, якщо на вашому комп'ютері встановлені різні права доступу для різних користувачів. Справа в тому, що R наприкінці кожної сесії роботи зберігає на диску «робочий простір» (workspace) – сукупність даних та програм, які були завантажені під час сесії. На початку наступної сесії workspace завантажується з диску. Якщо під час інсталяції для зберігання workspace буде обрано директорію, недоступну певному користувачеві, то при роботі з R можуть виникати повідомлення про неможливість завантаження або зберігання workspace. Для усунення таких повідомлень потрібно або вибрати директорію вільного доступу при інсталяції, або змінити директорію, використовуючи пункт File->ChangeDir… у головному меню головного вікна програми R.

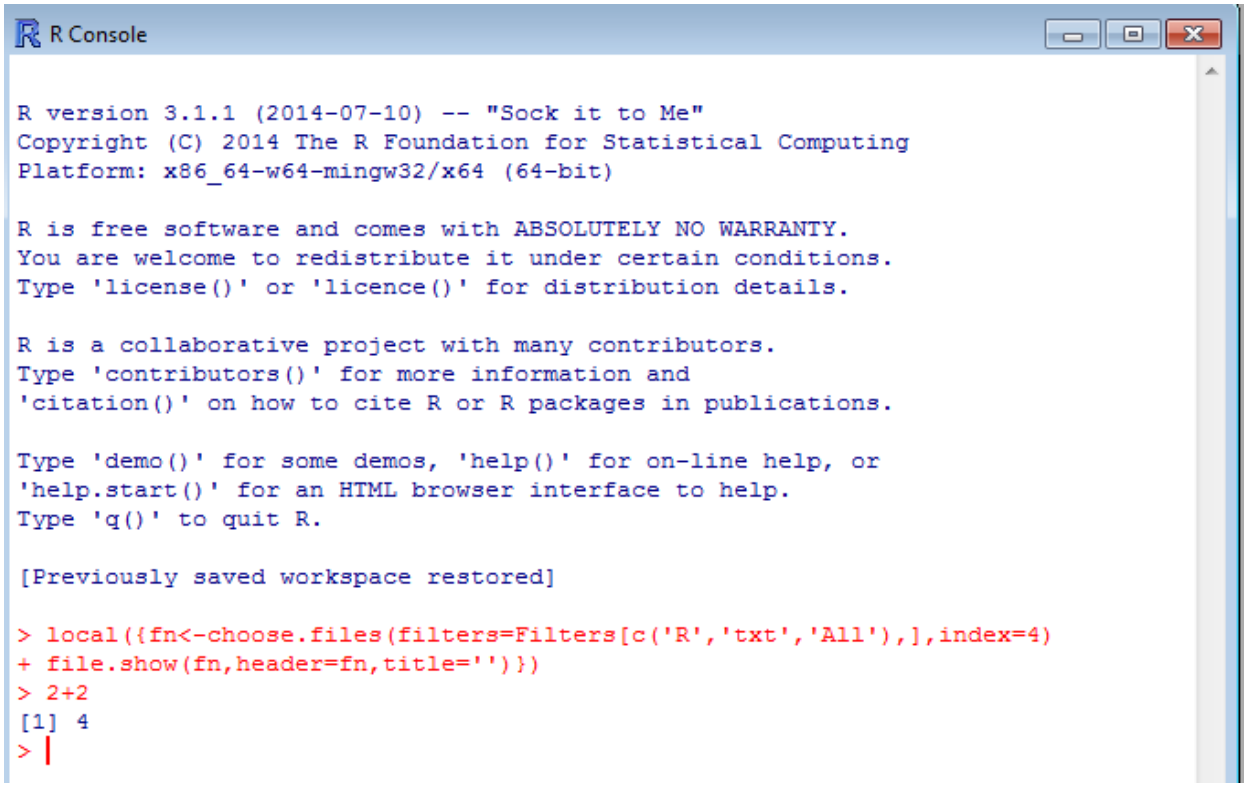
Після інсталяції R його можна запустити і отримати приблизно таке вікно:



Тут вгорі знаходиться головне меню, а нижче відкрито вікно «консолі R», в якій можна давати інструкції програмі та отримувати її відповіді. Синім кольором у цьому вікні виведено початкову інформацію про вашу версію базової програми R. Далі червоним кольором вказана інструкція, яку R виконав автоматично при завантаженні. Нарешті, червоний символ > є запрошенням користувачу вводити власні інструкції. Для перевірки роботи системи можна після > ввести

2+2

і натиснути Enter. Результат буде виведений на консоль:



Як бачимо, R виводить результати виконання безпосередньо після інструкції синім кольором, після чого переходить у режим очікування наступної інструкції, про що повідомляє червоним знаком >.

При роботі з R можна виконувати одразу багато інструкцій, що записані в окремому файлі. Найпростіший спосіб зробити це – завантажити такий файл в якому-небудь текстовому редакторі, зробити там сору, а потім – paste на консолі. При цьому, якщо інструкції у файлі розміщені у окремих рядочках, розділових знаків між ними не потрібно. Інструкції, вміщені в одному рядочку, розділяють символом ;. Якщо довга інструкція не вміщується у одному рядочку, її можна розбити на декілька рядочків, причому, при переході до наступного рядочку R автоматично виводить символ продовження +. R сам здогадується, що інструкція не закінчена за її синтаксисом. Тому деякі синтаксичні помилки (як от – забуті дужки) можуть сприйматись як незакінчені інструкції. У цьому випадку R виставить + на початку наступного рядочка і перейде у режим очікування. Натисніть escape, щоб перейти у режим введення нової інструкції без продовження аналізу попередньої.

Програми, що складаються з інструкцій R, називають скриптами (script). Вони мають стандартне розширення .r. У базовій програмі є можливість відкрити вікно редактора для створення нового скрипту, або завантажити файл зі скриптом, використовуючи пункти головного меню File->Newscript або File->Openscript. Виконати завантажений у вікні редактора скрипт повністю можна, використовуючи Edit->Runall. Можна також виконати виділену частину скрипту, використовуючи кнопку  (Run line or selection). Закінчивши роботу зі скриптом, його можна зберегти, використовуючи File->Save.

Однак, редактор скриптів, реалізований у R, має дуже обмежену функціональність. Тому доцільно для редагування більш-менш складних скриптів використовувати зручніші текстові редактори. Ви можете користуватись тим редактором, до якого звикли, але спеціально для редагування R-скриптів зручно застосовувати редактор Notepad++. Це некомерційна, вільно розповсюджувана програма, яку можна отримати на офіційному сайті http://notepad-plus-plus.org/ . Працюючи з файлами, що мають розширення .r, Notepad++ трактує їх як скрипти системи R і виділяє логічну структуру скрипту:

```r
# Kendall correlation with NA deletion
kend<-function(x,y){
 x<-x[!(is.na(y))]
 y<-y[!(is.na(y))]
 b<-cor.test(x,y,method="kendall")
 c(cor(x,y,method="kendall"),b$p.value)
}


z<-read.csv2(file.choose()) #data1.csv
Z<-z[,-(12:13)] #nonnumbers deleted
n<-dim(z)[2]-1
nms<-names(z)[-1]
tau1<-rep(NA,n)
p<-rep(NA,n)
for(i in 1:n){
if(is.numeric(z[,i+1])){
tt<-kend(z[,1],z[,i+1])
tau1[i]<-tt[1]
p[i]<-tt[2]
 }
}
res<-data.frame(nms,tau1,p)
write.table(res,file=file.choose())
```

Базова програма R є лише інтерпретатором одного з діалектів мови програмування S. Конкретні статистичні алгоритми, як правило, виконуються у вигляді скриптів на цій мові і зібрані у пакети (packages) R. При інсталяції разом з базовою програмою інсталюються і основні пакети, які реалізують найбільш популярні технології статистичної обробки. Частина цих пакетів автоматично завантажується при запуску R. Ті пакети, які не завантажились автоматично, але інстальовані на комп'ютері, можна завантажити функцією library:

```
>library(splines)
```

завантажує пакет splines, призначений для роботи зі сплайнами. Ця ж функція, викликана без параметрів, дає перелік усіх завантажених пакетів з коротким поясненням їх функцій (у мінімальному варіанті цього переліку кілька десятків пакетів).

Якщо потрібний пакет не було інстальовано на комп'ютері, його можна завантажити з інтернет-архіву, використовуючи пункти головного меню Packages->Installpackage(s). Спочатку програма пропонує вибрати інтернет-архів, з якого робиться інсталяція. Варіант 0-cloud, що пропонується за умовчанням, як правило, працює цілком задовільно. Після цього треба у списку вибрати потрібний для вас пакет. Якщо цей пакет використовує які-небудь інші, котрих немає на вашому комп'ютері, вони будуть інстальовані автоматично. Після того, як пакети інстальовані, вони зберігаються на вашому комп'ютері, але для роботи з ними під час сеансу їх треба підключати, використовуючи функцію library.

Усі об'єкти, створені або завантажені під час сеансу роботи з R і не видалені спеціальною інструкцією, зберігаються у робочому просторі. Наприкінці сеансу R запитує, чи зберігати робочий простір на диску:

Save workspace image?

Якщо вибрати збереження, то цей робочий простір буде відновлено на початку наступного сеансу. Такою можливістю варто користуватись дуже обережно, оскільки «старі» об'єкти можуть спотворювати роботу R у новому сеансі. Зберігати робочий простір доцільно лише в тому випадку, коли ви збираєтесь наступного разу продовжити свою роботу з того самого місця, на якому зупинились зараз.

Існує великий обсяг англомовної літератури по R та мові програмування S. Найбільш корисною для початківця є книжка

W.N. Venables, B.D. Ripley «Modern Applied Statistics with S»,

котру можна знайти, наприклад, тут:

http://www.planta.cn/forum/files_planta/modern_applied_statistics_with_s_192.pdf

Крім того, дуже зручно користуватись офіційними керівництвами з мови S, що входили у стандартну поставку пакету S+ (На даний час комерційний проект S+ припинено, версій цієї програми під сучасну Windows, наскільки нам відомо, не існує). При роботі з такою літературою слід мати на увазі, що R як діалект дещо відрізняється від стандартного S з пакету S+. Крім того, кожен додатковий пакет може мати свої особливості, які пояснюються у документації до нього. Це може створити певні незручності, але досвід показує, що такі проблеми легко розв'язуються пошуком у інтернеті по назві пакету або функції із додаванням ключових слів "rsoftware".

2. Робота з даними(найпростіші операції і типи даних)

Для виконання певної операції необхідно натиснути клавішу Enter.

Якщо необхідно скопіювати команду, яку ви вже виконували раніше в цьому файлі, можна скористатись клавішами ↑ та ↓.

Для виклику **help,** що стосується певної команди, слід набрати ?<<назва команди>> і натиснути клавішу Enter; якщо вас цікавить опція, а не команда, можна спробувати набрати ??<<назва опції>>. Коментарі відмічаються символом #.

Як було сказано вище, зберігати робочий файл рекомендується, скопіювавши його в Блокноті, в текстовому файлі. R пропонує збереження лише значень змінних.

1) Арифметичні операції в R. Вони позначаються звичайним чином: +, -, *,/, ^.

```
> (7*3)+12/2-7^2
[1] -22
```

Також є в наявності **елементарні функції**: log( ), log10( ), exp( ), sin( ), cos( ), tan( ), sqrt( ), а також abs( ). Функція round(x,n) округлює число до n десяткових знаків після коми.

Логічні операції.

Це операції: <, >, <= (менше або дорівнює); >= (більше або дорівнює); == (дорівнює); != (не дорівнює); & (переріз); | (об'єднання).

2)Присвоєння значення об'єктам традиційно здійснюється за допомогою символу<-. Також використовується знак =.

```
> x<-7
> x+3
[1] 10
```

3)Статистичні операції:

mean(x) обчислює **вибіркове середнє** масиву x;

sd(x) обчислює вибіркове **середньоквадратичне відхилення** x;

var(x) обчислює **вибіркову дисперсію** масиву x;

summary(x) виводить елементи дескриптивної статистики масиву x: мінімальне значення, максимальне, обидві квартилі, медіану і середнє.

range(x) повертає найбільше і найменше значення в x. Якщо нас цікавить різниця між найбільшим і найменшим значеннями, можна скористатись функцією diff(range(x)). Тут 1:10 означає послідовність цілих чисел від 1 до 10, як і в мові С, наприклад.

```
> range(1:10)
[1]  1 10
> diff(range(1:10))
[1] 9
>
```

cor(x,y) обчислює **коефіцієнт кореляції Пірсона** між масивами x і y однакової довжини. Якщо x – матриця, то команда cor(x,x) видає **матрицю кореляцій** даних.

Для обрахування **коефіцієнта кореляції Спірмена** необхідно в тій самій команді задати додаткову опцію method. Операція c( ) задає вектор, записуючи його в змінну x.

```
> x<-c(-100,0,11)
> y<-c(134,2,-6)
> cor(x,y)
[1] -0.9992336
> cor(x, y, method = "spearman")
[1] -1
```

4)Вектор - це один з можливих типів даних. Створюємо вектор за допомогою операції c( ). Для зберігання величин 1.5; 2; і 3 у векторі x друкуємо

```
> x<-c(1.5,2,3)
> x
[1] 1.5 2.0 3.0
```

У вектор можна помістити дані інших типів. Наприклад, текстові:

```
> y<-c("A", "Hello", "world!")
> y
[1] "A"      "Hello"  "world!"
```

Можна з'єднати 2 вектори в одну структуру з двох стовпчиків. Для цього використовують команду cbind( ).

```
> x<-c(2,3,4,1)
> y<-c(1,1,1,10)
> cbind(x,y)
     x  y
```

```
[1,] 2  1
[2,] 3  1
[3,] 4  1
[4,] 1 10
```

Об‘єднання двох рядків в одну структуру здійснюється за допомогою rbind( ) :

```
> rbind(x,y)
  [,1] [,2] [,3] [,4]
x    2    3    4    1
y    1    1    1   10
```

Операції додавання, різниці, множення векторів відбуваються **поелементно.** Якщо ж вектори, що, наприклад, додаються, мають різні довжини, то коротший вектор «циклічно» продовжується до розміру довгого, і після цього проводиться додавання поелементно.

```
> x<-c(-100,0,1,5,-9,8,4)
> y<-c(1,2)
> y+x
[1] -99   2   2   7  -8  10   5
Предупреждение
In y + x :
длина большего объекта не является произведением длины меньшего объекта
```

5)Послідовності.

Команда seq( ) створює **послідовність** чисел. Її часто використовують при графічному аналізі. Три аргументи, які зазвичай використовують в команді: початкове значення, кінцеве і крок (приріст). Якщо ж приріст =1, то достатньо двох аргументів.

Приклад. Створимо послідовність чисел від 0.0 до 1.0 з кроком 0.2. А потім від 0 до 8 з одиничним кроком.

```
> seq(0,1,0.2)
[1] 0.0 0.2 0.4 0.6 0.8 1.0
>
> seq(0,8)
[1] 0 1 2 3 4 5 6 7 8
```

Для стислого запису послідовності цілих від 0 до 8 можна скористатись **діапазоном** 0:8.

```
> 0:8
[1] 0 1 2 3 4 5 6 7 8
```

Для **повторення** в послідовності n разів однакових чисел або символів використовують команду rep(a,n)

```
> rep(c(0,"x"),3)
[1] "0" "x" "0" "x" "0" "x"
```

Також можна регулювати довжину послідовності.

```
> rep(c(1,2,3), length=10)
 [1] 1 2 3 1 2 3 1 2 3 1
>
```

Відмітимо, що пакет R використовує круглі дужки ( ) для аргументів функцій і квадратні [ ] для того, щоб звернутись до конкретного елемента в послідовності, векторі, масиві, списку.

Створимо масив чисел

```
> t<-c(2,3,-1,77,128,3)
```

Викличемо другу компоненту вектора.

```
> t[2]
[1] 3
```

А тепер з другої по четверту:

```
> t[2:4]
[1]  3 -1 77
```

І окремо вилучимо 1-шу, 4-ту, 6-ту компоненти.

```
> t[c(1,4,6)]
[1]  2 77  3
>
```

За допомогою логічних операцій можна, записавши їх в квадратних дужках, «виудити» будь-які елементи масиву, за бажанням. Наприклад, знайдемо лише від'ємні елементи масиву:

```
> t[t<0]
[1] -1
```

Або за модулем більші за 70:

```
> t[abs(t)>70 ]
[1]  77 128
```

Або всі елементи, крім першого:

```
> t[-1]
[1]   3  -1  77 128   3
```

Аналогічним чином можна працювати з індексами в інших типах даних – наприклад, коли маємо справу з матрицями.

6)Списки. Список – це структура, тобто вектор, елементи якого можуть мати різні типи: числові, текстові і т.д. Елементом списку може бути інший список. Списки створюються за допомогою команди list().

Тут ми створюємо список student з чотирма полями.

```
>student<-list(name="Olexander", surname="Kovalenko", major="statistics", year.of.entering=2015 )
```

Вилучити потрібну компоненту списка можна або за допомогою подвійних квадратних дужок

```
>student[[4]]
[1] 2015
>
```

або виписавши ім'я потрібного поля списку, використавши символ $.

```
>student$year.of.entering
[1] 2015
>
```

7)Матриці.

В матриці дані розташовані в рядках і стовпчиках прямокутної таблиці.

Приклад.

Розглянемо матрицю

$$b = \begin{pmatrix} 190 & 8 & 22.0 \\ 191 & 4 & 1.7 \\ 223 & 80 & 2.0 \end{pmatrix}.$$

Створити таку матрицю в R можна за допомогою команди matrix(). Присвоїмо їй ім'я b.data. Якщо ми хочемо, щоб дані записувались в матрицю по рядках, використовуємо опцію byrow=TRUE. В даному прикладі опція nrow=3 показує кількість рядків матриці.

1)Створимо матрицю

```
> Data<-c(190,8,22,191,4,1.7,223,80,2)
> b.data<-matrix(Data, nrow=3, byrow=TRUE)
> b.data
     [,1] [,2] [,3]
[1,]  190    8 22.0
[2,]  191    4  1.7
[3,]  223   80  2.0
>
```

2)Для визначення розмірності матриці можна надрукувати:

```
> dim(b.data)
[1] 3 3
```

3)Подивимось, як можна надписати рядки або стовпчики матриці. Для цього використаємо команду dimnames().

```
>region<-c("East","Middle","West")
>dimnames(b.data)<-list(region,NULL)
> b.data
       [,1] [,2] [,3]
East    190    8 22.0
Middle  191    4  1.7
West    223   80  2.0
>type<-c("type A","type B","type C")
> dimnames(b.data)<-list(NULL,type)
> b.data
     type A type B type C
[1,]    190      8   22.0
[2,]    191      4    1.7
[3,]    223     80    2.0
> dimnames(b.data)<-list(region,type)
> b.data
       type A type B type C
East      190      8   22.0
Middle    191      4    1.7
West      223     80    2.0
>
```

4)Додавання матриць A+B відбувається поелементно. Множення A*B також – поелементно. Якщо матриці мають різну розмірність, то операція не виконується.

5)Операція множення матриць – в математичному сенсі - виглядає як A%*%B. Наприклад,

```
> Data<-c(1,0,0,1)
> A<-matrix(Data, nrow=2, byrow=TRUE)
> A
     [,1] [,2]
[1,]    1    0
```

```
[2,]    0    1
> Data<-c(1,-9,1,1)
> B<-matrix(Data, nrow=2, byrow=TRUE)
> C<-A%*%B
> C
     [,1] [,2]
[1,]    1   -9
[2,]    1    1
```

6)Операція diag( ) перетворює діагональні елементи матриці в елементи вектора.

```
> z<-diag(C)
> z
[1] 1 1
```

8)Фрейми даних.

Фрейм - найбільш широковживаний тип змінних в R, який використовується для зберігання даних. Фрейми складаються з різноманітних типів даних (числових, текстових, логічних і т.д.). Традиційно, стовпчики розглядаються як змінні, рядки містять характеристики об'єктів.

Приклад.

