Introduction to Probability and Statistics Using R/RCommander With IPSUR Plug-in

G. Andy Chang & G. Jay Kerns

February 19, 2010

Timestamp: February 19, 2010

Table of Contents

Introduction

This book is written for learning the R software, with a main focus on the use of R Commander and the IPSUR plug-in. Through the examples in the book, you will be able to also learn some basic skills in probability computations and statistical analysis. This material covers most of the topics in the introductory probability and statistics courses and can be used as a supplementary material for these courses.

The R program was initially developed by Robert Gentleman and Ross Ihaka, and was named after the first name of the two authors. R is an open source program for statistical computing and graphics which is a popular software among statisticians, scientists and engineers. It is part of the GNU project, and its source code is freely available under the GNU General Public License. The current R program is the result of a collaborative effort with contributions from all over the world and it is organized and founded by the R Development Core Team. There are more than 2,000 packages available at the Comprehensive R Archive Network (CRAN) and information about R can be found in CRAN web site: http://www.r-project.org/.

R Commander, developed by John Fox with the contributions from my other R program users and developers, is a platform-independent graphical user interface (GUI) for R. It makes the use of R as easy as SPSS and Minitab. The best part is that it is free. It is an excellent tool for basic statistical analysis and for teaching introductory probability and statistics courses. Many R users and developers have written plug-ins for the R Commander.

IPSUR (Introduction to Probability and Statistics Using R) developed by G. Jay Kerns and G. Andy Chang is an R Commander plugin that includes additional options for graphing, simulations, probability computations and statistical analysis. Jay Kerns has also written an introductory probability and statistics book, Introduction to Probability and Statistics Using R, which can be downloaded for free from the Internet (http://ipsur.r-forge.r-project.org/book/index.php). In this book, readers will learn R programming and using R codes for statistical computing. Instructions for downloading R, R Commander, and the free book written by Dr. Kerns will be given in the next few sections.

The followings are links to R related articles and information for understanding and learning R, R Commander, and R Commander Plug-ins:

- Introduction to R, by Venables, Smith, and The R Development Core Team: http://cran.r-project.org/doc/manuals/R-intro.pdf
- Getting Started with the R Commander, by John Fox: http://socserv.mcmaster.ca/jfox/Misc/Rcmdr/Getting-Started-with-the-Rcmdr.pdf
- Extending the R Commander By "Plug-in" Packages, by John Fox: http://tolstoy.newcastle.edu.au/R/e3/help/att-2806/wrapper.pdf

More information about R can be found in R package itself.

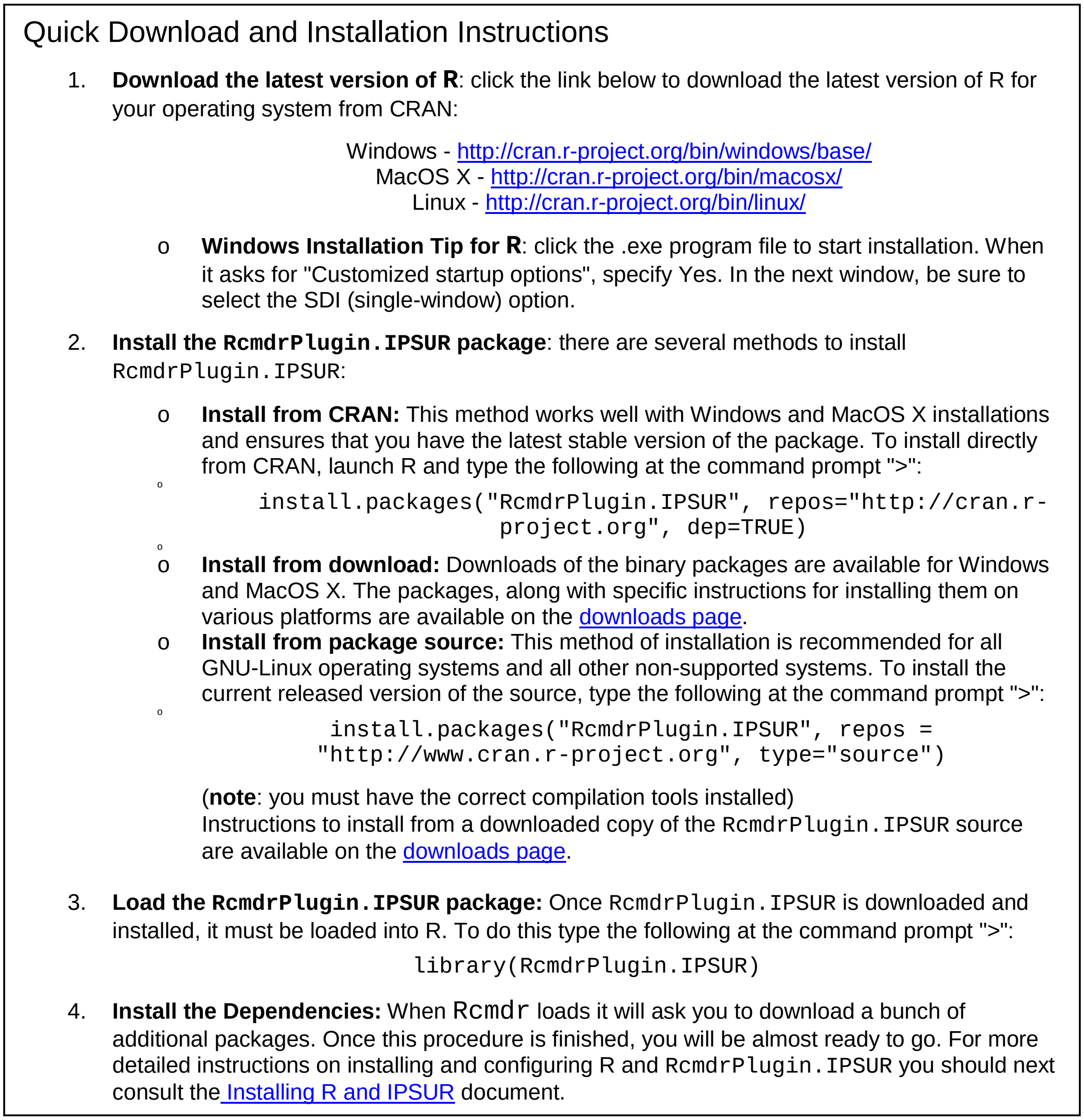
There are four major sections in this book: 1) Download/Installation and The Use of R and R Commander, 2) Descriptive Statistics (showing the use of R Commander for various charts), 3) Probability, and 4) Inferential Statistics.

The Use of R and R Commander

Download and Installation

How To Install R, R Commander, and IPSUR

The instructions for installing R, R Commander, and IPSUR plugin can be found in the following web site: http://ipsur.r-forge.r-project.org/rcmdrplugin/installation.php. An image of this page is Following is an image of the web page. Following the instructions in this page, you will be able to download all three components mentioned above on your computer.



Quick Download and Installation Instructions

1. **Download the latest version of R**: click the link below to download the latest version of R for your operating system from CRAN:

   Windows - http://cran.r-project.org/bin/windows/base/
   MacOS X - http://cran.r-project.org/bin/macosx/
   Linux - http://cran.r-project.org/bin/linux/

   - **Windows Installation Tip for R**: click the .exe program file to start installation. When it asks for "Customized startup options", specify Yes. In the next window, be sure to select the SDI (single-window) option.

2. **Install the `RcmdrPlugin.IPSUR` package**: there are several methods to install `RcmdrPlugin.IPSUR`:

   - **Install from CRAN:** This method works well with Windows and MacOS X installations and ensures that you have the latest stable version of the package. To install directly from CRAN, launch R and type the following at the command prompt ">":

     ```
     install.packages("RcmdrPlugin.IPSUR", repos="http://cran.r-
                        project.org", dep=TRUE)
     ```

   - **Install from download:** Downloads of the binary packages are available for Windows and MacOS X. The packages, along with specific instructions for installing them on various platforms are available on the downloads page.
   - **Install from package source:** This method of installation is recommended for all GNU-Linux operating systems and all other non-supported systems. To install the current released version of the source, type the following at the command prompt ">":

     ```
     install.packages("RcmdrPlugin.IPSUR", repos =
     "http://www.cran.r-project.org", type="source")
     ```

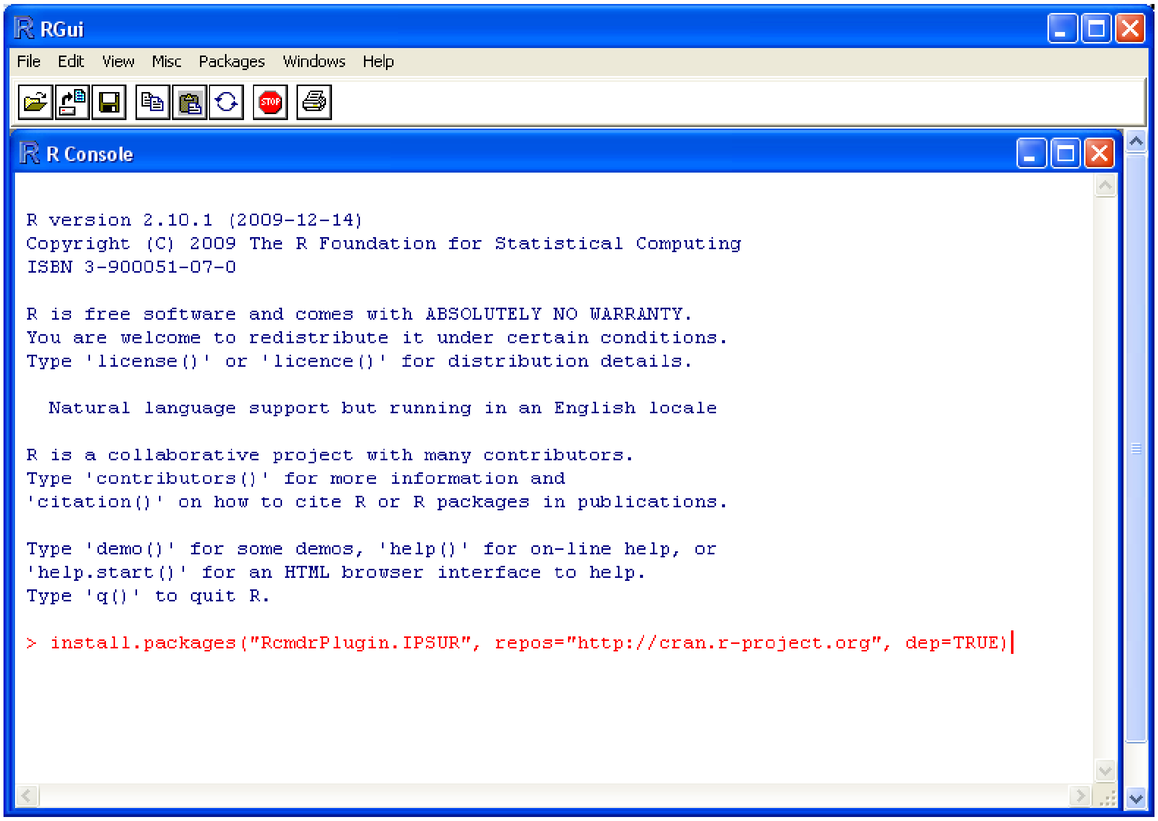
     (**note**: you must have the correct compilation tools installed)
     Instructions to install from a downloaded copy of the `RcmdrPlugin.IPSUR` source are available on the downloads page.

3. **Load the `RcmdrPlugin.IPSUR` package:** Once `RcmdrPlugin.IPSUR` is downloaded and installed, it must be loaded into R. To do this type the following at the command prompt ">":

   ```
   library(RcmdrPlugin.IPSUR)
   ```

4. **Install the Dependencies:** When `Rcmdr` loads it will ask you to download a bunch of additional packages. Once this procedure is finished, you will be almost ready to go. For more detailed instructions on installing and configuring R and `RcmdrPlugin.IPSUR` you should next consult the Installing R and IPSUR document.

In the step 1 above, if you wish to install R under the Microsoft Windows operation system then you will click on Windows - http://cran.r-project.org/bin/windows/base/. You will be asked to download and install R. The file for R 2.10.1 is about 30 MB. After completing R download, you can just click Next through all the steps in Setup Wizard to install R. After you have downloaded and installed R following step 1 of the procedure described above, you can run **R GUI** and then copy and paste the installation statement described in Step 2 in the installation web page into **R Console** window (as in the following figure) and hit Enter key to install IPSUR plugin package. This is for installing IPSUR from CRAN. There will be a message indicating the success of the installation. R Console is the window where the R users can run all the R commands.



```
R version 2.10.1 (2009-12-14)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> install.packages("RcmdrPlugin.IPSUR", repos="http://cran.r-project.org", dep=TRUE)
```

When installing IPSUR, you may see "Warning dependencies …" message that is normal. Installing IPSUR is the part that will take a while. It took more than 30 minutes for my last download of IPSUR plug-in. Be patient. The last thing you will see in R Console window after IPSUR is installed completely is:

```
The downloaded packages are in
C:\Documents and Settings\Andy Chang\Local Settings\Temp\Rtmpz3uiJp\downloaded_packages
>
```

The installation of all three components is complete if you see the message above.

Download IPSUR Free Book

You can visit the following web site for the book to download the Introduction to Probability and Statistics Using R book written by Dr. Jay Kerns. The web address for the book is: http://ipsur.r-forge.r-project.org/book/installation.php. The steps are similar to those in downloading ISPUR. If you have the R installed on your computer, you can use the following R command to load and install the free book, that is, after the R command prompt ">" enter

```
> install.packages("IPSUR")
```

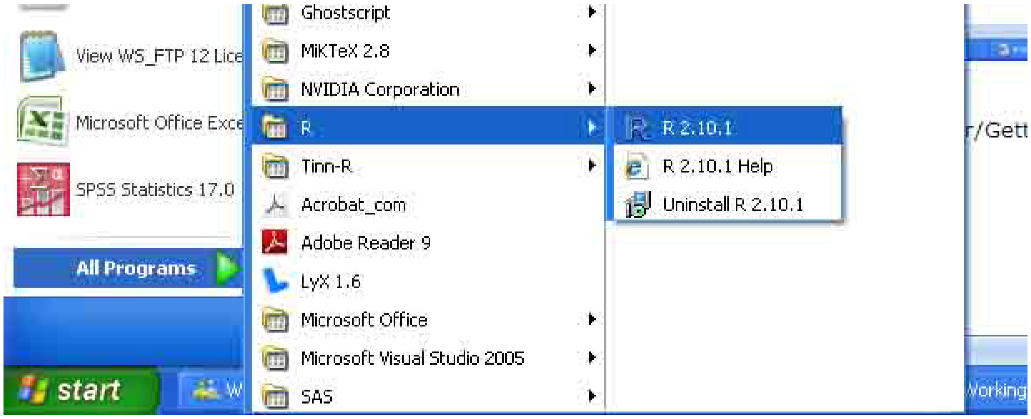
The command above allows you to download and install the book from CRAN. This will give you an updated stable version of the book. There may be problems if you try to download it from R-Forge (the 2nd option on the installation web page) since it is a place that stores the beta or trial version. After installed the book on your computer, you can use of R commands inside the R Console to open the book and view it. They are the following R Commands after ">" command prompt:

```
> library(IPSUR)
> read(IPSUR)
```

The Downloads section of the book website mentioned above also shows you several different methods for downloading PDF version of this book. A PDF file of this free book can be downloaded from the following web site: http://www.lulu.com/items/volume_67/8123000/8123594/2/print/IPSUR.pdf

To Start R Program

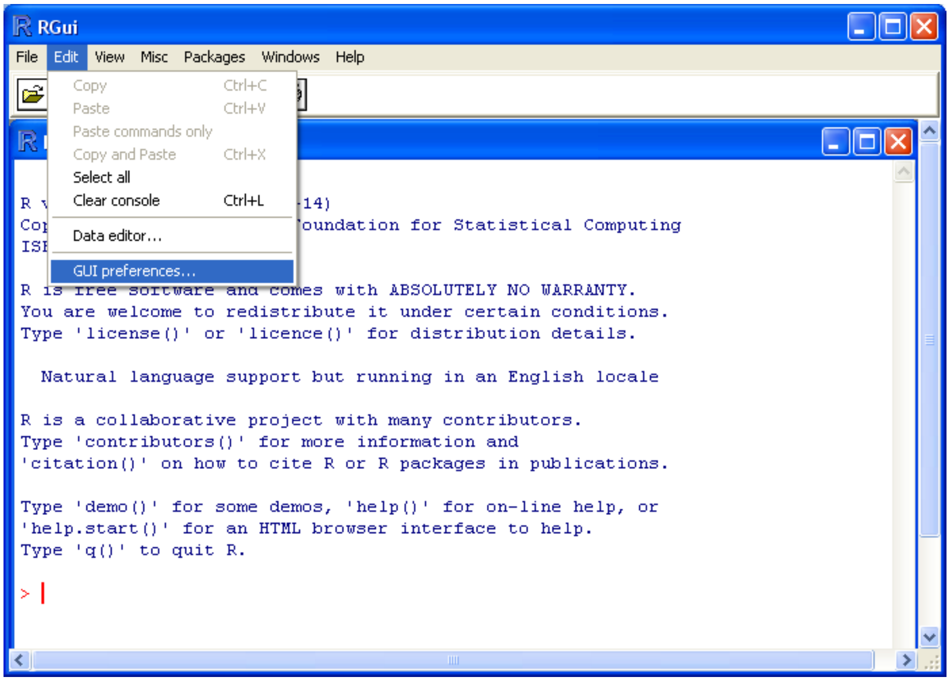
To use R or R Commander with IPSUR plugin, one has to first run the R GUI. This can be done by clicking on Start and select All Programs and Select R with version that one wishes to use, as what is shown in the following figure. This procedure will turn on the R Console window where the R commands can be entered.



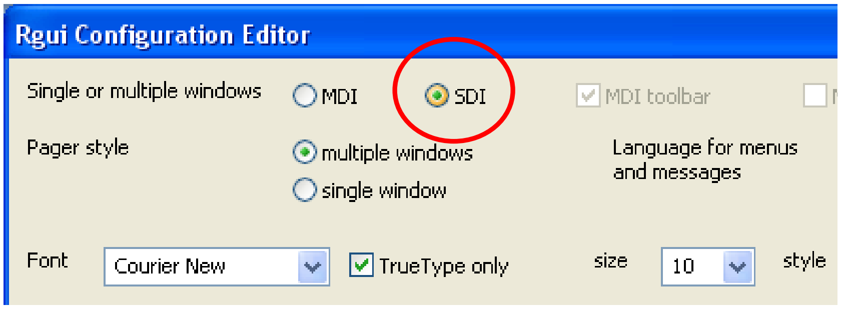
To run **R codes** in R Console, one can just type the actual code after the command prompt ">", and hit enter key. The use of R codes and R Commander with IPSUR will be explained in later sections.

Setting Rgui Configuration to SDI

After you ran R GUI, the R Console window will appear on the computer screen. If this is your first time running R GUI and you did not select the SDI (single-document interface) option during the installation, you can do it now, by clicking the mouse pointer on Edit that is on the R Console menu bar, and select GUI Preferences option… . And then, in the Rgui Configuation Editor window check SDI box (see the following two figures) and click on Save to save the new setting. Setting up the Rgui Configuration Editor preferences to SDI is not a necessary step but it is an option that allows user to run R windows independently rather than being contained in a master window as in the default "multiple-document interface" (MDI). It will be helpful when one wants to run R Commander or Tinn-R (an R editor).



```
R v                           -14)
Cop                           oundation for Statistical Computing
ISI

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```



Rgui Configuration Editor

Single or multiple windows ○ MDI ● SDI ☑ MDI toolbar □

Pager style ● multiple windows ○ single window

Language for menus and messages

Font Courier New ☑ TrueType only size 10 style

After it is set to SDI, you will be seeing an R Console window instead of RGui window next time you run Rgui.

To Run R Commands from R GUI

User of R can enter R commands in the R console for many purposes from simple calculations and graphing to statistical modeling and analysis.

The following are examples for simple calculations. To run an R command in R Console, one can simply enter it after the command prompt, “>”.

For a simple calculation such as adding 3 and 5, it can be done by typing 3 + 5, and R will return the answer, 8, as shown blow.

```
> 3 + 5  # you typed in
[1] 8    # the answer
> 3 * 5  # you typed in
[1] 15   # the answer
> 3 / 5
[1] 0.6
> 3 - 5
[1] -2
```

You may type # to include a comment. The text after # will not be executed.

To store a set of numbers can be done by using “=” or “->” or “<-“.

```
>  x = 7  # store value 7 in x
>  x <- 6 # store value 6 in x
>  6 -> x # another way to store value 6 in x
```

The x above is a variable name. The acceptable variable name may consist of letters, numbers, period “.”, underscore “_” characters. For example, x1, x.1, x_1, x.r, x_r.

One may **create a data vector** using either c() or scan() functions. For example, you can do the following to create a data vector using the name x with values 1, 3, 5, 7, in it. The scan() function is only good for storing numeric values.

```
>  x = c(1,3,5,7)  # store a vector of values 1, 3, 5, 7 in x
>  x               # type the variable name to view values in x
[1] 1 3 5 7
>  y = scan()      # enter this, then enter values one at a time
1:  1
2:  3
3:  5
4:  7
5:
Read 4 items
> y                # type the variable name to view values in y
[1] 1 3 5 7
```

The c() function can be used for storing character data with the use of single or double quotes.

```
> z = c("Mon", "Tue", "Wed", "Thu", "Fri") # For character data
> z
[1] "Mon" "Tue" "Wed" "Thu" "Fri"
```

R functions can be applied to the data vector. The followings are some basic functions in R.

```
> y             # view values in y
[1] 1 3 5 7
> mean(y)       # find the mean of values in y
[1] 4
> sd(y)         # find standard deviation for values in y
[1] 2.581989
> mean(y)/sd(y) # mean of y divided by standard deviation of y
[1] 1.549193
```

More Examples for R commands

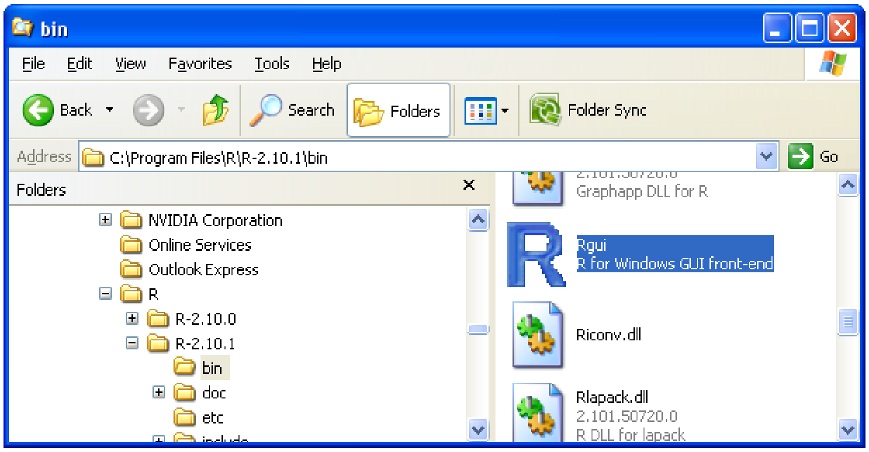
To gain a little bit more experience on R functions for statistics using R Console, you can try the following examples for producing basic descriptive statistics using R commands. Although, the main focus of this book is for learning the use of R Commander, knowing some basic R commands using R Console will help later on.

```
> test = scan()   # Enter discrete data such as 1,1,2,1,4,4,3,5,6,4,5,4

# For one discrete variable
> table(test)     # Produce frequency table


# Make bar chart
> barplot(test)

# Make relative frequency bar chart with labels
> barplot(table(test),xlab="Sample", ylab="Frequency")

> barplot(table(test)/length(test), xlab="Sample", ylab="Relative
Frequency")

# Make pie chart
> pie(table(test))

> pie(table(test), col=gray(c(.3, .5, .7, .9)))
```

```
# For one continuous variable
> stem(test)       # Make stemplot

> hist(test)       # Make histogram
> hist(test,breaks=3) # Make histogram with 3 intervals specified
> hist(test, prob=TRUE)  # Make relative frequency histogram

> lines(density(test))   # Fit the data with curve

> boxplot(test)          # Make boxplot

> quantile(test, 0.9)    # Find 90th percentile
> quantile(test, .1)     # Find 10th percentile

> sd(test)               # Find sample standard deviation
> mean(test)             # Find mean
> IQR(test)              # Find Interquartile Range

# Make dotplot
> stripchart(test,method="stack")
> stripchart(test,method="stack",pch=1,offset=1,cex=2)

# For bivariate categorical data
> rbind(c(20,30),c(5,45))   # Create row data for contingency table
> cbind(c(20,5),c(30,45))   # Create column data for contingency table

> x = matrix(c(20,5,30,45),nrow=2) # Use of matrix for entering data
> x

> rownames(x) = c("Smoker","Non-smoker")  # Name the row variable
> colnames(x) = c("Cancer","No Cancer")   # Name the column variable
> x

# Make cluster bar chart
> barplot(x,xlab="Smoking", main="Smoking and Lung Cancer",beside=TRUE)

> prop.table(x)           # Make relative frequency distribution table

> barplot(prop.table(x), xlab="Smoking", main="Smoking and Lung
Cancer",legend.text=TRUE,beside=TRUE)  # Cluster bar chart with legend
```

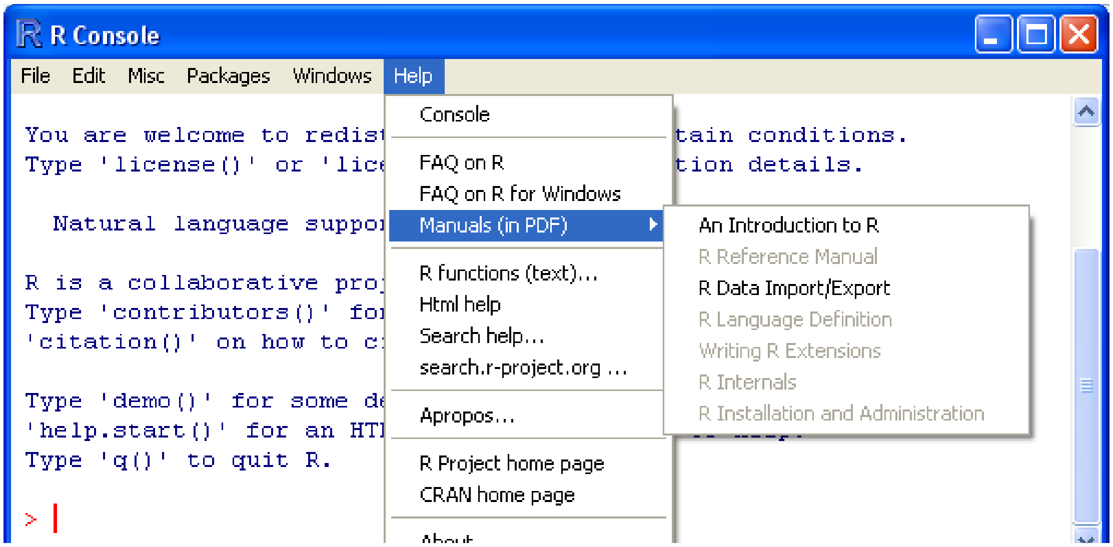
To Run R From A Portable Memory Device

The **Rgui** file in the **bin** folder inside the R folder (see the figure below) where the R program is installed is the program that runs (or say turns on) the R console. So, user can go to this folder and double-click the Rgui file to run and start the R Console. Using the same idea, after installed the R program with IPSUR, user can copy the whole folder into a flash drive, hard-drive, or CD, and connect the device to any computer that accept this device. To run the R program, just run Rgui file in the bin folder.



To Get Help About Using R

There are **manuals**, **help functions** and **search** engine for finding information about R and R functions. There are manuals and search engine downloaded along with the R package. One can click **Help** on the R Console Menu bar and select one of the options in the drop-down list as shown in the following figure to get help. The following figure showed that there are two manuals, Introduction to R and R Data Import/Export, downloaded along with the R package since they are of darker color. If you wish to have more manuals in your R package, you can check them all when downloading R. (This description of Help is for R 2.10.1 with new help settings.)



```
You are welcome to redist          tain conditions.
Type 'license()' or 'lice          tion details.

  Natural language suppo

R is a collaborative proj
Type 'contributors()' fo
'citation()' on how to ci

Type 'demo()' for some de
'help.start()' for an HTI
Type 'q()' to quit R.

