Automation: `for` Loops

Michael Malick

Updated: 15 May 2013

`for` Loops

`for` loops are used when you want to repeat an operation or computation multiple times

```
for(i in 1:n) {
     statement(i)
     statement ...
}
```

- `1:n` can be any vector
- If you find yourself copying and pasting code a lot you should probably be using a for loop (or an `apply` type function)

Example: `for` Loop

```
1:10 # a vector
cat("Iteration", 1:10, "\n")

for(i in 1:10) {
    cat("Iteration", i, "\n")
}
```

Whats Happening:

1. R runs the code in the braces and plugs in the first element of the vector (in this case 1) wherever `i` is found

2. R then loops over the code in the braces again and plugs in the second element of the vector (in this case 2) wherever `i` is found

3. ...

Example: `for` Loop

In this example, we will save the output of each iteration to a matrix

```
# Create a matrix
mat <- matrix(NA, 10, 10)


for(i in 1:10) {
     x <- rep(i, times = 10)
     mat[, i] <- x
}
```

Example: `for` Loop

In this example, we will compute some summary statistics and save them to a dataframe

```
dat <- data.frame(Variable = rep(NA,4),
     Mean = rep(NA,4), SD = rep(NA,4),
     Minimum = rep(NA,4), Maximum = rep(NA,4))

for(i in 1:4) {
     dat$Variable[i] <- names(iris[i])
     dat$Mean[i]     <- mean(iris[, i])
     dat$SD[i]       <- sd(iris[, i])
     dat$Minimum[i]  <- min(iris[, i])
     dat$Maximum[i]  <- max(iris[, i])
}
```

Graphics: `for` Loop

In this example, we will use a `for` loop to create multiple graphics

```
for(i in 1:4) {
      dev.new()
      par(pch = 19)
      plot(iris[, i], col = "slategrey",
      main = names(iris[i]),
      ylab = names(iris[i]), xlab = "")
}
```

Lattice: `for` Loop

When using `for` loops with lattice graphics you need to assign the graphic and print it

```
library(lattice)

for(i in names(iris[1:4])) {
    dev.new()

    p <- histogram( ~ iris[, i] | Species, data =
         iris, main = names(iris[i]),
         ylab = names(iris[i]), xlab = "")

    print(p)
}
```

You Try...

1. Create a vector of 100 random numbers
   - `x <- rnorm(100)`
2. Plot the first element of `x` and set the `ylim` to range from -3 to 3
   - `plot(x[1], ylim = c(-3, 3))`
3. Write a `for` loop that sequentially adds the other 99 elements of `x` to the plot
   - Hint: Use the `points()` function to add points to the plot