Sane's STATS Academy of Statistics

Apply Family Functions

Sane's STATS Academy of Statistics

Functions in apply family

- In this section, we are going to cover the following functions:
  - apply
  - lapply
  - sapply
  - mapply
  - tapply

Function apply

- Function apply() applies a function on the margins of an array.

Syntax : apply(array, margin, func , ...)

Where

array : One or multidimensional array

margin : a vector giving the subscripts which the function will be applied over.

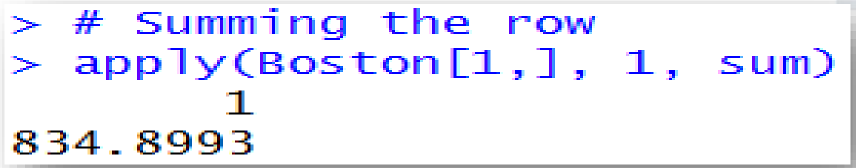
func : Name of the function to be applied

… : Optional arguments for function specified

Return data type of apply() is matrix

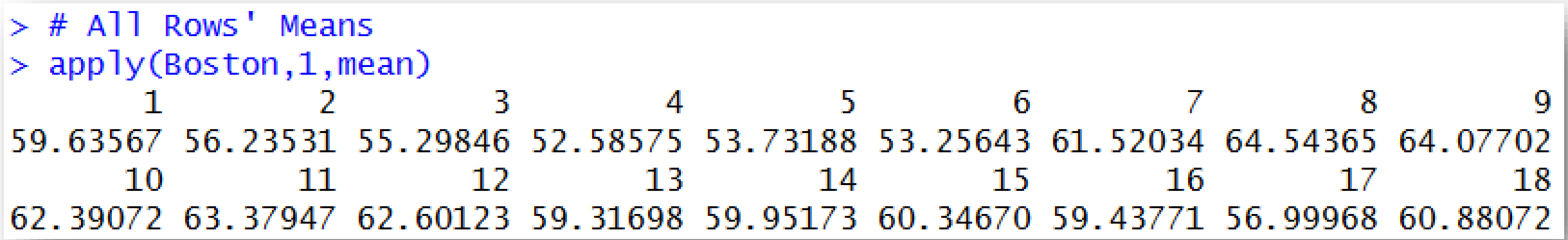
Examples

- Considering a dataset *Boston* from package **MASS** with 506 rows and 14 columns



```
> # Summing the row
> apply(Boston[1,], 1, sum)
       1
834.8993
```

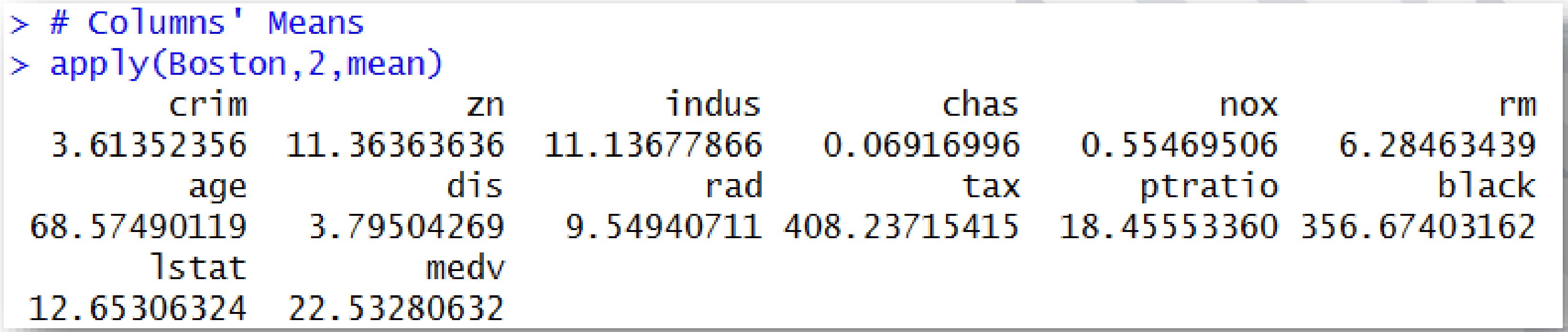
Here dimension being one, we can specify only 1 in the second argument



```
> # All Rows' Means
> apply(Boston,1,mean)
       1        2        3        4        5        6        7        8        9
59.63567 56.23531 55.29846 52.58575 53.73188 53.25643 61.52034 64.54365 64.07702
      10       11       12       13       14       15       16       17       18
62.39072 63.37947 62.60123 59.31698 59.95173 60.34670 59.43771 56.99968 60.88072
```