1)Створюємо файл *.txt з даними. Зчитуємо ці дані у фрейм за допомогою команди

```
> da<-read.table(file.choose(),header=T)
> da
       Value_A Value_B Value_C
First      190       8    22.0
Second     191       4     1.7
Third      223      80     2.0
```

2)Тепер ми хочемо залучити дві додаткові змінні: логічну змінну Opinion і текстову Danger. Створюємо відповідні стовпчики-масиви і заносимо їх у фрейм.

```
> Opinion<-c(F,T,F)
> Danger<-c("no","no","yes")
> y<-data.frame(da,Opinion,Danger)
> y
       Value_A Value_B Value_C Opinion Danger
First      190       8    22.0   FALSE     no
Second     191       4     1.7    TRUE     no
Third      223      80     2.0   FALSE    yes
> rm("Danger")
> rm("Opinion")
```

3)Тепер ми хочемо надрукувати дані, що відповідають лише змінній Value_A.

```
>y$Value_A
[1] 190 191 223
```

4)Для того, щоб звертатись до стовпчиків фрейму спрощено, лише за ім'ям, треба скористатись операцією attach(). А щоб зняти це маскування, треба застосувати detach().

```
>attach(y)
>Value_A
[1] 190 191 223
```

5)Відсортуємо дані відповідно змінній Value_C.

```
>y[sort.list(y[,3]),]
Value_AValue_B Value_C OpinionDanger
Second    191      4    1.7    TRUE     no
Third     223     80     2.0   FALSE    yes
First     190      8    22.0       FALSE    no
```

Читання і запис даних. Редагування.

Як вже зазначалося, спершу треба створити файл *.txt з даними, в Блокноті або Notepad. В пакеті можна зчитати ці дані у фрейм da за допомогою команди

```
> da<-read.table(file.choose(),header=T)
```

Перша з опцій вказує на те, що файл для зчитування треба задати вручну. При виконанні операції треба відкрити необхідний файл, вказавши шлях до нього. Друга опція вказує на те, що перший рядок таблиці – імена змінних. В іншому випадку, якщо файл одразу починається з даних, замість TRUE друкуємо FALSE. Тоді знов-таки дані зчитаються в фрейм, але змінні будуть називатись V1, V2 тощо.

Нехай ми зчитали дані у фрейм, і вирішили їх трохи змінити. Наприклад, дати ім'я змінним. Для цього є зручна команда edit( )

```
>da<-edit(da)
```

Відкриється таблиця, що містить фрейм, і можна вручну виправити всі дані чи константи.

3. Робота з розподілами

Кожному розподілу в R відповідають чотири функції. Корінь назви функції вказує на вид розподілу. Перша літера – префікс – розшифровується наступним чином

- p позначає функцію розподілу
- q позначає квантильну функцію, тобто, обернену до функції розподілу.
- d позначає щільність розподілу.
- r – функція з таким префіксом генерує випадкову величину, яка має вказаний розподіл.

Для дискретної випадкової величини під поняттям «щільність» розуміємо $f(x) = P\{X = x\}$.

Розподіл	F	Q	f	random
Бета	pbeta	qbeta	dbeta	rbeta
Біноміальний	pbinom	qbinom	dbinom	rbinom
Коші	pcauchy	qcauchy	dcauchy	rcauchy
Хі-квадрат	pchisq	qchisq	dchisq	rchisq
Експоненціальний	pexp	qexp	dexp	rexp
Гама	pgamma	qgamma	dgamma	rgamma
Геометричний	pgeom	qgeom	dgeom	rgeom
Гіпергеометричний	phyper	qhyper	dhyper	rhyper
Логістичний	plogis	qlogis	dlogis	rlogis
Логнормальний	plnorm	qlnorm	dlnorm	rlnorm
Від'ємний	pnbinom	qnbinom	dnbinom	rnbinom

біноміальний				
Нормальний	pnorm	qnorm	dnorm	rnorm
Пуассона	ppois	qpois	dpois	rpois
Стьюдента	pt	qt	dt	rt
Рівномірний	punif	qunif	dunif	runif
Вейбулла	pweibull	qweibull	dweibull	rweibull

Приклади.

1)Біноміальний розподіл.

Генерування вибірки обсягу 100 з біноміальним розподілом з параметрами $n = 5$, $p = 0.5$.

```
> x<-rbinom(100,5,0.5)
> x
 [1] 1 2 5 3 1 2 3 3 2 3 2 3 1 2 3 2 1 3 2 2 4 2 1 3 2 3 4 2 2 3 2 3 3 2 1 0 4
[38] 3 2 2 1 4 2 2 4 1 0 3 2 2 3 3 1 3 2 3 3 3 2 2 2 2 4 0 1 5 1 2 3 2 2 5 3 1
[75] 4 4 3 3 4 2 2 2 3 3 3 5 2 3 1 0 1 2 1 1 1 2 2 3 3 2
>
```

2)Розподіл Пуассона.

Знаходження $P\{X \leq 3\}$ для розподілу Пуассона з параметром $\lambda = 8$.

```
> ppois(3,8)
[1] 0.04238011
>
```

3)Геометричний розподіл.

Знаходження $P\{X = 3\}$ для геометричного розподілу з параметром p= 0.2. (Геометричний розподіл визначений як $P\{X = k\} = p(1-p)^k; k = 0,1, ...$).

```
> dgeom(3,0.2)
[1] 0.1024
>
```

4)Гіпергеометричний розподіл.

Знаходження $P\{X = 25\}$для гіпергеометричного розподілу з параметрами m=147, n=3, k=25. (Гіпергеометричний розподіл визначений як $P\{X = i\} = \frac{C_m^i C_n^{k-i}}{C_{n+m}^k}$).

```
> dhyper(25,147,3,25)
[1] 0.576365
>
```

5)Рівномірний розподіл.

Генерування вибірки обсягу 100 з рівномірного розподілу на відрізку [3,5].

```
> runif(100,3,5)
 [1] 4.708462 4.148540 3.911683 4.404110 3.504621 4.503454 4.671398 3.472399
 [9] 4.536314 4.403720 3.065475 3.309524 4.716290 3.498786 4.120027 3.759941
[17] 4.654183 4.446405 3.544601 4.837599 3.681989 4.703181 4.457970 4.703332
[25] 3.778597 3.880072 3.051437 4.037928 4.233277 3.552749 4.269502 4.793736
[33] 3.931118 3.256315 3.589360 3.592465 3.595315 3.102945 4.393103 3.221857
[41] 4.856228 3.928156 3.364604 3.871798 3.445257 4.852213 4.631684 3.255897
[49] 4.443173 4.317664 3.889451 4.081390 3.132968 3.895170 3.065156 3.401125
[57] 3.061146 3.909691 3.145242 3.426145 3.629901 3.327554 3.687854 4.343435
[65] 4.960866 3.510414 4.442051 3.701867 4.907187 3.132439 4.000265 4.573773
[73] 4.877307 3.222933 3.040170 4.706944 4.330527 4.063808 3.446007 3.762330
[81] 3.886572 3.157416 3.938254 4.661562 3.827205 4.479402 3.511800 3.615316
[89] 4.569078 4.572645 3.304311 4.447321 4.354902 3.251734 4.115363 3.007065
[97] 3.309011 4.714925 4.810585 3.534666
```

>

6)Експоненціальний розподіл.

Знаходження квантиля рівня 0.95 експоненціального розподілу з параметром $\lambda = 1/8$.

```
> qexp(0.95,1/8)
[1] 23.96586
>
```

7)Нормальний розподіл.

Знайдемо $P\{X < 27{,}4\}$, де X – нормальна випадкова величина з параметрами $a = 50$, $\sigma = 20$.

```
> pnorm(27.4, mean=50, sd=20)
[1] 0.1292381
```

Або:

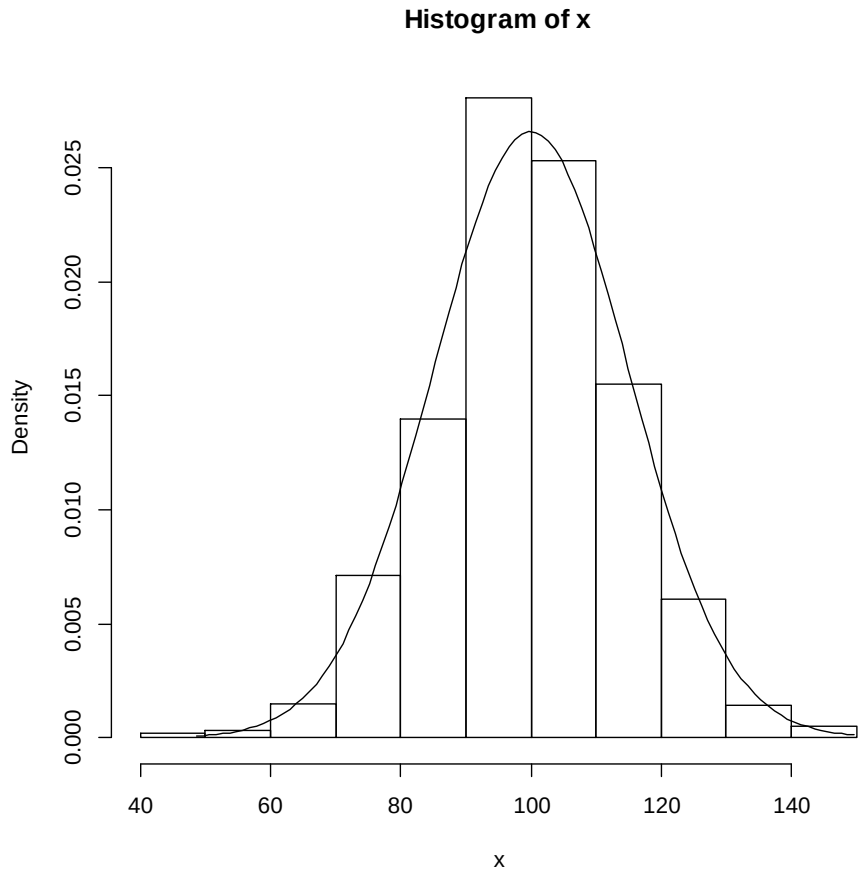
```
> pnorm(27.4, 50, 20)
[1] 0.1292381
```

Знайдемо квантиль рівня 0,95 з нормального розподілу з іншими параметрами.

```
> qnorm(0.95, mean=100, sd=15)
[1] 124.6728
```

Тут ми моделюємо 1000 значень нормальної в.в. з параметрами 100 і 15, будуємо для них гістограму і графік оцінки щільності за даними.

```
> x <- rnorm(1000, mean=100, sd=15)
> hist(x, probability=TRUE)
> xx <- seq(min(x), max(x), length=100)
> lines(xx, dnorm(xx, mean=100, sd=15))
```



8)Гама-розподіл.

Знайдемо $P\{X > 1\}$, де X має гама-розподіл з параметрами $\alpha = 2,\ \lambda = 3$.

> 1-pgamma(1,2,3)

[1] 0.1991483

9)Розподіл χ^2.

Знайдемо $P\{40 \leq \chi^2 \leq 50\}$, де χ^2розподілена з 65 степенями свободи.

> pchisq(50,65)-pchisq(40,65)

[1] 0.07861696

>

10)Розподіл Стьюдента.

Знайдемо значення щільності розподілу Стьюдента з 2 степенями свободи в точці 0,8.

>pt(0.8,1)

[1] 0.7147767

Тут другий параметр обчислений за формулою = кількість степенів свободи -1=2-1=1.

11)Розподіл Фішера.

Знайдемо c, d зі співвідношень $P\{\mathrm{X} < c\} = 0{,}95$ та $P\{\mathrm{X} < d\} = 0{,}05$, де X має розподіл Фішера зі степенями свободи 5, 10.

>c<-qf(0.95,5,10)

>c

[1] 3.325835

>d<-qf(0.05,5,10)

>d

[1] 0.2111904

>

4. Графічний аналіз

Приклад

В файлі Cancer.txt наводяться дані по кількості викурених цигарок на душу населення в 43 регіонах Америки та округу Колумбія за 1960 рік, а також дані по кількості смертей від різних форм раку на 100 000 душ населення. Обсяг вибірки n=44.

Назви змінних:

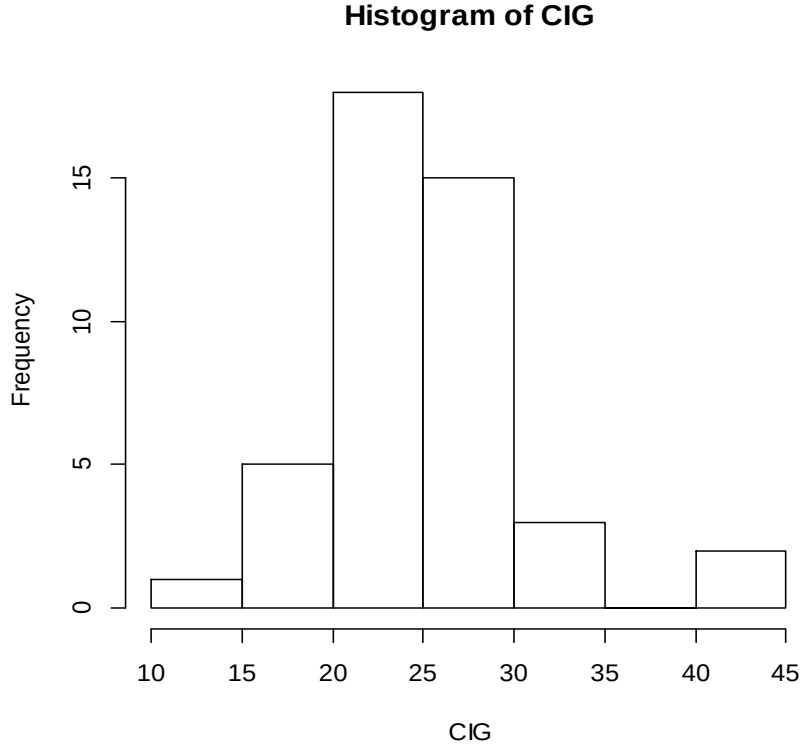
1. CIG = Кількість викурених цигарок (умовних упаковок на душу населення).
2. BLAD = Кількість смертей на 100 тис. населення від раку сечового міхура
3. LUNG = Кількість смертей на 100 тис. населення від раку легенів.
4. KID = Кількість смертей на 100 тис. населення від раку нирки.
5. LEUK = Кількість смертей на 100 тис. населення від лейкемії.

1) Читаємо дані з файлу у фрейм da.

```
> da<-read.table(file.choose(),header=T)
```

2)Побудуємо **гістограму** для змінної CIG. Розбиття на часткові інтервали обирається за замовчанням.

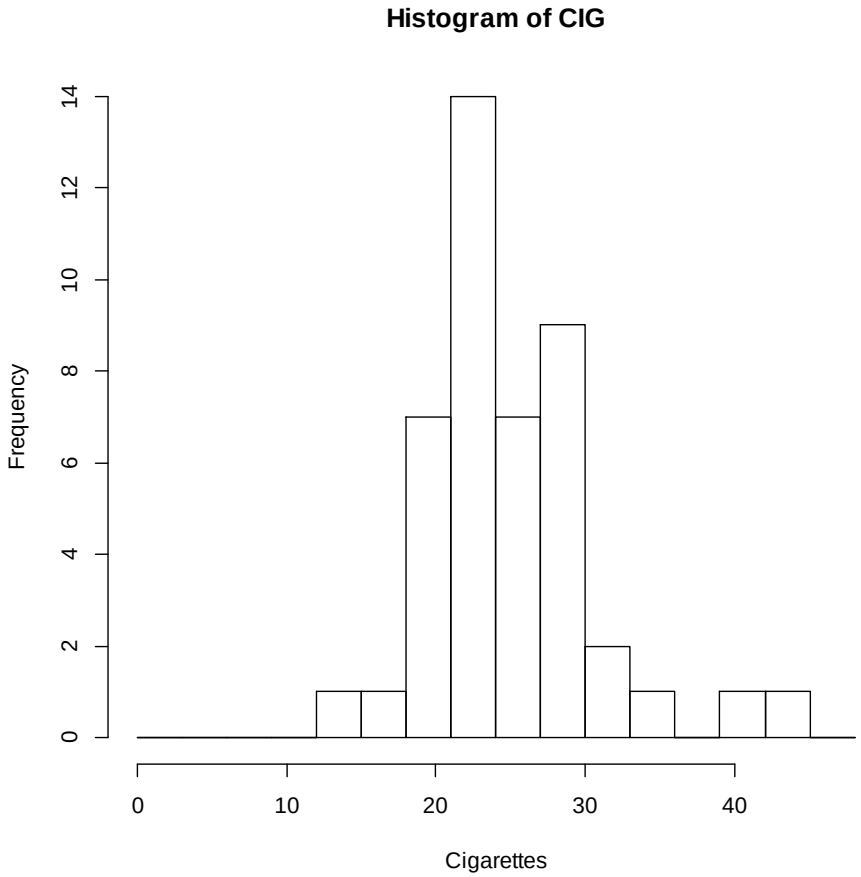
```
> attach(da)
> hist(CIG)
```



3)Якщо ми хочемо самотужки розбити на часткові інтервали наші дані, слід скористатись опцією breaks.

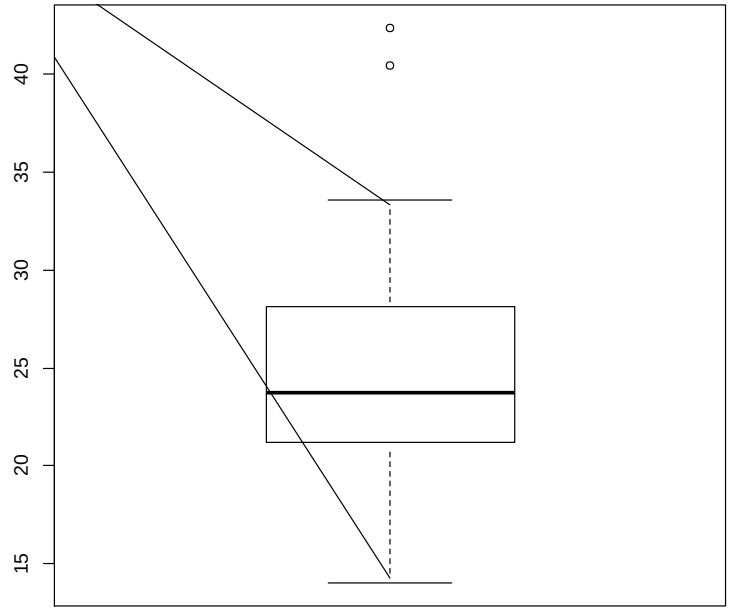
```
>bin<-seq(0,50,3)
>hist(CIG, breaks=bin, xlab="Cigarettes", right=F)
```

Часткові інтервали довжини 3, від 0 до 50. Xlab використовується для надпису на осі OX.



4)Для побудови **вусатої коробочки (boxplot)** можна скористатись оператором

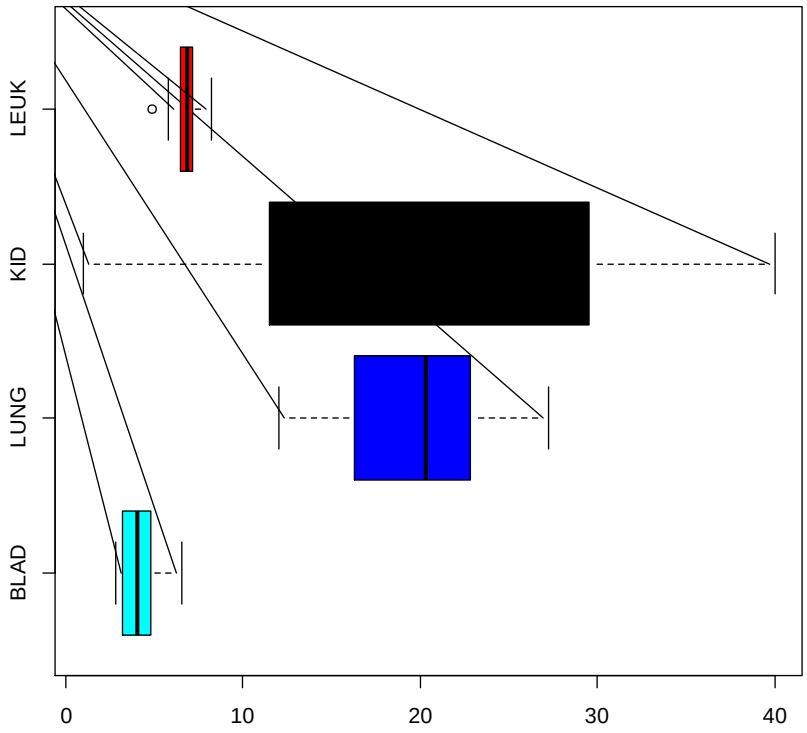
> boxplot(CIG)



В пакеті R вусата коробочка має традиційну структуру. Середня риска коробочки відмічає вибіркову медіану. Прямокутник відмічає нижній (H_l) і верхній (H_u) квартилі. Тобто, H_l – вибірковий квантиль рівня ¼; H_u– вибірковий квантиль рівня ¾. Позначимо ширину прямокутника $H = H_u - H_l$. Тоді відстань від прямокутника до верхнього вуса коробочки становить 1,5 H; в нижній частині коробочки так само 1.5H.

