Notes on Building an R Package for Windows

Yiwen Chen
NRCan Contract Work Report
March, 08, 2007

R was created with and for UNIX. That is why building an R package is a challenge for Windows users. A UNIX environment has to be set up within Windows. That requires installing the essential software tools, and setting up them correctly, on the Windows computer.

This paper records the whole procedure I undertook to make the rgr package, the GSC[1] Applied Geochemistry EDA[2] Package, with Windows 2000 and R 2.4.1. The rgr package contains 40 R functions written by Dr. Robert G. Garrett of Natural Resources Canada (NRCan).

Follow the steps listed below, you will find that building R packages for Windows is not as difficult as you thought. I also used these steps on Windows 2000 with R 2.3.0, Windows XP with R 2.2.1 and R 2.4.1, all worked successfully.

Step One: Download and Install Essential Software Tools

1. Unix Tools: Rtools , cygwin and hhc.exe
   You can get these tools freely from:
   http://www.murdoch-sutherland.com/Rtools/tools.zip
   http://www.maths.bris.ac.uk/~maman/computerstuff/Rhelp/cygwin.zip
   http://www.maths.bris.ac.uk/~maman/computerstuff/Rhelp/hhc.exe

   Create a subdirectory “Rtools” in “c:/” and extract “tools.zip” to “c:/Rtools”.

   Unzip “cygwin.zip” and save it in the C directory “c:/”.

   You may encounter a “Can’t find chm files” problem which is because “R CMD build” does not know where hhc.exe is. So I recommend installing hhc.exe in “c:/cygwin”.

2. Perl
   From http://www.activestate.com/Products/ActivePerl/Download.html, select MSI (Windows installer package), download Perl freely and save into: “ c:/ ”. Perl 5.8.4 or later are all suitable. I installed ActivePerl 5.8.8 Build 819 to build rgr.

[1] Geological Survey of Canada
[2] Exploratory Data Analysis

3. Html Help
   Html Help is the html compiler for building compiled Windows html help files. You can freely download it from:
   http://msdn.microsoft.com/library/enus/htmlhelp/html/hwmicrosofthtmlhelpdownloads.asp and save it to "c:/program files".

4. MikTex
   MikTex 2.5 was used to build the documentation in the package. You can download the "Basic MikTex 2.5 Installer" freely from:
   http://www.miktex.org/2.5/Setup.aspx and follow the MikTex Setup Wizard to install it in c:/program files.

   Package "lm" is required to check manual(s) created in the R package. It may not be included in MikTex Basic. If not, you can get it as follows:

   Start "MikTex Options" from menu "Start→Programs→MikTex2.5→Settings"

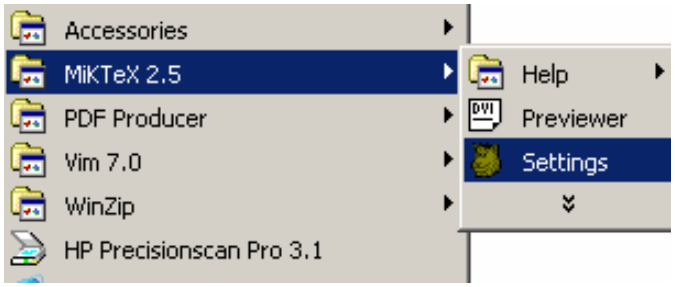

Figure 1 Open MikTex Options

Under the "General" tab, set "Install missing packages on-the-fly" to "yes".

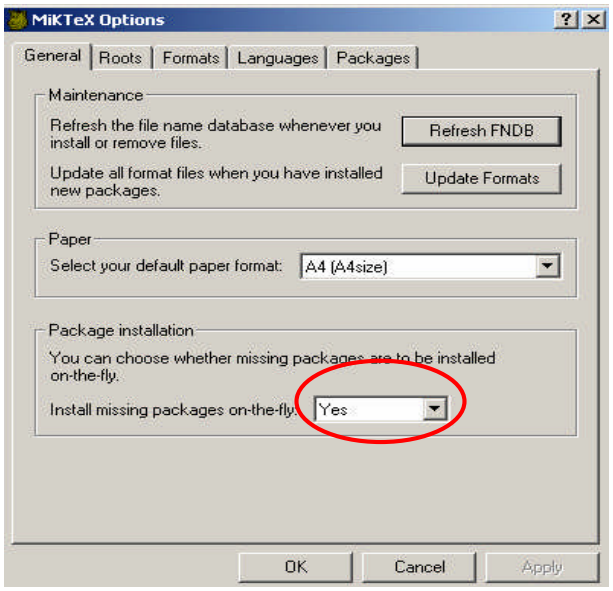

Figure 2 Set Install Missing Packages On-the-fly On

Under the “Packages” tab, click the “Change” button to change the package repository.

