R for Windows FAQ

Frequently Asked Questions on R for Windows
Version for rw2011
B. D. Ripley and D. J. Murdoch

Table of Contents
****

1 Introduction

This FAQ is for the Windows port of R: it describes features specific to that version. The main R FAQ can be found at

`http://www.ci.tuwien.ac.at/~hornik/R/R-FAQ.html'.

The information here applies only to recent versions of R for Windows, (`2.1.0' or later); the current version is often called something like `rw2011' (although not officially).

2 Installation and Usage

2.1 Where can I find the latest version?

Go to any CRAN site (see `http://cran.r-project.org/mirrors.html' for a list), navigate to the `bin/windows/base' directory and collect the file(s) you need. The current release is distributed as an installer `rw2011.exe' of about 25Mb. This contains all the components and allows as complete as installation as you choose.

There are also links on that page to the `r-patched' and `r-devel' snapshots. These are frequently updated builds of development versions of R. The `r-patched' build includes bug fixes to the current release, and `r-devel' contains these as well as changes that will eventually make it into the next `x.y.0' release.

2.2 How do I install R for Windows?

You need Windows 95/98/ME/NT4/2000/XP/2003 Server: Windows 3.11+win32s will not work. Your file system must allow long file names (as is likely except perhaps for some network-mounted systems). A full installation takes up about 50Mb of disk space and a minimal one about 18Mb.

If you want to be able to build packages from sources, we recommend that you choose an installation path not containing spaces. (Using a path with spaces in will probably work, but is little-tested.)

To install use `rw2011.exe'. Just double-click on the icon and follow the instructions. If you installed R this way you can uninstall it from the Control Panel or Start Menu (unless you supressed making a group for R).

Choose a working directory for R. You will have a shortcut to `rw2011\bin\Rgui.exe' on your desktop and/or somewhere on the Start menu file tree. Right-click each shortcut, select Properties... and change the `Start in' field to your working directory.

You may also want to add command-line arguments at the end of the Target field (_after_ any final double quote), for example `--sdi --max-mem-size=1Gb'. You can also set environment variables at the end of the Target field, for example `R_LIBS=p:/myRlib'.

2.3 The installer does not offer my language

Two things may be happening. First, only languages which can be displayed using the current codepage are offered so you cannot install in Japanese unless running Windows in Japanese.

Second, only a limited range of languages is supported (but still wider than those for which R has translations), currently Brazilian Portuguese, Catalan, (Simplified) Chinese, Czech, Danish, Dutch, French, German, Hungarian, Italian, Japanese, Korean, Norwegian Polish, Portuguese, Russian and Slovenian.

2.4 I want to run R in Chinese/Japanese/Korean

As from R 2.1.0 this is possible by installing R with support for `East Asian' languages (with a suitable version of Windows - Western installations of Windows XP do not by default have such support). You will need `msvcp60.dll' installed: it is part of modern versions of Windows, and is often installed by other programs. So if it is not found automatically it is worth searching your machine and copying it from another program's directory to R's `bin' directory.

In such locales `Rterm.exe' should be used for batch use only, but `RGui.exe' will support single- and double-width characters.

It will be necessary to select suitable fonts in `Rconsole' and `Rdevga' (see `?Rconsole' or the comments in the files: the system versions are in the `etc' folder); in the latter you can replace `Arial' by `Arial Unicode MS', and we tried `FixedSys' and `MS Mincho' in `Rconsole'. (Note that `Rdevga' only applies to Windows graphics devices and not, say, to `pdf'.)

You do need to ensure that R is running in a suitable locale: use `Sys.getlocale()' to find out. (CJK users may be used to their language characters always being available, which is the case for so-called `Unicode' Windows applications. However, R has to run on

Windows 9x and is not therefore `Unicode'.) You can find suitable
locale names from
`http://msdn.microsoft.com/library/default.asp?url=/library/en-
us/vccore98/html/_crt_language_and_country_strings.asp':
beware that `"Chinese"' is Traditional Chinese (code page 950, Big5)
and `"chs"' is needed for Simplified Chinese (code page 936, GB2312).
Note that this can be rather tricky: on Windows XP there are several
places to select the language under the `Regional and Language Options'
part of the control panel, and the appropriate language has to be set
under _both_ the `Regional Options' and `Advanced' tabs. (The `RGui'
menus may be corrupted if this is not done.)

2.5 How do I check an installation is not corrupt?

Run the program `bin\md5check.exe'. This compares checksums on all the
installed files with those put into the installer, and will report any
changed or missing files.