5)Побудуємо 4 коробочки на одному графіку: BLAD, LUNG, KID, LEUK.

>boxplot(BLAD, LUNG,KID,LEUK, horizontal=T, col=c(13,4,1,2), names =c("BLAD", "LUNG","KID","LEUK"))

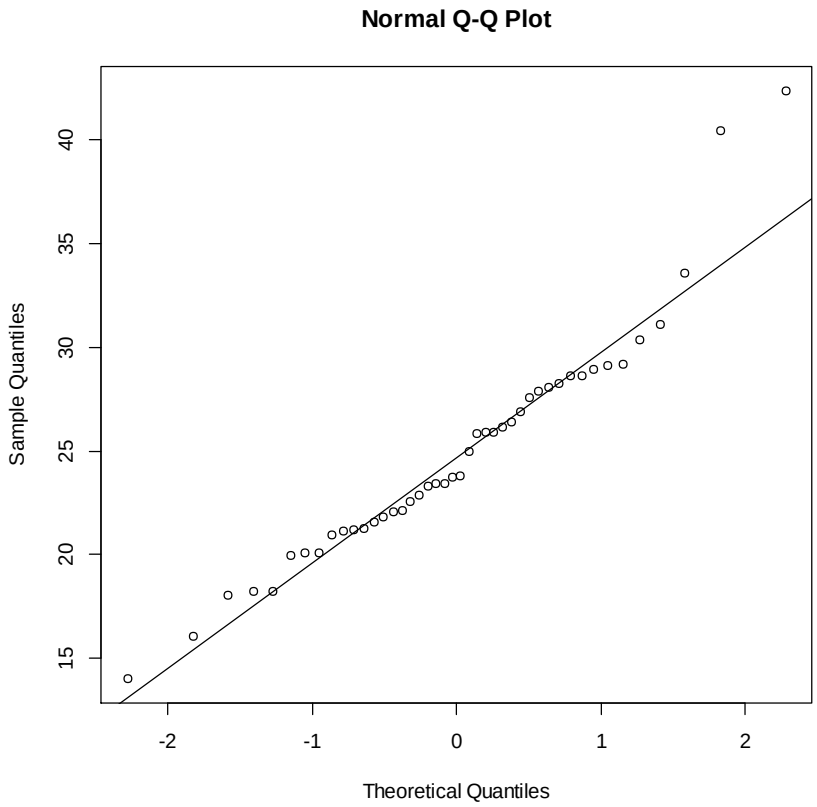


Опція horizontal=T задає горизонтальне положення вусатих коробочок, опція col описує кольори, в які вони пофарбовані, names друкує надписи до коробочок.

6)Побудувати **Q-Q(quantile-quantile) діаграму** на порівняння з нормальним розподілом можна наступним чином

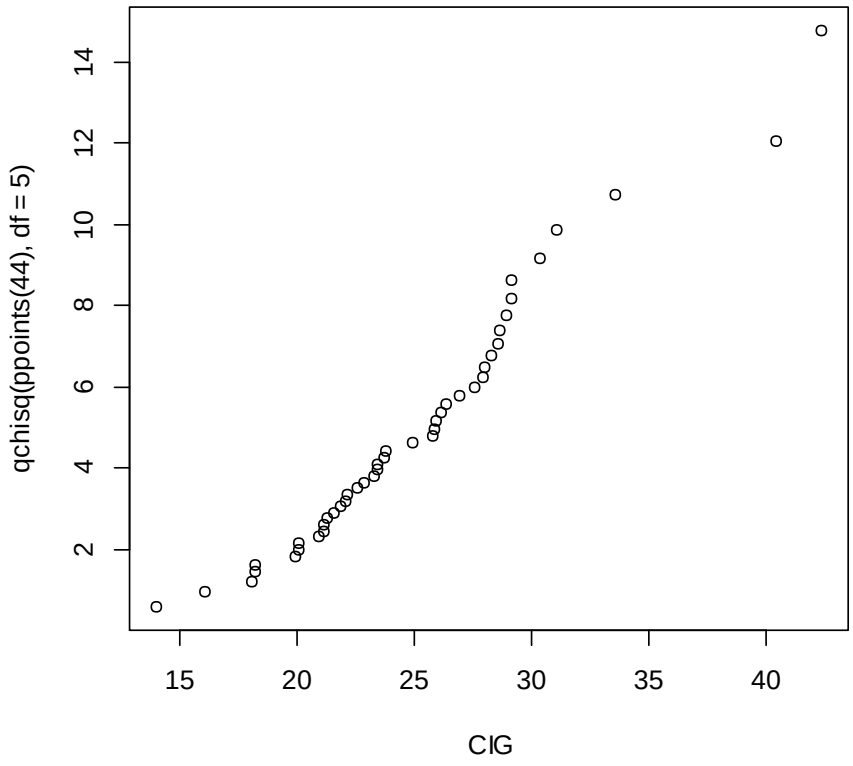
> qqnorm(CIG)
> qqline(CIG)

Друга з команд проводить пряму.



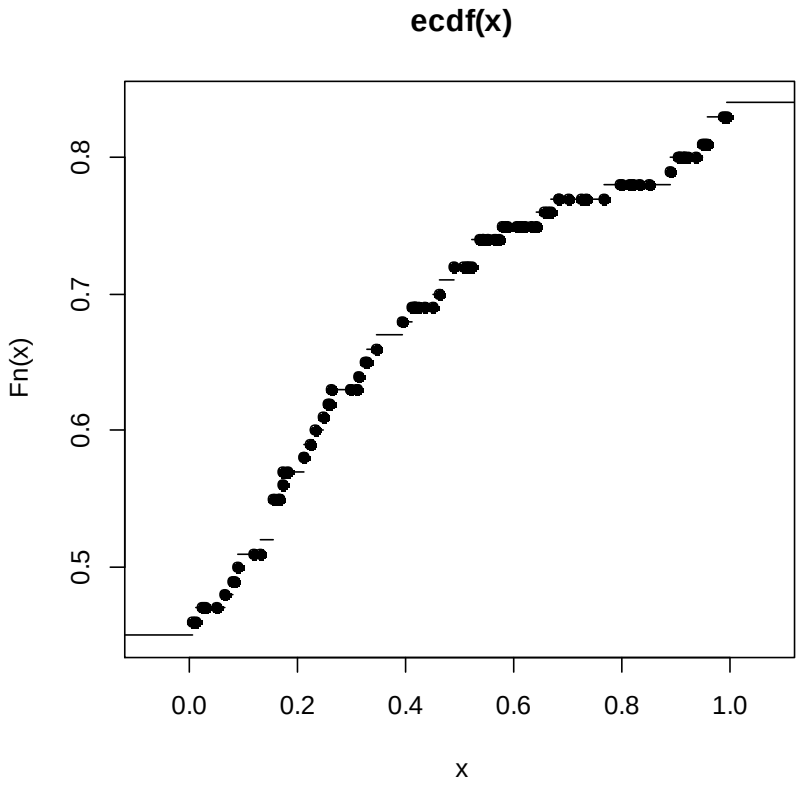
Тепер побудуємо Q-Q-діаграму на порівняння змінної CIG з розподілом χ^2 з п’ятьма степенями свободи. Як ми знаємо, обсяг вибірки =44. Опція ppoints задає послідовність чисел $\frac{i-\frac{1}{2}}{n}$, в даному випадку $n = 44$.

> qqplot(CIG,qchisq(ppoints(44), df = 5))



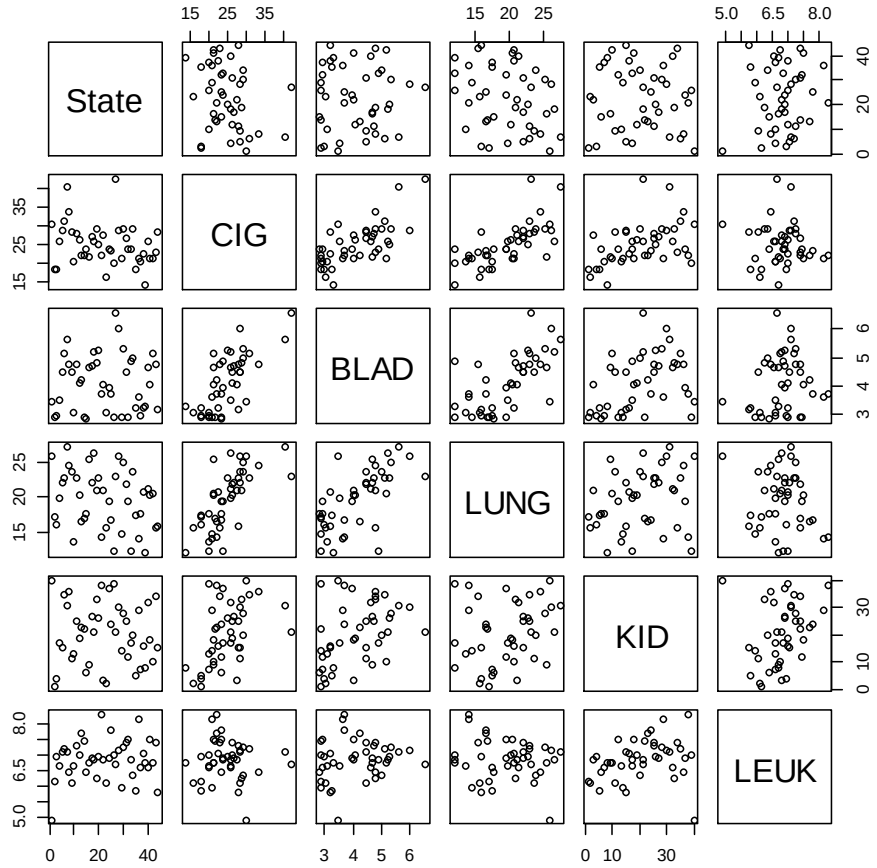
7)Побудувати **P-P(probability-probability plot) діаграму** на порівняння з нормальним розподілом можна наступним чином. Як відомо, абсциса точки на p-p-діаграмі є емпірична функція розподілу, куди замість аргументу підставляємо дані з вибірки, ордината - теоретична функція розподілу, куди так само замість аргументу підставляємо дані з вибірки. Емпіричну функцію розподілу можна побудувати за допомогою функції ecdf( ). В наступному прикладі ми генеруємо вибірку зі стандартним нормальним розподілом і будуємо для неї p-p-діаграму на порівняння з тим самим нормальним розподілом.

```
> x<-rnorm(100,0,1)
> plot(ecdf(x),pnorm(x, mean(x),sd(x)))
```



8) Побудуємо **матричну діаграму** всіх змінних, розміщених в Cancer.txt, тобто, тепер в фреймі da.

> pairs(da)



Схоже, що змінні BLAD та LUNG залежні лінійно від CIG, а от KID та LEUK – не дуже.

Окремо розглянемо функцію **plot( )**. Plot малює різні картинки в залежності від того, якого типу є параметри підпрограми.

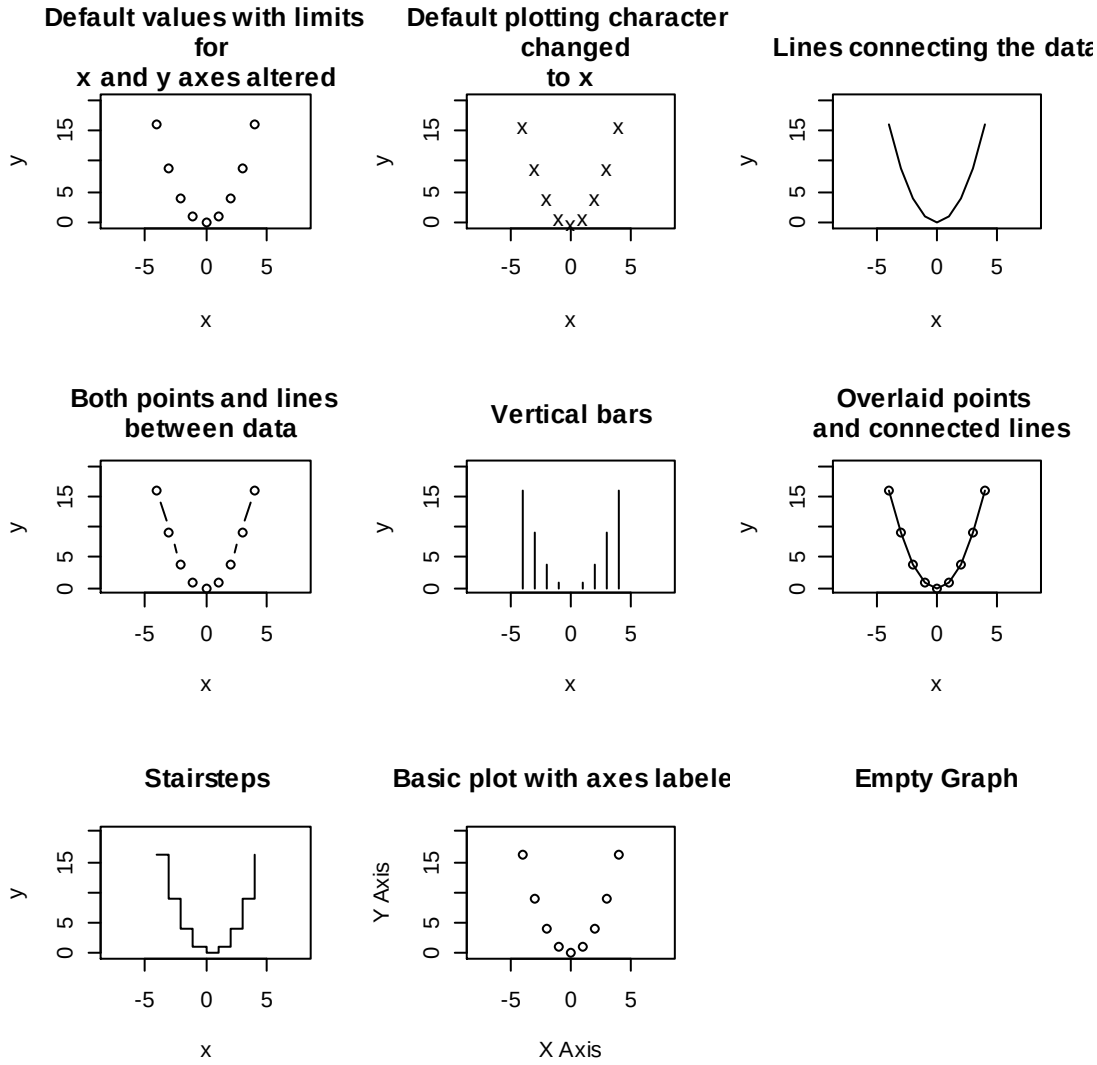
Наступний приклад ілюструє використання різних параметрів цієї функції. Останній графік є порожнім, але інколи корисно створити такий, щоб потім додати необхідні точки, лінії або текст. Перша команда par(mfrow=c(3,3)) розділяє віконце для виводу графіка на 9 частин (структурою в 3 рядка і 3 стовпчики). Існує аналогічна опція mfcol. Символ \n робить перехід на інший рядок в назві.

Опція pty визначає, як використати місце, призначене для графіка: pty=”m” – графік займає максимально можливу площу; pty=“s” означає, що масштаб по осях графіка береться однаковий. Опція type визначає тип лінії. Команда x<- -4:4 задає абсциси точок на графіку. Опції xlim=c(-8,8), ylim=c(0,20) задають відрізки вісей графіка.

Приклад.

```
> par(mfrow=c(3,3), pty="m")
> x<- -4:4
> y<-x^2
> plot(x,y, main="Default values with limits \n for
+ x and y axes altered", xlim=c(-8,8), ylim=c(0,20))
> plot(x,y, pch="x", main="Default plotting character \n changed
+ to x", xlim=c(-8,8), ylim=c(0,20))
> plot(x,y, type="l", main="Lines connecting the data"
+ , xlim=c(-8,8), ylim=c(0,20))
> plot(x,y, type="b", main="Both points and lines \n between data",
+  xlim=c(-8,8), ylim=c(0,20))
```

```
> plot(x,y, type="h", main="Vertical bars",
+  xlim=c(-8,8), ylim=c(0,20))
> plot(x,y, type="o", main="Overlaid points \n and connected lines",
+  xlim=c(-8,8), ylim=c(0,20))
> plot(x,y, type="s", main="Stairsteps",
+  xlim=c(-8,8), ylim=c(0,20))
> plot(x,y, xlab="X Axis", ylab="Y Axis", main="Basic plot with axes labeled",
+  xlim=c(-8,8), ylim=c(0,20))
> plot(x,y, type="n", main="Empty Graph", xlab="", ylab="", axes=FALSE)
>
```



Приклад. Символи.

Команда **points(x,y,…)**додає точки або лінії до поточного графіку.

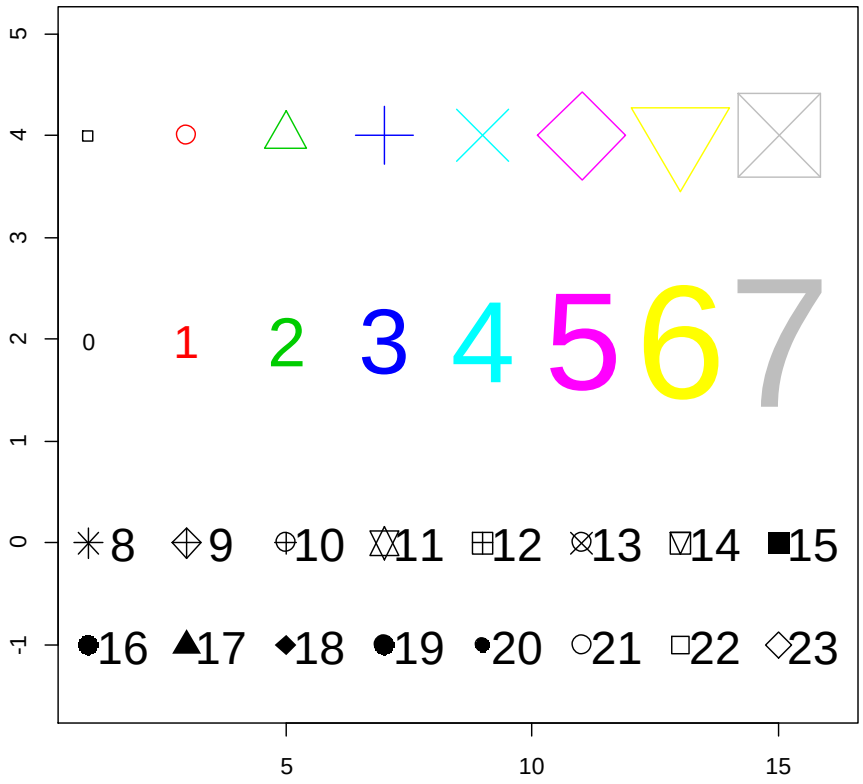
Команда **text(x, y, labels)**малює рядочки, задані у векторі labels в точці, що задається координатами x і y.

cex – параметр, який вказує величину, на яку треба збільшити розмір тексту і символів у порівнянні з деяким взірцем, заданим за умовчанням.

pch- ціле число, що визначає символ або сам символ в лапках, який буде використовуватися на графіку.

```
> plot(1,1, xlim=c(1,16), ylim=c(-1.5,5), type="n", xlab="", ylab="")
> points(seq(1,15,2), rep(4,8), cex=1:8, col=1:8, pch=0:7)
```

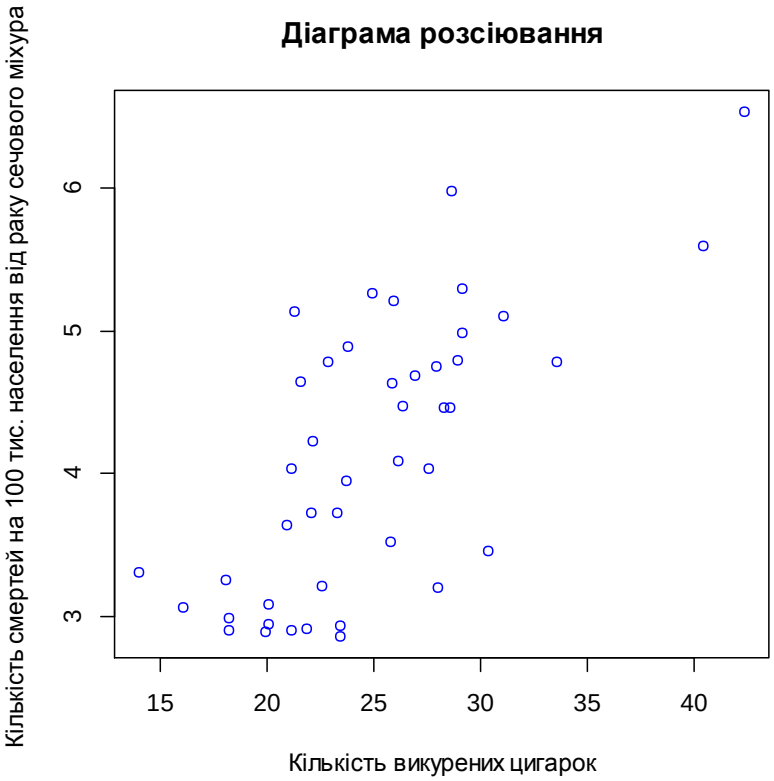
```
> text(seq(1,15,2), rep(2,8), labels=paste(0:7),cex=1:8,col=1:8)
> points(seq(1,15,2), rep(0,8), pch=(8:15), cex=2)
> text(seq(1,15,2)+.7, rep(0,8), paste(8:15),cex=2)
> points(seq(1,15,2), rep(-1,8), pch=(16:23), cex=2)
> text(seq(1,15,2)+.7, rep(-1,8), paste(16:23),cex=2)
>
```



Перша команда створює порожнє віконце, є лише розмітка по осях. Друга друкує різнокольорові символи в першому рядку. Третя надписує цифри – виводить другий рядок. Різноманітні символи першого рядка знаходяться в точній відповідності з цифрами в другому, також символізує характер розміру від 1 до 8. Чотири останні команди виводять інші символи, їх номери написані справа.

