



Ruby and R

© 2010-2014. Protected by International Copyright law. All rights reserved worldwide.

Version: 3.0, 15 August 2015

This document remains the property of Red Centre Software Pty Ltd and may only be used by explicitly authorised individuals who are responsible for its safe-keeping and return upon request.

No part of this document may be reproduced or distributed in any form or by any means - graphic, electronic, or mechanical, including, but not limited to, photocopying, recording, taping, email or information storage and retrieval systems - without the prior written permission of Red Centre Software Pty Ltd.

Such permission is granted to current Ruby licencees.

Confidential

Ruby and R

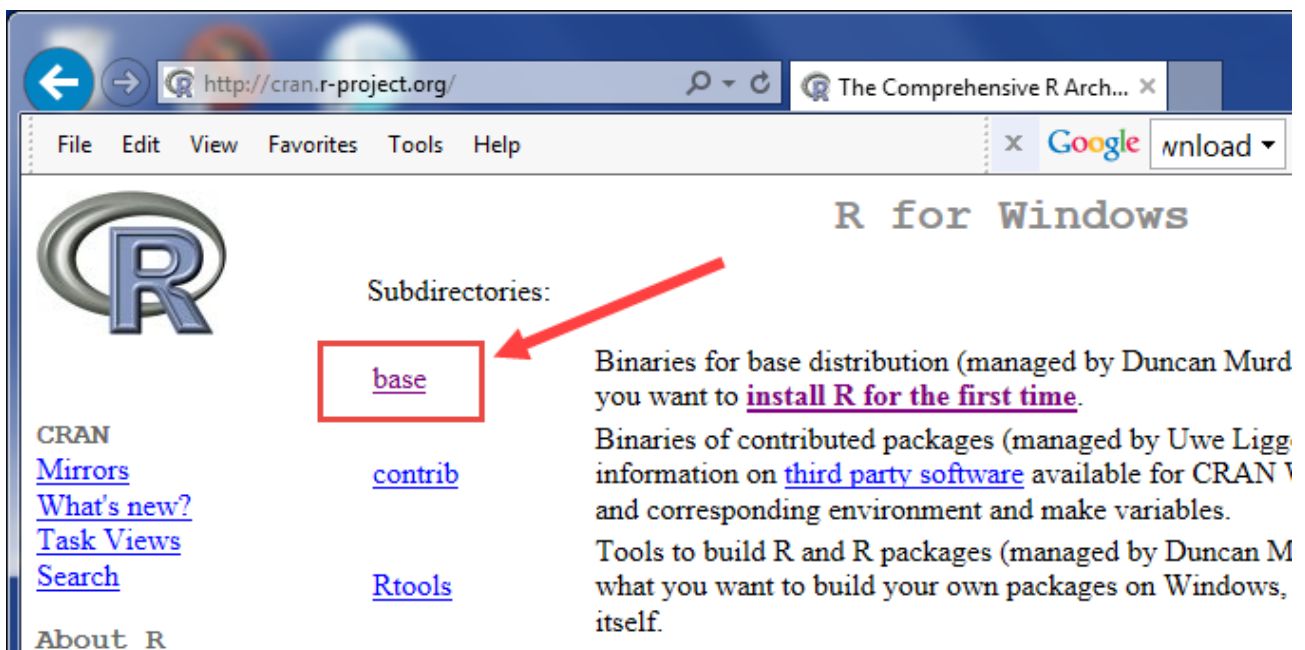
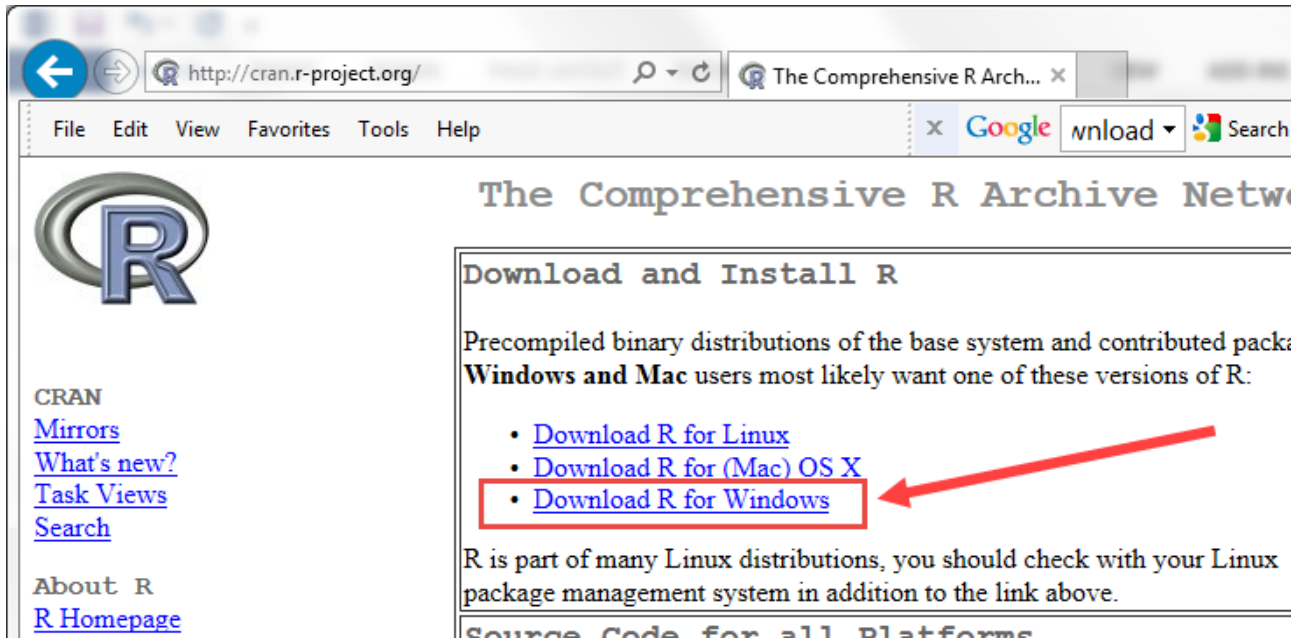
This document shows how to access, modify and execute R routines using a Ruby report as the input.

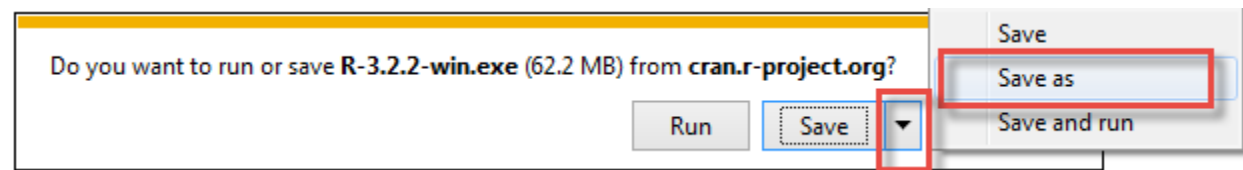
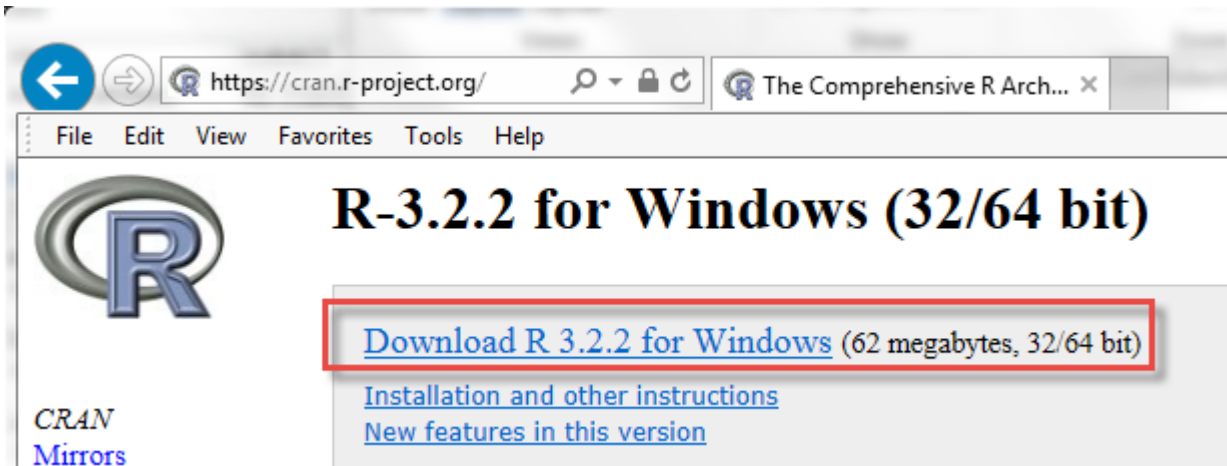
This is not an R tutorial. For details on any of the procedures, see the official R help. It is assumed that you know what you are doing, and can make the appropriate choices regarding inputs. The examples here, though useful in their own right, are intended primarily as the starting point for developing your own set of procedures.

INSTALLING R	3
Installing R Packages	6
INSTALLING THE REQUIRED RUBY FILES	8
Run_RSScripts.vbs.....	8
The R Procedures	10
The Example Reports.....	10
VIEW OR EDIT AN R SCRIPT	13
THE PROCEDURES	15
Correspondence Analysis	15
Principal Components Biplot	17
Principal Components Scree Plot.....	19
Cluster Dendrogram	21
Cluster Plot.....	23
K-Means Cluster Analysis	25
<i>Diagnostic and Chart Outputs</i>	28
<i>Comparing to Ruby Cluster</i>	29
Plot	30
Factor Analysis	34
Summary Stats	36
Covariance	38
Correlation	39
Regression Model.....	42
Anova	43
HANDY HINTS	46
MISCELLANEOUS	47

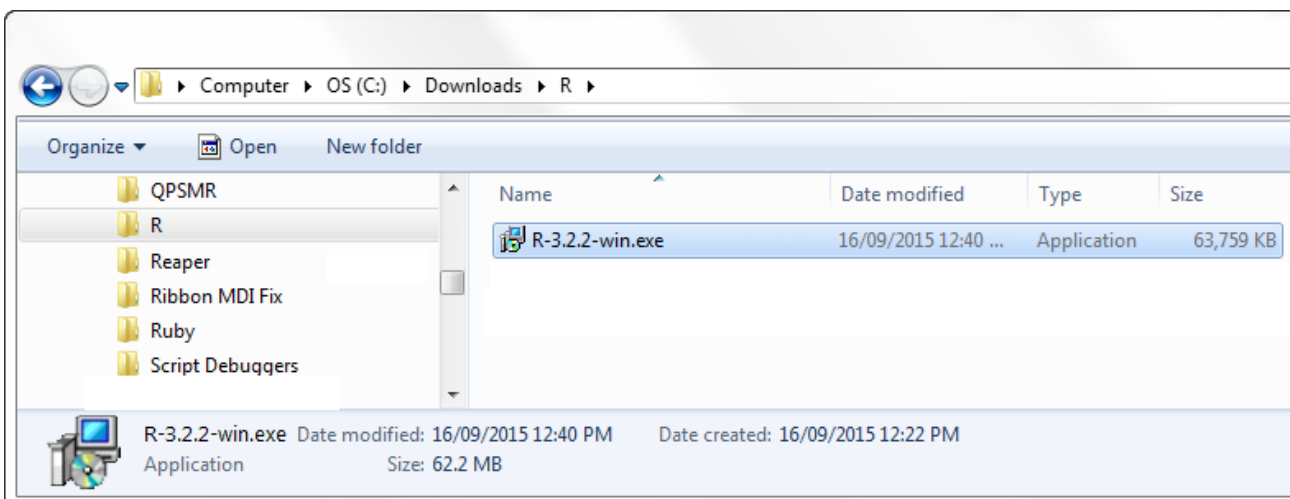
INSTALLING R

R can be installed from a single download at
<http://cran.r-project.org/>

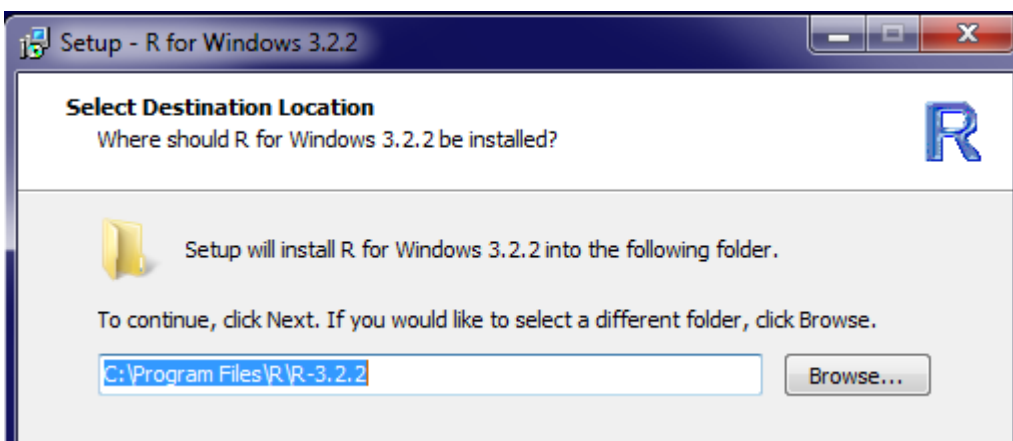




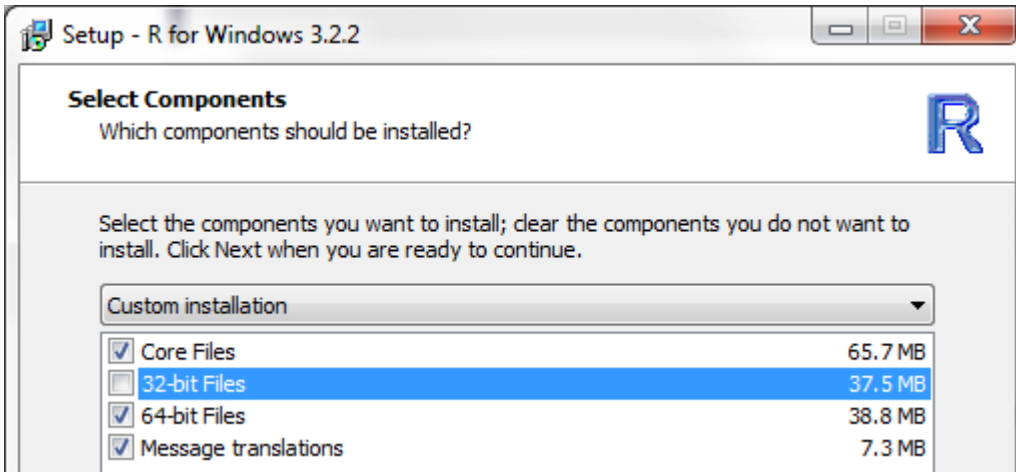
- Save to a subdirectory of your choice
- Run the self-extracting executable by a double-click



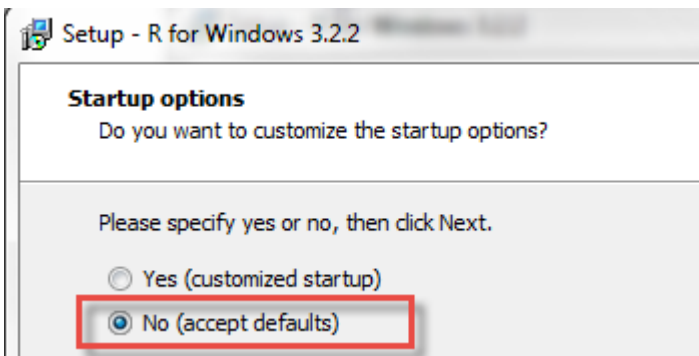
- Install to the default sub-directory



- Deselect the 32 bit option - we want 64 bit (unless you have only Windows 32)



- Accept defaults

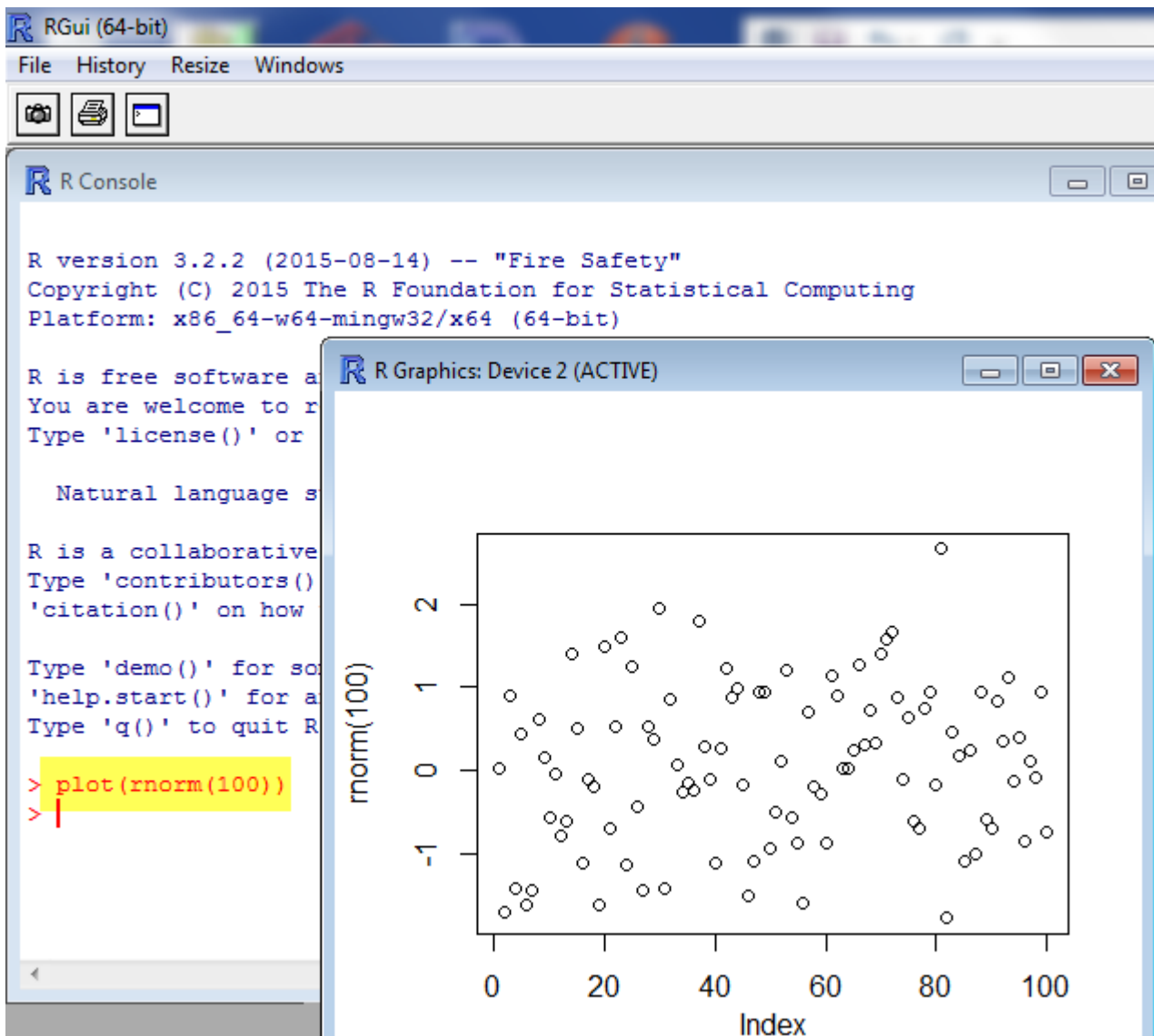


- Continue to the end of the installation procedure

R should now be on your desktop.