Users who installed `East Asian' support will see a report that `R.dll'
has changed, which is correct as it will have been replaced by a
version with support for those languages.

2.6 Can I customize the installation?

The normal way to customize the installation is by selecting components
from the wizards shown. However, sysadmins might like to install R from
scripts, and the following command-line flags are available for use with
the installer.

`/SILENT'
     only show the installation progress window and error messages.

`/VERYSILENT'
     only show error messages.

`/DIR="x:\dirname"'
     set the default installation directory

`/GROUP="folder name"'
     set the default Start-menu group name

`/COMPONENTS="comma separated list of component names"'
     set the initial list of components: Components are named `main',
     `chtml', `html', `latex', `manuals', `refman', `libdocs', `devel',
     `tcl', `mbcs', `Rd' and `trans'.

It is also possible to save the settings used to a file and later
reload those settings using

`/SAVEINF="filename"'
     save the settings to the specified file. Don't forget to use
     quotes if the filename contains spaces.

`/LOADINF="filename"'
     instructs the installer to load the settings from the specified

file after having checked the command line.

A successful installation has exit code 0: unsuccessful ones may give 1, 2, 3, 4 or 5. See the help for Inno Setup (`http://jrsoftware.org/isinfo.php') for details.

We have some facilities for building a customized installer, in particular to add packages to the installer. See the `R Installation and Administration' manual in the subsection `Building the installers'.

2.7 How do I run it?

Just double-click on the shortcut you prepared at installation.

If you want to set up another project, make a new shortcut or use the existing one and change the `Start in' field of the Properties.

You may if you prefer run R from the command line of any shell you use, for example an `MS-DOS window' (Windows 9x/ME), a `Command Prompt' (NT-based versions) or a port of a Unix shell such as `tcsh' or `bash'. (The command line can be anything you would put in the Target field of a shortcut, and the starting directory will be the currect working directory of the shell.)

2.8 Can I run R from a CD or USB drive?

Yes, with care. A basic R installation is relocatable, so you can burn an image on the R installation on your hard disc or install directly onto a removable storage device such as a flash-memory USB drive. (If you have installed packages into a private library, their absolute paths will appear in the HTML packages list.)

Running R does need access to a writable temporary directory and to a home directory, and in the last resort these are taken to be the current directory. This should be no problem on a properly configured NT-based version of Windows, but otherwise does mean that it may not be possible to run R without creating a shortcut in a writable folder.

R 2.0.x ran more slowly from a slow storage medium but this is no longer true as from 2.1.0.

2.9 How do I UNinstall R?

Normally you can do this from the R group on the Start Menu or from the `Add/Remove Programs' in the Control Panel. If it does not appear there or if you want to remove an old version, run `unins000.exe' in the top-level installation directory. (There should be a separate uninstall item in the R group for each installed version of R.)

Uninstalling R only removes files from the initial installation, not (for example) packages you have installed or updated.

If all else fails, you can just delete the whole directory in which R was installed.

2.10 What's the best way to upgrade?

That's a matter of taste. For most people the best thing to do is to uninstall R (see the previous Q), install the new version, copy any installed packages to the library folder in the new installation, run `update.packages()' in the new R (`Update packages from CRAN' from the Packages menu, if you prefer) and then delete anything left of the old installation. Different versions of R are quite deliberately installed in parallel folders so you can keep old versions around if you wish.

Upgrading from R 1.x.y to R 2.x.y is special as all the packages need to be reinstalled. Rather than copy them across, make a note of their names and re-install them from CRAN.

2.11 There seems to be a limit on the memory it uses!

Indeed there is. It is set by the command-line flag `--max-mem-size' (*note How do I install R for Windows?::) and defaults to the smaller of the amount of physical RAM in the machine and 1Gb. It can be set to any amount over 16M. (R will not run in less.) Be aware though that Windows has (in most versions) a maximum amount of user virtual memory of 2Gb, and parts of this can be reserved by processes but not used.

Use `?Memory' and `?memory.size' for information about memory usage. The limit can be raised by calling `memory.limit' within a running R session.

R can be compiled to use a different memory manager which might be better at using large amounts of memory, but is substantially slower (making R several times slower on some tasks).