Actually 506 values generated. Only first 18 values shown due to space shortage. Here we see that the margin=1 signifies row means

Examples



```
> # Columns' Means
> apply(Boston,2,mean)
        crim           zn        indus         chas          nox           rm 
  3.61352356  11.36363636  11.13677866   0.06916996   0.55469506   6.28463439 
         age          dis          rad          tax      ptratio        black 
 68.57490119   3.79504269   9.54940711 408.23715415  18.45553360 356.67403162 
       lstat         medv 
 12.65306324  22.53280632 
```

Here we see that the margin=2 signifies column means

Function lapply

- Function lapply() loops over the list and evaluates the specified function for each element in the list

Syntax : lapply(list, function)

Where

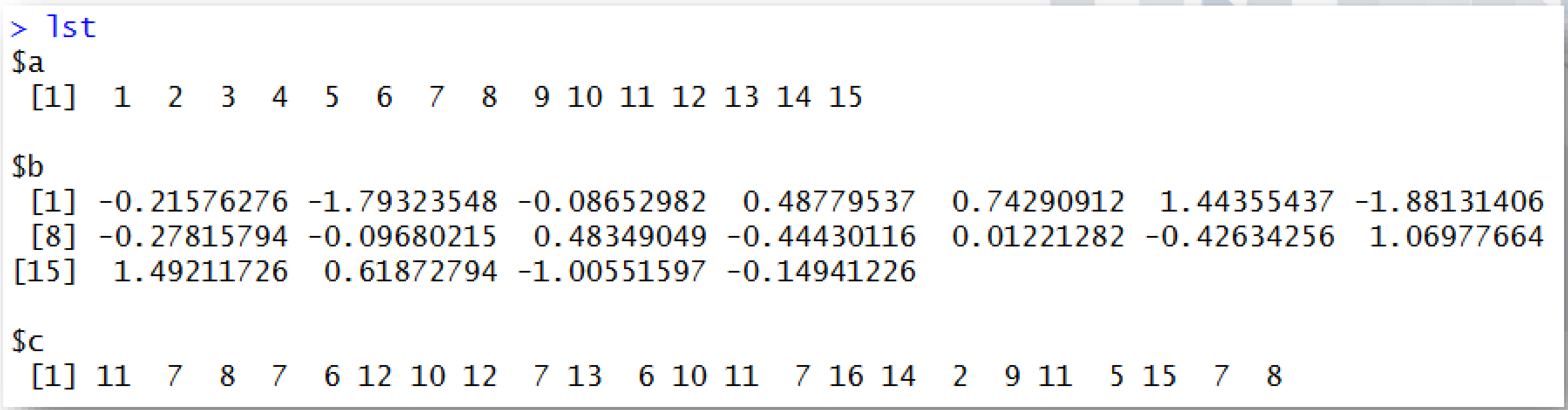
list : A list object. If it is not a list then it be coerced to a list using as.list

function : Function to be applied on each element in the list

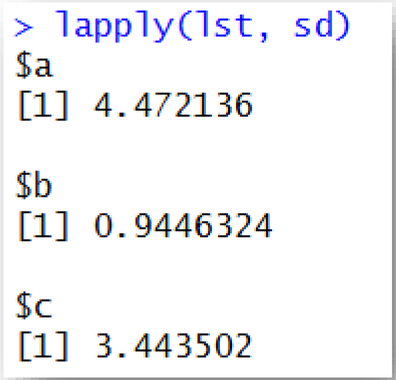
Return type of function lapply() is list

Examples

- Consider an object of list as follows:



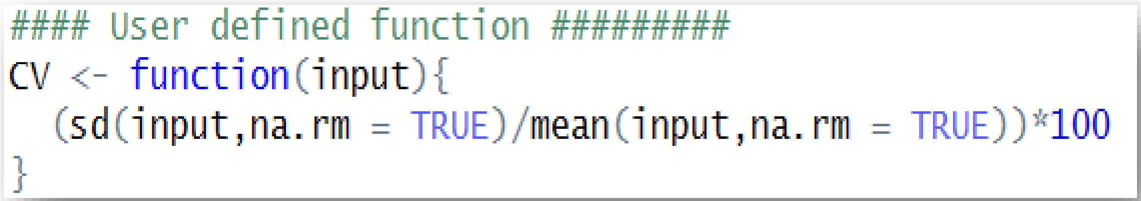
```
> lst
$a
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15

$b
 [1] -0.21576276 -1.79323548 -0.08652982  0.48779537  0.74290912  1.44355437 -1.88131406
 [8] -0.27815794 -0.09680215  0.48349049 -0.44430116  0.01221282 -0.42634256  1.06977664
[15]  1.49211726  0.61872794 -1.00551597 -0.14941226

$c
 [1] 11  7  8  7  6 12 10 12  7 13  6 10 11  7 16 14  2  9 11  5 15  7  8
```



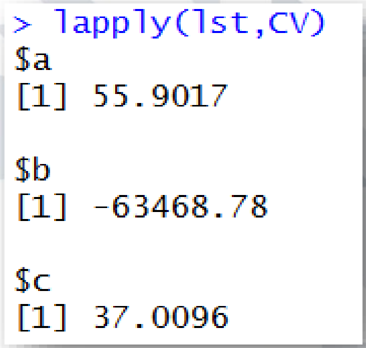
```
> lapply(lst, sd)
$a
[1] 4.472136

$b
[1] 0.9446324

$c
[1] 3.443502
```

Notice that here the aggregate function **sd** has been applied on each of the elements *a*, *b* and *c.*

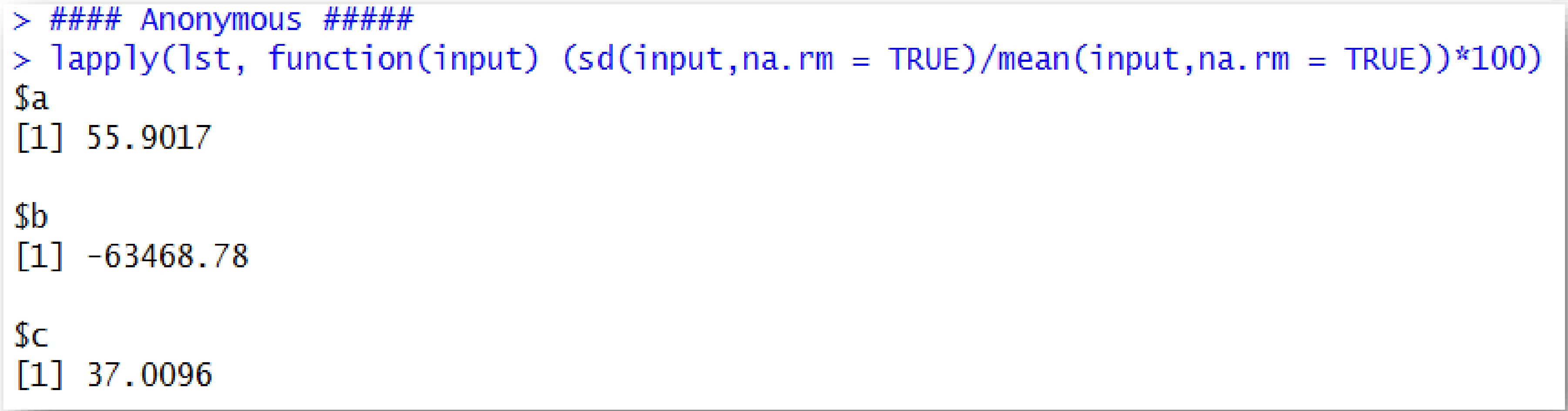
Examples



```r
#### User defined function #########
CV <- function(input){
  (sd(input,na.rm = TRUE)/mean(input,na.rm = TRUE))*100
}
```



```
> lapply(lst,CV)
$a
[1] 55.9017

$b
[1] -63468.78

$c
[1] 37.0096
```



```
> #### Anonymous #####
> lapply(lst, function(input) (sd(input,na.rm = TRUE)/mean(input,na.rm = TRUE))*100)
$a
[1] 55.9017

$b
[1] -63468.78

$c
[1] 37.0096
```