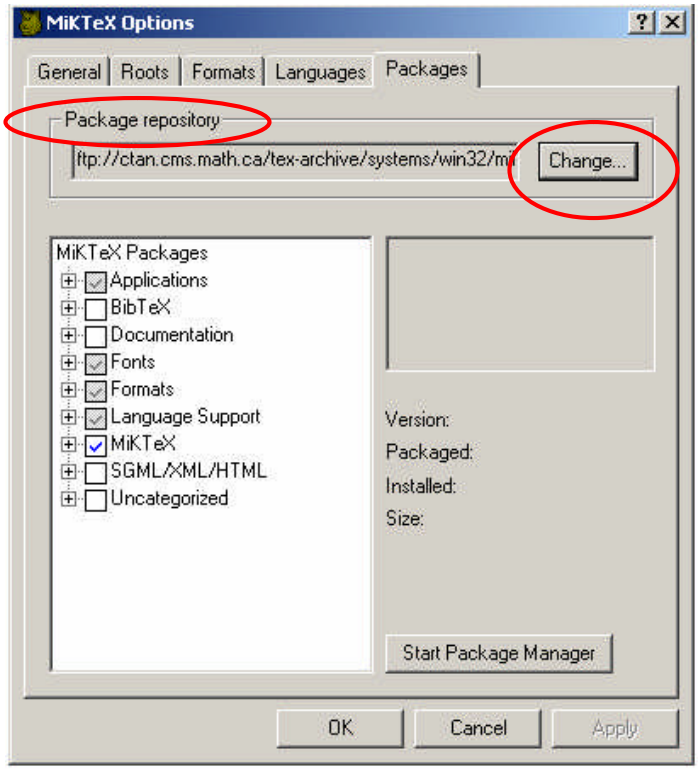

Figure 3 Change the Package Repository

Check on the “Packages shall be installed from the Internet” box and then click “Next”.

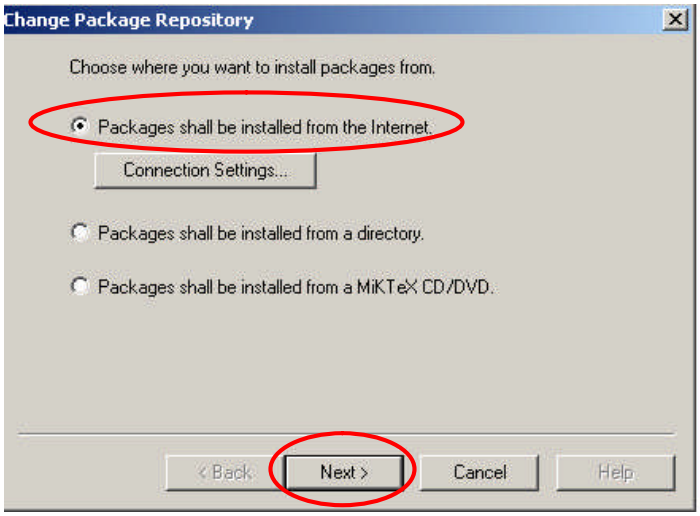

Figure 4 Select to Install Missing Packages from Internet

Select a download mirror near to your location and then click “Finish”.

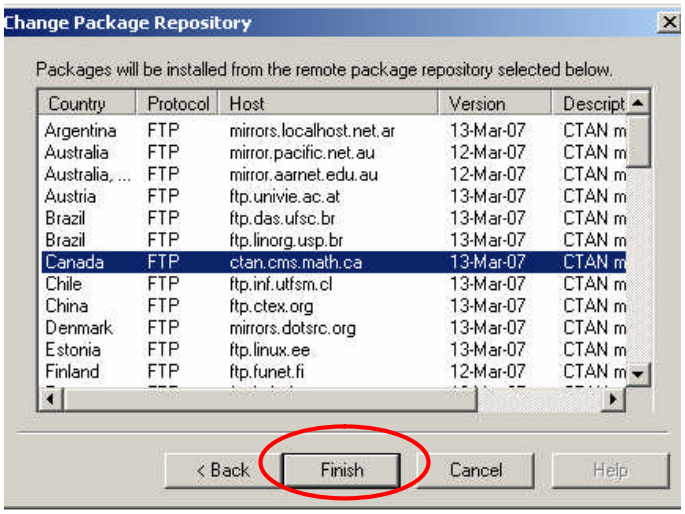

Figure 5 Select a Package Repository

It will take several minutes to download the package repository.

Then, check on the box “lm” in “Fonts→Type 1 Fonts” and click “Apply” to install package “lm”.

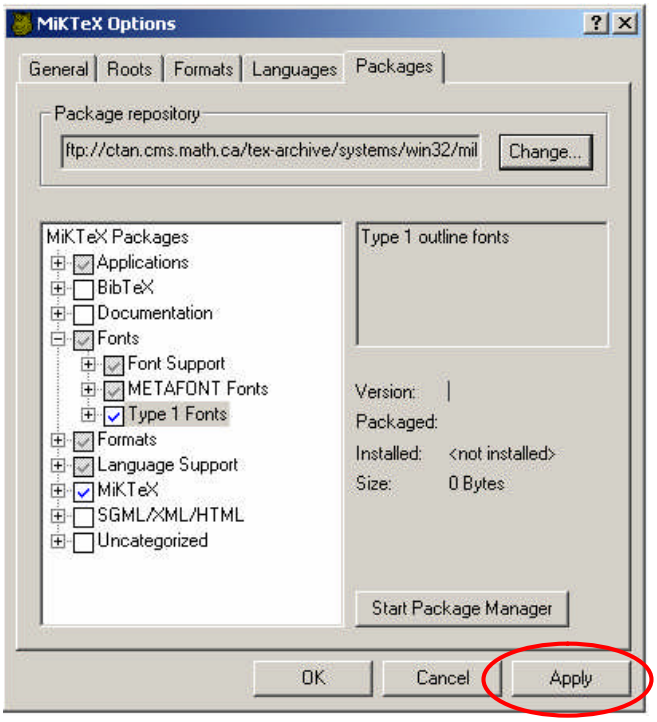

Figure 6 Select Packages to Install

5. Tcl/Tk Support

   From http://www.murdoch-sutherland.com/Rtools/ download "R_Tcl.zip". Then create a subdirectory e.g. "R_Tcl" in "c:/program files/" and extract the zip file into this directory.

6. Bitmap graphics devices
   Download jpegsrc.v6b.tar.gz from ftp://ftp.uu.net/graphics/jpeg/ and libpng-1.2.15-no-config.tar.gz from http://www.libpng.org/pub/png/libpng.html freely and extract them into "c:/program files", where their contents will be present as new folders (subdirectories).

