R Cheat Sheet

Basic Syntax

		Example
#	Comments	# This is not interpretted
<- or =	Assignment	a <- 1; b = 2
<<-	Global Assignment	a <<- 10 # Not recommended
v[1]	First element in a vector	v[1]
*	Scalar Multiplication	c(1,1)*c(1,1) # 1 1
%*%	Matrix Multiplication	c(1,1)%*%c(1,1) # 2
/	Division	1/2 # 0.5
%/%	Integer Division	1%/%2 # 0
%%	Remainder	7%%6 # 1

Vector and Matrix Operations

Construction

c()	Concatenate	v <- c(1,2,3,4) # 1 2 3 4
cbind()	Column Concatenate	cbind(v,v) # 2 Columns
rbind()	Row Concatenate	rbind(v,v) # 2 Rows
matrix()	Create matrix	mat <- matrix(v,nrow=2,ncol=2)

Selection

v[1]	Select first
tail(v,1)	Select last
mat[2,1]	Select row 2, column 1
mat[1,]	Select row 1
mat[,2]	Select column 2
v[c(1,3)]	Select the first and third values
v[-c(1,3)]	Select all but the first and third values
mat[,c(1,2)]	Select columns 1 and 2
mat[,1:5]	Select columns 1 to 5
mat[“col”]	Select column named “col”

Utility

length()	Length of vector
dim()	Dimensions of vector/matrix/dataframe
sort()	Sorts vector
order()	Index to sort vector e.g. sort(v) == v[order(v)]
names()	Names of columns

Apply

`apply(data, axis, fun)`	Apply the function fun to data along axis
`lapply(data, fun)`	Apply the function fun to the list or vector data
`rapply(data, fun, how)`	Apply the function fun to data depending on the value of how

For a great introduction to using apply see **this article**.

I/O

`read.table("filename", sep=",")`	Reads information in from file with a variable delimiter
`read.csv()`	Read csv file into dataframe
`fromJSON()`	Read JSON formatted file or string into a list
`xmlTreeParse()`	Read XML formatted file or string into a list
`write.csv()`	Writes a dataframe or matrix (or tries to convert input to one) and writes to csv

Structures

`data.frame(…)`	Takes multiple variables with the same number of rows and creates a dataframe
`list(name=var)`	R's implementation of a hash, takes multiple variables or variable tag pairs
`ts(data, frequency)`	Creates a regularly spaced time series object

Time Series

Time Series Classes

`ts(data, frequency)`	From R base, handles regularly spaced time series
`zoo(data, index)`	Part of the zoo package, handles irregularly spaced data using arbitrary time date classes
`xts(data, index)`	Modification to zoo, gives more power to selecting date subsets
`tis()`	Uses POSIXct time stamps
`irts()`	From the package tseries and uses POSIXct time stamps
`timeSeries()`	Uses timeDate time stamps

Tests

`adf.test`	Computes the Augmented Dickey-Fuller test for the null that the series has a unit root
`jarque.bera.test`	Tests the null of normaility for the series using the Jarque-Bera test statistics
`kpss.test`	Computes the KPSS test for the null that the series is level or trend stationary
`po.test`	Computes the Phillips-Ouliaris test for the null that the series in not cointegrated
`pp.test`	Computes the Phillips-Perron test for the null that the series has a unit root
`runs.test`	Computes the runs test for randomness of the binary series
`terasvirta.test`	Generically computes Teraesvirta's neural network test for neglected nonlinearity for the time series
`white.test`	Generically computes the White neural network test for neglected nonlinearity for the time series
`box.test`	Compute the Box-Pierce or Ljung-Box test statistics for examining the null of independence in a given series

`shapiro.test`	Test for normality
`ks.test`	Test for specified distribution
`punitroot`	Computes the cumulative probability of MacKinnon's unit root tes statistic

Decomposition

`decompose()`	Decompoes the series into seasonal, trend and irregular components using moving averages
`filter()`	Applies a linear filter to a series
`stl()`	Decomposes the series into seasonal, trend and irregular components using loess methods

Models

`ar()`	Fits an autoregressive model
`ma()`	Fits a simple moving average model
`arima()`	Fits an arima model with specified parameters
`auto.arima()`	Automatically fits an arima model
`ets()`	Fits an exponential smoothing model
`HoltWinters()`	Computes Holt-Winters Filtering of a given time series
`forcast(model, n)`	Forecasts the next n points from the time series model

Utility

`index()`	Returns the index (dates) of the time series
`coredata()`	Returns a matrix of data from the time series sans index
`lag(ts, n)`	Returns the time series shifted n times
`diff(ts, n)`	Returns the time series differences n times
`acf() or Acf()`	Returns the autocorrelation function of the time series (Acf() from the forecast package)
`pacf() or Pacf()`	Returns the partial autocorrelation function of the time series (Pacf() from the forecast package)
`ndiff()`	Returns the number of differences needed for a time series to have stationarity
`accuracy()`	Computes the accuracy of a time series model

Quandl

The Quandl package enables Quandl **API** access from within R which makes acquiring and manipulating numerical data as quick and easy as possible. In your first Quandl function call you should specifiy your authtoken (found on Quandl's website after signing up) to avoid certain API call limits.

See **www.quandl.com/help/packages/R** for more.

Quandl is a search engine for numerical data, alowing easy access to financial, social, and demographic data from hundreds of sources.

`Quandl.auth("AUTHETICATION_TOKEN")`	Call this first to increase your daily limit
`Quandl("QUANDL/CODE")`	Download Quandl data for a certain Quandl code as a data frame
`Quandl.search("query")`	Search Quandl. Prints first three outputs to screen, returns all in a list.

Plotting

`plot(ts)`	R base plot function
`title(main, sub, xlab, ylab)`	Adds labels to the currently open plot

Aside from the built in plotting function in R, ggplot2 is a very powerful plotting package.
See http://docs.ggplot2.org/current/ for complete documentation.

`ggplot()`	Creates a ggplot object
`aes()`	Creates a properly formatted list of variables for use in ggplot
`geom_line()`	Plots data with a line connecting them
`geom_boxplot()`	Plots data in the form of box and whiskers plot
`xlab()`	Edit the x axis label
`ylab()`	Edit the y axis label
`ggtitle()`	Edit the plot title
`theme()`	Modify a large number of options for the plot from grid elements to colors

Plotting example with ggplot2

```
library(Quandl)
library(ggplot2)
data_series <- Quandl("GOOG/NASDAQ_AAPL", start_date="2005-01-01")[,c(1,5)]
my.plot <- ggplot(data=data_series, aes(x=Date, y=Close)) +
geom_line(color="#FAB521") + # Adding a colored line
theme(panel.background = element_rect(fill='#393939'), panel.grid.major.x = element_blank(),
panel.grid.major.y = element_line(colour='white', size=0.1),
panel.grid.minor = element_line(colour='white', size=0.1)) + # modifying background color
# and grid options
xlab("Date") + ylab("Closing Price") + ggtitle("AAPL") # Adding titles

my.plot # Generates the plot
```