# 第 12 天

## 探索資料分析

當我們成功將資料輸入 R 語言之後，針對陌生的資料進行的會是**探索資料分析（Exploratory Data Analysis，EDA）**，透過這個步驟可以知道這個資料的外觀、維度以及變數的分佈等資訊。我們將透過一系列 R 語言的內建函數與繪圖輔助我們達成這個目的。

### 內建函數

常見的資料讀入之後會以資料框（Data Frame）格式儲存，針對資料框有幾個內建函數：

#### 資料的維度

使用 nrow()、ncol() 與 dim() 這三個函數可以得知這個資料的維度資訊：

> nrow(iris) # 告訴我們內建資料 iris 有幾個觀測值  
[1] 150  
> ncol(iris) # 告訴我們內建資料 iris 有幾個變數  
[1] 5  
> dim(iris) # 告訴我們內建資料 iris 有幾個觀測值與有幾個變數  
[1] 150 5

其中 nrow() 函數命名是 **number of rows** 的縮寫，ncol() 函數命名是 **number of columns** 的縮寫，dim() 函數命名是 **dimensions** 的縮寫。知道這些全稱，可以有效幫助我們記憶這些函數。

#### 資料的外觀

使用 head() 與 tail() 這兩個函數可以得知部分資料的外觀，預設是印出前六或後六個觀測值（含變數名稱）；使用 names() 函數可以得知變數名稱。

> head(iris) # 印出內建資料 iris 的前六個觀測值（含變數名稱）  
 Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
1 5.1 3.5 1.4 0.2 setosa  
2 4.9 3.0 1.4 0.2 setosa  
3 4.7 3.2 1.3 0.2 setosa  
4 4.6 3.1 1.5 0.2 setosa  
5 5.0 3.6 1.4 0.2 setosa  
6 5.4 3.9 1.7 0.4 setosa  
> tail(iris) # 印出內建資料 iris 的後六個觀測值（含變數名稱）  
 Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
145 6.7 3.3 5.7 2.5 virginica  
146 6.7 3.0 5.2 2.3 virginica  
147 6.3 2.5 5.0 1.9 virginica  
148 6.5 3.0 5.2 2.0 virginica  
149 6.2 3.4 5.4 2.3 virginica  
150 5.9 3.0 5.1 1.8 virginica  
> names(iris) # 印出內建資料 iris 的五個變數名稱  
[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width" "Species"

#### 變數的分佈情況

使用 summary() 函數可以得知每一個變數的描述性統計量：

> summary(iris) # 印出內建資料 iris 的變數描述性統計



圖 12-1 summary(iris) 的輸出

#### 包含很多資訊的 str() 函數

使用 str() 函數可以輸出關於資料很多的資訊：像是資料結構的種類、觀測值數、變數個數、變數名稱與前幾筆觀測值等，str() 函數命名是 **structure** 的縮寫。

> str(iris)

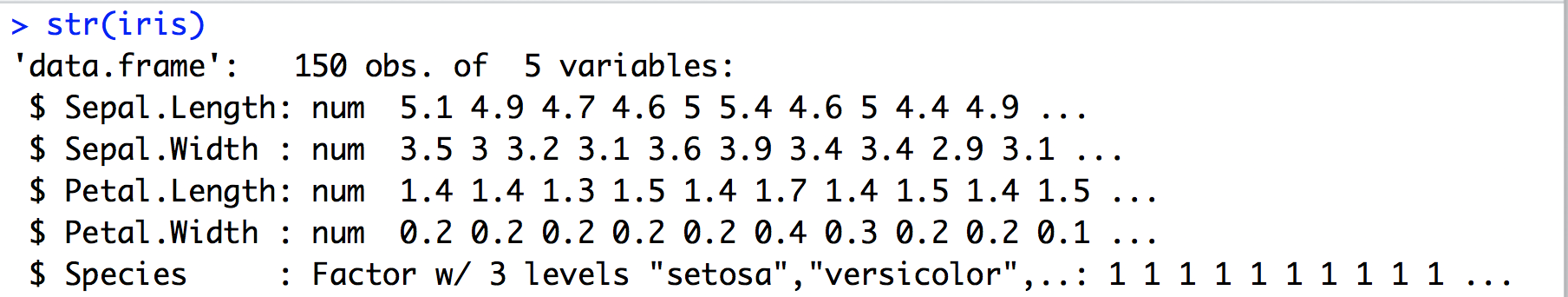


圖 12-2 str(iris) 的輸出

### 繪圖

我們今天會使用 **Base Plotting System** 這個 R 語言內建的繪圖系統來進行探索資料分析，它提供一系列的函數，每一個函數都負責繪製一種圖形，在一一檢視之前我們先用下表對這些函數有一個概觀：

|  |  |  |
| --- | --- | --- |
| 函數 | 圖形 | 範例 |
| hist() | 直方圖 | hist(rnorm(1000)) |
| boxplot() | 盒鬚圖 | boxplot(Sepal.Length ~ Species, data = iris) |
| plot(..., type = "l") | 線圖 | plot(AirPassengers, type = "l") |
| plot() | 散佈圖 | plot(cars) |
| barplot() | 長條圖 | barplot(table(mtcars$cyl)) |
| curve() | 曲線圖 | curve(sin, from = 0, to = pi \* 2) |

#### 探索數值分佈

使用 hist() 函數繪製直方圖來探索數值的分佈，hist() 函數命名是 **histogram** 的縮寫。

> hist(rnorm(1000))

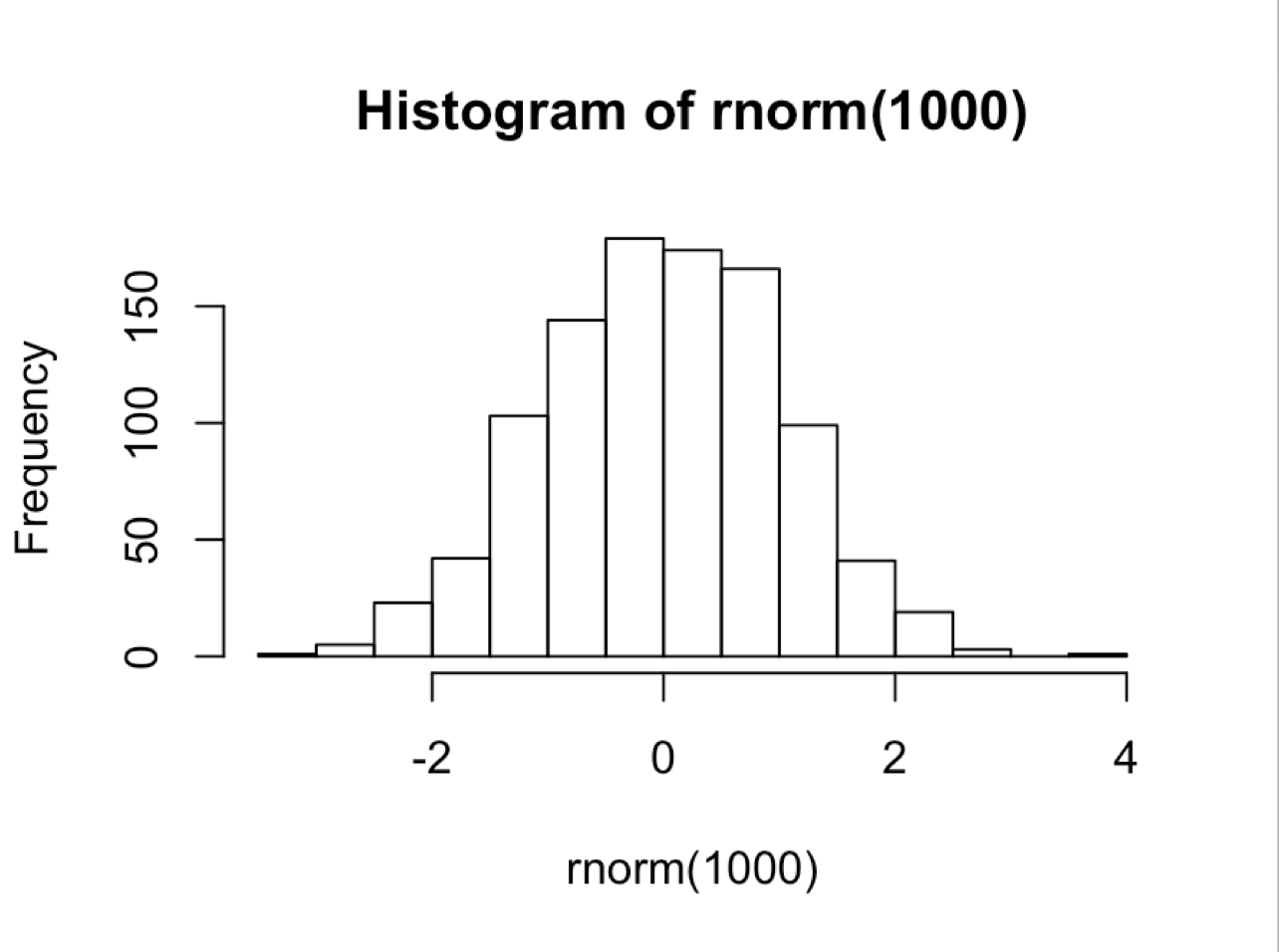


圖 12-3 hist(rnorm(1000)) 的輸出

其中 rnorm() 函數的作用是為了產生一組符合標準常態分佈（平均值為 0，標準差為 1）的數字來繪圖，其中的 1000 是指定要產生一千個符合指定分佈的數字，這個參數要稍微大一些才能夠看出明顯的鐘型外觀。

#### 探索不同類別與數值分佈的關係

使用 boxplot() 函數繪製盒鬚圖來探索不同類別與數值分佈的關係，以下面這段程式為例，我們探索內建資料 iris 中不同鳶尾花品種（Species 變數）的萼片長度（Sepal.Length 變數）分佈差異：

> boxplot(Sepal.Length ~ Species, data = iris)

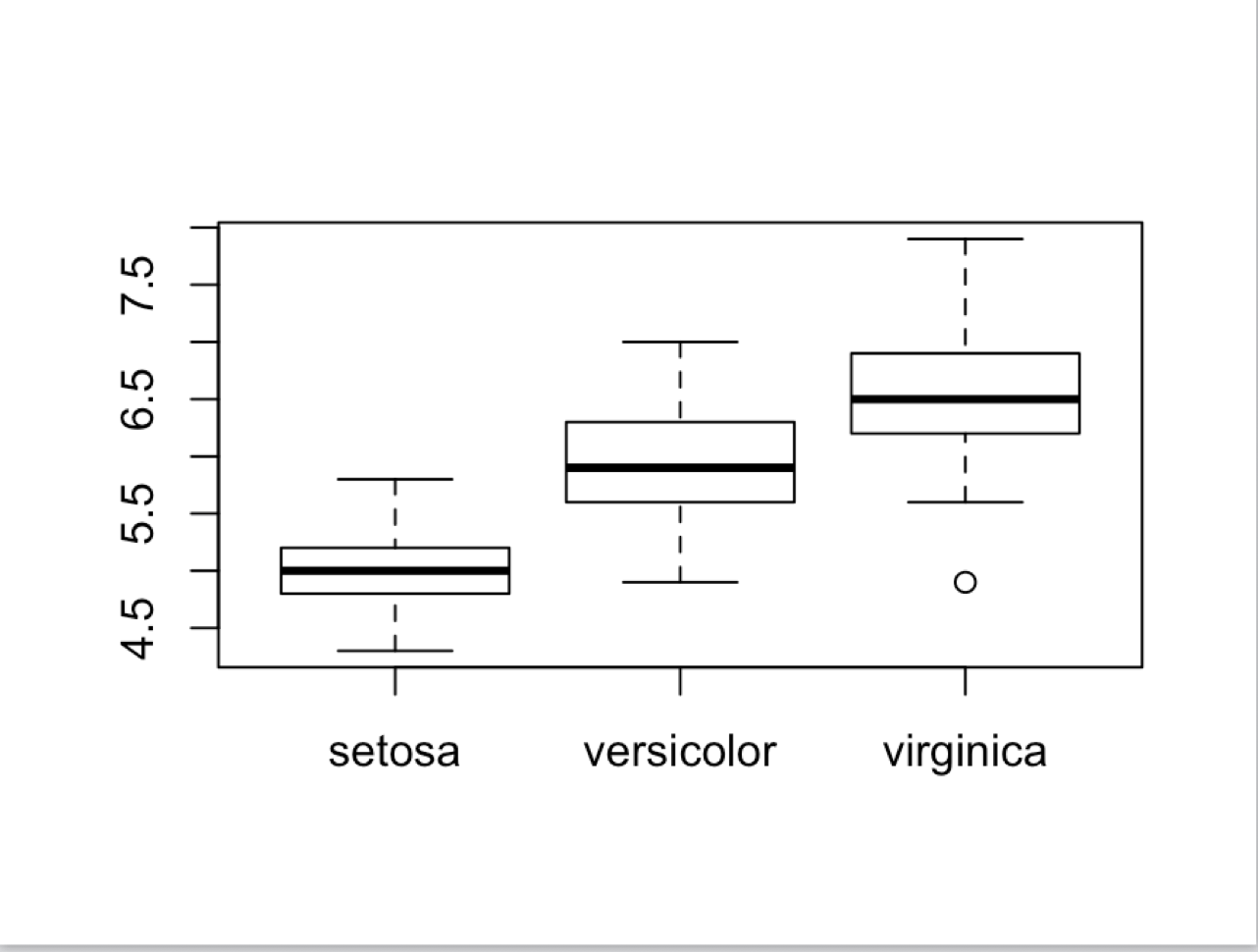


圖 12-4 boxplot(Sepal.Length ~ Species, data = iris) 的輸出

#### 探索數值與日期（時間）的關係

使用 plot(..., type = "l") 函數繪製線圖來探索數值與日期（時間）的關係：

> x <- seq(from = as.Date("2017-01-01"), to = as.Date("2017-01-31"), by = 1)  
> set.seed(123) # 確保我們得到一樣的 y  
> y <- sample(1:100, size = 31, replace = TRUE)  
> plot(x, y, type = "l")