7. Inno Setup
   Inno Setup 5.1.7 or later is required to build the installers in R 2.2.0 or later. You can get it from http://jrsoftware.org/isdl.php#stable freely and follow the setup wizard to install it in "c:/program files". I installed version 5.1.9.

Step Two: Set System Variables Path

1. Open the "System Properties" window
   To do so, you can go to the "Start" menu and select "Control Panel" then "System"
   OR
   right click "My Computer" and select "Properties"

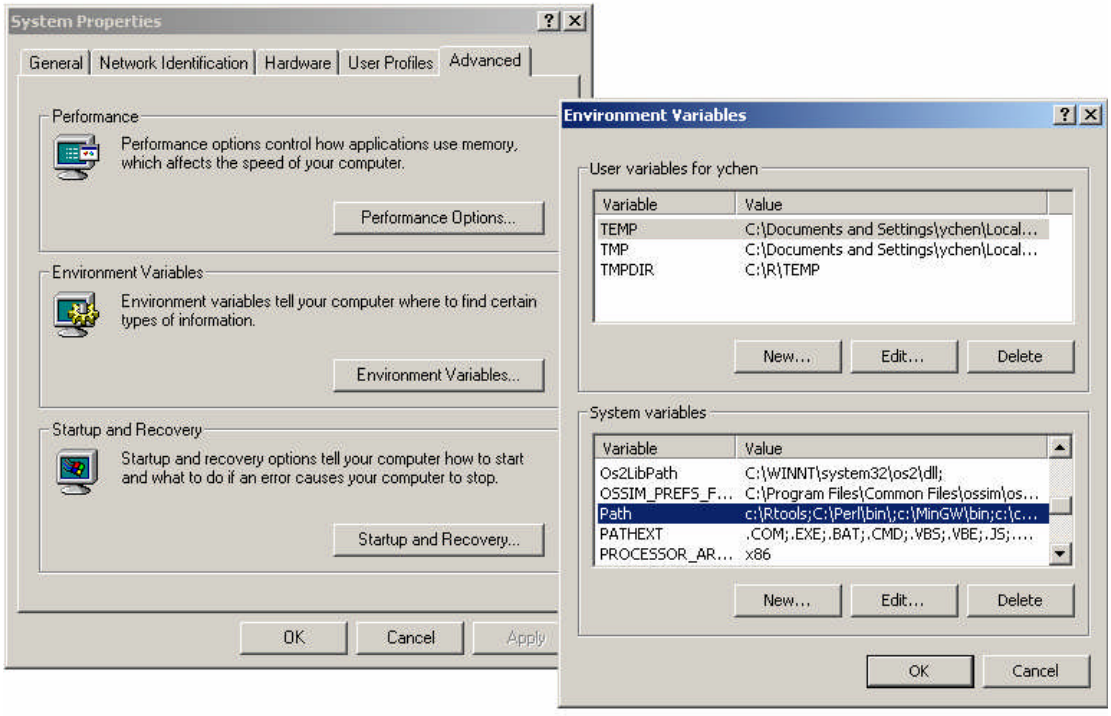

Figure 7 Set "System variables" Path

2. Edit the "System variables " path
   Under the "Advanced" tab, click "Environment Variables" to open the "Environment Variables" window. Then in this window, under the "System variables" tab, select "path" then click "edit" button to edit the path.

   You should set your path to have Rtools, Perl, MikTex, R and the html help compiler at the beginning.

```
c:\Rtools\tools\bin;c:\Perl\bin\;c:\cygwin;c:\Program Files\MikTex2.5\miktex\bin;
 c:\ Program Files\R\R-2.4.1\bin\;c:\ Program Files \HTMLHE~1\;…
```

Figure 8 Example Path

Notes:

- There is no space after semicolons
- Path designations are NOT case-sensitive under Windows
- Rtools, Perl and MikTex have the binary or executable (.exe) files stored in the "bin" directories. Therefore, do remember to have these "bin" subdirectories in your path (Html Help is an exception).

3. Check the "System variables" path setting
   Open the command window and check carefully whether your path has been set correctly. If you find an error, go back to "System variables" to edit your path again.

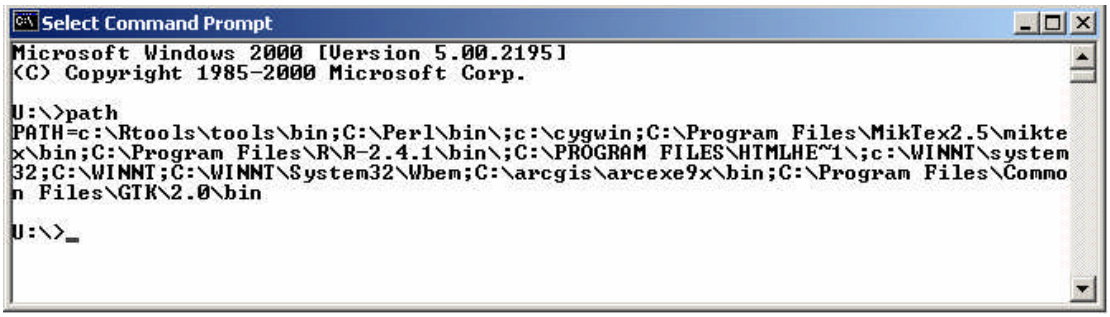

Figure 9 Check "System variables" Path Setting

Restart the command window and type "R" to invoke "Rterm", the command prompt version of R. If you see the usual R welcome message and the prompt ">", the "R.exe" file has been set correctly in your path. Otherwise, you may see "R is not a recognized as an internal or external command…", in which case check and edit the path again . If R starts, quit R by typing "`quit()`" or "`q()`" at ">".