- Double-click to run
- Type the text `plot(rnorm(100))` at the command prompt and press Enter



A scatter plot appears.

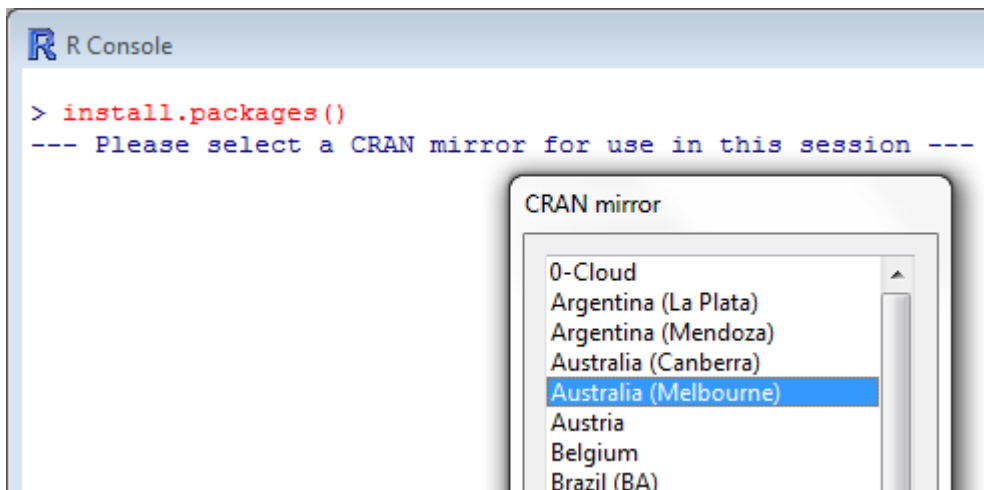
Installing R Packages

In addition to the base functionality, R supports third-party packages for special or extended procedures, and there are thousands of them. The R routines which need packages are

KMeans	psych
Plot	psych
Summary Stats	psych
Correlation	Hmisc, xlsx, corrgram, ellipse

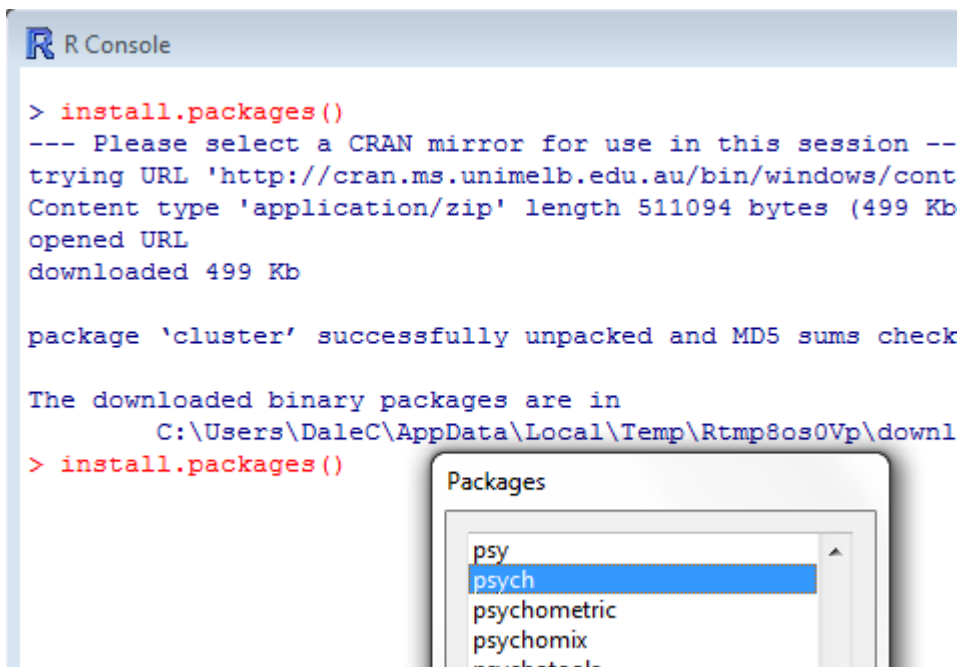
To install the psych package,

- Enter `install.packages()` at the R console, and then select a mirror which is geographically close



The Packages list should then appear.

- Scroll down to psych, select it and click OK



```
> install.packages()
trying URL 'http://cran.ms.unimelb.edu.au/bin/windows/contrib/3.1/psych_1.4.4.z$
Content type 'application/zip' length 2909325 bytes (2.8 Mb)
opened URL
downloaded 2.8 Mb

package 'psych' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\DaleC\AppData\Local\Temp\Rtmp8os0Vp\downloaded_packages
> |
```

The package is downloaded and installed. Repeat for Hmisc, xlsx, corrgram and ellipse.

INSTALLING THE REQUIRED RUBY FILES

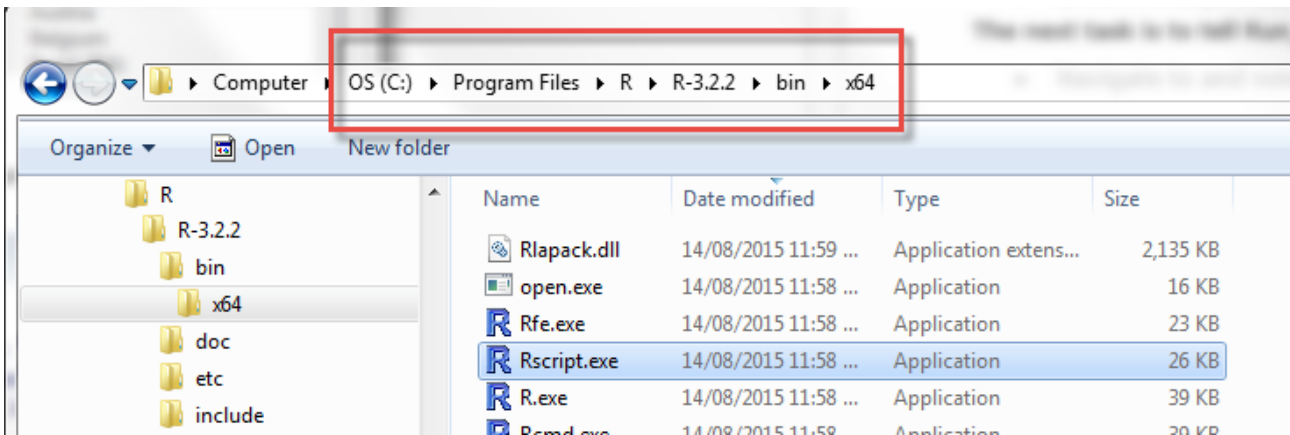
Ruby version 3.0 installations and later will have the files \Ruby\Run_RScripsts.vbs, thirteen R procedure files \Ruby*.r, this document \Ruby\Docs\Ruby and R.pdf, and four example reports \Ruby\Jobs\Demo\Reports\Session\r_*.rpt.

Run_RScripsts.vbs

- Ensure that you have the file \Ruby\Run_RScripsts.vbs

The next task is to tell Run_RScripsts.vbs where Rscript.exe is installed on your machine.

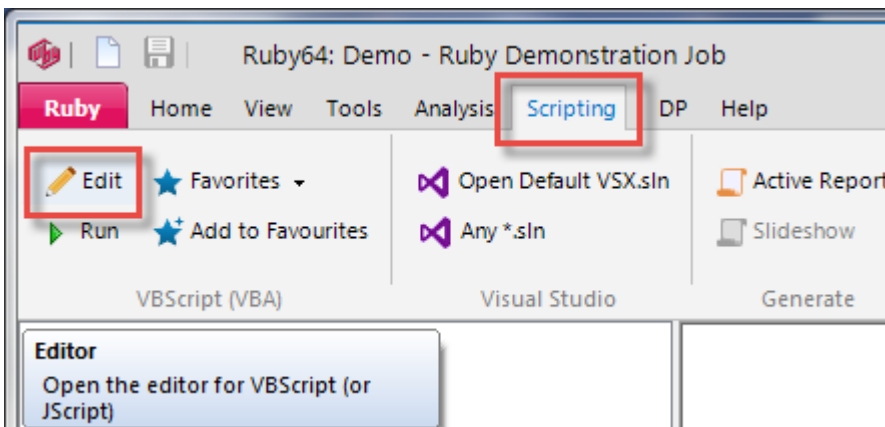
- Navigate to and note the path for Rscript.exe



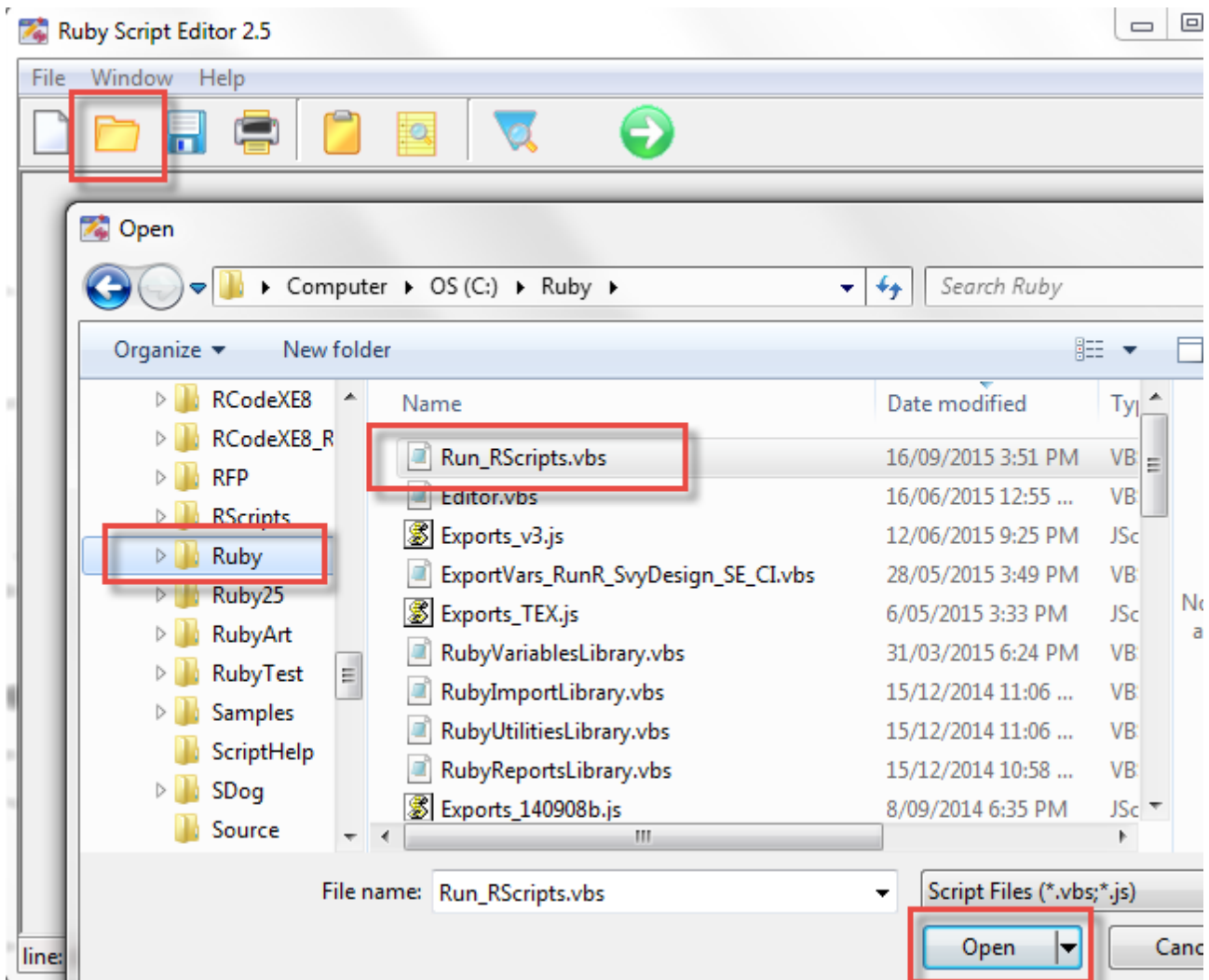
(If 32 bit R and OS, then select the Rscript.exe in just \bin, not \bin\x64.)

The path for this installation is c:\Program Files\R\R-3.2.2\bin\x64\Rscript.exe.

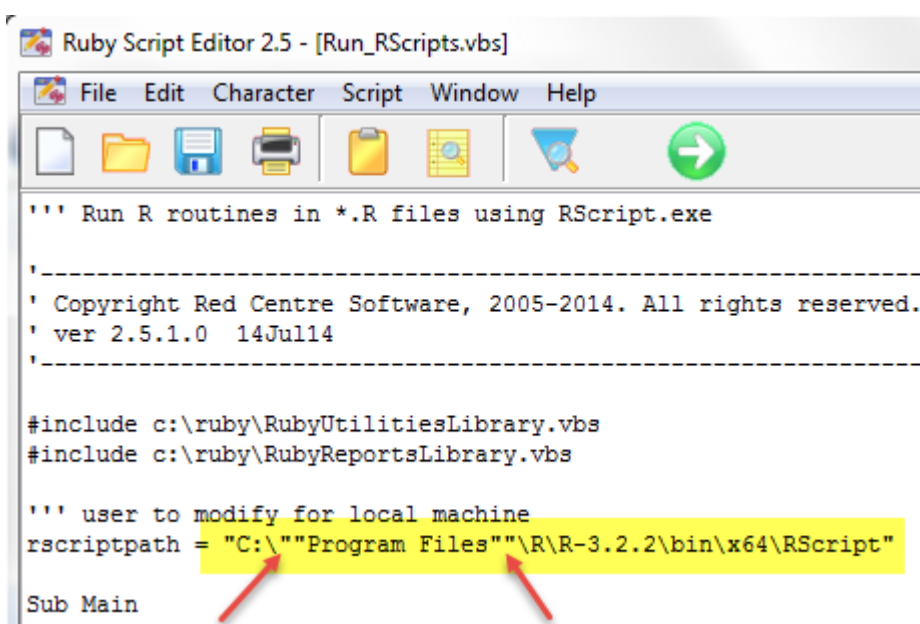
- Scripting | Edit



- File | Open



- Set the path assignment at line 12



Note the double-double quotes, needed because "Program Files" has a space.

The R Procedures

- Ensure that the following R scripts are in your \Ruby subdirectory:

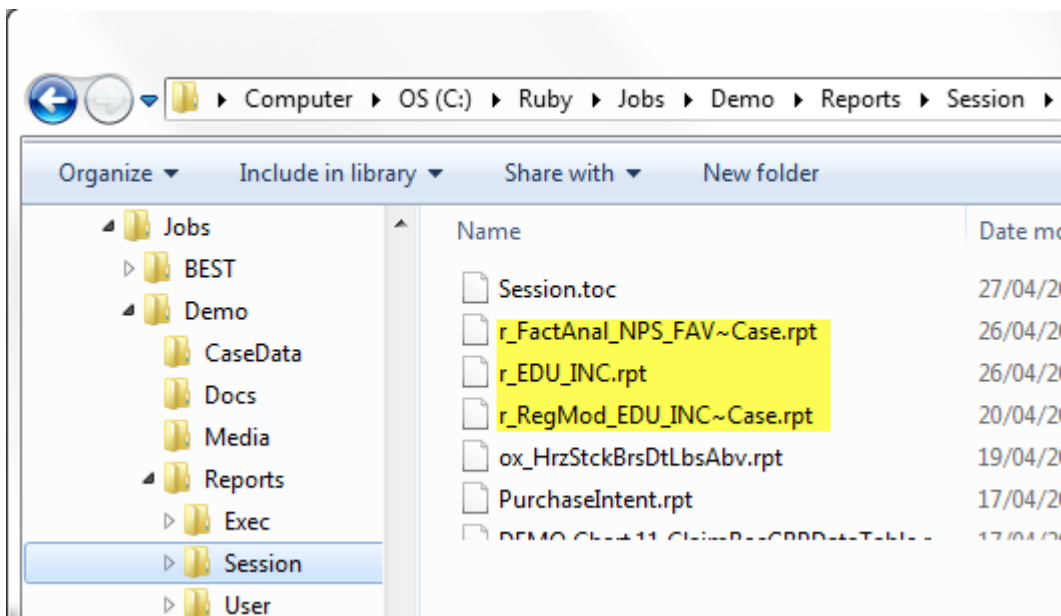
Anova.r
ClusterDendro.r
ClusterPlot.r
Correlation.r
CorrespAnaBiPlot.r
CoVariance.r
FactorAnalysis.r
InteractiveKMeans.r
Plot.r
PrinCompBiPlot.r
PrinCompScreePlot.r
RegressionModel.r
SummaryStats.r

These scripts implement the procedures. They are Ruby-specific, and will not work unless stored in the \Ruby subdirectory and are called by Run_RSScripts.vbs.