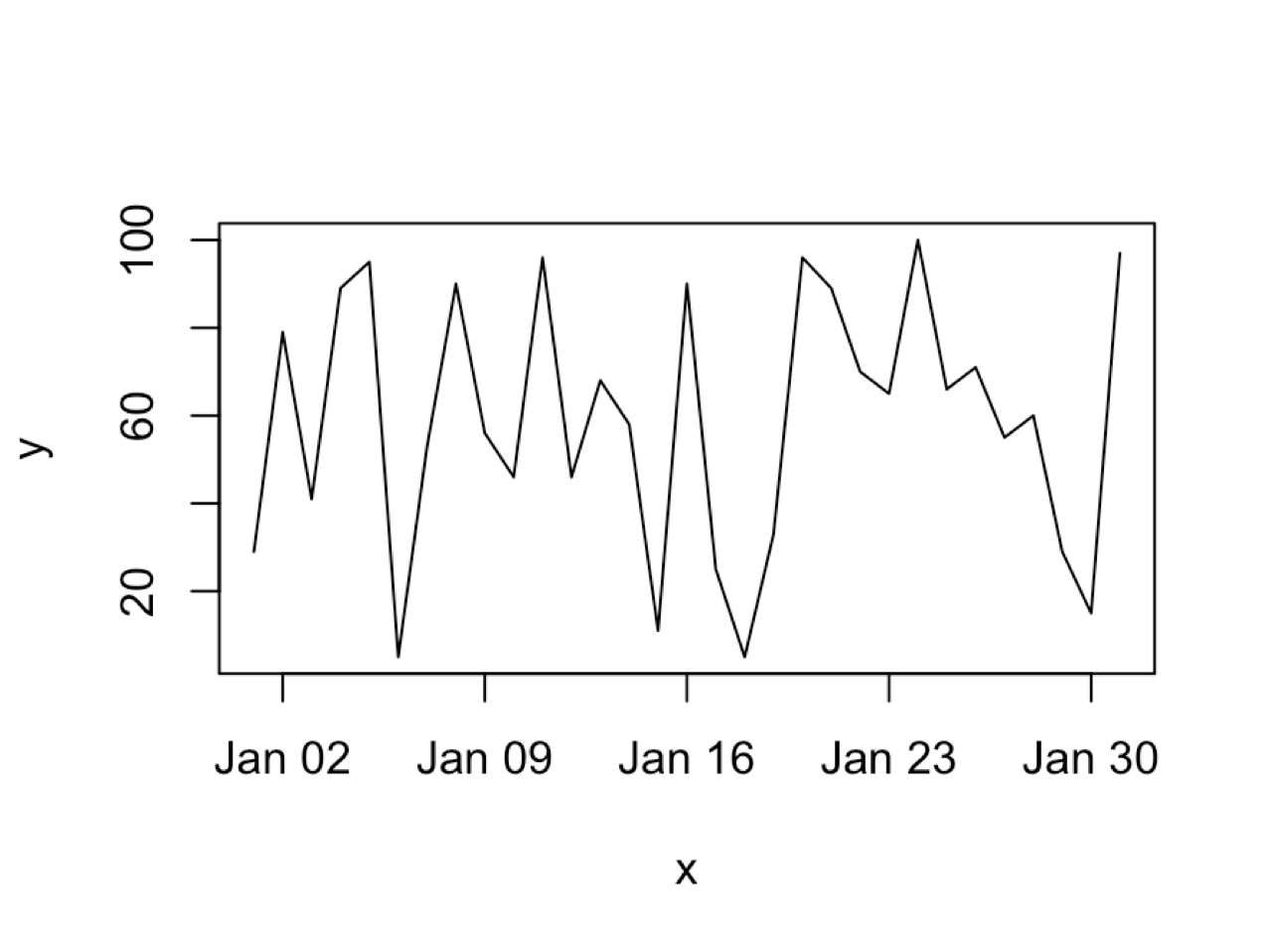


圖 12-5 plot(x, y, type = “l”) 的輸出

我們先建立一個向量 x 是 2017 年 1 月的 31 天，然後隨機從 1 到 100 之中選 31 個數值（可以重複）指派給向量 y，然後參數指定 type = "l"，亦即 **line** 的縮寫，就能夠順利繪製出線圖。

set.seed() 函數是設定隨機種子，由於我們使用了 sample() 這個函數做隨機抽樣，為了讓我們能夠得到一樣的向量 y，於是我們設定了一個整數 123 作為依據，只要在執行 sample() 函數之前執行 set.seed(123)，每次都會獲得相同的向量 y。

在 R 語言的內建資料中，像是 AirPassengers 與 LakeHuron 等，它們的資料結構是 ts 亦即 **time series** 時間序列的縮寫，它們可以直接作為 plot() 函數的輸入，就可以完成線圖的繪製（不需另外指派參數 type = "l"）。

> class(AirPassengers)  
[1] "ts"  
> class(LakeHuron)  
[1] "ts"  
> plot(AirPassengers)  
> plot(LakeHuron)