If you are running a version of Windows (see `http://www.microsoft.com/whdc/system/platform/server/PAE/PAEmem.mspx') which supports more than 2Gb per process, you can let the R executables use it by marking their headers suitably. If you have Visual Studio use

```
editbin /LARGEADDRESSAWARE \path_to_R\bin\Rgui.exe
editbin /LARGEADDRESSAWARE \path_to_R\bin\Rterm.exe
```

and check this by

```
dumpbin /headers \path_to_R\bin\Rgui.exe
dumpbin /headers \path_to_R\bin\Rterm.exe
```

You should see a line like

```
                    Application can handle large (>2GB) addresses
```

2.12 How can I keep workspaces for different projects in different directories?

Create a separate shortcut for each project: see Q2.4. All the paths to files used by R are relative to the starting directory, so setting the `Start in' field automatically helps separate projects.

Alternatively, start R by double-clicking on a saved `.RData' file in the directory for the project you want to use, or drag-and-drop a file with extension `.RData' onto an R shortcut. In either case, the working directory will be set to that containing the file.

2.13 How do I print from R?

It depends what you want to print.

* You can print the graphics window from its menu or by using `dev.print' with suitable arguments (see its help page: most likely `dev.print(win.graph)' will work).

* You can print from the R console or pager by `File | Print'. (This will print the selection if there is one, otherwise the whole console or pager contents.)

* You can print help files from the pager or HTML browser.

* If you have configured `RHOME\bin\helpPRINT.bat' and have LaTeX installed you can print help files by `help(fn_name, offline=TRUE)'.

2.14 Can I use `R CMD BATCH'?

Yes: use `R CMD BATCH --help' or `?BATCH' for full details.

You can set also up a batch file using `Rterm.exe'. A sample batch file might contain (as one line)

```
path_to_R\bin\Rterm.exe --no-restore --no-save < %1 > %1.out 2>&1
```

The purpose of `2>&1' is to redirect warnings and errors to the same file as normal output, and users of Windows 95/98/ME's default `command.com' `shell' will need to omit it. (That program has no means to redirect `stderr', and `Rterm.exe' sends warnings and errors to the normal output file on such systems.)

2.15 Can I use rw2011 with ESS and (X)emacs?

Yes. Recent versions of ESS (e.g. 5.2.x) come with support for this version of R, and there is support for interrupting the R process from ESS (by `C-c C-c').

For help with ESS, please send email to <ESS-help@stat.ethz.ch>, not the R mailing lists.

2.16 What are HOME and working directories?

Several places in the documentation use these terms.

The working directory is the directory from which `Rgui' or `Rterm' was launched, unless a shortcut was used when it is given by the `Start in' field of the shortcut's properties. You can find this from R code by the call `getwd()'.

The home directory is set as follows: If environment variable `R_USER' is set, its value is used. Otherwise if environment variable `HOME' is

set, its value is used. After those two user-controllable settings, R tries to find system-defined home directories. It first tries to use the Windows "personal" directory (typically `C:\Documents and Settings\username\My Documents' in Windows XP). If that fails, if both environment variables `HOMEDRIVE' and `HOMEPATH' are set (and they normally are under Windows NT/2000/XP/2003), the value is `${HOMEDRIVE}${HOMEPATH}'. If all of these fail, the current working directory is used.

You can find this from R code by `Sys.getenv("R_USER")'.

2.17 How do I set environment variables?

Environment variables can be set for `RGui.exe' and `Rterm.exe' in three different ways.

1. On the command line as name=value pairs. For example in the shortcut to `RGui' you could have

   ```
   "path_to_R\bin\Rgui.exe" HOME=p:/ R_LIBS=p:/myRlib
   ```

2. In an environment file `.Renviron' in the working directory or your home directory, for example containing the line

   ```
   R_LIBS=p:/myRlib
   ```

   If you have permission to do so, you can also create an environment file `etc\Renviron.site' and set environmental variables in that file in the same way. This is useful for variables which should be set for all users and all usages of this R installation. (Their values can be overridden in a `.Renviron' or on the command line.)

   See `?Startup' for more details of environment files.

3. For all applications via Windows. How you set an environment variable is system specific: in Windows 9x you can set them in `autoexec.bat' or in an MS-DOS window from which you launch `Rgui' / `Rterm'. Under Windows NT/2000/XP/2003 you can use the control panel or the properties of `My Computer'. Under Windows ME you can use the System Configuration Utility (under Programs, Accessories, System Tools on the Start menu). You may have to log out or reboot for such changes to take effect.

The order of precedence for environmental variables is the order in which these options are listed, that is the command line then `.Renviron' then the inherited environment.

2.18 R can't find my file, but I know it is there!

How did you specify it? Backslashes have to be doubled in R character strings, so for example one needs `"d:\\rw2011\\library\\xgobi\\scripts\\xgobi.bat"'. You can make life easier for yourself by using forward slashes as path separators: they do work under Windows.