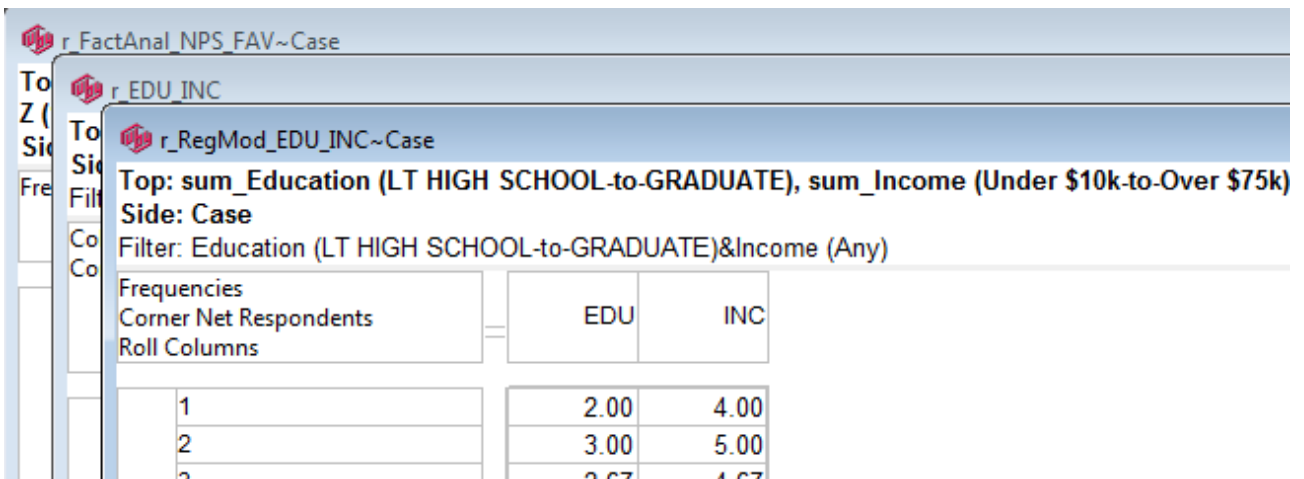
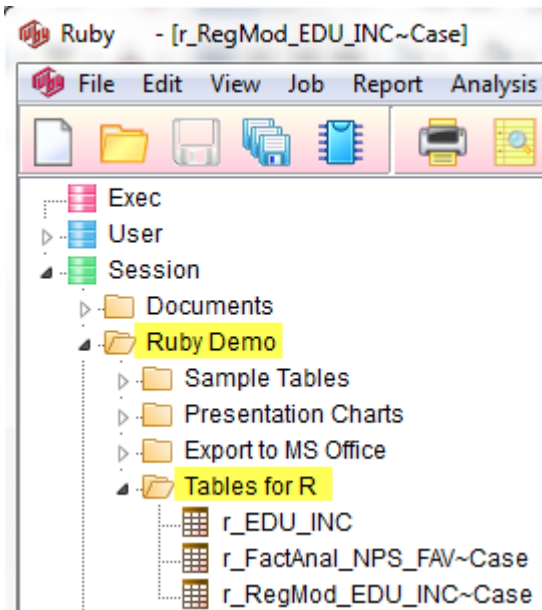
The Example Reports

Three reports are used as example inputs.

- Ensure that you have these three *.rpt in ..\Session



- Ensure that you can open these reports from the TOC



If you do not have all of the above files, then download the current Ruby install as a zip, and extract them to the appropriate sub-directories.

The two reports which use Education and Income use actual data from GSS2004, and as such there is a strong positive linear relationship between income and education.

r_EDU_INC		Education				
Top: Education		LT HIGH SCHOOL	HIGH SCHOOL	JUNIOR COLLEGE	BACHELOR	GRADUATE
Side: Income						
Filter: Education (LT HIGH SCHOOL-to-GRADUATE)&Income (Any)						
Column Percents Corner Net Respondents						
Income	Under \$10k	35%	19%	7%	13%	7%
	\$10k to 20k	28%	21%	13%	9%	9%
	\$20k to 30k	17%	21%	16%	16%	7%
	\$30k to 50k	14%	26%	38%	25%	28%
	\$50k to 75k	3%	10%	16%	21%	28%
	Over \$75k	4%	3%	11%	15%	22%

The NPS and Favourability variables used for factor analysis aggregate as

SReport1		Net Promoter Score Brand		
Top: Net Promoter Score Brand		Net Promoter Score Brand		
Side: Net Promoter Score Score		NPS - BrandX	NPS - BrandY	NPS - BrandZ
Column Percents Corner Net Respondents				
Net Promoter Score	1	3%	4%	3%
	2	11%	11%	11%
	3	4%	6%	3%
	4	5%	6%	6%
	5	6%	5%	6%
	6	10%	11%	11%
	7	8%	8%	13%
	8	20%	18%	15%
	9	23%	21%	26%
	10	9%	10%	7%
Net Promoter		-8%	-11%	-7%

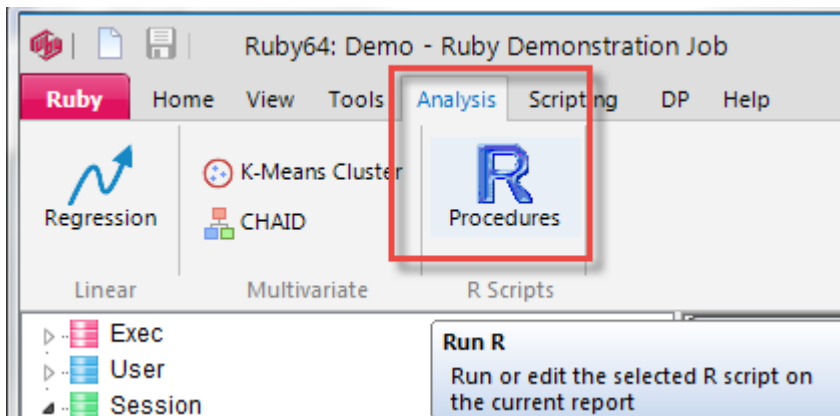
SReport2		Favourability Brand		
Top: Favourability Brand		Favourability Brand		
Side: Favourability Rating		BrandX	BrandY	BrandZ
Column Percents Corner Net Respondents				
Favourability Rating	1	0%	2%	7%
	2	1%	3%	16%
	3	3%	11%	29%
	4	1%	20%	22%
	5	14%	23%	14%
	6	23%	18%	6%
	7	22%	11%	3%
	8	19%	8%	1%
	9	14%	5%	1%
	10	4%	1%	1%
Code Mean		6.89	5.25	3.71

BrandZ is least favoured, but nonetheless well promoted.

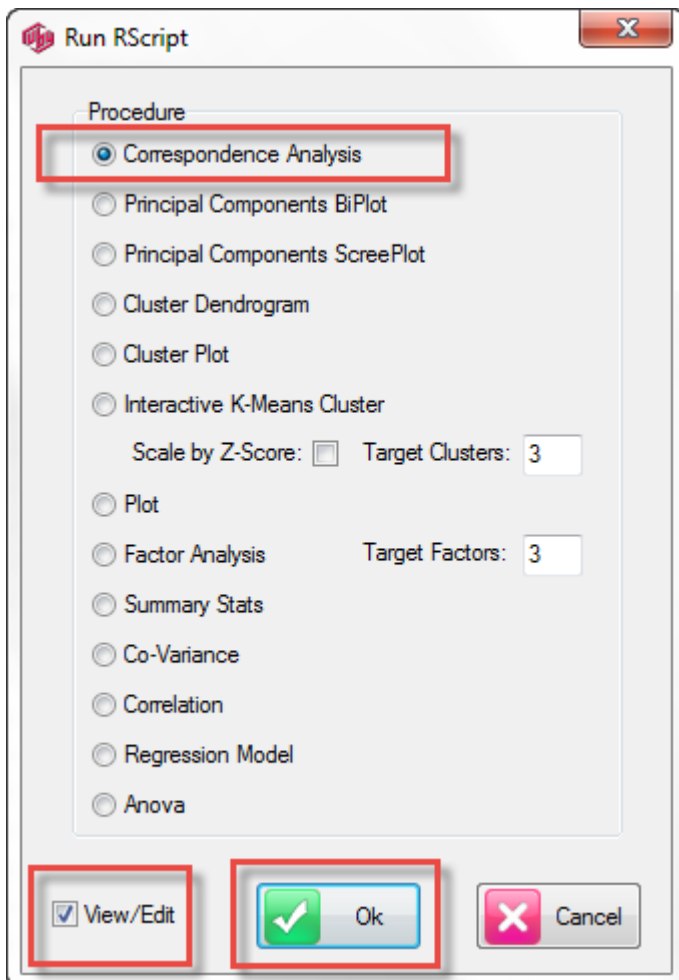
r_EDU_INC is a crosstab, whereas the other two are score tables, one row per case. Examine the specifications to see how they are made.

VIEW OR EDIT AN R SCRIPT

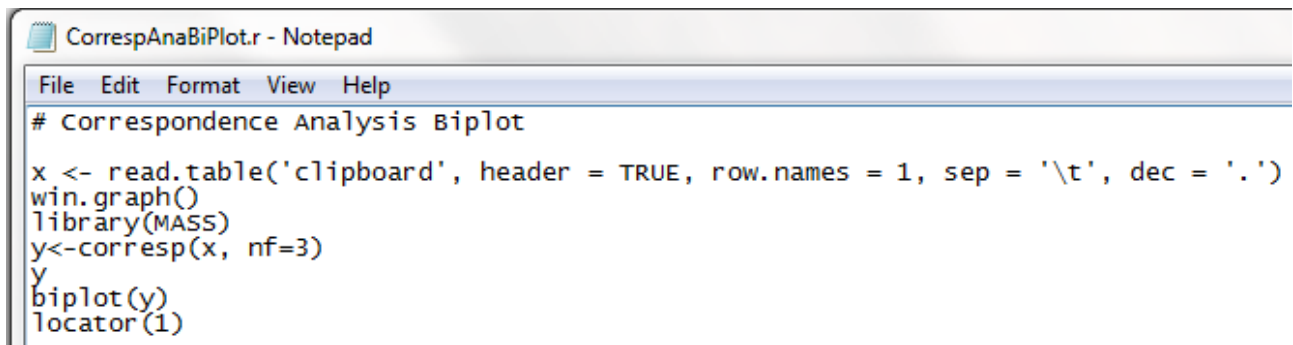
- Analysis | R Procedures



- Select Correspondence Analysis
- Check View/Edit
- Click OK



The R script CorrespAnaBiPlot.r is opened in Notepad.exe.



```
CorrespAnaBiPlot.r - Notepad
File Edit Format View Help
# Correspondence Analysis Biplot
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
win.graph()
library(MASS)
y<-corresp(x, nf=3)
y
biplot(y)
locator(1)
```

This is how you can both review what R is going to do, and edit the script to meet any custom requirements. Most of the supplied scripts are intentionally as basic as possible, while still doing something useful. The above R script

- Reads the Ruby table from the clipboard (placed there by Run_RSScripts.vbs) into x
- Sets up the Windows Graphics device for plotting
- Loads the MASS library (which defines the `corresp()` routine)
- Runs a correspondence analysis on x and stores the results in y
- Displays the contents of y in a Command window
- Displays the biplot of y on the graphics device
- Waits for a click on the graphics device to close it

THE PROCEDURES

Correspondence Analysis

- Open the supplied table r_EDU_INC
- Job | Scripting | Favourites | Run_RScript
- Select Correspondence Analysis
- Click OK

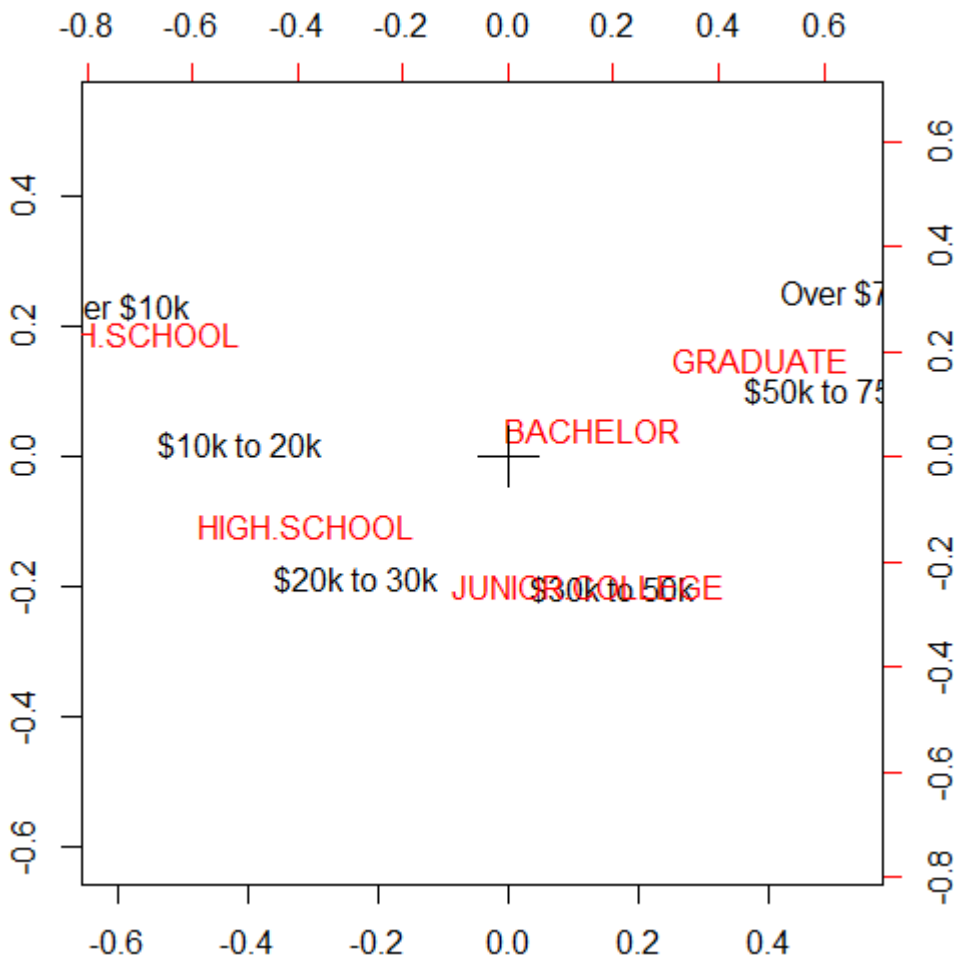
The screenshot shows the 'r_EDU_INC' application window. The main area displays a table with 'Education' as columns and 'Income' as rows. The 'Run RScript' dialog box is open, with 'Correspondence Analysis' selected under the 'Procedure' section.

		Education				
		LT HIGH SCHOOL	HIGH SCHOOL	JUNIOR COLLEGE	BACHELOR	GRADUATE
Income	Under \$10k	35%	19%	7%	13%	7%
	\$10k to 20k	28%				
	\$20k to 30k	17%				
	\$30k to 50k	14%				
	\$50k to 75k	3%				
	Over \$75k	4%				

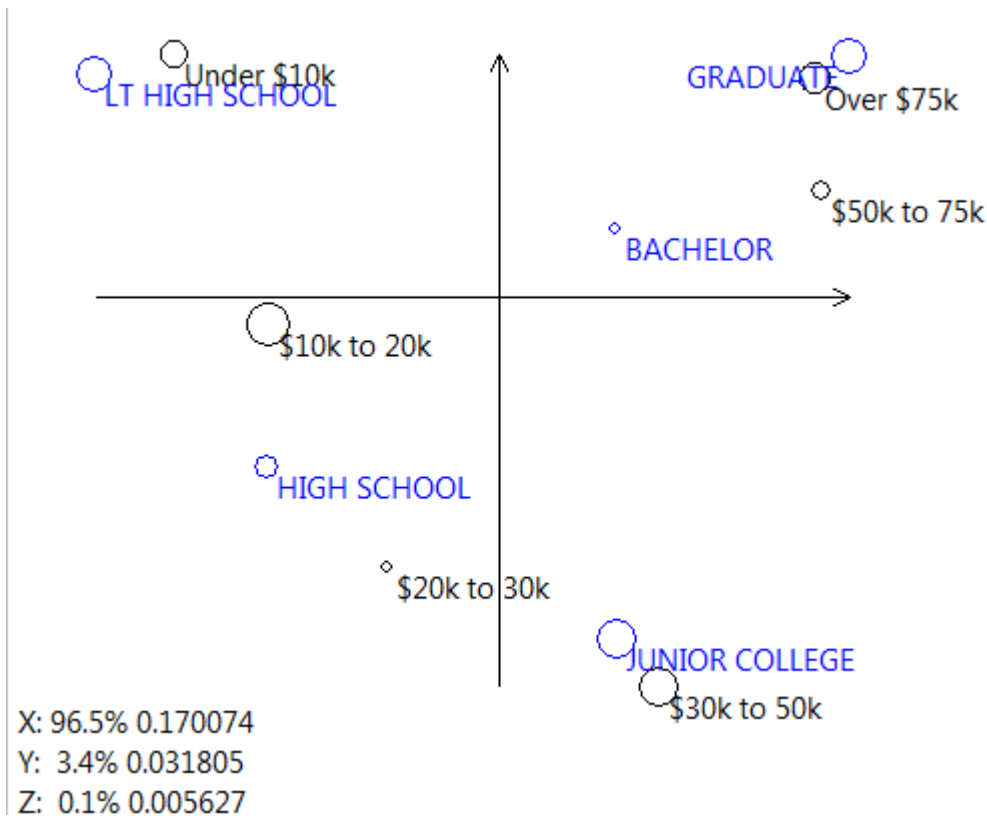
The outputs are

```

C:\Windows\System32\cmd.exe - C:\Program Files\R\R-3.1.0\bin\x64\RScript C:\Ruby\CorrespAnaBiPlot.r
First canonical correlation(s): 0.43765523 0.17607597 0.07067659
Row scores:
      [,1]      [,2]      [,3]
Under $10k -1.3822394  1.16122954 -0.3373648
$10k to 20k -1.0285936  0.04251221  1.2937772
$20k to 30k -0.4751312 -1.00482888 -1.7742088
$30k to 50k  0.4675504 -1.12863040  0.7272233
$50k to 75k  1.2320534  0.59447034 -0.7112454
Over $75k    1.3360857  1.47987545  0.3754423
Column scores:
      [,1]      [,2]      [,3]
LT.HIGH.SCHOOL -1.5761315  0.9364417  0.5331221
HIGH.SCHOOL    -0.6707487 -0.9714755 -0.5061525
JUNIOR.COLLEGE  0.4870062 -1.3773193  0.9042215
BACHELOR        0.5256343  0.3366075 -1.7511865
GRADUATE        1.2380073  1.0727926  0.7803070
  
```



The correspondence map is very close to the Ruby Perceptual Map, allowing for different scaling:



Principal Components Biplot

The R script is

```
PrinCompBiPlot.r - Notepad
File Edit Format View Help
# Principal Components Biplot
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
win.graph()
y<-princomp(x)
y
y$loadings
biplot(y)
locator(1)
```

If you are new to R, note the use of . and \$. In R, a dot is used like an underscore in most other programming languages. It has no intrinsic meaning - it is just a character. The name could just as well have been wingraph(), but the original programmer decided to call it win.graph(), and in most other languages, it would probably have been win_graph(). In short, the . does not mean 'member of', as it does in C, C#, .Net dialects, VBA, JScript, etc. The 'member of' symbol in R is \$, so y\$loadings means to display the loadings member of y, which has just been populated by Y<-princomp(x).