> |
```

As shown in the figure above, you can view some **R manuals** through this Help option. You can also use R functions (text) …, Html help, Search help…, and search.r-project.org for searching information for the use of R. If you enter the function **help.start()** in R Console, it will take you to the R Manuals and Reference page which provides you similar information as show in the Help option above. The **R Search Engine** page allows you to use keywords, object name, and others to help you searching the related materials for R.

The Use of help() or RSiteSearch() in R Console

The **help()** function allows user to search local documentation for a R function through R Console. For instance, if you wish to know how to use mean(), then enter **help("mean")** in R Console. R will use a web browser to open the document about the function mean(). One can also enter **help()** in R Console to get a document to learn how to use help() function for search information about the use of R. If user wishes to search on the Internet through R web site for the description of a R function then **RSiteSearch()** function should be used. So, for finding information about mean() function, the user should enter RSiteSearch("mean") in R Console.

To Start R Commander and Process Data Files

This section describes how to start R Commander with IPSUR plug-in and how to create and process the data files.

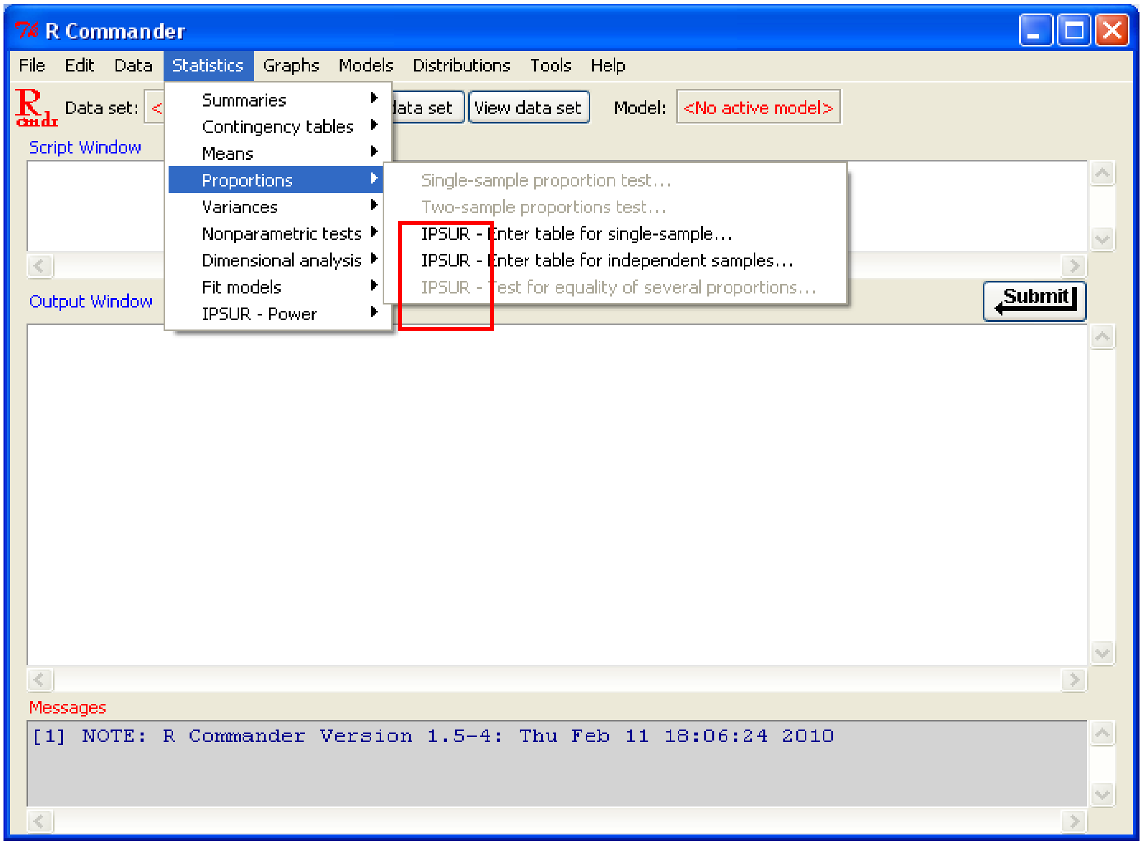
The Use of R Commander With IPSUR Plug-in

In previous section, you learned how to open the R GUI, or say R Console.

To **run R Commander with IPSUR plugin**, user needs to enter the following after the R Console command prompt:

```
>    library(RcmdrPlugin.IPSUR)
```

You must use this command for using this book since all the examples in this book are based on R Commander with IPSUR Plug-in. After you entered this command, a R Commander window will appear and it looks very similar to R Commander window except that there will be more other options to select for using IPSUR functions. All IPSUR functions will have an "IPSUR" right next to the IPSUR functions listed on the drop-down menu. You can see them in the following figure.



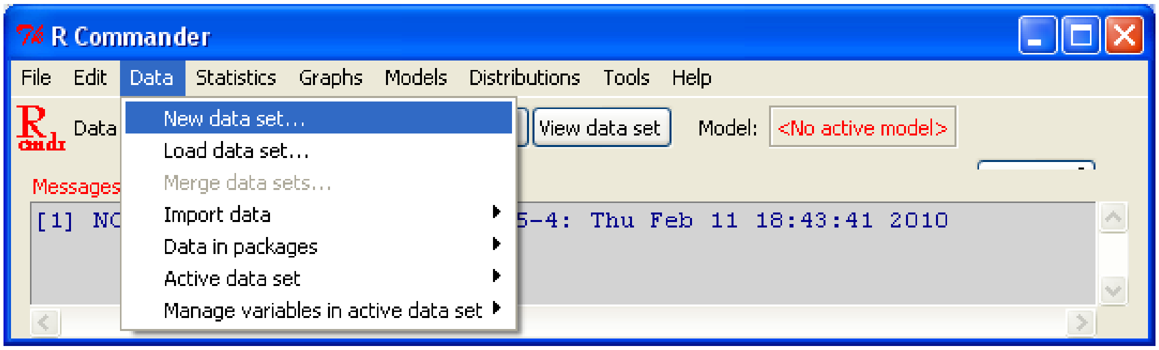
```
[1] NOTE: R Commander Version 1.5-4: Thu Feb 11 18:06:24 2010
```

If you wish to run the R Commander without IPSUR plug-in, you need to enter the R command: `library(Rcmdr)` in R Console. The general principle in using either R Commander with or without IPSUR is the same. So, if you have experience with R Commander then the way to use the R Commander with IPSUR is exactly the same.

On the top part of the R Commander window, there is a menu bar that allows user to select different tools for performing different tasks with the software just like other window software. There is a Script Window that shows the actual R command script used when certain tasks are executed by R Commander. User can modify the script and highlight few lines of script and click the Submit button to run the selected lines of R commands. The results from using the R Commander menu or from the submitted script will be shown in the Output Window. The materials in the Script Window and the Output Window are all plain text and user can copy and paste them.

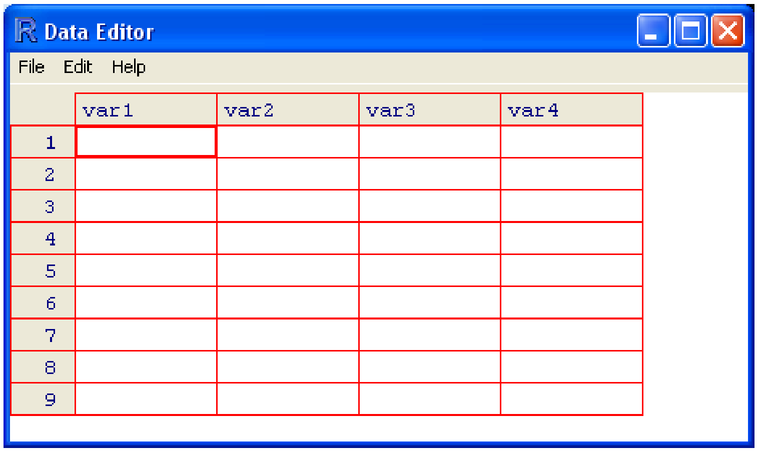
Create Data File

To create a new data file, click on Data on the menu bar, and select New Data Set … option. A New Data Set window will appear on the screen for user to name the data set.



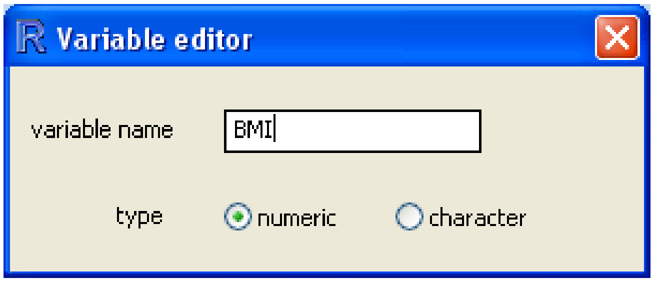
```
[1] NC
5-4: Thu Feb 11 18:43:41 2010
```

After naming the data set, a Data Editor will show on the screen with default variable names var1, var2, … . These names can be changed.



	var1	var2	var3	var4
1				
2				
3				
4				
5				
6				
7				
8				
9				

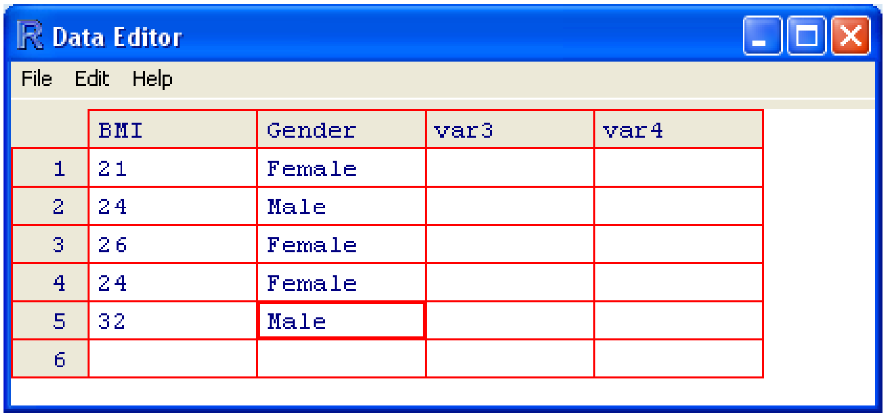
To change the variable name, simply just click on the variable name to be changed and a window for naming the variable will appear. User can specify the data type to be numeric or character. In the following figure, the name var1 was changed to BMI in the Variable editor window, and the type remained as numeric.



variable name: BMI

type: ● numeric ○ character

The following figure shows that the second variable is for gender. The actual gender data were entered. User may enter 1 or 0 and then label them later.



	BMI	Gender	var3	var4
1	21	Female		
2	24	Male		
3	26	Female		
4	24	Female		
5	32	Male		
6				

User may click on the red ☒ icon on the right-hand top corner to close the editor. The data set is ready at this time. User can view it any time by clicking on Edit data set or View data set button in the R Commander window to open the editor for editing data or to view the data.

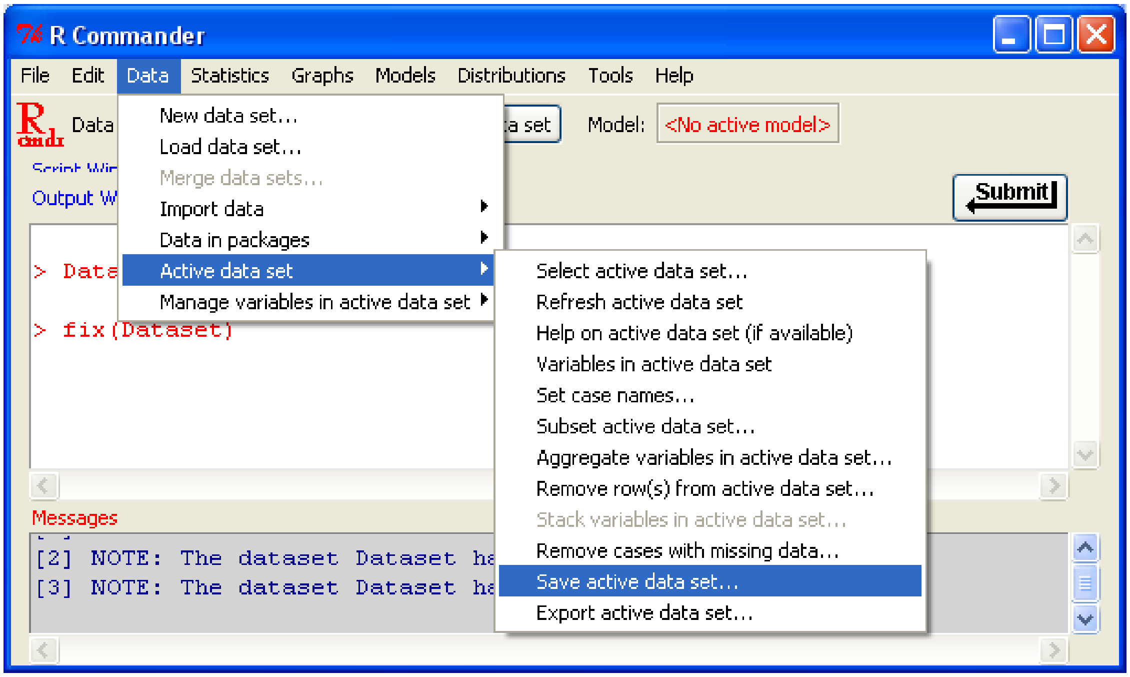
Save Data File

To save this data as an R data file, one can click on Data, and select **Active data set** option in the R Commander window, and then choose **Save active data set …** option to save the data set is a file. That is, in shorthand, (see figure below)

Data / Active data set/ Save active data set …

The file name will have a file extension **.rda**. The data file can be exported to different format such as **.txt**, **.dat**, and **.csv**. This can be done by (in the R Commander window)

Data / Active data set/ Export active data set …

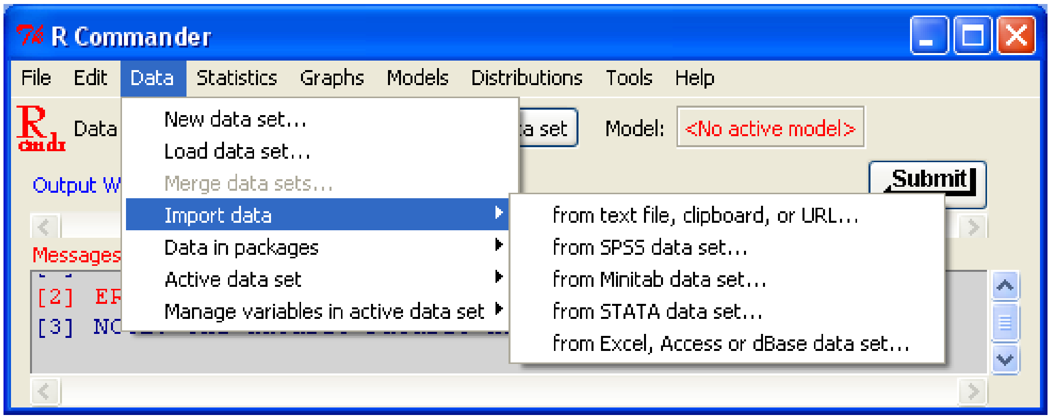

R Commander
File
Edit
Data
Statistics
Graphs
Models
Distributions
Tools
Help
Data
a set
Model:
<No active model>
Script Wir
Output W
Submit
New data set...
Load data set...
Merge data sets...
Import data
Data in packages
Active data set
Manage variables in active data set
Select active data set...
Refresh active data set
Help on active data set (if available)
Variables in active data set
Set case names...
Subset active data set...
Aggregate variables in active data set...
Remove row(s) from active data set...
Stack variables in active data set...
Remove cases with missing data...
Save active data set...
Export active data set...
Messages

```
> Data
> fix(Dataset)
```

```
[2] NOTE: The dataset Dataset ha
[3] NOTE: The dataset Dataset ha
```

Import Data File

User can import data in various formats such as text, SPSS, Minitab, STATA, and EXCEL. To do that, click on **Data** in the R Commander window, select **Import Data** and then select the format of the file to be imported.



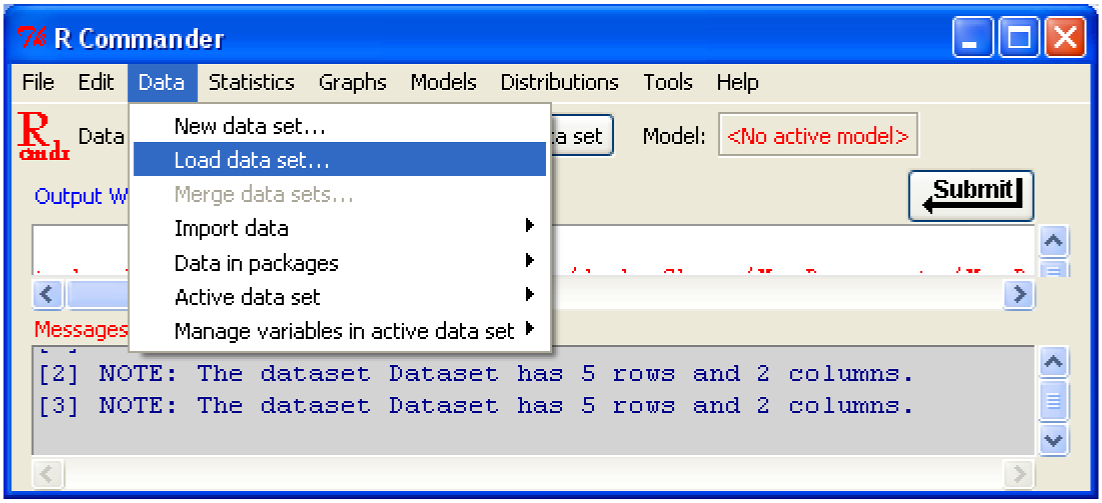
```
[2] ER
[3] NO
```

To learn more about import and export data file, user can review the R Data Import/Export document. It can be found by clicking on **Help** in the R Console window, select **Manuals (in PDF)**, and then select **R Data Import/Export**, in shorthand,

Help / Manuals (in PDF) / R Data Import/Export

To Open An R Data File

If a data is saved as an R data file, then to open the file user needs to click Data in the R Commander window, and select **Load data set …** option, and then in the **Open** window find and select the R data file to be processed.



```
[2] NOTE: The dataset Dataset has 5 rows and 2 columns.
[3] NOTE: The dataset Dataset has 5 rows and 2 columns.
```

There are also data file that come with the packages which users can use for practice the use of the R Commander. To use them, just select **Data in packages** option and then load the data file.

There are other options in R Commander, such as save script, save output, and save R workspace that allow users to save some of their works after the analysis. To perform these tasks, user will need to use the **File** function on the menu of the R Commander window. These files can be saved and retrieve using the R Commander's File function.

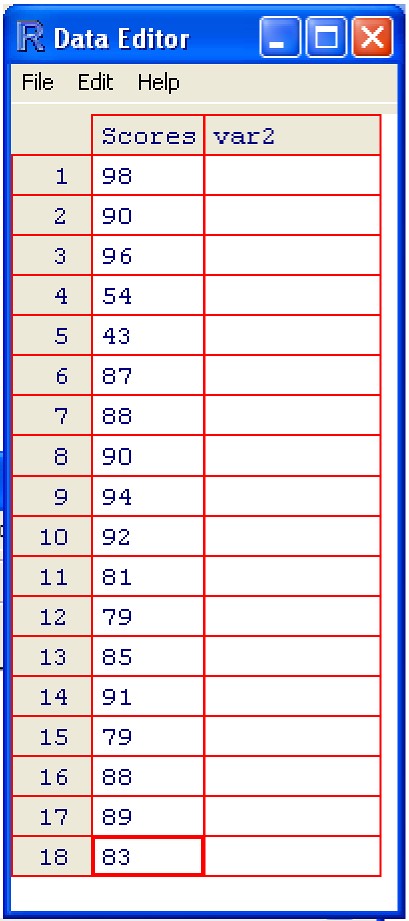
Starting from the next section, the use of R Commander with IPSUR Plug-in for probability and statistics will be explained.

Descriptive Statistics

Exploring One Quantitative Variable: Histogram, Stem Plots, and Strip Charts (Dot Plots)

Example: The scores of an exam were recorded as the followings:
98, 90, 96, 54, 43, 87, 88, 90, 94, 92, 81, 79, 85, 91, 79, 88, 89, 83
How to make histogram, stemplot, and strip chart to describe this data?

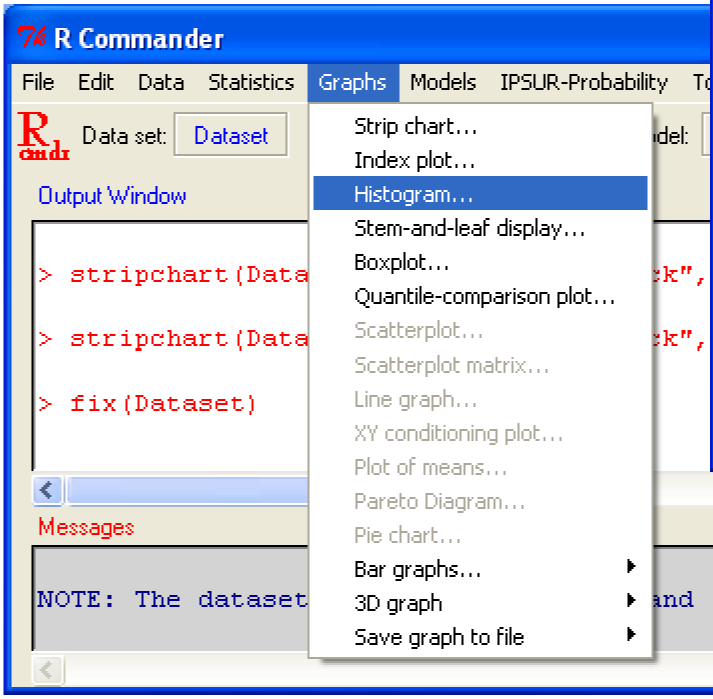
Data Entry:
The data can be entered in the first column of the Data Editor and variable name can be changed to Scores by clicking the heading var1 on top of the first column and change the variable name to Scores. The data would appear as the following.



	Scores	var2
1	98	
2	90	
3	96	
4	54	
5	43	
6	87	
7	88	
8	90	
9	94	
10	92	
11	81	
12	79	
13	85	
14	91	
15	79	
16	88	
17	89	
18	83	

Histogram

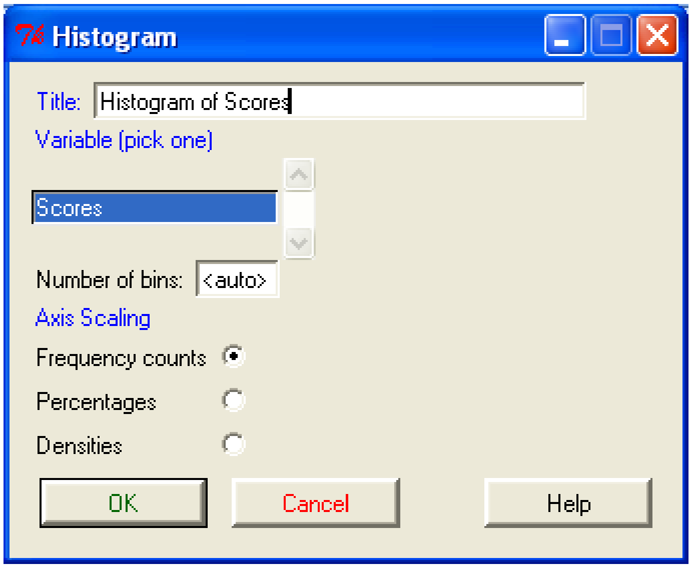
1) In R Commander, click on **Graphs** and select **Histogram…**



```
> stripchart(Data                    k",

> stripchart(Data                    k",

> fix(Dataset)
```

```
NOTE: The dataset                    and 1 columns.
```

2) In the Histogram dialog box enter the chart title in the Title box. One can specify number of bins for the chart or use <auto> for default setting, and then select Scaling. In this example Frequency counts is chosen. And then, click OK.



Title: Histogram of Scores

Variable (pick one)

Scores

Number of bins: <auto>

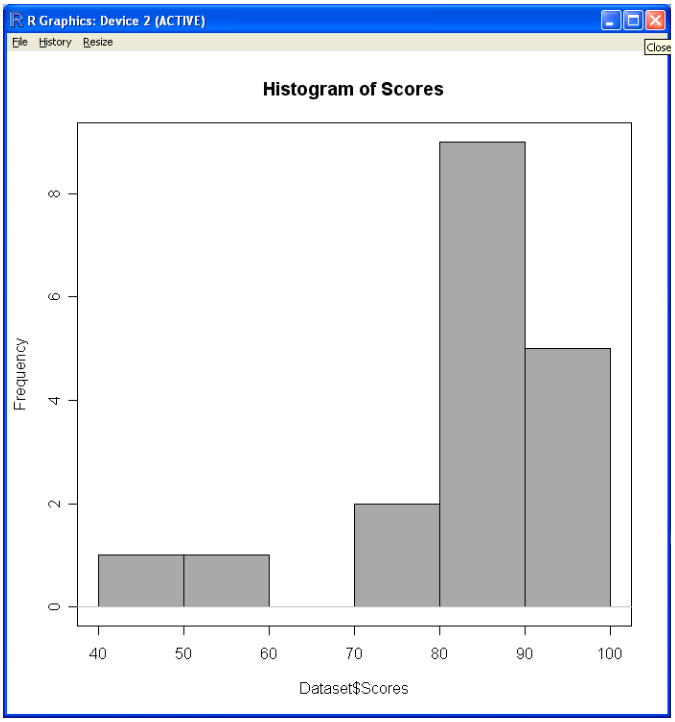
Axis Scaling

Frequency counts ●

Percentages ○

Densities ○

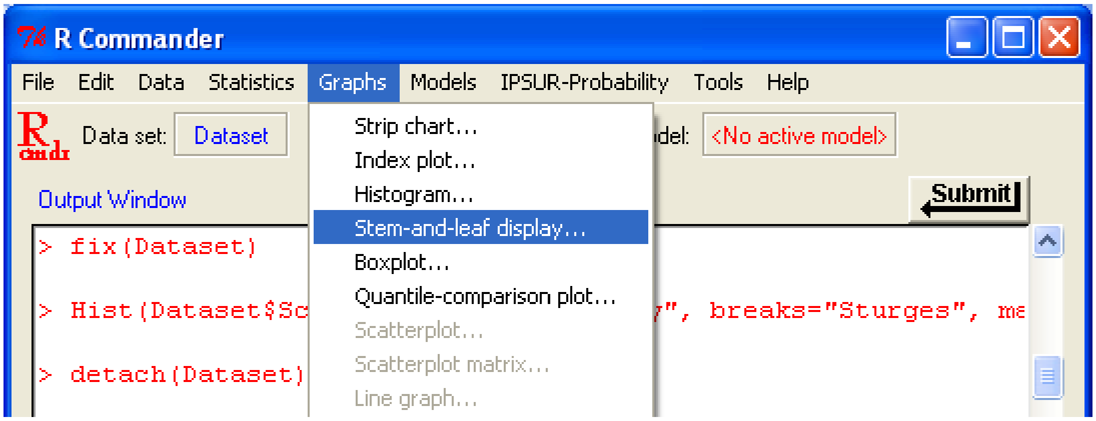
Result:



Label for horizontal axis can be changed to just "Scores" by using the instructions in this document: http://www.cc.ysu.edu/~ghchang/class/s3743/R_ProblemHistogramLabel.pdf

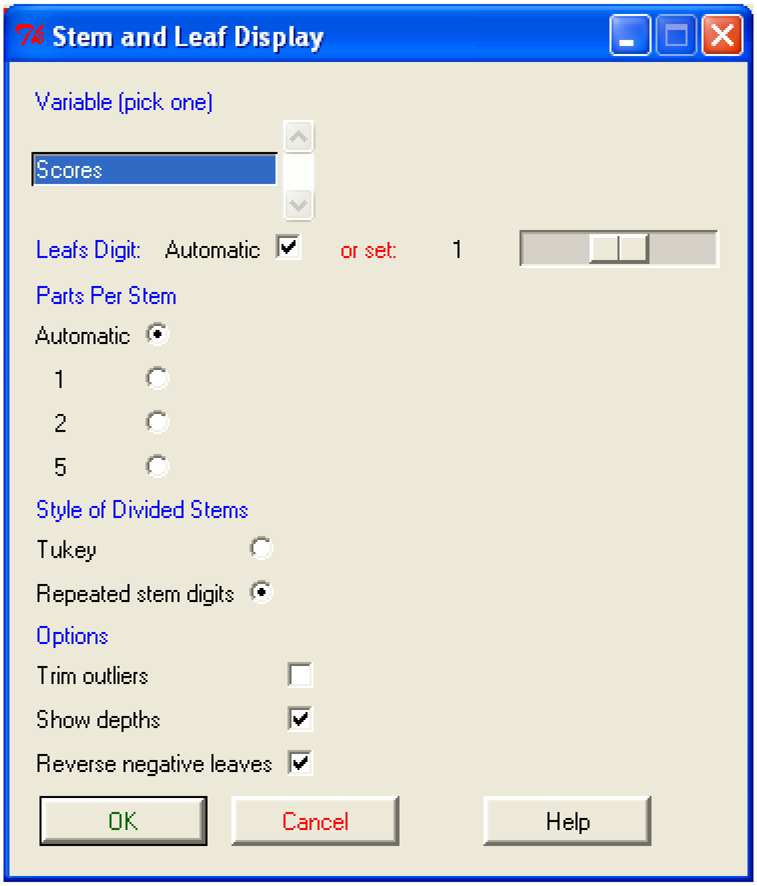
Stemplot

1) In R Commander, click on **Graphs** and select **Stem-and-leaf display …**



```
> fix(Dataset)

> Hist(Dataset$Sc          y", breaks="Sturges", ma

> detach(Dataset)
```

2) In the Stem-and-leaf dialog box, check **Repeated stem digits** and bullet and uncheck **Trim outliers box** as shown in the following figure. Click OK.



Variable (pick one)

Scores

Leafs Digit: Automatic ☑ or set: 1

Parts Per Stem

Automatic ●

1 ○

2 ○

5 ○

Style of Divided Stems

Tukey ○

Repeated stem digits ●

Options

Trim outliers □

Show depths ☑

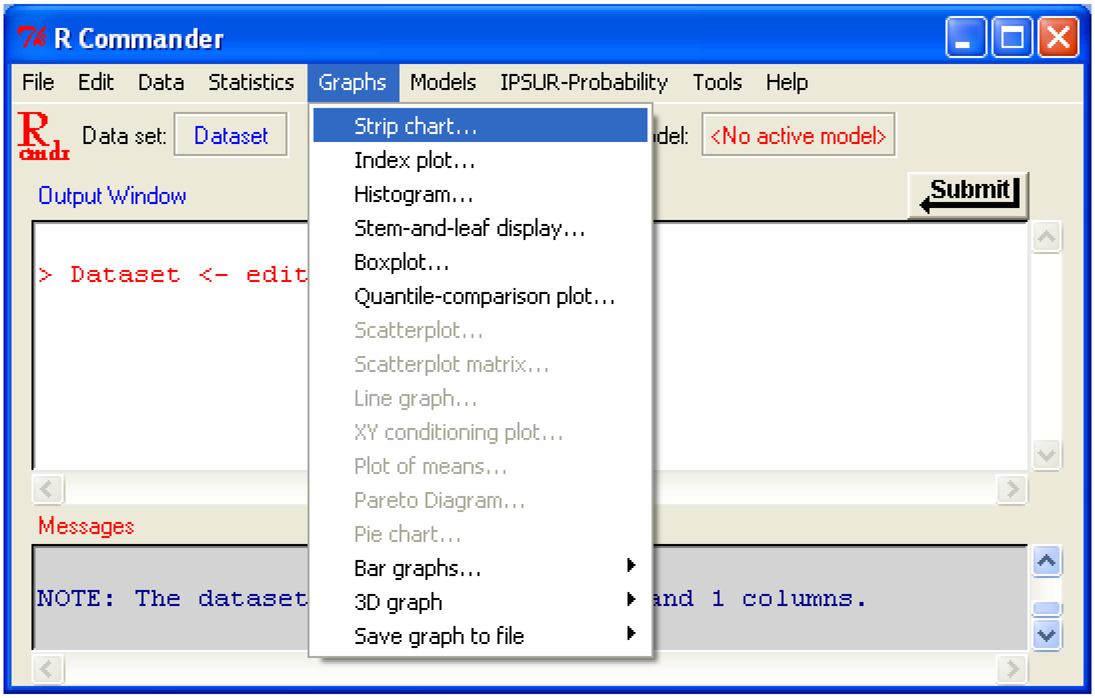
Reverse negative leaves ☑

Result:

```
  1 | 2: represents 12
   leaf unit: 1
           n: 18

      1      4 | 3
             4 |
      2      5 | 4
             5 |
             6 |
             6 |
             7 |
      4      7 | 99
      6      8 | 13
     (5)     8 | 57889
      7      9 | 00124
      2      9 | 68
```

Strip Chart (Dot Plot)

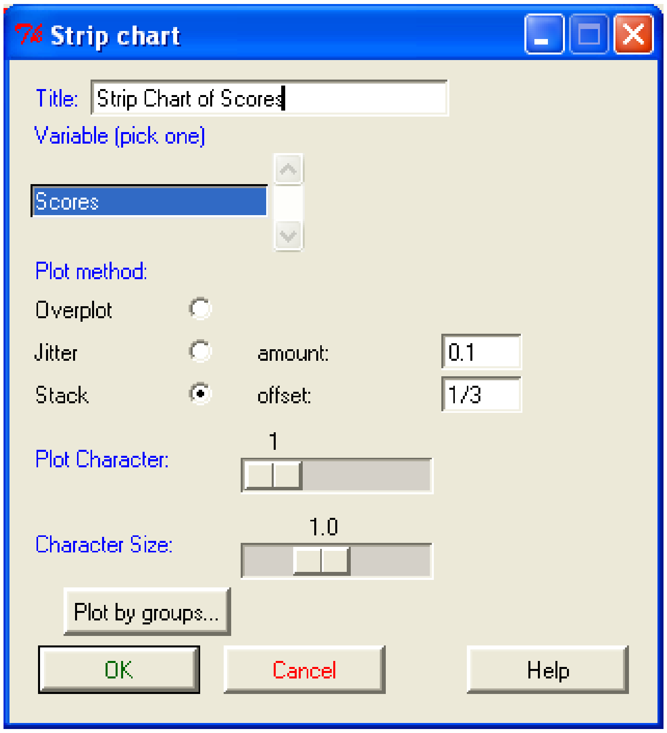
1) Select **Graphs** from menu bar and select **Strip chart …** option.



