Functions

Gerardo Ferrara

Master in Economics and Complexity, Collegio Carlo Alberto

Fall 2013

Introduction

One of the aspects of R which makes these packages different from other statistical packages is that they are based on the computer language S. In other words, we have an entire computer language at our disposal when we program in R which allows us to easily write any function that we want to implement.

$$function.name <\!- function(arguments)\{ \\ purpose\ of\ \ function \\ \}$$

Furthermore:

- Functions can be passed as arguments to other functions.
- Functions can be nested, so that you can define a function inside of another function.
- The return value of a function is the last expression in the function body to be evaluated.

Example

Let's build a function, called *f1*, which adds a pair of numbers:

$$
\begin{aligned}
> f1 <\!\!- \mathit{function}(x,\ y)\{ \\
x + y \\
\} \\
> f1(3,\ 4) \\
[1]\ \ 7
\end{aligned}
$$

Multiple Results

If we have a function which performs multiple tasks and therefore has multiple results to report then we have to include a **return** statement (with **c()**) inside the function is order to see all the results.

$$> f.mult1 <\!-function(x,\ y)\{$$

$$\begin{aligned} &p1 <\!-2 * x + y \\ &p2 <\!-x + 2 * y \\ &p3 <\!-2 * x + 2 * y \\ &p4 <\!-x/y \\ &return(c(p1,\ p2,\ p3,\ p4)) \quad \} \end{aligned}$$

BEWARE: The **return** signals that a function should exit and return a given value. Thus, it is important to include the **return** statement at the end of the function!

Multiple Results (cont.)

Moreover, when we have a function which performs multiple computations, it is often useful to save the results in a **list**.

$$
\begin{aligned}
> f.mult2 <\!-function(x,\ y)\{ \\
& p1 <\!-2 * x + y \\
& p2 <\!-x + 2 * y \\
& list(c(result1 = p1,\ result2 = p2)) \\
& \}
\end{aligned}
$$

Now we can access the results using either the list indices (double square brackets) or the names of the elements in the list.

$$
\begin{aligned}
&> f.mult2(2,\ 5)\$result2 \\
&[1]\ \ 12 \\
&> f.mult2(2,\ 5)\$[[2]] \\
&[1]\ \ 12
\end{aligned}
$$

Note: The variables p1 and p2 exist only inside the function f2 and you can not refer to them outside the function. Thus, we can not make a call to **f.mult2(2, 5)$p1**.

Modifying an already existing function

One of the most common tasks is to modify an existing function by changing only one or a few of the parameters in the existing function. Now we change the default symbol for a scatter plot from an empty circle to a solid square in a function called *my.plot*:

$> x <- rnorm(100)$

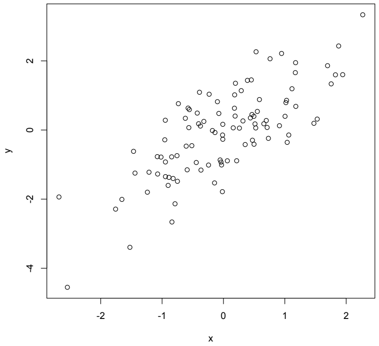
$> y <- x + rnorm(100)$

$> my.plot <- function(..., \; pch.new = 15)\{$

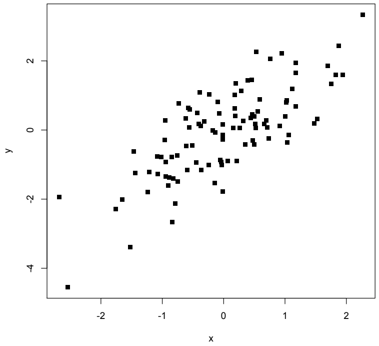
$$plot(..., \; pch = pch.new)$$

$$\}$$

Modifying an already existing function (cont.)



y
x
-4
-2
0
2
-2
-1
0
1
2



y
x
-4
-2
0
2
-2
-1
0
1
2