Advanced Scientific Computing with R

2. Objects, Arrays, Lists, File I/O

Michael Hahsler

Southern Methodist University

September 3, 2015

These slides are largely based on "An Introduction to R"
`http://CRAN.R-Project.org/`

Table of Contents

Intrinsic attributes: mode

All entities in R are called **objects**. Objects have the intrinsic attributes "mode" and "length".

```
R> x <- c(1.5, 2.6, 3.7)
R> mode(x)
[1] "numeric"
R> y <- as.character(x)
R> y
[1] "1.5" "2.6" "3.7"
R> mode(y)
[1] "character"
```

Modes are types "numeric", "complex", "logical", "character" and "raw".

Intrinsic attributes: length

```
R> x
[1] 1.5 2.6 3.7
R> length(x)
[1] 3
R> e <- numeric()
R> e
numeric(0)
R> length(e)
[1] 0
R> e[5] <- 12
R> e
[1] NA NA NA NA 12
R> length(e) <- 7
R> e
[1] NA NA NA NA 12 NA NA
```

Regular attributes

```
R> z <- 1:4
R> z
[1] 1 2 3 4
R> attributes(z)
NULL
R> class(z)
[1] "integer"
R> attr(z, "dim") <- c(2,2)
R> z
     [,1] [,2]
[1,]    1    3
[2,]    2    4
R> attributes(z)
$dim
[1] 2 2
R> class(z)
[1] "matrix"
```

Regular attributes can be read and set using `attr` and `attributes`.

The dim attribute allows R to treat z as a matrix.

`class()` returns the class of an object. If the object does not have a class attribute (S3) or an implicite class (matrix, array, integer) then it returns the storage type (`storage.mode`).

Table of Contents

Arrays

Arrays are just vectors with a dim attribute.

```
R> x <- 1:8
R> dim(x) <- c(2,2,2)
R> x
, , 1

     [,1] [,2]
[1,]    1    3
[2,]    2    4

, , 2

     [,1] [,2]
[1,]    5    7
[2,]    6    8
R> x <- array(1:8, dim=c(2,2,2))
R> x[2,2,2]
[1] 8
```

Table of Contents

Matrix: 2-dimensional arrays

```
R> x <- matrix(1:6, nrow=2, ncol=3)
R> x
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
R> x[1,]
[1] 1 3 5
R> x[,2]
[1] 3 4
R> dim(x)
[1] 2 3
R> nrow(x)
[1] 2
R> ncol(x)
[1] 3
R> length(x)
[1] 6
```

Matrix: Dimnames

```
R> colnames(x) <- c("X1", "X2", "X3")
R> x
     X1 X2 X3
[1,]  1  3  5
[2,]  2  4  6
R> rownames(x) <- c("Michael", "peter")
R> x
        X1 X2 X3
Michael  1  3  5
peter    2  4  6
R> dimnames(x)
[[1]]
[1] "Michael" "peter"

[[2]]
[1] "X1" "X2" "X3"
```

Matrix: rbind, cbind

```
R> m1 <- matrix(TRUE, nrow=2, ncol=2)
R> m0 <- matrix(FALSE, nrow=2, ncol=2)
R> x <- cbind(m0, m1)
R> x
      [,1]  [,2] [,3] [,4]
[1,] FALSE FALSE TRUE TRUE
[2,] FALSE FALSE TRUE TRUE
R> x <- rbind(x, cbind(m1,m0))
R> x
      [,1]  [,2]  [,3]  [,4]
[1,] FALSE FALSE  TRUE  TRUE
[2,] FALSE FALSE  TRUE  TRUE
[3,]  TRUE  TRUE FALSE FALSE
[4,]  TRUE  TRUE FALSE FALSE
```

Matrix algebra

```
R> a <- 1:3
R> b <- 3:1
R> ab <- outer(a, b, "*")
R> ab
     [,1] [,2] [,3]
[1,]    3    2    1
[2,]    6    4    2
[3,]    9    6    3
R> t(ab)
     [,1] [,2] [,3]
[1,]    3    6    9
[2,]    2    4    6
[3,]    1    2    3
R> ab*ab
     [,1] [,2] [,3]
[1,]    9    4    1
[2,]   36   16    4
[3,]   81   36    9
R> ab %*% ab
     [,1] [,2] [,3]
[1,]   30   20   10
[2,]   60   40   20
[3,]   90   60   30
```

Other important facilities: `crossprod`, `solve` (linear equations), `svd`, `eigen`

Table of Contents

List

An R list is an object consisting of an ordered collection of objects known as its components.

```
R> lst <- list(name="Fred", wife="Mary", no.children=3,
+                    child.ages=c(4,7,9))
R> lst
$name
[1] "Fred"

$wife
[1] "Mary"

$no.children
[1] 3

$child.ages
[1] 4 7 9
R> lst[[2]]
[1] "Mary"
R> lst$wife
[1] "Mary"
```

Lists can contain arbitrary R objects and can be combined with `c()`. Names can be retrieved and changes with `names()`.

Table of Contents

Data Frame

A data frame is a list with class "data.frame" that look like a matrix with mixed data types.

```
R> df <- data.frame(name = c("Michael", "Mark", "Maggie"),
children = c(2,0,2))
R> df
     name children
1 Michael        2
2    Mark        0
3  Maggie        2
R> df$name
[1] Michael Mark    Maggie
Levels: Maggie Mark Michael
R> df[1,]
     name children
1 Michael        2
R> df[,1]
[1] Michael Mark    Maggie
Levels: Maggie Mark Michael
```

Table of Contents

Simple File I/O

`read.table()` and `write.table()` can be used to read/write complete file to/from data.frames. The file format can be space or tab-separated, CSV, with our without column/row labels, etc.

```
R> df
     name children
1 Michael        2
2    Mark        0
3  Maggie        2
R> write.table(df, file="df.dat", sep=",")
R> df2 <- read.table("df.dat", sep=",")
R> df2
     name children
1 Michael        2
2    Mark        0
3  Maggie        2
R> unlink("df.dat")
```

See `?read.table` and `?write.table` for all options.

Accessing data sets

R (an extension packages) come with data sets. These sets can be loaded using `data()`. Without arguments `data()` shows all available data sets.

```
R> data(iris)
R> head(iris)
  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
R> str(iris)
'data.frame':  150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9
...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1
...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4
1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2
0.1 ...
 $ Species     : Factor w/ 3 levels
"setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Table of Contents

Exercises

1. Create m, a 10×10 matrix filled with random numbers between 0 and 1 (use `runif()` to create the random numbers).
2. Multiply m with itself (matrix multiplication).
3. Extract a 3×3 submatrix (first 3 rows and first 3 columns) from m and call it n.
4. Replace the first row of m with all 0s.
5. How many elements in m are larger than .5? Hint: You can use sum for this problem.
6. Create a list called l with m and n as its two elements.
7. Add a vector called count with the values 1, 2 and 3 to l
8. Create a data.frame containing the names, year of birth, month of birth and day of birth as separate columns. Make sure the data.frame has column names (see `colnames()`).
9. Write the data.frame to a file in CSV format.