Another possible source of grief is spaces in folder names. We have tried to make R work on paths with spaces in, but many people writing packages for Unix do not bother. So it is worth trying the alternative short name (something like `PROGRA~1'; you can get it as the `MS-DOS name' from the Properties of the file on most versions of Windows, and from `dir /X' in a `Command Prompt' window on 2000/XP/2003).

2.19 Does R use the Registry?

Not itself.

The installer sets some entries to allow uninstallation. In addition (by default, but this can be de-selected) they set a Registry key `LOCAL_MACHINE\Software\R-core\R' giving the version and install path. Again, this is not used by R itself, but it may be used by third-party front-ends such as Thomas Baier's (D)COM interface (`http://cran.r-project.org/other-software.html'). Finally, a file association for extension `.RData' is set in the Registry.

You can add the Registry entries by running `RSetReg.exe' in the `bin' folder, and remove them by running this with argument `/U'. Note that the settings are all per machine and not per user, and that this neither sets up nor removes the file associations.

2.20 Does R support automation (OLE, COM)?

Directly, no.

There is a (D)COM server written by Thomas Baier available on CRAN (`http://cran.r-project.org/other-software.html') which works with `Rproxy.dll' (in the R distribution) and `R.dll' to support transfer of data to and from R and remote execution of R commands, as well as embedding of an R graphics window. An R-Excel interface making use of the DCOM server is included in the distribution.

Another (D)COM server is available from `http://www.omegahat.org/', which allows R objects to be exported as COM values. That site also has packages `RDCOMClient' and `SWinTypeLibs' which allow R to act as a (D)COM client.

2.21 The internet download functions fail.

for example `update.packages()' and the menu items on the Packages menu.

We have had several reports of this, although they do work for us on _all_ of our machines. There are two known possible fixes.

(a) Use the alternative `internet2.dll' by starting R with the flag `--internet2' (*note How do I install R for Windows?::) which uses the Internet Explorer internals (and so needs Internet Explorer 4 or later installed). Note that this does not work with proxies that need authentication.

(b) A proxy needs to be set up: see `?download.file'. Here are two

versions of an example (a real one, but from a machine that is only available locally) of a command-line in a short cut:

```
"path_to_R\bin\RGui.exe" http_proxy=http://user:pass@gannet:80/

"path_to_R\bin\RGui.exe" http_proxy=http://gannet/ http_proxy_user=ask
```

The second version will prompt the user for the proxy username and password when HTTP downloads are first used.

2.22 Entering certain characters crashes Rgui.

This used to happen occasionally, and all the occurrences we have solved have been traced to faulty versions of `msvcrt.dll'. We have installed a workaround that seems to avoid this. A few other people have discovered this was caused by desktop switcher and keyboard macro programs, for example `Macro Magic' and `JS Pager'.

If it still happens, try extracting the `msvcrt.dll' to be found in the self-extracting archive `ftp://ftp.microsoft.com/softlib/mslfiles/msvcrt.exe' and put it in the `rw2011\bin' directory. Removing `msvcrt.dll' from that directory reverts to the standard behaviour. It seems that on some versions of Windows (but not 2000/XP/2003) you also need to put the `rw2011\bin' directory early in your path.

This fix has solved other problems too, for example incorrect results in the date-time functions. However, you are probably better off re-installing Windows.

2.23 Other strange crashes.

Some users have found that `Rgui.exe' fails to start, exiting with a "Floating- point invalid operation" or other low level error. This error may also happen in the middle of a session. In some cases where we have tracked this down, it was due to bugs in the video driver on the system in question: it makes changes to the floating point control word which are incompatible with R. (Good practice would restore the control word to the state it was in when the driver code was called.) For example, one user reported that the virtual screen manager JSP2 caused this crash.

These errors are essentially impossible for us to fix or work around. The only solution we know of is for the user to replace the buggy driver that is causing the error.

3 Packages

3.1 Can I install packages into libraries in this version?

Yes, but you will need a lot of tools to do so, unless the author or the maintainers of the `bin/windows/contrib' section on CRAN have been kind

enough to provide a pre-compiled version for Windows as a `.zip' file.

You can install pre-compiled packages either from CRAN or from a local `.zip' file by using `install.packages': see its help page. There are menu items on the `Packages' menu to provide a point-and-click interface to package installation. The packages for each minor (2.x) version will be stored in a separate area, so for R 2.1.? the files are in `bin/windows/contrib/2.1'. You can try those compiled for earlier versions (but not before 2.0.0), at your own risk.