Function sapply

- Function sapply() tries to simplify the result of the function lapply()

Syntax : sapply(list, function)

Where

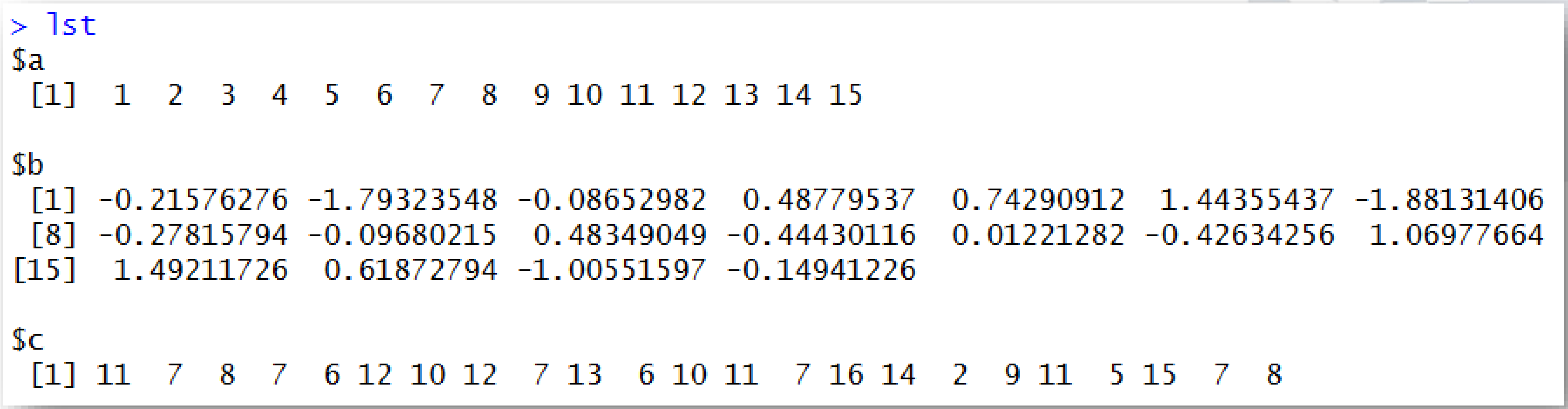
list : A list object. If it is not a list then it be coerced to a list using as.list

function : Function to be applied on each element in the list

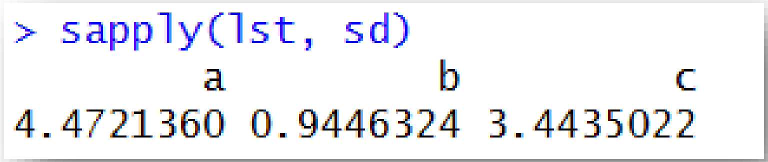
- If result is list where every element is of length=1,then it returns vector.
- If result is list where every element is a vector of same length(>1), then it returns matrix.
- In case if nothing of result can be figured out, then it returns list.

Examples

- Consider an object of list as follows:

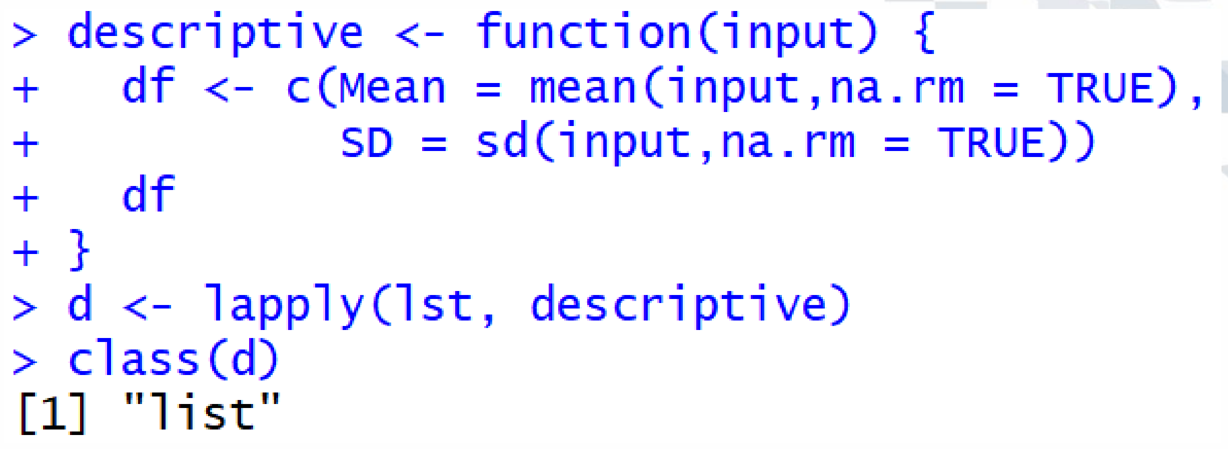


```
> lst
$a
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15

$b
 [1] -0.21576276 -1.79323548 -0.08652982  0.48779537  0.74290912  1.44355437 -1.88131406
 [8] -0.27815794 -0.09680215  0.48349049 -0.44430116  0.01221282 -0.42634256  1.06977664
[15]  1.49211726  0.61872794 -1.00551597 -0.14941226

$c
 [1] 11  7  8  7  6 12 10 12  7 13  6 10 11  7 16 14  2  9 11  5 15  7  8
```

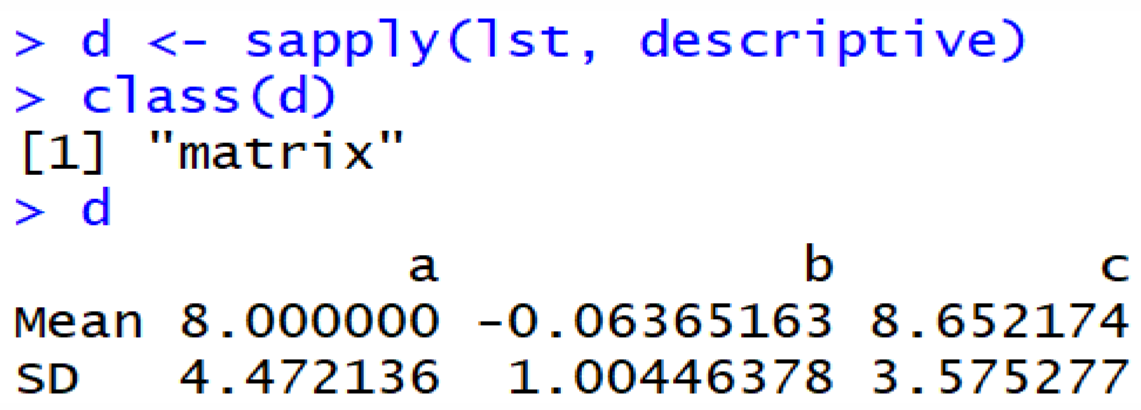


```
> sapply(lst, sd)
        a         b         c 
4.4721360 0.9446324 3.4435022 
```

Examples



```
> descriptive <- function(input) {
+   df <- c(Mean = mean(input,na.rm = TRUE),
+           SD = sd(input,na.rm = TRUE))
+   df
+ }
> d <- lapply(lst, descriptive)
> class(d)
[1] "list"
```



```
> d <- sapply(lst, descriptive)
> class(d)
[1] "matrix"
> d
            a           b        c
Mean 8.000000 -0.06365163 8.652174
SD   4.472136  1.00446378 3.575277
```

Function mapply

- Function mapply() is the multivariate version of lapply()

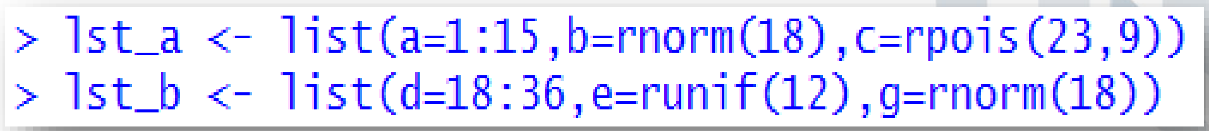
Syntax : mapply(function,…)

