

R programming for beginners

Ni Shuai

Computational Genome Biology
German Cancer Research Center (DKFZ)

November, 2016



Vectors

- A sequence of elements of the same type*, the most data structure in R
- The function `c()` combines elements of the same type into a sequence of them, or longer vectors
- A single element can take the form of one of the following:

| | |
|-----------|--|
| Integer | <code>c(5, 4, 3, 2, 1)</code> |
| Numeric | <code>c(5.1, 4.2, 3.3, 2.4, 1.5)</code> |
| Character | <code>c('John', 'Wang', 'Michael', 'Mary', 'Linda')</code> |
| Logical | <code>c(TRUE, FALSE, FALSE, FALSE, TRUE)</code> |



Vectors

- All the elements of the vector must be of the same data type, if one combines vectors of non-consistent types, the more flexible type will be chosen, e.g.:

```
X = c("John", "Wang", "Linda")
Y = c(15, 22, 30.8)
Z = c(X, Y)
Z
## [1] "John" "Wang" "Linda" "15" "22" "30.8"
Z = c(Z, TRUE)
Z
## [1] "John" "Wang" "Linda" "15" "22" "30.8" "TRUE"
```

- Character is the most flexible type of data, therefore all types can be represented as characters



Creating vectors

- Use `seq()` and `rep()` to generate useful sequences efficiently

```
10:1                                # 1,2,3,4,...
seq(from=1, to=10)                  # 1,2,3,4,...
seq(from=1, to=100, by=5)           # 1,6,11,16,...
rep(5, times=10)                    # 5,5,5,5,...
rep(1:2, 3)                         # 1,2,1,2,1,2
rep(1:5, each=3)                    # 1,2,3,4,5,1,2,3,...
rep(3:5, 1:3)                       # 3,4,4,5,5,5
rep(TRUE, 5)                        # TRUE, TRUE, TRUE, TRUE, TRUE
```

- Exercises:
 - (4,6,3, 4,6,3,...,4,6,3) where there are 10 occurrences of 4.
 - (4,4,...,4, 6,6,...,6, 3,3,...,3) where there are 10 occurrences of 4, 20 occurrences of 6 and 30 occurrences of 3.
 - Create a vector of the values of $\cos(x)$ at $x = 3, 3.1, 3.2, \dots, 6$.
 - Calculate $\sum_{n=1}^{20} 2^{-n}$



Element-wise calculation on vectors

Calculations on vectors are performed element by element, as in:

```
2:4 * 1:3           # 2, 6, 12
c(2.3, 4.5, 0.2)*3   # 6.9, 13.5, 0.6
seq(from=1, to=100, by=5)-1 # 0, 5, 10, 15, 20, 25, ...
1:100-c(1,2)         # 0, 0, 2, 2, 4, 4, 6, 6, 8, ...
```

If the length of the two vectors are unequal, the shorter vector is recycled to match the length of the longer vector.

If the length of the longer object is not a multiple of the shorter object, R will worry that you may have made a mistake by raising a warning message:

```
c(1,2,3)==c(4,3,2,1,2)

## Warning in c(1, 2, 3) == c(4, 3, 2, 1, 2): longer object length is
## not a multiple of shorter object length
```



Manipulate vectors: Indexing

Indexing helps to add, delete and change values in a vector, or to select a subset of a sequence of values, which is done by declaring an index inside a single square bracket `"[]"` operator.

For example, we use the index position 3 for retrieving the third member from the vector `s`:

```
s = c("aa", "bb", "cc", "dd", "ee")
s[3]

## [1] "cc"
```

We can even put a vector inside `"[]"` operator to specify multiple elements from a vector.

```
s = c("aa", "bb", "cc", "dd", "ee")
x=s[1:3]; x

## [1] "aa" "bb" "cc"
```



Manipulate vectors: Indexing

If the index is negative, it would remove the member whose position has the same absolute value as the negative index.

For example, to remove the 3rd element from the vector s:

```
s = c("aa", "bb", "cc", "dd", "ee")
```

```
n=s[-3]; n
```

```
## [1] "aa" "bb" "dd" "ee"
```

```
s[c(-1,-3)]
```

```
## [1] "bb" "dd" "ee"
```

If an specified index is even greater than the length of the vector, R will return a missing value **NA**.

```
s = c("aa", "bb", "cc", "dd", "ee")
```

```
s[10]
```

```
## [1] NA
```



Named vectors

We can assign names to vector members:

```
height=c(170, 166, 182, 175, 163)    ### height of 5 people
names(height)= c('John', 'Wang', 'Michael', 'Mary',
                  'Linda') ## and their names
```

```
height
```

```
##      John      Wang Michael      Mary      Linda
##      170      166      182      175      163
```

```
height['John'] # show John's height
```

```
## John
## 170
```

```
height[c('John', 'Mary')] # show both John's and Mary's height
```

```
## John Mary
## 170 175
```



Named vectors

Indexing can also be used in assignments for replacing these element.

```
VC_content=c('apple'=0.2, 'banana'=0.18, 'orange'=0.54, 'kiwi'=0.6,
             'blueberry'=0.41) ### another way of creating a named vector
VC_content['orange']=0.36 ### change of VC content of orange to 0.36
VC_content
```

| ## | apple | banana | orange | kiwi | blueberry |
|----|-------|--------|--------|------|-----------|
| ## | 0.20 | 0.18 | 0.36 | 0.60 | 0.41 |

```
VC_content[3:5]=VC_content[3:5]/2 # reduce the last 3 VC content to half
VC_content
```

| ## | apple | banana | orange | kiwi | blueberry |
|----|-------|--------|--------|-------|-----------|
| ## | 0.200 | 0.180 | 0.180 | 0.300 | 0.205 |



Useful functions

| | |
|--|--|
| <code>length()</code> | length of a vector |
| <code>summary()</code> | print the summary of a vector (min, max, ...) |
| <code>sort()</code> | sort the vector |
| <code>append()</code> | Add elements to a vector |
| <code>rev()</code> | List the vector in reverse order |
| <code>order()</code> , <code>rank()</code> | list sorted element numbers of x |
| <code>identical()</code> | Test if 2 objects are *exactly* equal |
| <code>unique()</code> | Remove duplicate elements from a vector |
| <code>duplicated()</code> | Check if there are duplicated elements in a vector |
| <code>which()</code> | Which (by index) elements are TRUE? |



Vectors

Exercises:

First run the below commands:

```
species=iris$Species # we use another inbuilt dataset  
Sepal_length=iris$Sepal.Length  
Petal_length=iris$Petal.Length
```

- Check the summary of the vector 'species'
- check which ones in vector 'species' are 'setosa', and store this index as 'Setosa'
- Check the length of the above three vectors
- Use 'Setosa' as an index to select the corresponding values from the other two vectors, these are actually sepal and petal length of setosa
- Check the summary of setosa sepal length
- Compare the mean sepal length of 'setosa' and 'virginica'
- Well done!