Also, the assignment operator in R is usually <-, although nearly always = will do the same thing. Using <- is considered better style.

On the same table as above, run as

Top: Education

Side: Income

Filter: Education (LT HIGH SCHOOL-to-GRADUATE)&Income (Any)

Column Percents Corner Net Respondents		Education				
		LT HIGH SCHOOL	HIGH SCHOOL	JUNIOR COLLEGE	BACHELOR	GRADUATE
Income	Under \$10k	35%	19%	7%	13%	7%
	\$10k to 20k	28%				
	\$20k to 30k	17%				
	\$30k to 50k	14%				
	\$50k to 75k	3%				
	Over \$75k	4%				

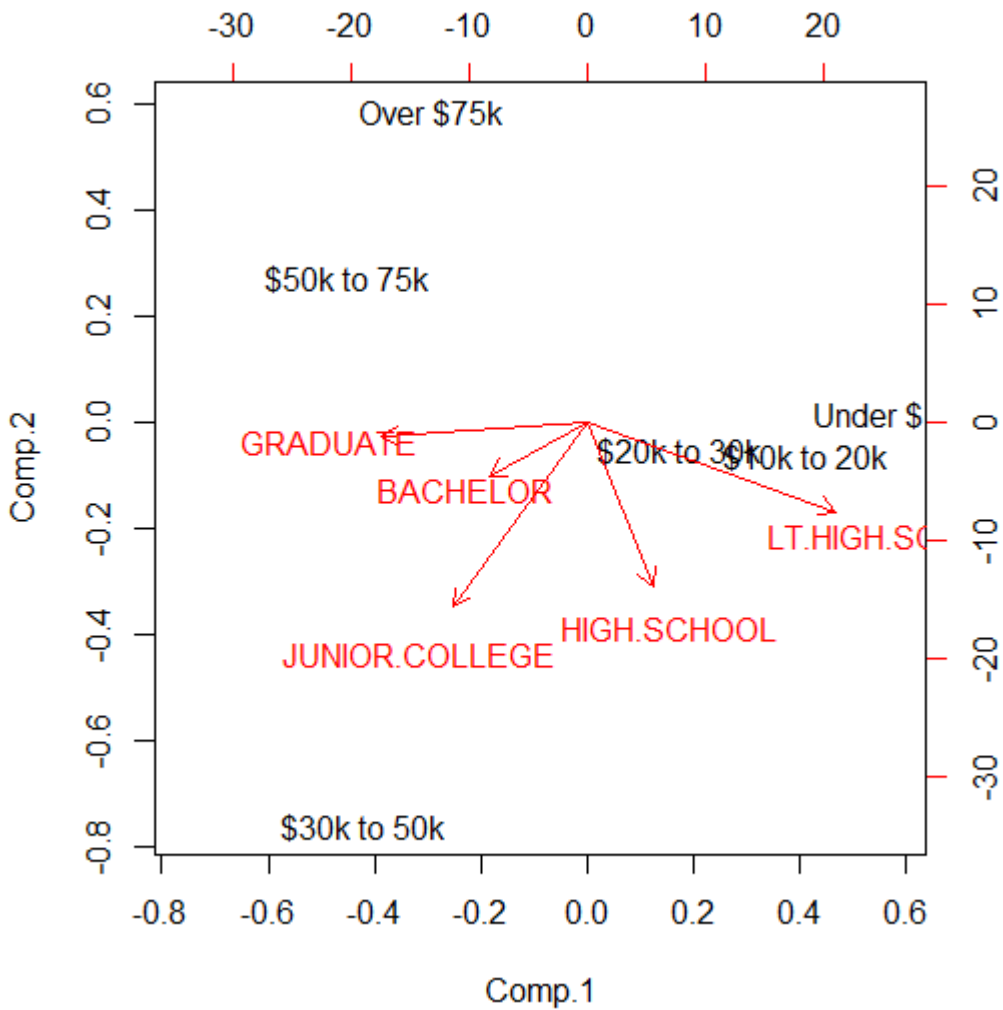
Run RScript

Procedure

- Correspondence Analysis
- Principle Components BiPlot
- Principle Components ScreePlot

The outputs are

```
cmd: C:\Windows\System32\cmd.exe - C:\Program Files\R\R-3.1.0\bin\x64\RScript C:\Ruby\PrinCompBiPlot.r
Call:
princomp(x = x)
Standard deviations:
  Comp.1  Comp.2  Comp.3  Comp.4  Comp.5
15.963837 11.623225 3.912949 2.187536 1.231106
5 variables and 6 observations.
Loadings:
      Comp.1  Comp.2  Comp.3  Comp.4  Comp.5
LT.HIGH.SCHOOL  0.673 -0.335  0.616  -0.234
HIGH.SCHOOL     0.178 -0.610 -0.259  0.157  0.711
JUNIOR.COLLEGE -0.364 -0.688 -0.145 -0.396 -0.465
BACHELOR        -0.265 -0.199  0.899 -0.282
GRADUATE        -0.559  0.727 -0.103  0.380
SS loadings     1.0  1.0  1.0  1.0  1.0
Proportion Var  0.2  0.2  0.2  0.2  0.2
Cumulative Var  0.2  0.4  0.6  0.8  1.0
```



Principal Components Scree Plot

The script is the same as for the biplot, except `screeplot()` is called instead.

```

PrinCompScreePlot.r - Notepad
File Edit Format View Help
# Principal Components Scree Plot
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
win.graph()
y <- princomp(x)
y
y$loadings
screeplot(y)
locator(1)

```

On the same table as above, run as

r_EDU_INC

Top: Education
Side: Income
 Filter: Education (LT HIGH SCHOOL-to-GRADUATE)&Income (Any)

Column Percents Corner Net Respondents		Education				
		LT HIGH SCHOOL	HIGH SCHOOL	JUNIOR COLLEGE	BACHELOR	GRADUATE
Inco me	Under \$10k	35%	19%	7%	13%	7%
	\$10k to 20k	28%				
	\$20k to 30k	17%				
	\$30k to 50k	14%				
	\$50k to 75k	3%				
	Over \$75k	4%				

Run RScript

Procedure

Correspondence Analysis

Principle Components BiPlot

Principle Components ScreePlot

The outputs are

```

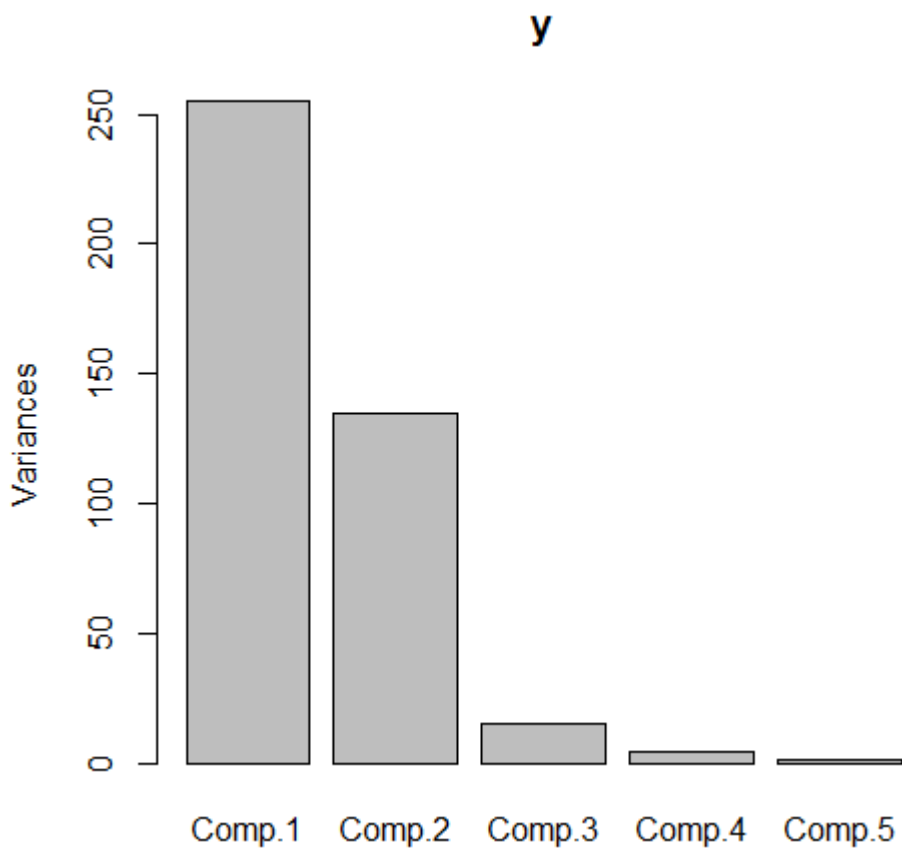
C:\Windows\System32\cmd.exe - C:\Program Files\R\R-3.1.0\bin\x64\RScript C:\Ruby\PrinCompScreePlot.r
Call:
princomp(x = x)

Standard deviations:
  Comp.1   Comp.2   Comp.3   Comp.4   Comp.5
15.963837 11.623225  3.912949  2.187536  1.231106

 5 variables and 6 observations.

Loadings:
      Comp.1  Comp.2  Comp.3  Comp.4  Comp.5
LT.HIGH.SCHOOL  0.673 -0.335  0.616  -0.234
HIGH.SCHOOL     0.178 -0.610 -0.259  0.157  0.711
JUNIOR.COLLEGE -0.364 -0.688 -0.145 -0.396 -0.465
BACHELOR        -0.265 -0.199  0.899 -0.282
GRADUATE        -0.559  0.727 -0.103  0.380

SS loadings      Comp.1  Comp.2  Comp.3  Comp.4  Comp.5
Proportion Var  0.2    0.2    0.2    0.2    0.2
Cumulative Var  0.2    0.4    0.6    0.8    1.0
  
```



This shows that there are two dominant components.

Cluster Dendrogram

The script is

```

ClusterDendro.r - Notepad
File Edit Format View Help
# Cluster Dendrogram
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
win.graph()
y <- dist(x)
y
z <- hclust(y, 'ave')
z
plot(z)
locator(1)

```

Run on the same table as above, as

r_EDU_INC

Top: Education
Side: Income
 Filter: Education (LT HIGH SCHOOL-to-GRADUATE)&Income (Any)

Column Percents Corner Net Respondents		Education				
		LT HIGH SCHOOL	HIGH SCHOOL	JUNIOR COLLEGE	BACHELOR	GRADUATE
Income	Under \$10k	35%	19%	7%	13%	
	\$10k to 20k	28%				
	\$20k to 30k	17%				
	\$30k to 50k	14%				
	\$50k to 75k	3%				
	Over \$75k	4%				

Run RScript

Procedure

- Correspondence Analysis
- Principle Components BiPlot
- Principle Components ScreePlot
- Cluster Dendrogram
- Cluster Plot

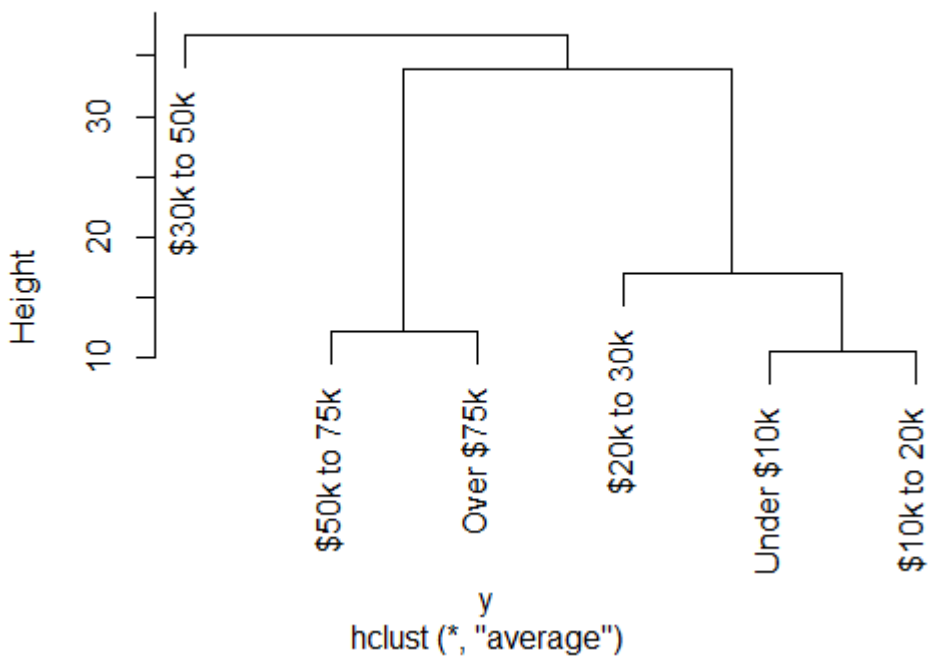
The outputs are

```
C:\Windows\System32\cmd.exe - C:\Program Files\R\R-3.1.0\bin\x64\RSscript C:\Ruby\ClusterDendro.r
Under $10k $10k to 20k $20k to 30k $30k to 50k $50k to 75k
$10k to 20k 10.44031
$20k to 30k 20.44505 13.52775
$30k to 50k 45.12206 38.24918 32.24903
$50k to 75k 41.12177 35.49648 27.98214 29.61419
Over $75k 38.23611 33.30165 27.27636 38.65230 12.12436

Call:
hclust(d = y, method = "ave")

Cluster method : average
Distance : euclidean
Number of objects: 6
```

Cluster Dendrogram



Cluster Plot

This script is a bit more complicated:

```
ClusterPlot.r - Notepad
File Edit Format View Help
# Cluster Plot
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
win.graph()
library(cluster)
y.diss <- daisy(x)
y.diss
y.clus <- pam(y.diss, 2, diss = TRUE)$clustering
y.clus
###uncomment the preferred clusplot
clusplot(y.diss, y.clus, diss = TRUE, col.p = y.clus, labels = 2) # color points and labels
#clusplot(y.diss, y.clus, diss = TRUE, labels=2, shade = TRUE)
#clusplot(y.diss, y.clus, diss = TRUE, labels=2, span = FALSE) # simple ellipses
locator(1)
```

Run on the same table as above, as

r_EDU_INC

Top: Education
Side: Income
 Filter: Education (LT HIGH SCHOOL-to-GRADUATE)&Income (Any)

Column Percents		Education	
Corner Net Respondents		LT HIGH SCHOOL	ST
Inco me	Under \$10k	35%	
	\$10k to 20k	28%	
	\$20k to 30k	17%	
	\$30k to 50k	14%	
	\$50k to 75k	3%	
	Over \$75k	4%	

Run RScript

Procedure

- Correspondence Analysis
- Principle Components BiPlot
- Principle Components ScreePlot
- Cluster Dendrogram
- Cluster Plot
- Interactive K-Means Cluster

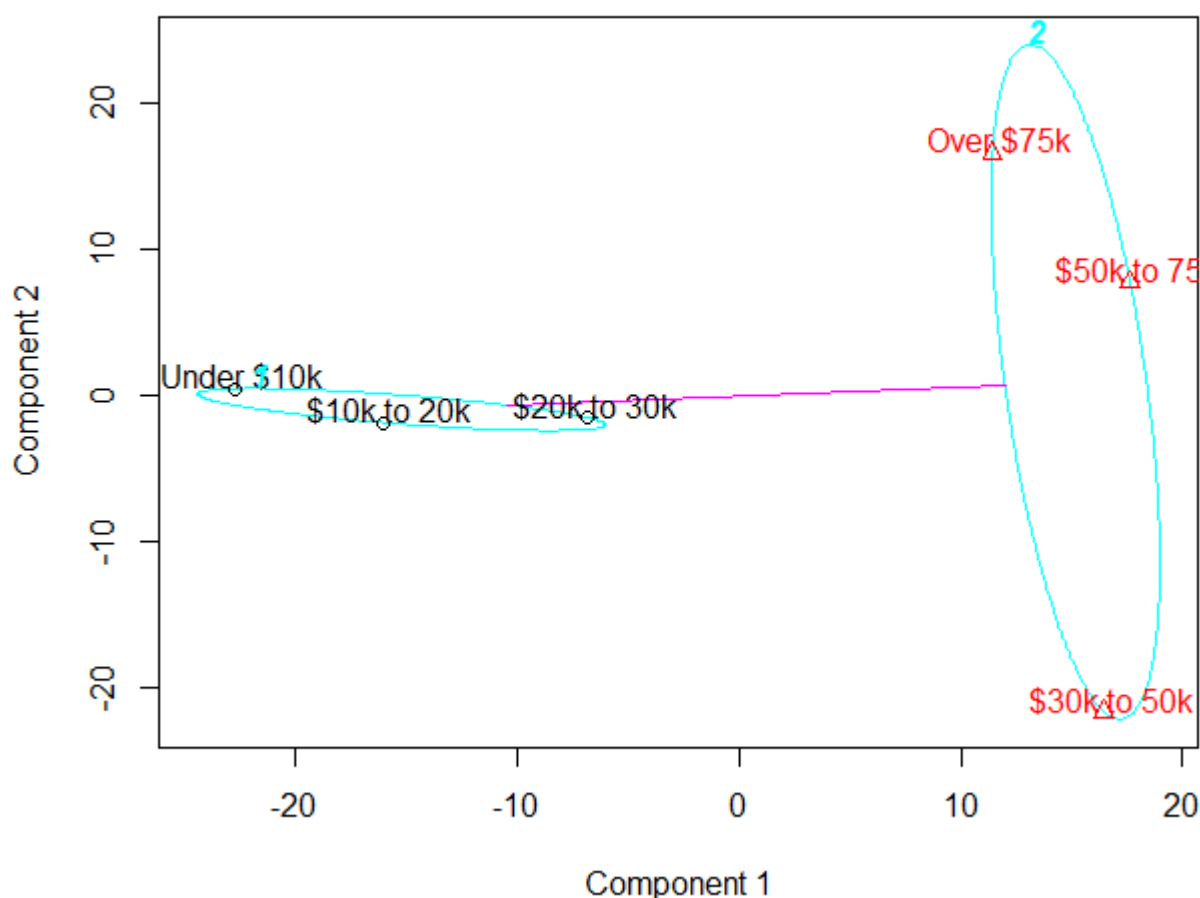
Outputs are

```

C:\Windows\System32\cmd.exe - C:\Program Files\R\R-3.1.0\bin\x64\RScript C:\Ruby\ClusterPlot.r
Dissimilarities :
      Under $10k $10k to 20k $20k to 30k $30k to 50k $50k to 75k
$10k to 20k  10.44031
$20k to 30k  20.44505      13.52775
$30k to 50k  45.12206      38.24918      32.24903
$50k to 75k  41.12177      35.49648      27.98214      29.61419
Over $75k    38.23611      33.30165      27.27636      38.65230      12.12436

Metric : euclidean
Number of objects : 6
      Under $10k $10k to 20k $20k to 30k $30k to 50k $50k to 75k Over $75k
          1         1         1         2         2         2
  
```


CLUSPLOT(y.diss)



These two components explain 94.75 % of the point variability.

