R Programming #3

JC Wang

Department of Statistics
Western Michigan University

October 22, 2007 / Talk #3
Recursive Objects

Objects consisting of components of various modes/length

- **Lists.** `list(...)`, `is.list(obj)`, `as.list(obj,...)`
- **Data Frames.** `data.frame(...)`
  `is.data.frame(...)`, `as.data.frame(obj)`
- **Expressions.** `expression(...)`
  `is.expression(obj)`, `as.expression(obj,...)`
- **Other objects.** Examples: `lm (linear model)`, `aov` and `anova` (analysis objects), etc.
- `is.recursive(obj)`, `is.atomic(obj)`
Lists

Examples

- `list(1:3, matrix(sample(6),3,2), c("A","B","D"))`
- `list(first=1:3, b=letters[1:2], e=c(F,T))`
- `list(a=1, "B 2"=diag(rep(1,3)))`
- `as.list(1:5)`
- `unlist(mylist)`
- `vector("list", length=5)` creates a length-5 list (unnamed)
Lists
Examples

- \text{list}(1:3, \text{matrix}(\text{sample}(6), 3, 2), \text{c}("A", "B", "D"))
- \text{list}(\text{first}=1:3, b=\text{letters}[1:2], e=\text{c}(\text{F,T}))
- \text{list}(a=1, "B 2"=\text{diag}(\text{rep}(1, 3)))
- \text{as.list}(1:5)
- \text{unlist}(\text{mylist})
- \text{vector}("list", \text{length}=5) \text{ creates a length-5 list (unnamed)}

objects of basic mode (logical, integer, numeric, complex, and raw), list, and expression are ‘vectors’
Lists
Examples

- `list(1:3, matrix(sample(6),3,2), c("A","B","D"))`
- `list(first=1:3, b=letters[1:2], e=c(F,T))`
- `list(a=1, "B 2"=diag(rep(1,3)))`
- `as.list(1:5)`
- `unlist(mylist)`
- `vector("list", length=5)` creates a length-5 list (unnamed)

Objects of basic mode (logical, integer, numeric, complex, and raw), list, and expression are ‘vectors’

Note that matrix and array are not vector.
Subscripting Lists

Assume that `mylist` is a (length-4) list with named components "First", "second", "term 3", and "seven".

- Extracting components

  1. `mylist$F, mylist$"F", mylist$First, mylist$"First", mylist[[1]], mylist[["First"]], mylist[["F"]]] (this gives warning message) all extract component 1.

  2. `mylist$sec and mylist$sev` subscript components 2 and 4, respectively.

  3. Assuming component 2 is a numeric vector, `mylist[["second"]][1:3], (mylist[["second"]])[1:3]` both extract component 2 and then subscript elements 1 to 3.
Subscripting Lists

Assume that `mylist` is a (length-4) list with named components "First", "second", "term 3", and "seven".

- Extracting components
  1. `mylist$F`, `mylist"F"`, `mylist$First`, `mylist"First"`, `mylist[[1]]`, `mylist["First"]`, `mylist["F"]` (this gives warning message) all extract component 1.
  2. `mylist$sec` and `mylist$sev` subscript components 2 and 4, respectively.
  3. Assuming component 2 is a numeric vector, `mylist[["second"]][1:3]`, `(mylist[["second"]])[1:3]` both extract component 2 and then subscript elements 1 to 3.

- Creating sub-list
  1. `mylist[-3]`, `mylist[c("First","second","seven")]` both create a sub-list of components 1, 2, & 4.
Subscripting Lists

Assume that `mylist` is a (length-4) list with named components "First", "second", "term 3", and "seven".

- **Extracting components**
  1. `mylist$F, mylist$"F", mylist$First, mylist$"First", mylist[[1]], mylist[["First"]], mylist[["F"]]` (this gives warning message) all extract component 1.
  2. `mylist$sec` and `mylist$sev` subscript components 2 and 4, respectively.
  3. Assuming component 2 is a numeric vector,
     `mylist[["second"]][1:3], (mylist[["second"]])[1:3]` both extract component 2 and then subscript elements 1 to 3.

- **Creating sub-list**
  1. `mylist[-3], mylist[c("First", "second", "seven")]` both create a sub-list of components 1, 2, & 4.

Note: `mylist$five <- c("A", "two")` adds new component.
Data Frames

- `data.frame(a=1:10, b=logical(10), c=rep(LETTERS[1:2], each=5))`

- OzDASL <- "http://www.statsci.org/data"
  s <- paste(OzDASL, "general/carinsca.txt", sep="/")
  read.table(s, header=T) -> carins
Data Frames

- `data.frame(a=1:10, b=logical(10), c=rep(LETTERS[1:2], each=5))`
- `OzDASL <- "http://www.statsci.org/data"
s <- paste(OzDASL, "general/carinsca.txt", sep="/ ")
read.table(s, header=T) -> carins`

- Extracting data frame (assume `a` is a 5-column data frame with first column named "X1")
  1. `a[,1]`, `a[,"X1"], a[["X1"]]
     give a vector.
  2. `a[1], a["X1"], a[,1,drop=F]
     give a (1-column) data frame.
  3. `a[,-1], a[,2:5]
     give a (4-column) data frame.
Data Frames

- \texttt{data.frame(a=1:10, b=logical(10), c=rep(LETTERS[1:2], each=5))}

- \texttt{OzDASL<-"http://www.statsci.org/data"}
  \texttt{s<- paste(OzDASL, "general/carinsca.txt", sep="/")}
  \texttt{read.table(s, header=T) -> carins}

- **Extracting data frame (assume \texttt{a} is a 5-column data frame with first column named "X1")**
  1. \texttt{a[,1], a[,"X1"], a[["X1"]]} give a vector.
  2. \texttt{a[1], a["X1"], a[,1,drop=F]} give a (1-column) data frame.
  3. \texttt{a[,-1], a[,2:5]} give a (4-column) data frame.

Note that data frame \texttt{a} is of length 5!