Note that the pre-compiled versions on CRAN are unsupported: see `http://cran.r-project.org/bin/windows/contrib/ReadMe', which also gives the locations of a few other precompiled packages.

If there is not a pre-compiled version or that is not up-to-date or you prefer compiling from source, read the `R Installation and Administration' manual section on `Add-on Packages'. You need to make sure you installed the necessary files, and you will need to collect and install several tools: you can download them via the portal at `http://www.murdoch-sutherland.com/Rtools/'. Once you have done so, just run `R CMD INSTALL pkgname'. To check the package (including running all the examples on its help pages and in its test suite, if any) use `R CMD check pkgname': see the _`Writing R Extensions'_ manual.

Note that this is rather tricky; please do ensure that you have followed the instructions *exactly*. At least 90% of the questions asked are because people have not done so.

3.2 I don't have permission to write to the `rw2011\library' directory.

You can install packages anywhere and use the environment variable `R_LIBS' (*note How do I set environment variables?::) to point to the library location(s).

Suppose your packages are installed in `p:\myRlib'. Then you can EITHER

```
set the environment variable R_LIBS to p:\myRlib before starting R
```

OR use a package by, e.g.

```
library(mypkg, lib.loc="p:/myRlib")
```

3.3 The packages I installed do not appear in the HTML help system.

To update the HTML indices after you have installed a pre-compiled package, run at the R prompt.

```
> link.html.help()
```

This is done automatically when installing from the Packages menu or by `install.packages()', and when `help.start' is run, _provided_ you have write permission in `rw2011'. If you do not have sufficient permission, you will get warnings and the packages you install will not appear in the list of packages or the search system.

3.4 My functions are not found by the HTML help search system.

The following conditions need to hold for functions in a package you installed.

* the package contains a `CONTENTS' file in its top-level directory.

* you updated the indices as described in the answer to Q3.3 after installing the package.

If those hold, this works for us. Note that if you were unable to update the indices (for which you need write permission in the `rw2011' directory), only the functions in packages installed in the main library will be found.

If the help search system does not work _at all_, this probably indicates that Java support is either not installed or not enabled in your browser. The search page contains a link to the appropriate section in the _R Installation and Administration_ manual.

3.5 Loading a package fails.

Is the package compiled for this version of R? Many of the packages need to be compiled for a fairly recent version.

You can tell the version the package was compiled for by looking at the `Built:' line in its `DESCRIPTION' file or at the `Version' tab of its DLL in the `libs' directory. (Right-click on the DLL in Windows Explorer and select `Version' tab of the `Properties', or use the `DLL.version' function inside R.)

3.6 Package TclTk does not work.

For package `tcltk' to work (try `demo(tkdensity)' or `demo(tkttest)' after `library(tcltk)') you need to have Tcl installed. This is an optional part of the installation although it is selected by default. If the message is

```
     Tcl/Tk support files were not installed
```

the optional files were not installed, and you need to go back to the installer and install them.

Alternatively, if you have the environment variable `MY_TCLTK' set to a non-empty value, it is assumed that you want to use a different Tcl/Tk 8.4.x installation, and that this is set up correctly (with the DLLs in your path and `TCL_LIBRARY' set). In that case you do not need the Tcl/Tk support files installed (but they can be). Note that you do need 8.4.x and not 8.3.x. (If you build R from the sources yourself you can configure it to use 8.3.x.)

3.7 Hyperlinks in HTML sometimes do not work.

They may well not work between packages installed in different libraries. This is solved under Unix using symbolic links which Windows does not implement.

With HTML, `help.start()' fixes up links to the most of the standard packages if it has write permission in the main library tree. Not even these work for Compiled HTML help.

Currently links to the `base', `datasets', `utils', `grDevices', `graphics' and `stats' packages are fixed.

3.8 `update.packages()' fails

You may not be able to update a package which is in use: Windows `locks' the package's DLL when it is loaded. So use `update.packages()' (or the menu equivalent) in a new session.

If you put `library(foo)' in your `.Rprofile' you will need to start R with `--vanilla' to be able to update package `foo'. If you set `R_DEFAULT_PACKAGES' to include `foo', you will need to unset it temporarily.

3.9 How do I add to the list of repositories?

as shown in the `Select repositories...' item on the `Packages' menu?

This reads from the tab-delimited file `R_HOME/etc/repositories', which you can edit, or put a modified copy at `.R/repositories' in your HOME directory (*note What are HOME and working directories?::).

