Basics of Vectors

The elements are

- **Ordered**
- **Homogeneous type**

Vectors can be created within R using:

- **c()** function to catenate individual values together
- **:** the infix function to create a sequence of numbers \(1:10\)
- **seq()** to create more complex sequences
- **rep()** to create replicates of values
- **sort()** and **order()** are useful for ordering elements in a vector,
  - `sort(x, decreasing = TRUE)`

Examples of `c()`

- **c(3, 2)** – a vector of three numeric elements \(3, 2, 1\) in that order.
- **c(2, 3, 1)** – a different vector of the same three numeric elements, but with a different ordering.
- \(x = c(bob = 3, alice = 2, John = 1)\) – elements can have names.
  - `names(x)`
- Vectors can also consist of characters, logics, factors, integers provided they are all of the same type.

Examples of `rep()`

- **rep(3, 2)** – a vector of two threes
- The arguments of `rep()` can be vectors
  - \(x = c(7, 1, 3)\)
    - `rep(x, 2)`
      - `[1] 7 1 3 7 1 3`
    - `rep(x, c(3, 2, 1))`
      - `[1] 7 7 7 1 1 3`
    - `rep(x, c(2, 1))`
      - Error in rep.default(x, c(2, 1)):
        invalid number of copies in "rep"
Examples of seq()

There are several ways to call the seq function. Here are three popular ones:

- `seq(from, to)`
- `seq(from, to, by =)`
- `seq(from, to, length =)`

Consider arguments of `from = 1`, `to = 19`, `by = 2`, and `length = 10`. Evaluate the following function calls to `seq()` with the various combinations and ordering of arguments (named and unnamed).

- `seq(1, 19, by = 2)`
- `seq(1, 19, length = 10)`
- `seq(1, 19, 2)`
- `seq(1, 19, 10)`
- `seq(1, length = 10, by = 2)`
- `seq(1, 19, length = 10, by = 2)`
- `seq(1, length = 10, 2)`
- `seq(1, length = 10, 19)`

---

Operators

- **Vectorized** – Most functions work on vectors in a vectorized fashion, i.e. they work on all the elements without the need for an explicit loop over the elements.
- **Element-wise** – Most operators work element-wise, i.e. they operate on each element.
  - `x = c(1, 2, 3)`
  - `2 + x`
  - `x > 1`
- **Recycling** – When two vectors have different lengths, the elements of the shorter vector may be recycled.
  - Typically a Warning is issued when this happens.
  - For some functions, an error results.

- `x + c(1, 2)`
  - `2.2 3.0 4.0`  
  - Warning message: longer object length is not a multiple of shorter object length in:
  - `x + c(1, 2)`

---

Results from calls to seq()

- `> seq(1, 19, by = 2)`
  - `[1] 1 3 5 7 9 11 13 15 17 19`
- `> seq(1, 19, length = 10)`
  - `[1] 1 3 5 7 9 11 13 15 17 19`
- `> seq(1, 19, 2)`
  - `[1] 1 3 5 7 9 11 13 15 17 19`
- `> seq(1,19,10)`
  - `[1] 1 3 5 7 9 11 13 15 17 19`
- `> seq(1, length = 10, by = 2)`
  - `[1] 1 3 5 7 9 11 13 15 17 19`
- `> seq(1,19,length=10,by=2)`
  - Error in seq.default(1, 19, length = 10, by = 2) : Too many arguments
- `> seq(1,1,length=10,by=2)`
  - `[1] 1 3 5 7 9 11 13 15 17 19`
### Subsetting

There are five basic ways to refer to a subset.

\[ x = c(11, 30, 2) \]

1. **Position** – \( x[2] \) gives the second element of \( x \), namely 30.
2. **Exclusion** – \( x[-2] \) excludes the second element and returns a vector with 11 and 2
3. **Name** – \( x[ \text{"bob"}] \) returns the element named bob, remember we can name elements.
4. **Logical** - \( x[ \text{c(TRUE, FALSE, TRUE)}] \) subsets the first and third elements of \( x \), 11 and 2.
5. **All** – \( x[] \) returns all of \( x \)
   - This can be helpful when we wish to reset all the values in a vector, \( x[] = 0 \)
   - How do you think this differs from the command \( x = 0 \)?

