# R Cheatsheet

**Notes:**

1. This is by no means a comprehensive list, as a large number of useful functions have been left out, and not all options for the functions listed have been given. This list is purely intended to give a place to begin, as I remember how frustrating it was to not even know what to start looking for!
2. Typing `?functionname` at the command line brings up a help window for the function name listed.
3. Assume in the examples that all vectors and matrices (v’s and mat’s) have been created.

<table>
<thead>
<tr>
<th>Command</th>
<th>Example</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operators</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;-</td>
<td>Assignment operator (suggested)</td>
<td>ans1 &lt;- 1</td>
</tr>
<tr>
<td>=</td>
<td>Assignment operator</td>
<td>ans2 = 1 + 1</td>
</tr>
<tr>
<td>#</td>
<td>Comment</td>
<td>#This is a comment</td>
</tr>
<tr>
<td>Mathematical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+</td>
<td>Addition</td>
<td>2.5 + ans3</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction</td>
<td>ans3 - 2.5</td>
</tr>
<tr>
<td>*</td>
<td>Scalar multiplication</td>
<td>2 * 3</td>
</tr>
<tr>
<td>/</td>
<td>Division operator</td>
<td>6 / 2</td>
</tr>
<tr>
<td>^</td>
<td>Exponentiation</td>
<td>2 ^ 3</td>
</tr>
<tr>
<td>Logical/Relational</td>
<td></td>
<td></td>
</tr>
<tr>
<td>==</td>
<td>Equals</td>
<td>ans3 == 3</td>
</tr>
<tr>
<td>!=</td>
<td>Not Equal</td>
<td>ans3 != 3</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater Than</td>
<td>ans3 &gt; 3</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater Than or Equal To</td>
<td>ans3 &gt;= 3</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less Than</td>
<td>ans3 &lt; 3</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less Than or Equal To</td>
<td>ans3 &lt;= 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Or (use with vectors and matrices)</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>And</td>
<td>ans1 == 2 &amp;&amp; ans2 == 2</td>
</tr>
<tr>
<td>&amp;</td>
<td>And (use with vectors and matrices)</td>
<td>v2[v1==3 &amp; v1==4]</td>
</tr>
<tr>
<td>%*%</td>
<td>Matrix multiplication</td>
<td>mat1 %*% mat1</td>
</tr>
<tr>
<td><strong>Functions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sqrt</td>
<td>Square root</td>
<td>sqrt(16)</td>
</tr>
<tr>
<td>exp</td>
<td>Exponentiation</td>
<td>exp(1)</td>
</tr>
<tr>
<td>log</td>
<td>Natural log</td>
<td>log(16)</td>
</tr>
<tr>
<td>sum</td>
<td>Sum</td>
<td>sum(2,3,4)</td>
</tr>
<tr>
<td>prod</td>
<td>Product</td>
<td>prod(2,3,4)</td>
</tr>
<tr>
<td>ceiling</td>
<td>Smallest integer ≥number</td>
<td>ceiling(2.1)</td>
</tr>
<tr>
<td>floor</td>
<td>Integer part of a number</td>
<td>floor(2.1)</td>
</tr>
<tr>
<td>abs</td>
<td>Absolute value</td>
<td>abs(-0.2)</td>
</tr>
<tr>
<td>sin</td>
<td>Sine</td>
<td>sin(pi/2)</td>
</tr>
<tr>
<td>cos</td>
<td>Cosine</td>
<td>cos(pi)</td>
</tr>
<tr>
<td>tan</td>
<td>Tangent</td>
<td>tan(pi/4)</td>
</tr>
<tr>
<td>table</td>
<td>Calculate frequency counts of a vector</td>
<td>table(v4)</td>
</tr>
<tr>
<td>Vector/Matrix Functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vector creation functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>Concatenate</td>
<td>v1 &lt;- c(2,3,4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>v2 &lt;- c(1,3,5)</td>
</tr>
<tr>
<td>seq</td>
<td>Sequence</td>
<td>v3 &lt;- seq(from=2, to=10, by=2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>seq(from=2, to=4, length=5)</td>
</tr>
<tr>
<td>:</td>
<td>Integer sequence</td>
<td>2:10</td>
</tr>
<tr>
<td>rep</td>
<td>Repeat</td>
<td>v4 &lt;- rep(v2, 3)</td>
</tr>
</tbody>
</table>
Combining vectors to create matrices

- **cbind**: Column bind
  - `mat1 <- cbind(v1, v2)`
  - 
  - \[
  \begin{pmatrix}
  2 \\
  3 \\
  4 \\
  \end{pmatrix}
  \]

- **rbind**: Row bind
  - `mat2 <- rbind(v1, v2)`
  - 
  - \[
  \begin{pmatrix}
  2 & 3 & 4 \\
  1 & 3 & 5 \\
  \end{pmatrix}
  \]

- **matrix**: Create matrix
  - `matrix(0, nrow=2, ncol=3)`
  - 
  - \[
  \begin{pmatrix}
  0 & 0 & 0 \\
  0 & 0 & 0 \\
  \end{pmatrix}
  \]