K-Means Cluster Analysis

This is a much more sophisticated example. The script is too long to review here.

The analysis both determines the optimal number of clusters and does the clustering itself. On the first pass, the number of cluster targets is automatically set to number of rows -1, and you make an educated guess as to the most likely number of target clusters. You then examine the scree plots for an 'elbow' and use the elbow value as the target clusters for a second run if the educated guess was not appropriate. You may of course prefer a coarser clustering than the scree plots suggest, but there is never any reason to ask for more, since the extra clusters will be just noise.

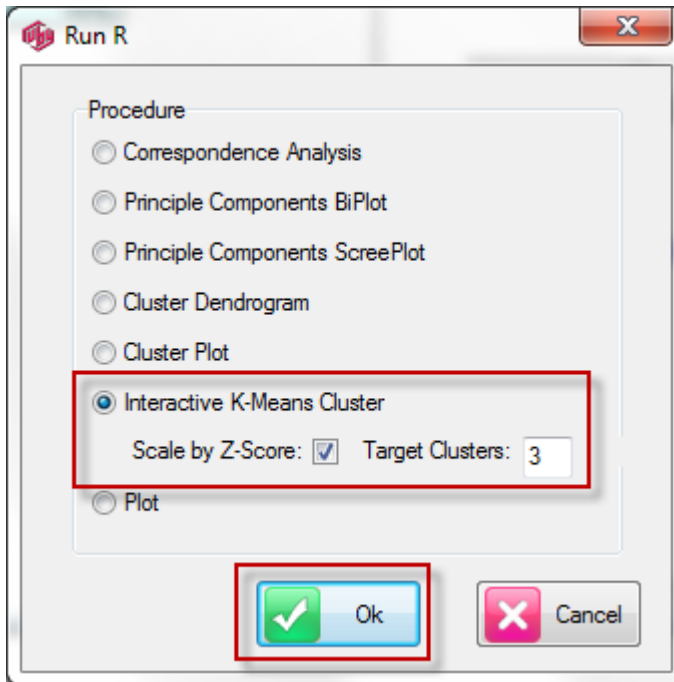
The k-means routine is adapted from

<http://www.mattpeeples.net/kmeans.html>

(Peeples, Matthew A. (2011) R Script for K-Means Cluster Analysis)

where the meaning of the various charts is fully discussed. The original version of this script has four user prompts: percents yes/no, zscore scaling yes/no, number of test clusters and number of target clusters. These inputs here come from the Ruby table (percent status) and from the Run_RScripts form, with test clusters set to number of rows -1. Apart from this, the only other difference is that a different example table is used.

Continuing with the above table, run as



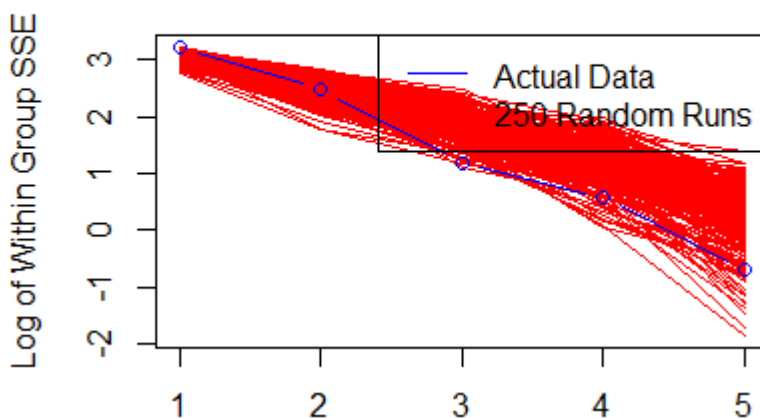
The Command window output is

```

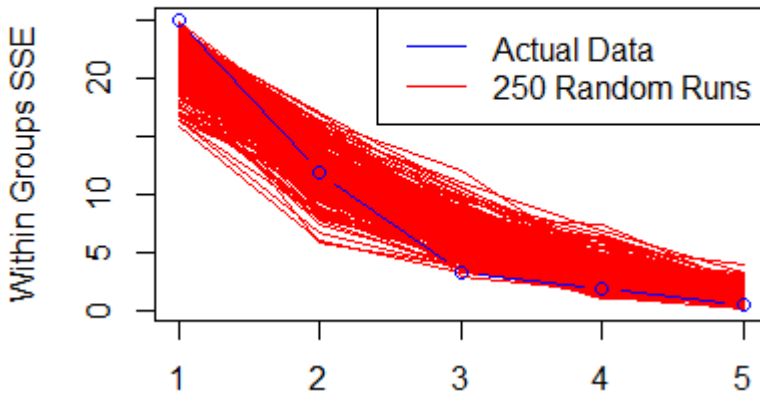
Administrator: C:\Windows\System32\cmd.exe
[1] "1" "1" "5" "3" "C"
      LT.HIGH.SCHOOL HIGH.SCHOOL JUNIOR.COLLEGE BACHELOR GRADUATE
Under $10k          35           19              7           13           7
$10k to 20k        28           21             13           9           9
$20k to 30k        17           21             16           16           7
$30k to 50k        14           26             38           25          28
$50k to 75k         3           10             16           21          28
Over $75k          4            3             11           15          22
Group.1 LT.HIGH.SCHOOL HIGH.SCHOOL JUNIOR.COLLEGE BACHELOR GRADUATE
1         1         0.7684012  0.4313228  -0.443133 -0.6703517 -0.8896462
2         2        -1.0418999 -1.1959405 -0.305609  0.2623115  0.7925939
3         3        -0.2214037  1.0979126  1.940617  1.4864320  1.0837509
pdf
2
  
```

Note the pdf - this means that a PDF file has been written. On closing the Command window, the PDF is displayed. Scroll down to see the charts.

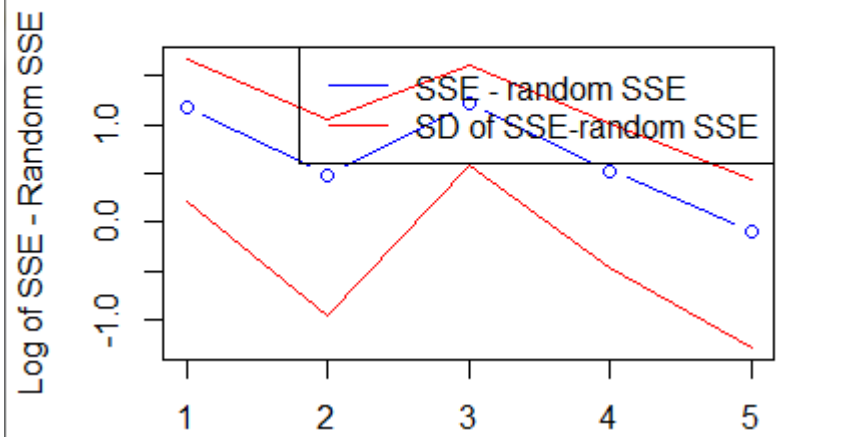
Cluster Solutions against Log of SSE



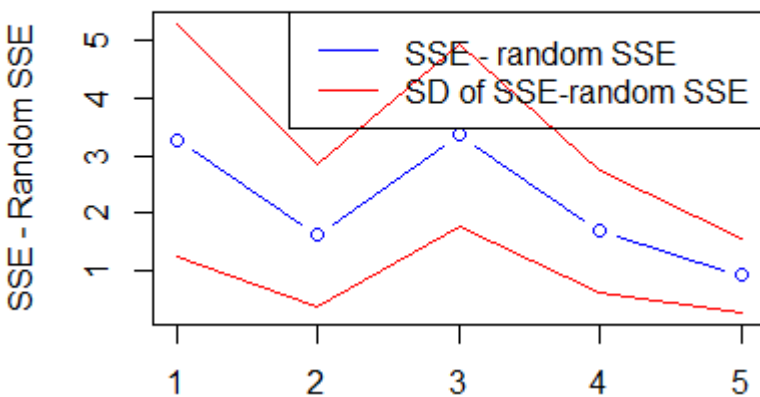
Cluster Solutions against SSE



Cluster Solutions against (Log of SSE - Random

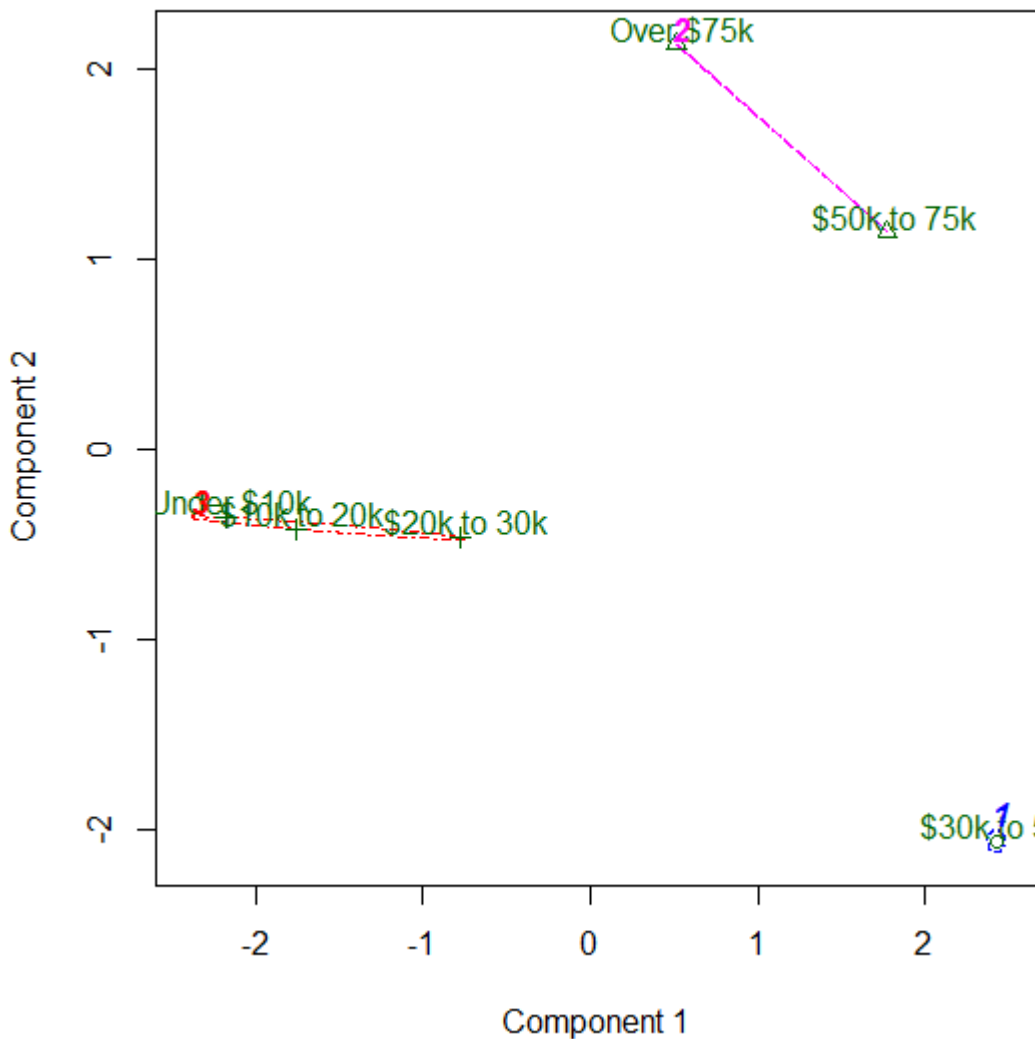


Cluster Solutions against (SSE - Random SS



Finally, the cluster plot is shown:

Principal Components plot showing K-means clusters

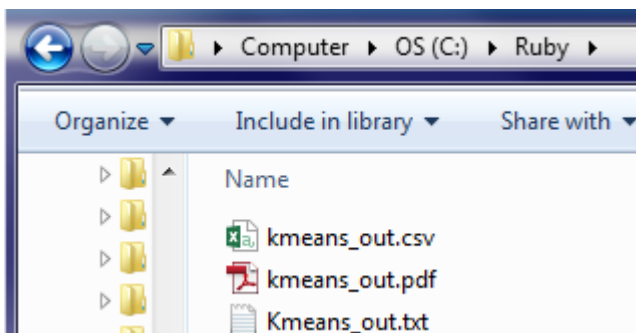


These two components explain 94.41 % of the point variability.

This shows three clear clusters as \$0-30k, \$30-50k and over \$50k.

Diagnostic and Chart Outputs

Three files are written to the \Ruby subdirectory:



kmeans_out.csv is the source table, Kmeans_out.txt has diagnostic details on the clustering, and kmeans_out.pdf for the charts as above.

Comparing to Ruby Cluster

	Cluster 1	Cluster 2	Cluster 3
LT HIGH			0.628
HIGH SC	1.000		
JUNIOR		0.315	
BACHEL		0.685	
GRADU			0.364
Under	0.126	0.079	0.116
\$10k to	0.136	0.077	0.101
\$20k to	0.140	0.117	0.065
\$30k to	0.166	0.213	0.112
\$50k to	0.063	0.139	0.082
Over \$75	0.021	0.097	0.069

EDU(6) No Answer has been removed.

- Run and save as EDU_INC_Clust3

To check the clusters

- Run the table INC, EDU by EDU_INC_Clust3

This shows that the three clusters are High School, College/Graduate and Less than High School. The highlighting on the left is an intuitive guess as to the income clusters.

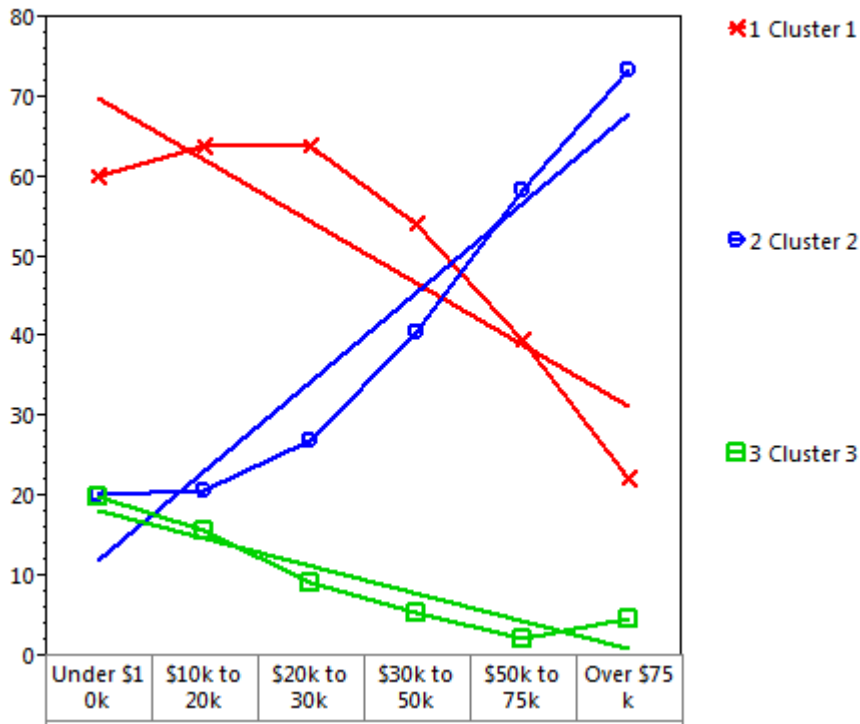
Top: Income, Education

Side: EDU_INC_Clust3

Column Percents		Income						Education				
		Under \$10k	\$10k to 20k	\$20k to 30k	\$30k to 50k	\$50k to 75k	Over \$75k	LT HIGH SCH	HIGH SCH	JUNIOR COL	BACHELOR	GRADUATE
EDU_IN	Cluster 1	60%	64%	64%	54%	40%	22%	0%	100%	0%	0%	0%
	Cluster 2	20%	21%	27%	41%	58%	73%	0%	0%	100%	100%	100%
	Cluster 3	20%	16%	9%	5%	2%	5%	100%	0%	0%	0%	0%

Notice how the Income rows are close to linear. A chart of Income by EDU_INC_Clust3 with trend lines is

Income by EDU_INC_Clust3



So, there is some difference about where to place \$30-50k between R and Ruby, but otherwise agreement on the cluster solution is reasonable.