At the command window type

```
perl --help
Tex --help
```

You should get a long list of options for each of these two commands. If you get a fail for any of these checks, check and edit your path again.

Step Three: Set a Valid Temporary Directory

Create a directory, e.g. C:\RTEMP, as the valid temporary directory. At the command window, type:

```
set TMPDIR=c:\RTEMP
```

Step Four: Construct the Package

Use R function “package.skeleton” to create a directory tree, e.g.,

```
package.skeleton(list=c("shape","display.lty","display.marks",
      "gx.hist","bxplot","gx.ecdf","cnpplt","kola.o","remove.na",
      "inset","inset.exporter","gx.stats","kola.c","ltdl.fix.df"),
      name="rgr2a", path="u:/0202"))
```

In the above example:

list --- R objects to be put in the package
name --- name of your package
path --- path to put your package directory tree in

The directory tree will include top directory files “DESCRIPTION” and “Read-and-delete-me” and subdirectories “data”, “man” and “R”.

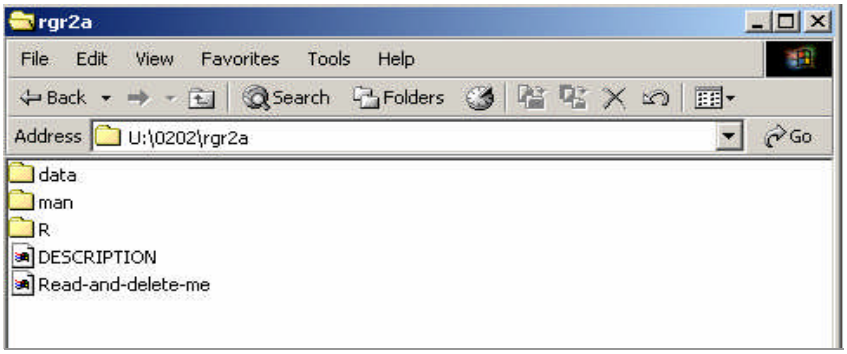

Figure 10 Example Package Directory Tree

“DESCRIPTION” --- a template for the DESCRIPTION file to be included in the package;
“data” --- contains the data sets to be provided with the package;
“man” --- contains the templates of .Rd files for each function in the package;
“R” --- contains the R source code of each function in the package.

If you would like to add PDF documents into your package, create a directory “inst” in your package and a subdirectory “doc” in “inst”. Then put your PDF files in the subdirectory “doc”.

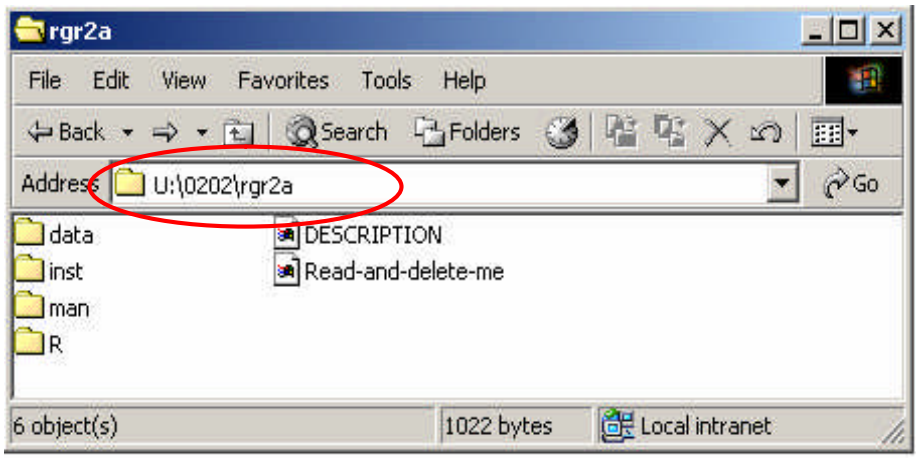

Figure 11 Create the Directory "inst" in the Package

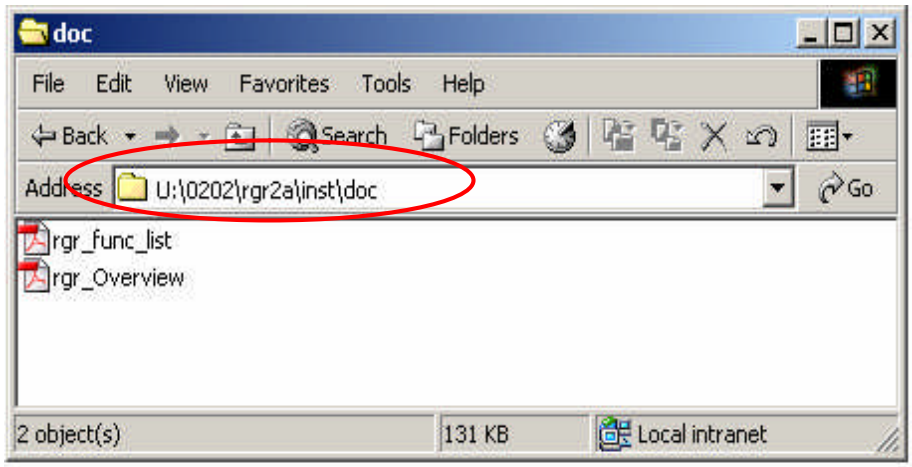

Figure 12 Add PDF Documents in the Package

Step Five: Edit Files in the Package

You can edit your files in a text editor such as VIM which can be freely downloaded from http://www.vim.org

For instructions on how to edit files in your package see the R manual, Writing R Extensions, section 1.1 and chapter 2. Here I copied the documentation in package rgr as examples to help you modify your files.