4 Windows Features

4.1 What should I expect to behave differently from the Unix version of R?

* R commands can be interrupted by <Esc> in `Rgui.exe' and <Ctrl-break> or <Ctrl-C> in `Rterm.exe': <Ctrl-C> is used for copying in the GUI version.

* Command-line editing is always available, but is simpler than the readline-based editing on Unix. For `Rgui.exe', the menu item `Help | Console' will give details. For `Rterm.exe' see file `README.rterm'.

* Using `help.start()' does not automatically send help requests to the browser: use `options(htmlhelp=TRUE)' to turn this on.

* The HTML help system has limitations in supporting cross-library links.

* Paths to files (e.g. in `source()') can be specified with either "/" or "\\".

* `system()' is slightly different: see its help page and that of `shell()'.

4.2 I hear about some nifty features: please tell me about them!

You have read the file `README.rw2011'? There are file menus on the R console, pager and graphics windows. You can source and save from those menus, and copy the graphics to `png', `jpeg', `bmp', `postscript', `PDF' or `metafile'. There are right-click menus giving shortcuts to menu items, and optionally toolbars with buttons giving shortcuts to frequent operations.

If you resize the R console the `options(width=)' is automatically set to the console width (unless disabled in the configuration file).

The graphics has a history mechanism. As `README.rw2011' says:

> `The History menu allows the recording of plots. When plots have been recorded they can be reviewed by <PgUp> and <PgDn>, saved and replaced. Recording can be turned on automatically (the Recording item on the list) or individual plots can be added (Add or the <INS> key). The whole plot history can be saved to or retrieved from an R variable in the global environment. The format of recorded plots may change between R versions. Recorded plots should *not* be used as a permanent storage format for R plots.
>
> There is only one graphics history shared by all the windows devices.'

The R console and graphics windows have configuration files stored in the `RHOME\etc' directory called `Rconsole' and `Rdevga'; you can keep personal copies in your `HOME' directory. They contain comments which should suffice for you to edit them to your preferences. For more details see `?Rconsole'. There is a Preferences editor invoked from the `Edit' menu which can be used to edit the file `Rconsole'.

4.3 Circles appear as ovals on screen

The graphics system asks Windows for the number of pixels per inch in the X and Y directions, and uses that to size graphics (which in R are in units of inches). Sometimes the answer is a complete invention, and in any case Windows will not know exactly how the horizontal and vertical size have been set on a CRT. You can specify correct values either in the call to `windows' or as options: see `?windows'. (Typically these are of the order of 100.)

On one of our systems, the screen height was reported as 240mm, and the width as 300mm in 1280 x 1024 mode and 320mm in 1280 x 960 and 1600 x 1200 modes. In fact it is a 21" monitor and 400mm x 300mm!

4.4 How do I move focus to a graphics window or the console?

You may want to do this from within a function, for example when calling `identify' or `readline'. Use the function `bringToTop()'. With its default argument it brings the active graphics window to the top and gives it focus. With argument `-1' it brings the console to the top and gives it focus.

This works for `Rgui.exe' in MDI and SDI modes, and can be used for graphics windows from `Rterm.exe' (although Windows may not always act on it).

5 Workspaces

5.1 My workspace gets saved in a strange place: how do I stop this?

Have you changed the working directory?: see Q5.2.

5.2 How do I store my workspace in a different place?

Use the `File | Change Dir...' menu item to select a new working directory: this defaults to the last directory you loaded a file from. The workspace is saved in the working directory. You can also save a snapshot of the workspace from the `Save Workspace...' menu item.

From the command line you can change the working directory by the function `setwd': see its help page.

5.3 Can I load workspaces saved under Unix/GNU-Linux or MacOS?

Yes. All ports of R use the same format for workspaces, so they are interchangeable (for the same 2.x.? version of R, at least).

6 The R Console and Fonts

6.1 I would like to be able to use Japanese fonts

for example, in the console and to annotate graphs. Similar comments apply to any non-Western European language.

We believe this is possible by setting suitable fonts in the `Rconsole' and `Rdevga' configuration files (see Q4.2 and the next Q). You can specify additional fonts in Rdevga, and use them by

```
par(font=, font.lab=, font.main=, font.sub=)
```

Nineteen fonts are specified (as 1 to 19) by default: you can add to these (up to 13 more) or replace them.

In addition, the Hershey vector fonts (see `?Hershey', `?Japanese' and `demo(Japanese)') can be used on any graphics device to display Japanese characters.