Plot

This procedure simply demonstrates various plotting techniques. The R script is

```

Plot.r - Notepad
File Edit Format View Help
# Plot (chart) the table

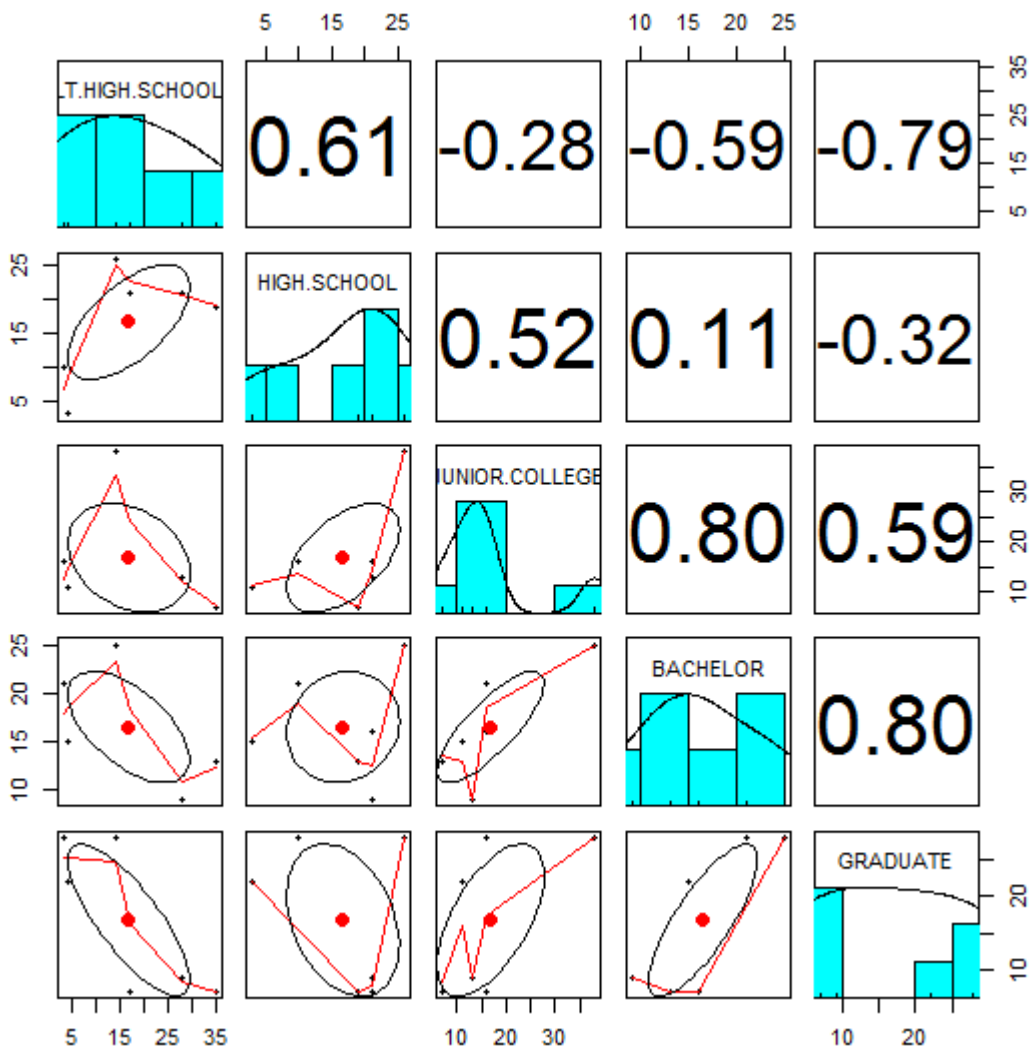
args <- commandArgs(trailingonly=T)
print(args)
title = args[1]
xlabel = args[2]
ylabel = args[3]

x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
win.graph()
# uncomment the required plot method, or add new ones
library(psych); pairs.panels(x)
#boxplot(x)
#plot(x)
#library(lattice); plot.new(); xyplot(ts(x))
#matplot(as.matrix(x), type = "b")
#barplot(as.matrix(x), main=title, xlab=xlabel, ylab=ylabel, col=c("blue","red","gre
locator(1)
    
```

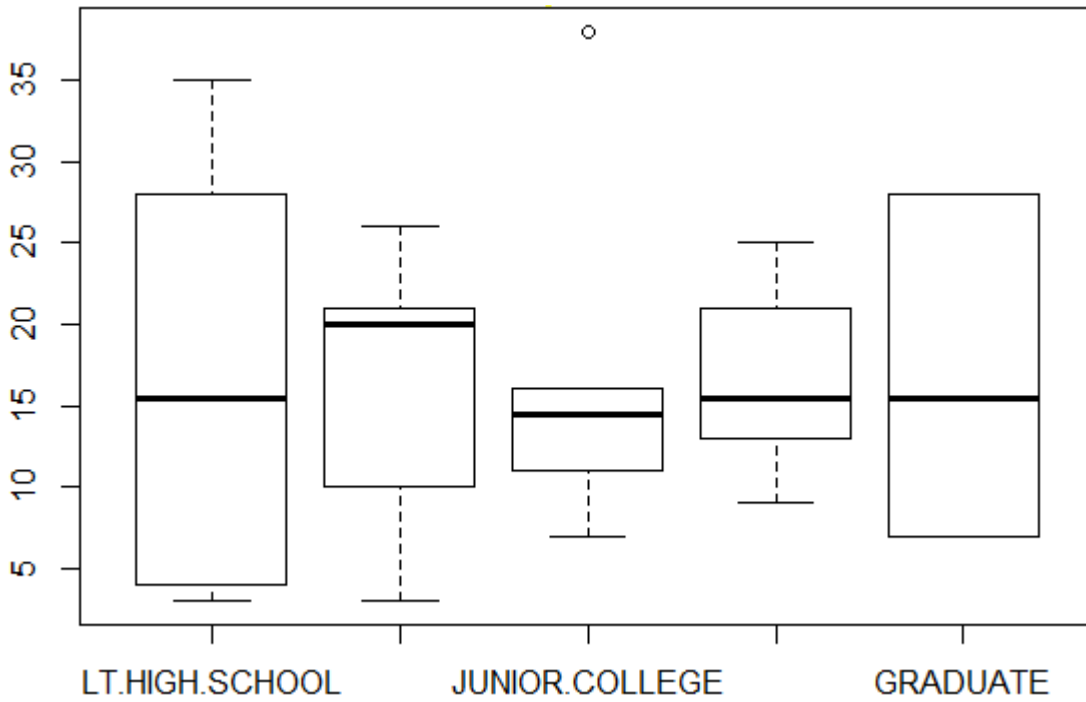
Only one of the plot methods can be uncommented.

The plot types are

pairs.panels:



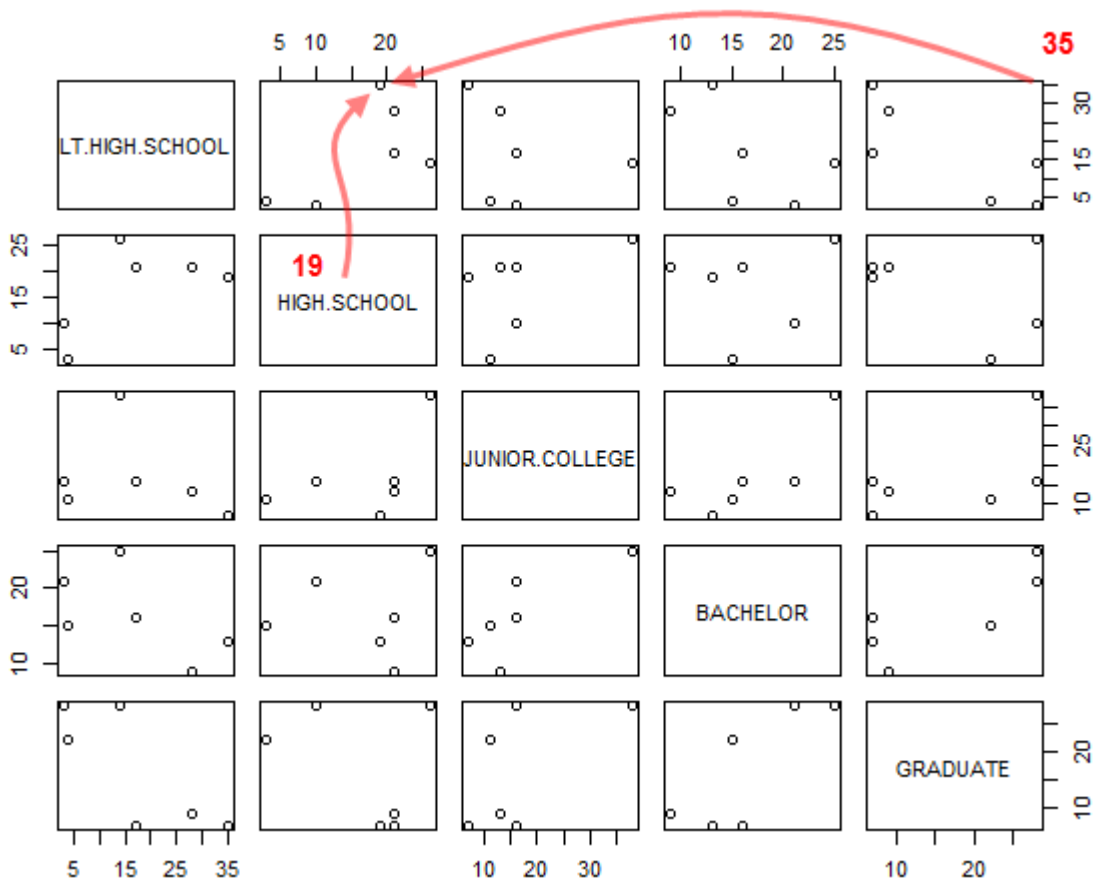
boxplot:



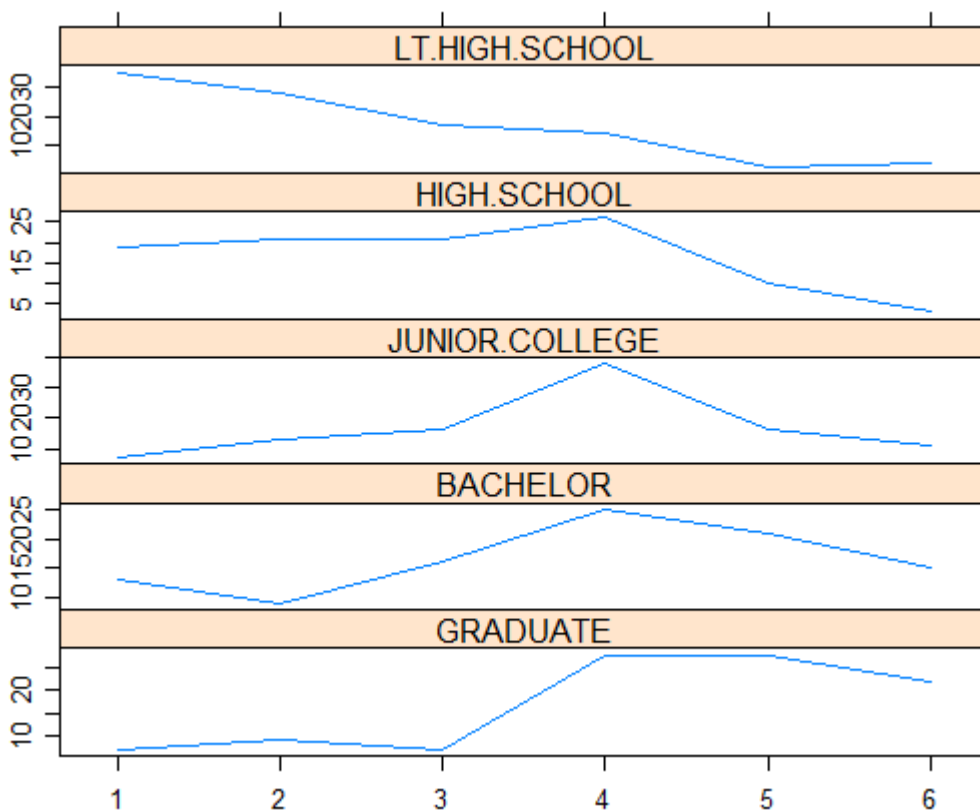
plot:

The annotated point is LT High School=35%, High School=19%

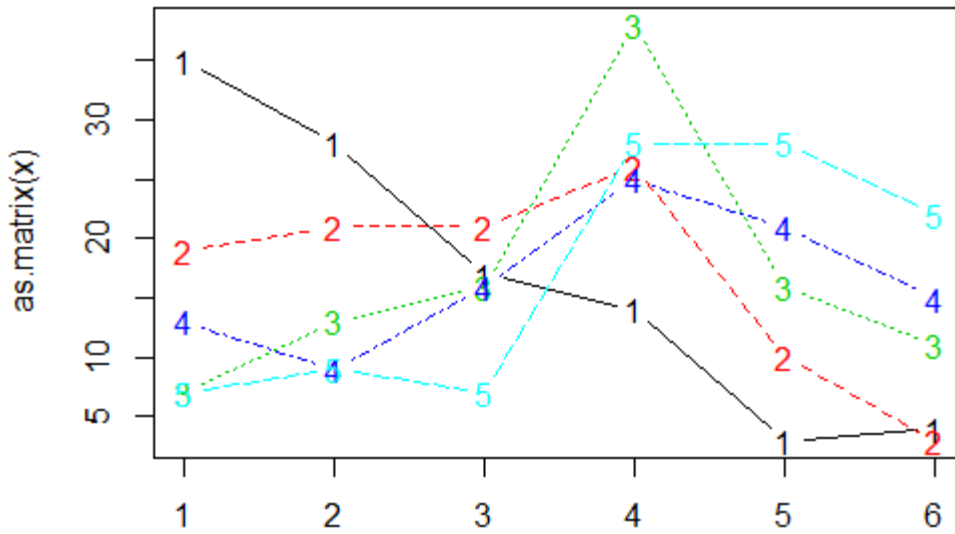
	LT HIGH SCHOOL	HIGH SCHOOL
Under \$10k	35%	19%
\$10k to 20k	28%	21%



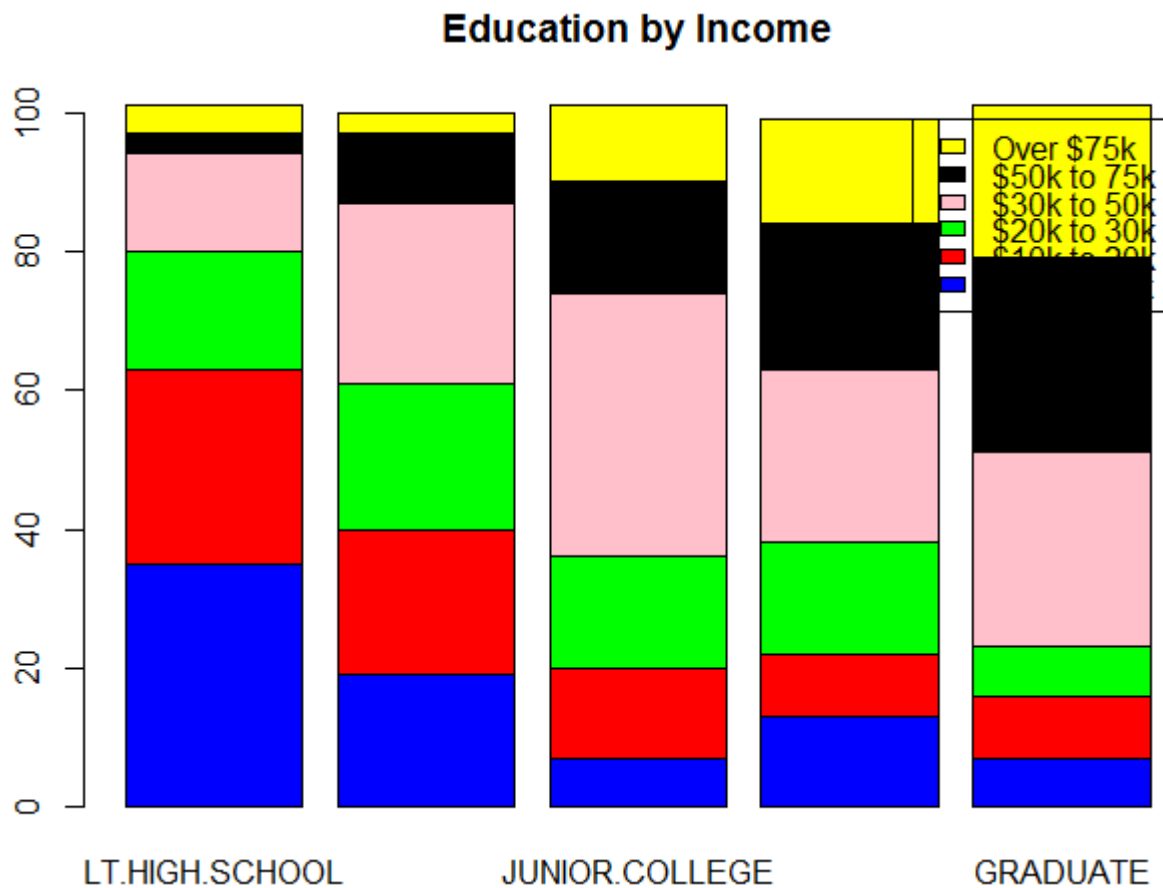
xyplot:



matplot:



barplot:

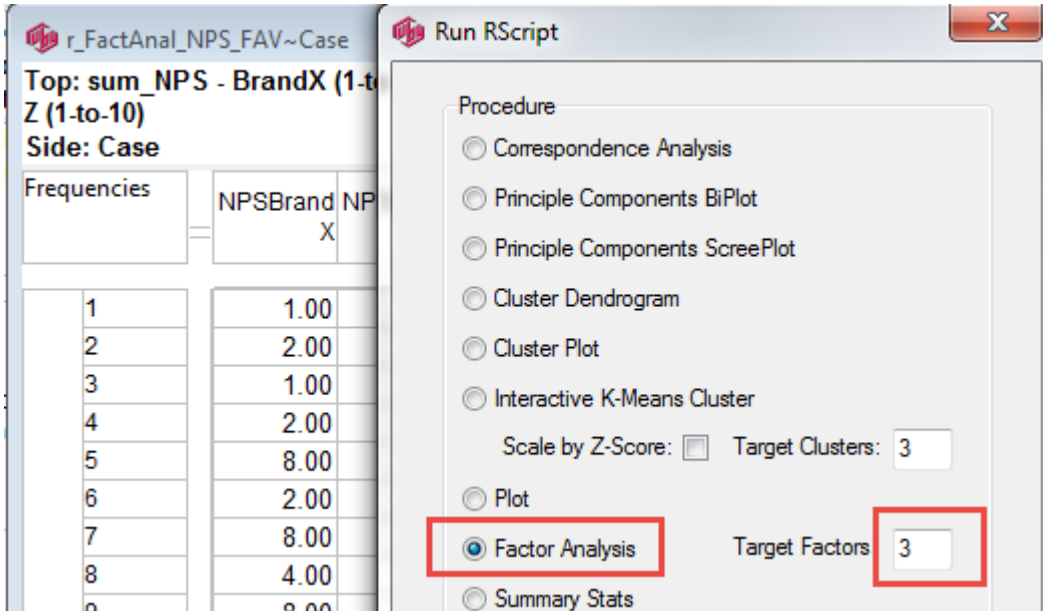


Factor Analysis

For factor analysis the rows would usually be cases, so a different report is used.