1. Edit "DESCRIPTION" file

"DESCRIPTION" file contains the basic information about the package written as ASCII text. This gives a brief description of what the package does as well as

information about the package such as title, release date and other packages which this package depends on.

Here is the copy of the "DESCRIPTION" file of package rgr:

```
Package: rgr
Type: Package
Title:The GSC (Geological Survey of Canada) Applied Geochemistry EDA Package
Version: 1.0
Author: Robert G. Garrett
Maintainer: Robert G. Garrett <garrett@NRCan.gc.ca>
Depends: akima, MASS
Description: R functions for Exploratory Data Analysis with applied geochemical data
License: GPL
URL: http://gsc.nrcan.gc.ca/dir/index_e.php?id=4961
```

Figure 13 Example DESCRIPTION File

2. Create and Edit "NAMESPACE" file

The "NAMESPACE" file specifies which variable(s) and function(s) are available to package users, and which variables(s) and function(s) should be imported from other packages. The package directory tree generated by "package.skeleton" doesn't contain this file. You can create it by copying a "NAMESPACE" file from any R package installed on your computer and edit it. Below is a copy of the "NAMESPACE" file of package rgr.

```
export(shape, display.lty, display.marks, gx.hist, bxplot, gx.ecdf, cnpplt, remove.na,
        inset, inset.exporter,  gx.stats, ltdl.fix, ltdl.fix.df, bwplot, bwplot.by.var,
       tbplot, tbplot.by.var, cat2list, var2fact, caplot,  edamap, edamap7, edamap8,
       syms.pfunc, syms,  cutter, gx.subset, display.rainbow, fences, fences.summary,
       framework.stats, framework.summary, anova1, anova2, dftest,  display.alts,
       display.ascii.d, display.ascii.o, thplot1,  thplot2)
```

Figure 14 Example NAMESPACE File

3. Edit .Rd files in the "man" directory

Here is a copy of “framwork.summary.Rd” in package rgr edited with VIM 7.0.

```
\name{framework.summary}
\alias{framework.summary}
\title{ Generate and Save Framework/Subset Summary Statistics }
\description{
Function to generate \sQuote{framework} or subset summary statistics and save them in as a .csv file that can be directly imp
orted into a spreadsheet, e.g., MS Excel®, for inspection or other software, for example, a Geographical Information System (
GIS) where the spatial information concerning the \sQuote{framework} units is available, e.g., ecoclassification units.
}
\usage{
framework.summary(group, x, file = NULL)
}
\arguments{
  \item{group}{ the name of the factor variable by which the data are to be subset. }
  \item{x}{ name of the variable to be processed. }
  \item{file}{ the folder and first part of the file name for saving the function output. }
}
\details{
\code{file} contains the folder name where the output file is to be saved and the first part of the unique file name that wil
l be created and used to store the .csv file of results. The function concatenates this with an \code{_}, \code{group} as a c
haracter string, another \code{_}, and \code{x} as a character string. Subsequently the suffix \code{.csv} is appended and fi
le saved in the folder indicated.

Output to the current device is suppressed. The output file can be inspected with spread sheet software or a viewer of the us
er's choice.
}
\author{ Robert G. Garrett }
\note{
Any less than detection limit values represented by negative values or zeros or other numeric codes representing blanks in th
e data vector must be removed prior to executing this function, see \code{\link{ltdl.fix.df}}.

Any \code{NA}s in the data vector are counted and then removed prior to computing the summary statistics.

The function \code{\link{framework.stats}} is employed to compute the summary statistics.
}
\seealso{ \code{\link{framework.stats}}, \code{\link{ltdl.fix.df}}, \code{\link{remove.na}} }
\examples{
## Make test data available
data(kola.c)
attach(kola.c)

## Saves the file c:\temp\kola_c_summary_COUNTRY_Cu.csv for later use.
## Note the necessity of using double back-slashes in the file name.
framework.summary(COUNTRY, Cu, file = "c:\\\temp\\\kola_c_summary")

## Detach test data
detach(kola.c)
}
\keyword{ univar }
```

Figure15 Example .Rd file

Step Six: Check the Package

At the command window, change to the directory level above the start of the package, type "R CMD check packagename" to check your package. For example, package rgr is in "u:/0312":

```
U:\>cd 0312

U:\0312>R CMD check rgr
* checking for working latex ... OK
* using log directory 'U:/0312/rgr.Rcheck'
* using R version 2.4.1 (2006-12-18)
* checking for file 'rgr/DESCRIPTION' ... OK
* checking extension type ... Package
* this is package 'rgr' version '1.0'
* checking package dependencies ... OK
* checking if this is a source package ... OK
* checking whether package 'rgr' can be installed ... OK
* checking package directory ... OK
* checking for portable file names ... OK
* checking DESCRIPTION meta-information ... OK
* checking top-level files ... OK
* checking index information ... OK
* checking package subdirectories ... OK
* checking R files for syntax errors ... OK
* checking R files for non-ASCII characters ... OK
* checking whether the package can be loaded ... OK
* checking whether the package can be loaded with stated dependencies ...
OK
* checking whether the name space can be loaded with stated dependencies
... OK
* checking S3 generic/method consistency ... OK
* checking replacement functions ... OK
* checking foreign function calls ... OK
* checking R code for possible problems ... OK
* checking Rd files ... OK
* checking Rd cross-references ... OK
* checking for missing documentation entries ... OK
* checking for code/documentation mismatches ... OK
* checking Rd \usage sections ... OK
* creating rgr-Ex.R ... OK
* checking examples ... OK
* creating rgr-manual.tex ... OK
* checking rgr-manual.tex ... OK
```

Figure 16 Check Package rgr

If you see the list shown in the above figure, you are ready to build your package.