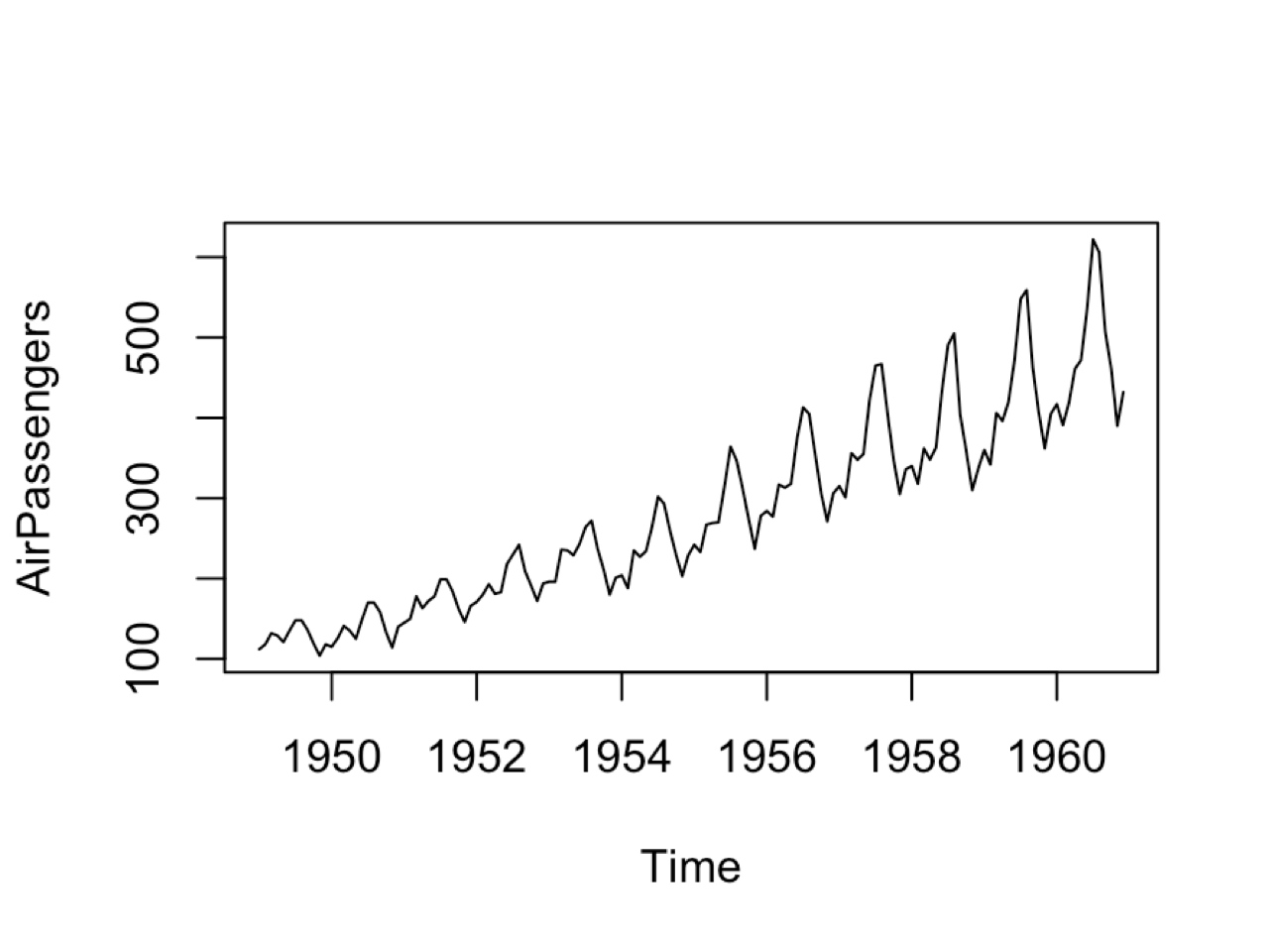


圖 12-6 plot(AirPassengers) 的輸出

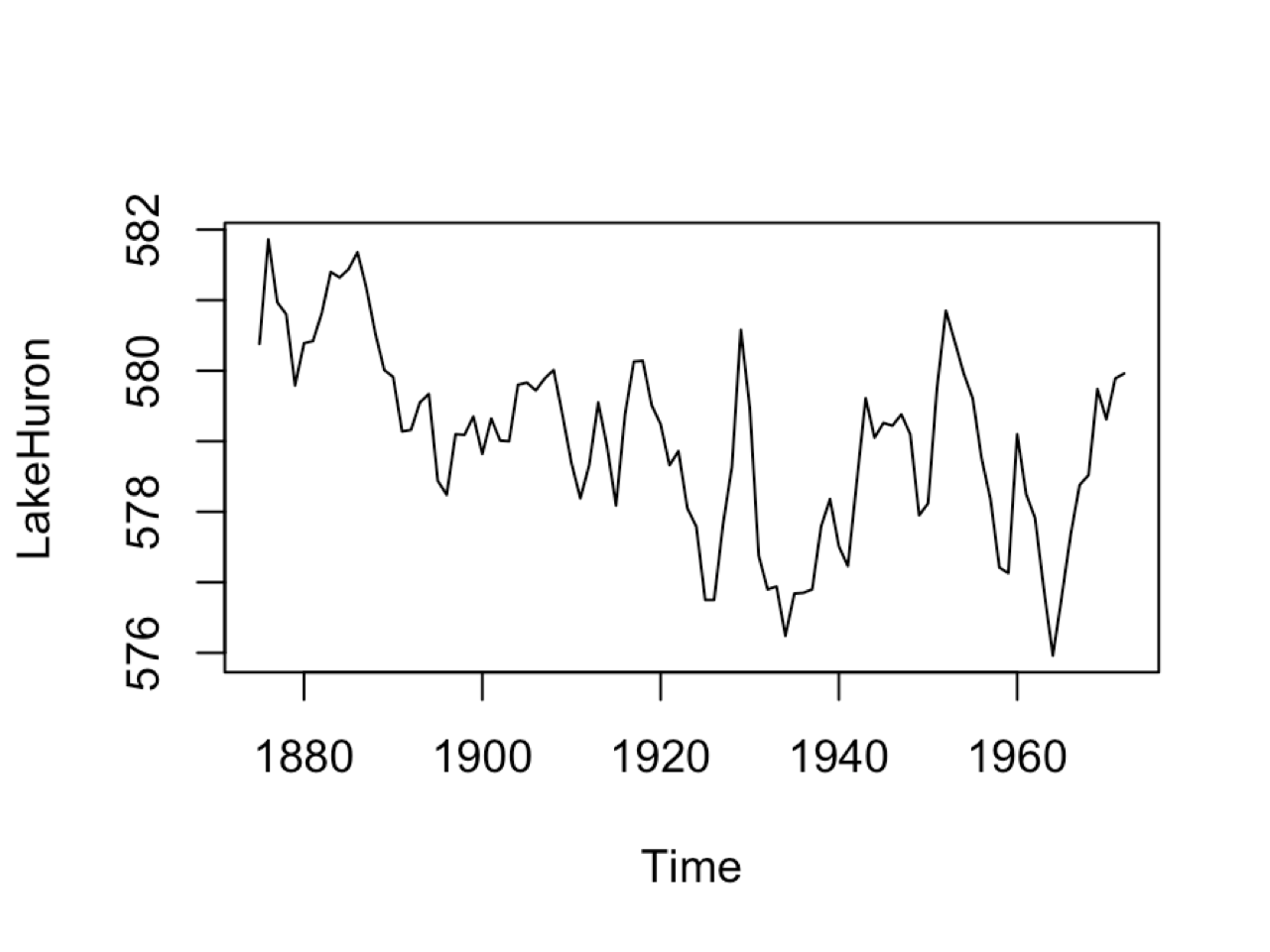


圖 12-7 plot(LakeHuron) 的輸出

#### 探索兩個數值的關係

使用 plot() 函數繪製散佈圖來探索兩個數值的關係：

> plot(cars$speed, cars$dist)

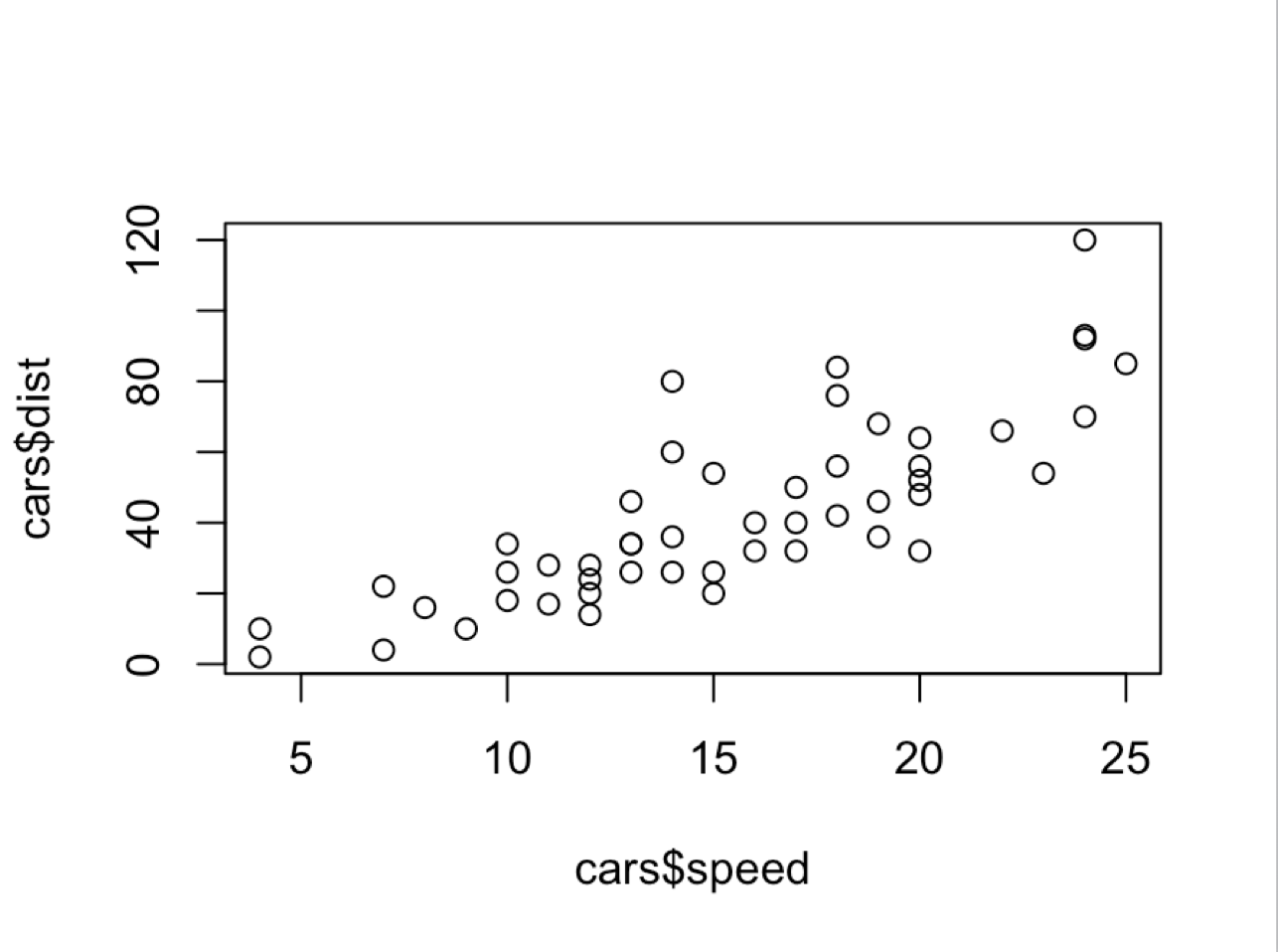


圖 12-8 plot(carsdist) 的輸出

從圖中我們可以看出數據的軌跡，隨著車速增加，所需要的煞車距離也隨之增加。

plot() 函數除了可以繪製單一散佈圖，還支援散佈圖矩陣（scatter matrix）的繪製：

> plot(iris)

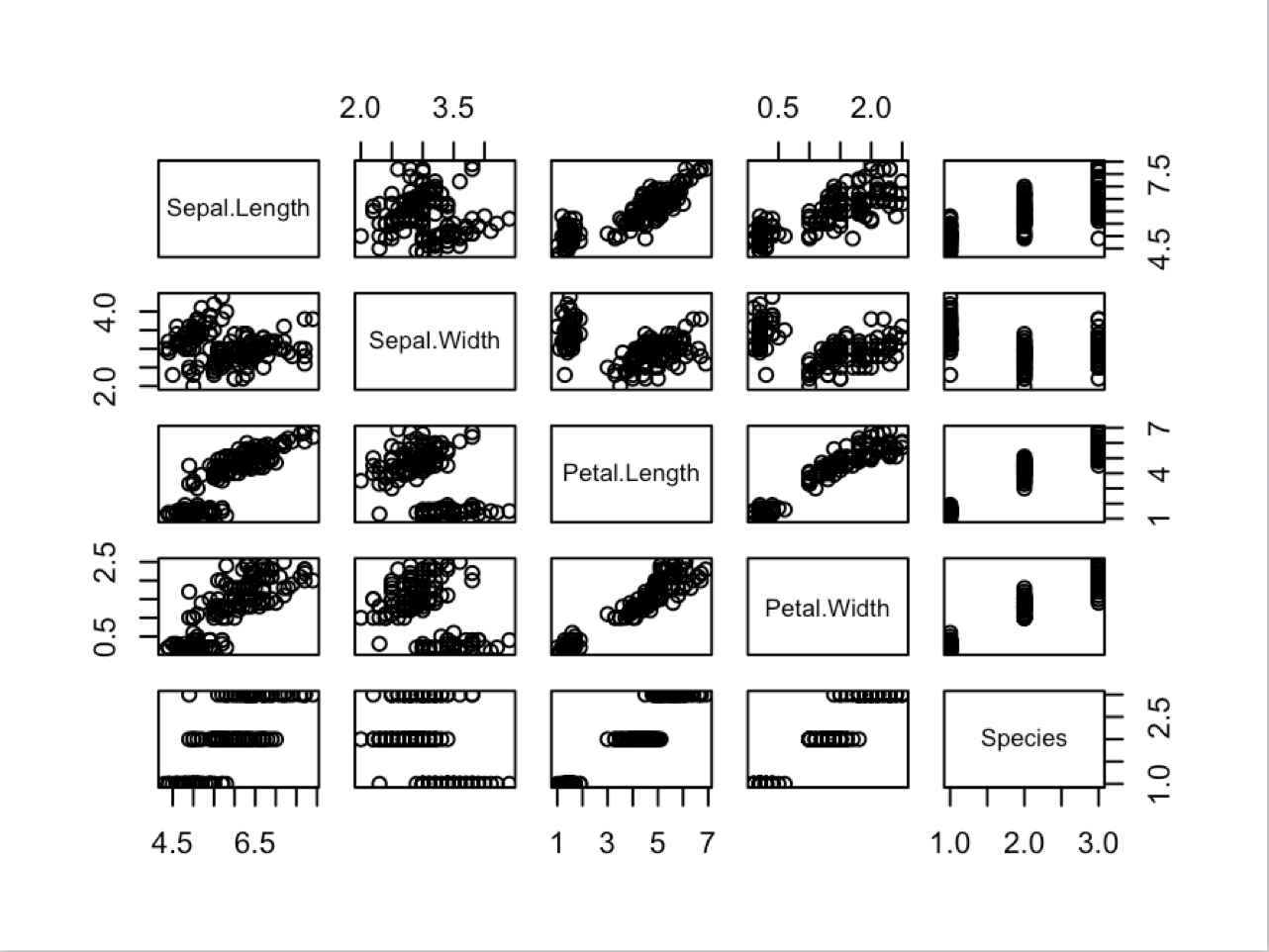


圖 12-9 plot(carsdist) 的輸出

#### 探索類別

使用 barplot() 函數繪製長條圖來探索類別的分佈，假如我們有一個向量 ice\_cream\_flavor 紀錄了 100 個人最喜歡的冰淇淋口味：

> ice\_cream\_flavor <- rep(NA, times = 100)  
> for (i in 1:100){  
+ ice\_cream\_flavor[i] <- sample(c("vanilla", "chocolate", "matcha", "other"), size = 1)  
+ }  
> ice\_cream\_flavor



圖 12-10 ice\_cream\_flavor 的輸出

在這個眼花撩亂的 ice\_cream\_flavor 向量中，我們該怎麼樣很快就得知這 100 個人最愛口味的分佈呢？

我們可以使用 table() 函數，它的作用就像是樞紐分析，可以幫我們把凌亂的資料統整起來：

> ice\_cream\_flavor <- rep(NA, times = 100)  
> for (i in 1:100){  
+ ice\_cream\_flavor[i] <- sample(c("vanilla", "chocolate", "matcha", "other"), size = 1)  
+ }  
> table(ice\_cream\_flavor)  
ice\_cream\_flavor  
chocolate matcha other vanilla   
 32 20 25 23

我們可以一目瞭然，有 32 個人最喜歡的口味是巧克力、20 個人最喜歡抹茶、25 個人喜歡其他的口味而有 23 個人最喜歡香草。這時我們就可以使用 barplot() 函數將這個清晰的結果繪畫成清楚的長條圖：

> ice\_cream\_flavor <- rep(NA, times = 100)  
> for (i in 1:100){  
+ ice\_cream\_flavor[i] <- sample(c("vanilla", "chocolate", "matcha", "other"), size = 1)  
+ }  
> barplot(table(ice\_cream\_flavor))