```
> Dataset <- edit
```

```
NOTE: The dataset                      and 1 columns.
```

3) In the Strip chart dialog window, enter chart title, choose **Stack** bullet, and specify the **Stack offset** for dot plot, and click **OK**.



Title: Strip Chart of Scores

Variable (pick one)

Scores

Plot method:

Overplot ○

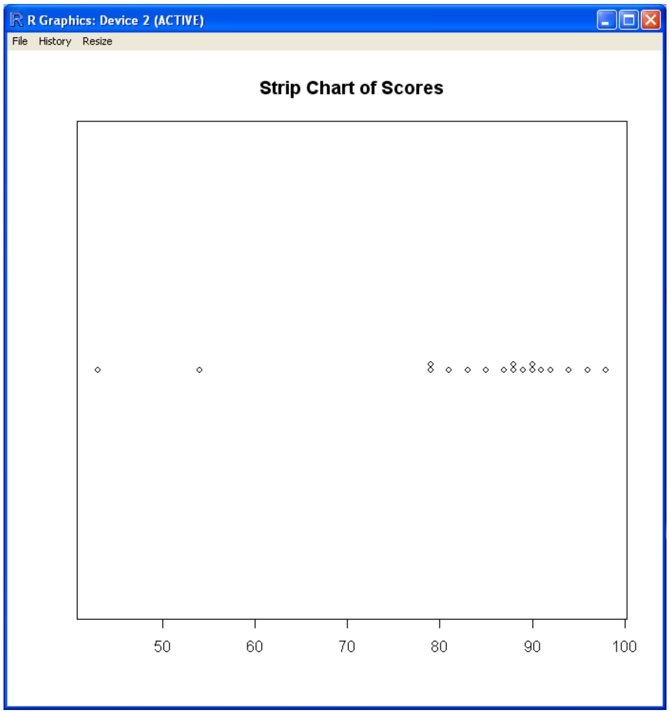
Jitter ○ amount: 0.1

Stack ● offset: 1/3

Plot Character: 1

Character Size: 1.0

Result:



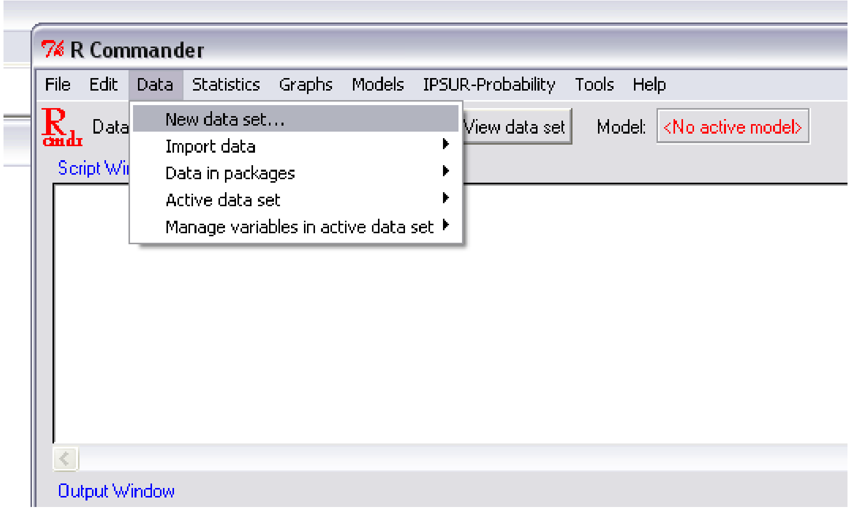
Box Plot and Side-by-Side Box Plot

Example: The data below shows numbers of visits to a website in randomly selected days in two separate months. Create a Box Plot for **Month** 1 as well as a side-by-side box plot for **both months**.

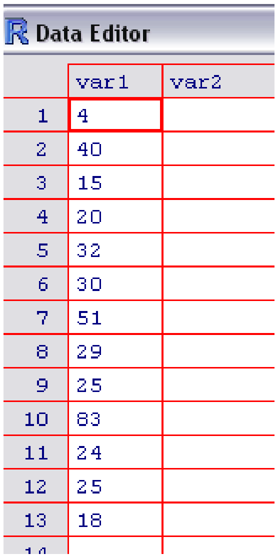
ID	1	2	3	4	5	6	7	8	9	10	11	12	13
Month 1	4	40	15	20	32	30	51	29	25	83	24	25	18
Month 2	7	4	7	55	6	8	9	12	32	5	7	14	9

Boxplot: Exploring One Quantitative Variable

1) From R Commander, click **Data** and select a **New data set…** option.

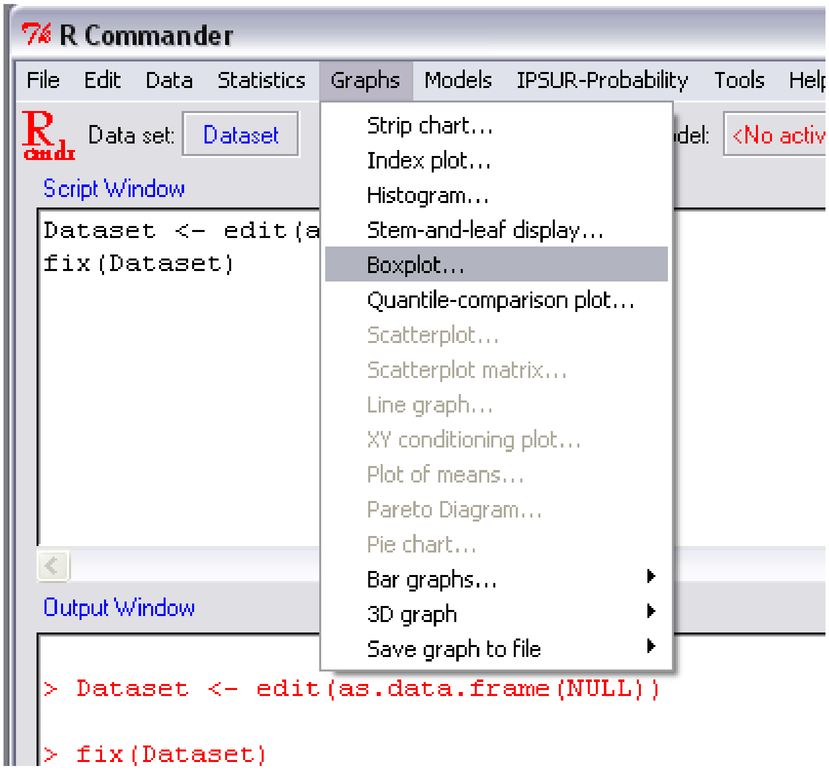


2) Enter Data.


Data Editor

	var1	var2
1	4	
2	40	
3	15	
4	20	
5	32	
6	30	
7	51	
8	29	
9	25	
10	83	
11	24	
12	25	
13	18	
14		

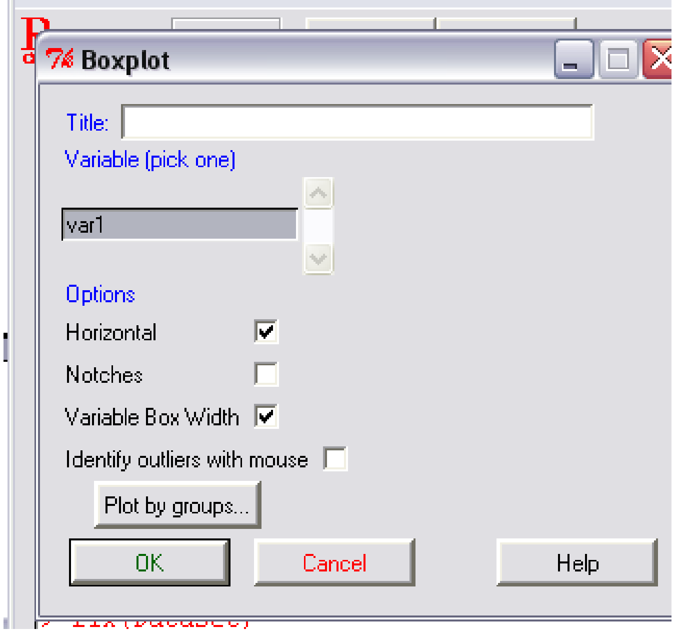
3) Click Graph, Boxplot



```
Dataset <- edit(a
fix(Dataset)
```

```
> Dataset <- edit(as.data.frame(NULL))

> fix(Dataset)
```

4) Accept the default Settings in the dialogue box unless one wishes to do other thing.



Title:

Variable (pick one)

var1

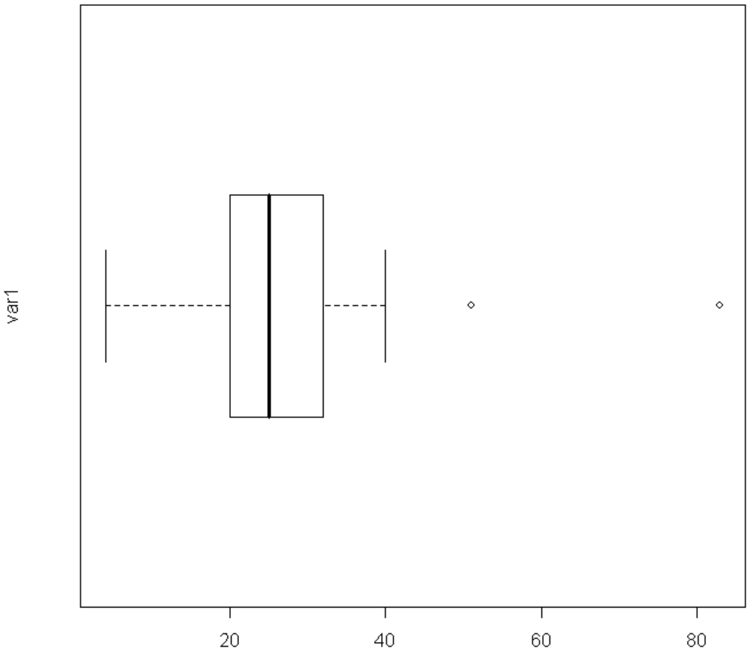
Horizontal ☑

Notches ☐

Variable Box Width ☑

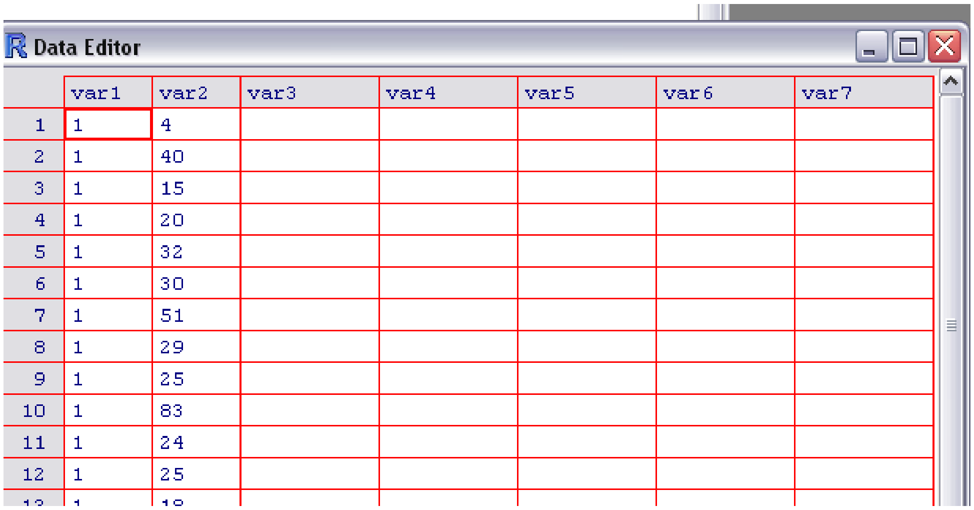
Identify outliers with mouse ☐

5) View Plot.



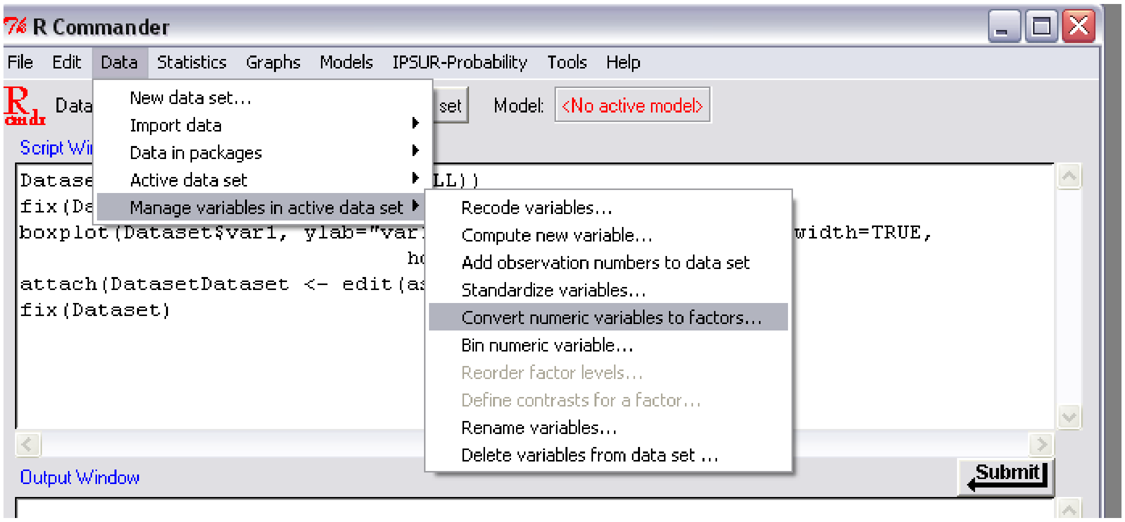
Side by side Box Plots: Comparing Groups on One Quantitative Variable

1) Enter Data in the first column with 1 or 2 in the column denoting month 1 and month 2, enter the a.

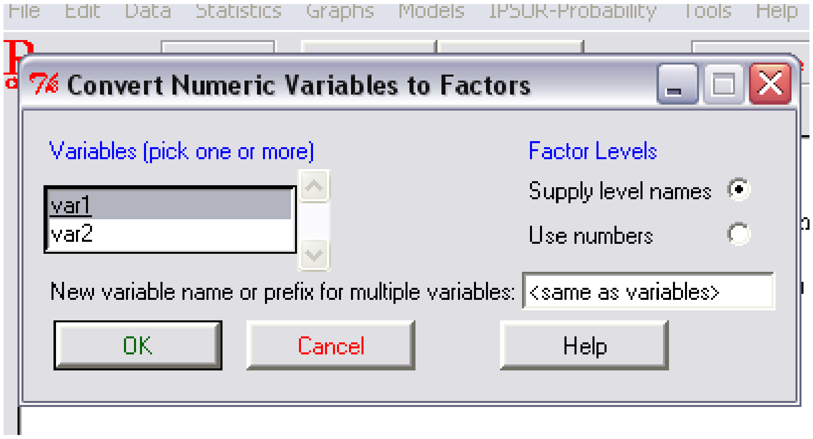


	var1	var2	var3	var4	var5	var6	var7
1	1	4					
2	1	40					
3	1	15					
4	1	20					
5	1	32					
6	1	30					
7	1	51					
8	1	29					
9	1	25					
10	1	83					
11	1	24					
12	1	25					

2) Covert 1 and 2 to factors



```
Datase                                              LL))
fix(Da
boxplot(Dataset$var1, ylab="var1                        width=TRUE,
                                    ho
attach(DatasetDataset <- edit(as
fix(Dataset)
```

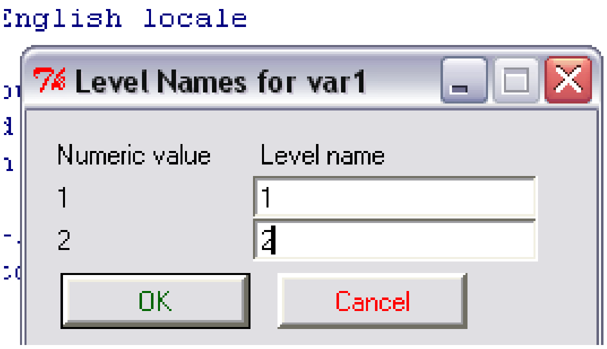

File Edit Data Statistics Graphs Models IPSUR-Probability Tools Help
OK
Cancel
Help

Convert Numeric Variables to Factors

Variables (pick one or more)
var1
var2

Factor Levels
Supply level names ●
Use numbers ○

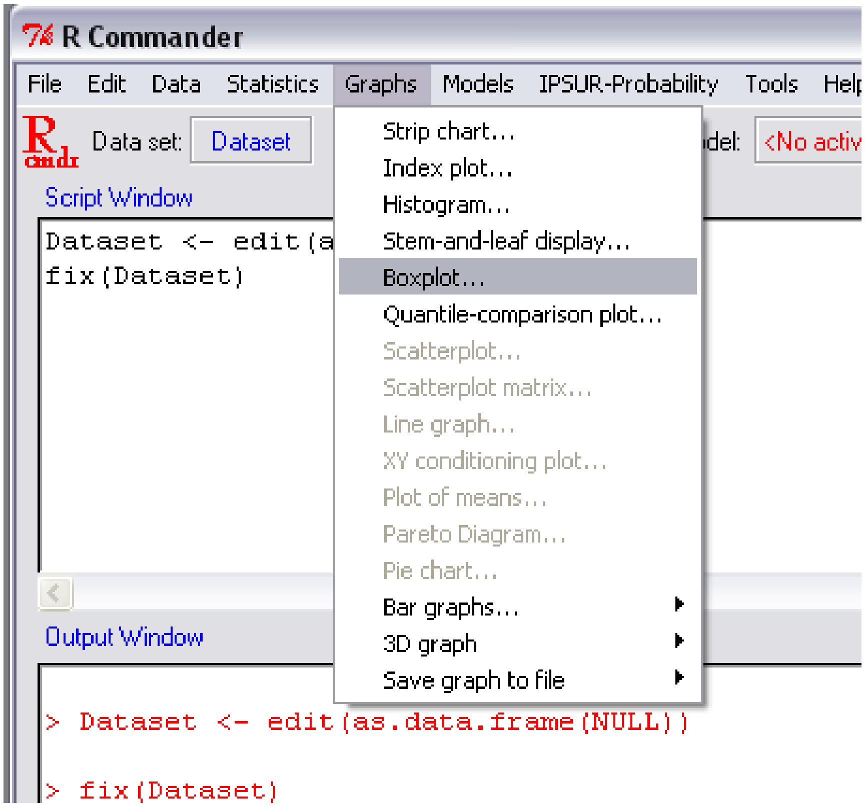
New variable name or prefix for multiple variables: <same as variables>



Level Names for var1

Numeric value	Level name
1	1
2	2

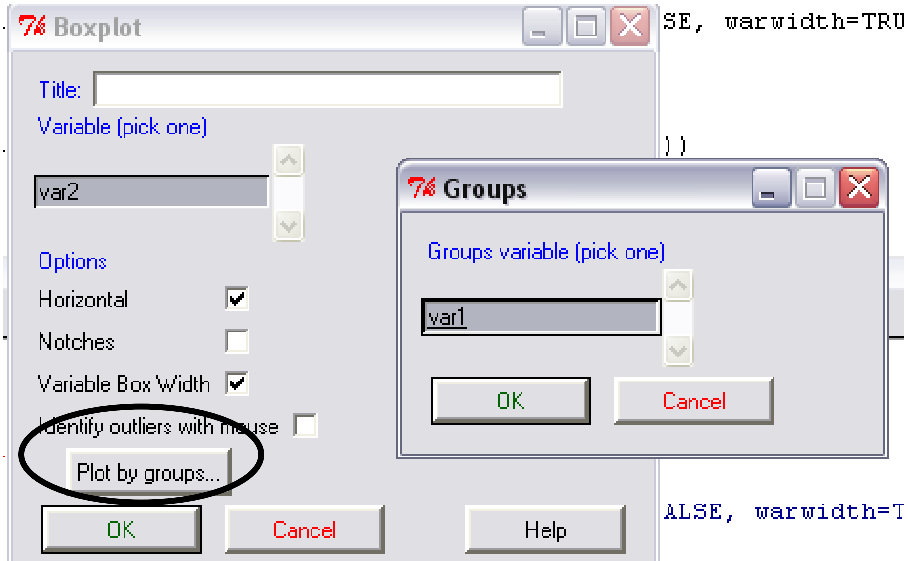
3) Click Graph, Box Plot



```
Dataset <- edit(a
fix(Dataset)
```

```
> Dataset <- edit(as.data.frame(NULL))

> fix(Dataset)
```

4) In Dialogue box select plot by groups and select **var1** and click “ok”, since var1 is group variable. Be sure **Var2** is also selected in the Variable (pick one) box as the Variable to be plotted.



Variable (pick one): var2

☑ Horizontal

□ Notches

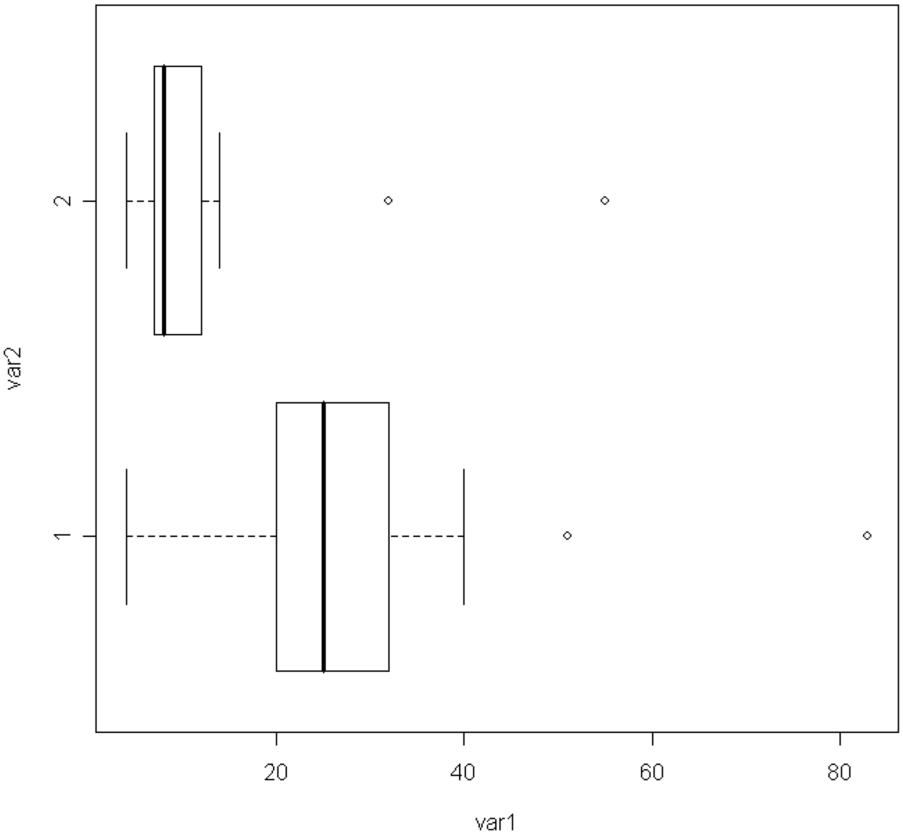
☑ Variable Box Width

□ Identify outliers with mouse

Groups variable (pick one): var1

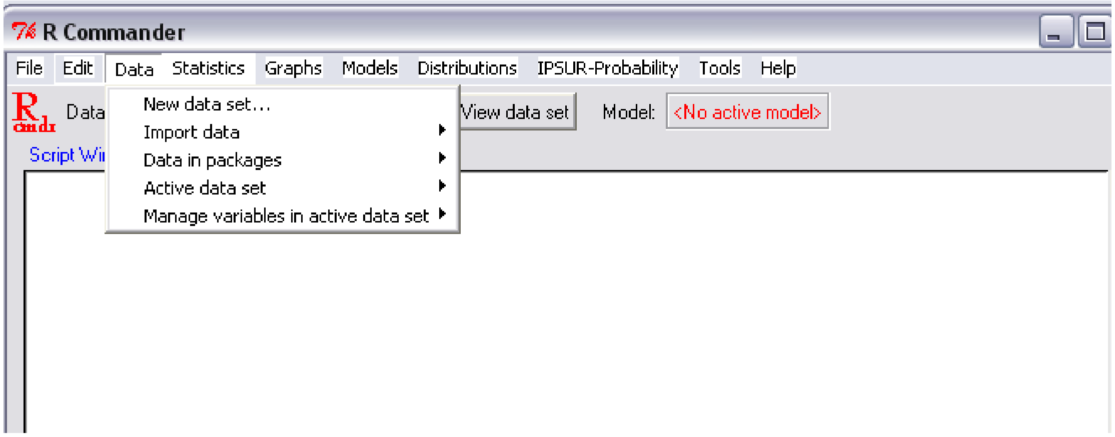
```
SE, varwidth=TRU
))
ALSE, varwidth=T
```

5) View Plot



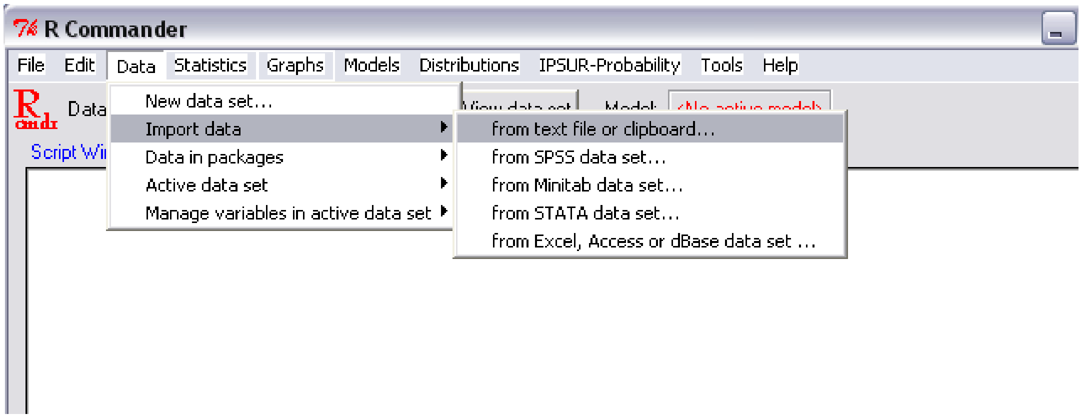
Line Graph

1) From R Commander, click and select **Data.**

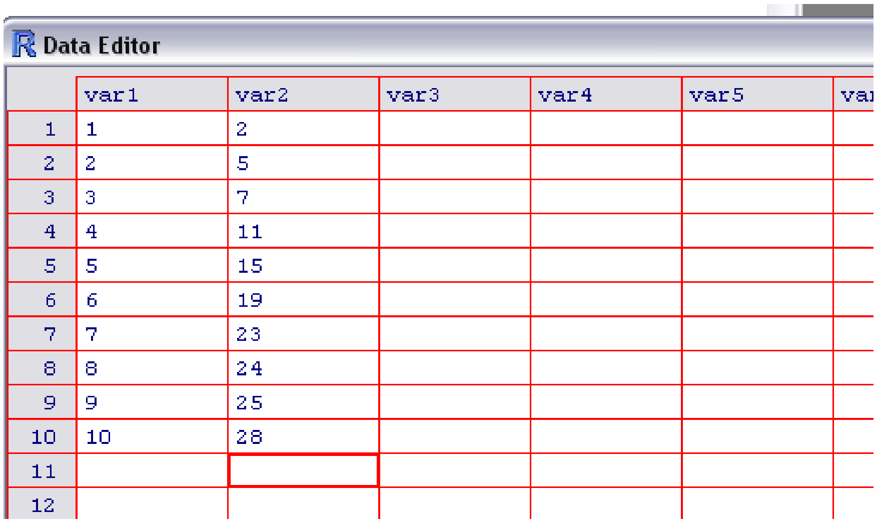


2) From here you can either import data from an existing file, or create a new data set.

   a. To import data from and existing file, simply select **Import data.** Then select the type of file you would like to import (**text file, SPSS, Minitab, STATA, or Excel file**).

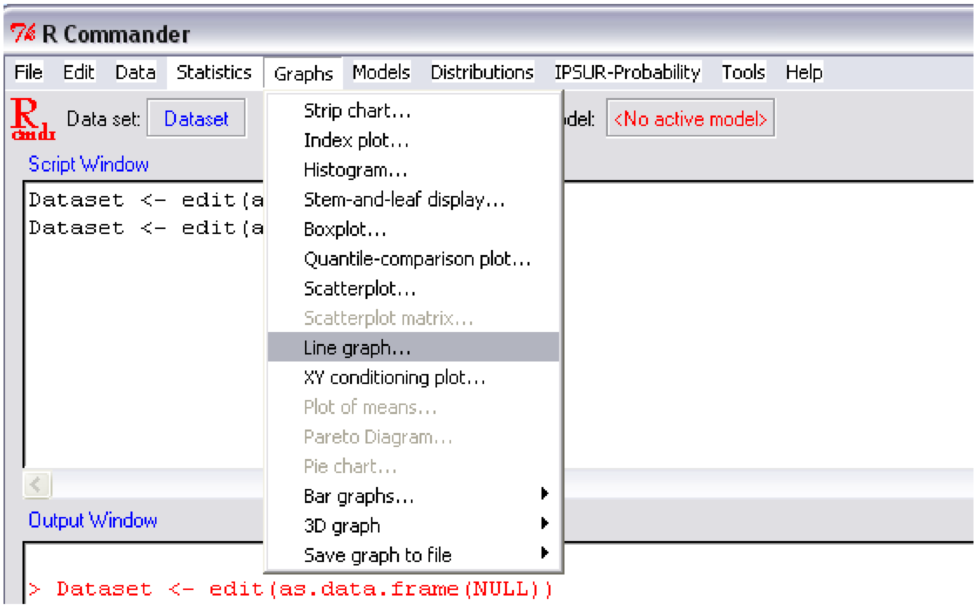


b. To create a new data set, select new data set, and enter your data in two columns that are provided.



	var1	var2	var3	var4	var5	va
1	1	2				
2	2	5				
3	3	7				
4	4	11				
5	5	15				
6	6	19				
7	7	23				
8	8	24				
9	9	25				
10	10	28				
11						
12						

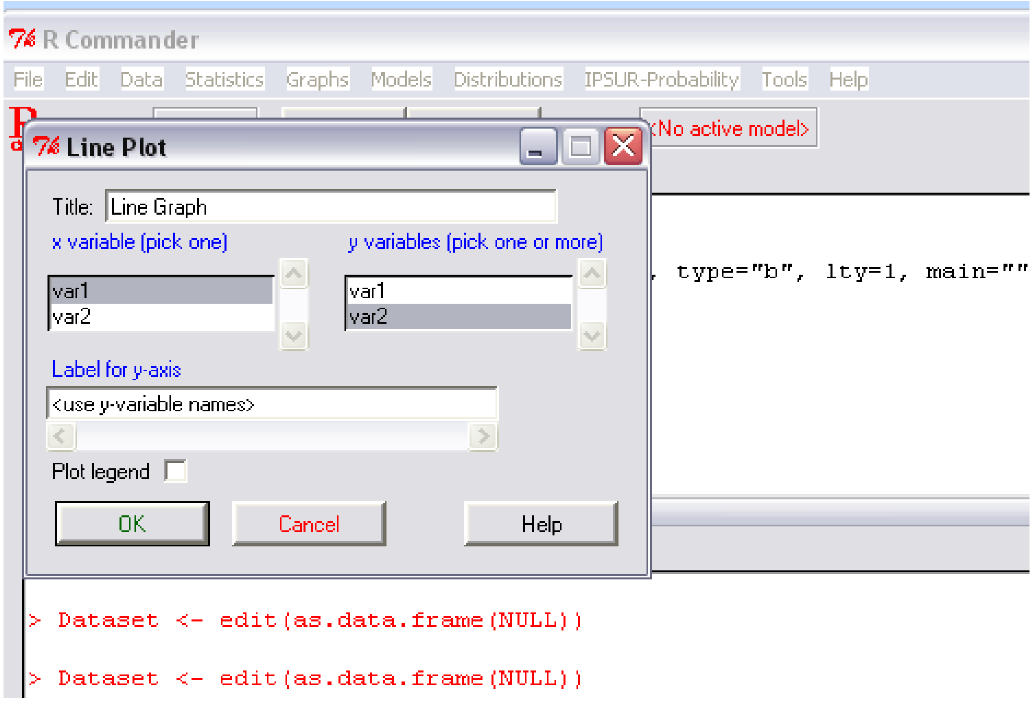
3) From IPSUR, select **Graphs, Line graph**