Where

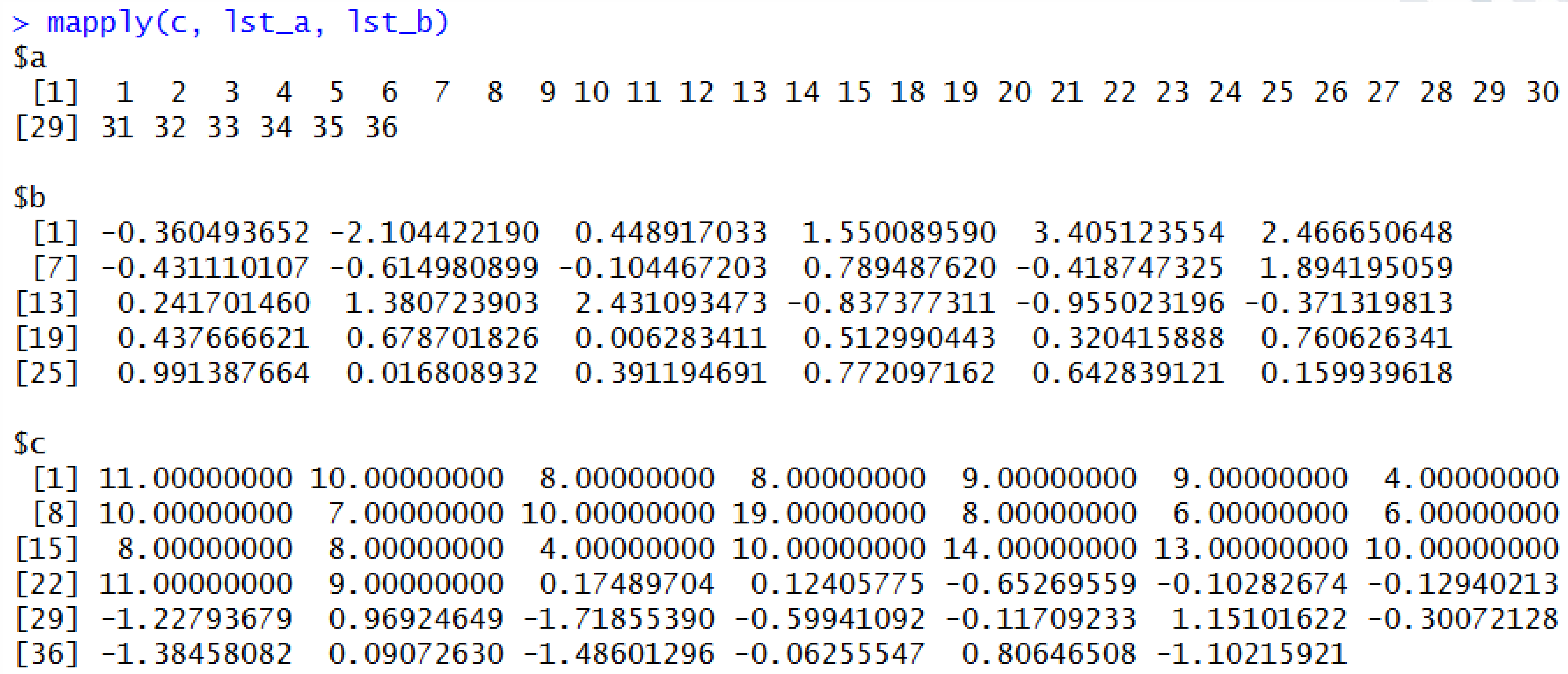
function : function to be specified

… : list elements and other arguments

Examples

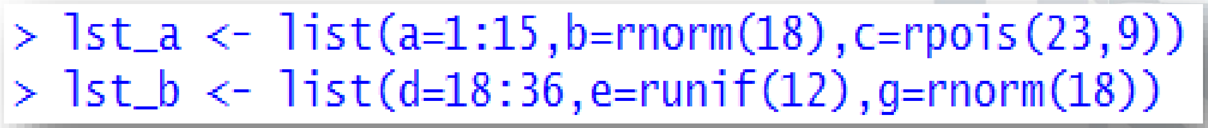


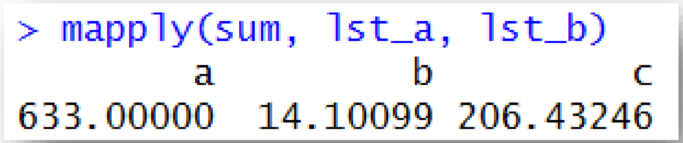
```
> lst_a <- list(a=1:15,b=rnorm(18),c=rpois(23,9))
> lst_b <- list(d=18:36,e=runif(12),g=rnorm(18))
```



```
> mapply(c, lst_a, lst_b)
$a
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 18 19 20 21 22 23 24 25 26 27 28 29 30
[29] 31 32 33 34 35 36

$b
 [1] -0.360493652 -2.104422190  0.448917033  1.550089590  3.405123554  2.466650648
 [7] -0.431110107 -0.614980899 -0.104467203  0.789487620 -0.418747325  1.894195059
[13]  0.241701460  1.380723903  2.431093473 -0.837377311 -0.955023196 -0.371319813
[19]  0.437666621  0.678701826  0.006283411  0.512990443  0.320415888  0.760626341
[25]  0.991387664  0.016808932  0.391194691  0.772097162  0.642839121  0.159939618

$c
 [1] 11.00000000 10.00000000  8.00000000  8.00000000  9.00000000  9.00000000  4.00000000
 [8] 10.00000000  7.00000000 10.00000000 19.00000000  8.00000000  6.00000000  6.00000000
[15]  8.00000000  8.00000000  4.00000000 10.00000000 14.00000000 13.00000000 10.00000000
[22] 11.00000000  9.00000000  0.17489704  0.12405775 -0.65269559 -0.10282674 -0.12940213