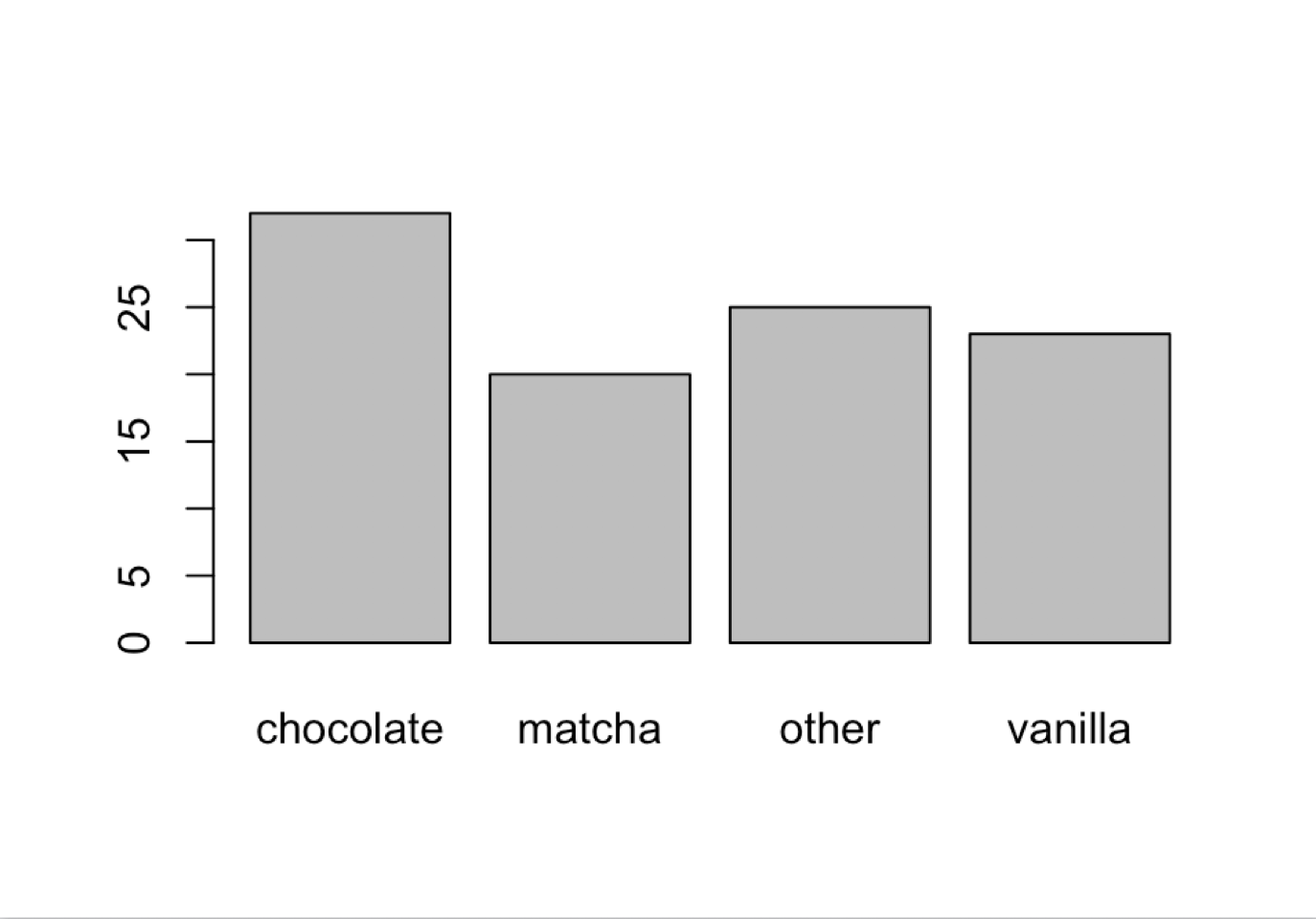


圖 12-11 barplot(table(ice\_cream\_flavor)) 的輸出

#### 繪畫函數

使用 curve() 函數將內建或者自訂的函數繪製出來，由於輸入的值域很廣，所以會有兩個參數 from 與 to 讓我們指定輸入的範圍。舉例來說我們可以將 sin() 函數在

與

之間的輸出描繪出來：

> curve(sin, from = -pi, to = pi)

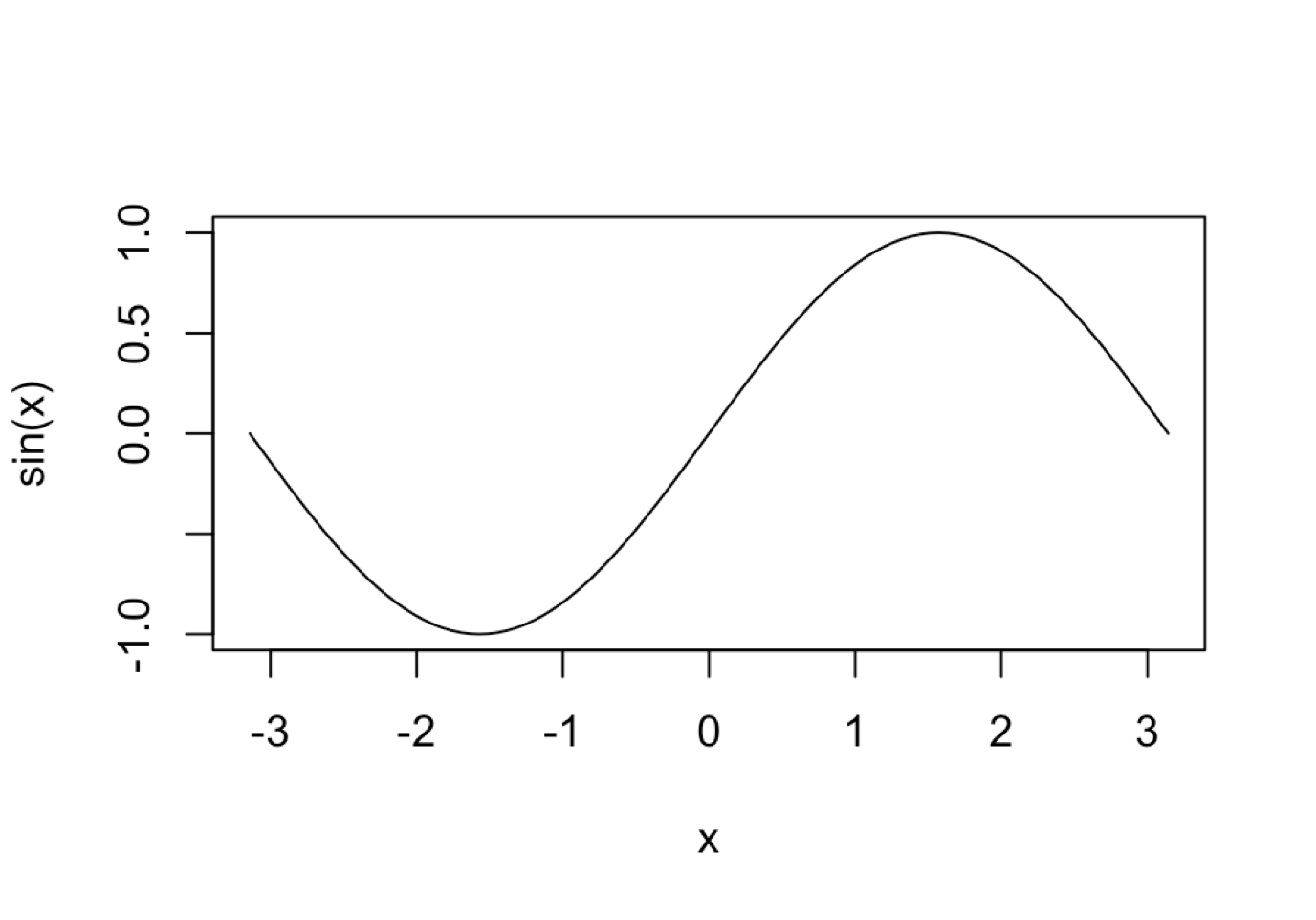


圖 12-12 curve(sin, from = -pi, to = pi) 的輸出

如果是一個自訂函數 my\_sqr()，很簡單的

，我們可以將它在 -3 與 3 之間的輸出描繪出來：

> my\_sqr <- function(x){  
+ return(x^2)  
+ }  
> curve(my\_sqr, from = -3, to = 3)

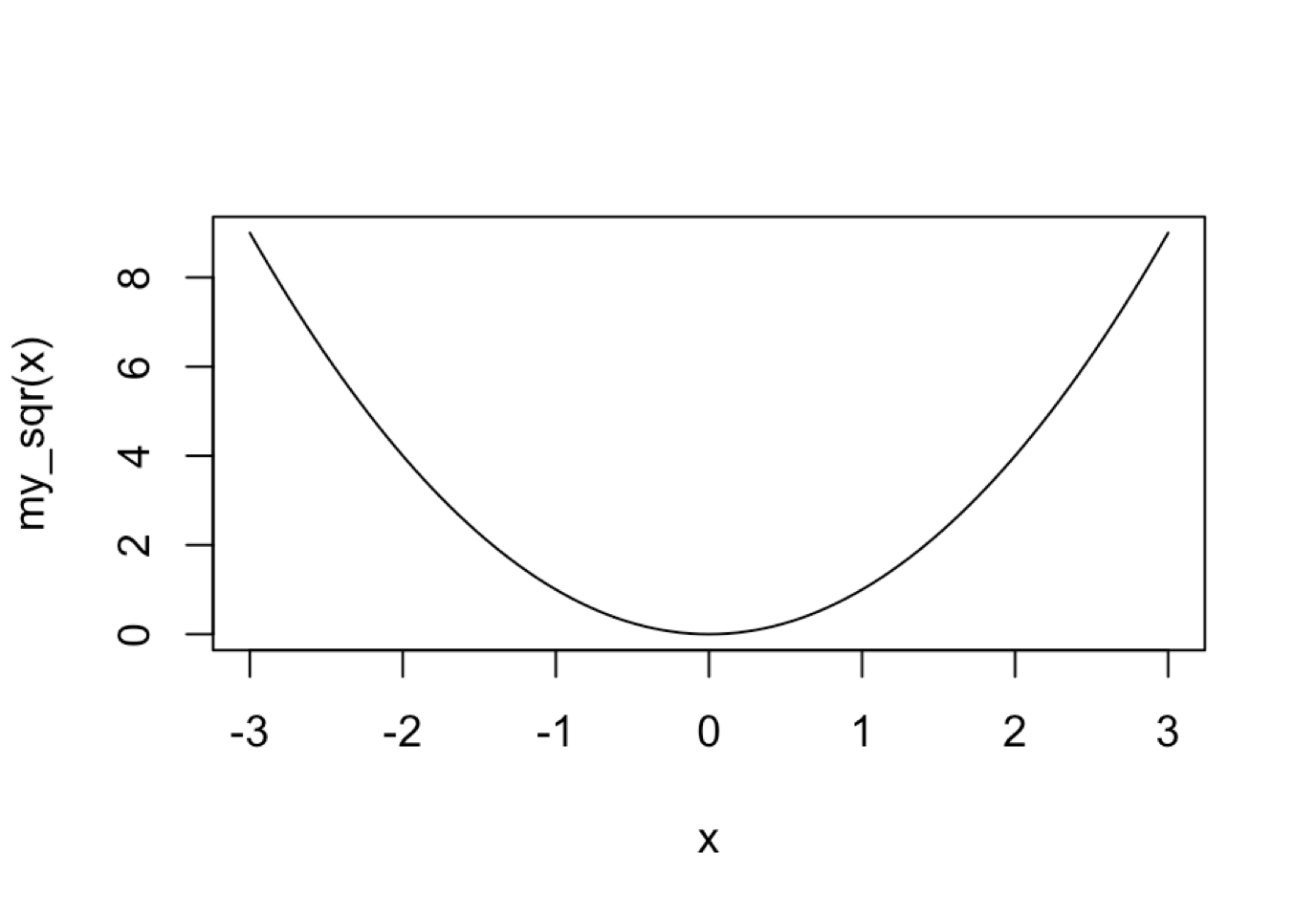


圖 12-13 curve(my\_sqr, from = -3, to = 3) 的輸出

#### 常用的自訂元素

##### 自訂標題、X 軸標籤與 Y 軸標籤

在繪圖函數中加入 main、xlab 與 ylab 這三個參數分別指定：

> plot(cars, main = "Car speed vs. braking distance", xlab = "Car speed(mph)", ylab = "Braking distance(ft)")