Приклад. Повернемось до наших даних про кількість викурених цигарок. Побудуємо діаграму розсіювання даних.

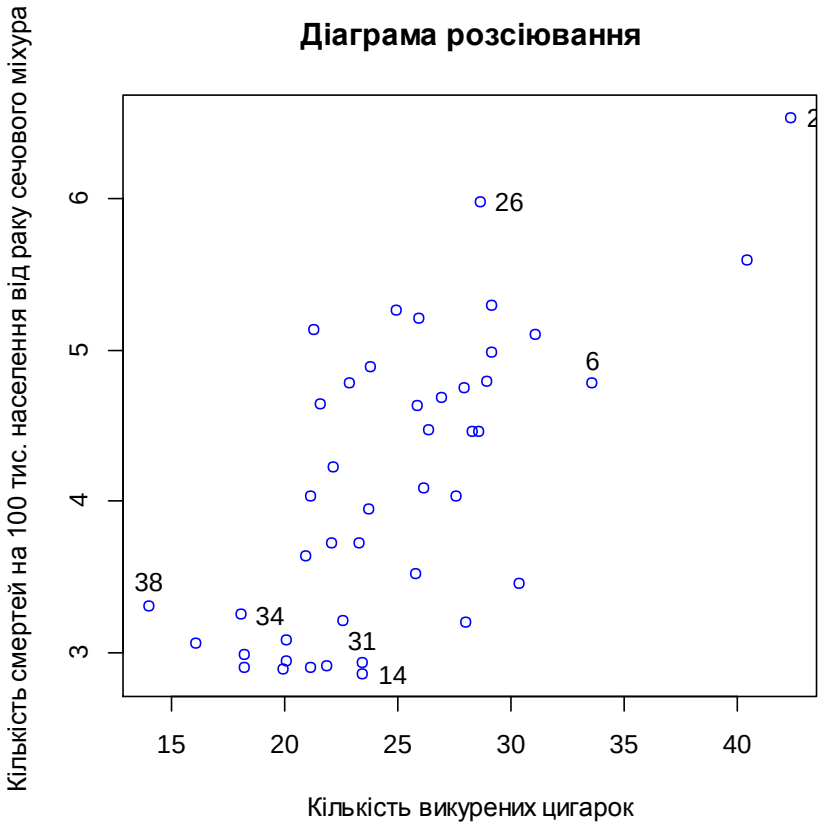
>plot(CIG,BLAD, col=4, main="Діаграма розсіювання",xlab="Кількість викурених цигарок", ylab="Кількість смертей на 100 тис. населення від раку сечового міхура ")



Функція identify() використовується, щоб дати змогу користувачеві вибрати на рисунку цікаві для нього експериментальні точки.

> identify(CIG, BLAD, labels=row.names(da))

Після виконання даної команди на графіку курсор перетворюється на хрестик, за допомогою кліка мишки відмічаємо необхідні точки (наприклад, викиди), і поряд з ними з'являються надписи – в нашому прикладі це лише номери спостережень.



5. Проста лінійна регресія

Специфікація моделі: $Y_i = a + b * X_i + \varepsilon_i; \quad i = 1, \dots, n.$

Тут X_i – незалежна змінна, регресор; Y_i - залежна; ε_i– похибка, н.о.р.в.в. .$E\varepsilon_i = 0;\ D\varepsilon_i = \sigma^2.$

Приклад.

Дані є результатами дослідження методу газової хроматографії, за допомогою якого виявляють субстанції дуже малої концентрації. Було зроблено по 5 вимірювань для кожного з різних зразків, які містили дуже малі об'єми субстанції. В кожному зразку об'єм субстанції був визначений до експерименту. Величиною - відгуком є дані, зчитані з газового хроматографа. Метою наших досліджень є калібрування хроматографа для визначення справжніх об'ємів субстанції. Для цього необхідно побудувати модель простої регресії і оцініти її якість, а також проаналізувати (візуально) залишки.

Назви змінних:

1. amount: об'єм субстанції у зразку (в нанограмах)
2. response: величина, зчитана з газового хроматографа

Дані

amount	response
0.25	6.55
0.25	7.98
0.25	6.54
0.25	6.37
0.25	7.96
1.00	29.7
1.00	30.0
1.00	30.1
1.00	29.5
1.00	29.1
5.00	211
5.00	204
5.00	212
5.00	213
5.00	205
20.00	929
20.00	905
20.00	922
20.00	928
20.00	919

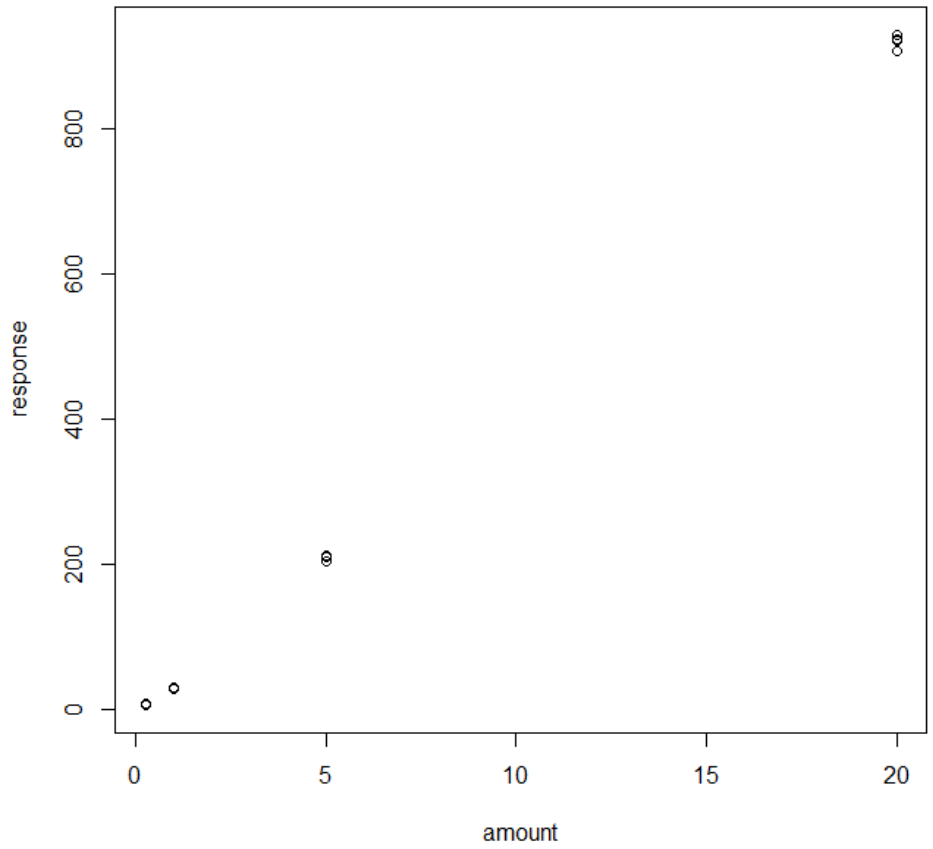
Розв'язок.

1)Заносимо дані в два одновимірні масиви.

```
> amount<-c(0.25, 0.25, 0.25, 0.25, 0.25,1 ,1, 1, 1, 1, 5, 5, 5, 5, 5, 20, 20, 20, 20,20)
> response<-c(6.55, 7.98, 6.54, 6.37, 7.96, 29.7, 30, 30.1, 29.5, 29.1, 211, 204, 212, 213, 205, 929, 905, 922, 928, 919)
```

2)Будуємо **діаграму розсіювання даних** для змінних amount і response.

```
> plot(amount,response)
```



3)Викликаємо функцію для обчислення коефіцієнтів **простої лінійної регресії і її аналізу** response на amount, результат записується у змінну z.

>z<-lm(response~amount)

4)Запитуємо про результат.

> summary(z)

Call:
lm(formula = response ~ amount)

Residuals:
   Min   1Q Median   3Q   Max
-14.733 -5.983 -2.168   9.296 10.837

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -14.4107   2.6142 -5.512 3.11e-05 ***
amount   46.6287   0.2533 184.086 < 2e-16 ***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.023 on 18 degrees of freedom
Multiple R-squared: 0.9995,   Adjusted R-squared: 0.9994
F-statistic: 3.389e+04 on 1 and 18 DF, p-value: < 2.2e-16

5)Відкоментуємо результат. Залежність має вигляд

$$response = -14{,}4107 + 46{,}6287amount.$$

Коефіцієнт детермінації і скоригований коефіцієнт детермінації близькі до 1: $R^2 = 0{,}9995$; $Adjusted\ R^2 = 0{,}9994$, це показує, що прогноз за даною формулою є хорошим.

При перевірці гіпотези $\{b = 0\}$ за критерієм Фішера досягнутий рівень значущості $p < 2{,}2 * 10^{-16}$. Гіпотезу відхиляємо при рівні значущості $\alpha = 0{,}05$. Отже, amount – значуща змінна.

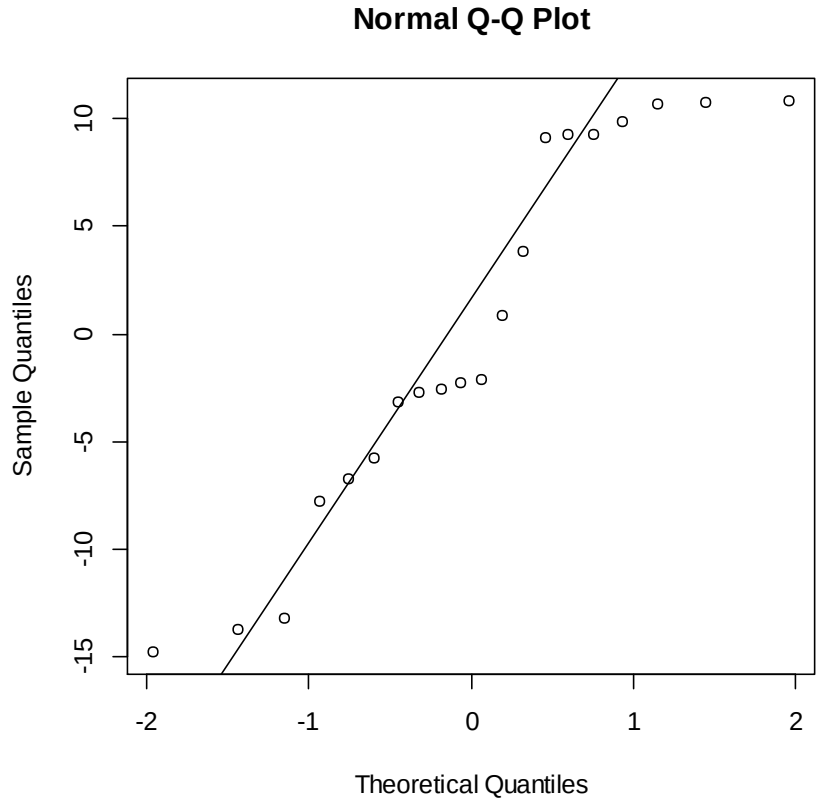
Хроматограф відкалібрований непогано ☺.

Зауважимо, що якщо б нам знадобилось побудувати модель, де відсутній вільний член, тобто вигляду $response = b \cdot amount + \varepsilon$, функцію lm слід було б задати з аргументом $amount - 1$:

```
> z<-lm(response~amount-1)
> summary(z)

Call:
lm(formula = response ~ amount - 1)

Residuals:
   Min     1Q Median     3Q    Max
-24.70 -16.34 -12.73  -3.47  14.18

Coefficients:
      Estimate Std. Error t value Pr(>|t|)
amount   45.741      0.312   146.6   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.4 on 19 degrees of freedom
Multiple R-squared: 0.9991,     Adjusted R-squared: 0.9991
F-statistic: 2.15e+04 on 1 and 19 DF,  p-value: < 2.2e-16
```

6)Насамкінець будуємо q-q діаграмму залишків на порівняння з нормальним розподілом. Т.ч. перевіряємо: чи є гіпотеза про нормальність похибок в моделі справді вірогідною.

```
> qqnorm(z$residuals)
> qqline(z$residuals)
```



Аналогічний графік можна побудувати, скориставшись командою plot(z). В робочому полі з'являється напис «Ожидаю подтверждения смены страницы...». Система видає одразу декілька графіків, переключати вперед їх можна за допомогою мишки. Другий графік буде "Normal Q-Q plot".

Бачимо, що не дуже ідеальна підгонка. Але причиною такого графіка може бути те, що у нас нестандартна вибірка – кожне значення регресора зустрічається по декілька разів.

6. Модель множинної регресії

Специфікація моделі: $Y_i = b_1 + b_2 * X_{i1} + b_3 * X_{i3} + \cdots + b_k * X_{ik} + \varepsilon_i; \quad i = 1, \dots, n.$

Тут X_{ik} – незалежні змінні, регресори; Y_i - залежна; ε_i– похибки, н.о.р.в.в. $E\varepsilon_i = 0;\ D\varepsilon_i = \sigma^2$.

Приклад.

Під час виготовлення сиру відбувається багато хімічних процесів, які в подальшому визначають кінцевий смак продукту. Наводимо дані про концентрацію різних хімічних елементів в 30 голівках витриманого сиру «Чеддер», а також оцінка смаку кожного такого шматка (оцінка смаку є суто суб'єктивною характеристикою). Треба дослідити залежність смаку сира від кількості оцтової кислоти, сірководню та молочної кислоти.

Назви змінних:

1. Case: Номер елементу вибірки
2. Taste: Оцінка смаку (суб'єктивна характеристика), отримана як середнє арифметичне оцінок кількох дегустаторів.
3. Acetic: Натуральний логарифм від величини концентрації оцтової кислоти.
4. H2S: Натуральний логарифм від величини концентрації сірководню.
5. Lactic: Концентрація молочної кислоти.

Розв’язок.

1)Заносимо дані в текстовий файл cheese.txt. Для зчитування даних з файлу в матрицю da застосовуємо команду read.table( ).

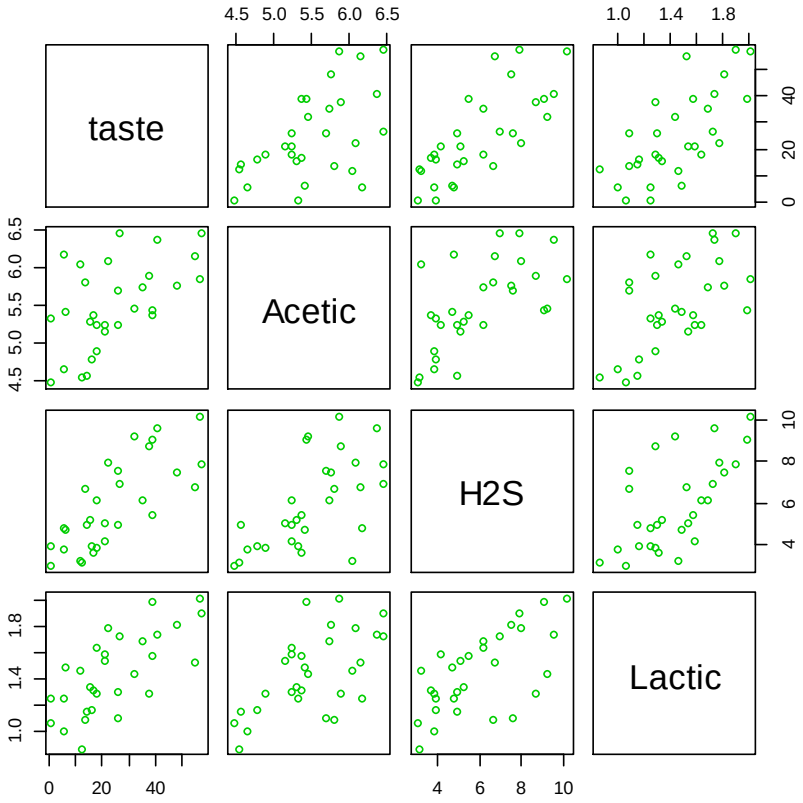
```
> da<-read.table(file.choose(),header=TRUE)
```

Подивимось, що ми зчитали:

```
> da
   taste Acetic    H2S Lactic
1   12.3  4.543  3.135   0.86
2   20.9  5.159  5.043   1.53
3   39.0  5.366  5.438   1.57
4   47.9  5.759  7.496   1.81
5    5.6  4.663  3.807   0.99
6   25.9  5.697  7.601   1.09
7   37.3  5.892  8.726   1.29
8   21.9  6.078  7.966   1.78
9   18.1  4.898  3.850   1.29
10  21.0  5.242  4.174   1.58
11  34.9  5.740  6.142   1.68
12  57.2  6.446  7.908   1.90
13   0.7  4.477  2.996   1.06
14  25.9  5.236  4.942   1.30
15  54.9  6.151  6.752   1.52
16  40.9  6.365  9.588   1.74
17  15.9  4.787  3.912   1.16
18   6.4  5.412  4.700   1.49
19  18.0  5.247  6.174   1.63
20  38.9  5.438  9.064   1.99
21  14.0  4.564  4.949   1.15
22  15.2  5.298  5.220   1.33
23  32.0  5.455  9.242   1.44
24  56.7  5.855 10.199   2.01
25  16.8  5.366  3.664   1.31
26  11.6  6.043  3.219   1.46
27  26.5  6.458  6.962   1.72
28   0.7  5.328  3.912   1.25
29  13.4  5.802  6.685   1.08
30   5.5  6.176  4.787   1.25
```

2)Побудуємо матричну діаграму даних.

```
> pairs(da,col=3)
```



В принципі, спостерігається лінійна залежність taste від інших трьох регресорів, особливо від H2S і Lactic.

3)Запускаємо програму для побудови моделі множинної регресії, де залежною змінною є taste, а незалежними - Acetic, H2S, Lactic. Потім дивимось на summary.

```
> z<-lm(taste~Acetic+H2S+Lactic,data=da)
> summary(z)

Call:
lm(formula = taste ~ Acetic + H2S + Lactic, data = da)

Residuals:
   Min     1Q  Median     3Q    Max
-17.390  -6.612  -1.009   4.908  25.449

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -28.8768    19.7354  -1.463  0.15540
Acetic        0.3277     4.4598   0.073  0.94198
H2S           3.9118     1.2484   3.133  0.00425 **
Lactic       19.6705     8.6291   2.280  0.03108 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.13 on 26 degrees of freedom
Multiple R-squared: 0.6518,     Adjusted R-squared: 0.6116
F-statistic: 16.22 on 3 and 26 DF,  p-value: 3.81e-06
```

4) Аналіз результатів. Отже, модель має вигляд

$$Taste = -28{,}8768 + 0{,}3277 * Acetic + 3{,}9118 * H2S + 19{,}6705 * Lactic.$$

Коефіцієнт детермінації і скоригований коефіцієнт детермінації: $R^2 = 0{,}6518;\ Adjusted\ R^2 = 0{,}6116$ - прогноз за формулою регресії є так собі.

При перевірці гіпотези $\{b_2 = b_3 = b_4 = 0\}$ за критерієм Фішера досягнутий рівень значущості $p = 3{,}81 * 10^{-6}$. Гіпотезу відхиляємо при рівні значущості $\alpha = 0{,}05$, оскільки $p < \alpha$. Отже, модель є значущою.