To use non-Latin 1 characters in the `postscript' graphics device, see its help page.

6.2 I don't see characters with accents at the R console, for example in ?text.

==========

You need to specify a font in Rconsole (see Q4.2) that supports the encoding in use, in Western European languages Latin-1. The default font, `Courier New', does on our systems, as does `FixedSys'. This may be a problem in other locales, especially for non-Western European languages.

Support for these characters within `Rterm' depends on the environment (the terminal window and shell, including locale settings) within which it is run as well as the font used by the terminal window.

If you are using a non-Latin 1 language, you do need to ensure that the fonts you selected support the language. For example, it was found by one Czech user (under Windows 98) that he had to select `Times New Roman CE' or `Courier' (not `Courier New') to get certain Czech characters displayed correctly.

6.3 When using Rgui the output to the console seems to be delayed.

This is deliberate: the console output is buffered and re-written in chunks to be faster and less distracting. You can turn buffering off or on from the `Misc' menu or the right-click menu: <Ctrl-W> toggles the setting.

If you are sourcing R code or writing from a function, there is another option. A call to the R function `flush.console()' will write out the buffer and so update the console.

6.4 Long lines in the console or pager are truncated.

They only *seem* to be truncated: that $ at the end indicates you can scroll the window to see the rest of the line. Use the horizontal scrollbar or the <CTRL + left/right arrow> keys to scroll horizontally. (The <left/right arrow> keys work in the pager too.)

7 Building from Source

7.1 How can I compile R from source?

Get the R sources. Suppose you want to compile R-2.1.1. Start in a directory whose path does not contain spaces, and run

```
tar zxvf R-2.1.1.tgz
cd R-2.1.1
cd src\gnuwin32
```

Now read the `R Installation and Administration' manual and set up all the tools needed. Then you can just use `make all bitmapdll recommended vignettes', sit back and wait. (A complete build takes about 20 minutes on a 2.6GHz P4 with a fast local disc.)

You may need to compile under a case-honouring file system: we found

that a `samba'-mounted file system (which maps all file names to lower case) did not work.

7.2 Can I use a fast BLAS?

Fast BLAS (Basic Linear Algebra Subprograms, `http://www.netlib.org/blas/faq.html') routines are used to speed up numerical linear algebra. There is support in the R sources for the `tuned' BLAS called ATLAS (`http://math-atlas.sourceforge.net'). The savings can be appreciable: on a 2.6GHz P4 and a 1000 x 1000 matrix `svd' took 16.2 sec with the standard BLAS and 7.8 sec with ATLAS. Because ATLAS is tuned to a particular chip we can't use it generally: the optimal routines for a P4 or an Athlon XP are quite different and neither will not run at all on a PII.

BLAS support is supplied by the single DLL `R_HOME\bin\Rblas.dll', and you can add a fast BLAS just by replacing that. Replacements for some of the more common chips are available on CRAN in directory `bin/windows/contrib/ATLAS'.

If you are building R from source, in the file `MkRules' there are macros `USE_ATLAS' and `ATLAS_PATH'. Set `USE_ATLAS = YES' and `ATLAS_PATH' to where the ATLAS libraries are located. You will need to make the libraries yourself: none of the binaries we have seen are compiled for the correct compiler.

Even faster hand-coded routines are available as DLLs from Dr Kazushige Goto for certain CPUs (Pentium III and 4 and Opteron). He does not allow redistribution: they are currently available _via_ `http://www.cs.utexas.edu/users/kgoto/signup_first.html' and there is support in the R sources to build `R_HOME\bin\Rblas.dll' to link to one of his DLLs. On the `svd' problem they took 6.8 sec.

7.3 How do I include compiled C code?

We strongly encourage you to do this _via_ building an R package: see the _`Writing R Extensions'_ manual. In any event you should install the parts of the R system for building R packages (installed by default), and get and install the tools (including Perl) and compilers mentioned in the _`R Installation and Administration'_ manual. Then you can use

```
     ...\bin\R CMD SHLIB foo.c bar.f
```

to make `foo.dll'. Use `...\bin\R CMD SHLIB --help' for further options, or see `?SHLIB'.

If you want to use Visual C++, Borland C++ or other compilers, see the appropriate section in `README.packages'.