- **as.data.frame**: Create dataset from matrix
  - `A <- as.data.frame(mat1)`
  - 
  - \[
  \begin{pmatrix}
  2 \\
  3 \\
  4 \\
  \end{pmatrix}
  \]

Utility functions

- **Subscript operator (Vectors)**
  - `[ ]`: `answer <- v1[3]`
  - `4`

- **Subscript operator (2D)**
  - `[ , ]`:
    - `answer <- mat1[1,1]`
    - `2`
    - `answer <- mat1[1,1]`
    - `2,1`
    - `answer <- mat1[1,1]`
    - `2,3,4`
    - `answer <- mat1[1,-1,]`
    - 
  - 
  - \[
  \begin{pmatrix}
  3 \\
  4 \\
  \end{pmatrix}
  \]

- **Subscript operator (3D)**
  - `[,, ]`:
    - `answer <- arr1[2,4,3]`
    - `114`

- **length**: Length of vector
  - `length(v4)`
  - `9`

- **sort**: Sort a vector
  - `sort(v4)`
  - `1,1,3,3,5,5,5`

- **order**: Indices to sort a vector
  - `order(v4)`
  - `1,4,7,2,5,8,3,6,9`

- **rev**: Reverse order of vector
  - `rev(v3)`
  - `10,8,6,4,2`

- **unique**: Lists unique objects in vector or matrix
  - `unique(v4)`
  - `1,3,5`

Statistics

- **max**: Maximum of vector or matrix
  - `max(v4)`
  - `5`

- **min**: Minimum of vector or matrix
  - `min(mat1)`
  - `1`

- **pmax**: Parallel maximum of vectors/matrices
  - `pmax(v1,v2)`
  - `2,3,5`

- **pmin**: Parallel minimum of vectors/matrices
  - `pmin(v1,v2)`
  - `1,3,4`

- **mean**: Calculates mean of vector or matrix
  - `mean(mat1)`
  - `3`

- **median**: Calculates median of vector or matrix
  - `median(v3)`
  - `6`

- **quantile**: Calculate quantiles requested
  - `quantile(1:5, prods=c(0,0.25,0.5,0.75,1))`
  - `1,2,3,4,5`

- **var**: Calculate variance of vector
  - `var(v3)`
  - `10`

- **cor**: Calculates correlation of 2 vectors
  - `cor(v4,1:9)`
  - `0.3162`

Distributions

- **d<dist>(x,<parameters>)**: Density at x
  - `dunif(1.4,min=1,max=3)`
  - `0.5`

- **p<dist>(x,<parameters>)**: CDF evaluated at x
  - `pnorm(1.645,0,1)`
  - `0.95`

- **q<dist>(x,<parameters>)**: Inverse CDF
  - `qnorm(0.95,0,1)`
  - `1.645`

- **r<dist>(x,<parameters>)**: Generates n random numbers
  - `rbeta(3, shape1=0.5, shape2=1)`
  - `0.175083, 0.668609, 0.009384`

<table>
<thead>
<tr>
<th>&lt;dist&gt;</th>
<th>Distribution</th>
<th>Parameters</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta</td>
<td>Beta</td>
<td>shape1, shape2</td>
<td>-,-</td>
</tr>
<tr>
<td>cauchy</td>
<td>Cauchy</td>
<td>location, scale</td>
<td>0,1</td>
</tr>
<tr>
<td>chisq</td>
<td>Chi-square</td>
<td>df</td>
<td>-</td>
</tr>
<tr>
<td>exp</td>
<td>Exponential</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>f</td>
<td>F</td>
<td>df1, df2</td>
<td>-,-</td>
</tr>
<tr>
<td>gamma</td>
<td>Gamma</td>
<td>shape</td>
<td>-</td>
</tr>
<tr>
<td>lnorm</td>
<td>Log-normal</td>
<td>mean, sd (of log)</td>
<td>0,1</td>
</tr>
<tr>
<td>Logis</td>
<td>Logistic</td>
<td>location, scale</td>
<td>0,1</td>
</tr>
<tr>
<td>norm</td>
<td>Normal</td>
<td>mean, sd</td>
<td>0,1</td>
</tr>
<tr>
<td>stab</td>
<td>Stable</td>
<td>index, skew</td>
<td>-0</td>
</tr>
<tr>
<td>t</td>
<td>Student’s t</td>
<td>df</td>
<td>-</td>
</tr>
</tbody>
</table>
For Loops

for(i in <vector>){ do stuff }

ans <- 1
for(i in 1:5){ ans <- ans*i }
ans     120

if/else

if(<logical value>) { do stuff }
else { do other stuff }

if(ans > 100){ ans2 <- 100}
else { ans2 <- ans}
ans2     100

Functions

func.name <- function(arg1, arg2, ...){ do stuff; return(ans)}

my.factorial <- function(x){
  if(!is.integer(x))
    stop("x must be an integer")
  ans <- 1
  for(i in 1:x){ ans <- ans*i }
  return(ans)
}
my.factorial(5)    120

Useful links:
http://cran.r-project.org/doc/contrib/usingR-2.pdf
http://www.isds.duke.edu/computing/S/Snotes/Splus.html
http://lib.stat.cmu.edu/S/cheatsheet