При перевірці за критерієм Фішера гіпотези $\{b_2 = 0\}$ досягнутий рівень значущості дорівнює $p = 0{,}94198$. Отже, гіпотезу приймаємо, і змінна Acetic є незначущою.

Аналогічно бачимо, що змінні H2S $(p = 0{,}00425)$ та Lactic $(p = 0{,}03108)$ при $\alpha = 0{,}05$ є значущими.

5)Побудуємо іншу регресійну модель, вилучивши регресор Acetic.

```
> z<-lm(taste~H2S+Lactic,data=da)
> summary(z)

Call:
lm(formula = taste ~ H2S + Lactic, data = da)

Residuals:
   Min     1Q  Median     3Q    Max
-17.343  -6.530  -1.164   4.844  25.618

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -27.592      8.982  -3.072  0.00481 **
H2S            3.946      1.136   3.475  0.00174 **
Lactic        19.887      7.959   2.499  0.01885 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.942 on 27 degrees of freedom
Multiple R-squared: 0.6517,     Adjusted R-squared: 0.6259
F-statistic: 25.26 on 2 and 27 DF,  p-value: 6.551e-07
```

Нова модель має вигляд

$$Taste = -27{,}592 + 3{,}946 * H2S + 19{,}887 * Lactic.$$

Коефіцієнт детермінації і скоригований коефіцієнт детермінації: $R^2 = 0{,}6517;\ Adjusted\ R^2 = 0{,}6259$. Коефіцієнт детермінації через відкидання регресора зменшився лише на 0,0001, а скоригований коефіцієнт детермінації навіть збільшився.

При перевірці гіпотези $\{b_2 = b_3 = 0\}$ за критерієм Фішера досягнутий рівень значущості $p = 6{,}55 * 10^{-7}$. Модель є значущою.

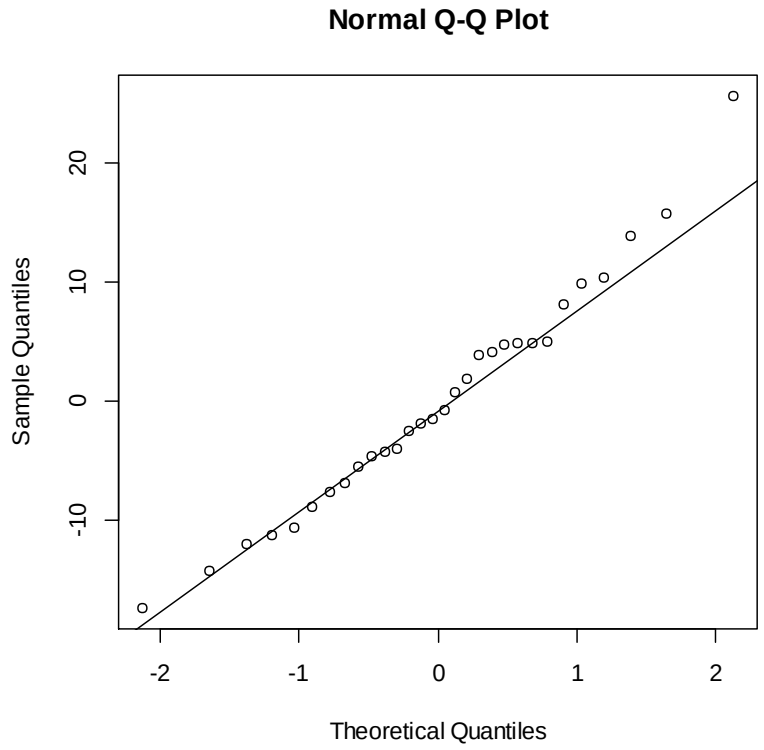
Обидві змінні H2S $(p = 0{,}00174)$ та Lactic $(p = 0{,}01885)$ є значущими.

Очевидно, що друга модель є кращою.

6)Аналіз залишків. Будуємо q-q-діаграмму на порівняння залишків з нормальним розподілом.

```
> qqnorm(z$residuals)
```

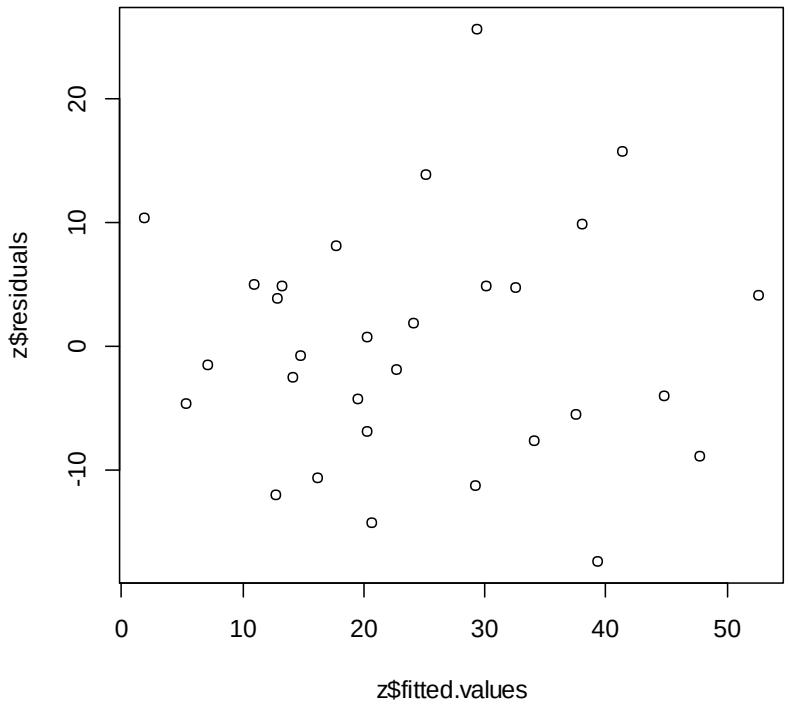
> qqline(z$residuals)



Схоже на те, що залишки справді нормальні.

А тепер будуємо просто графік залишків – залежність між залишками і передбаченими регресією значеннями taste (називаються z$fitted.values).

> plot(z$fitted.values,z$residuals)



Бачимо, що залежність відсутня, і це добре характеризує нашу модель.

7. Нелінійна регресія

Як відомо, в рамках методу найменших квадратів існують 2 підходи до підгонки даних нелінійною залежністю.

Перший з цих підходів запроваджує наступну специфікацію моделі, з адитивною похибкою:

$$Y_i = g(\bar{X}_i, \bar{b}) + \varepsilon_i; \quad i = 1, \dots, n.$$

Тут $g(\,)$ – нелінійна функція; $\bar{X}_i$ – вектор незалежних змінних, регресорів ;$\bar{b}$ – вектор коефіцієнтів, які підлягають оцінці; Y_i – залежна змінна; ε_i– похибки, н.о.р.в.в. $E\varepsilon_i = 0$; $D\varepsilon_i = \sigma^2$.

Задача зводиться до пошуку **мінімуму функціонала найменших квадратів**:

$$J(\bar{b}) = \sum_{i=1}^{n} \left(Y_i - g(\bar{X}_i, \bar{b})\right)^2 \to min.$$

Тоді шукані оцінки параметрів

$$\hat{\bar{b}} = argmin\, J(\bar{b}).$$

Такі оцінки пакет будує, використовуючи наближені чисельні методи розв'язку.

Другий підхід полягає в лінеаризації моделі, тобто в підборі такого перетворення змінних, щоб залежність стала лінійною.

Подивимось на прикладах, як в пакеті R реалізуються такі підходи.

Приклад. Крива Гомперця.

Досліджуючи плодових мух-дрозофіл в 1825 році Гомперц висунув теорію, що рівень смертності зростає експоненціально з віком (тобто організм стає старіший і можливість померти за одиницю часу зростає експоненційно). Пізніше це припущення було підтверджене, випробуване часом, і відповідна залежність застосовується в актуарій математиці під назвою «Крива Гомперця».

Популяцію для експерименту взяли у вигляді 1 203646 плодових мух, що народились в один день. Щодня фіксувалась кількість комах, які були знайдені мертвими. Спостереження велися на протязі 171 дня.

Нам треба побудувати нелінійну регресійну модель залежності кількості мух, які вижили, від часу двома способами : за допомогою підгонки в пакеті R та за допомогою лінеаризації . Залежність задати у вигляді кривої Гомперця: $living = 1203646g^{-a^t}$, де g, a - сталі, які треба знайти; t - day.

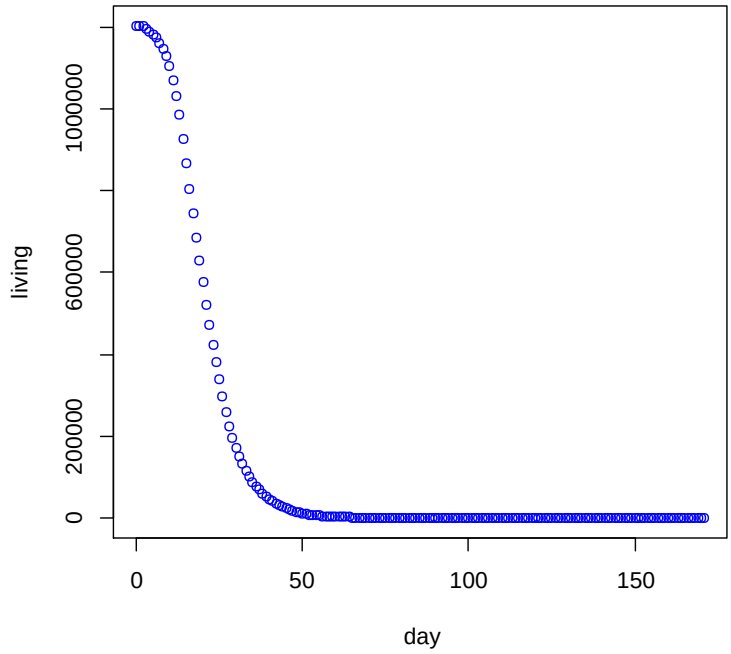
Назви змінних:

1. day: Номер дня.
2. living: Кількість мух, які залишились живими на початку дня.

Розв'язок.

1)Зчитуємо дані, які знаходяться у файлі Gomperts.txt. Будуємо діаграму розсіювання даних

```
> da<-read.table(file.choose(),header=TRUE)
> attach(da)
> plot(day,living,col=4)
```



Справді схоже, що існує чітка нелінійна залежність між змінними.

2)Спершу застосуємо нелінійний МНК. Будуємо модель, що має адитивні похибки.

$$living = 1203646(g)^{-a^{day}} + \varepsilon.$$

Для застосування підпрограми знаходження оцінок в моделі нелінійної регресії підключаємо бібліотеку nls2. Звертаємось до підпрограми nls (nonlinear least of squares), вказуючи в аргументах залежну змінну, формулу залежності (в нашому випадку $living = 1203646g^{-a^{day}}$), в опції start в списку задаються початкові ітерації для оцінки параметрів a, g, trace=T – опція виводу, означає, що всі ітерації виводяться на друк.

```
> library(nls2)
> z<-nls(living~1203646*g^(-a^day),start=list(g=1.5,a=1.5),trace=T)
1.913763e+13 :  1.5 1.5
1.349661e+13 :  1.445757 1.189187
6.836775e+12 :  1.397101 1.102163
676186298112 :  1.147246 1.084359
440069298576 :  1.076817 1.104485
260924784735 :  1.047890 1.129196
168093153721 :  1.045908 1.138757
160180628191 :  1.039543 1.147149
158524385951 :  1.039211 1.148720
158408075786 :  1.038568 1.149748
158398495034 :  1.038472 1.149952
158397785709 :  1.038431 1.150024
158397732899 :  1.038422 1.150041
```

158397728986 : 1.038419 1.150046

158397728698 : 1.038418 1.150048

158397728677 : 1.038418 1.150048

3)Потім дивимось на результат.

> summary(z)

Formula: living ~ 1203646 * g^(-a^day)

Parameters:

Estimate Std. Error t value Pr(>|t|)

g 1.038418 0.002953 351.7 <2e-16 ***

a 1.150048 0.003891 295.5 <2e-16 ***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 30520 on 170 degrees of freedom

Number of iterations to convergence: 15

Achieved convergence tolerance: 2.818e-06

Отже, наша модель з оціненими параметрами має вигляд

$$living = 1203646(1{,}038)^{-1{,}15^{day}}.$$

Значущість кожного параметру перевіряється за критерієм Стьюдента, і за значеннями p-level (4-й стовпчик таблиці) робимо висновок, що обидва параметри є значущими.

4)Знайдемо відсоток поясненої дисперсії, який обчислюється за формулою

$$VAR.EXP = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum\left(Y - \hat{Y}\right)^2}{\sum(Y - \bar{Y})^2}$$

> RSS<-sum((living-fitted.values(z))^2)

> RSS

[1] 158397728679

> TSS<-sum((living-mean(living))^2)

> TSS

[1] 1.98068e+13

> VAR.EXP<-1-RSS/TSS

> VAR.EXP

[1] 0.9920029

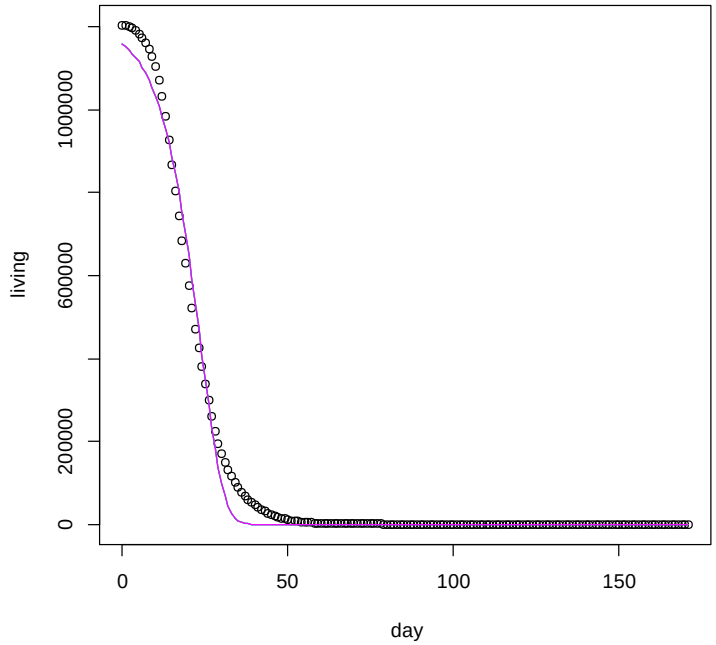
>

Відсоток поясненої дисперсії дуже хороший – 99,2%.

5)Побудуємо на діаграмі розсіювання даних підгоночну криву.

> plot(day,living)

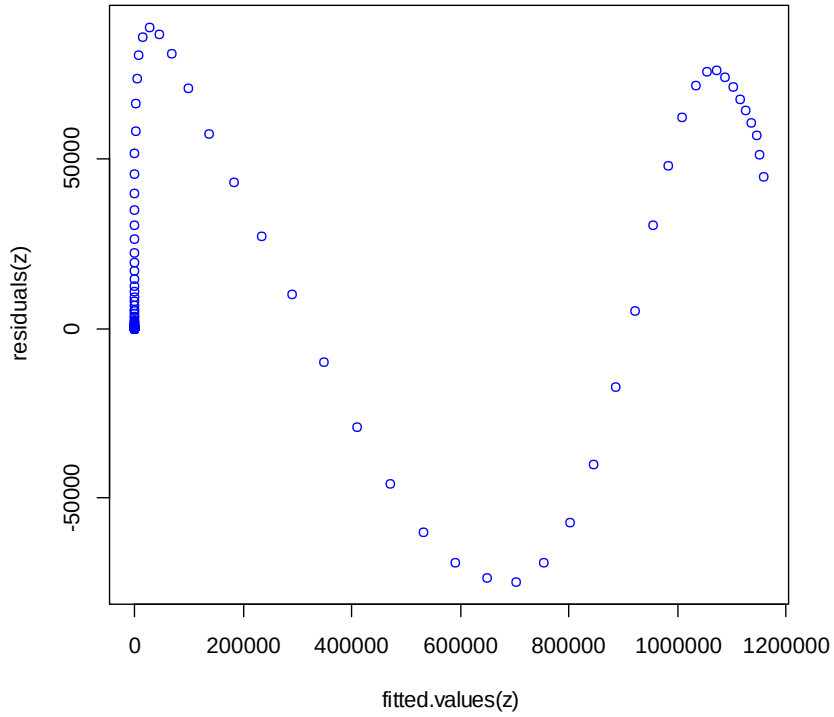
> lines(day,fitted.values(z),col=6)



Здається, підгонка все ж не така ідеальна, як хотілося б.

6)Побудуємо графік залишків “Predicted versus Residuals”.

> plot(fitted.values(z),residuals(z),col=4)



В розташуванні залишків є певна закономірність. Це означає, що наша модель не ідеальна, можна знайти кращу залежність між залежною і незалежною змінними.

7)Тепер застосуємо метод лінеаризації. Прологарифмуємо формулу залежності

$$living = 1203646(g)^{-a^{day}}$$

два рази. Після трансформації отримуємо лінійну модель:

$$Y^*{}_i = k + bX^*{}_i + \varepsilon_i; \quad i = 1, \dots, n,$$

де $Y^* = \ln\left(ln\left(\frac{1203646}{living}\right)\right); \quad X^* = day; \quad k = lnlng; \quad b = lna.$

8)Тепер можна застосувати підпрограму лінійної регресії. Проблема лише в тому, що після такого перетворення для першого і другого спостереження $\ln\left(ln\left(\frac{1203646}{living}\right)\right) = -\infty.$ Тому відкидаємо їх.

```
> y1<-living[3:171]
> y1<-log(log(1203646/y1))
> day1<-day[3:171]
> z<-lm(y1~day1)
> summary(z)
Call:
lm(formula = y1 ~ day1)
Residuals:
    Min      1Q  Median      3Q     Max
-5.9375 -0.5022  0.2445  0.8180  1.0100
```

Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-0.654943	0.170752	-3.836	0.000177	***
day1	0.024935	0.001727	14.439	< 2e-16	***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

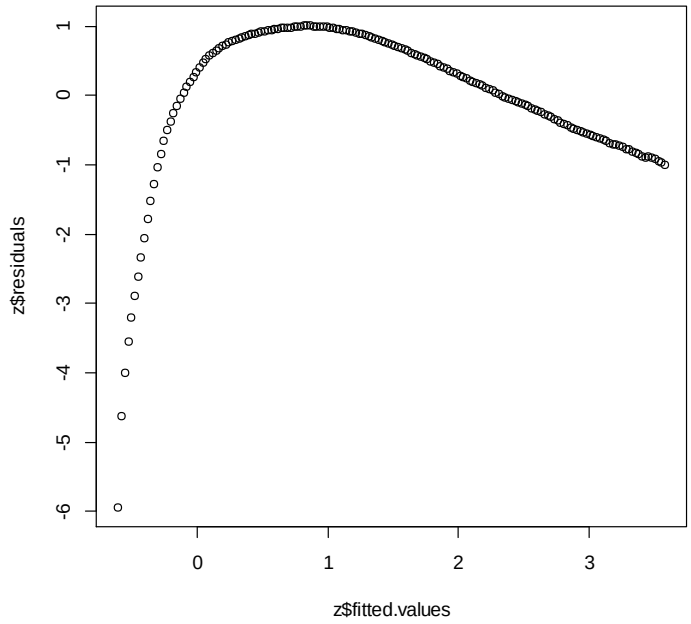
Residual standard error: 1.095 on 167 degrees of freedom

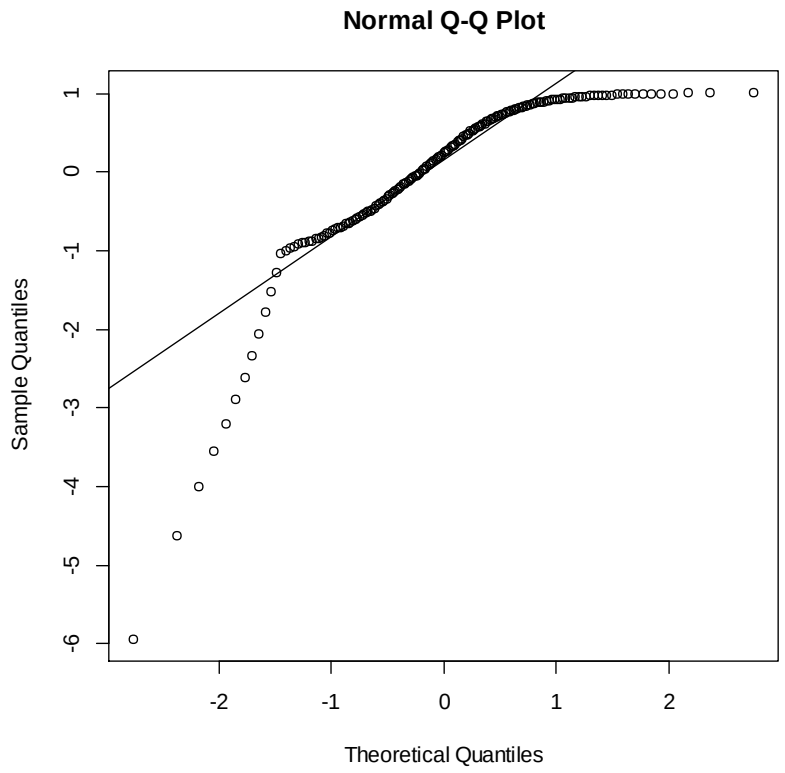
Multiple R-squared: 0.5552, Adjusted R-squared: 0.5526

F-statistic: 208.5 on 1 and 167 DF, p-value: < 2.2e-16

9)Таке враження, що ця модель гірша, ніж розглянута вище. Змінні є значущими, а от коефіцієнт детермінації дорівнює лише 0.555.