```
Dataset <- edit(a
Dataset <- edit(a
```

```
> Dataset <- edit(as.data.frame(NULL))
```

4) From this dialog box, select your x and y variable for your line graph, and a title if necessary. Click OK and your line graph will appear. You can select to add a legend if necessary as well in this dialog box.



Title: Line Graph

x variable (pick one): var1 (selected), var2

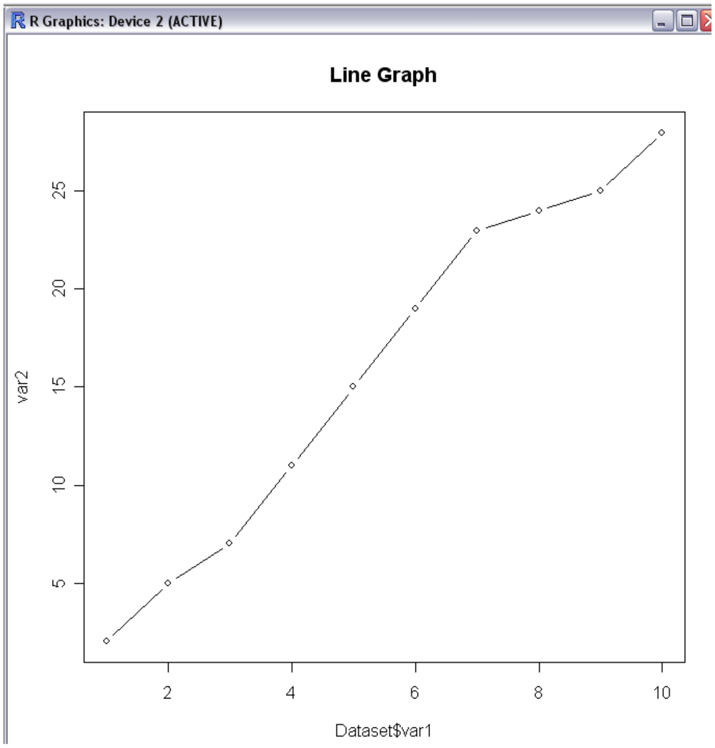
y variables (pick one or more): var1, var2 (selected)

Label for y-axis: <use y-variable names>

Plot legend □

```
, type="b", lty=1, main=""
```

```
> Dataset <- edit(as.data.frame(NULL))

> Dataset <- edit(as.data.frame(NULL))
```



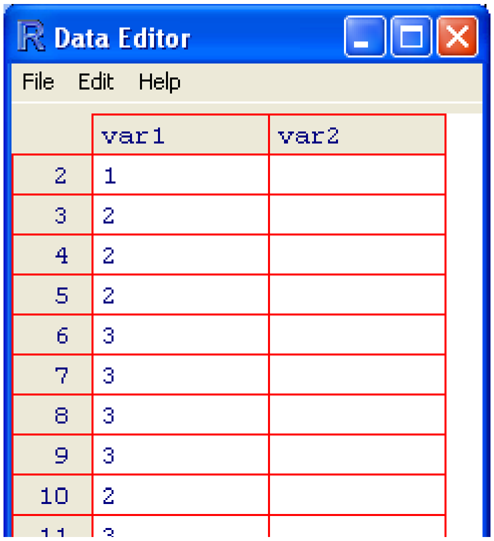
Exploring One Qualitative/Categorical Variable: Pie Chart (or Bar Chart)

When making bar chart or pie chart for a categorical variable, R can make the chart using the raw data from cases of subjects or use the summarized data such as frequencies for all different categories of outcomes for a categorical variable.

Making Pie chart from categorical variable **with raw data.**

1) Enter your data into a new data table for the categorical variable.

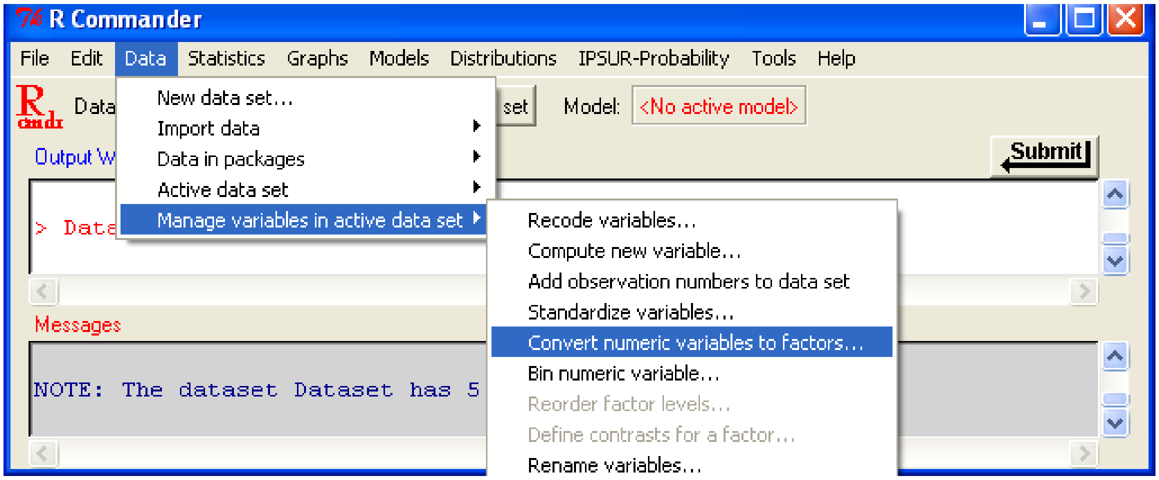
   This data needs to be converted to a factor variable in order to make bar chart or pie chart.



	var1	var2
2	1	
3	2	
4	2	
5	2	
6	3	
7	3	
8	3	
9	3	
10	2	
11	3	

2) Convert your data into factors.
   a. Perform the following menu selections:

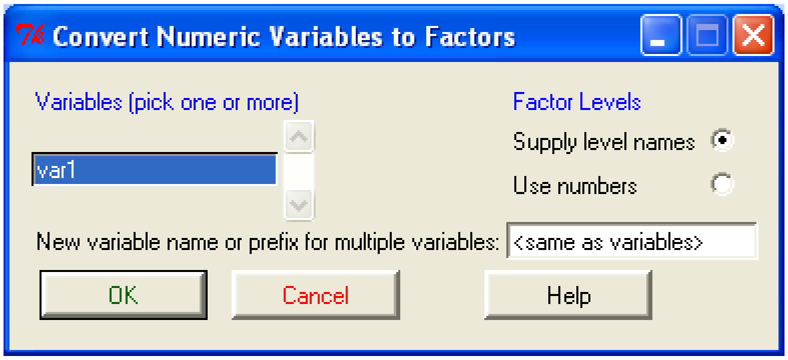
Data/Manage variables in active data set/Convert numeric variables to factors…



```
> Data
```

```
NOTE: The dataset Dataset has 5
```

b. Select the variable to be converted:



Convert Numeric Variables to Factors

Variables (pick one or more): var1

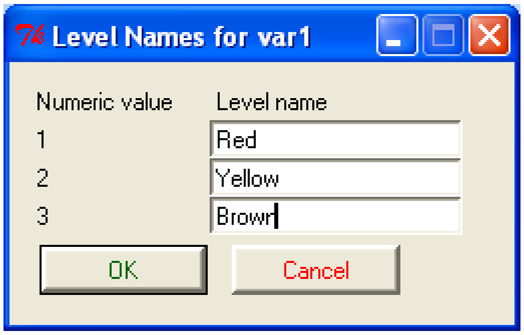
Factor Levels

Supply level names ●

Use numbers ○

New variable name or prefix for multiple variables: <same as variables>

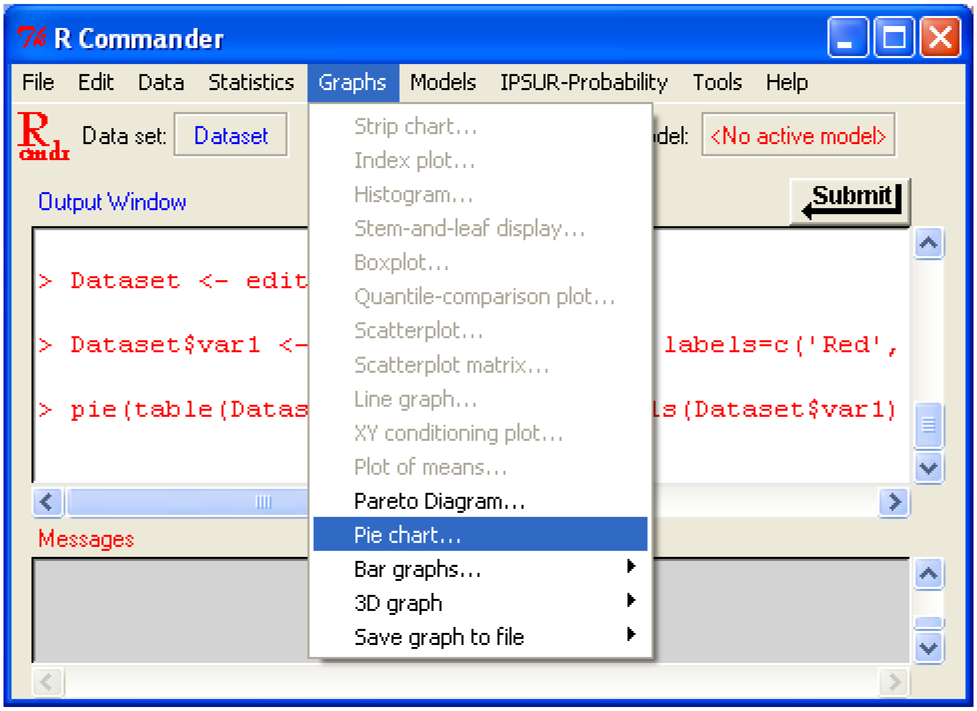
c. If “Supply level names” is selected, one needs to supply the names or labels for all different data values for this categorical variable.



Level Names for var1

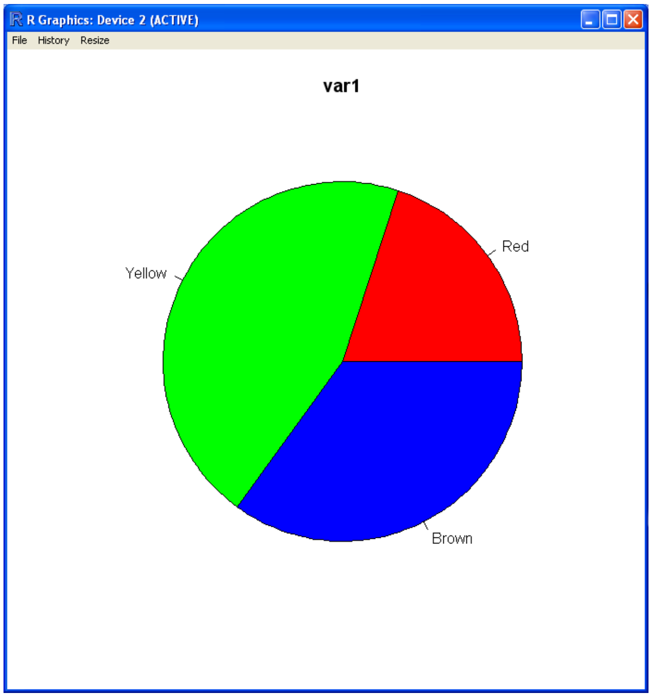
Numeric value	Level name
1	Red
2	Yellow
3	Brown

3) Go to the “Graphs” menu and select “Pie chart”. Then Choose the variable you want to graph and click OK. (If Bar Graphs is selected, R will make bar chart for you.)



```
> Dataset <- edit
> Dataset$var1 <-                    labels=c('Red',
> pie(table(Datas                    ls(Dataset$var1)
```

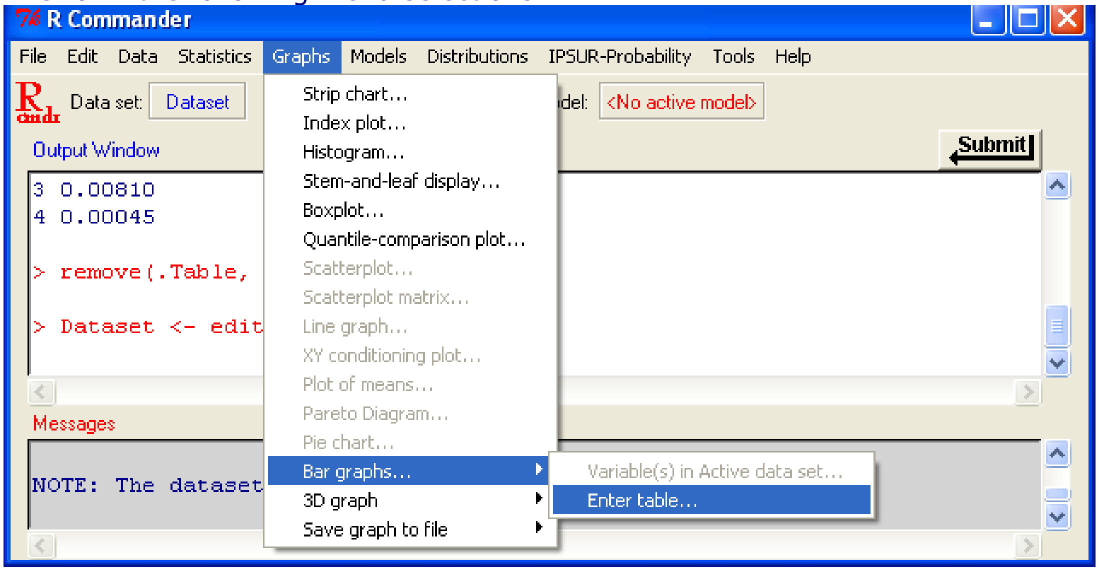
4) The result:



Exploring Relation Between Two Categorical Variable: Cluster Bar Chart

Making bar chart **when frequency data** is available. The example data (see step 2 in this instruction) is for comparing among male and female subjects to see if there is difference in their preference in a policy based on their Yes and No votes.

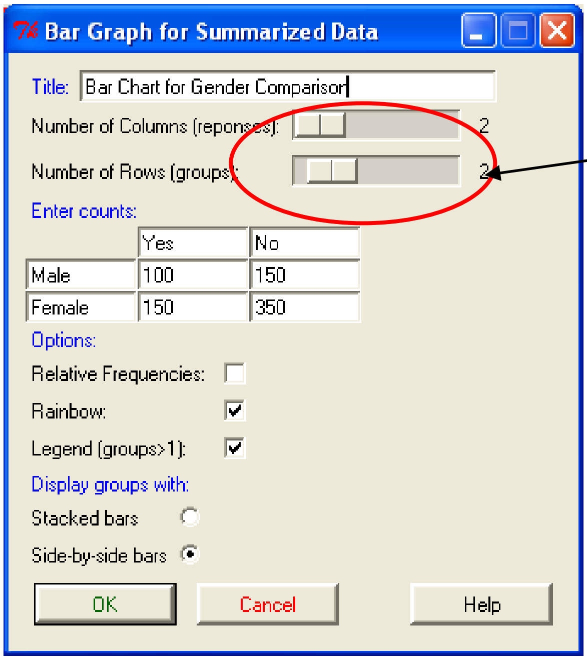
1) Perform the following menu selections:



```
3 0.00810
4 0.00045

> remove(.Table,

> Dataset <- edit
```

```
NOTE: The dataset
```

2) Enter the information asked to formulate your Bar Chart with the **side-by-side bar** chart (or cluster bar chart) and then click OK



Title: Bar Chart for Gender Comparison

Number of Columns (reponses): 2

Number of Rows (groups): 2

Enter counts:

	Yes	No
Male	100	150
Female	150	350

Options:

Relative Frequencies: □

Rainbow: ☑

Legend (groups>1): ☑

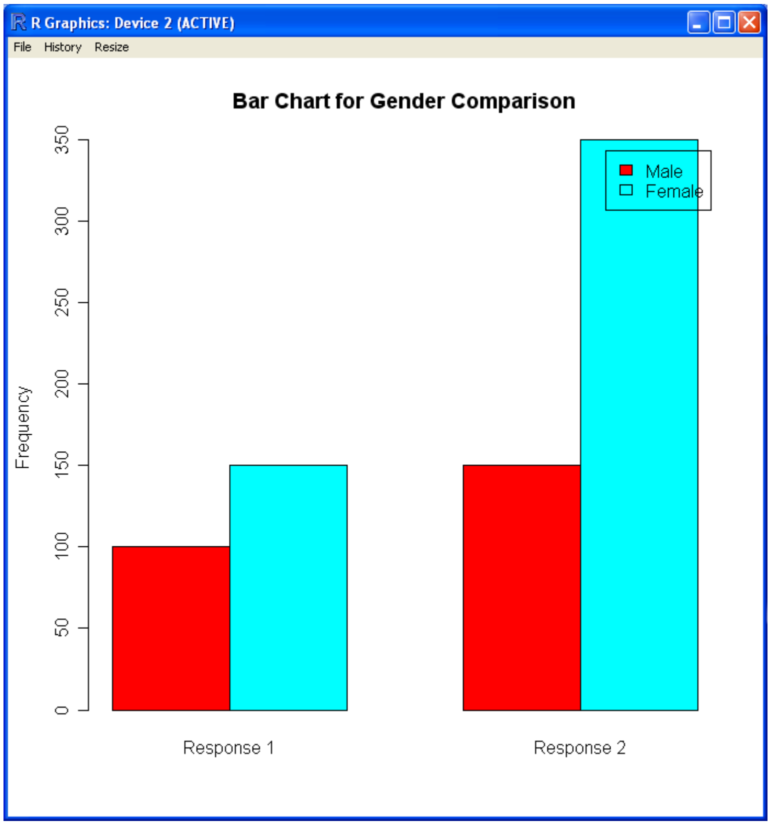
Display groups with:

Stacked bars ○

Side-by-side bars ●

Click and drag the adjustment buttons to determine the dimension of the table. This example is a 2x2 data. Use can make a 1x5, 3x5, and chart from any other possible dimension.

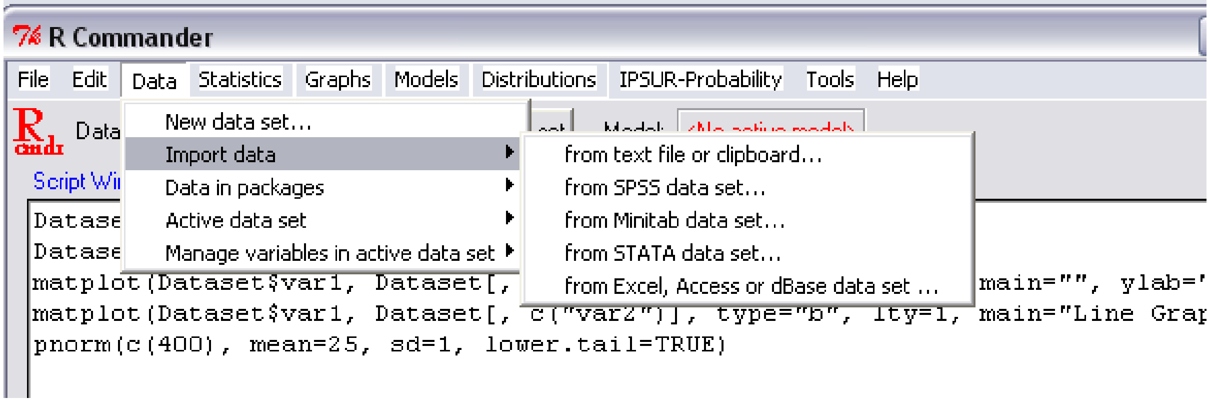
3) Result:


R Graphics: Device 2 (ACTIVE)
File
History
Resize
Bar Chart for Gender Comparison
Frequency
0
50
100
150
200
250
300
350
Male
Female
Response 1
Response 2

Exploring Relation Between Two Categorical Variable: Scatter Plot

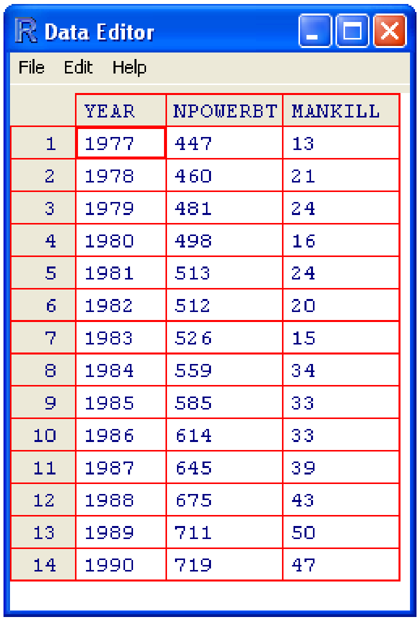
The following example is for making a scatter plot for observing the paired data (NPOWERBT, MANKILL) with NPOWERBT and the *x* variable and MANKILL as the *y* variable.

1) First, enter your data in R commander (or upload an existing file) into R by clicking on **Data** and then select **New data set … (or Import Data)**.



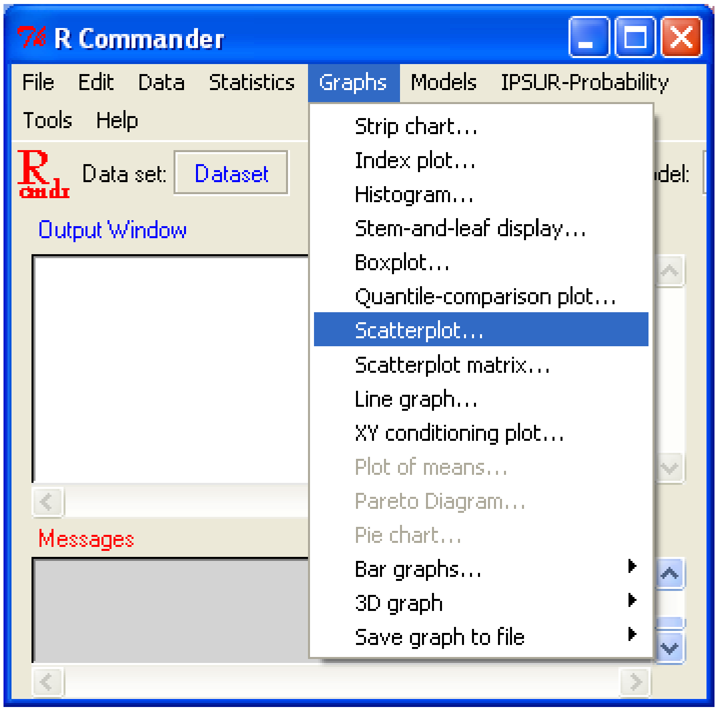
```
Datase
Datase
matplot(Dataset$var1, Dataset[,                                    main="", ylab='
matplot(Dataset$var1, Dataset[, c("var2")], type="b", lty=1, main="Line Grap
pnorm(c(400), mean=25, sd=1, lower.tail=TRUE)
```

The following chart shows the 14 cases entered in this example for scatter plot.



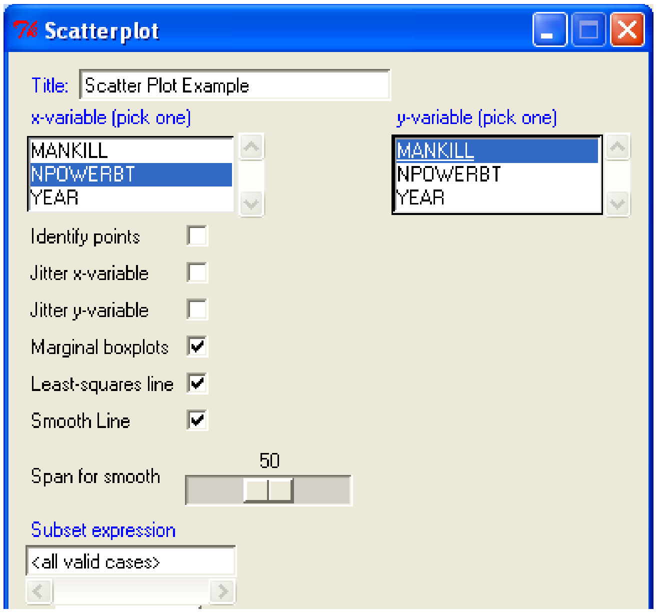
	YEAR	NPOWERBT	MANKILL
1	1977	447	13
2	1978	460	21
3	1979	481	24
4	1980	498	16
5	1981	513	24
6	1982	512	20
7	1983	526	15
8	1984	559	34
9	1985	585	33
10	1986	614	33
11	1987	645	39
12	1988	675	43
13	1989	711	50
14	1990	719	47

2) To make a scatter plot, use the following menu selections also figure on the right.
Graphs -> Scatterplot…



3) Enter information to the dialog box for making scatter plot.

Enter Graph title Scatter Plot Example
Choose X Variable NPOWERBT
Choose Y Variable MANKILL
Then, Click **OK**



Title: Scatter Plot Example

x-variable (pick one): MANKILL, NPOWERBT (selected), YEAR

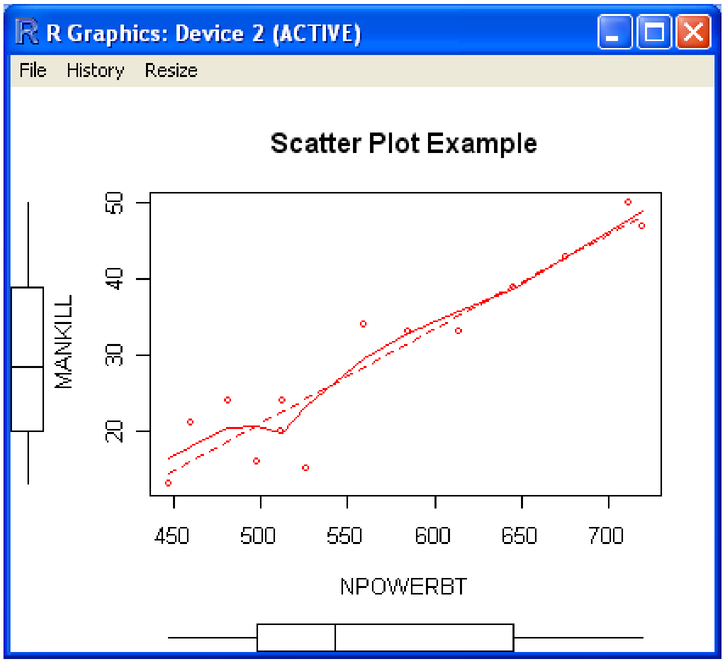
y-variable (pick one): MANKILL (selected), NPOWERBT, YEAR

Identify points □
Jitter x-variable □
Jitter y-variable □
Marginal boxplots ☑
Least-squares line ☑
Smooth Line ☑

Span for smooth 50

Subset expression
<all valid cases>

Results: This is positive linear correlation shown in the chart. As the value of NPOWERBT increases and the value of MANKILL increases too.

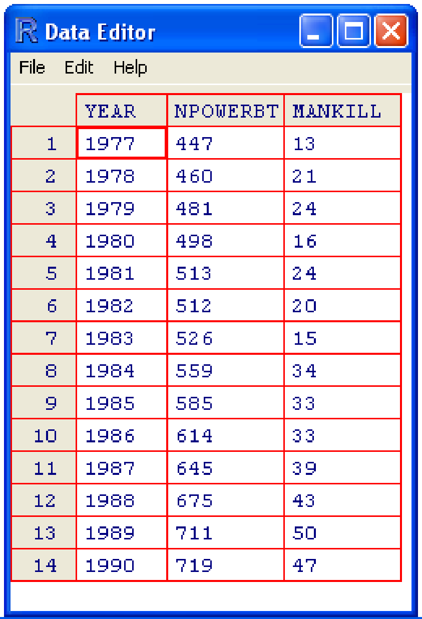


One can uncheck the marginal boxplot, Least-square line, and smooth line to obtain a scatter plot with only data points on the chart.

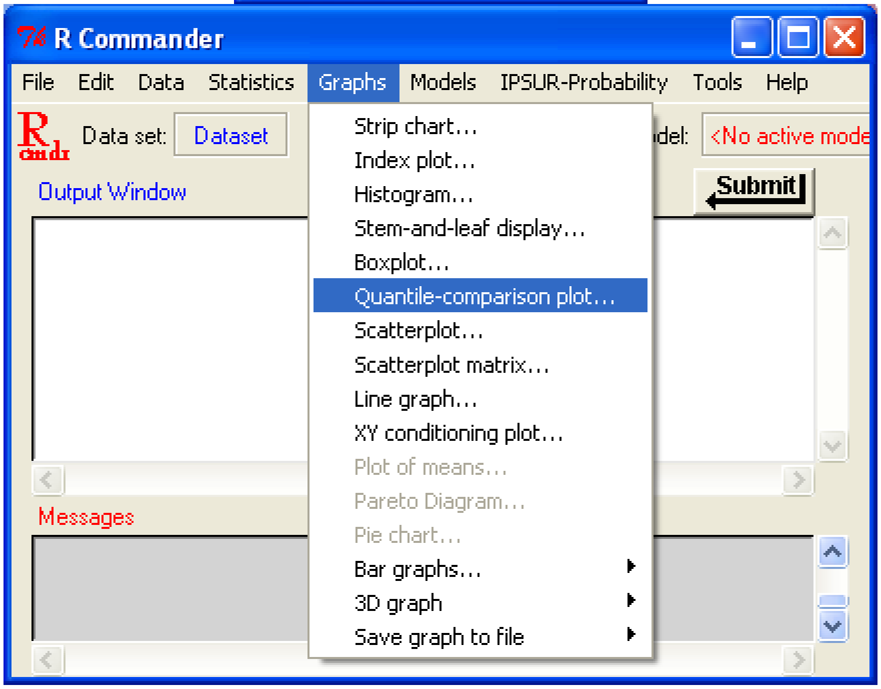
Quantile-comparison Plot (or Quantile-Quantile Plot/ QQ Plot) for Checking Normality Assumption

Example: Given a set of 14 values in the variable NPOWERBT, test the normality using a Quantile-comparison plot.

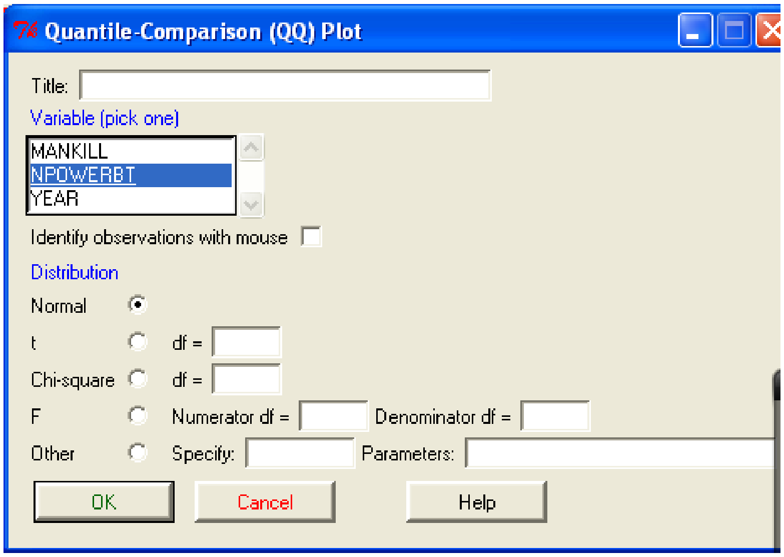
1) With the data set in the following Data Editor, from R Commander, click and select **Graphs** > **Quantile-comparison plot…**



	YEAR	NPOWERBT	MANKILL
1	1977	447	13
2	1978	460	21
3	1979	481	24
4	1980	498	16
5	1981	513	24
6	1982	512	20
7	1983	526	15
8	1984	559	34
9	1985	585	33
10	1986	614	33
11	1987	645	39
12	1988	675	43
13	1989	711	50
14	1990	719	47



2) Once in the Quantile-Comparison (QQ) Plot dialog box, enter the title of the plot, although, this is not necessary. Then click on the variable you wish to test and be sure the Normal distribution bullet is checked for testing normality and click “OK.”



Quantile-Comparison (QQ) Plot

Title:

Variable (pick one)

MANKILL
NPOWERBT
YEAR

Identify observations with mouse □

Distribution

Normal ●

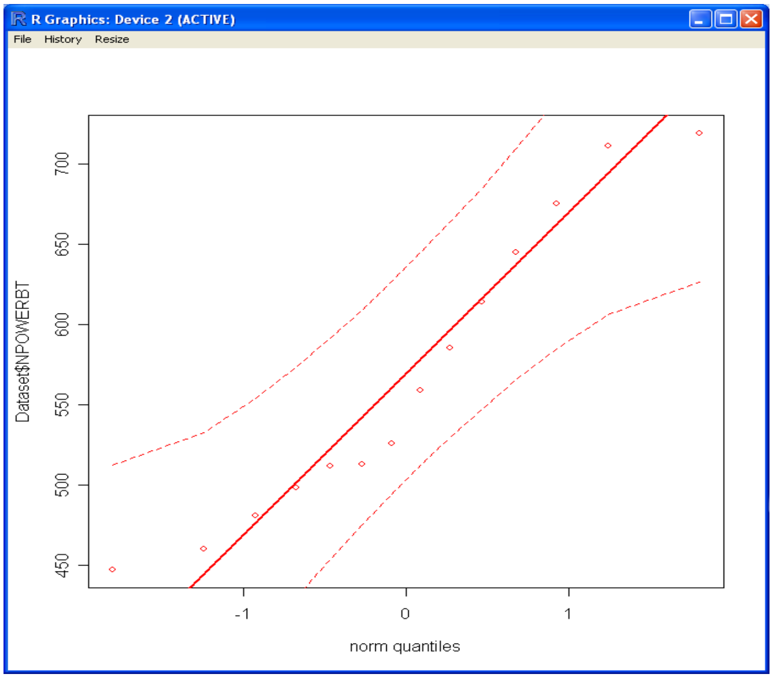
t ○ df =

Chi-square ○ df =

F ○ Numerator df = Denominator df =

Other ○ Specify: Parameters:

3) Go back to the RGui and there should be a new window with a quantile comparison plot in front of all other windows.



If all of the points on this plot are within the dotted line boundaries, the sample data set can be assumed to be from a normal distribution. Otherwise, one can assume the set does not follow a normal distribution.

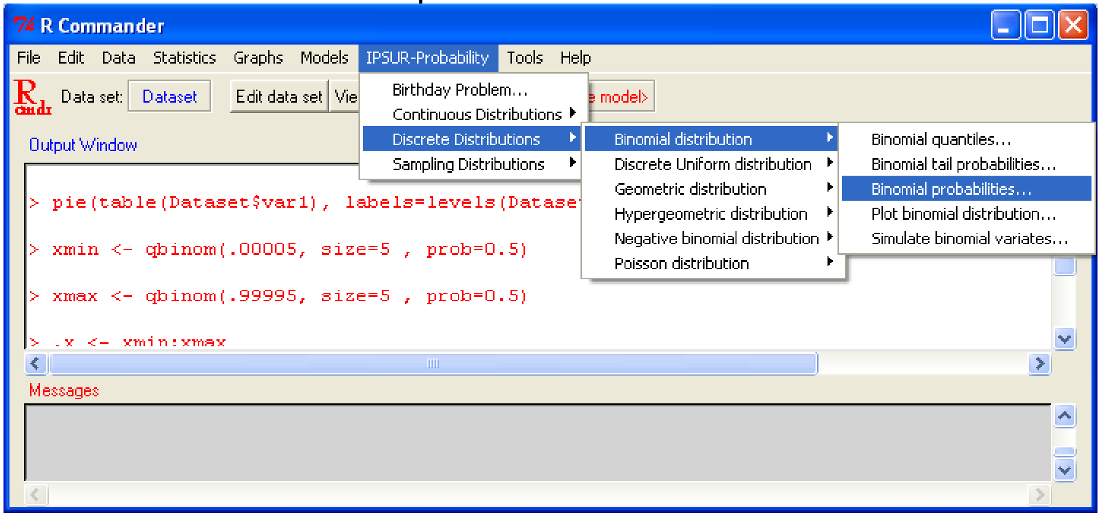
Probability

Binomial Probabilities

Example Problem: If 10% of the population in a community have a certain disease, what is the probability that 4 people in a random sample of 5 people from this community has the disease?
Identify n = 5, p = .10, x = 4

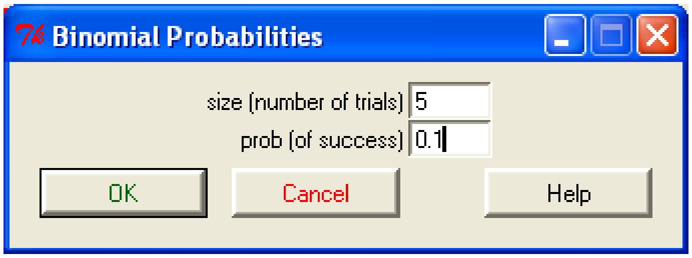
1) To solve this problem perform the following menu selections:

IPSUR-Probability /
Discrete Distributions /
Binomial distribution /
Binomial probabilities …