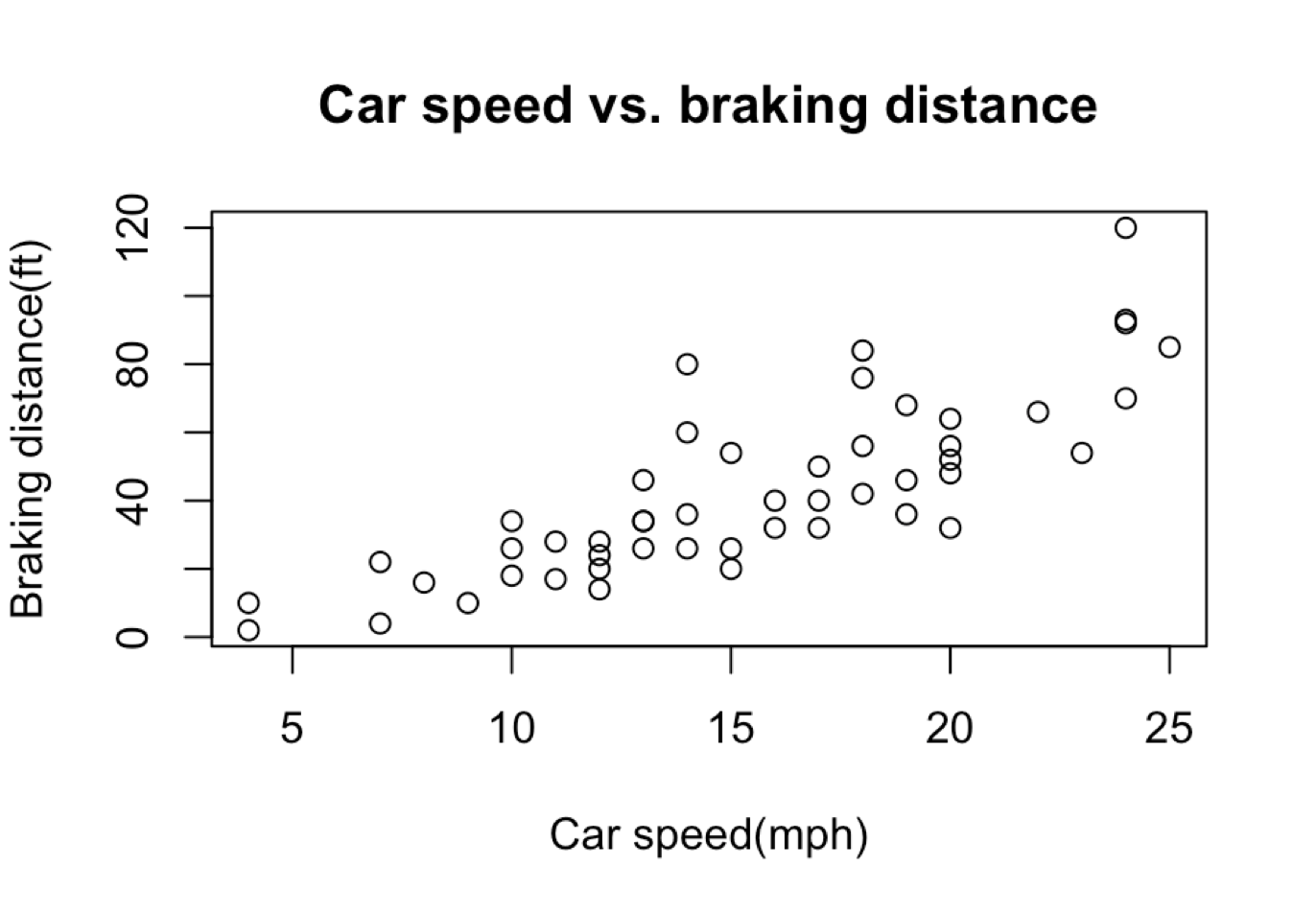


圖 12-14 自訂標題與座標軸標籤

##### 加入格線

繪圖之後呼叫 grid() 函數為圖形加上 X 軸與 Y 軸的格線：

> plot(cars, main = "Car speed vs. braking distance", xlab = "Car speed(mph)", ylab = "Braking distance(ft)")  
> grid()

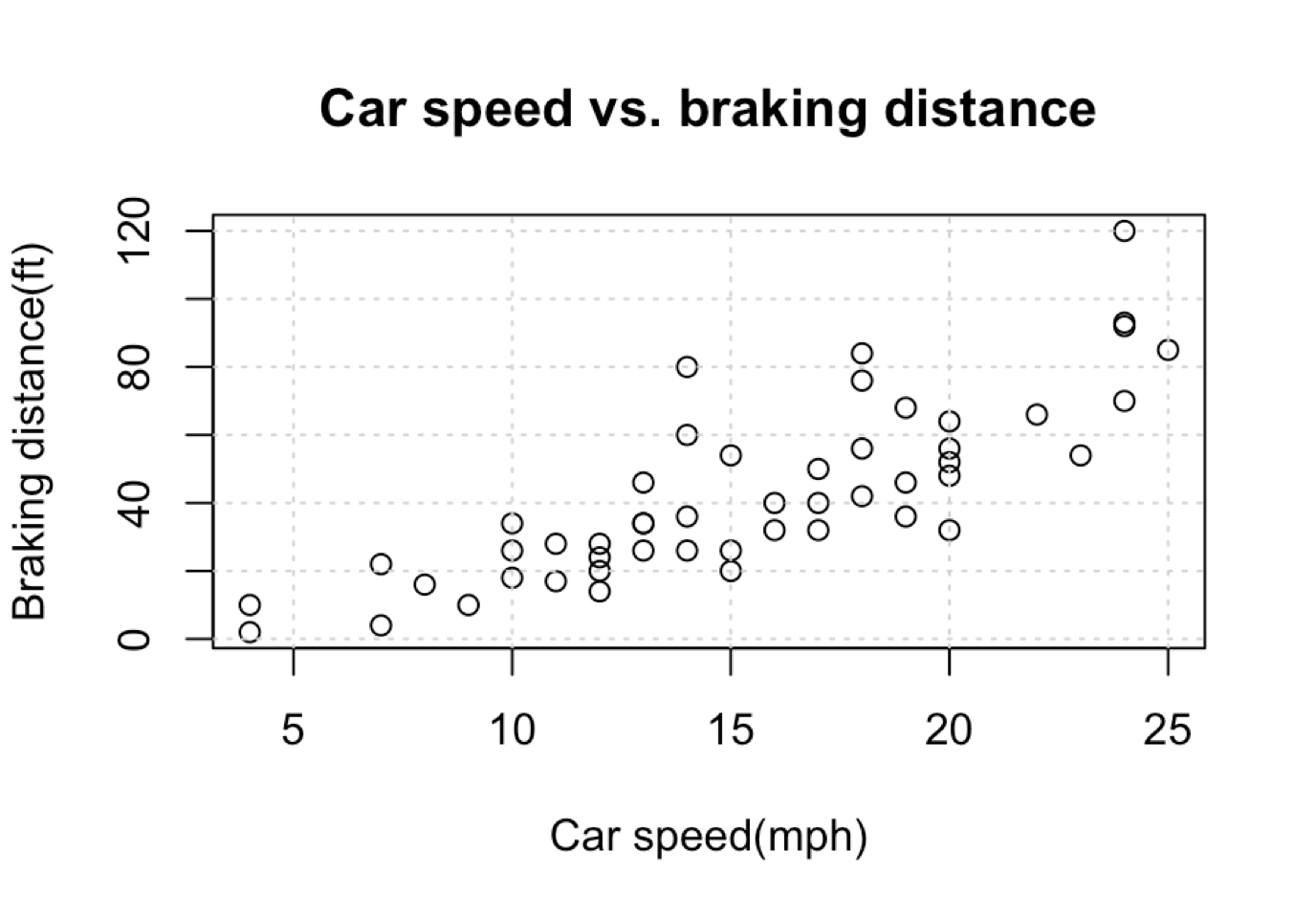


圖 12-15 加上格線

##### 調整圖形為水平方向

在繪圖函數中指定參數 horiz = TRUE：

> ice\_cream\_flavor <- rep(NA, times = 100)  
> for (i in 1:100){  
+ ice\_cream\_flavor[i] <- sample(c("vanilla", "chocolate", "matcha", "other"), size = 1)  
+ }  
> barplot(table(ice\_cream\_flavor), horiz = TRUE)

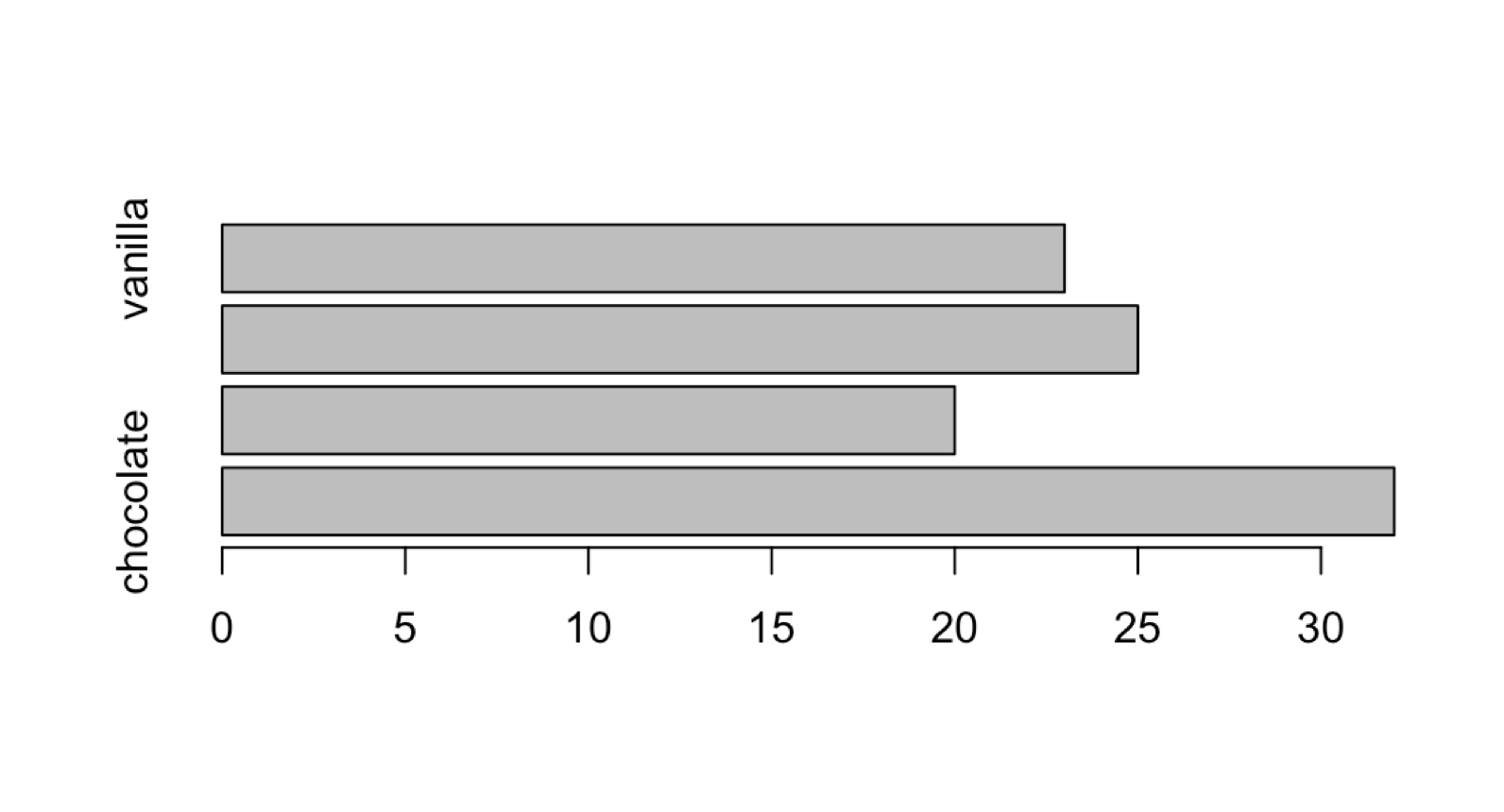


圖 12-16 水平的長條圖

##### 調整刻度顯示方向

承上，當我們將長條圖水平放置時，就發現冰淇淋的口味文字方向不適當，這時我們可以指定參數 las = 1 讓文字與座標軸垂直。

> ice\_cream\_flavor <- rep(NA, times = 100)  
> for (i in 1:100){  
+ ice\_cream\_flavor[i] <- sample(c("vanilla", "chocolate", "matcha", "other"), size = 1)  
+ }  
> barplot(table(ice\_cream\_flavor), horiz = TRUE, las = 1)