Побудуємо графік залишків і q-q-діаграму на порівняння з нормальним розподілом.





Графіки показують не надто високу якість моделі.

10)Випишемо коефіцієнти моделі, зробимо обернені перетворення.

> g<-exp(exp(-0.654943))

> g

[1] 1.681139

> a<-exp(0.024935)

> a

[1] 1.025248

>

Модель має вигляд:

$$living = 1203646(1.6811)^{-1.025^{day}e^{\varepsilon}}$$

Похибку в формулі важко проінтерпретувати. Як ми бачимо, оцінки параметрів відрізняються в двох побудованих моделях.

Висновок: для прогнозу доцільно використовувати першу модель

$$living = 1203646(1{,}038)^{-1{,}15^{day}}.$$

Приклад. Поліноміальна регресія.

Дані з файлу LTemp.txt містять середню мінімальну температуру січня в градусах Фаренгейта разом з географічною довготою і широтою 56-ти міст Сполучених Штатів (n=56). (Мається на увазі, що для кожного року з 1931 по 1960 щоденні мінімальні температури січня були додані разом і поділені на 31. Потім ці середні для кожного року були усереднені по 30 роках.)

Треба дослідити залежність температури від широти та довготи. Щодо вигляду залежності маємо такі теоретичні міркування. Від широти середня температура січня повинна залежати лінійно – чим далі на північ, тим холодніше. Щодо довготи – то інша справа. Америка справа і зліва обмивається океаном, і через його вплив мінімальна температура січня на узбережжі вища. Тому слід очікувати нелінійну залежність JanTemp від довготи.

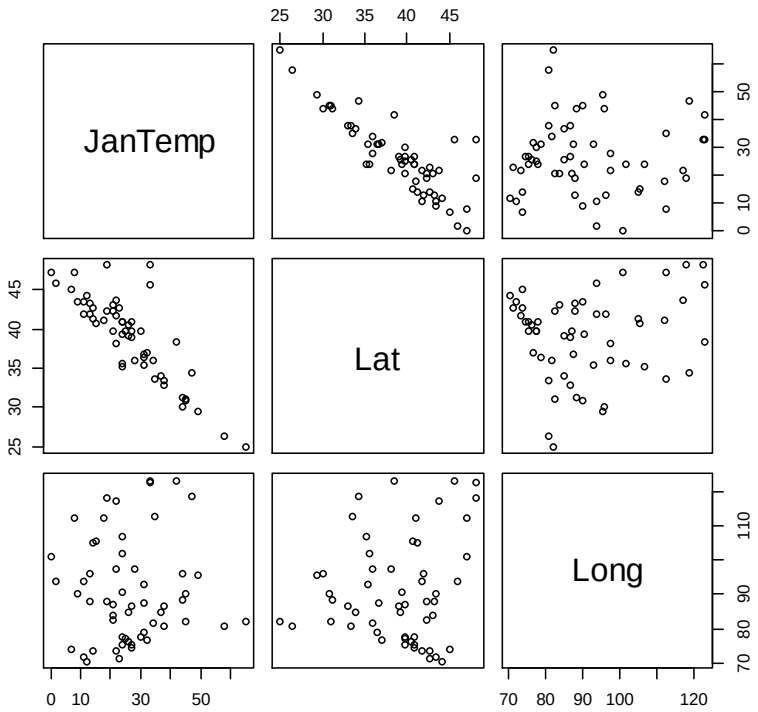
Назви змінних:

1. City: Назва міста
2. JanTemp: Середня мінімальна температура січня в градусах Фаренгейта за 1931-1960 рр.
3. Lat: Географічна широта в градусах (на північ від екватора).
4. Long: Географічна довгота в градусах (на захід від нульового меридіана)

Розв'язок.

1)Зчитуємо файл з даними Temp.txt. Будуємо матричну діаграму розсіювання.

```
>da<-read.table(file.choose(),header=TRUE)
> attach(da)
> pairs(da[-1])
```



Судячи з діаграми розсіювання даних, зв'язок між JanTempта регресорами сильний. Від змінної Lat спостерігається лінійна залежність, від Long – складніша, спробуємо підібрати залежність у вигляді полінома.

2)Розглянемо поліноміальну модель вигляду

$$JanTemp_i = b1 + b2Lat_i + b3Long_i + b4(Long_i)^2 + b5(Long_i)^3 + \varepsilon_i.$$

Створюємо 2 нові змінні відповідно лінеаризації поліноміальної моделі. Запускаємо підпрограму побудови множинної регресії.

```
>Long2<-Long^2
>Long3<-Long^3
>z<-lm(JanTemp~Lat+Long+Long2+Long3)
```

>summary(z)

Call:
lm(formula = JanTemp ~ Lat + Long + Long2 + Long3)

Residuals:
   Min   1Q Median   3Q   Max
-8.569 -1.624 0.218 1.472 7.039

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -7.907e+02 1.351e+02 -5.854 3.46e-07 ***
Lat   -2.358e+00 8.998e-02 -26.202 < 2e-16 ***
Long   3.121e+01 4.285e+00   7.285 1.93e-09 ***
Long2   -3.515e-01 4.498e-02 -7.815 2.84e-10 ***
Long3   1.296e-03 1.552e-04   8.350 4.13e-11 ***

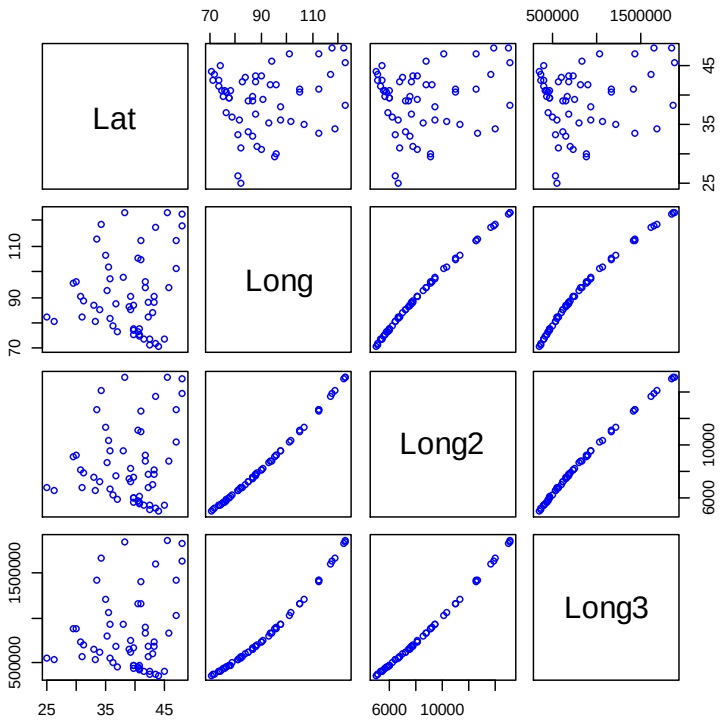
Signif.codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.225 on 51 degrees of freedom
Multiple R-squared: 0.9461,   Adjusted R-squared: 0.9419
F-statistic: 223.9 on 4 and 51 DF, p-value: < 2.2e-16

3)Схоже, що модель прекрасна: коефіцієнт детермінації дорівнює 0.946, всі змінні значущі. Але, на жаль, спостерігається дуже велика ступінь мультиколінеарності, лінійної залежності між Long, Long2 iLong3.

Перевіримо це, побудувавши матричну діаграму розсіювання даних для всіх регресорів (попередньо створивши з них фрейм), а також їх кореляційну матрицю.

> y<-data.frame(Lat,Long,Long2,Long3)
> pairs(y,col=4)



> cor(y,y)

```
          Lat      Long     Long2     Long3
Lat   1.0000000 0.1447689 0.1719480 0.1956674
Long  0.1447689 1.0000000 0.9970916 0.9888531
Long2 0.1719480 0.9970916 1.0000000 0.9973131
Long3 0.1956674 0.9888531 0.9973131 1.0000000
```
Наприклад, коефіцієнт кореляції між Long2, Long3 складає 0.9973.

4) Отже, спробуємо прибрати змінну Long3 з моделі.

```
> z<-lm(JanTemp~Lat+Long+Long2)
>summary(z)

Call:
lm(formula = JanTemp ~ Lat + Long + Long2)

Residuals:
    Min      1Q  Median      3Q     Max
-11.4826  -1.8054   0.3543   2.7986  14.0631

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 323.951482  31.355464  10.332 3.33e-14 ***
Lat          -2.521840   0.133800 -18.848  < 2e-16 ***
Long         -4.401407   0.621513  -7.082 3.67e-09 ***
Long2         0.023687   0.003238   7.316 1.55e-09 ***
---
Signif.codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.914 on 52 degrees of freedom
Multiple R-squared: 0.8724,     Adjusted R-squared: 0.8651
F-statistic: 118.6 on 3 and 52 DF,  p-value: < 2.2e-16
```

Зупиняємось на даній моделі. Коефіцієнт детермінації достатньо великий ($R^2 = 0.872$) і всі змінні є значущими.

$$JanTemp_i = 323{,}95 - 2{,}52Lat_i - 4{,}4Long_i + 0{,}024(Long_i)^2 + \varepsilon_i.$$

Приклад. Логістична регресія.

Оболонка пакету R містить багато різних даних. Файл USPop містить демографічні дані, результат перепису населення США з 1790 по 1990 рік. Перепис проводився кожні 10 років.

Нам треба побудувати модель залежності кількості населення (population) від року (year).

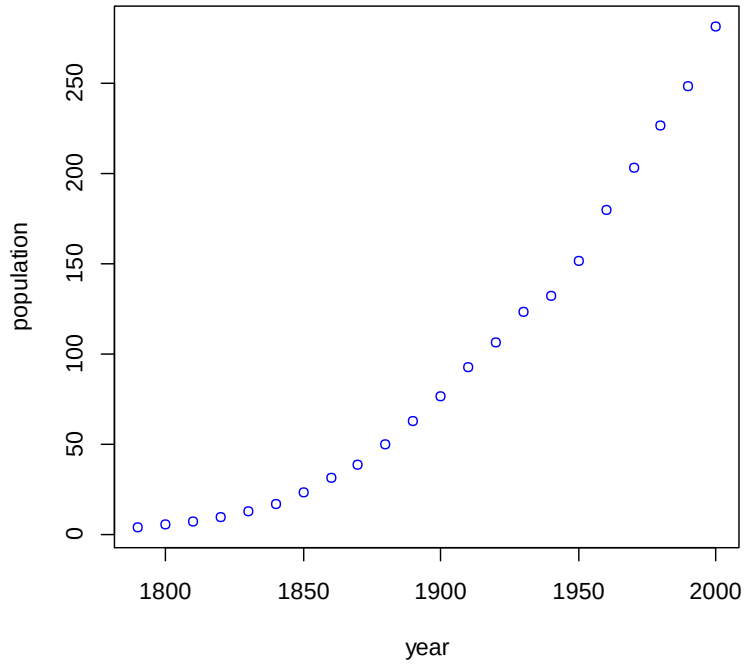
Назви змінних:

1.year: Рік
2.population: Кількість населення

Розв’язок.

1)Підключимо бібліотеку даних car. Потрібні дані знаходяться в фреймі USPop .Будуємо діаграму розсіювання даних.

```
>library(car)
Предупреждение
пакет ‘car’ был собран под R версии 2.15.3
>data(USPop)
>attach(USPop)
>USPop
year      population
1  1790   3.929214
2  1800   5.308483
3  1810   7.239881
4  1820   9.638453
5  1830  12.860702
6  1840  17.063353
7  1850  23.191876
8  1860  31.443321
9  1870  38.558371
10 1880  50.189209
11 1890  62.979766
12 1900  76.212168
13 1910  92.228496
14 1920 106.021537
15 1930 123.202624
16 1940 132.164569
17 1950 151.325798
18 1960 179.323175
19 1970 203.302031
20 1980 226.542199
21 1990 248.709873
22 2000 281.421906

>plot(year,population)
```



2)Застосуємо нелінійний МНК. Припустимо залежність між змінними у вигляді логістичної кривої.

$$population = \frac{b1}{1+e^{b2+b3\cdot x}} + \varepsilon,$$

де $x = year - 1790,\ \ b1,\ b2,\ b3$ – коефіцієнти, які нам треба оцінити.

Підключаємо бібліотеку nls2, звертаємось до підпрограми nls. В параметрах підпрограми нам треба вказати початкові ітерації для всіх трьох параметрів. Як в принципі можна підібрати такі початкові значення?

Зважаючи на графік логістичної кривої ($population \to b1$ при $x \to -\infty$) і на діаграму розсіювання даних, припускаємо, що $b1 = 350000$. Підставляючи в рівняння логістичної кривої першу з точок наших даних $x = 0,\ population = 3.927\ 000$, маємо:

$$3929 = \frac{350}{1 + e^{b2+b3\cdot 0}} \implies b2 \approx 4.5.$$

Потім підставляємо другу точку: $x = 1,\ \ population = 5.308.$

$$5308 = \frac{350}{1 + e^{4.5+b3}} \implies b3 \approx -0.3.$$

```
> library(nls2)
>pop.mod<-nls(population~b1/(1+exp(b2+b3*(year-1790))),start=list(b1=350,b2=4.5,b3=-0.3),trace=T)
1316527 :  350.0   4.5  -0.3
126178.2 :  110.1529450   2.9708687  -0.1313537
102005.1 :  113.56859344   1.92890304  -0.05316144
80518.74 :  126.7964174   1.5529176  -0.0197162
70727.55 :  153.90613630   1.86291739  -0.01655289
67316.21 :  224.76259376   2.57170312  -0.01452447
39837.77 :  386.05420462   3.75268130  -0.01734211
937.057 :  350.95547820   3.94573026  -0.02367056
909.2746 :  417.03766915   3.98091093  -0.02152612
458.0982 :  438.90600557   4.03514202  -0.02166182
457.8066 :  440.91580042   4.03203931  -0.02160155
457.8056 :  440.82044373   4.03242722  -0.02160639
457.8056 :  440.83445759   4.03239636  -0.02160586
457.8056 :  440.83338311   4.03239959  -0.02160591
> summary(pop.mod)

Formula: population ~ b1/(1 + exp(b2 + b3 * (year - 1790)))

Parameters:
    Estimate Std. Error t value Pr(>|t|)
b1 440.833383  35.000174   12.60 1.14e-10 ***
b2   4.032400   0.068180   59.14  < 2e-16 ***
b3  -0.021606   0.001007  -21.45 8.87e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.909 on 19 degrees of freedom

Number of iterations to convergence: 13
Achieved convergence tolerance: 1.186e-06
```

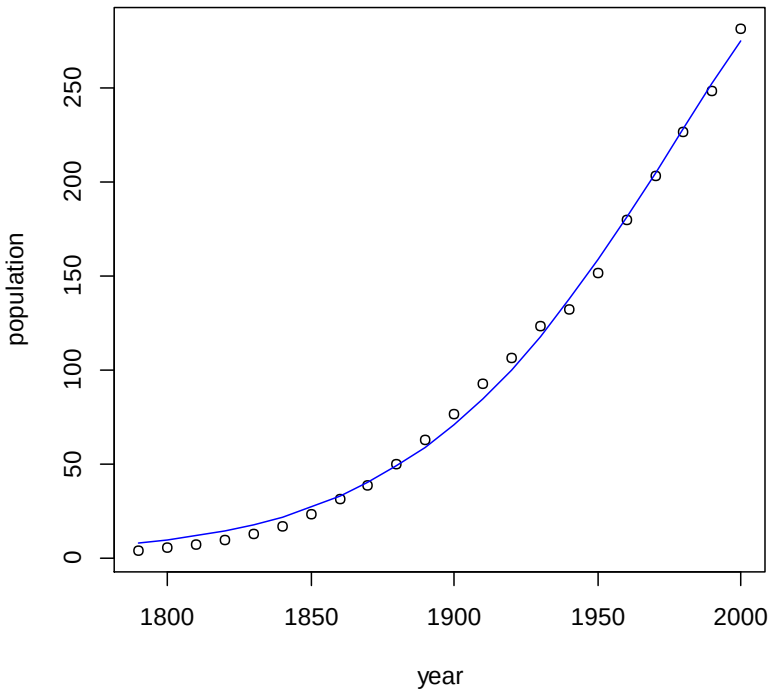
3)Знайдемо відсоток поясненої дисперсії, як ми це робили в іншому прикладі.

```
> RSS<-sum((population-fitted.values(pop.mod))^2)
> RSS
[1] 457.8056
> TSS<-sum((population-mean(population))^2)
> TSS
[1] 159914.6
> VAR.EXP<-1-RSS/TSS
> VAR.EXP
[1] 0.9971372
>
```

Відсоток поясненої дисперсії=99,7%. Модель, як ми бачимо непогана.

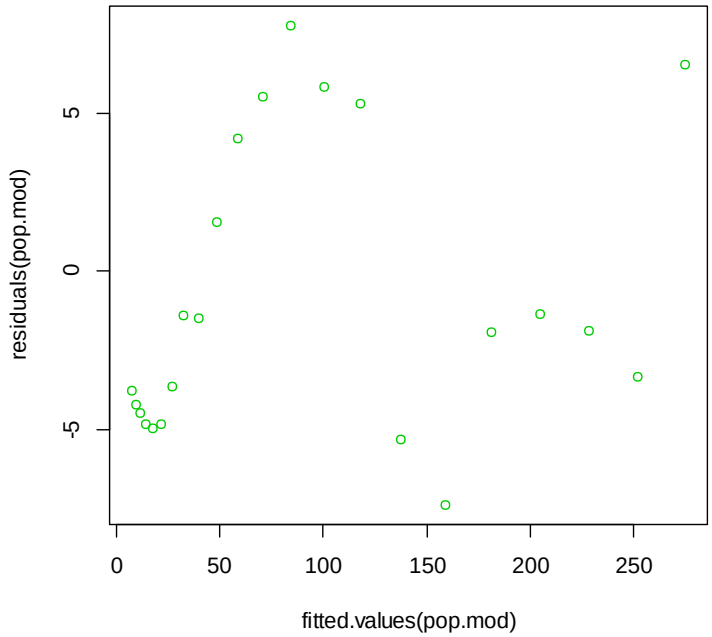
4)Побудуємо на діаграмі розсіювання даних підгоночну криву. Як ми бачимо, підгонка хороша.

```
> plot(year,population)
> lines(year,fitted.values(pop.mod),col=4)
```



5) Побудуємо графік залишків “Predicted versus Residuals”.

```
> plot(fitted.values(pop.mod),residuals(pop.mod),col=3)
```



Графік не демонструє чіткої залежності, тому вважаємо, що всі необхідні залежності в нашій моделі ми врахували.

8. Критерій χ^2.

а)Критерій згоди χ^2

Приклад (узгодження з дискретним розподілом).

Згенерувати вибірку обсягу 232, розподілену за законом Пуассона з параметром $\lambda = 2.5$, а потім перевірити за критерієм χ^2 гіпотезу про узгодженість даної вибірки з розподілом Пуассона з параметром $\lambda = 2.5$ з рівнем значущості $\alpha = 0.05$.

$H_0 = \{$Вибірка розподілена за законом Пуассона з параметром $\lambda = 2.5\}$.

Розв'язок.