```
> pie(table(Dataset$var1), labels=levels(Datase
> xmin <- qbinom(.00005, size=5 , prob=0.5)
> xmax <- qbinom(.99995, size=5 , prob=0.5)
> .x <- xmin:xmax
```

2) Then enter your data. And click OK.



size (number of trials) 5
prob (of success) 0.1

Output from R Commander gives the probability distribution (p.m.f.):

```
        Pr
0 0.59049
1 0.32805
2 0.07290
3 0.00810
4 0.00045
```

← P(X = 4) = 0.00045

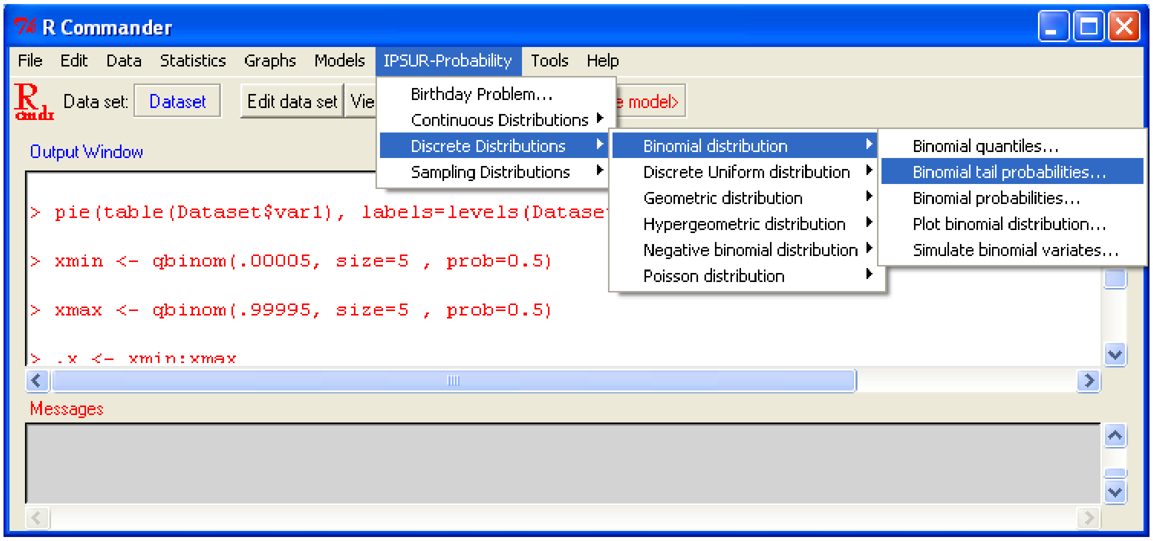
So, the probability that 4 people in a random sample of 5 people from this community has the disease is 0.00045.

For Computing Tail Probability

Example: What is the probability that 4 people or less in a random sample of 5 people from this community has the disease?
Identify n = 5, p = .10, x = 4

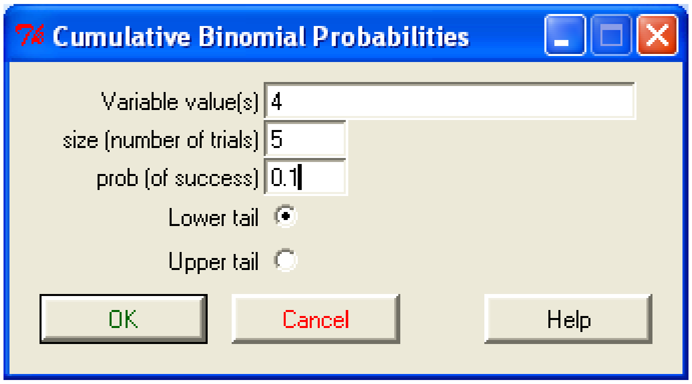
1) To solve this problem perform the following menu selections:

IPSUR-Probability /
Discrete Distributions /
Binomial distribution /
Binomial tail probabilities …



```
> pie(table(Dataset$var1), labels=levels(Datase
> xmin <- qbinom(.00005, size=5 , prob=0.5)
> xmax <- qbinom(.99995, size=5 , prob=0.5)
> .x <- xmin:xmax
```

2) Then enter the data in the dialog box for specifying the event, probability of success for each Bernoulli trial and choose whether and choose Lower tail since the probability of 4 or less is to be computed. (If Upper tail is selected, the probability computed would be P(X>4) = P(X≥5).)Click OK.



Cumulative Binomial Probabilities

Variable value(s) 4

size (number of trials) 5

prob (of success) 0.1

Lower tail ●

Upper tail ○

Result:

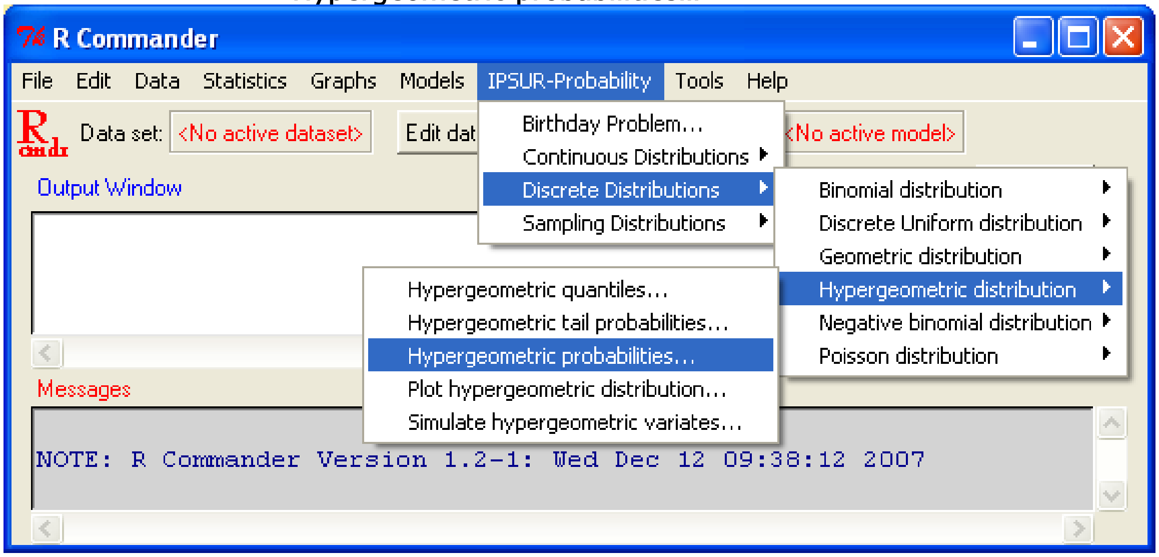
```
> pbinom(c(4), size=5, prob=0.1, lower.tail=TRUE)
[1] 0.99999
```

So, the probability of observing 4 or less is P(X ≤ 4) = 0.99999.

Hypergeometric Probability

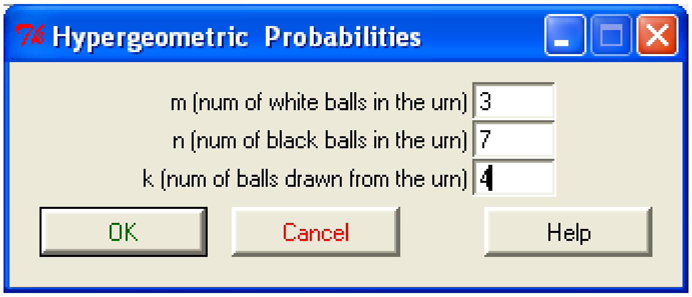
Example: In a box containing 10 products, 3 of them are defective and 7 are good. If 4 are selected at random from these 10 products without replacement what is the probability that 2 of them will be defective products?

Step 1: Click through the following menu selections:
IPSUR-Probability
Discrete distributions
Hypergeometric probabilities…


R Commander
File Edit Data Statistics Graphs Models IPSUR-Probability Tools Help
Data set: <No active dataset>
Edit dat
<No active model>
Output Window
Messages
Birthday Problem…
Continuous Distributions
Discrete Distributions
Sampling Distributions
Binomial distribution
Discrete Uniform distribution
Geometric distribution
Hypergeometric distribution
Negative binomial distribution
Poisson distribution
Hypergeometric quantiles…
Hypergeometric tail probabilities…
Hypergeometric probabilities…
Plot hypergeometric distribution…
Simulate hypergeometric variates…

```
NOTE: R Commander Version 1.2-1: Wed Dec 12 09:38:12 2007
```

Step 2: The m in the dialog should be 3 and n should be 7, and the number of products selected in the random sample k would be 4. Enter these values into the dialog box as shown below and click OK. The probability distribution for this sampling will be displayed in the R Commander window.



m (num of white balls in the urn) 3
n (num of black balls in the urn) 7
k (num of balls drawn from the urn) 4

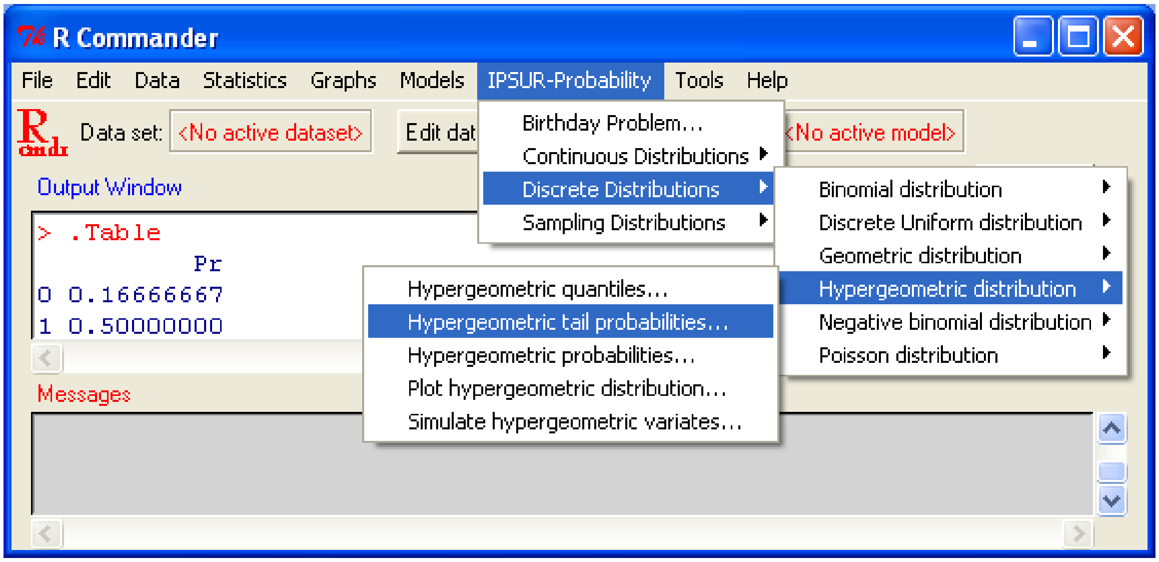
R Output

```
          Pr
  0 0.16666667
  1 0.50000000
  2 0.30000000
  3 0.03333333
```

(The answer to this problem is $P(X = 2) = 0.3$.)

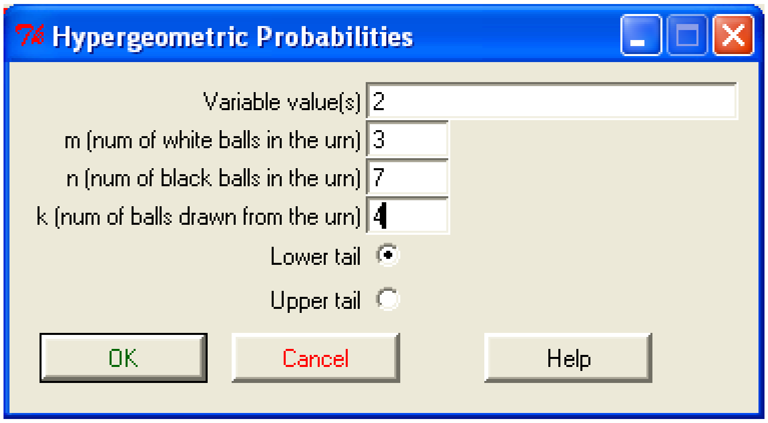
Tail Probability (Cumulative Probability)

If one wishes to find the cumulative probability such as the probability of having 2 or less defective products, then one should choose the tail probability option.



```
> .Table
          Pr
0 0.16666667
1 0.50000000
```

And, in the dialog box enter value 2 in the variable value(s) box to specify the event and check Lower tail bullet since the probability P(X ≤ 2) is to be calculated. The rest of boxes would be the same as first example. And, click OK.



Hypergeometric Probabilities

Variable value(s) 2

m (num of white balls in the urn) 3

n (num of black balls in the urn) 7

k (num of balls drawn from the urn) 4

Lower tail ●

Upper tail ○

R Output

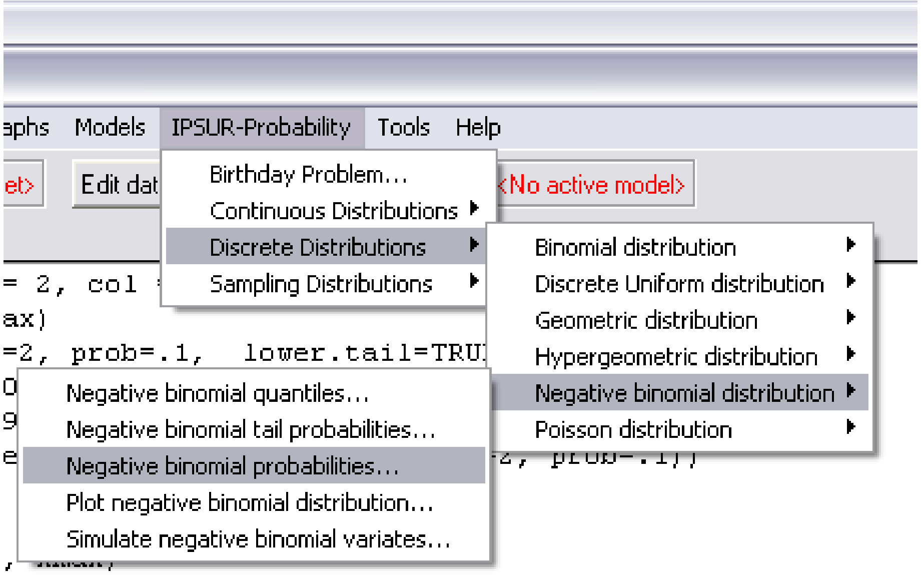
```
> phyper(c(2), m=3, n=7, k=4, lower.tail=TRUE)
[1] 0.9666667
```

So, the answer would be 0.9666667.

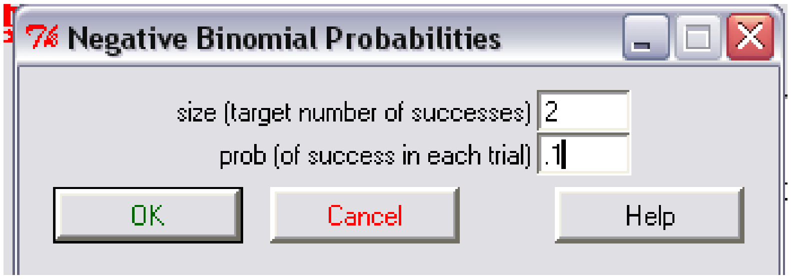
Negative Binomial Distribution

Example: It is reported that 10% of the apples from the Apple Farm are bad. If apples are randomly selected from this farm one after another, what is the probability that the <u>10th apple</u> selected will be the <u>2nd bad apple</u> selected?

1.) In R Commander, select **IPSUR-Probability, Discrete Distributions, Negative binomial distribution, and Negative binomial probabilities.**


aphs
Models
IPSUR-Probability
Tools
Help
et>
Edit dat
<No active model>
Birthday Problem...
Continuous Distributions
Discrete Distributions
Sampling Distributions
Binomial distribution
Discrete Uniform distribution
Geometric distribution
Hypergeometric distribution
Negative binomial distribution
Poisson distribution
Negative binomial quantiles...
Negative binomial tail probabilities...
Negative binomial probabilities...
Plot negative binomial distribution...
Simulate negative binomial variates...

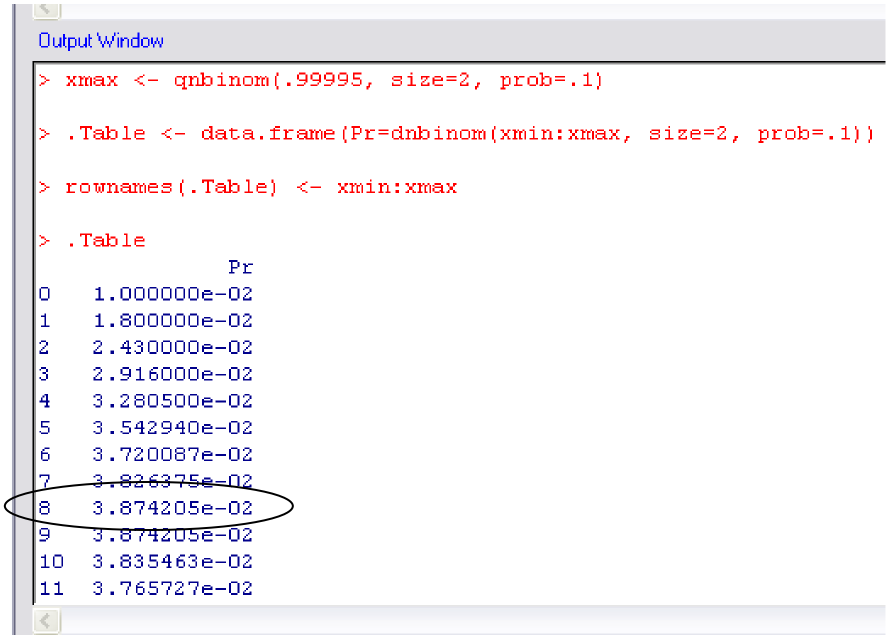
2.) Enter information into the Negative binomial probabilities dialogue box. The number of successes equals 2 and the probability of success equals .1.



size (target number of successes) 2

prob (of success in each trial) .1

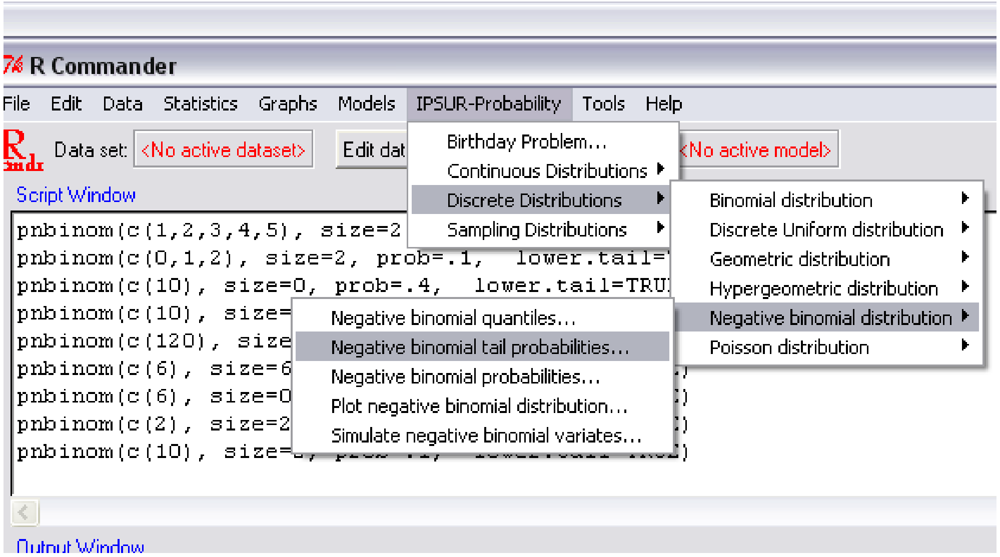
3.) R displays a list of values similar to the one shown below. Find the number of successes (10 in this case) and read the probability. **Note that the Negative Binomial random variable takes on number failures before the rth success** So, the answer to this problem is around 0.0387 that is probability next to number 8 in the R output. It is the probability of observing 8 failures before the 2nd success.