- Open the table r_FactAnal_NPS_FAV~Case

- Run as



The output is

```

Administrator: C:\Windows\System32\cmd.exe

Call:
factanal(x = x, factors = numfacs)

Uniquenesses:
NPSBrandX NPSBrandY NPSBrandZ FAUBrandX FAUBrandY FAUBrandZ
  0.972    0.938    0.974    0.991    0.659    0.943

Loadings:
          Factor1 Factor2 Factor3
NPSBrandX          0.160
NPSBrandY          0.240
NPSBrandZ          0.162
FAUBrandX
FAUBrandY    0.584
FAUBrandZ          0.238

SS loadings          Factor1 Factor2 Factor3
Proportion Var    0.057  0.018  0.012
Cumulative Var    0.057  0.075  0.087

The degrees of freedom for the model is 0 and the fit was 1e-04

```

Since factor analysis takes each row as a case, you could use it on r_EDU_INC by considering a 'case' as a circumstance, ie the income range.

- On the report r_EDU_INC, run as

r_EDU_INC

Top: Education
Side: Income
 Filter: Education (LT HIGH SCHOOL-to-GRADUATE)&Income (Any)

Column Percents
 Corner Net Respondents

Income	Percentage
Under \$10k	35%
\$10k to 20k	28%
\$20k to 30k	17%
\$30k to 50k	14%
\$50k to 75k	3%
Over \$75k	4%

Education

Run RScript

Procedure

- Correspondence Analysis
- Principle Components BiPlot
- Principle Components ScreePlot
- Cluster Dendrogram
- Cluster Plot
- Interactive K-Means Cluster
- Scale by Z-Score: Target Clusters: 3
- Plot
- Factor Analysis
- Summary Stats

Target Factors: 2

Note target factors = 2. The output is

```

Administrator: C:\Windows\System32\cmd.exe

Call:
factanal(x = x, factors = numfacs)

Uniquenesses:
LT.HIGH.SCHOOL    HIGH.SCHOOL  JUNIOR.COLLEGE    BACHELOR    GRADUATE
      0.135          0.005          0.028          0.181          0.089

Loadings:
      Factor1 Factor2
LT.HIGH.SCHOOL -0.548  0.751
HIGH.SCHOOL    0.221  0.973
JUNIOR.COLLEGE 0.933  0.320
BACHELOR       0.900  0.000
GRADUATE       0.806 -0.511

SS loadings      Factor1 Factor2
      2.679      1.883
Proportion Var  0.536      0.377
Cumulative Var  0.536      0.912

Test of the hypothesis that 2 factors are sufficient.
The chi square statistic is 0.4 on 1 degree of freedom.
The p-value is 0.527
  
```

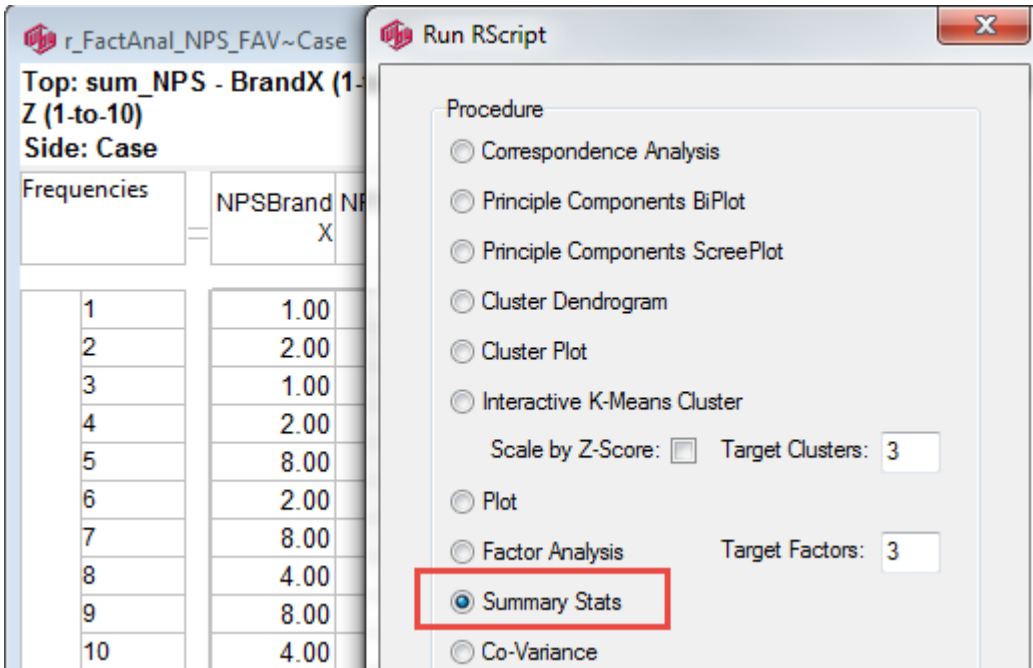
The loadings show a strong diagonal structure.

Summary Stats

The script is

```
SummaryStats.r - Notepad
File Edit Format View Help
# Summary Stats per column
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
summary(x)
library(psych); describe(x)
```

The summaries are by column, so this procedure will work on any table.
Using the factor analysis table above, run as



The output is in two parts:

```

Administrator: C:\Windows\System32\cmd.exe
NPSBrandX      NPSBrandY      NPSBrandZ      FAUBrandX
Min. : 1.000    Min. : 1.000    Min. : 1.000    Min. : 1.000
1st Qu.: 5.000  1st Qu.: 4.000  1st Qu.: 5.000  1st Qu.: 6.000
Median : 8.000  Median : 8.000  Median : 7.000  Median : 7.000
Mean   : 6.635  Mean   : 6.533  Mean   : 6.639  Mean   : 6.894
3rd Qu.: 9.000  3rd Qu.: 9.000  3rd Qu.: 9.000  3rd Qu.: 8.000
Max.   :10.000  Max.   :10.000  Max.   :10.000  Max.   :10.000
FAUBrandY      FAUBrandZ
Min. : 1.000    Min. : 1.000
1st Qu.: 4.000  1st Qu.: 3.000
Median : 5.000  Median : 3.000
Mean   : 5.251  Mean   : 3.713
3rd Qu.: 6.000  3rd Qu.: 5.000
Max.   :10.000  Max.   :10.000
summary()
vars      n mean  sd median trimmed mad min max range skew kurtosis
NPSBrandX 1 10000 6.63 2.69      8   6.84 1.48  1 10    9 -0.66  -0.85
NPSBrandY 2 10000 6.53 2.75      8   6.71 2.97  1 10    9 -0.58  -0.98
NPSBrandZ 3 10000 6.64 2.61      7   6.87 2.97  1 10    9 -0.70  -0.70
FAUBrandX 4 10000 6.89 1.62      7   6.94 1.48  1 10    9 -0.36   0.16
FAUBrandY 5 10000 5.25 1.85      5   5.20 1.48  1 10    9  0.19  -0.18
FAUBrandZ 6 10000 3.71 1.67      3   3.58 1.48  1 10    9  0.89   1.27
se
NPSBrandX 0.03
NPSBrandY 0.03
NPSBrandZ 0.03
FAUBrandX 0.02
FAUBrandY 0.02
FAUBrandZ 0.02
describe()

```

summary() gives 1st and 3rd quartiles.

describe() does not deliver quartiles, but gives the values for number of cases (rows), standard deviation, trimmed, mad, skew, kurtosis and standard error.

Both methods give min, max, median and mean.

Covariance

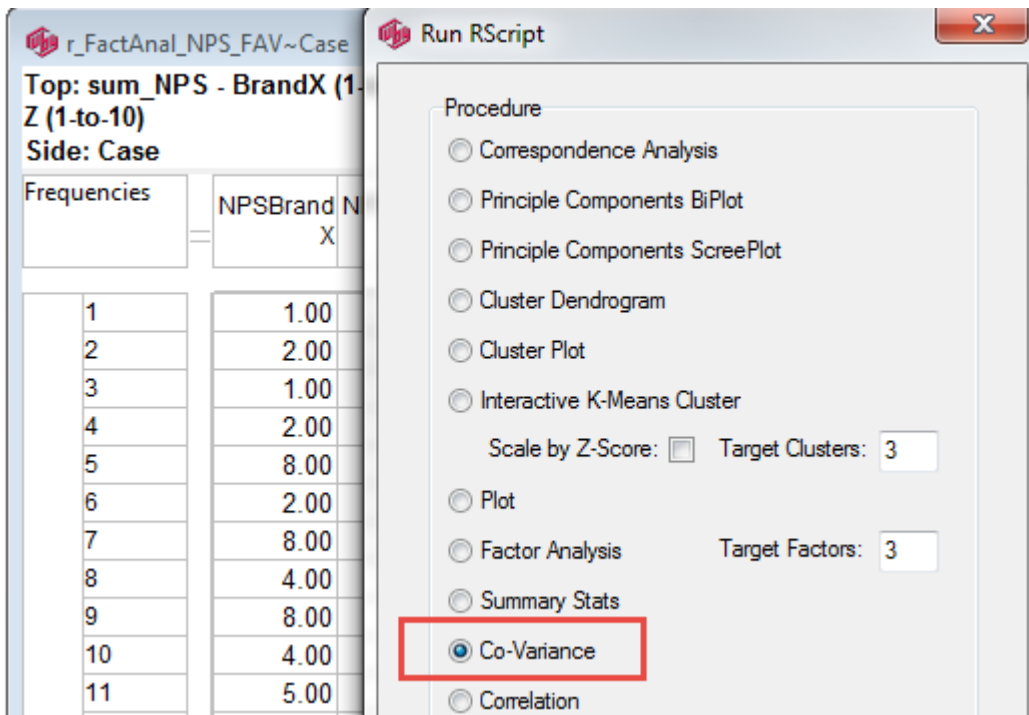
The script is

```

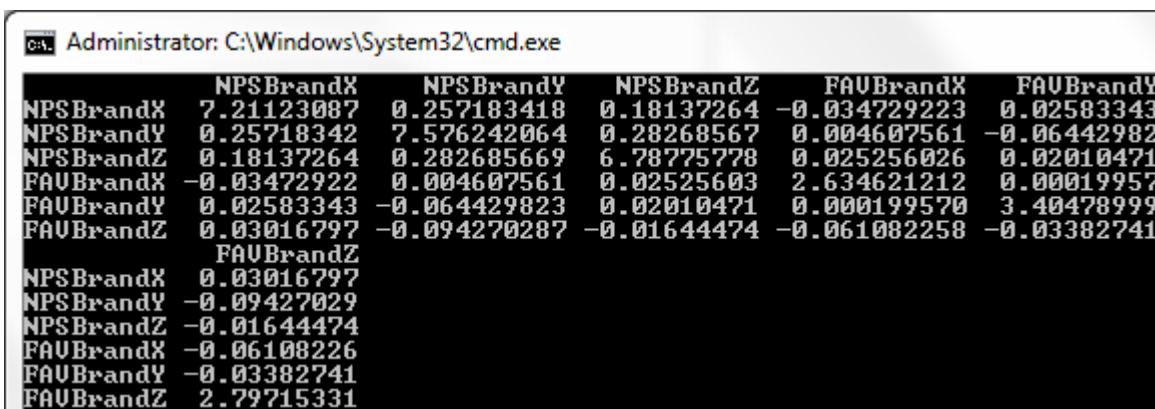
CoVariance.r - Notepad
File Edit Format View Help
# Co-variance matrix
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
var(x)

```

Call on the FactAnal table, as

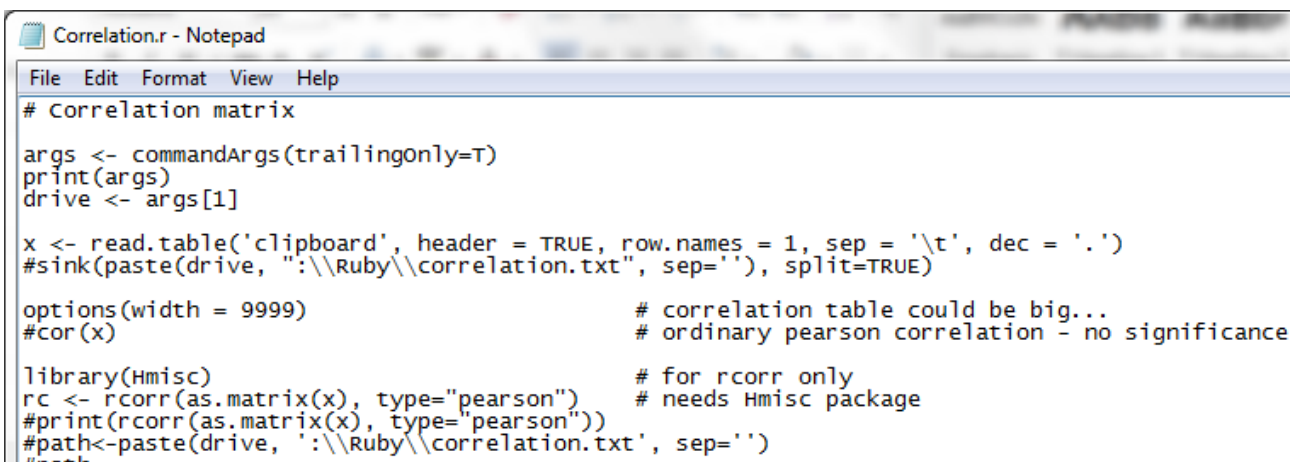


The output is



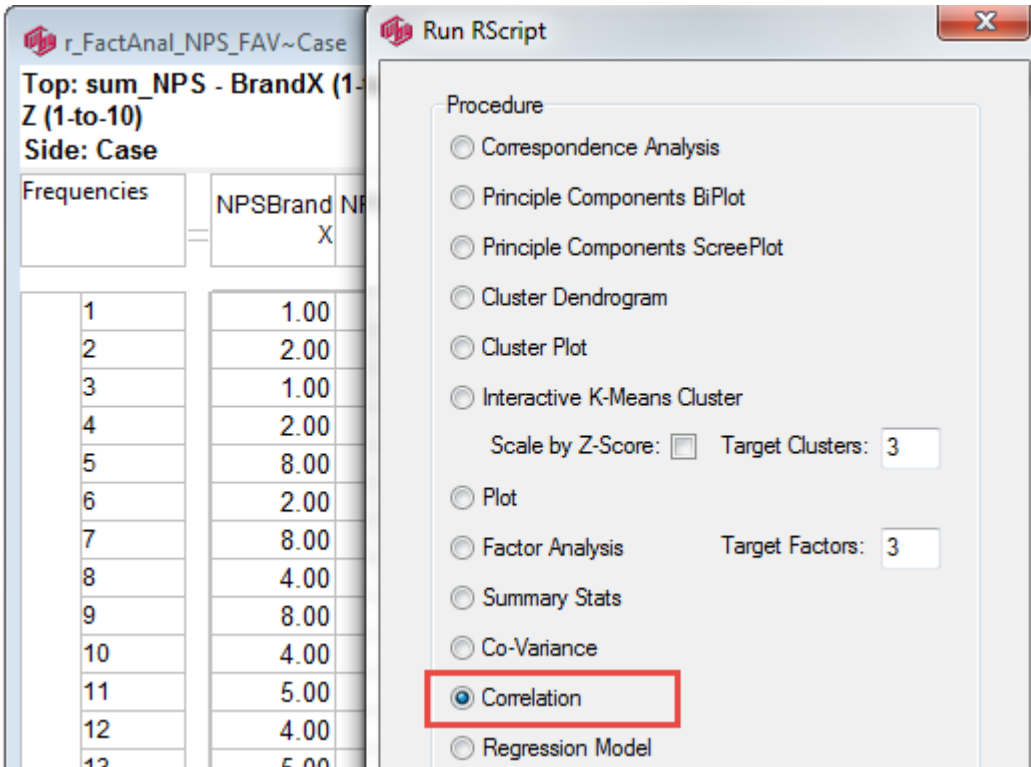
Correlation

The top part of the R script is



It is a bit too long to show all of it here. This script shows several techniques: sinking the output to a text file (this is commented out), how to call a simple Pearson correlation (commented out), how to call correlation with significance (active), and how to send output to Excel and a PDF.

