12 Some useful references

12.1 Functions

This is a subset of the functions explained in the R reference card.

Data creation

- `read.table`: read a table from file. Arguments: `header=TRUE`: read first line as titles of the columns; `sep=","`: numbers are separated by commas; `skip=n`: don't read the first `n` lines.
- `write.table`: write a table to file
- `c`: paste numbers together to create a vector
- `array`: create a vector, Arguments: `dim`: length
- `matrix`: create a matrix, Arguments: `ncol` and/or `nrow`: number of rows/columns
- `data.frame`: create a data frame
- `list`: create a list
- `rbind` and `cbind`: combine vectors into a matrix by row or column

Extracting data

- `x[n]`: the n^{th} element of a vector
- `x[m:n]`: the m^{th} to n^{th} element
- `x[c(k,m,n)]`: specific elements
- `x[x>m & x<n]`: elements between `m` and `n`
- `x$n`: element of list or data frame named `n`
- `x[["n"]]`: idem
- `[i,j]`: element at i^{th} row and j^{th} column
- `[i,]`: row `i` in a matrix

Information on variables

- `length`: length of a vector
- `ncol` or `nrow`: number of columns or rows in a matrix
- `class`: class of a variable
- `names`: names of objects in a list
- `print`: show variable or character string on the screen (used in scripts or for-loops)
- `return`: show variable on the screen (used in functions)
- `is.na`: test if variable is `NA`
- `as.numeric` or `as.character`: change class to number or character string
- `strptime`: change class from character to date-time (POSIX)

Statistics

- `sum`: sum of a vector (or matrix)
- `mean`: mean of a vector
- `sd`: standard deviation of a vector
- `max` or `min`: largest or smallest element
- `rowSums` (or `rowMeans`, `colSums` and `colMeans`): sums (or means) of all numbers in each row (or column) of a matrix. The result is a vector.
- `quantile(x,c(0.1,0.5))`: sample the 0.1 and 0.5^{th} quantiles of vector `x`

Data processing

- `seq`: create a vector with equal steps between the numbers
- `rnorm`: create a vector with random numbers with normal distribution (other distributions are also available)
- `sort`: sort elements in increasing order
- `t`: transpose a matrix
- `aggregate(x,by=ls(y),FUN="mean")`: split data set `x` into subsets (defined by `y`) and computes means of the subsets. Result: a new list.
- `na.approx`: interpolate (in `zoo` package). Argument: vector with `NA`s. Result: vector without `NA`s.
- `cumsum`: cumulative sum. Result is a vector.
- `rollmean`: moving average (in the `zoo` package)
- `paste`: paste character strings together
- `substr`: extract part of a character string

Fitting

- `lm(v1~v2)`: linear fit (regression line) between vector `v1` on the y-axis and `v2` on the x-axis
- `nls(v1~a+b*v2, start=ls(a=1,b=0))`: non-linear fit. Should contain equation with variables (here `v1` and `v2` and parameters (here `a` and `b`) with starting values
- `coef`: returns coefficients from a fit
- `summary`: returns all results from a fit

Plotting

- `plot(x)`: plot `x` (y-axis) versus index number (x-axis) in a new window
- `plot(x,y)`: plot `y` (y-axis) versus `x` (x-axis) in a new window
- `image(x,y,z)`: plot `z` (color scale) versus `x` (x-axis) and `y` (y-axis) in a new window
- `lines` or `points`: add lines or points to a previous plot
- `hist`: plot histogram of the numbers in a vector
- `barplot`: bar plot of vector or data frame
- `contour(x,y,z)`: contour plot
- `abline`: draw line (segment). Arguments: `a,b` for intercept `a` and slope `b`; or `h=y` for horizontal line at `y`; or `v=x` for vertical line at `x`.
- `curve`: add function to plot. Needs to have an

x in the expression. Example: `curve(x^2)`
• `legend`: add legend with given symbols (`lty` or `pch` and `col`) and text (`legend`) at location (`x="topright"`)
• `axis`: add axis. Arguments: `side` – 1=bottom, 2=left, 3=top, 4=right
• `mtext`: add text on axis. Arguments: `text` (character string) and `side`
• `grid`: add grid
• `par`: plotting parameters to be specified before the plots. Arguments: e.g. `mfrow=c(1,3))`: number of figures per page (1 row, 3 columns); `new=TRUE`: draw plot over previous plot.

Plotting parameters
These can be added as arguments to `plot`, `lines`, `image`, etc. For help see `par`.
• `type`: `"l"`=lines, `"p"`=points, etc.
• `col`: color – `"blue"`, `"red"`, etc
• `lty`: line type – 1=solid, 2=dashed, etc.
• `pch`: point type – 1=circle, 2=triangle, etc.
• `main`: title - character string
• `xlab` and `ylab`: axis labels – character string
• `xlim` and `ylim`: range of axes – e.g. c(1,10)
• `log`: logarithmic axis – `"x"`, `"y"` or `"xy"`

Programming
• `function(arglist){expr}`: function definition: do `expr` with list of arguments `arglist`
• `if(cond){expr1}else{expr2}`: if-statement: if `cond` is true, then `expr1`, else `expr2`
• `for(var in vec) {expr}`: for-loop: the counter `var` runs through the vector `vec` and does `expr` each run
• `while(cond){expr}`: while-loop: while `cond` is true, do `expr` each run

12.2 Keyboard shortcuts

There are several useful keyboard shortcuts for RStudio (see `Help` → `Keyboard Shortcuts`):
• `CRL+ENTER`: send commands from script window to command window
• ↑ or ↓ in command window: previous or next command
• `CTRL+1`, `CTRL+2`, etc.: change between the windows

Not R-specific, but very useful keyboard shortcuts:
• `CTRL+C`, `CTRL+X` and `CTRL+V`: copy, cut and paste
• `ALT+TAB`: change to another program window
• ↑, ↓, ← or →: move cursor
• `HOME` or `END`: move cursor to begin or end of line
• `Page Up` or `Page Down`: move cursor one page up or down
• `SHIFT+`↑/↓/←/→`/HOME/END/PgUp/PgDn`: select

12.3 Error messages

• `No such file or directory` or `Cannot change working directory`
Make sure the working directory and file names are correct.
• `Object 'x' not found`
The variable `x` has not been defined yet. Define `x` or write apostrophes if `x` should be a character string.
• `Argument 'x' is missing without default`
You didn't specify the compulsory argument `x`.
• `+`
R is still busy with something or you forgot closing brackets. Wait, type `}` or `)` or press `ESC`.
• `Unexpected ')' in ")"` or `Unexpected '}' in "}"`
The opposite of the previous. You try to close something which hasn't been opened yet. Add opening brackets.
• `Unexpected 'else' in "else"`
Put the `else` of an if-statement on the same line as the last bracket of the "then"-part: `}else{`.
• `Missing value where TRUE/FALSE needed`
Something goes wrong in the condition-part (`if(x==1)`) of an if-statement. Is `x` `NA`?
• `The condition has length > 1 and only the first element will be used`
In the condition-part (`if(x==1)`) of an if-statement, a vector is compared with a scalar. Is `x` a vector? Did you mean `x[i]`?
• `Non-numeric argument to binary operator`
You are trying to do computations with something which is not a number. Use `class(...)` to find out what went wrong or use `as.numeric(...)` to transform the variable to a number.
• `Argument is of length zero` or `Replacement is of length zero`
The variable in question is `NULL`, which means that it is empty, for example created by `c()`. Check the definition of the variable.