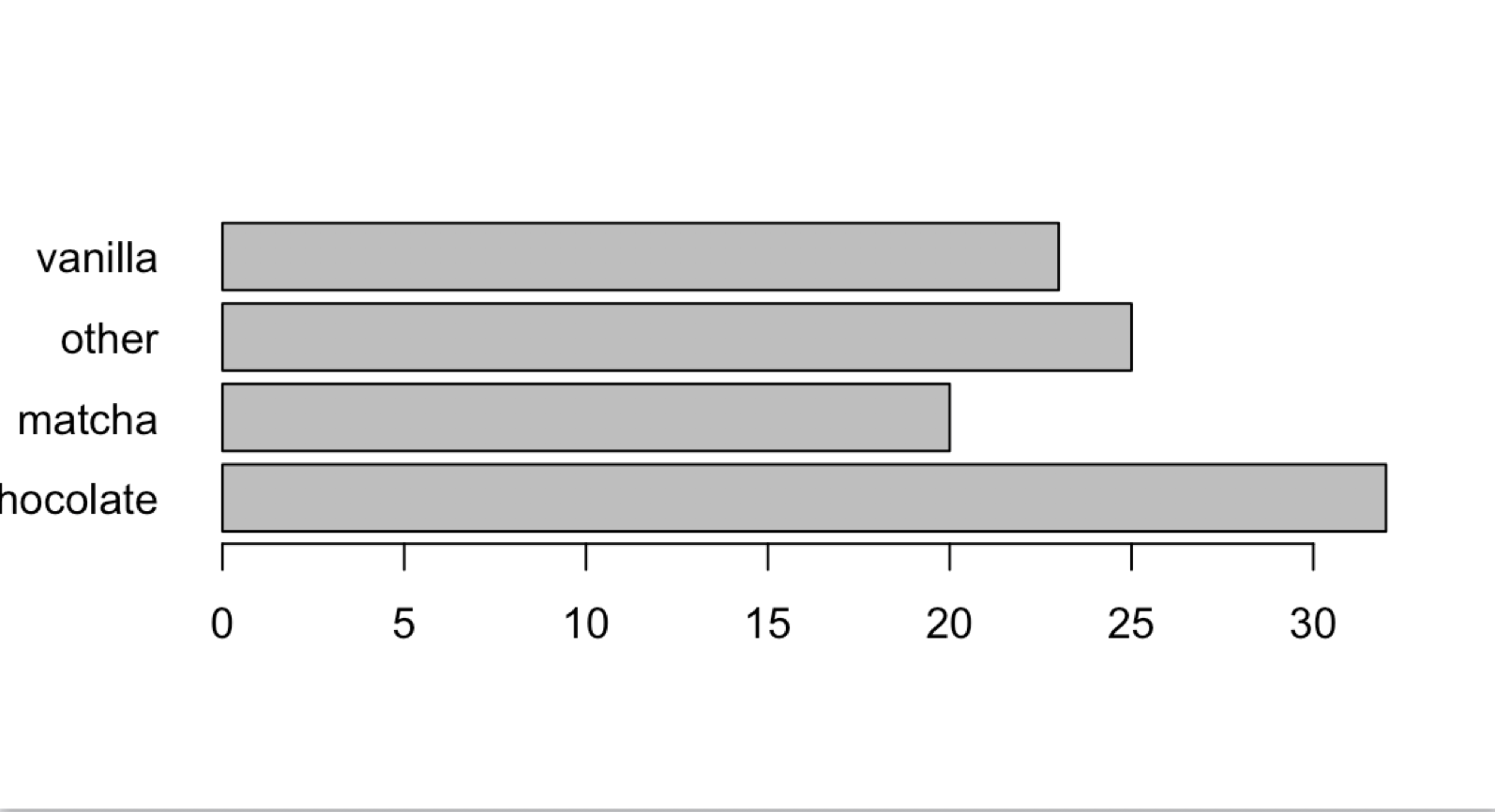


圖 12-17 調整刻度顯示方向長條圖

##### 調整刻度文字大小

承上，我們的 chocolate 文字被切去了開頭，我們可以指定參數 cex.name 將 Y 軸的刻度文字縮小一些：

> barplot(table(ice\_cream\_flavor), horiz = TRUE, las = 1, cex.name = 0.8)

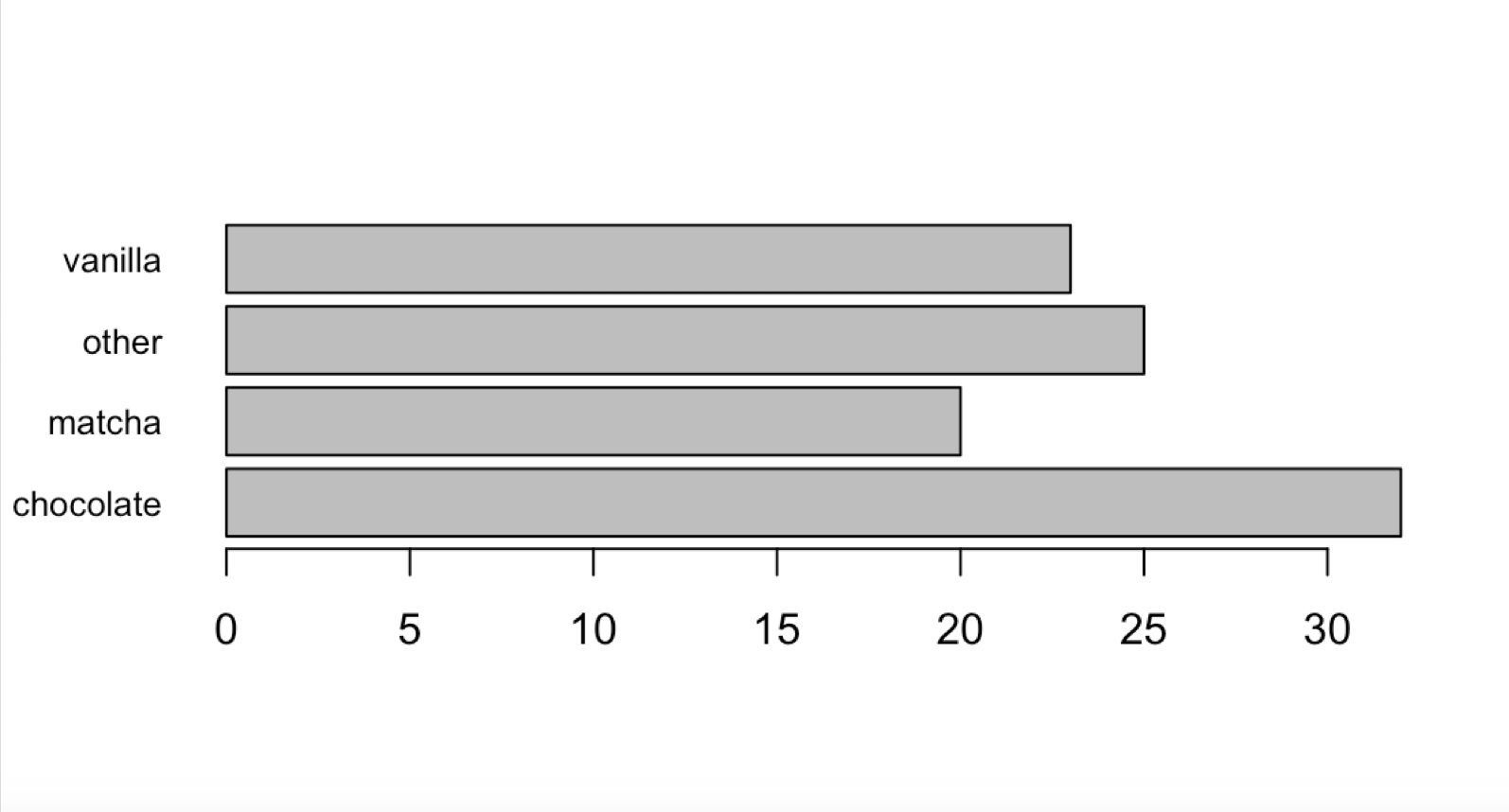


圖 12-18 調整 cex.name 長條圖

假如想調整 X 軸的刻度文字，我們可以指定 cex.axis：

> barplot(table(ice\_cream\_flavor), horiz = TRUE, las = 1, cex.name = 0.8, cex.axis = 1.2)

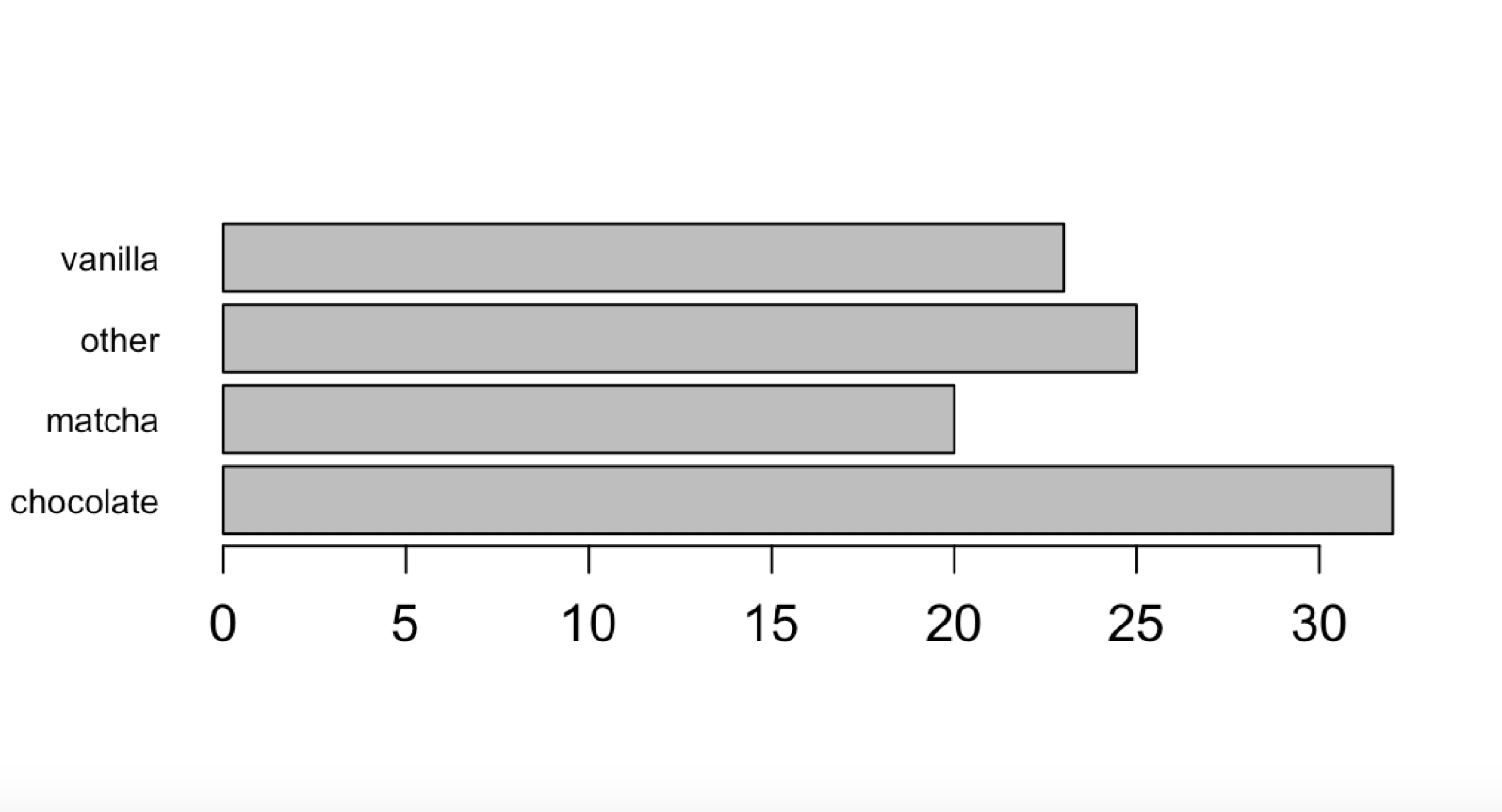


圖 12-19 調整 cex.axis 長條圖

其中 cex 是 **character expansion factor** 的縮寫，指的是相對於預設值 **1** 的文字縮放倍數。

##### 在直方圖上加上密度曲線

使用 density() 函數搭配 lines() 函數產生密度曲線，並且調整直方圖的參數 freq = FALSE：

> norm\_dist <- rnorm(1000)  
> hist(norm\_dist, freq = FALSE)  
> lines(density(norm\_dist))