Спершу генеруємо вибірку

```
> x<-rpois(232,2.5)
> table(x)
x
 0  1  2  3  4  5  6  7
23 49 61 46 27 20  4  2
```

Оператор table(x) групує вибірку, виявляючи емпіричні частоти варіантів. Обираємо такі множини для групування – значення 0, 1, 2, 3, 4, 5 і «6 або більше». Заносимо їх в масив.

```
> n_emp<-c(23,49,61,46,27,20,6)
```

Обчислюємо відповідні теоретичні частоти варіантів.

```
> pteor<-c(dpois(0:5,2.5),1-ppois(5,2.5))
> pteor
[1] 0.08208500 0.20521250 0.25651562 0.21376302 0.13360189 0.06680094 0.04202104
```

Тепер запускаємо підпрограму, що реалізує метод χ^2.

```
> chisq.test(x=n_emp,p=pteor)

        Chi-squared test for given probabilities

data:  n_emp
X-squared = 4.4247, df = 6, p-value = 0.6194

>
```

Оскільки досягнутий рівень значущості 0,6194>0.05, гіпотезу приймаємо.

Приклад (узгодження з абсолютно неперервним розподілом).

Згенерувати вибірку обсягу 500, розподілену за нормальним законом з параметрами $a = 5, \sigma = 1$, а потім перевірити за критерієм χ^2гіпотезу про узгодженість даної вибірки з нормальним розподілом з рівнем значущості $\alpha = 0.05$.

$H_0 = \{\text{Вибірка розподілена нормально}\}$.

Розв'язок.

На жаль, при розв'язку задачі неможливо скористатись підпрограмою chisq.test(), як в попередньому прикладі, бо вона налаштована на обчислення кількості степенів свободи в розподілі χ^2 за формулою $df = l - 1$ ($l -$ кількість множин групування), а у випадку, коли ми маємо складну нульову гіпотезу, і деякі параметри повинні бути оцінені за вибіркою, слід покласти $df = l - r - 1$; ($r -$ кількість параметрів, оцінених за вибіркою). Тому chisq.test() обчислює неправильний p-level.

1)Спершу генеруємо вибірку, оцінюємо параметри нормального розподілу за формулами $a = \bar{x};\ \sigma = \sqrt{D_B X}$.

```
> x<-rnorm(500,5,1)
>tableObPc
> sig<-sd(x)
> c(a,sig)
[1] 4.9433128 0.9664947
```

2)Нам треба побудувати емпіричний розподіл вибірки інтервального типу. Інтервали розбиття оберемо наступним чином: $(-\infty, a-2\sigma], (a-2\sigma, a-\sigma], (a-\sigma, a], (a, a+\sigma], (a+\sigma, a+2\sigma], (a+2\sigma, +\infty]$ (Саме так розбиває на півінтервали table(), якщо ж ми хочемо розбити на півінтервали $[a, b)$, застосовуємо опцію right=TRUE). Спершу створюємо послідовність bin з такими точками розбиття. Потім – інтервальний розподіл вибірки.

```
> bin<-seq(a-2*sig,a+2*sig,sig)
> bin
[1] 3.010323 3.976818 4.943313 5.909807 6.876302
> OB<-table(cut(x,breaks=c(-Inf,bin[1:5],Inf)))
> OB

(-Inf,3.01] (3.01,3.98] (3.98,4.94] (4.94,5.91] (5.91,6.88] (6.88, Inf]
         14          68         164         176          69           9
>
```

Отже, в масиві OB у нас містяться емпіричні частоти.

3)Знайдемо теоретичні ймовірності і частоти. Межі інтервалів вибрані таким чином, що ймовірності попадання в часткові інтервали вираховуються через стандартний нормальний розподіл в цілих точках.

```
> PR<-c(pnorm(-2),pnorm(-1:2)-pnorm(-2:1),1-pnorm(2))
> EX<-500*PR
```

```
> ans<-cbind(PR,EX,OB)
> ans
                PR       EX  OB
(-Inf,3.01] 0.02275013  11.37507  14
(3.01,3.98] 0.13590512  67.95256  68
(3.98,4.94] 0.34134475 170.67237 164
(4.94,5.91] 0.34134475 170.67237 176
(5.91,6.88] 0.13590512  67.95256  69
(6.88, Inf] 0.02275013  11.37507   9
>
```

4) Обчислимо значення хі-квадрат спостереження за формулою $\chi^2 = \sum_{k=1}^{l} \frac{(OB-EX)^2}{EX}$.

```
>chi<-sum((OB-EX)^2/EX)
>chi
[1] 1.544976
```

Значення p-level можна знайти таким чином:

```
> p.val<-1-pchisq(chi,6-2-1)
> p.val
[1] 0.6719294
>
```

5)Оскільки p=0.6719>0.05, приймаємо гіпотезу про нормальний розподіл вибірки.

б)Критерій χ^2 для гіпотези незалежності випадкових величин

Приклад У таблиці наведені дані про 1426 ув'язнених, яких було класифіковано щодо алкогольної залежності (алкоголік, неалкоголік) і характеру злочинів, за які їх засудили. Чи можна на підставі цих даних дійти висновку про наявність зв'язку між алкоголізмом і характером злочину?

Вид злочину	Алкоголіки	Неалкоголіки
1.Підпал	50	43
2.Згвалтування	88	62
3.Насильницькі дії	155	110
4.Крадіжка	379	300
5.Виготовлення фальшивих грошей	18	14
6.Шахрайство	63	144

Висуваємо гіпотезу

H_0={Вид злочину і алкогольна залежність не пов'язані між собою}

1)Створюємо матрицю з відповідними даними. Для наочності надписуємо рядки і стовпчики, в принципі, цього можна не робити.

```
> al<-c(50,43,88,62,155,110,379,300,18,14,63,144)
> Alc<-matrix(data=al, nrow=6, byrow=TRUE)
> dimnames(Alc)<-list(Crime=c("Arson","Rape","Violence","Theft","Counterfeiting",
+ "Scam"),Category=c("Alcoholic","Nonalcoholic"))
> Alc
          Category
Crime      Alcoholic Nonalcoholic
  Arson           50          43
  Rape            88          62
  Violence       155         110
```

```
  Theft            379       300
  Counterfeiting    18        14
  Scam              63       144
>
```

2)Для застосування критерію скористаємось підпрограмою chisq.test().

```
> chisq.test(Alc)

        Pearson's Chi-squared test

data:  Alc
X-squared = 49.7306, df = 5, p-value = 1.573e-09
```

Оскільки досягнутий рівень значущості $p = 1.573 \cdot 10^{-9} < 0.05$, то гіпотезу про відсутність зв'язку між характером злочину і алкоголізмом відхиляємо. Зв'язок є!

9. Критерій Колмогорова.

Критерій Колмогорова застосовується як критерій узгодженості вибірки з певним розподілом.

Приклад.

Згенерувати вибірку обсягу 500, розподілену за нормальним законом з параметрами $a = 5, \sigma = 1$, а потім перевірити за критерієм Колмогорова гіпотезу про узгодженість даної вибірки з нормальним розподілом з параметрами $a = 5, \sigma = 1$ з рівнем значущості $\alpha = 0.05$.

$H_0 = \{$Вибірка розподілена нормально з параметрами $a = 5, \sigma = 1\}$.

Для застосування критерію Колмогорова призначена підпрограма ks.test().

```
> x<-rnorm(500,5,1)
> ks.test(x,y="pnorm",mean=5,sd=1)

        One-sample Kolmogorov-Smirnov test

data:  x
D = 0.0406, p-value = 0.3823
alternative hypothesis: two-sided

>
```

Висновк: приймаємо основну гіпотезу – вибірка має нормальний розподіл.

10. Дисперсійний аналіз

Дисперсійний аналіз – це сукупність методів, які дозволяють перевіряти: чи змінюється середнє значення характеристик деяких об'єктів в залежності від дії деякого фактора або кількох факторів.

а) Однофакторний дисперсійний аналіз

Нехай ми маємо k рівнів фактора. Специфікація моделі має вигляд

$$x_{ij} = a_i + \varepsilon_{ij}; \quad i = \overline{1,k}; \quad j = \overline{1,n},$$

де a_i - середнє значення, обчислене при $i-$му рівні фактора; ε_{ij} - випадкова величина, результат впливу неврахованих факторів, ε розподілена $N(0,\sigma^2)$.

Приклад.

Кожний співак Нью-Йоркської Хорової спільноти в 1979 році офіційно доповідав про свій зріст. Голоси співаків поділяються від найнижчого за тоном до найвищого на такі типи: сопрано, альт, тенор, бас. Зазвичай 2 перші типи належать жінкам, 2 останні – чоловікам.

Ми хочемо дослідити – як зріст впливає на голосовий діапазон співака чи співачки. Також треба зробити порівняння сопрано та альтів і, окремо, порівняння тенорів і басів. (Правда, дані можна вважати не дуже достовірними – співаки дещо прибріхували щодо свого зросту, занижуючи його. Їх шикували на сцені згідно зі зростом, а всім хотілося стояти в першому ряду. Вважаємо, що це зумовлює лише систематичну похибку).

Назви змінних:

1. Soprano: Зріст співаків-сопрано (в дюймах)
2. Alto: Зріст співаків-альтів (в дюймах)
3. Tenor: Зріст співаків-тенорів (в дюймах)
4. Bass: Зріст співаків-басів (в дюймах)

Дані

Soprano	Alto	Tenor	Bass
64	65	69	72
62	62	72	70
66	68	71	72
65	67	66	69
60	67	76	73
61	63	74	71
65	67	71	72
66	66	66	68
65	63	68	68
63	72	67	71
67	62	70	66
65	61	65	68
62	66	72	71
65	64	70	73
68	60	68	73
65	61	73	70
63	66	66	68
65	66	68	70
62	66	67	75
65	62	64	68
66	70		71
62	65		70
65	64		74
63	63		70
65	65		75
66	69		75
65	61		69
62	66		72
65	65		71
66	61		70
65	63		71
61	64		68

65	67	70
66	66	75
65	68	72
62		66
	72	
	70	
	69	

Розв'язок.

1)Спершу обробимо дані, створимо відповідний текстовий файл. Потім занесемо дані у фрейм з двома стовпчиками-змінними: один відповідає зросту людини (Hei, залежна змінна), другий – тип голосу (Factor, фактор).

```
> da<-read.table(file.choose(),header=TRUE)
> da
   Hei  Factor
1    64 Soprano
2    62 Soprano
…….
130  66 Bass
```

2)Для того, щоб запустити програму однофакторного дисперсійного аналізу, скористаємось комбінацією функцій summary() та aov() (скорочення від ANOVA – analisis of variances). В аргументах вказуємо спершу залежну змінну, потім – фактор.

```
>attach(da)
> summary(aov(Hei~Factor))
            Df Sum Sq Mean Sq F value Pr(>F)
Factor       3  956.5   318.8    44.7 <2e-16 ***
Residuals   126  898.7     7.1
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Як ми бачимо, досягнутий рівень значущості при перевірці гіпотези $\{a_1 = a_2 = a_3 = a_4\}$ є $p = 2 \cdot 10^{-16}$. Це означає, що при рівні значущості $p = 0{,}05$ гіпотезу про рівність середніх відхиляємо. Є зв'язок між зростом людини і тембром голосу.

3)Отже, зріст і тембр пов'язані. Але серед співаків у вибірці є і жінки, і чоловіки. Може, ефект залежності просто показує відмінність між жінками і чоловіками – жінки в середньому нижчі і їх тембр в середньому вищий.Чи буде така сама залежність всередині статі?

Вилучимо з вибірки чоловіків, для цього створимо новий фрейм da1, де змінна Factor не може приймати значення Bass і Tenor.

```
da1<-da[as.character(da[,"Factor"])!="Bass" & (da[,"Factor"])!="Tenor",]
```

Запускаємо програму однофакторного дисперсійного аналізу.

```
> summary(aov(da1$Hei~da1$Factor))
             Df Sum Sq Mean Sq F value Pr(>F)
da1$Factor   1    0.5   0.519   0.075  0.784
Residuals   72  495.4   6.881
```

Досягнутий рівень значущості $p = 0{,}784$. Отже, серед жінок відсутній зв'язок між зростом і тембром голосу.

4)Зробимо такий самий аналіз всередині чоловіків.

```
> da2<-da[as.character(da[,"Factor"])!="Soprano" & (da[,"Factor"])!="Alto",]
> summary(aov(da2$Hei~da2$Factor))
             Df Sum Sq Mean Sq F value Pr(>F)
da2$Factor    1   32.9   32.91   4.407 0.0405 *
Residuals    54  403.3    7.47
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Тут $p = 0{,}0405$. Отже, серед чоловіків суттєвий зв’язок між зростом і тембром.

б) Двофакторний дисперсійний аналіз

Нехай на досліджувану ознаку впливають одразу 2 фактори A і B. Ми маємо p рівнів фактора A і q рівнів фактора B. Таким чином, кожне спостереження x_{ijk} залежить від трьох індексів – рівня фактору A, рівня фактору B і номера спостереження. Специфікація моделі має вигляд

$$x_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \varepsilon_{ijk}; \quad i = \overline{1,p}; \quad j = \overline{1,q}; \quad k = \overline{1,n},$$

де μ - загальне середнє, α_i, β_j - ефекти впливу факторів A і B відповідно; γ_{ij} - ефект взаємодії; ε_{ijk} - випадкова величина, результат впливу неврахованих факторів (неспецифічна компонента), ε розподілена $N(0, \sigma^2)$.

Дослідити, чи присутня залежність між кількістю помилок, яких припускаються щури при проходженні лабіринту, від їх оточуючого середовища та від генетичних властивостей.

Приклад.

Назви змінних:

1. ENVIRONMET = Оточуюче середовище: FREE – вільне; RESTRCTED - обмежене.
2. STRAIN = Генетична характеристика: BRIGHT – чиста лінія «розумних» пацюків; MIXED - мішаної крові; DULL– чиста лінія «дурних» пацюків.
3. ERRORS = Кількість помилок, яких припустився щур при проходженні лабіринту.

Дані

ENVIRNMNT	ERRORS	STRAIN
Free	26	Bright
Free	14	Bright
Free	41	Bright
Free	16	Bright
Free	41	Mixed
Free	82	Mixed
Free	26	Mixed
Free	86	Mixed
Free	36	Dull
Free	87	Dull
Free	39	Dull

Free	99	Dull
Restricted	51	Bright
Restricted	35	Bright
Restricted	96	Bright
Restricted	36	Bright
Restricted	39	Mixed
Restricted	114	Mixed
Restricted	104	Mixed
Restricted	92	Mixed
Restricted	42	Dull
Restricted	133	Dull
Restricted	92	Dull
Restricted	124	Dull

Розв'язок.

1)Створимо відповідний текстовий файлrat.txt. Потім занесемо дані у фрейм з трьома стовпчиками-змінними: ERRORS – залежна змінна; ENVIRONMENT, STRAIN – фактори.

```
> da<-read.table(file.choose(),header=TRUE)
> da
> attach(da)
```

2)Запускаємо програму дисперсійного аналізу. В аргументах вказуємо, що перевіряємо вплив і середовища, і генетичних властивостей, і сукупності дій цих факторів.

```
> z<-aov(ERRORS~ENVIRONMENT+STRAIN+ENVIRONMENT:STRAIN)
> anova(z)
Analysis of Variance Table

Response: ERRORS
                Df  Sum Sq Mean Sq F value  Pr(>F)
ENVIRONMENT      1  5551.0  5551.0  5.8225 0.02670 *
STRAIN           2  7939.7  3969.9  4.1640 0.03264 *
ENVIRONMENT:STRAIN  2    16.1     8.0  0.0084 0.99160
Residuals       18 17160.8   953.4
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3)Як ми бачимо, гіпотезу про рівність середніх при зміні фактора середовища ENVIRONMENT відхиляємо, бо $p = 0{,}0267$- вплив є. Аналогічно спостерігається вплив генетичного фактора STRAIN ($p = 0{,}03264$). А от впливу комбінації цих факторів нема : $p = 0{,}9916$.

11. Факторний аналіз.

Теоретичні відомості

Дослідницький факторний аналіз (exploratory factor analysis) - це статистична техніка, призначена для виявлення та опису прихованих факторів (змінних), які впливають на спостережувані характеристики досліджуваних об'єктів. Нехай

спостерігається n об’єктів, кожний з яких описується p характеристиками (variables), X_j^i - значення i-тої спостережуваної характеристики j-того об’єкту (i=1,…, p, j=1,…,n). Припустимо, що існує k прихованих факторів (latent variables, factors), вплив яких описує поведінку спостережуваних X_j^i. Позначимо f_j^m – значення m-того прихованого фактора для j-того об’єкту (m=1,…,k). Тоді модель факторного аналізу має вигляд:

$$X_j^i = \mu_i + \sum_{m=1}^{k} \lambda_m^i f_j^m + \varepsilon_j^i ,$$

де μ_i – середнє значення i-тої змінної (математичне сподівання),

λ_m^i – навантаження (loading), тобто коефіцієнт, який характеризує вплив m-того фактора на i-ту змінну,

ε_j^i – збурення або похибка (error) факторної моделі, що характеризує вплив на об’єкт факторів, які не були враховані моделлю.

У матричному вигляді цю модель можна записати як

$$X_j = \mu + \Lambda f_j + \varepsilon_j, \qquad (1)$$

де X_j – вектор-стовпчик спостережуваних змінних j-того об’єкту, μ – вектор середніх, $\Lambda = \begin{pmatrix} \lambda_1^1 & \cdots & \lambda_k^1 \\ \vdots & \ddots & \vdots \\ \lambda_1^p & \ldots & \lambda_k^p \end{pmatrix}$ – матриця навантажень, f_j – вектор значень факторів j-того об’єкту, ε_j – вектор похибок.

Для підгонки моделі факторного аналізу можна застосовувати різні методи. З сучасних методів найбільш популярним є метод найбільшої вірогідності у гаусовій моделі факторного аналізу. В цій моделі вважається, що фактори f_j^m є незалежними стандартними гауссовими випадковими величинами, а похибки ε_j^i теж є незалежними гаусовими з нульовим математичним сподіванням та дисперсіями σ_i^2. Похибки вважають незалежними від факторів. У цьому випадку коваріаційна матриця спостережуваних змінних Σ записується у вигляді

$$\Sigma = \Lambda\Lambda^T + \Psi, \qquad (2)$$

де Ψ – діагональна матриця $\Psi = \begin{pmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_з^2 \end{pmatrix}$ (коваріаційна матриця ε_j).

При застосуванні методу найбільшої вірогідності для оцінювання Λ і Ψ використовується вибіркова коваріаційна матриця Σ_n спостережень X_j, j=1,…,n, для якої підшукують наближення у вигляді (2). Якщо різні спостережувані змінні вимірюються у різних шкалах, буває корисно попередньо пронормувати їх коренем з їх дисперсії, щоб розкид всіх змінних був приблизно однаковим. Це відповідає використанню у факторному аналізі креляційної матриці замість коваріаційної.

Слід відмітити, що формули (1), (2) не дають змоги визначити параметри моделі однозначно. Дійсно, якщо вибрати довільну ортогональну матрицю G і замінити матрицю навантажень Λ на ΛG, а значення факторів f_j на $G^T f_i$, то ліві частини (1) і (2) не зміняться, а елементи нового вектора факторів $G^T f_i$ будуть незалежними гауссовими випадковими величинами, якщо такими були елементи f_j . Можна сказати, що фактори і навантаження визначаються у моделі з точністю до множення на довільну ортогональну матрицю G.

Геометрично таке множення на ортогональну матрицю можна інтерпретувати як обертання (поворот) простору факторів. Якщо $s = \frac{[(p-k)^2-(p+k)]}{2} < 0$ то (2) не дає можливості оцінити параметри навіть з точністю до обертання. Тому кількість факторів k у факторному аналізі звичайно вибирають так, щоб $s > 0$. (s називають кількістю ступенів вільності).

З точки зору математики, всі повороти факторів рівноправні, але для змістовної інтерпретації може бути корисним той чи інший варіант повороту. Існують спеціальні алгоритми вибору поворотів, що, як можна сподіватись, дають зручні для інтерпретації набори факторів. Одним з найбільш популярних з них є метод varimax (нормований), у якому обертання вибирають так, щоб максимізувати $\sum_{i,m}(d_{i,m} - d_m)^2$, де $d_{i,m} = (\lambda_m^i)^2 / \sum_l (\lambda_l^i)^2$.

Якщо модель (2) є вірною, то дисперсії спостережуваних змінних можна представити у вигляді:

$$DX_j^i = h_i + \sigma_i^2,$$

де $h_i = \sum_{m=1}^{k} (\lambda_m^i)^2$ – величина, що зветься загальністю (communality) і визначає ту частину дисперсії змінної, яка пояснюється впливом прихованих факторів нашої моделі; σ_i^2 – дисперсія збурення ε_j^i, тобто та частина дисперсії змінної, яка пов'язана з факторами, не врахованими у нашій моделі (ця величина зветься специфічністю – uniqueness) .

Чим більша специфічність порівняно з загальністю, тим гірше дана спостережувана змінна пояснюється обраними нами прихованими факторами.

Загальною характеристикою якості моделі є частка сумарної дисперсії даних, яку пояснюють приховані фактори. Вважається, що доцільно використовувати моделі, які пояснюють не менше 80% сумарної дисперсії даних. Збільшуючи кількість прихованих факторів можна збільшити і цю частку, але моделі з великою кількістю прихованих факторів важко інтерпретувати змістовно.

Факторний аналіз в R

У R для підгонки факторної моделі за методом найбільшої вірогідності можна використовувати функцію **factanal**. Специфікація цієї функції:

```
factanal(x, factors, data = NULL, covmat = NULL, n.obs = NA,
    subset, na.action, start = NULL,
    scores = c("none", "regression", "Bartlett"),
    rotation = "varimax", control = NULL, ...)
```

Викликати функцію можна різними способами. Якщо потрібно обробити дані X_j^i, які містяться у певній матриці, або фреймі даних, то назву матриці слід передати як параметр x, а у параметрі factors вказати кількість прихованих факторів, які використовуються у моделі. Виклик функції може виглядати так:

```
fm<- factanal(Z,2)
```

(провести факторний аналіз даних, що містяться у Z, на основі моделі з двома факторами. Для аналізу буде використано кореляційну матрицю, підраховану за даними).

Параметр x може містити формулу, наприклад:

```
fm<-factanal(~v1+v2+v3+v4+v5+v6, factors = 3,data=Z)
```

(для факторного аналізу використовуються змінні v1 – v6 з фрейму даних Z, модель з трьома факторами).

Якщо потрібно провести факторний аналіз на основі коваріаційної матриці, яка вже підрахована, використовується параметр covmat:

```
fm<-factanal(covmat=U,factors=2)
```

(факторний аналіз на основі коваріаційної матриці U. За потреби можна вказувати у цьому параметрі кореляційну функцію).

Параметр scores вказує спосіб обчислення прогнозів для значень факторів. За умовчанням він "none", тобто значення факторів не прогнозуються. Якщо вказати

```
fm<-factanal(Z,2,scores="regression")
```

то для всіх об'єктів, що описані у Z, будуть спрогнозовані значення прихованих факторів за регресійним методом і результат можна буде отримати в компоненті fm$scores. Прогнозування значень факторів можливе лише якщо параметр x заданий.

Параметр rotation вказує на метод обертання факторного простору, що буде використаний програмою. За умовчанням це varimax. Якщо обертання непотрібне, слід вказати rotation=”none”.

Результатом роботи функції є об’єкт, що містить результати аналізу. Навантаження містяться у компоненті $loadings, а прогнози значень факторів – у компоненті $scores цього об’єкту.

Розглянемо приклад застосування факторного аналізу в R.

Приклад застосування функції factanal

Дані для прикладу містяться у файлі Teachers.txt . У цьому файлі кожен рядочок відповдає одному викладачу університету, що читає певний лекційний курс. Стовпчики відповідають характеристикам, визначеним за результатами опитування студентів про відповідний курс. Ці характеристики мають наступний зміст:

Pres (P) – відсоток відвідувань лекцій студентами.
Stud.est (S) – середній бал студентів отриманий за даний курс (5-бальна шкала).
Underst (Un) – рівень розуміння студентами даного курсу, за їх оцінкою.
Interest (I) – наскільки цікавим був курс.
Useful (Us) – корисність курсу для студентів.
Org. (O) – рівень організації курсу.
Justif. (J) – справедливість викладача.
Demand. (D) – вимогливість викладача.
Respect (R) – повага викладача до студентів.

Спробуємо подивитись, наскільки адекватно будуть описуватись такі дані факторною моделлю.

Читаємо дані з файлу і застосовуємо функцію factanal.

```
> x<-read.table(file.choose(),header=T)
> x.fa<-factanal(as.matrix(x),factors=3,scores="regression")
```

Результати аналізу було збережено у об’єкті x.fa . Виведемо на екран ці результати:

```
> x.fa


Call:
factanal(x = as.matrix(x), factors = 3, scores = "regression")

Uniquenesses:
   Pres Stud.est  Underst Interest   Useful     Org.  Justif.  Demand.  Respect
  0.547    0.509    0.073    0.082    0.091    0.034    0.108    0.238    0.170

Loadings:
           Factor1 Factor2 Factor3
Pres               0.605   0.296
Stud.est   0.677  -0.107   0.147
Underst    0.801   0.209   0.491
Interest   0.910   0.232   0.190
Useful     0.915   0.266
```

Org.	0.936	0.295	
Justif.	0.924		-0.196
Demand.	0.125	0.848	-0.166
Respect	0.908		

	Factor1	Factor2	Factor3
SS loadings	5.335	1.354	0.459
Proportion Var	0.593	0.150	0.051
Cumulative Var	0.593	0.743	0.794

Test of the hypothesis that 3 factors are sufficient.
The chi square statistic is 17.99 on 12 degrees of freedom.
The p-value is 0.116

За рядочком Uniquenesses (специфічності) бачимо, що найменшу залежність від прихованих факторів у нашій моделі мають змінні Pres та Stud.est. Всі інші змінні досить сильно пов'язані з прихованими факторами, їх специфічності невеликі.

Далі у таблиці Loadings (навантаження) можна побачити коефіцієнти, що показують вплив факторів на змінні. Коефіцієнти малі за абсолютним значенням не відображаються. З таблиці видно, що другий фактор пов'язаний з вимогливістю викладача та рівнем відвідуваності помітними позитивними навантаженнями і має невеликий негативний вплив на оцінки студентів (вимогливість викладача природно викликає більшу відвідуваність і призводить до нижчих оцінок які він виставляє). Перший фактор пов'язаний позитивно зі змінними Stud.est, Underst, Interest, Useful, Org., Justif., Respect. Його можна охарактеризувати як фактор взаємної симпатії викладача і студентів, що приводить до загальних позитивних оцінок курсу студентами і студентів – лектором. Третій фактор інтерпретувати важче.

Якість моделі характеризують три наступних рядочки. З них найбільш важливим є останній:
Cumulative Var 0.593 0.743 0.794

Тут вказано, яку частку дисперсії (розкиду) даних пояснює факторна модель. Якщо обрати модель з одним фактором, то пояснено буде 59.3% дисперсії, два фактори пояснюють 74.3%, три фактори – 79.4%. Помітно, що добавка, внесена третім фактором – невелика. (Ці добавки, порівняно з попереднім відсотком вказані у другому рядочку - Proportion Var, а відповідні абсолютні значення внеску у сумарну пояснювану дисперсію чергового доданого до моделі фактора – у першому - SS loadings).

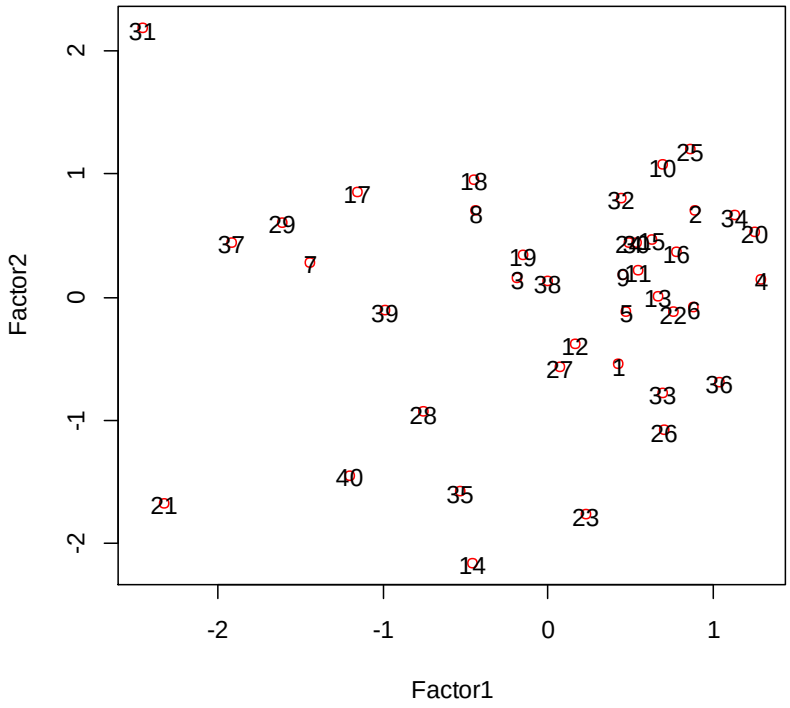
Нарешті, в останніх трьох рядочках міститься повідомлення про застосування тесту для перевірки того, чи є отримані три фактори достатніми для опису даних. Результатом тесту є p-value=0.116, згідно з яким слід прийняти основну гіпотезу про те, що обрана трифакторна модель адекватна для опису даних.

Візуалізація результатів факторного аналізу.

Для того, щоб краще відобразити результати факторного аналізу, доцільно використовувати різні техніки візуалізації. Наприклад, викладачі, що розглядались у нашому прикладі характеризуються прогнозованими значеннями їх прихованих факторів. Ці значення знаходяться у компоненті $scores і їх можна відобразити на звичайній діаграмі розсіювання:

> plot(x.fa$scores[,1:2],col=2)
> text(x.fa$scores[,1:2],labels=1:40)

В результаті на екран виводиться діаграма розсіювання, де кожному викладачу відповідає точка у просторі перших двох прихованих факторів:



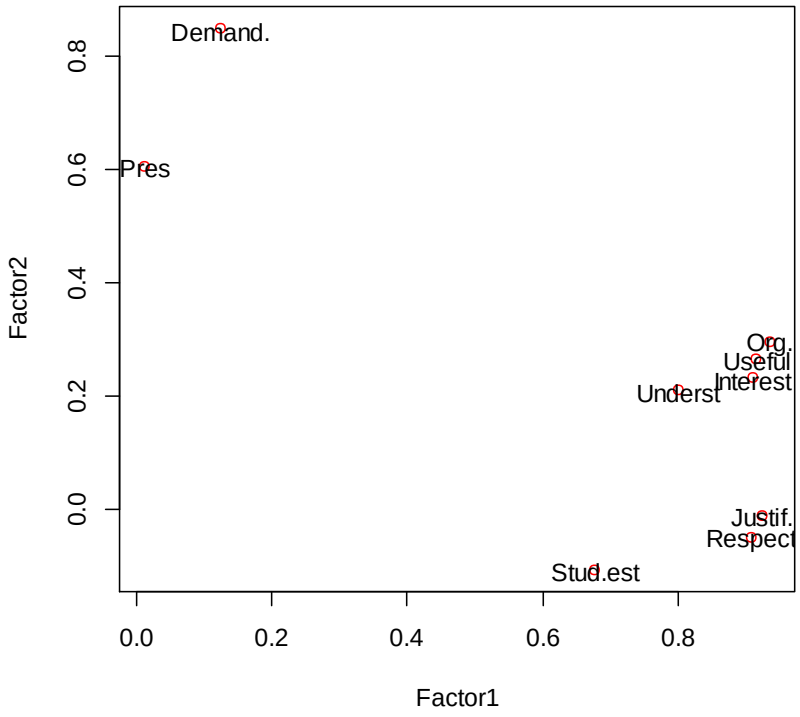
Як ми пам’ятаємо, у прикладі перший фактор пов’язаний з симпатичністю, а другий – з вимогливістю викладача. Таким чином, 31 викладач є найменш симпатичним і найбільш вимогливим, 21-й – найменш вимогливим і, тим не менше, практично так само несимпатичним, як 31-й. Можна помітити щільну групу викладачів, що характеризуються високою симпатичністю і більш розмиту хмару не так симпатичних викладачів. По осі вимогливості такого ефекту не помітно, лише 31-й викладач виділяється своєю вимогливістю, а вимогливість інших розподілена більш-менш рівномірно.

Аналогічно можна вивести діаграму навантажень:

> plot(x.fa$loadings[,1:2],col=2)
> text(x.fa$loadings[,1:2],labels=dimnames(x.fa$loadings)[[1]])

(тут як назви точок використовуються назви рядочків у матриці навантажень)

Ця діаграма ілюструє сказане вище: змінні Pres та Demand опинились поруч і «притягаються» до другого фактора (мають великі значення навантажень по другому фактору і малі по всіх інших). Змінні Ord, Useful, Interest та Underst розташувались дуже близько одна від одної і притягаються до першого фактора. І т.д.



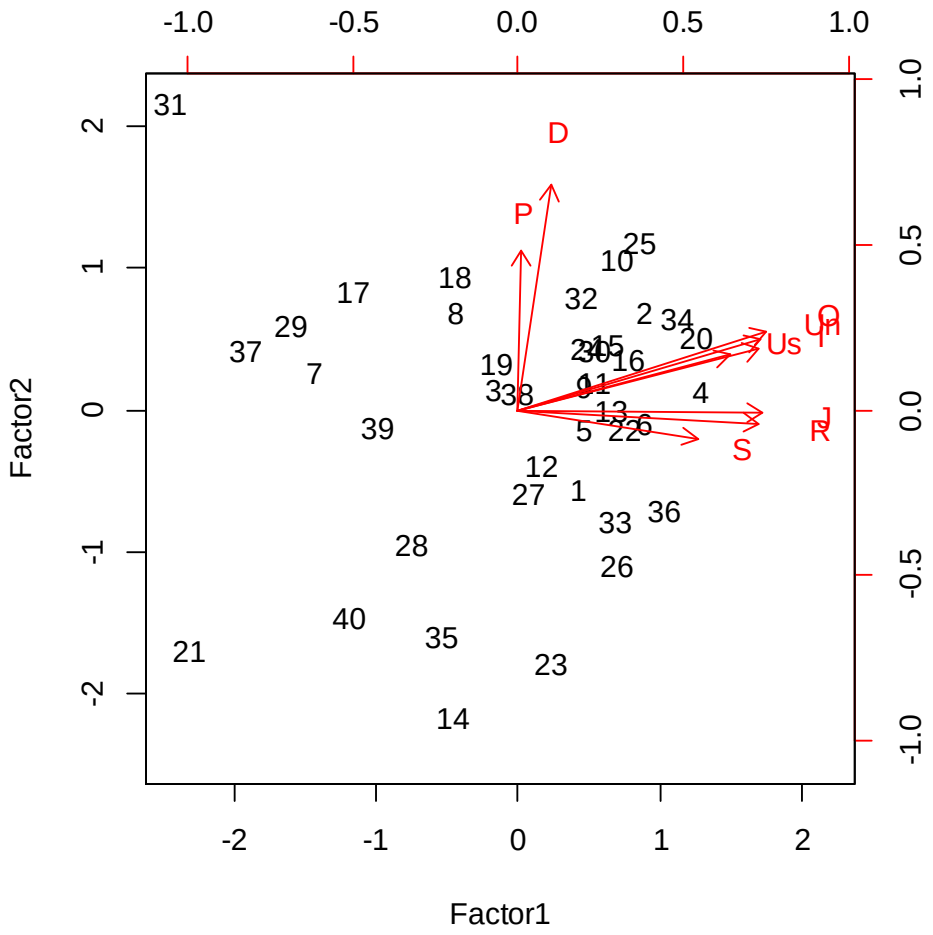
Для того, щоб подивитись, як положення об’єктів пов’язане з початковими змінними, можна використати так званий подвійний графік (biplot). Це робиться за допомогою функції **biplot**. Дві перші (основні) змінні цієї функції – x (координати точок) та y (координати змінних). Щоб на діаграмі довгі назви змінних не заважали розглядати деталі, ми створимо нову матрицю навантажень z зі скороченими назвами:

```
> z<-x.fa$loadings
> u<-c("P","S","Us","I","Un","O","J","D","R")
> dimnames(z)[[1]]<-u
```

Тепер виконуємо функцію biplot:

```
> biplot(x.fa$scores,z)
```

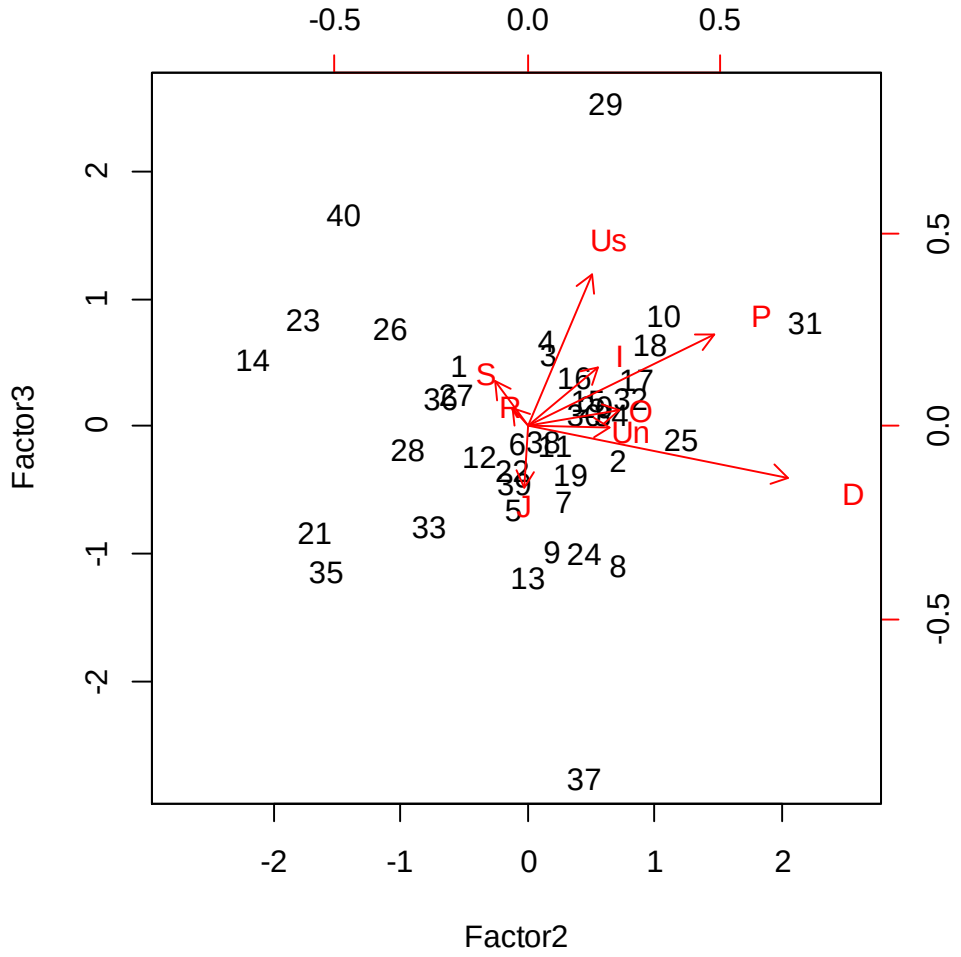
Результат:



На цій діаграмі початковим характеристикам (змінним) відповідають стрілки, що зображають приблизний напрямок зміни відповідної характеристики. Скажімо, характеристики P і D зростають разом із другим фактором, S, R, J – разом з першим. Всі інші характеристики утворюють третю групу, більше пов’язану з першим фактором, але залежну також від другого. Довжина стрілочки вказує на те, наскільки вдається описати дану змінну за допомогою обраних двох факторів. Чим коротша стрілочка, тим менше зв’язок змінної з факторами.

Аналогічно можна будувати діаграми розсіювання факторів та змінних у просторі інших пар факторів. Наприклад, подвійна діаграма для другого-третього факторів у нашому прикладі має вигляд:

> biplot(x.fa$scores[,2:3],z[,2:3])



Зміст

Література

1. Майборода Р.Є. Регресія: Лінійні моделі: Навчальний посібник. – К.:ВПЦ «Київський університет», 2007. – 296 с.
2. Ugarte M.D., Militino A.F., Arnholt A.T. Probability and statistics with R. – Boca Raton, London, New York: CRC Press, Taylor&Francis Group, 2008. – 700 p.
3. Venables W.N., Ripley B.D. Modern applied statistics with S. – Springer, 2002. – 495 p.
4. Matloff N. The Art of R Programming: A Tour of Statistical Software Design. - No Starch Press, 2011. – 154 p.
5. Teetor P. R Cookbook. - O'Reilly Media, 2011. - 436 p.

Навчальне видання

МАЙБОРОДА *Ростислав Євгенович*
СУГАКОВА *Олена Володимирівна*

АНАЛІЗ ДАНИХ ЗА ДОПОМОГОЮ ПАКЕТА R