7.4 How do I debug code that I have compiled and dyn.load-ed?

You will need a suitable version of `gdb': we normally use that from the Cygwin distribution. Debugging under Windows is often a fraught process, and sometimes does not work at all. If all you need is a _just-in-time_ debugger to catch crashes, consider `Dr. Mingw' from the

`mingw-utils' bundle on `http://www.mingw.org' (see also
`http://jrfonseca.dyndns.org/projects/gnu-win32/software/drmingw/').
That will be able to pinpoint the error, most effectively if you build a
version of R with debugging information as described below.

First, build a version of the R system with debugging information by

```
make clean
make DEBUG=T
```

and make a debug version of your package by either

```
make pkgclean-mypkg
make DEBUG=T pkg-mypkg
```

or

```
Rcmd install -c mypkg
set DEBUG=T
Rcmd install mypkg
```

Then you can debug by

```
gdb /path/to/rw2011/bin/Rgui.exe
```

However, note

* `gdb' will only be able to find the source code if we run in the location where the source was compiled (`rw2011/src/gnuwin32' for the main system, `rw2011/src/library/mypkg/src' for a package), unless told otherwise by the `directory' command. It is most convenient to set a list of code locations via `directory' commands in the file `.gdbinit' in the directory from which `gdb' is run.

* It is only possible to set breakpoints in a DLL after the DLL has been loaded. So a way to examine the function `tukeyline' in package `stats' might be

```
gdb ../../../../bin/Rgui.exe
(gdb) break WinMain
(gdb) run
[ stops with R.dll loaded ]
(gdb) break R_ReadConsole
(gdb) continue
[ stops with console running ]
(gdb) continue
Rconsole> library(stats)
(gdb) break tukeyline
(gdb) clear R_ReadConsole
(gdb) continue
```

  Alternatively, in Rgui you can use the `Misc|Break to debugger' menu item after your DLL is loaded. The C function call `breaktodebugger()' will do the same thing.

* Fortran symbols need an underline appended.

* Windows has little support for signals, so the Unix idea of running a program under a debugger and sending it a signal to interrupt it and drop control back to the debugger does not work with a `mingw' version of `gdb'. It does often work with the `cygwin' version.

* Other debuggers than `gdb' can be used. We have successfully used the `insight' front-end to `gdb' (see `http://sources.redhat.com/insight/'), and Borland's debugger with a DLL compiled in a Borland compiler.

7.5 How do I include C++ code?

You need to do two things:

(a) Write a wrapper to export the symbols you want to call from R as `extern "C"'.

(b) Include the C++ libraries in the link to make the DLL. Suppose `X.cc' contains your C++ code, and `X_main.cc' is the wrapper, as in the example in _`Writing R Extensions'_. Then build the DLL by (`gcc')

```
     ...\bin\R CMD SHLIB X.cc X_main.cc
```

or (VC++, which requires extension `.cpp')

```
     cl /MT /c X.cpp X_main.cpp
     link /dll /out:X.dll /export:X_main X.obj X_main.obj
```

or (Borland C++, which also requires extension `.cpp')

```
     bcc32 -u- -WDE X.cpp X_main.cpp
```

and call the entry point(s) in `X_R', such as `X_main'. Construction of static variables will occur when the DLL is loaded, and destruction when the DLL is unloaded, usually when R terminates.

Note that you will not see the messages from this example in the GUI console: see the next section.

This example used to be in package `cxx_0.0-x.tar.gz' in the `src/contrib/Devel' section on CRAN, and could be compiled as a package in the usual way on Windows.

7.6 The output from my C code disappears. Why?

The `Rgui.exe' console is a Windows application: writing to `stdout' or `stderr' will not produce output in the console. (This will work with `Rterm.exe'.) Use `Rprintf' or `REprintf' instead. These are declared in header file `R_ext/PrtUtil.h'.

Note that output from the console is delayed (*note The output to the console seems to be delayed::), so that you will not normally see any output before returning to the R prompt.

7.7 The output from my Fortran code disappears. Why?

Writing to Fortran output writes to a file, not the `Rgui' console. Use one of the subroutines `dblepr', `intpr' or `realpr' documented in the _`Writing R Extensions'_ manual.

Note that output from the console is delayed (*note The output to the console seems to be delayed::), so that you will not normally see any output before returning to the R prompt even when using the `xxxpr' subroutines.

7.8 The console freezes when my compiled code is running.

The console, pagers and graphics window all run in the same thread as the R engine. To allow the console etc to respond to Windows events, call `R_ProcessEvents()' periodically from your compiled code. If you want output to be updated on the console, call `R_FlushConsole()' and then `R_ProcessEvents()'.

Last edited 2005 June 12 by Professor B. D. Ripley <ripley@stats.ox.ac.uk>