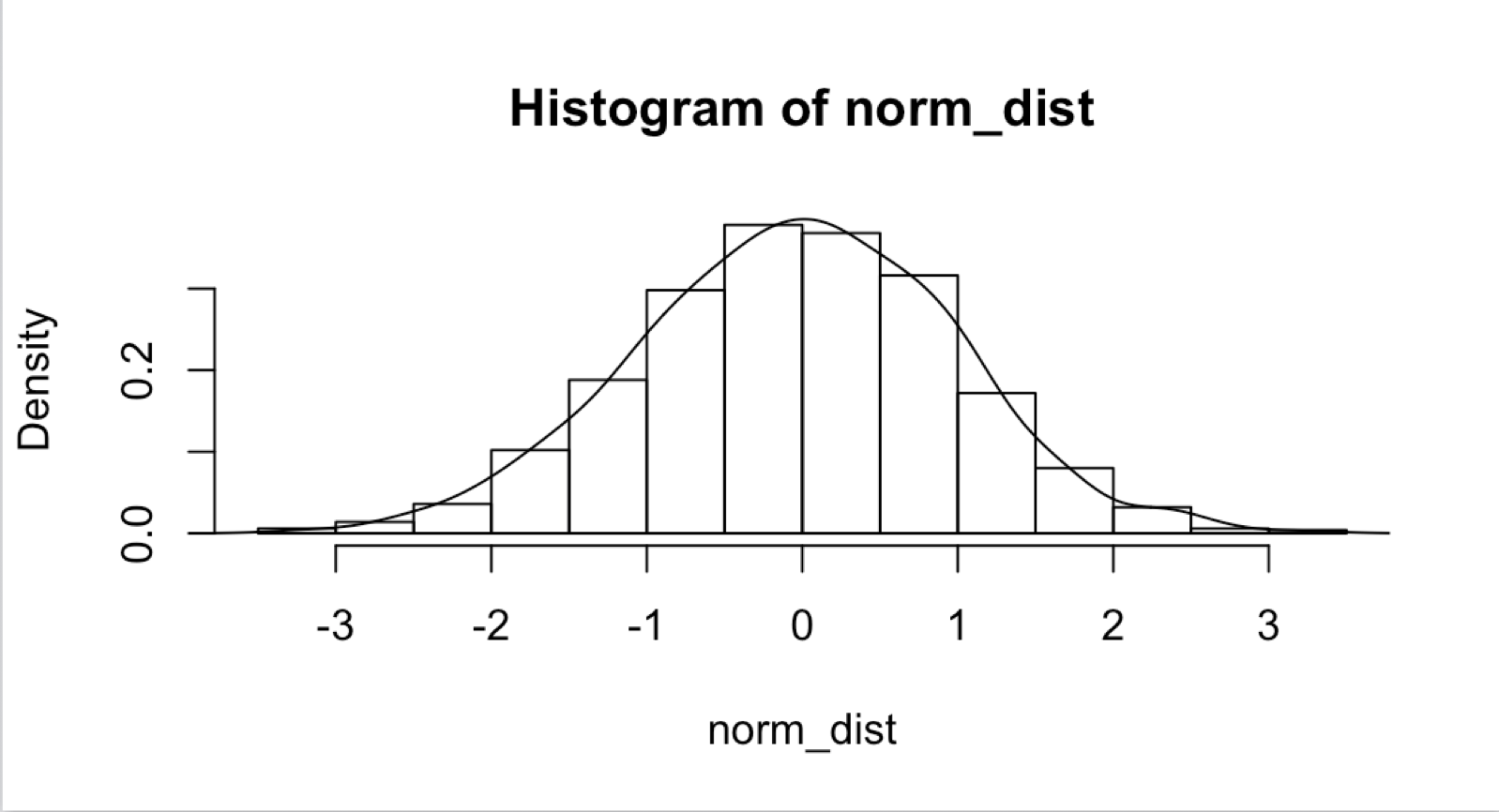


圖 12-20 直方圖加上密度曲線

##### 調整資料點的形狀與顏色

指定參數 pch 與 col 可以分別調整資料點的形狀與顏色。pch 是 **plotting character** 的縮寫，而 col 是 color 的縮寫：

> plot(cars, pch = 2, col = "red") # 資料點改成紅色的空心三角形

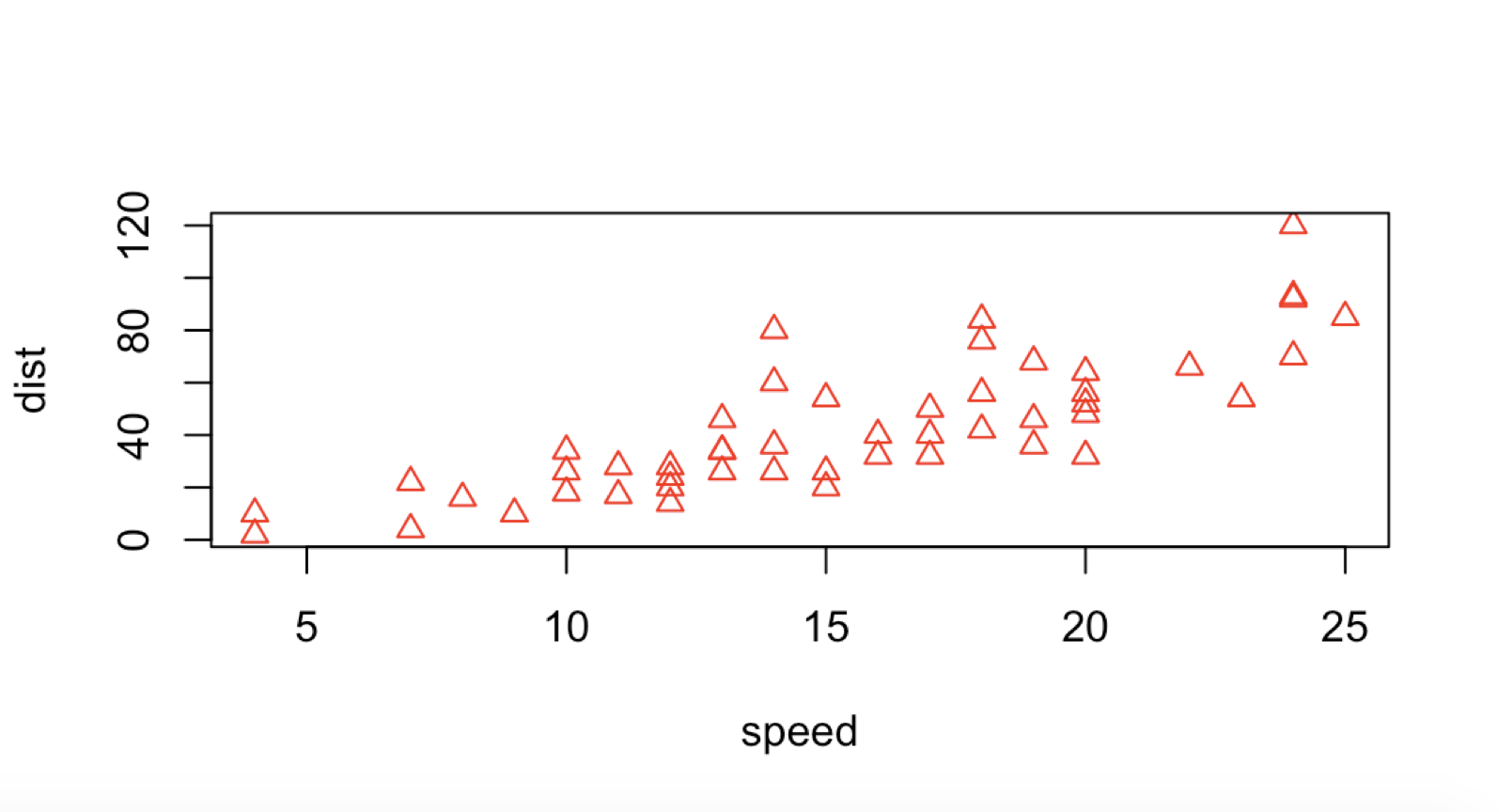


圖 12-21 調整資料點

實務上更多的應用會是以不同的**類別**區分資料點的形狀與顏色：

> iris\_pch <- c(1, 2, 3)[as.numeric(iris$Species)]  
> plot(iris$Sepal.Length, iris$Sepal.Width, col = iris$Species, pch = iris\_pch)

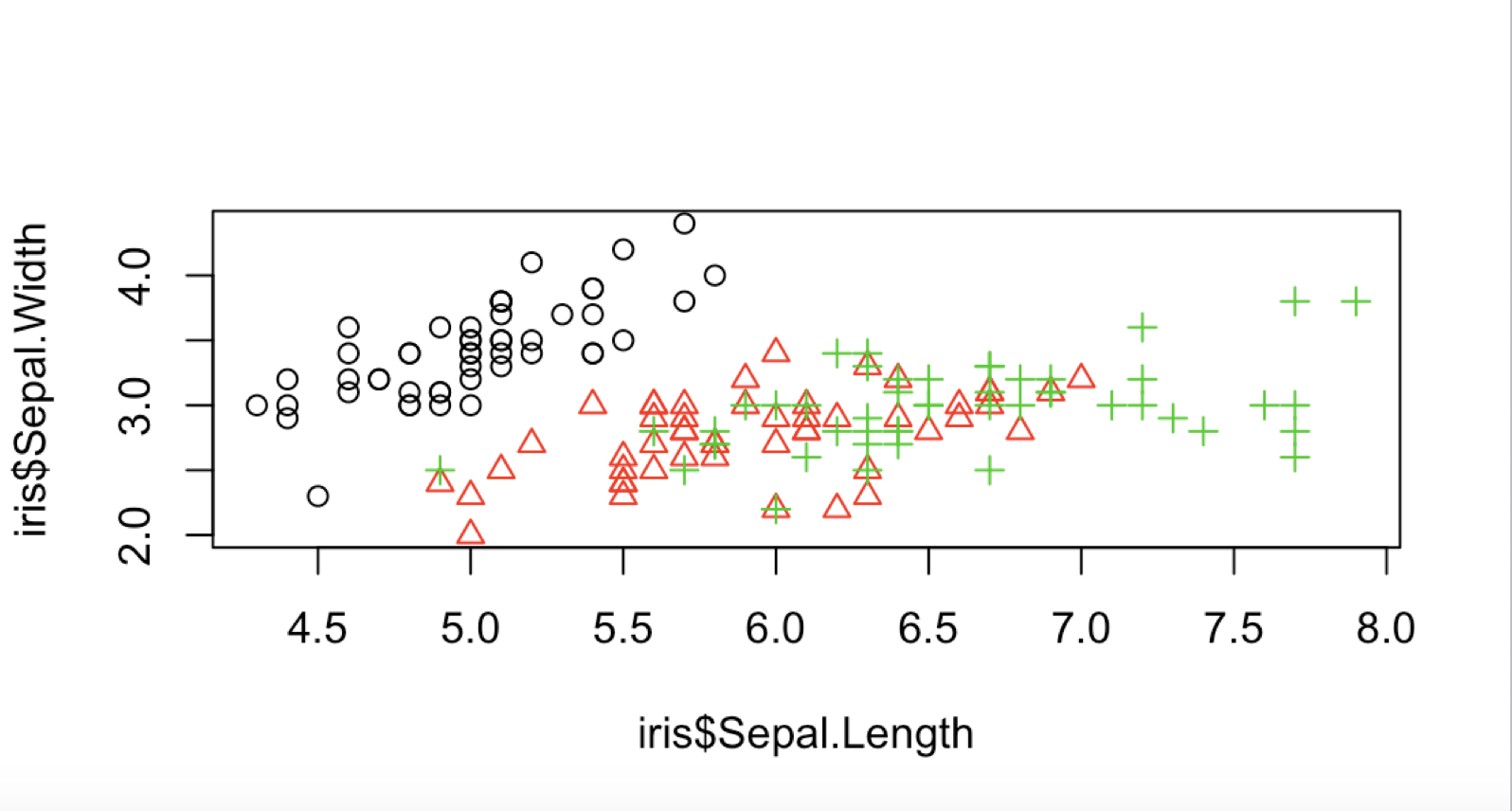


圖 12-22 調整資料點（2）

如果對於 pch 與 col 的參數感興趣，可以參考：[Graphical Parameters](http://www.statmethods.net/advgraphs/parameters.html)

