# Arithmetics, Logical Operators & Elementary Functions

# Arithmetical Operators

In R the elementary arithmetic is the usual:

- + addition
- − subtraction
- * multiplication
- / division
- ^ raising to a power

and operators are used as in a common calculator.

Example:

```
> (4+3)*2
> 5^2
```

# Logical operators

To compare objects (numbers, vectors, arrays,…) the following operators are used:

- < less than
- > more than
- <= less than or equal to
- >= more than or equal to
- == equal to
- != inequal to

Example:

4>2

A == B

The output is “True” or “False”.

To compose logical espressions the following symbols are used:

- & “and”
- | “or”
- ! “not”

If c1 and c2 are logical expressions:

- c1 & c2 is their intersection
- c1 | c2 is their union
- !c1 is the negation of c1

# Elementary functions available in R:

- **sqrt()** square root
- **abs()** modulus
- **exp()** exponential function
- **log()** natural logarithm
- **log10()** logarithm base10
- **ceiling()** rounds to the smallest integer greater than the argument
- **floor()** rounds to the greatest integer smaller than the argument
- **trunc()** truncates decimals
- **round(x,n)** round x to the specified n decimal places
- **signif(x,n)** round to the n significant places
- **sin() cos() tan()** trigonometric functions
- **asin(), acos(), atan()** inverse trigonometric functions
- **sinh(), cosh() tanh()** hyperbolic functions
- **asinh(), acosh(), atanh()** inverse hyperbolic functions

# Variables and Assignments

# Assign a value to a variable

It has been said that if a command is an *expression*, R evaluate it and the output is printed on the terminal and "*lost*".

In order to keep memory of the output it is necessary to pass its value to a variable, making an *assignment*.

To assign a value to a variable it is used the simbol "<-" pointing to the variable which receives the value:

**x <- 3*4** (the command 3*4 -> x is equivalent)

**y < 2**

**xy <- x*y**

or the function **assign()**:

**assign("x", 3*4)**

**assign("y", 2)**

**assign("xy", x*y)**

To display the value of a variable just type the name of the variable itself.

In the previous example:

```
> x
[1] 12
> y
[1] 2
> xy
[1] 24
```

To display the list of the names of all the variables in the workspace type:

**ls()**

or

**objects()**