```
> xmax <- qnbinom(.99995, size=2, prob=.1)

> .Table <- data.frame(Pr=dnbinom(xmin:xmax, size=2, prob=.1))

> rownames(.Table) <- xmin:xmax

> .Table
             Pr
0   1.000000e-02
1   1.800000e-02
2   2.430000e-02
3   2.916000e-02
4   3.280500e-02
5   3.542940e-02
6   3.720087e-02
7   3.826375e-02
8   3.874205e-02
9   3.874205e-02
10  3.835463e-02
11  3.765727e-02
```

Tail Probability Example: If 30% of the cars passing your house are red what is the probability that more than 5 cars will pass before you observe the first red one? (Geometric Distribution: a special case of Negative Binomial Distribution.)

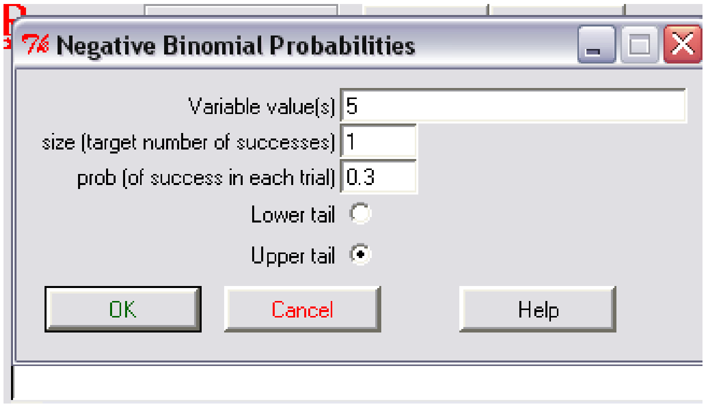
1) In R Commander, select **IPSUR-Probability, Discrete Distributions, Negative binomial distribution, Negative binomial tail probabilities**



R Commander
File Edit Data Statistics Graphs Models IPSUR-Probability Tools Help
Data set: <No active dataset>
Edit dat
<No active model>
Script Window
Birthday Problem...
Continuous Distributions
Discrete Distributions
Sampling Distributions
Binomial distribution
Discrete Uniform distribution
Geometric distribution
Hypergeometric distribution
Negative binomial distribution
Poisson distribution
Negative binomial quantiles...
Negative binomial tail probabilities...
Negative binomial probabilities...
Plot negative binomial distribution...
Simulate negative binomial variates...
Output Window

```
pnbinom(c(1,2,3,4,5), size=2
pnbinom(c(0,1,2), size=2, prob=.1,  lower.tail=
pnbinom(c(10), size=0, prob=.4,  lower.tail=TRU
pnbinom(c(10), size=
pnbinom(c(120), size
pnbinom(c(6), size=6
pnbinom(c(6), size=0
pnbinom(c(2), size=2
pnbinom(c(10), size=
```

2) Enter information into the Negative binomial probabilities dialogue box. The number of successes equals 1 and the probability of success equals .3. Since more than 5 cars is requested, the variable value is 5 and Upper tail probability option should be checked. (If Lower tail is selected, then the probability would be P(X ≤ 5).)



Negative Binomial Probabilities

Variable value(s) 5

size (target number of successes) 1

prob (of success in each trial) 0.3

Lower tail ○

Upper tail ●

3) Results should indicate approximately an 11% chance that more than 5 car will pass before seeing the first red one.

R output:

```
> pnbinom(c(5), size=1, prob=0.3,  lower.tail=FALSE)
[1] 0.117649
```

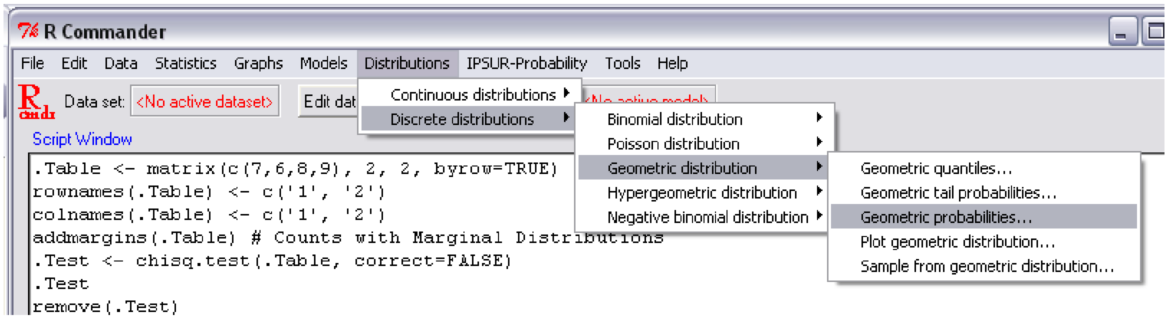
Probabilities for Geometric Distributions

Example: A study stated that 1 in every 20 (5.0%) children born is diagnosed with autism. Suppose you are an OBGYN and you deliver babies. The occurrence of autism is at random from the population. What is the probability that the 25th baby is the first to be diagnosed with autism?

From IPSUR,

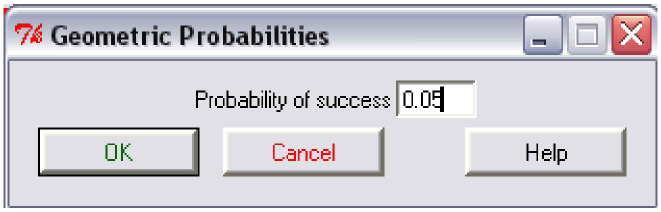
1) make the following menu selections:

Distributions/Discrete Distributions/Geometric Distribution/Geometric Probabilities…



```
.Table <- matrix(c(7,6,8,9), 2, 2, byrow=TRUE)
rownames(.Table) <- c('1', '2')
colnames(.Table) <- c('1', '2')
addmargins(.Table) # Counts with Marginal Distributions
.Test <- chisq.test(.Table, correct=FALSE)
.Test
remove(.Test)
```

2) Since geometric distributions are just negative binomial distributions with r = 1, then the p.m.f. (in R) is: $f(x) = p(1-p)^x$
The following window will come up, where you input the mean percentage of the occurrence, in this case 0.05.



Probability of success 0.05

Interpret:

3) A list of probabilities will appear in the IPSUR output window, where you will choose the probability value for the random variable x that is specified, which in this case is 25.
   (In R software, the Geometric random variable takes of number failures before the r-th success. Since it takes X=24 failures to get the 25th success, so the probability is shown next to number 24.)

```
            Pr
0   5.000000e-02
1   4.750000e-02
2   4.512500e-02
3   4.286875e-02
 …             …
24  1.459945e-02
25  1.386948e-02
```

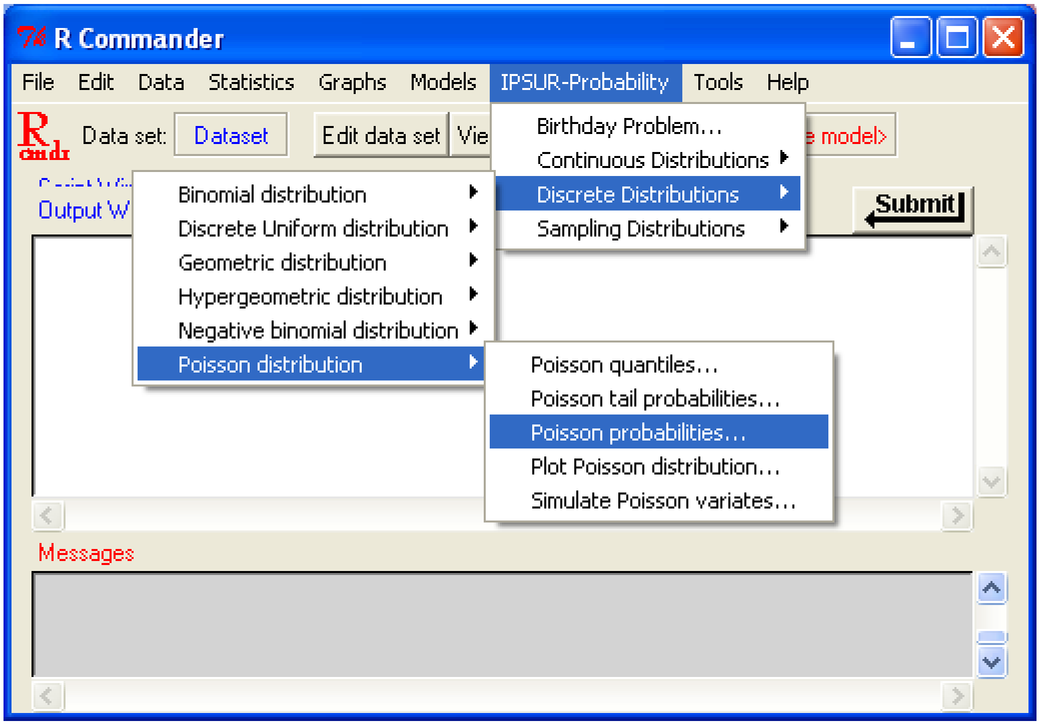
So, P(X=24) = 1.46% is found.

Poisson Probabilities with R

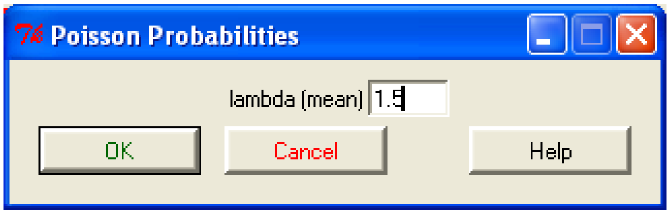
Example: Customers arrive at a travel agency at a mean rate of 3 per 20 minutes from 10:00 a.m. to 2:00 p.m. Assuming that the customers' arrivals follow a Poisson process. Find the probability that no customers will arrive between 12:50 to 1:00 (so that you can sneak out for a quick lunch).

1) First determine the average per time interval asked. In this example, the time interval would be 10 minutes. Given the average per 20 minutes is 3, the average per 10 minutes would be 1.5.

2) Then from R Commander, click on **IPSUR-Probability/Discrete Distributions/Poisson distributions/Poisson probability…**

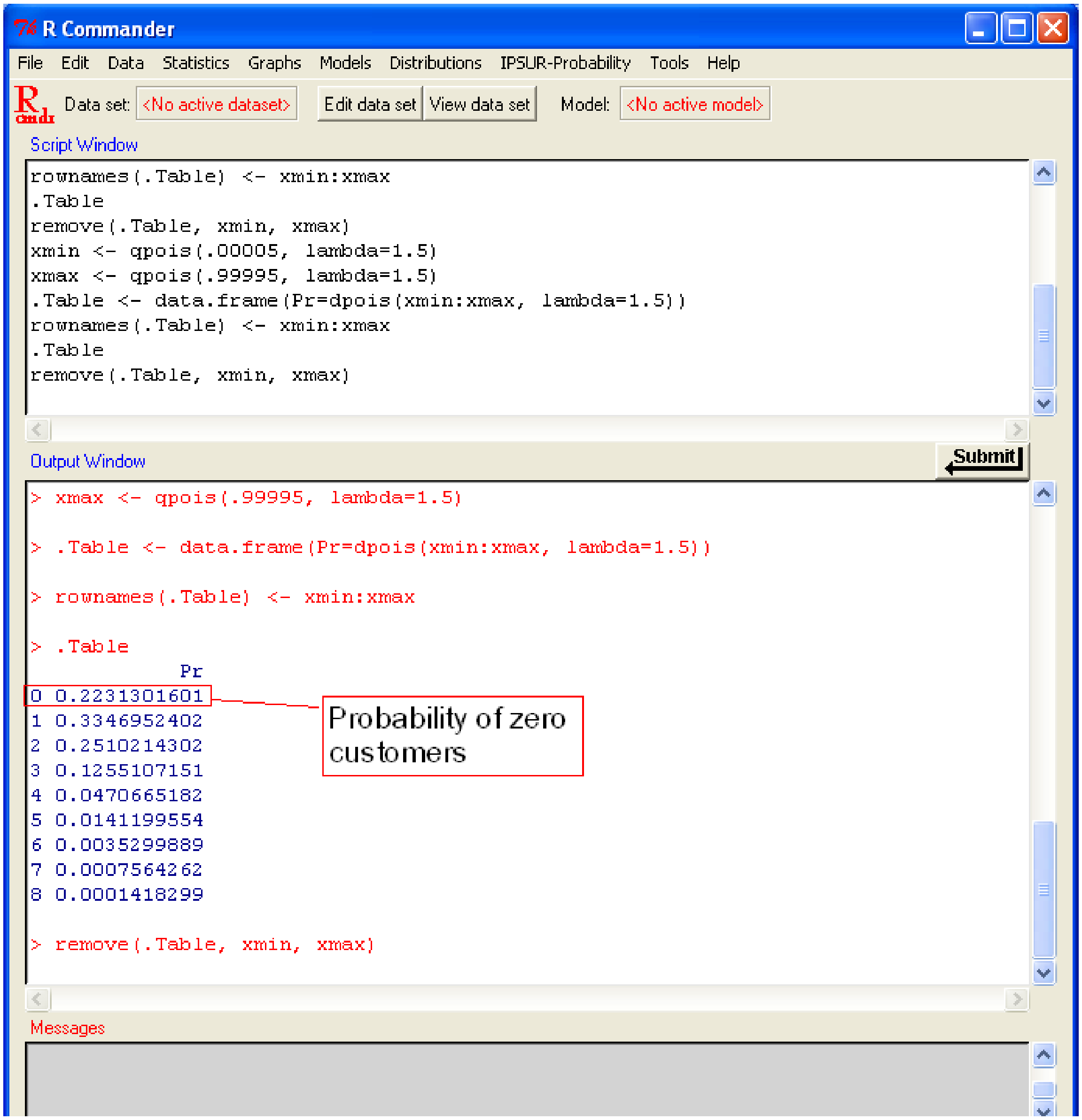


3) Enter the mean from part 1) into the dialog box.



lambda (mean) 1.5

4) Use the table provided in the display box of IPSUR to determine the probability of 0 customers within ten minutes.


R Commander
File Edit Data Statistics Graphs Models Distributions IPSUR-Probability Tools Help
Data set: <No active dataset>
Edit data set
View data set
Model: <No active model>
Script Window
Output Window
Submit
Messages

```
rownames(.Table) <- xmin:xmax
.Table
remove(.Table, xmin, xmax)
xmin <- qpois(.00005, lambda=1.5)
xmax <- qpois(.99995, lambda=1.5)
.Table <- data.frame(Pr=dpois(xmin:xmax, lambda=1.5))
rownames(.Table) <- xmin:xmax
.Table
remove(.Table, xmin, xmax)
```

```
> xmax <- qpois(.99995, lambda=1.5)

> .Table <- data.frame(Pr=dpois(xmin:xmax, lambda=1.5))

> rownames(.Table) <- xmin:xmax

> .Table
             Pr
0  0.2231301601
1  0.3346952402
2  0.2510214302
3  0.1255107151
4  0.0470665182
5  0.0141199554
6  0.0035299889
7  0.0007564262
8  0.0001418299

> remove(.Table, xmin, xmax)
```

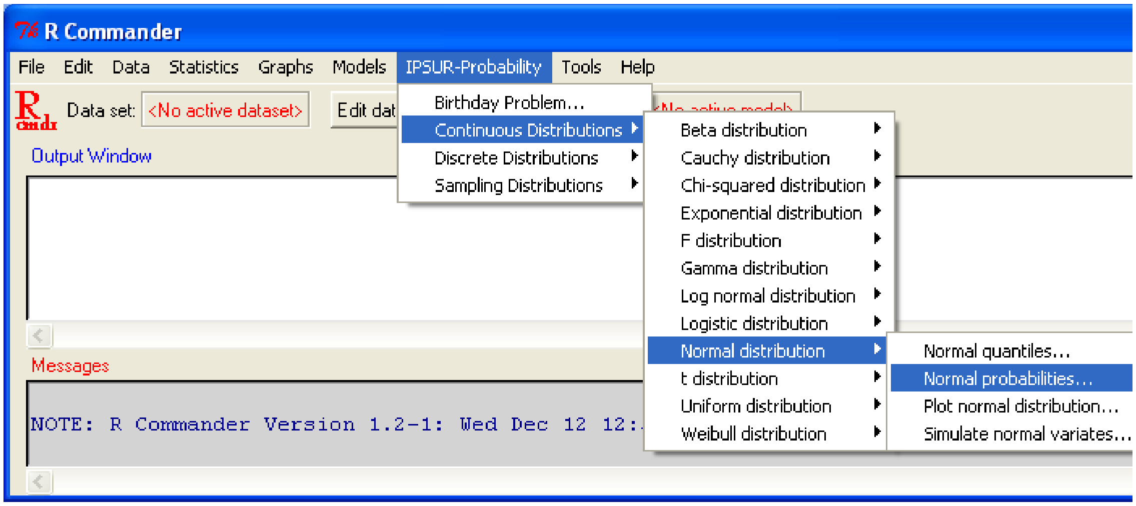
Probability of zero customers

The probability of X = x is listed and the probability of X = 0 is around 0.2231.

Normal Distribution

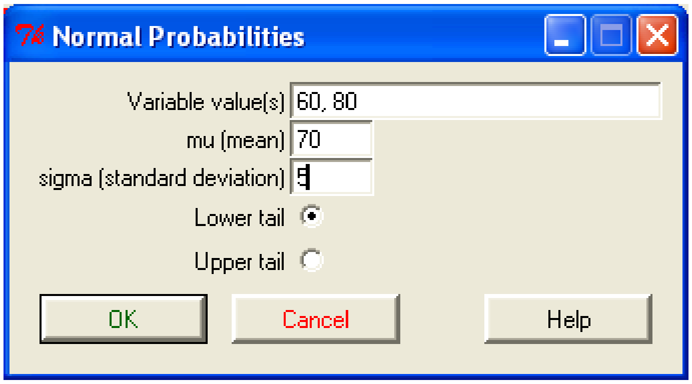
Example: The pulse rates for a certain population follow a normal distribution with a mean of 70 per minute and s.d. 5. What percent of this distribution that is in between 60 to 80 per minute?

1) From IPSUR, select **IPSUR-Probability, Continuous Distributions, Normal distribution, Normal probabilities …**



```
NOTE: R Commander Version 1.2-1: Wed Dec 12 12:
```

1) In this dialog box, fill in the given value of 70 for the mean, 5 for the standard deviation, and then type the interval for the percent of distribution that you wish to find (60, 80 for this example). Also, check the box for lower tail.



Variable value(s) 60, 80

mu (mean) 70

sigma (standard deviation) 5

Lower tail ●

Upper tail ○

2) Once you click OK, 2 values on the R output screen will appear and read as follows.

```
> pnorm(c(110,150), mean=130, sd=10, lower.tail=TRUE)

[1] 0.02275013  0.97724987
```

3) Simply subtract the two answers given to find the probability of the distribution between the given interval of 60-80.

   .97724987 - .02275013 = **.95449974**

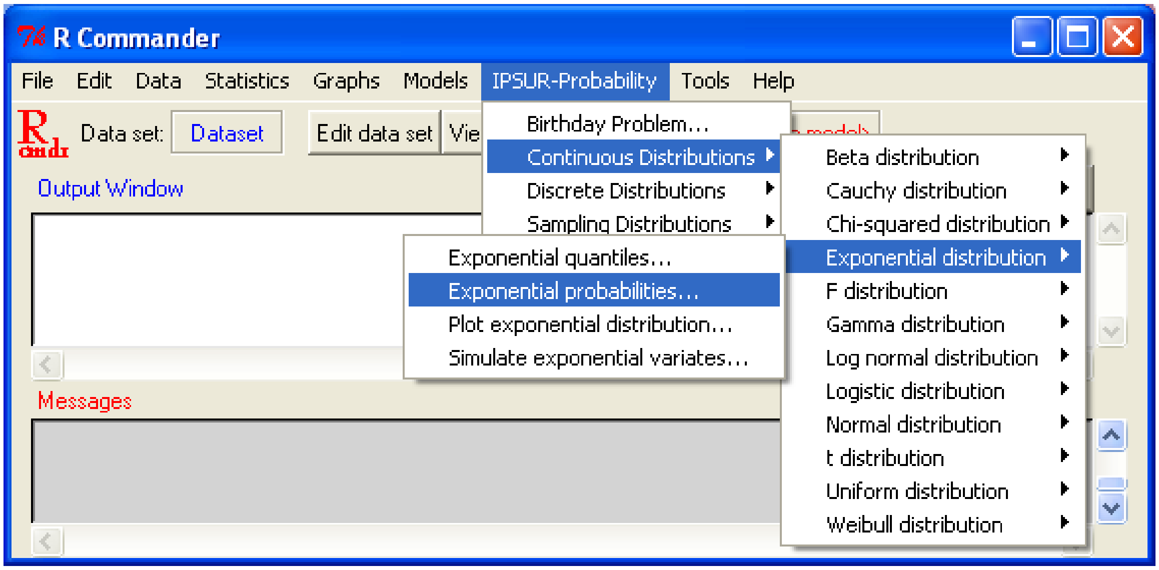
4) By selecting Upper tail in the dialog box instead of Lower Tail, the same answer will be given, only the values will be switched. You will still use the difference of the probability values to find your probability

Exponential Probabilities

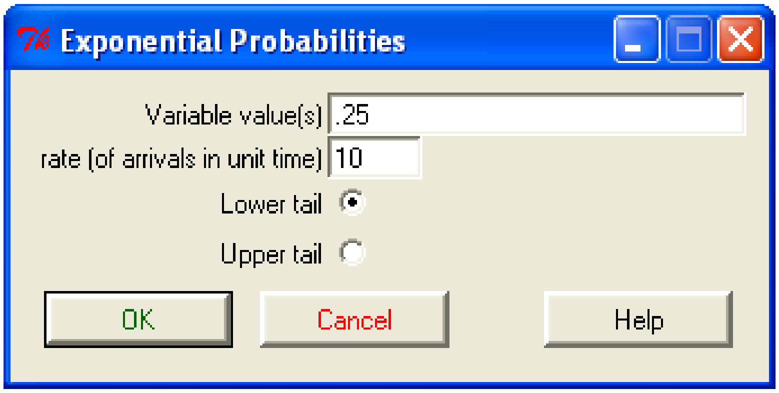
For this example we will analyze a specific Exponential Probability problem.: Suppose that number of arrivals of customers follows a Poisson process with a mean of 10 per hour. What is the probability that the next customer will arrive within 15 minutes? (15 min. = .25 hour)

1) To utilize R Commander to solve the probability: first select through the following menu selections:

IPSUR-Probability -> Continuous Distributions -> Exponential distribution -> Exponential probabilites



2) Enter the Data from the problem in the R Commander window as shown below:


Exponential Probabilities

Variable value(s) .25

rate (of arrivals in unit time) 10

Lower tail ●

Upper tail ○

3) From this problem the Rate is the 10 for the amount of customers arriving per hour. The variable value of 0.25 is the 15 minutes converted into hours to be of the same unit as the rate. 15 min = 0.25 hr
Select Lower tail because it is asking the "within 15 minutes" which is less than 15 minutes. And then, CLICK OK.

Interpret the Results:

```
> pexp(c(.25), rate=10, lower.tail=TRUE)
[1] 0.917915
```

This result means: With customers arriving on a mean of 10 per hour, there is 91.8% probability that the next customer will arrive within 15 minutes.

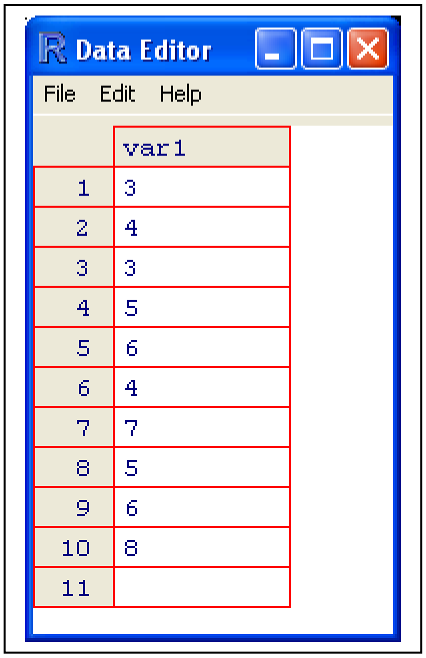
Remark: If **upper tail** option is chosen then the probability of great than 15 minutes, that is P(X>15), will be computed.

Statistical Inference

One Sample t-Test

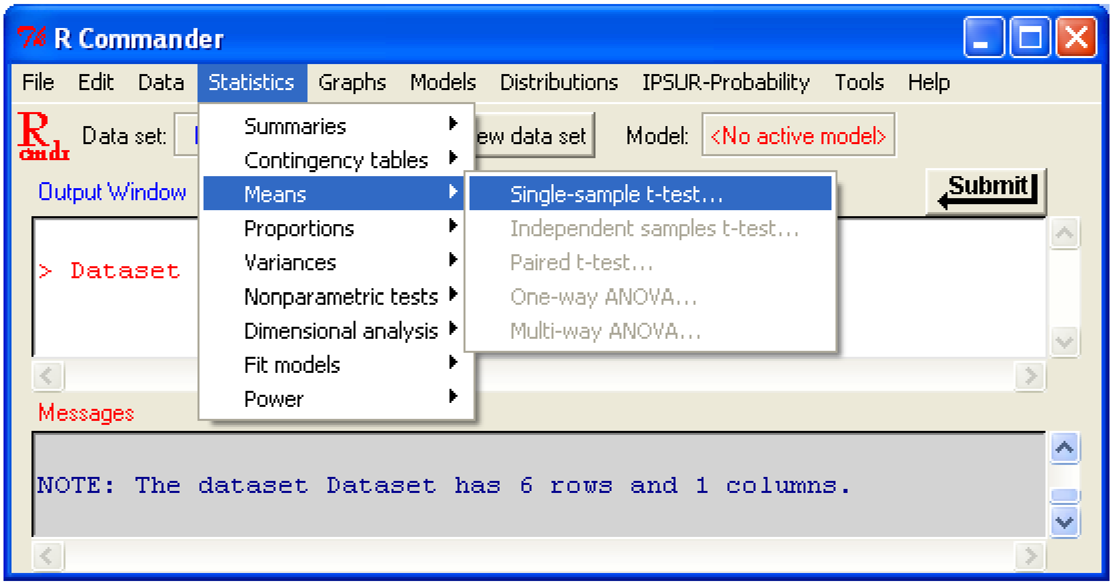
Example: In a study, one wishes to test whether the average of the test scores is significantly different from 6 or not, at 5% level of significance, using a sample 10 data values as shown in the Data Editor?

1) Enter your data into a new data table. Test for normality first. (Check the instruction for normality test.)



	var1
1	3
2	4
3	3
4	5
5	6
6	4
7	7
8	5
9	6
10	8
11	

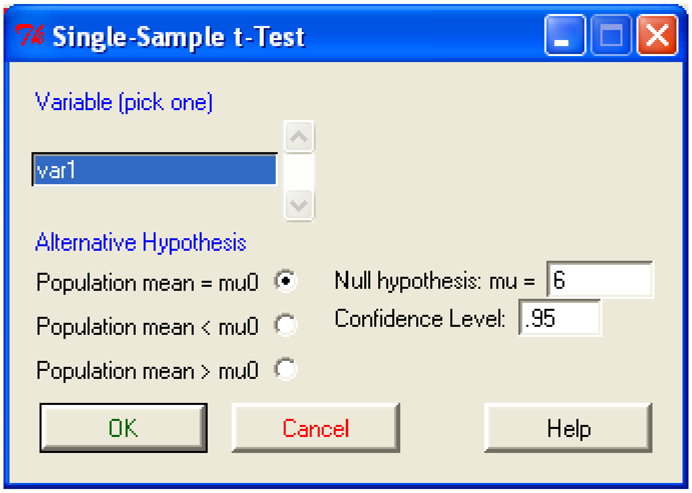
2) Perform the following menu selections:



```
> Dataset
```

```
NOTE: The dataset Dataset has 6 rows and 1 columns.
```

3) Then fill in the information that was given by the problem. For the null hypothesis that the mean is 6, put value 6 in the **Null hypothesis: mu =** box. Since the goal is for testing whether there is significant difference, one should choose Population mena = mu0 for choose two-tailed test, and click **OK.**



Population mean = mu0 ●
Population mean < mu0 ○
Population mean > mu0 ○
Null hypothesis: mu = 6
Confidence Level: .95

R Output:

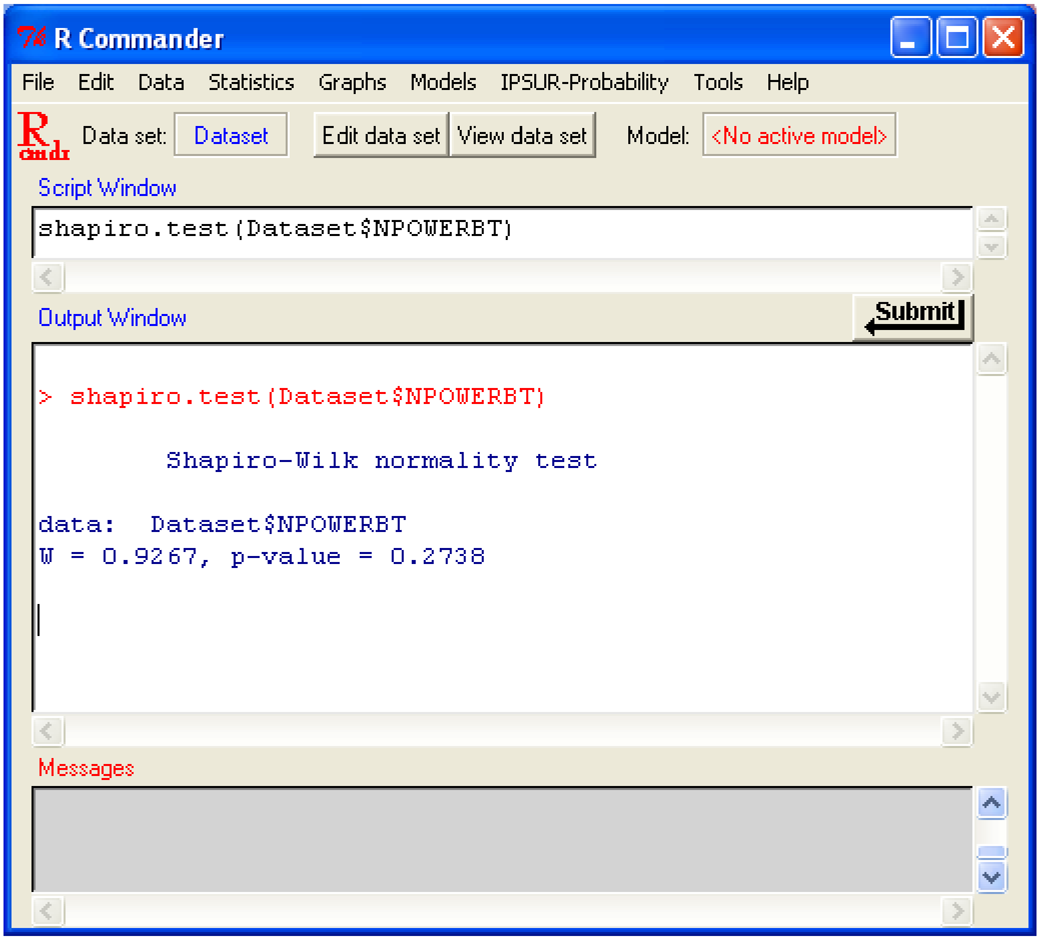
```
      One Sample t-test

data:  Dataset$var1
t = -1.7111, df = 9, p-value = 0.1212
alternative hypothesis: true mean is not equal to 6
95 percent confidence interval:
 3.910125 6.289875
sample estimates:
mean of x
      5.1
```

4) Interpret the result: The p-value is 0.1212 and it is greater than .05, the 5% level of significance. Therefore, there **is no sufficient evidence** to support the alternative hypothesis that the mean is significant difference from 60.

Normality Test

Example: Given the same data set, use the test for normality to check the normality of the MANKILL variable at 5% level of significance.

1) First, determine the name of the dataset as well as the name of the variable within the dataset being tested. In this case the dataset is named "Dataset" and the variable name is "NPOWERBT." Then in the **Script Window** type "shapiro.test(Dataset$NPOWERBT)" where "Dataset" is the name of the dataset, and "NPOWERBT" is the name of the variable being tested and click "Submit."



