Selecting Variables and Observations

In SAS and SPSS, variable selection is done using a very simple yet flexible set of commands using variable names, and the selection of observations is done using logic. Combining the two approaches is quite simple. For example, selecting the variables workshop and q1 to q4 for the males only is done in SAS with

```
PROC PRINT;
  VAR workshop q1-q4;
  WHERE gender="m";
```

SPSS uses a very similar approach:

```
TEMPORARY.
SELECT IF (gender EQ "m").
LIST workshop q1 TO q4.
```

In the previous two chapters, we focused on selecting variables and observations separately, and we examined a very wide range of ways to do both. Different books and help files use various approaches, so it is important to know the range of options to perform these basic tasks in R. However, you can still use the approach that is already most familiar to you: using names to select variables and logic to select observations.

As an example, we will use the various methods to select the variables workshop and q1 to q4 for only the males.

The explanations in this chapter are much sparser. If you need clarification, see the detailed discussions of each approach in the previous two chapters.

9.1 The subset Function

Although you can use any of the methods introduced in the previous two chapters to select both variables and observations, variables are usually chosen
by name and observations by logic. The `subset` function lets you use that combination easily.

When selecting variables, `subset` allows you to use the colon operator on lists of contiguous variables, like `gender:q4`. Variable selections that are more complex than a single variable or two contiguous variables separated by a colon must be combined with the `c` function.

When selecting observations, you perform logic like `gender == "m"` without having to use `which(gender == "m")` to get rid of the observations that have missing values for gender. The logic can be as complex as you like, so we can select the males who are happy with their workshop using `gender == "m" & q4 == 5`. Note that the result of a logical condition is always a single logical vector, so you never need the `c` function for logic. See Table 10.3, “Logical Operators,” for details.

We can perform our selection by nesting the `subset` function directly within other functions:

```r
summary(
  subset(mydata, 
    subset = gender == "m", 
    select = c(workshop, q1:q4) 
  )
)
```

Since R allows you to skip the names of arguments as long as you have them in proper order, you often see `subset` used in the form

```r
summary(
  subset(mydata, gender == "m", 
    c(workshop, q1:q4) 
  )
)
```

If you plan to use a subset like this repeatedly, it would make more sense to save the subset in a new data frame. Here we will add the `print` function just to make the point that selection is done once and then used repeatedly with different functions. Here I am using the name `myMalesWQ` to represent the males with workshop and the q variables.

```r
myMalesWQ <- subset(mydata, 
  subset = gender == "m", 
  select = c(workshop, q1:q4) 
)
print(myMalesWQ)
summary(myMalesWQ)
```

Performing the task in two steps like that often makes the code easier to read and less error prone.