Step Seven: Build Your Package

At the command window, type “R CMD INSTALL --build packagename”. For example:

```
R CMD INSTALL  --build  rgr
```

```
U:\0312>R CMD INSTALL --build rgr

Using auto-selected zip options 'rgr-ZIPDATA=zip rgr-HELP=ziponly'

---------- Making package rgr ------------
  adding build stamp to DESCRIPTION
  installing NAMESPACE file and metadata
  installing R files
  installing inst files
  installing data files
  installing man source files
  installing indices
  zipping data
  installing help
 >>> Building/Updating help pages for package 'rgr'
     Formats: text html latex example chm
  anova1                  text    html    latex   example
  anova2                  text    html    latex   example
  bwplot                  text    html    latex   example

                          …

  var2fact                text    html    latex   example
  zipping help files
  preparing package rgr for lazy loading
Loading required package: akima
Loading required package: MASS
  adding MD5 sums

packaged installation of package 'rgr' as rgr_1.0.zip
* DONE (rgr)
```

Figure 17 Build Package rgr

Then you will get a binary build, .zip file, created in the directory that contains your package.

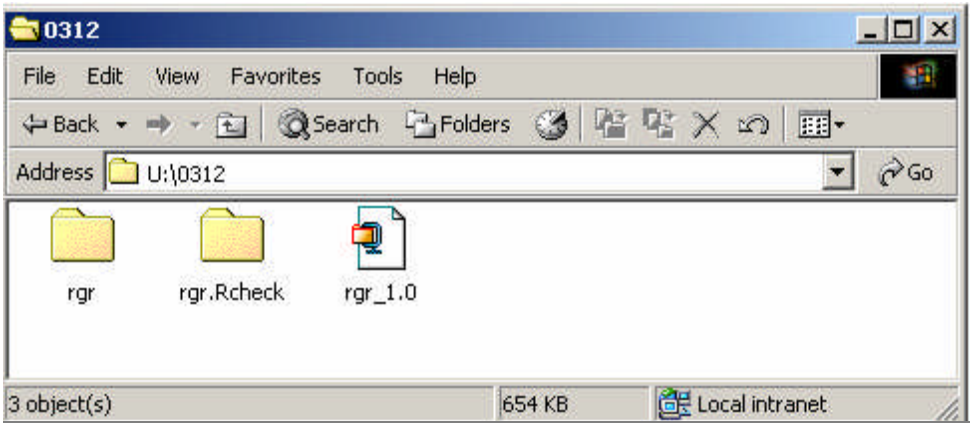

Figure 18 rgr_1.0 Binary

Step Eight: Install Your Package on Windows

Open RGui, from menu “Package”, select “Install package(s) from local zip files…”.

