

Using Data Frames and Indexing in R

In the following handout words and symbols in **bold** are R functions and words and symbols in *italics> are entries supplied by the user; underlined words and symbols are optional entries (all current as of version R-2.4.1). Sample texts from an R session are highlighted with gray shading.*

Suppose you conduct an experiment in which you perform five replicate analyses on three separate samples and enter the data into R as the objects 'S1,' 'S2' and 'S3.'

```
> S1 = c(1,2,3,4,5)
> S2 = c(6,7,8,9,10)
> S3 = c(11,12,13,14,15)
```

To find the mean for each sample you must work with them one-by-one; thus

```
> mean(S1)
[1] 3

> mean(S2)
[1] 8

> mean(S3)
[1] 13
```

This can become tedious, particularly if you have dozens of samples. Fortunately, R provides two approaches for simplifying the analysis of multiple objects.

Data Frames

A data frame combines two or more vectors of equal length into a single object in which the columns are the individual vectors. The syntax for creating a data frame is

DF = **data.frame**(*object 1*, *object 2*, *object 3...*)

For example

```
> Samples = data.frame(S1, S2, S3)
> Samples
```

	S1	S2	S3
1	1	6	11
2	2	7	12

3	3	8	13
4	4	9	14
5	5	10	15

To apply a function to each sample in the data frame we use the function **sapply**(*object, function*). For example, to find the mean for all three samples now requires just one command

```
> sapply(Samples, mean)
      S1      S2      S3
      3      8     13
```

Indexing

One limitation to a data frame is that each vector must be of equal length. Suppose, however, that the third object, S3, has but four elements (11, 12, 14 and 15). In this case we first combine the three objects into a single object

```
> Samples = c(S1, S2, S3)
> Samples
[1] 1 2 3 4 5 6 7 8 9 10 11 12 14 15
```

and then create an indexing vector whose elements indicate the source of the elements in the vector

```
> SampIndex = c(rep("Sample1", 5), rep("Sample2", 5), rep("Sample3", 4))
```

Note that this bit of code makes use of the function **rep**(*x, times*), to repeat the term *x*, which may be numeric or text, the specified number of time. To apply a function to each object included in Samples we use the function **tapply**(*object, index, function*). Thus, to find the mean for all three samples now requires only one command

```
> tapply(Samples, SampIndex, mean)
Sample1 Sample2 Sample3
      3      8     13
```