We provide more examples of each of these.

### Subsetting by Position

- One of the most important things we do in statistics is to divide our data into subgroups for comparison,
  - Lane 1 versus lane 2 on the freeway
  - Traffic at 5 in the morning vs 5 in the afternoon
  - or on different days of the week
- Vectors are ordered collections so we can extract subsets of elements by index or position.
- The [ ] is the subset operator for vectors (and matrices and lists).

### Subsetting by Exclusion

\[ x = c(11, 30, 2, 14) \]

- \( x[-3] \)
  - How long is the output vector?
- \( x[-(2:3)] \)
  - How does this differ from \( x[-2:3] \)?
- \( x[\text{c(4, 2)}] \)
  - Would we get the same result if we switched the order of 2 and 4?
- \( x[\text{c(-4, 1)}] \)
  - Can we exclude the fourth element and include the first? What about the second and third elements of the vector?

### Subsetting by Position

\[ x = c(11, 30, 2, 14) \]

- \( x[3] \)
- \( x[2 : 4] \)
  - How many elements are returned?
- \( x[\text{c(4, 2)}] \)
  - What is the order of the values returned?
- \( x[10] \)
  - Is this an error?
- \( x[0] \)
  - Is this the same as the previous operation?
- \( x[\text{c(4, 0, 1)}] \)
  - What is the length of the output?
- \( \text{ii} = c(3, 2) \)
  - \( y = x[\text{ii}] \)
  - \( x[\text{ii}] = 17 \)
  - What is the value of \( y \) of \( x \)?
**Subsetting with Logicals**

\( x = c(\text{bob}=11, \text{alice}=30, s=2, x=14) \)

- \( x[c(\text{TRUE, TRUE, FALSE, TRUE})] \)
  What is the length of the output vector?
- \( x[c(\text{TRUE, TRUE, FALSE, TRUE})] \)
  What effect does the exclamation point have on the subsetting?
- \( x[c(\text{TRUE, FALSE})] \)
  Remember the recycling rule...
- \( x[\text{FALSE}] \) Is this the same as \( x[0] \) or \( x[12] \)?
- \( x[x > 2] \)
  This is a compound expression. What does the inner expression evaluate to?

**Subsetting by Name**

\( x = c(\text{bob}=11, \text{alice}=30, s=2, x=14) \)

- \( x[\text{"bob"}] \)
  Is there an object \text{bob} in the workspace?
- \( x[\text{bob}] \)
  Can we negate names?
- \( x[c(\text{"bob"}, \"x"')] \)
  The \text{x} plays two roles here. What are they?

**Matrices**

- A matrix in R is a collection of homogeneous elements arranged in 2 dimensions
- A matrix is a vector with a \text{dim} attribute, i.e. an integer vector giving the number of rows and columns
- To create matrices use \text{matrix}()
- The functions \text{dim()}, \text{nrow()} and \text{ncol()} provide the attributes of the matrix.
- Rows and columns can have names, \text{dimnames()}, \text{rownames()}, \text{colnames()}
Matrix Subsetting

Matrix Subsetting

Arrays

Arrays are matrices in higher dimensions

Arrays are matrices in higher dimensions

Subsetting carries over to arrays in the same way. What is the output from,

\[ x[c(4, 3), 1:2, 2] \]

Subsetting carries over to arrays in the same way. What is the output from,

\[ x[c(4, 3), 1:2, 2] \]

We can subset the rows and columns of \( x \) using the \([\]) operator.

We can subset the rows and columns of \( x \) using the \([\]) operator.

- \( x[1:2, \] \) – gives all columns from the first two rows
  - \( x[3:4] \) – gives all rows from the third and fourth columns
  - \( x[c(2, 3), c(4, 3)] \) – returns a 2 by 2 matrix (notice the order of the rows and columns):

© 2023 FoilTEX