[29] -1.22793679  0.96924649 -1.71855390 -0.59941092 -0.11709233  1.15101622 -0.30072128
[36] -1.38458082  0.09072630 -1.48601296 -0.06255547  0.80646508 -1.10215921
```

Examples


```
> lst_a <- list(a=1:15,b=rnorm(18),c=rpois(23,9))
> lst_b <- list(d=18:36,e=runif(12),g=rnorm(18))
```


```
> mapply(sum, lst_a, lst_b)
        a         b         c 
633.00000  14.10099 206.43246 
```

Observe here that the number of elements in both lists are same. If they aren't same, error is fired.

Function tapply()

- Function tapply() is used to apply function on the subsets of vector

Syntax : tapply(vector, index, function)

Where

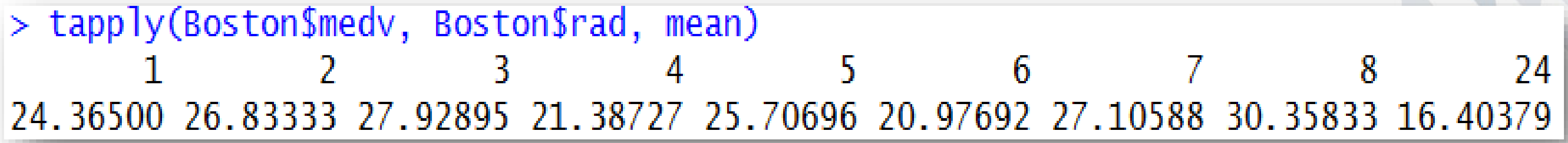
vector : vector for which function to be applied

index : grouping variable

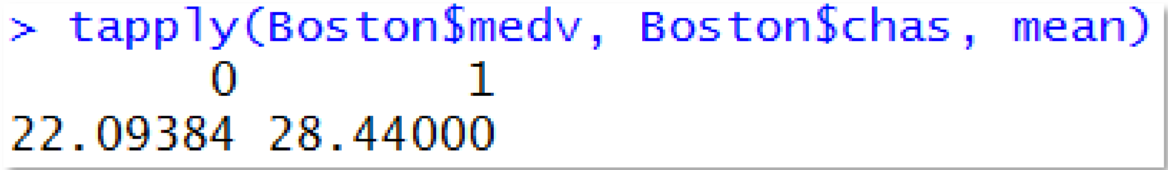
function : function to be specified

Examples

- Consider the variables medv and rad in the dataset *Boston* in package **MASS**



```
> tapply(Boston$medv, Boston$rad, mean)
       1        2        3        4        5        6        7        8       24 
24.36500 26.83333 27.92895 21.38727 25.70696 20.97692 27.10588 30.35833 16.40379 
```



```
> tapply(Boston$medv, Boston$chas, mean)
       0        1 
22.09384 28.44000 
```