```
shapiro.test(Dataset$NPOWERBT)
```

```
> shapiro.test(Dataset$NPOWERBT)

        Shapiro-Wilk normality test

data:  Dataset$NPOWERBT
W = 0.9267, p-value = 0.2738
```

Interpret:

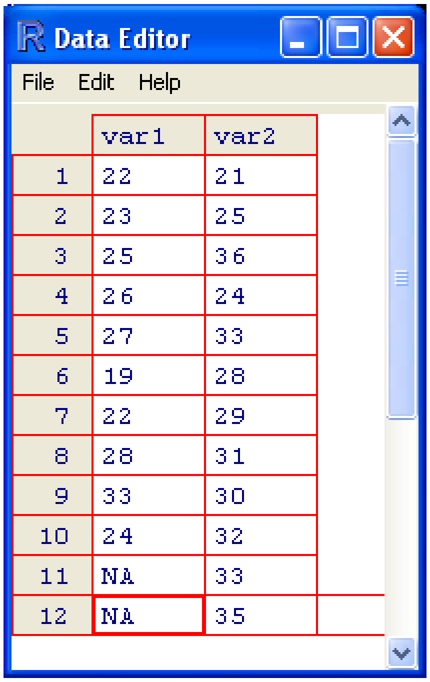
2) The output of the file will look like information in the lower half of the window above. The p-value inside the red box is the important part of this test. If the test p-value is greater than 5%, then the normality assumption is acceptable. In this case, the p-value is 0.2738 which is greater than 0.05 that implies the normality assumption of this data set is acceptable at 5% level of significance.

Two Independent Sample t-Test

Example: The following data is results from measuring the body mass index from two independent random samples from two populations.

Sample 1: 22, 23, 25, 26, 27, 19, 22, 28, 33, 24
Sample 2: 21, 25, 36, 24, 33, 28, 29, 31, 30, 32, 33, 35

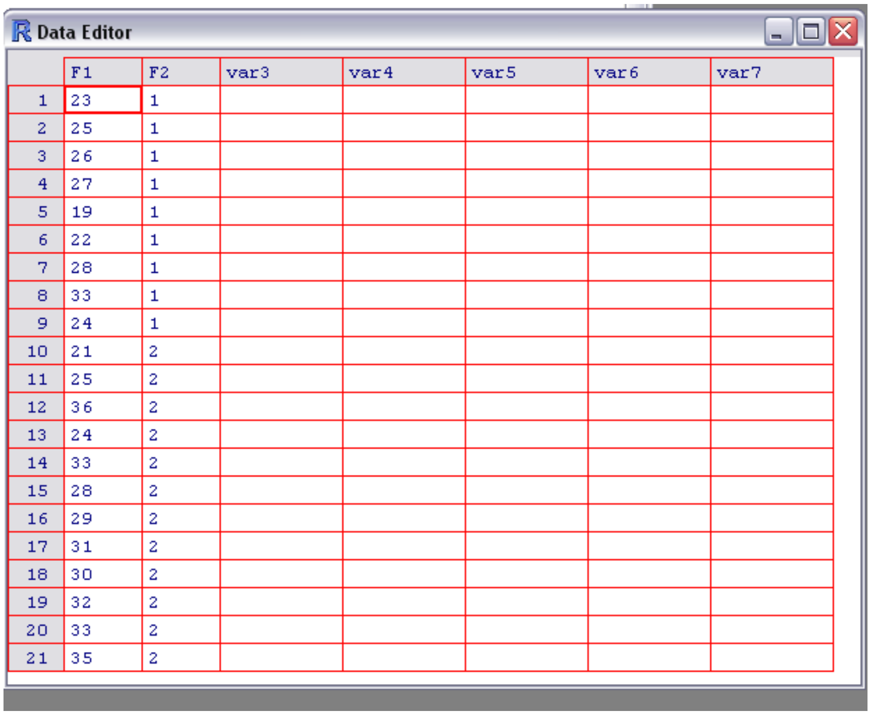
1) Use R to determine if **normality assumption** is correct. Start by arranging data into two columns as shown below.



	var1	var2
1	22	21
2	23	25
3	25	36
4	26	24
5	27	33
6	19	28
7	22	29
8	28	31
9	33	30
10	24	32
11	NA	33
12	NA	35

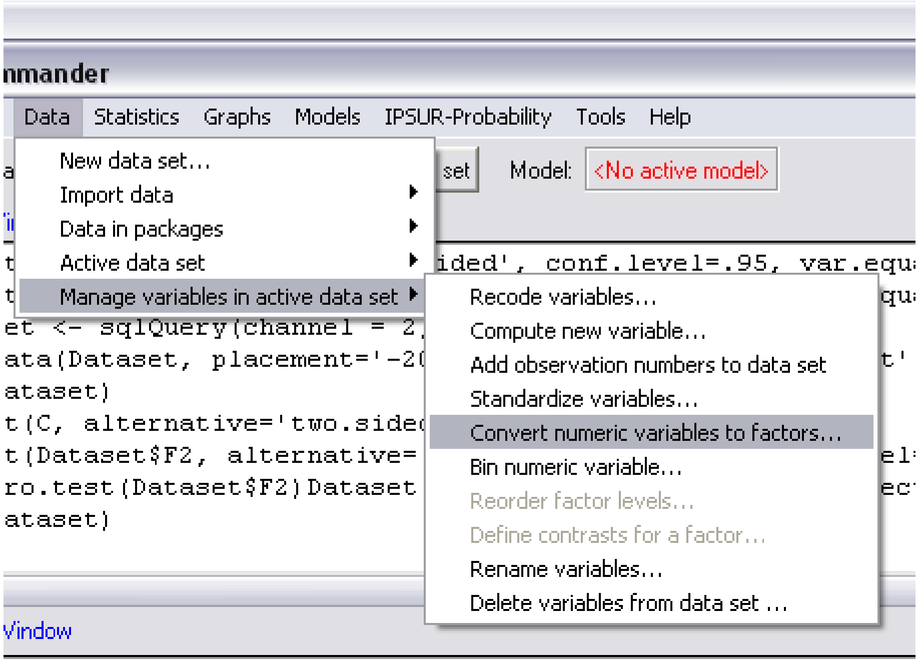
2) Perform shapiro.test() for each variable to determine if *p*-value is greater than .05; thus determining if the normality assumption is acceptable for both samples.

3) Next, data must be arranged in a single column with 1 or 2 placed in the column next to it denote which set the value is from (See Below).

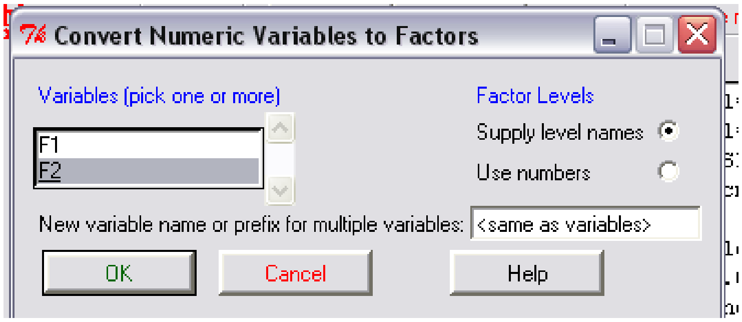


	F1	F2	var3	var4	var5	var6	var7
1	23	1					
2	25	1					
3	26	1					
4	27	1					
5	19	1					
6	22	1					
7	28	1					
8	33	1					
9	24	1					
10	21	2					
11	25	2					
12	36	2					
13	24	2					
14	33	2					
15	28	2					
16	29	2					
17	31	2					
18	30	2					
19	32	2					
20	33	2					
21	35	2					

4) Convert F1 into factor variables by clicking on Data, Manage variables in active data set, and Convert numeric variables to factors.



```
ided', conf.level=.95, var.equ
et <- sqlQuery(channel = 2,
ata(Dataset, placement='-20
ataset)
t(C, alternative='two.sided
t(Dataset$F2, alternative=
ro.test(Dataset$F2)Dataset
ataset)
```

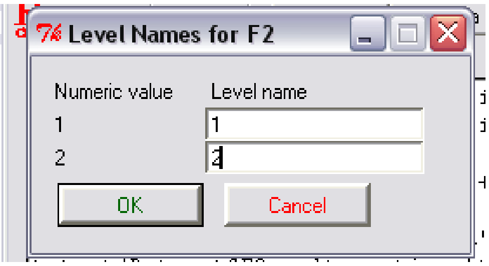


Convert Numeric Variables to Factors

Variables (pick one or more)
F1
F2

Factor Levels
Supply level names ●
Use numbers ○

New variable name or prefix for multiple variables: <same as variables>

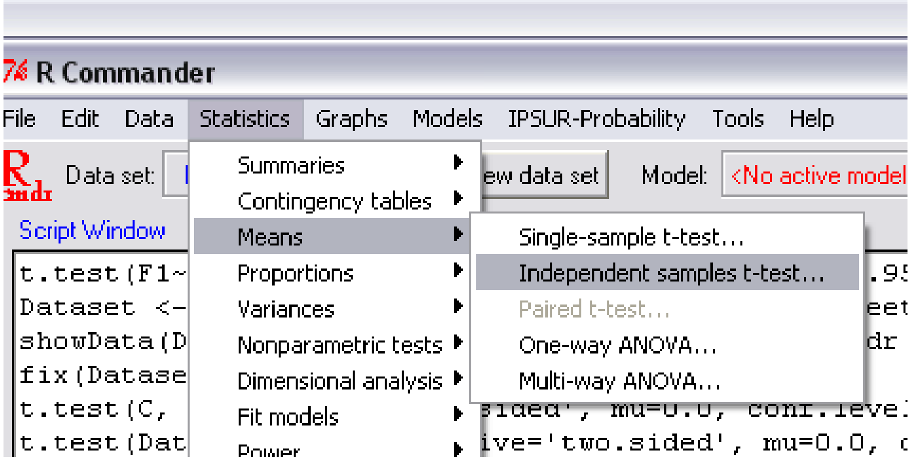


Level Names for F2

Numeric value	Level name
1	1
2	2

4.) Perform test of equality of variances and check p-value (p-value = 0.6421, not shown in this instruction) to determine if equal variances assumption is acceptable. If p-value is greater than 0.05, the equal variances assumption would be acceptable at 5% level of significance.

5.) Click Statistics, Means, Independent sample t-test to perform two independent samples t-test with Assume equal variances Yes bullet checked.

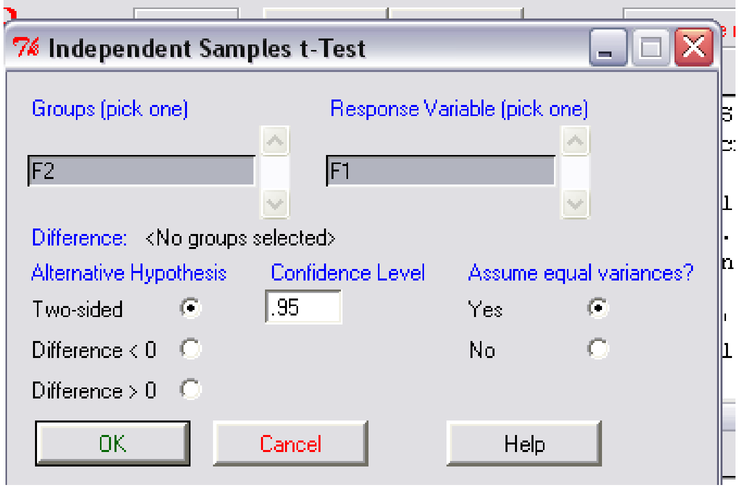


R Commander

Data set:
Model: <No active model

Script Window

```
t.test(F1~
Dataset <-
showData(D
fix(Datase
t.test(C,
t.test(Dat
```



Independent Samples t-Test

Groups (pick one): F2

Response Variable (pick one): F1

Difference: <No groups selected>

Alternative Hypothesis: Two-sided ● Difference < 0 ○ Difference > 0 ○

Confidence Level: .95

Assume equal variances? Yes ● No ○

R Output:

```
Two Sample t-test

data:  var1 by var2
t = -2.6437, df = 20, p-value = 0.01558
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -8.676786 -1.023214
sample estimates:
mean in group 1 mean in group 2
          24.90           29.75
```

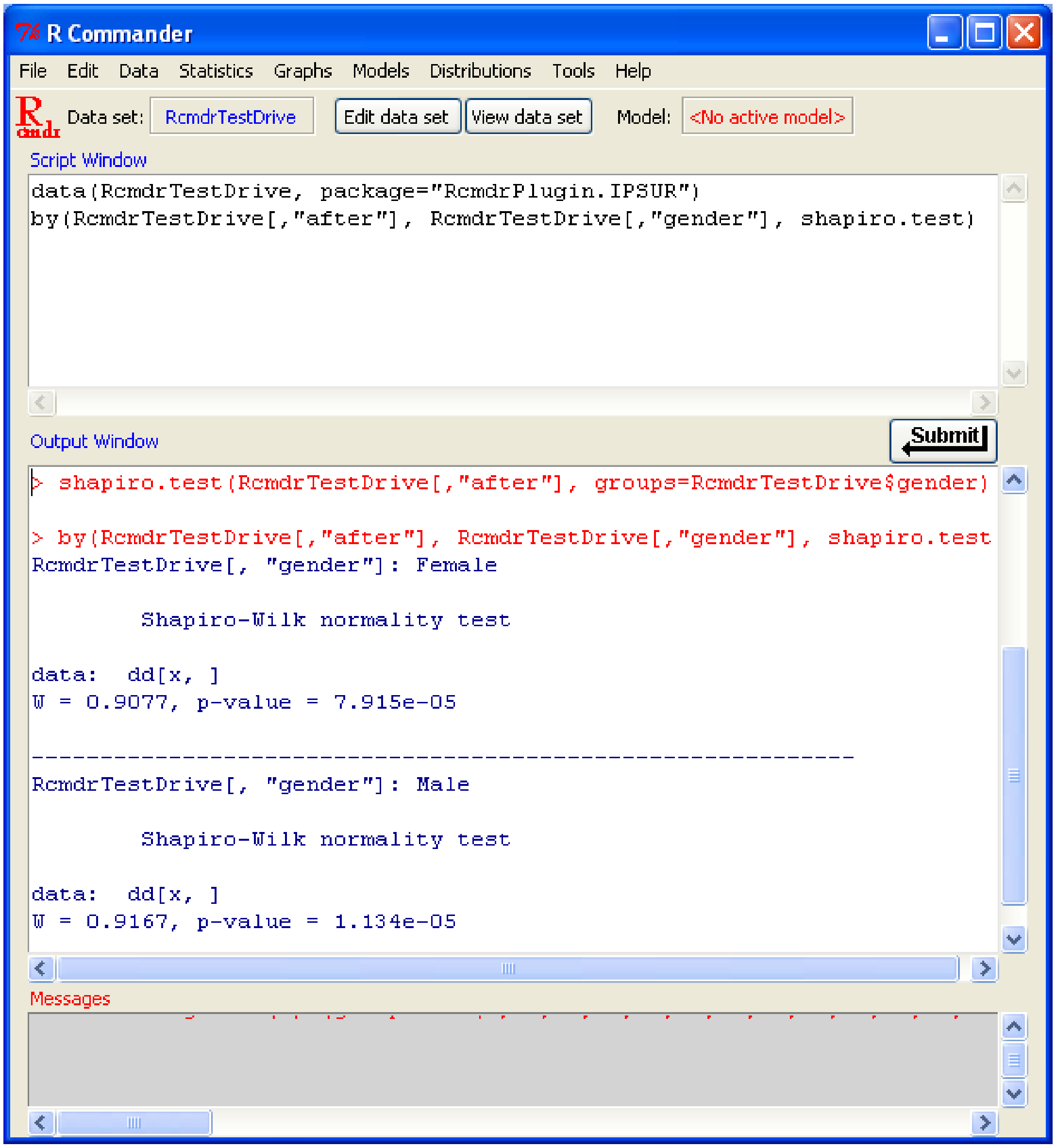
Interpretation: Since p-value = 0.01558 for the test is less than 0.05. One can conclude that the null hypothesis is rejected. There is sufficient evidence to support the alternative hypothesis that the average between the two populations is statistically significantly different.

Normality Tests for Subgroups in a Data File

If you have a quantitative variable **after** (in RcmdrTestDrive file) in a data file and you wish to test for normality for after variable for male and female subjects separately, the following is the R command to do it.

Assume the RcmdrTestDrive has the quantitative variable "after" and the qualitative variable "gender", the R command to do normality test on "after" variable for each gender is using a **by** command as the following: (You may enter this command in the Script Window and click on **Submit** button in the R Commander window)

```
by(RcmdrTestDrive[,"after"], RcmdrTestDrive[,"gender"], shapiro.test)
```



Script Window:

```
data(RcmdrTestDrive, package="RcmdrPlugin.IPSUR")
by(RcmdrTestDrive[,"after"], RcmdrTestDrive[,"gender"], shapiro.test)
```

Output Window:

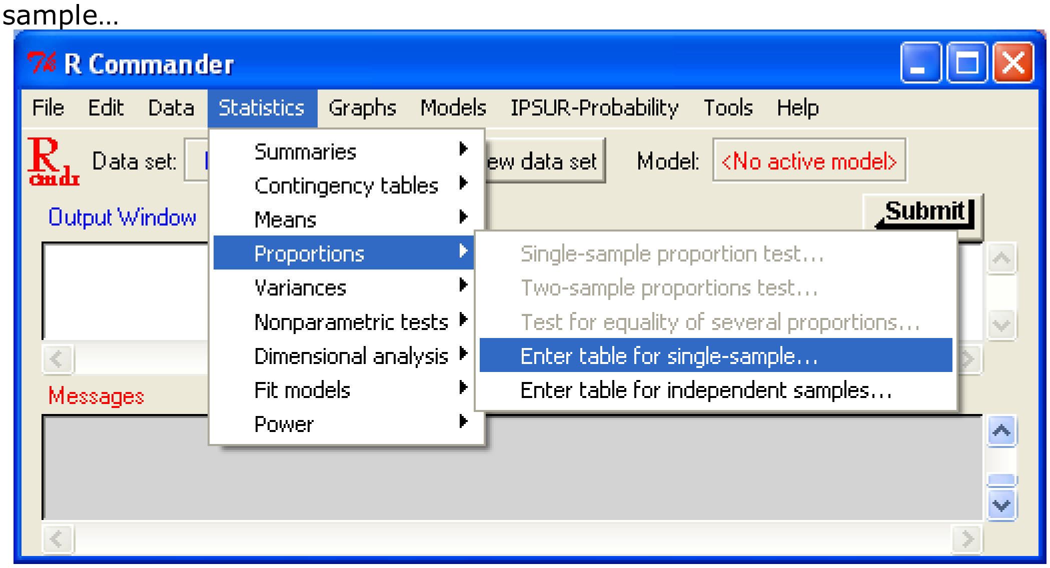
```
> shapiro.test(RcmdrTestDrive[,"after"], groups=RcmdrTestDrive$gender)

> by(RcmdrTestDrive[,"after"], RcmdrTestDrive[,"gender"], shapiro.test
RcmdrTestDrive[, "gender"]: Female

        Shapiro-Wilk normality test

data:  dd[x, ]
W = 0.9077, p-value = 7.915e-05

------------------------------------------------------------
RcmdrTestDrive[, "gender"]: Male

        Shapiro-Wilk normality test

data:  dd[x, ]
W = 0.9167, p-value = 1.134e-05
```

Tests of One Proportion and Equality of Two Proportions

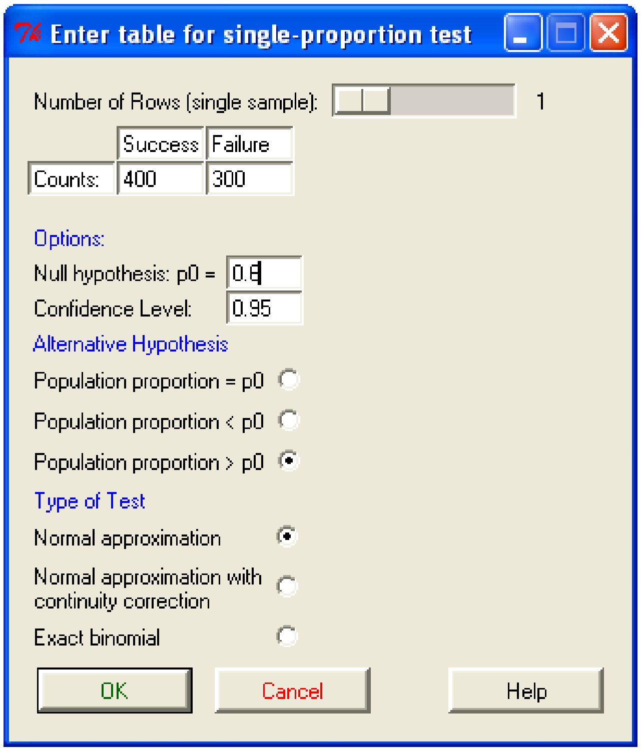
Test of One Proportion

Example: Henning et al. found that 400 of a sample of 700 infants had completed the hepatitis B vaccine series. Can we conclude on the basis of these data that, in the sampled population, **more than 60 percent** have completed the series?

1) In the IPSUR window, click on Statistics/Proportions/Enter table for single-sample…



2) Enter the number of infants who have completed the vaccine series in the Successes box, the total samples minus the number of infants who have completed the series in the failures box, and the probability for which you are testing in the p0 box (0.6 or 60 percent) and the confidence level in the confidence level box. Check the box that says population proportion > p_0 to test whether the proportion is actually greater than 60%.



Enter table for single-proportion test

Number of Rows (single sample): 1

	Success	Failure
Counts:	400	300

Options:

Null hypothesis: p0 = 0.6

Confidence Level: 0.95

Alternative Hypothesis

Population proportion = p0 ○

Population proportion < p0 ○

Population proportion > p0 ●

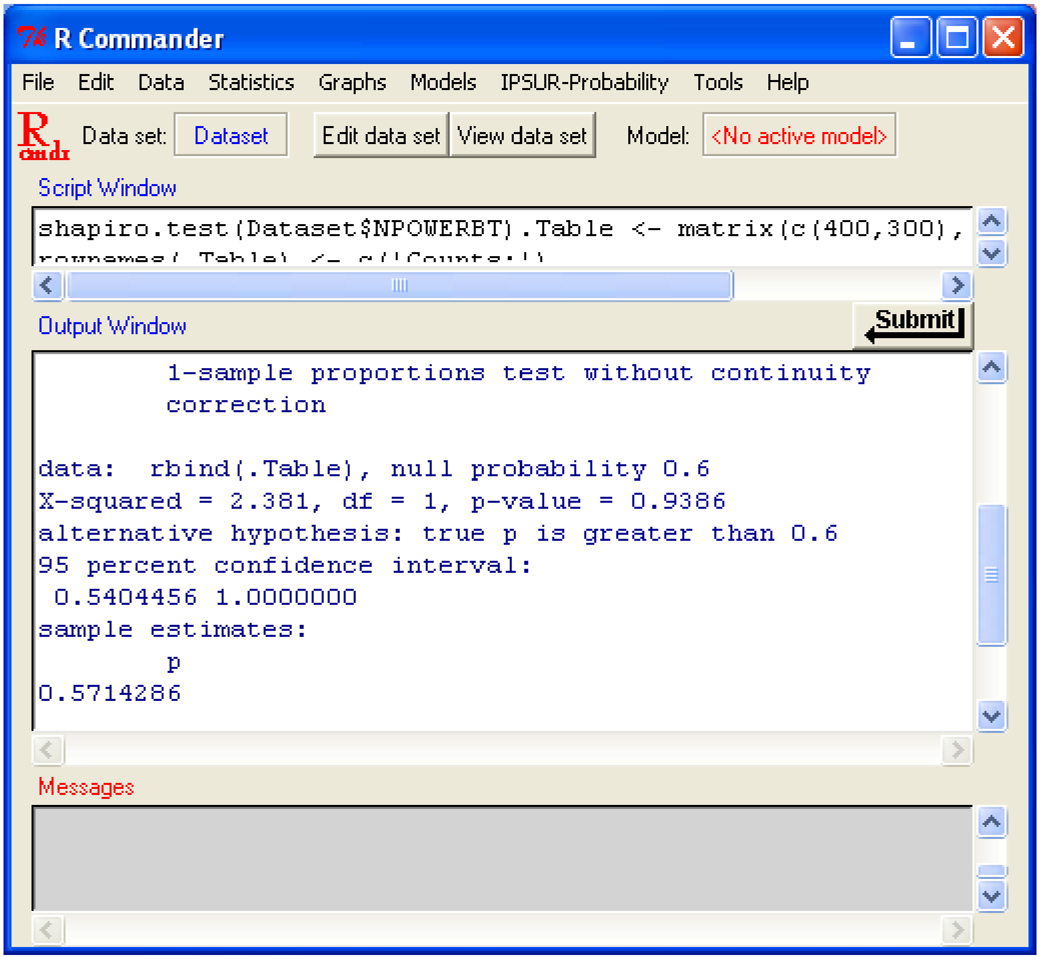
Type of Test

Normal approximation ●

Normal approximation with continuity correction ○

Exact binomial ○

3) Check the p-value in the output window of IPSUR to check whether or not the null hypothesis (that the proportion is equal to 60%) would be rejected.



```
shapiro.test(Dataset$NPOWERBT).Table <- matrix(c(400,300),
rownames(.Table) <- c('Counts:')
```

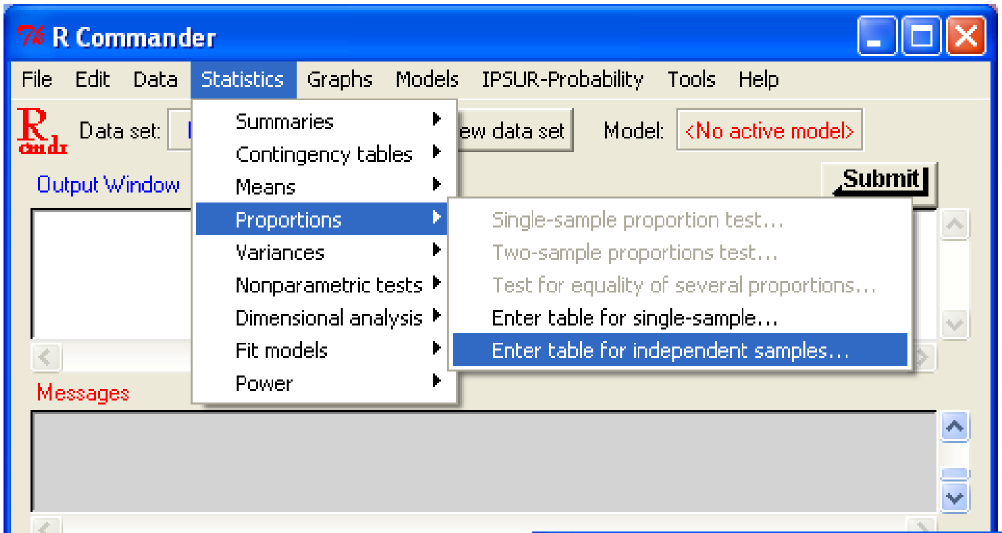
```
        1-sample proportions test without continuity
        correction