##### 繪畫多個圖形

在呼叫繪圖函數之前先呼叫 par(mfrow = c(m, n)) 將畫布切割為 mxn 個區塊，mfrow 是 **matrix of figures entered row-wise** 的縮寫：

> par(mfrow = c(2, 2))  
> boxplot(iris$Sepal.Length ~ iris$Species, main = "Sepal length by species")  
> boxplot(iris$Sepal.Width ~ iris$Species, main = "Sepal width by species")  
> boxplot(iris$Petal.Length ~ iris$Species, main = "Petal length by species")  
> boxplot(iris$Petal.Width ~ iris$Species, main = "Petal width by species")

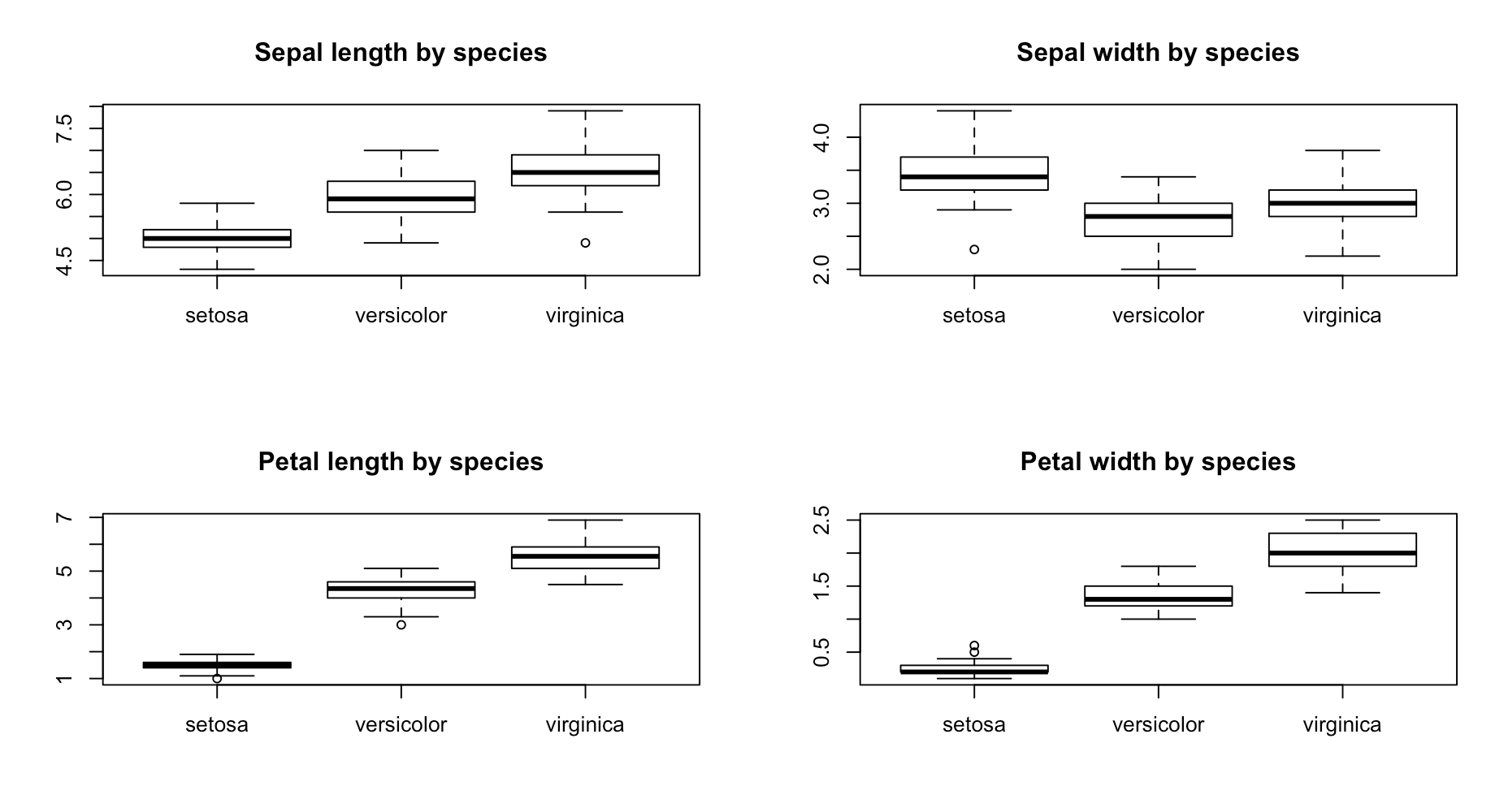


圖 12-23 多個圖形

#### 輸出圖形

我們可以利用 RStudio 介面中右下角的 Plots 頁籤點選 Export 輸出繪圖成為圖檔：

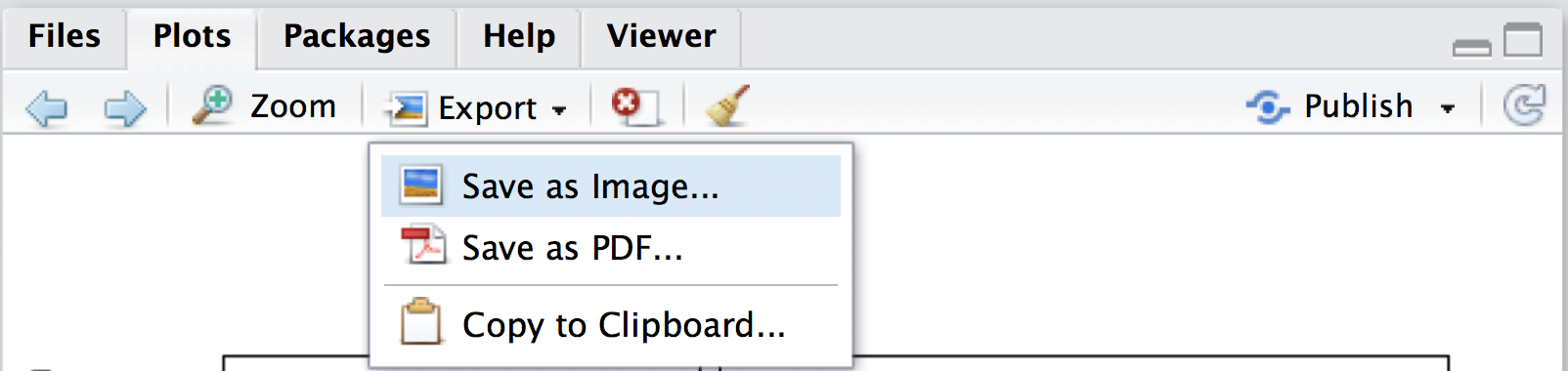


圖 12-24 輸出圖形

可以指定輸出圖檔的格式、輸出圖檔的路徑、圖檔檔名、長與寬。

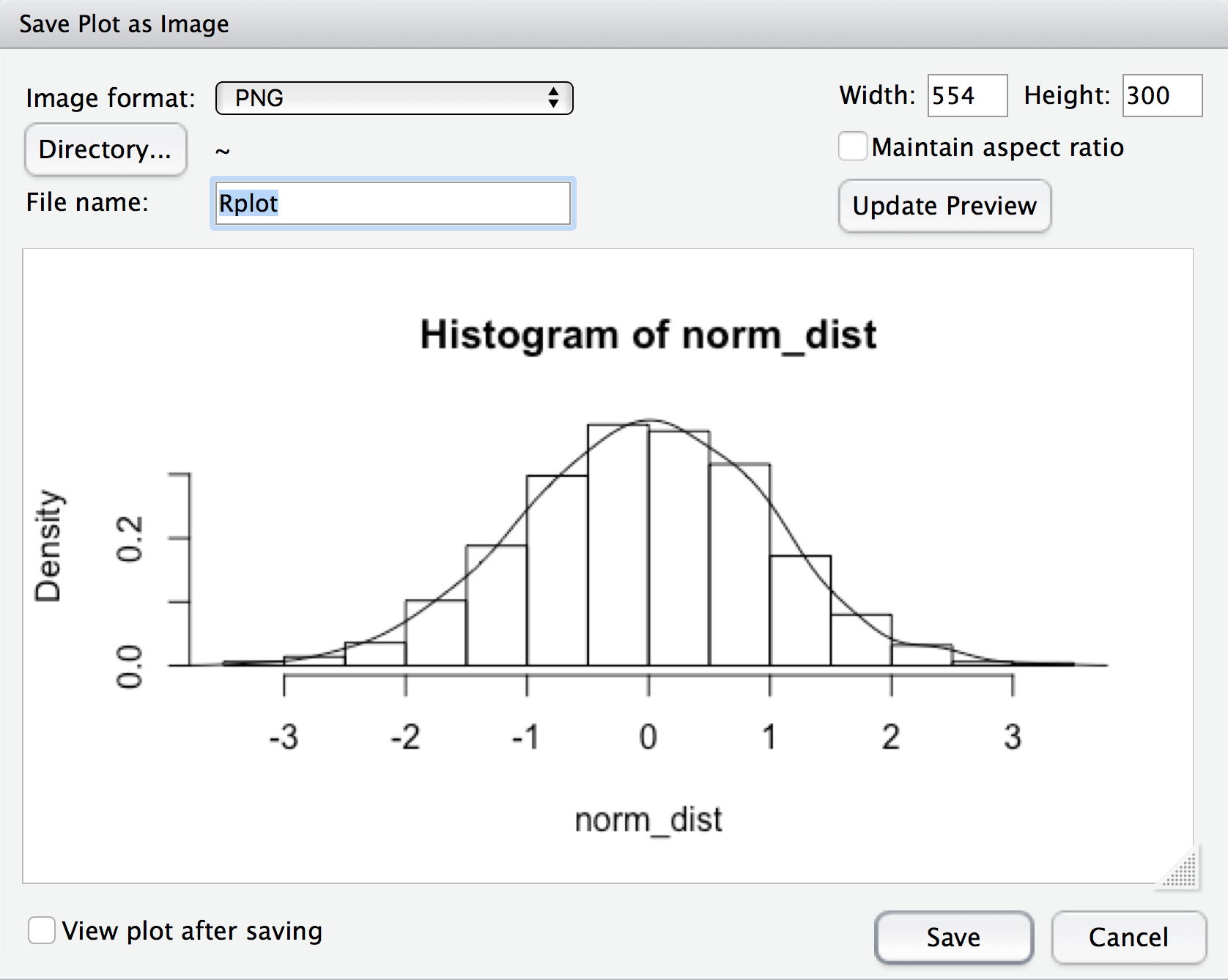


圖 12-25 輸出圖形（2）

### 滄海一粟

我們雖然已經介紹了一定程度關於 **Base Plotting System** 的技巧，但是這些功能都只是其中非常小的一部分，因為繪圖可以調整的參數與功能實在太豐富了，在有限的章節內容我們只能夠探討探索性分析常用的繪圖需求。實務上在使用繪圖功能時多半會伴隨大量的文件、Google 與 StackOverflow 查詢，所以如果您在一開始遭遇到很多的困難，這是再正常不過的情形了，請耐住性子，多畫幾次就會漸入佳境的！

### 小結

好啦！第十二天的內容就是這麼多，R 語言用來進行探索性分析的工具除了函數回傳的資訊，更大一部分是使用繪圖來協助，今天說明的是 **Base Plotting System**，希望您還喜歡這個內建的繪圖系統！

### 練習題

###### 將一個畫布切割成為 2X2 個區塊，並使用 **Base Plotting System** 繪製任意四種圖形。

### 延伸資訊

* [R Graphics Cookbook](http://shop.oreilly.com/product/0636920023135.do)
* [Graphical Parameters](http://www.statmethods.net/advgraphs/parameters.html)