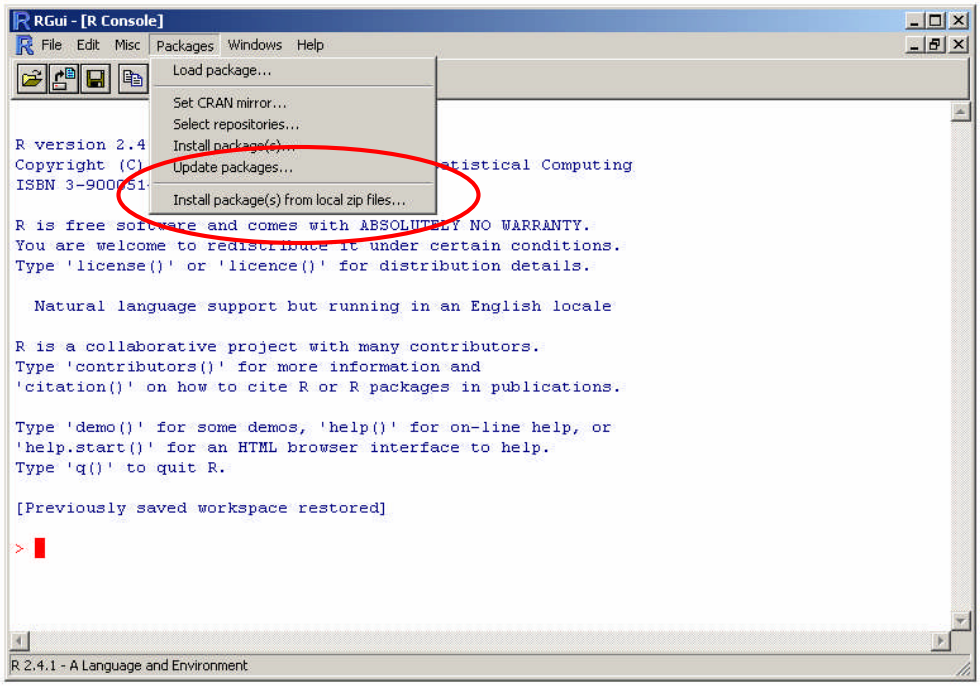

Figure 19 Install your package

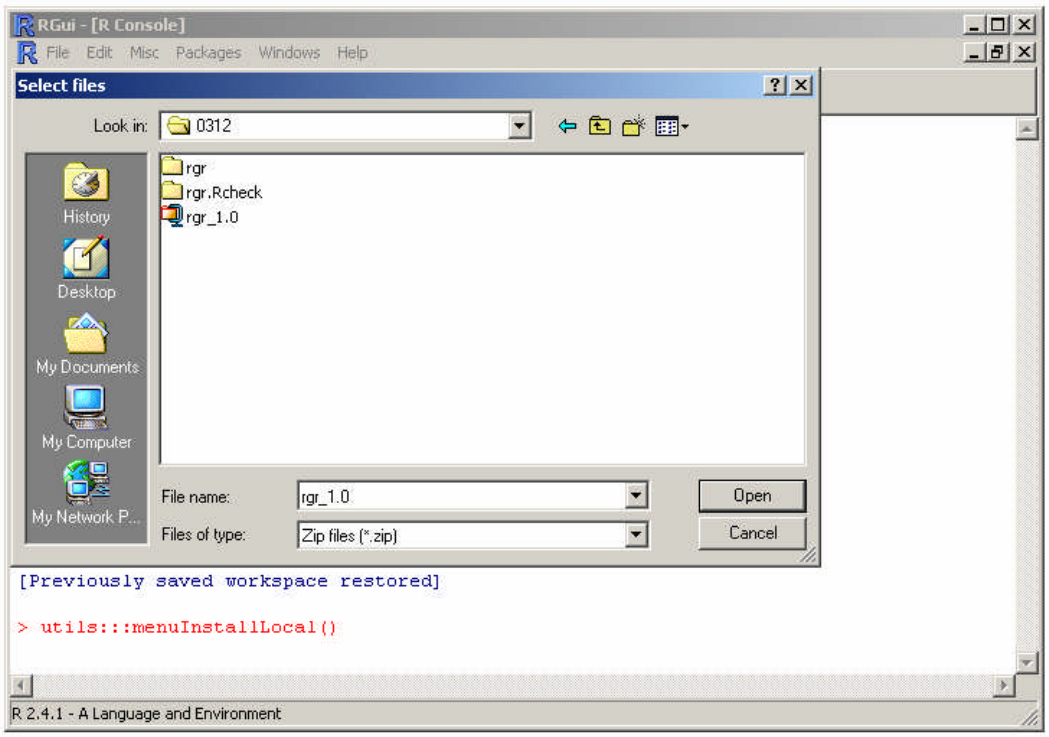

Figure 20 Choose the zip file

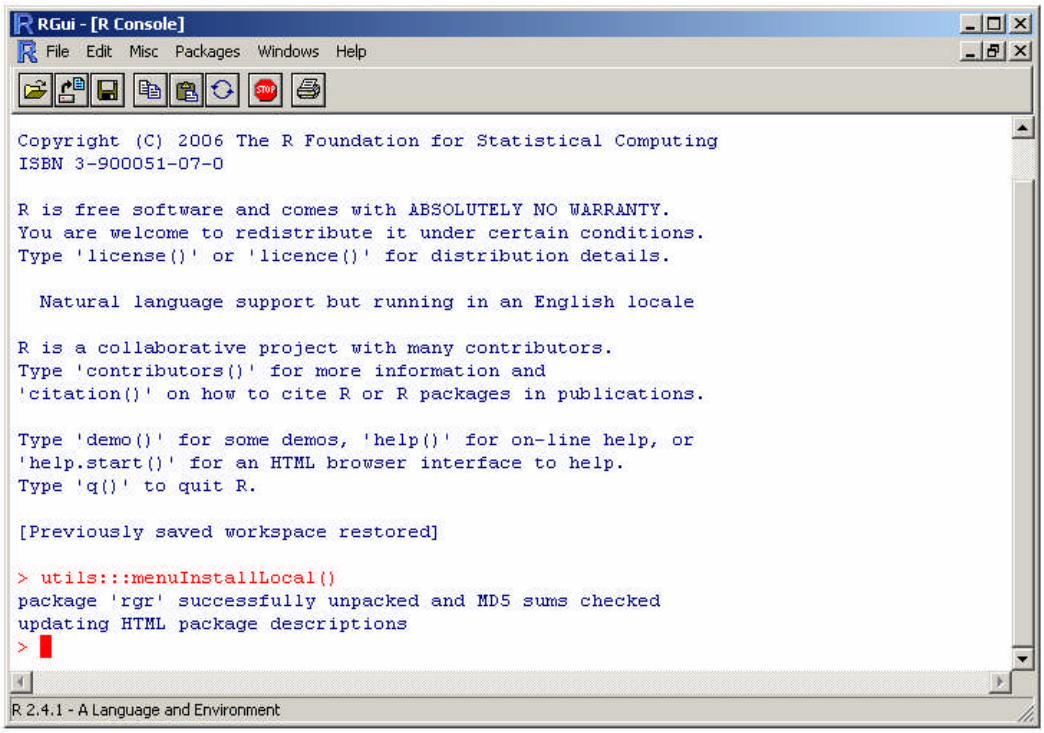

Figure 21 Get Package rgr Installed

You will see your package in R library when you installed it successfully.

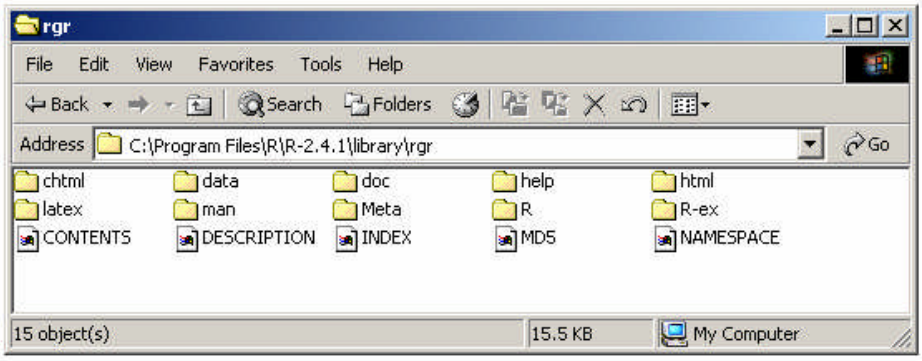

Figure 22 Package rgr in R Library

References

Chen,Y. (2006), *Handling Metadata for Statistical and Other databases*, MSc thesis, University of Ottawa (see Appendix C: Build an R Package Under Windows XP).