data:  rbind(.Table), null probability 0.6
X-squared = 2.381, df = 1, p-value = 0.9386
alternative hypothesis: true p is greater than 0.6
95 percent confidence interval:
 0.5404456 1.0000000
sample estimates:
        p
0.5714286
```

Because the p-value = .9386 is greater than 0.05, we do not reject the null hypothesis and so there is not sufficient evidence to support the alternative (the proportion is greater than 60%). This does not imply that the proportion is 60%.

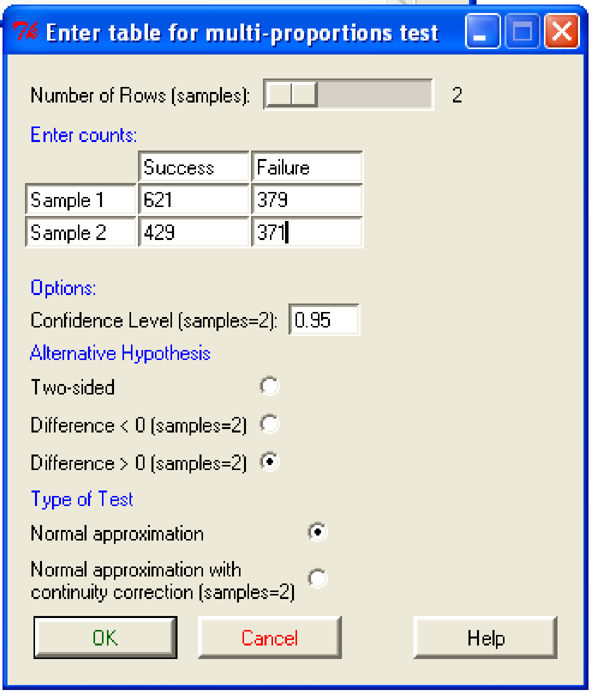
Test of Two Proportions

Example: A research company did a survey on preference of a new policy. They reported that out of 1000 male participants, 621 were in favor of this policy, and out of 900 female participants, 429 were in favor of this policy. Test whether there is a statistically significantly **higher percentage in male population** in favor of the policy than the female population. Use p-value approach to perform the test, at 5% level of significance.

1) In the IPSUR window, click on Statistics/Proportions/Enter table for independent samples...



2) Enter the number of male participants who supported the policy (621) in the sample 1 success box, and the total number of male participants supporting the policy (379) in the failure box. Enter the number of female participants who supported the policy (429) in the sample 2 success box, and the total number of female participants supporting the policy (371) in the failure box. Put the confidence level into the confidence level box. Then set the alternative hypothesis to the Difference > 0 to test whether there are more males in favor than females.



Number of Rows (samples): 2

Enter counts:

	Success	Failure
Sample 1	621	379
Sample 2	429	371

Options:

Confidence Level (samples=2): 0.95

Alternative Hypothesis

Two-sided ○

Difference < 0 (samples=2) ○

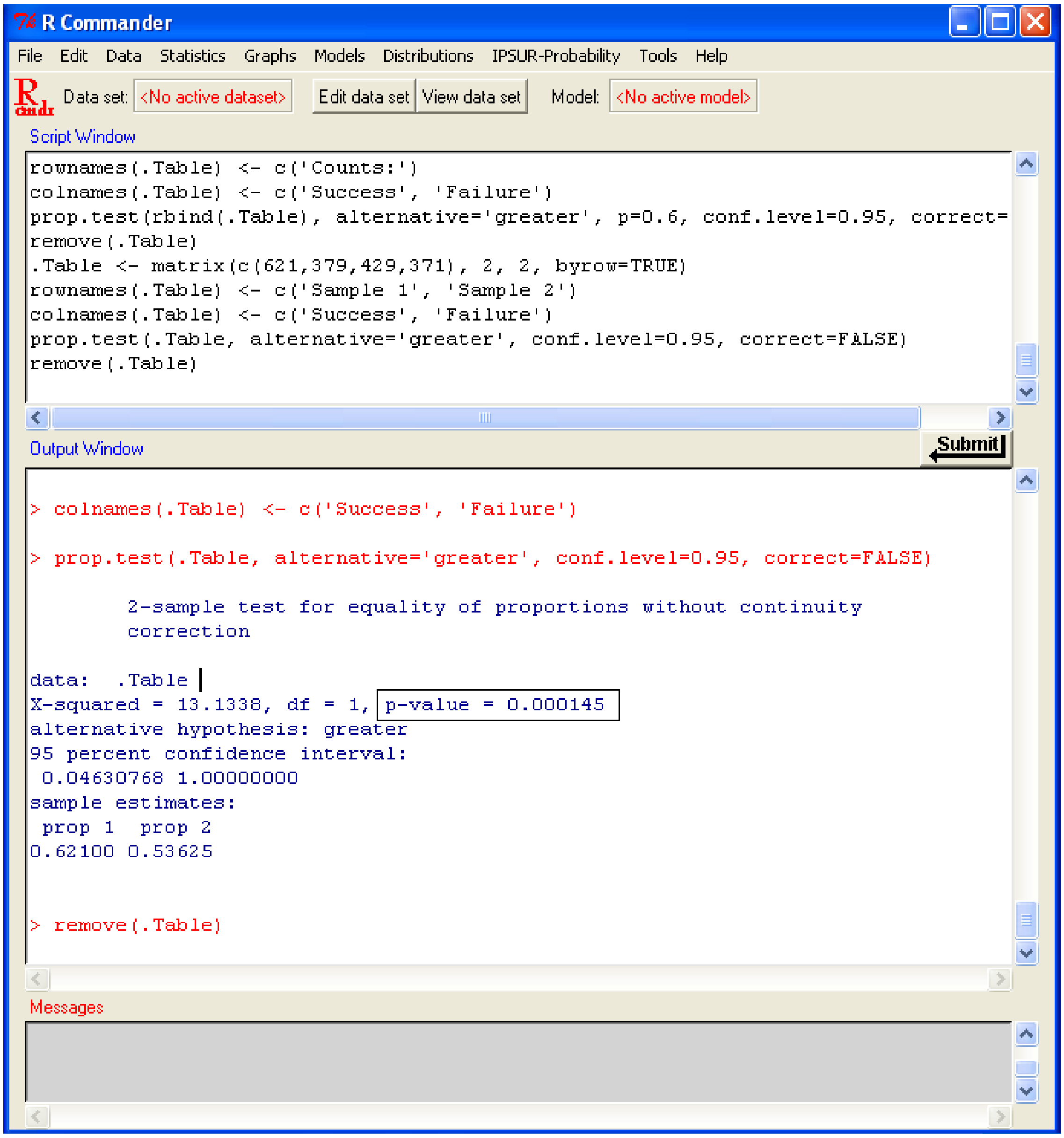
Difference > 0 (samples=2) ●

Type of Test

Normal approximation ●

Normal approximation with continuity correction (samples=2) ○

3) In the output window of IPSUR, the p-value is shown. Because the p-value is less than the level of significance (usually 5%), we reject the null hypothesis (that proportion of males in favor is equal to the proportion of females in favor) and support the alternative (that the proportion of males in favor is larger than the proportion of females in favor).

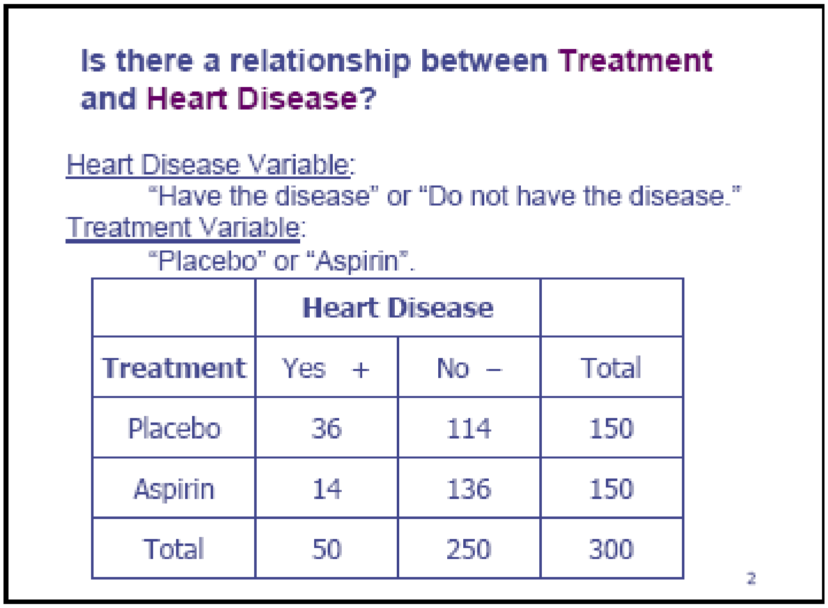


```
rownames(.Table) <- c('Counts:')
colnames(.Table) <- c('Success', 'Failure')
prop.test(rbind(.Table), alternative='greater', p=0.6, conf.level=0.95, correct=
remove(.Table)
.Table <- matrix(c(621,379,429,371), 2, 2, byrow=TRUE)
rownames(.Table) <- c('Sample 1', 'Sample 2')
colnames(.Table) <- c('Success', 'Failure')
prop.test(.Table, alternative='greater', conf.level=0.95, correct=FALSE)
remove(.Table)
```

```
> colnames(.Table) <- c('Success', 'Failure')

> prop.test(.Table, alternative='greater', conf.level=0.95, correct=FALSE)

        2-sample test for equality of proportions without continuity
        correction

data:  .Table
X-squared = 13.1338, df = 1, p-value = 0.000145
alternative hypothesis: greater
95 percent confidence interval:
 0.04630768 1.00000000
sample estimates:
 prop 1  prop 2
0.62100 0.53625


> remove(.Table)
```

Chi-square Test for Independence

For this example, we will perform a Chi-square Test of Independence using data in the following contingency table to see if there is correlation between treatment and outcome.



Is there a relationship between Treatment and Heart Disease?

Heart Disease Variable:
"Have the disease" or "Do not have the disease."
Treatment Variable:
"Placebo" or "Aspirin".

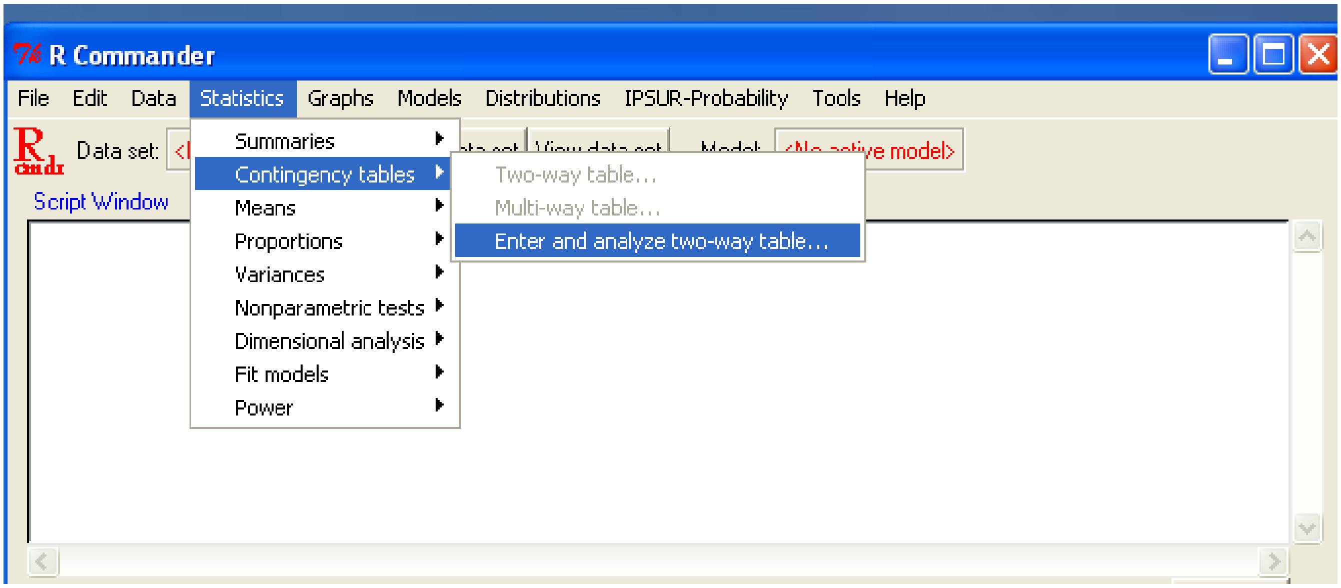
	Heart Disease		
Treatment	Yes +	No –	Total
Placebo	36	114	150
Aspirin	14	136	150
Total	50	250	300

2

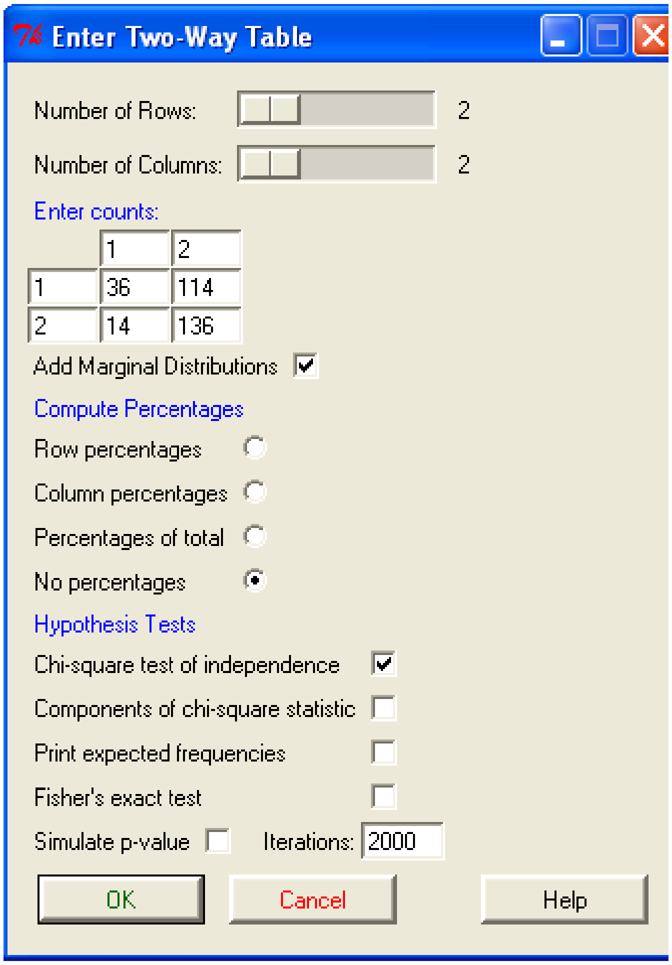
1) To create a Table to represent the "Yes +" and "No –"

Statistics -> Contingency tables

-> Enter and analyze two-way table…



2) Enter the Data from the above chart to match the table in the R Commander window as shown below:



Number of Rows: 2

Number of Columns: 2

Enter counts:

	1	2
1	36	114
2	14	136

Add Marginal Distributions ☑

Compute Percentages

Row percentages ○

Column percentages ○

Percentages of total ○

No percentages ●

Hypothesis Tests

Chi-square test of independence ☑

Components of chi-square statistic □

Print expected frequencies □

Fisher's exact test □

Simulate p-value □ Iterations: 2000

<u>Compute Percentages:</u>
If one needs the percentages, it can be calculated by selecting the bullet next to the needed percentage. For example, No percentages is selected.

<u>Hypothesis Tests</u>:
Choose: Chi-square test of independence

And click OK button.

Interpret Results:

```
        Pearson's Chi-squared test

data:  .Table
X-squared = 11.616, df = 1, p-value = 0.0006539
```

For this example: The resultant is a Chi-squared statistic = 11.616 and a p-value of 0.00065.

To test the hypothesis at 5% level of significance:

<u>P-value Approach</u>:

One would reject Null Hypothesis since the p-value: 0.00065 < 0.05

<u>Critical Value Approach</u>:

Reject null hypothesis since Chi-squared: 11.616 > 3.84

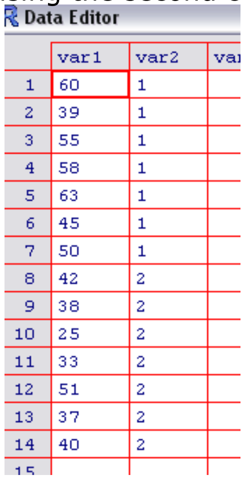
Test of Equality of Variances

When testing two independent samples (for differences in mean, one-sided tests, etc.), it is helpful to know that the samples have equal variances or not. In order to test this condition, a test of equality of variances must be done.

Example: At a 0.05 level of significance, test whether the average lifespan (in months) of aluminum bedpans is statistically significantly different from that of stainless steel bedpans by using the following data:

Lifespan (in months):
Aluminum: 60, 39, 55, 58, 63, 45, 50
Stainless Steel: 42, 38, 25, 33, 51, 37, 40

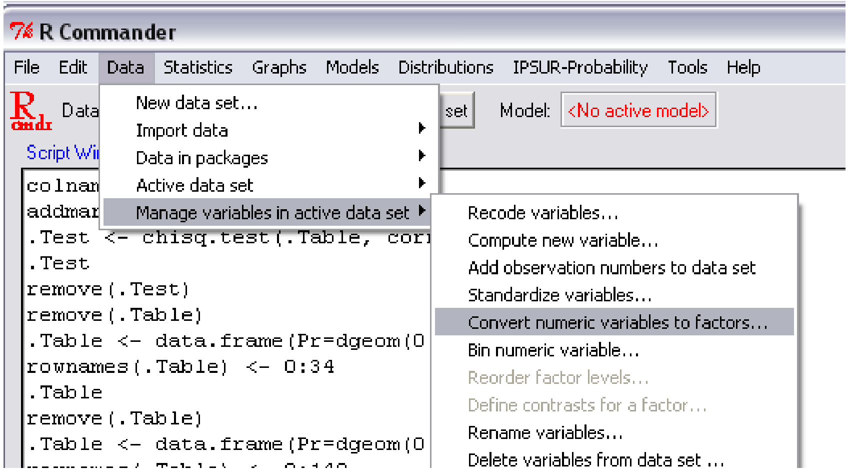
1) Create a new data set in R, and input all the lifespan values in the first column (var1), and separate the data into 2 groups by using the second column (var2) as a indicator or group variable (1 and 2):



Data Editor

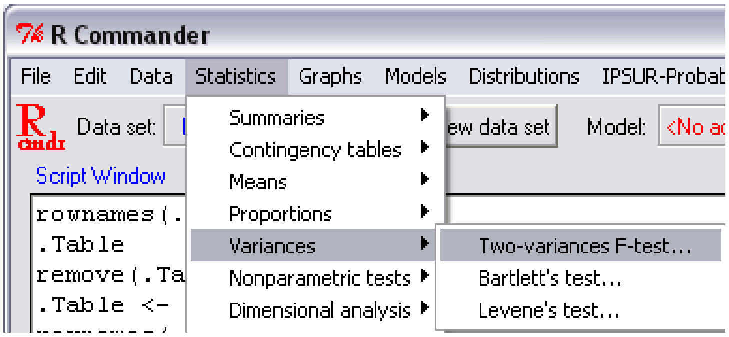
	var1	var2
1	60	1
2	39	1
3	55	1
4	58	1
5	63	1
6	45	1
7	50	1
8	42	2
9	38	2
10	25	2
11	33	2
12	51	2
13	37	2
14	40	2
15		

2) In IPSUR, select **Data / Manage variables in active data sheet / Convert numeric variables to factors…** (as shown in above right). Select the option to change "var2," and rename [1] as Aluminum and [2] as Stainless, to set var2 as a factor.

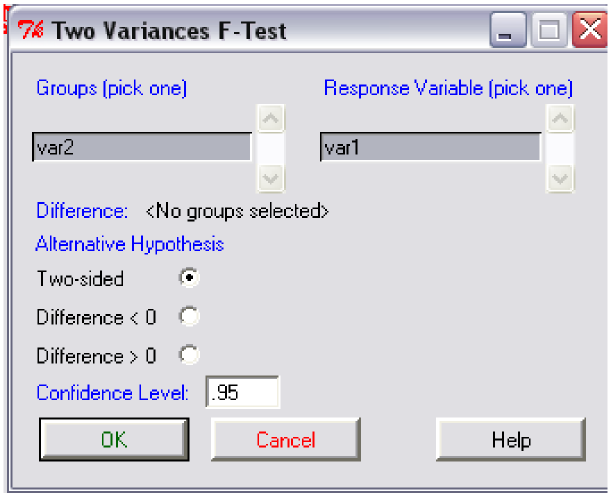


```
colnam
addmar
.Test <- chisq.test(.Table, corr
.Test
remove(.Test)
remove(.Table)
.Table <- data.frame(Pr=dgeom(0
rownames(.Table) <- 0:34
.Table
remove(.Table)
.Table <- data.frame(Pr=dgeom(0
```

3) In IPSUR, select **Statistics / Variances / Two-variances F-test…** (shown below left).



For Groups, choose var2, and the Response Variable is var1. Choose a two-sided test because we want to know if the variances can be assumed equal or not, and 0.95 for Confidence Level is confidence interval is needed.



Groups (pick one): var2

Response Variable (pick one): var1

Difference: <No groups selected>

Alternative Hypothesis

Two-sided ●

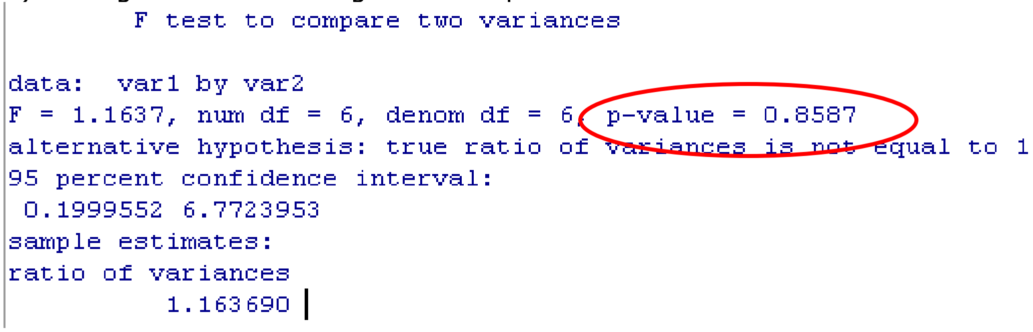
Difference < 0 ○

Difference > 0 ○

Confidence Level: .95

Interpret:

4) This gives the following in the output window of IPSUR:



```
        F test to compare two variances

data:  var1 by var2
F = 1.1637, num df = 6, denom df = 6, p-value = 0.8587
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.1999552 6.7723953
sample estimates:
ratio of variances
          1.163690
```

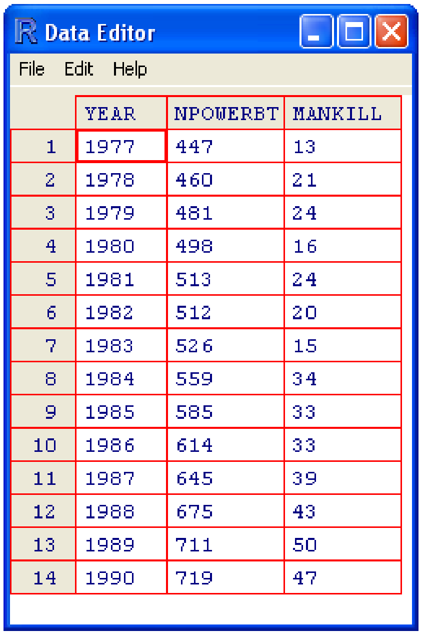
Since the p-value is 0.8587, and if the level of significance is 5%, and 0.8587 > 0.05, the null hypothesis of equal variances is not rejected due to sufficient evidence.

In other words, this evidence cannot reject the null hypothesis that $\frac{\sigma_1^{\ 2}}{\sigma_2^{\ 2}} = 1$

Correlation Test & Linear Regression

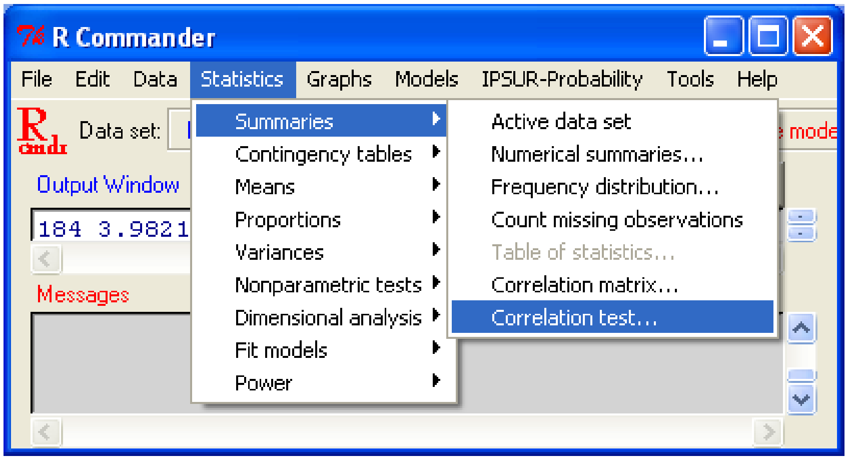
Test of Correlation (Testing for $\rho = 0$)

1) First, enter your data in R commander into R by selecting **Data, New data.**



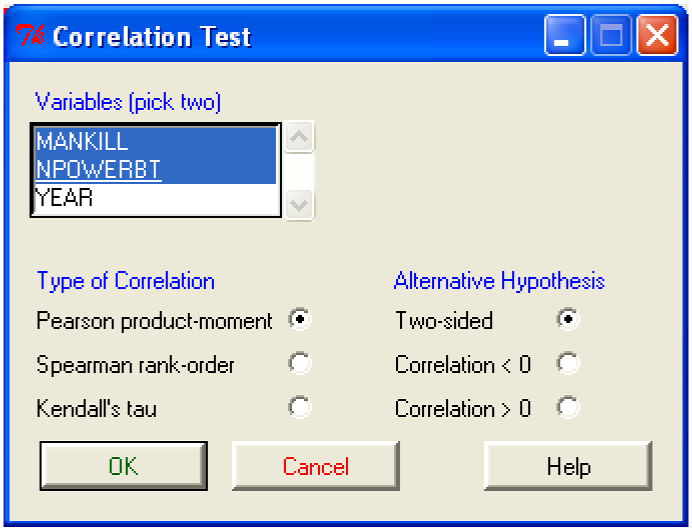
	YEAR	NPOWERBT	MANKILL
1	1977	447	13
2	1978	460	21
3	1979	481	24
4	1980	498	16
5	1981	513	24
6	1982	512	20
7	1983	526	15
8	1984	559	34
9	1985	585	33
10	1986	614	33
11	1987	645	39
12	1988	675	43
13	1989	711	50
14	1990	719	47

2) In R commander, select Statistics, Summaries, and select Correlation test as show in the following figure.



```
184 3.9821
```

3) In the correlation dialog box, click and drag mouse to select the two variables, MANKILL and NPOWERBT for computing the correlation, and have the Pearson product-moment bullet checked, and click OK.



Variables (pick two)

MANKILL
NPOWERBT
YEAR

Type of Correlation

Pearson product-moment ●
Spearman rank-order ○
Kendall's tau ○

Alternative Hypothesis

Two-sided ●
Correlation < 0 ○
Correlation > 0 ○

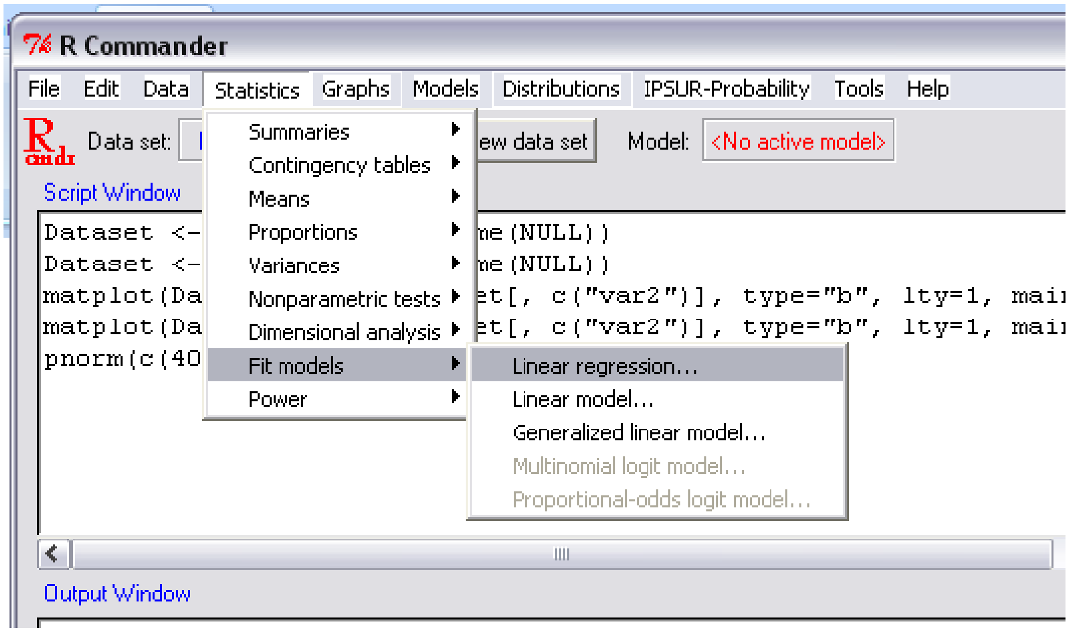
Interpret:

4) The p-value of the test is p-value = 5.109e-07 which is less than .05, so we can conclude that the correlation is statistically significant different from 0 at 5% less of significance.

```
Pearson's product-moment correlation

data:  Dataset$MANKILL and Dataset$NPOWERBT
t = 9.6755, df = 12, p-value = 5.109e-07
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8210214 0.9816797
sample estimates:
      cor
0.9414773
```

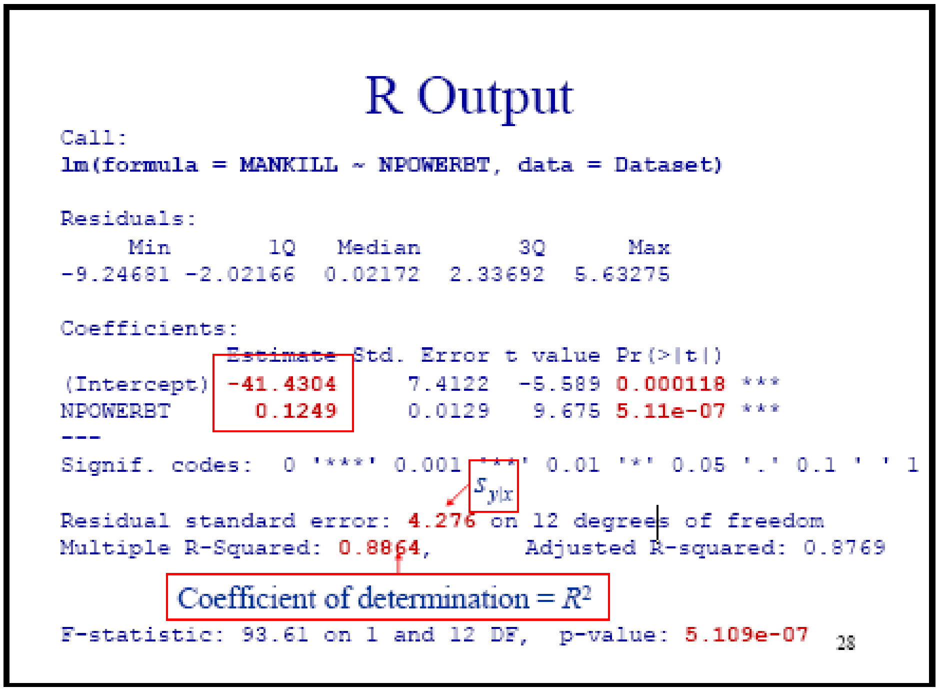
Linear Regression: (Using the same data in the Correlation Test)

From IPSUR, select Statistics, Fit Models, and then select Linear Regression.



```
Dataset <-                                me(NULL))
Dataset <-                                me(NULL))
matplot(Da                                et[, c("var2")], type="b", lty=1, mai
matplot(Da                                et[, c("var2")], type="b", lty=1, mai
pnorm(c(40
```

The following is the R Output.



R Output

```
Call:
lm(formula = MANKILL ~ NPOWERBT, data = Dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-9.24681 -2.02166  0.02172  2.33692  5.63275

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -41.4304     7.4122  -5.589 0.000118 ***
NPOWERBT      0.1249     0.0129   9.675 5.11e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.276 on 12 degrees of freedom
Multiple R-Squared: 0.8864,     Adjusted R-squared: 0.8769
```

$S_{y|x}$

Coefficient of determination = R^2

```
F-statistic: 93.61 on 1 and 12 DF,  p-value: 5.109e-07
```

28

Interpret:

The Intercept (p-value = .000118) and the beta coefficient (p-value = 5.11e-07) are both significantly different from zero. The following equation is used to find the equation of the regression line, the variables ɑ and β are highlighted above.

Equation of the regression line:

$$\hat{y} = \hat{\alpha} + \hat{\beta} \cdot x;$$

$$\hat{y} = -41.4304 + .1249 \cdot x$$

Example: If at a certain year the number of power boats registered is 700, estimate how many manatees on average would be killed.

To solve this example simply plug 700 into the equation that was found above.

$$\begin{aligned}\hat{y} &= -41.430439 + .124862 \cdot x \\ &= -41.430439 + .124862 \cdot 700 \\ &= 45.973\end{aligned}$$

The average response at $x = 700$ is 45.973.