Call on the FactAnal table, as



The runtime outputs are

```

Administrator: C:\Windows\System32\cmd.exe
[1] "C"
Loading required package: methods
Loading required package: grid
Loading required package: lattice
Loading required package: survival
Loading required package: Formula
Loading required package: ggplot2

Attaching package: 'Hmisc'

The following objects are masked from 'package:base':

  format.pval, round.POSIXt, trunc.POSIXt, units

Loading required package: rJava
Loading required package: xlsxjars
null device
1
C:\RCode\XE8\Ruby_TMS\Win64\Release>

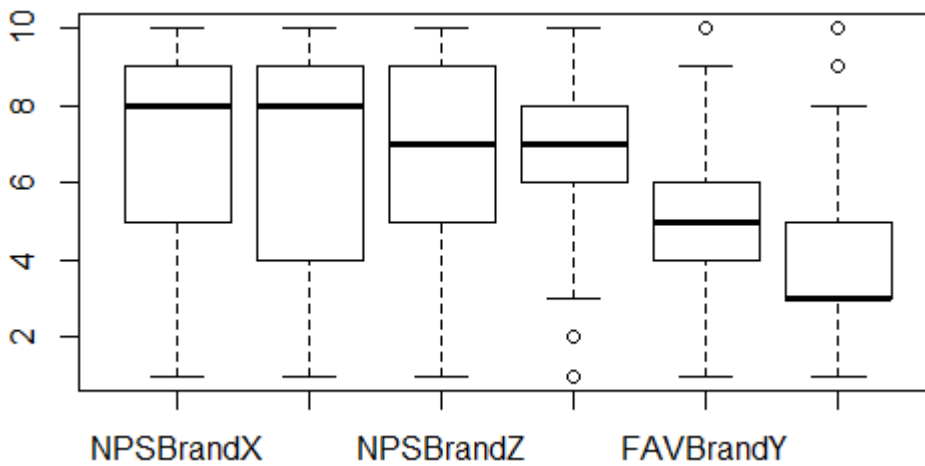
```

On closing the Command window, the three correlation tables are displayed in Excel:

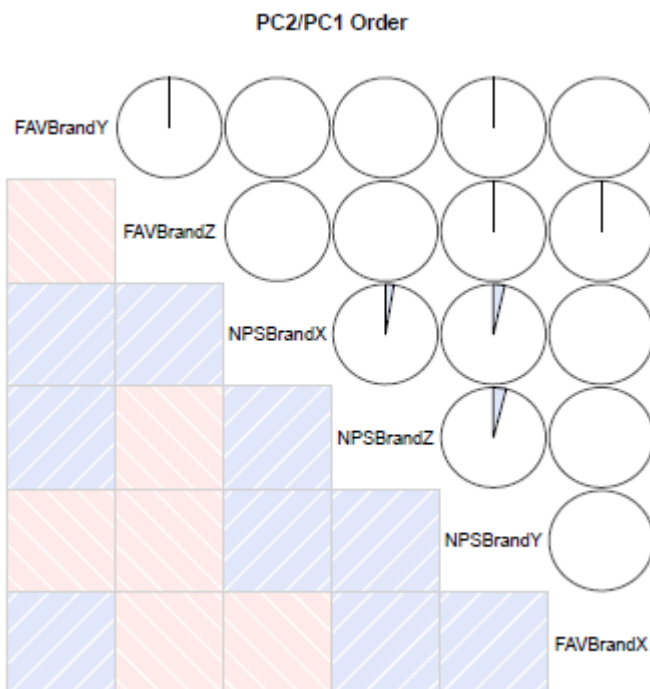
	A	B	C	D	E	F	G
1		NPSBrandX	NPSBrandY	NPSBrandZ	FAVBrandX	FAVBrandY	FAVBrandZ
2	NPSBrandX	1	0.035	0.026	-0.008	0.005	0.007
3	NPSBrandY	0.035	1	0.039	0.001	-0.013	-0.02
4	NPSBrandZ	0.026	0.039	1	0.006	0.004	-0.004
5	FAVBrandX	-0.008	0.001	0.006	1	0	-0.023
6	FAVBrandY	0.005	-0.013	0.004	0	1	-0.011
7	FAVBrandZ	0.007	-0.02	-0.004	-0.023	-0.011	1
8							

Correlation Frequencies Probabilities ⊕ ⋮ ⏪

And a series of visualisations are displayed in a PDF:



The box plot shows min, max, median and the 25% and 75% percentiles. Various corrgrams follow. The first is



You would not want all these - coment out unrequired.

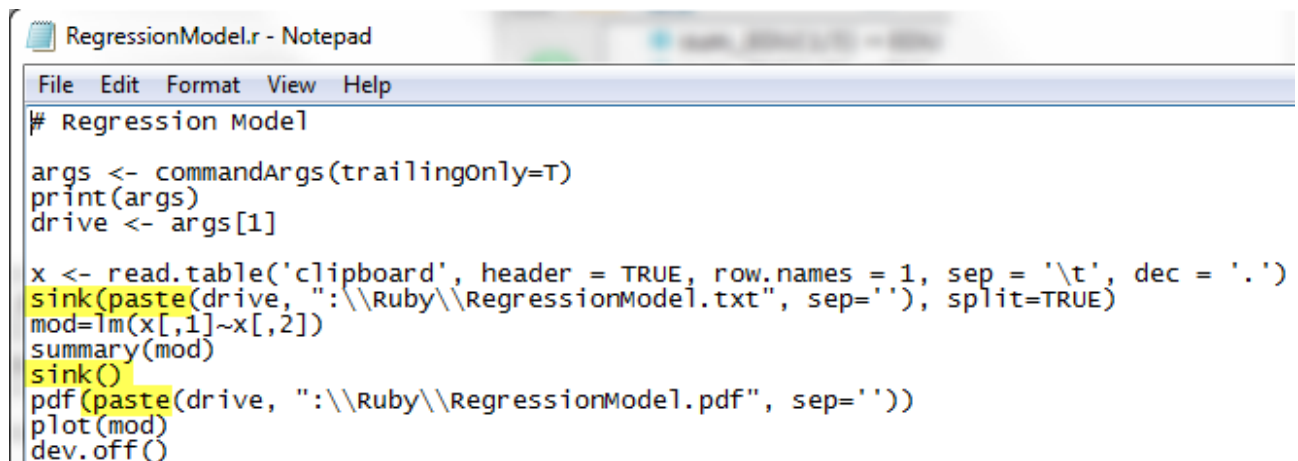
Regression Model

For a regression model, we need a table with an independent variable on the left, and the dependent variable(s) on the right.

- Open the supplied table r_RegMod_EDU_INC~Case

The idea is that education drives income, so the first column is EDU.

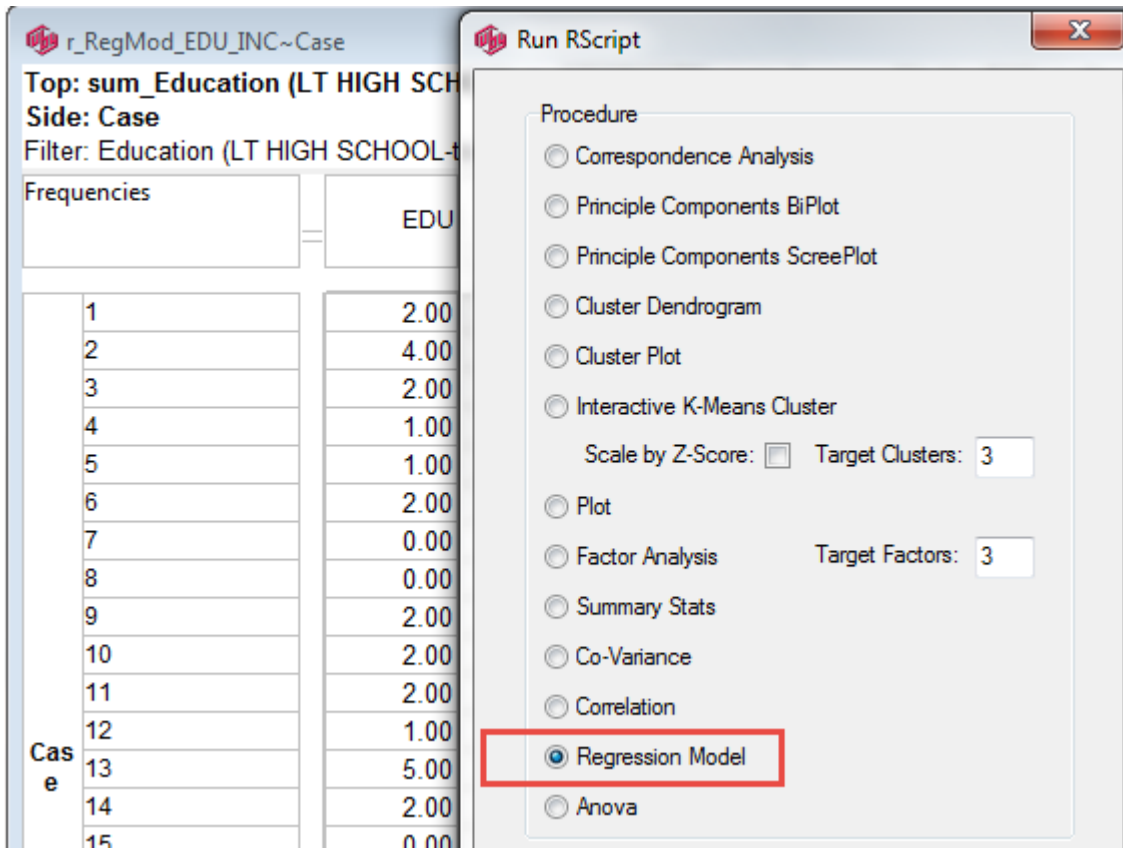
The R script is



```
RegressionModel.r - Notepad
File Edit Format View Help
# Regression Model
args <- commandArgs(trailingOnly=T)
print(args)
drive <- args[1]
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
sink(paste(drive, ":\Ruby\RegressionModel.txt", sep=''), split=TRUE)
mod=lm(x[,1]~x[,2])
summary(mod)
sink()
pdf(paste(drive, ":\Ruby\RegressionModel.pdf", sep=''))
plot(mod)
dev.off()
```

Note the use of sink() to 'sink' the outputs to file, and the use of paste() to concatenate strings. The drive letter is passed as an argument.

Run as



The Command Window output is

```

Administrator: C:\Windows\System32\cmd.exe

[1] "C"

Call:
lm(formula = x[, 1] ~ x[, 2])

Residuals:
    Min       1Q   Median       3Q      Max
-3.0634 -0.4331 -0.4331  0.3568  3.9619

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.433052   0.014789   29.28  <2e-16 ***
x[, 2]       0.605055   0.005172  116.98  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.018 on 9998 degrees of freedom
Multiple R-squared:  0.5778,    Adjusted R-squared:  0.5778
F-statistic: 1.368e+04 on 1 and 9998 DF,  p-value: < 2.2e-16

```

The sunk file comprises the above.

On closing the window, a PDF of various diagnostic charts appears.

Anova

The R script is

```
Anova.r - Notepad
File Edit Format View Help
# Anova
x <- read.table('clipboard', header = TRUE, row.names = 1, sep = '\t', dec = '.')
mod=lm(x[,1]~x[,2])
summary(mod)
anova(mod)
```

Since the parameter for anova() is a linear model, run on the same table as above, as

The screenshot shows the RStudio interface. On the left, a data table is displayed with 15 rows and 2 columns. The first column is labeled 'Case' and the second is 'EDU'. The data values are as follows:

Case	EDU
1	2.00
2	4.00
3	2.00
4	1.00
5	1.00
6	2.00
7	0.00
8	0.00
9	2.00
10	2.00
11	2.00
12	1.00
13	5.00
14	2.00
15	0.00

On the right, the 'Run RScript' dialog box is open, showing a list of statistical procedures. 'Anova' is selected with a radio button. Other options include Correspondence Analysis, Principle Components BiPlot, Principle Components ScreePlot, Cluster Dendrogram, Cluster Plot, Interactive K-Means Cluster (with 'Scale by Z-Score' checkbox and 'Target Clusters' set to 3), Plot, Factor Analysis (with 'Target Factors' set to 3), Summary Stats, Co-Variance, Correlation, and Regression Model.

The output is

```
Call:
lm(formula = x[, 1] ~ x[, 2])

Residuals:
    Min       1Q   Median       3Q      Max
-3.0634 -0.4331 -0.4331  0.3568  3.9619

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
<Intercept>  0.433052   0.014789   29.28  <2e-16 ***
x[, 2]       0.605055   0.005172  116.98  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.018 on 9998 degrees of freedom
Multiple R-squared:  0.5778,    Adjusted R-squared:  0.5778
F-statistic: 1.368e+04 on 1 and 9998 DF,  p-value: < 2.2e-16

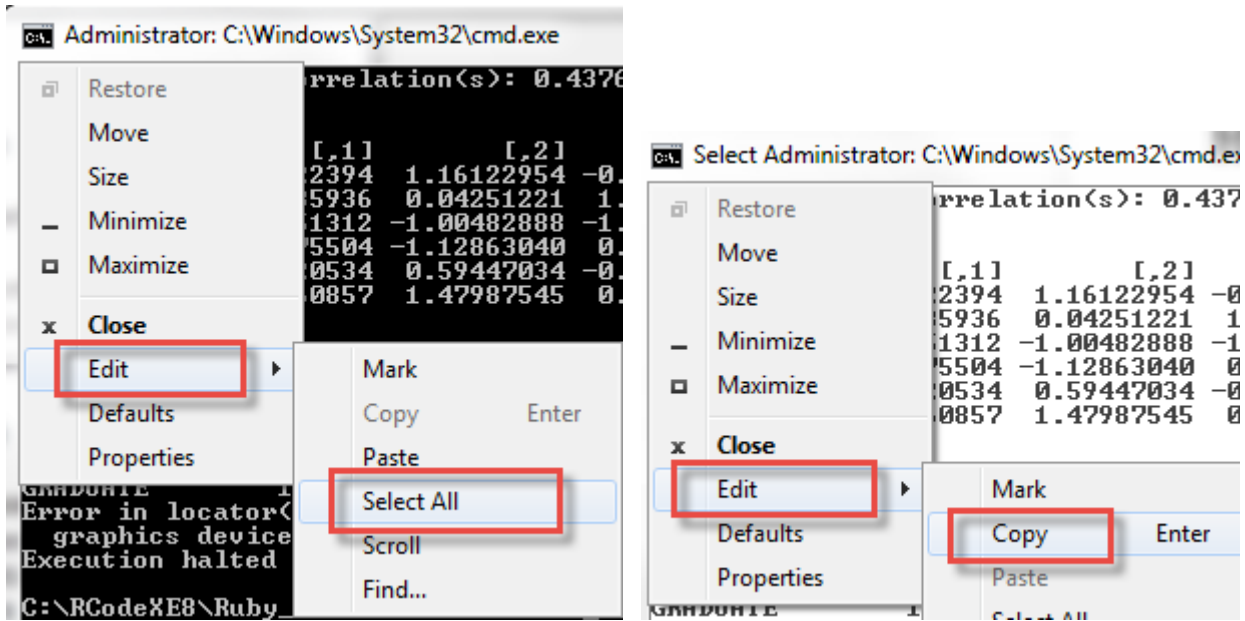
Analysis of Variance Table

Response: x[, 1]
          Df Sum Sq Mean Sq F value    Pr(>F)
x[, 2]     1  14174   14174   13684 < 2.2e-16 ***
Residuals 9998  10356         1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

HANDY HINTS

If you have many windows open, occasionally the Run RScript form can end up behind a window. Bringing Ruby to the front should restore visibility.

The Command window content can be selected and copied from here:



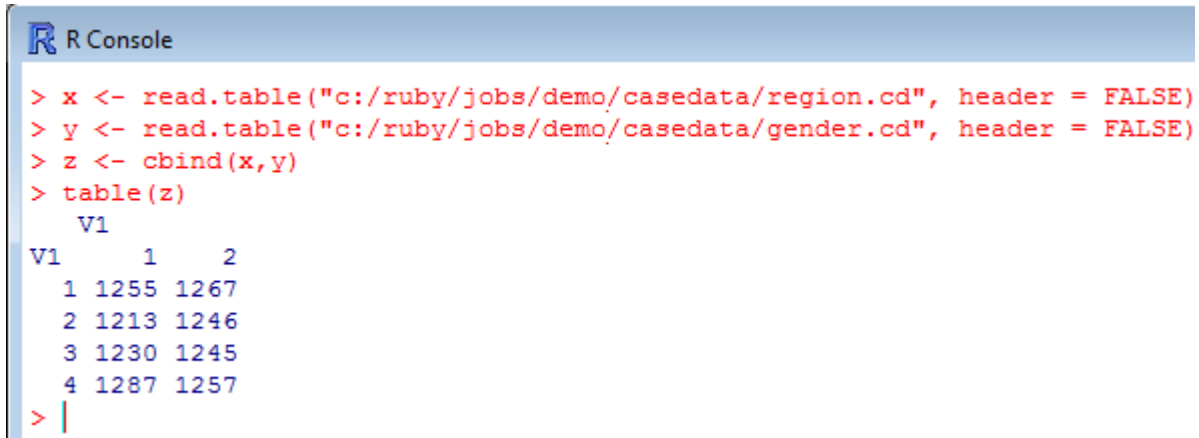
If editing and testing or interacting with an R procedure script (for example, commenting in/out the various plot types), then you can use any editor you like, but remember to save the file before starting Run_RScrips.vbs.

For procedures which write PDFs or other files, make sure you close the output files before running again, or else you will get Windows errors. Save anything you want to keep under your own name.

MISCELLANEOUS

You can read Ruby variable case data directly into R like this:

```
x <- read.table("c:/ruby/jobs/demo/casedata/region.cd", header = FALSE)
y <- read.table("c:/ruby/jobs/demo/casedata/gender.cd", header = FALSE)
z <- cbind(x,y)
table(z)
```



```
R Console
> x <- read.table("c:/ruby/jobs/demo/casedata/region.cd", header = FALSE)
> y <- read.table("c:/ruby/jobs/demo/casedata/gender.cd", header = FALSE)
> z <- cbind(x,y)
> table(z)
  V1
V1  1  2
1 1255 1267
2 1213 1246
3 1230 1245
4 1287 1257
> |
```

The Ruby equivalent is

Top: Gender
Side: Region

Frequencies Corner Net Respondents		Gender	
		Males	Females
Region	NE	1,255	1,267
	SE	1,213	1,246
	SW	1,230	1,245
	NW	1,287	1,257

Uninstall statCONN (if you have it)

The first release of this document relied on statCONN, a third party COM interface to R. Unfortunately, statCONN is now licence-limited, with free usage for non-commercial purposes only.

There were also other problems with the statCONN approach: cumbersome, a difficult installation procedure, slow, hard to edit the COM calls, and so on.

These various issues have been addressed by redesigning the Ruby-R interface to call Rscript.exe directly. The only disadvantage (compared to statCONN) is that the path to Rscript.exe needs to be known up front, and entered manually to the master VBS script.

This document replaces all previous versions.

[end of document]