

R help functions		Managing R workspace		Managing R workspace (cont)		Packages	
help.start()	General help	getwd()	Get working directory	save(objectlist, file = "myfile")	Save specific objects to a file	.libPaths()	Shows where library is located
help(plot) or ?plot	Help on function plot	setwd("mydirectory")	Set working directory to <i>mydirectory</i>	load("myfile")	Load a workspace in the current session (default = .RData)	library()	Shows the packages saved in the library
help.search("geo")	Search help system for instances of string <i>geo</i>	ls()	List the objects in workspace	q()	Quit R. Will be prompted to save the workspace.	search()	Shows which packages have been loaded and ready to use
example(plot)	Examples of function plot	rm(objectlist)	Remove one or more objects	<pre>setwd("C:/myproject/s/project1") options() options(digits=3) x <- runif(20) summary(x) hist(x) savehistory("myfile") save.image() q()</pre>		install.packages("ggplot")	Install a package for the first time
RSiteSearch("geo")	Search for string <i>geo</i> in online help manuals and archived mailing lists	help(options)	Learn about available options	Saving graphic output		update.packages("ggplot")	Update packages that have been installed
apropos("geo", mode="function")	List all available functions with <i>geo</i> in their name	options()	View or set current options			installed.packages()	Lists packages with version numbers, dependencies, and other info
data()	List all example datasets in loaded packages	history(#)	Display the last # of commands (default=25)	pdf("filename.pdf")	PDF file	library(ggplot)	Load package
vignette()	List all vignettes for installed packages	savehistory("myfile")	Save the commands history to <i>myfile</i> (default = .Rhistory)	win.metafile("filename.wmf")	Windows metafile	help(package="ggplot")	Brief description of package and index of functions and datasets
vignette("geo")	Display specific vignettes for topic <i>geo</i>	loadhistory("myfile")	Reload a commands history (default = .Rhistory)	png("filename.png")	PBG file	<pre>help.start() install.packages("vcd") help(package="vcd") library(vcd) help(Arthritis) example(Arthritis) q()</pre>	
		save.image("myfile")	Save the workspace to <i>myfile</i> (default = .RData)	jpeg("filename.jpg")	JPEG file		
				bmp("filename.bmp")	BMP file		
				postscript("filename.ps")	PostScript file		
				<pre>jpeg("plot.jpg") plot(x, y) dev.off()</pre>			

Data input	Accessing DBMSs - RODBC package (cont)	Data structures (cont)	Example code (cont)
Keyboard input <pre>mydata <- data.frame(age=numeric(0),gender=character(0),weight=numeric(0))</pre>	<pre>sqlQuery(channel, query)</pre>	Arrays <pre>array = character(0), weight = numeric(0)</pre>	<pre>> patientID <- c(1,2,3,4)) age <- c(25,24,28,52) diabetes <- c("Type1","Type2","Type2","Type1") dimnames <- list(pnames="status",cnames="Improved","Excellent","Poor")</pre>
Delimited text file <pre>mydata frame <- read.table("file", header=TRUE, as.is=TRUE, as.factor=FALSE, as.numeric=FALSE, as.logical=FALSE, as.character=TRUE, as.is.numeric=FALSE, as.is.logical=FALSE, as.is.character=TRUE)</pre>	<pre>mydata frame <- sqlFetch(channel, "myselect")</pre>	Data frames <pre>mydata <- data.frame(coll1=1:10, coll2=1:10, coll3=1:10)</pre>	<pre>patientdata <- data.frame(patientID, age, diabetes, status) patientdata[1:2,] patientdata[1:2, "diabetes", "status"] patientdata\$age # cross tabulate diabetes by status table(patientdata\$diabetes, patientdata\$status)</pre>
Excel file <pre>library(R ODBC) channel <- odbcConnectExcel("myfile.xlsx") mydata frame <- sqlFetch(channel, "mysheet") odbcClose(channel) or library(xlsx) workbook <- "c:/myworkbook.xlsx" mydata frame <- read.xlsx(workbook,1)</pre>	<pre>sqlDrop(channel, sqtable)</pre>	Factors <pre>status <- c("Poor", "Improved", "Excellent") status <- factor(status, levels=c("Poor", "Improved", "Excellent"))</pre>	<pre>patientdata\$age # cross tabulate diabetes by status table(patientdata\$diabetes, patientdata\$status)</pre>
Others XML, Webscrapping, SPSS, SAS, Stata, netCDF, HDF5, DBMSs	<pre>library(R ODBC) myconn <- odbcConnect("mydsn", uid="rob", pwd="aa", crimedat, myconn, Crimedat, pundat <- sqlQuery(myconn, "select * from Punishmen") close(myconn)</pre>	Lists <pre>mylist <- list(name1=object1, name2=object2, ...)</pre>	<pre>g <- TRUE Remove a table from the ODBC database j <- matrix(1:10, nrow=5) k <- c("one","two","three") mylist <- list(title=g, ages=h, j, k) Close the connection mylist[["ages"]]</pre>
Accessing DBMSs - RODBC package <pre>odbcConnect(dsn, uid="", pwd="")</pre>	<pre>sqlFetch(channel, sqtable)</pre>	Example code <pre>cells <- c(1,26,24,68) rnames <- c("R1","R2") cnames <- c("C1","C2") mymatrix <- matrix(cells, nrow=2, ncol=2, byrow=TRUE, dimnames=list(rnames, cnames)) dim1 <- c("A1","A2") dim2 <- c("B1","B2","B3") dim3 <- c("C1","C2","C3","C4") z <- array(1:24, dim1, dim2, dim3) mymatrix <- matrix(vector, nrow=nrow, ncol=ncol, dimnames=list(dim1, dim2, dim3))</pre>	
	Data structures Vectors Hold numeric, character, or logical data.		
	Matrices 2D array where each element has the same mode		

Useful tips	Useful functions (cont)	Arithmetic operators
Attach, Detach and With <code>attach (mtcars)</code> <code>summary(mpg)</code> <code>plot(mpg, wt)</code> <code>detach (mtcars)</code> <code>with(mtcars, {</code> <code>summary(mpg, disp, wt)</code> <code>plot(mpg, disp)</code> <code>plot(mpg, wt)</code> <code>})</code> <code>with(mtcars, {</code> <code>nokeep stats <- summary(mpg)</code> <code>keepstats <-<- summary(mpg)</code> <code>})</code> <code>nokeep stats # object not found</code> <code>keepstats # object exists in global env with superscript</code> <code>-'</code>	<code>names(object)</code> <code>cbind(object, object, ...)</code> <code>rbind(object, object, ...)</code> <code>head(object)</code> <code>tail(object)</code> <code>rm(object, object, ...)</code> <code>newobject <- edit(object)</code> <code>fix(object)</code> <code>set.seed()</code>	Arithmetic operators + Names of Addition - components in Subtraction * an object Multiplication / Combines Division ^ objects as Exponentiation %>% Columns %<% Combines Modulus (x mod y) %/% objects as rows Integer division %/% Lists the first part Logical operators < Lists the last part <= of the object Less than or = Deletes one or equal to > more objects. Greater than >= (list=ls()) Greater than or >= will remove most equal to == objects. Exactly equal to != Edits object and Not equal to != Saves as Not x != newobject Not x != Edits in place x or y != To have reproducible results isTRUE(x) Test if x is TRUE
Useful functions		
<code>length(object)</code> Number of elements/components <code>dim(object)</code> Dimensions of an object <code>str(object)</code> Structure of an object <code>class(object)</code> Class or type of an object <code>mode(object)</code> How an object is stored		