Keil, B. (2006), *MikTex 2.5 Installation and Configuration*, viewed from http://www.soft4science.com/MikTeX_Installation.html

Nunes, M. (2005), *Creating R Packages*, http://www.maths.bris.ac.uk/~maman

R Development Core Team (2005), *Writing R Extensions (version 2.4.1)*, also at http://www.r-project.org/

Rossi, P. (2005), *Making R Packages Under Windows: A Tutorial*, viewed from http://gsbwww.uchicago.edu/fac/peter.rossi/research

Sutherland, M. (2006), *Building R for Windows*, viewed from http://www.murdoch-sutherland.com/Rtools/

ADDENDUM I

Package Maintenance

To modify or update either the existing .R scripts or the .Rd help files in the rgr package, or add additional documentation as .pdf files, use the following steps:

1. Create a new working folder, e.g., U:\R, and within it a second folder U:\R\rgr;
2. Within U:\rgr create four additional folders: U:\R\rgr\R, U:\R\rgr\man, U:\R\rgr\data, U:\R\rgr\inst;
3. Copy the previous and modified .R scripts into U:\R\rgr\R;
4. Copy the previous and modified .Rd files into U:\R\rgr\man;
5. Copy the existing .rda data files (see note below) into U:\R\rgr\data;
6. In the U:\R\rgr\inst folder create a further folder U:\R\rgr\inst\doc;
7. Copy any .pdf files to be included in the package into U:\R\rgr\inst\doc;
8. Copy the previous DESCRIPTION and NAMESPACE files from the existing C:\Program Files\R\R-2.4.1\library\rgr folder into U:\R\rgr;
9. Edit the DESCRIPTION file and remove the last line starting: Built: R 2.4.1; ……;
10. Open the DOS Command window;
11. Move to the new folder, >cd U: and then >cd R. If necessary use cd.. repetitively to get to the root if you arrive in U:\...;
12. At U:\R type R CMD check rgr <CR> (see Step Six in main document);
13. Hopefully the check completes without ERROR occurring;
14. At U:\R type R CMD INSTALL --build rgr <CR> (see Step Seven in main document);
15. The result will be rgr_1.0.zip in folder U:\R, together with folder rgr.Rcheck;
16. To obtain the new .tar.gz file, type R CMD build rgr <CR>;
17. Move copies of these files to your folder for such items, back up as required;
18. In the RGui remove the previous rgr package with remove.packages("rgr")
19. In the RGui go to 'Packages', then 'Install packages(s) from local zip files…', and install the updated rgr package (see Step Eight in main document); and
20. Once the package is loaded use library(rgr) to make the package available and test it to ensure the maintenance work was implemented correctly.

Notes:

1. New test data files cannot be added using this procedure, to achieve that end a new package has to be built, see Addendum II;
2. Step 9 is important, failure to edit out the last line will cause the check to fail with an ERROR; and
3. Prior to testing it may be advantageous to set options(warn = -1) to suppress unwanted error messages arising from some graphics activities, and to use setwd(), e.g., setwd("C:\\R\\WDn"), so that saved files are located there.

RGG/20070410

ADDENDUM II

Extending the rgr Package, or Building a Completely New One

1. In the RGui collect together all the already existing function scripts and test data objects to be included in the new package, this can involve extracting objects from dump files using the source() command. In this context, it is useful to have single current dump files of all the data objects and script objects to be included in the new package;
2. Add any new function scripts or data objects, or, develop all new function scripts and their required test data in the work space, testing as required;
3. Once you are satisfied that everything is functioning properly, objlist <- ls();
4. Edit objlist, fix(objlist), and remove references to functions like first(), objects like rgrlist or datalist (for creating dump files), and any other objects not required in the package;
5. Now, assuming that the folder U:\R\ exists, type and enter, package.skeleton(list = objlist, name = "rgr", path = "U:\\R\\rgr1.1") <CR>;
6. At this point everything that is needed to complete the new package, rgr_1.1, is in U:\R\rgr1.1\rgr in folders: U:\R\rgr1.1\rgr\R, U:\R\rgr1.1\rgr\man and U:\R\rgr1.1\rgr\data;
7. Note the ultimate package name, rgr_1.1, as distinct from the folder name , rgr1.1, is defined in the DESCRIPTION file, see 12 below.
8. Read and delete the Read-and-delete-me file;
9. Copy any already existing .Rd files into U:\R\rgr1.1\rgr\man to replace the 'skeletons' just created;
10. Edit the skeletons for the new function and data description scripts to create informative help files consistent in style with the rest of the package;
11. If there are going to be .pdf files in the package, create U:\R\rgr1.1\rgr\inst, and within it U:\R\rgr1.1\rgr\inst\doc. Place any .pdf files to be included in the package in this new folder;
12. Edit the DESCRIPTION file skeleton as required, or if extending an existing package copy the earlier version DESCRIPTION file from the existing C:\Program Files\R\R-2.4.1\library\rgr folder into U:\R\rgr1.1\rgr\ and edit as required, remembering to update the Version number and to remove the last line (see Step Five in main document);
13. If extending an existing package, copy the NAMESPACE file from the earlier \library folder and edit to add the names of the new functions, alternately, build the file from scratch or with the help of a list of the current script objects (see Step Five in main document); and
14. Continue the procedure in ADDENDUM I at item 10, working in the DOS Command window from the appropriate folder.

RGG/20070406