Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)
March 20–22, Vienna, Austria ISSN 1609-395X
Kurt Hornik, Friedrich Leisch & Achim Zeileis (eds.)
http://www.ci.tuwien.ac.at/Conferences/DSC-2003/

R-WinEdt

Uwe Ligges

Abstract

In the first part of this paper the status quo of R-WinEdt, a plug-in for the shareware editor WinEdt for Windows, is described. This description includes the interface between R and WinEdt, convenient editing features like syntax highlighting, and the installation procedure. The second part is aimed at the future of R-WinEdt. Is it desirable to improve the plug-in? Possible improvements related to the interface, and integration of Sweave functionality are discussed.

1 Introduction

R-WinEdt[1] is a plug-in for WinEdt[2], a shareware editor for the operating system Windows[3]. It provides an "interface" to R (Ihaka and Gentleman, 1996), a statistical programming language and environment. Additionally, a highlighting scheme, which illuminates the syntactical structure of the R code, replaces WinEdt's defaults. Shortcuts, menus, and toolbar buttons are included to provide easy access to common operations for the user's convenience. The existing functionality will be presented in Section 3, just after some comments why WinEdt is the editor of my choice in Section 2.

Section 4 is aimed at the future of R-WinEdt. Its "interface" to R will be discussed, as well as possible changes and improvements. Further on, it will be discussed whether it is desirable to have full support of Sweave (Leisch, 2002), which provides convenient state of the art literate statistical analysis with R.

2 Why WinEdt?

The recommended editor for writing R code is the Emacs (Stallmann, 1999) with its ESS interface (*Emacs Speaks Statistics*; Rossini, Heiberger, Sparapani, Mächler,

[1] http://CRAN.R-Project.org/contrib/extra/winedt/
[2] http://www.WinEdt.com/
[3] Windows is a trademark of the Microsoft Corporation.

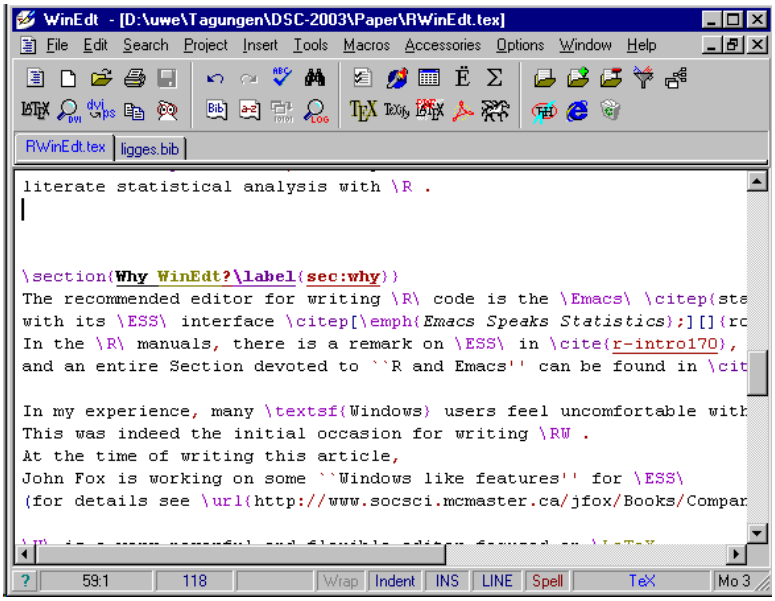

Figure 1: WinEdt in its "regular" LaTeX mode.

and Hornik, 2003). In the R manuals, there is a remark on ESS in Venables, Smith, and the R Development Core Team (2003), and an entire Section devoted to "R and Emacs" can be found in Hornik (2003). In my experience, many Windows users feel uncomfortable[4] with the Emacs. This was indeed the initial occasion for writing R-WinEdt.

WinEdt is a very powerful and flexible editor focused on LaTeX, and it is widely known among LaTeX users under Windows (see Figure 1 for a screenshot). It is easy to define Task Bars, Menus, and quite sophisticated highlighting schemes in WinEdt. The editor also provides an extended Macro language that includes, e.g., capabilities for interaction with the operating system. There are convenient ways to automate tasks in WinEdt using this macro language (the Linux user typically uses a Makefile, instead). For details I refer to the WinEdt manuals.

Because of all these features, it was easy to implement the simple plug-in in its current version. Unfortunately, WinEdt is a commercial (affordable) shareware editor, hence I have no access to its sources. So it is necessary to convince WinEdt's author, Aleksander Simonic[5], in order to get desirable new features into the editor.

3 Status quo

The most easiest way to get to know the current state of R-WinEdt is to try it out and play around. Nevertheless, in this Section R-WinEdt's status quo and some corresponding technicalities will be described. See Figure 2 for a screenshot of R-WinEdt in action.

[4] At the time of writing this article, John Fox is working on some "Windows like features" for ESS. For details see http://www.socsci.mcmaster.ca/jfox/Books/Companion/ESS/.

[5] alex@winedt.com

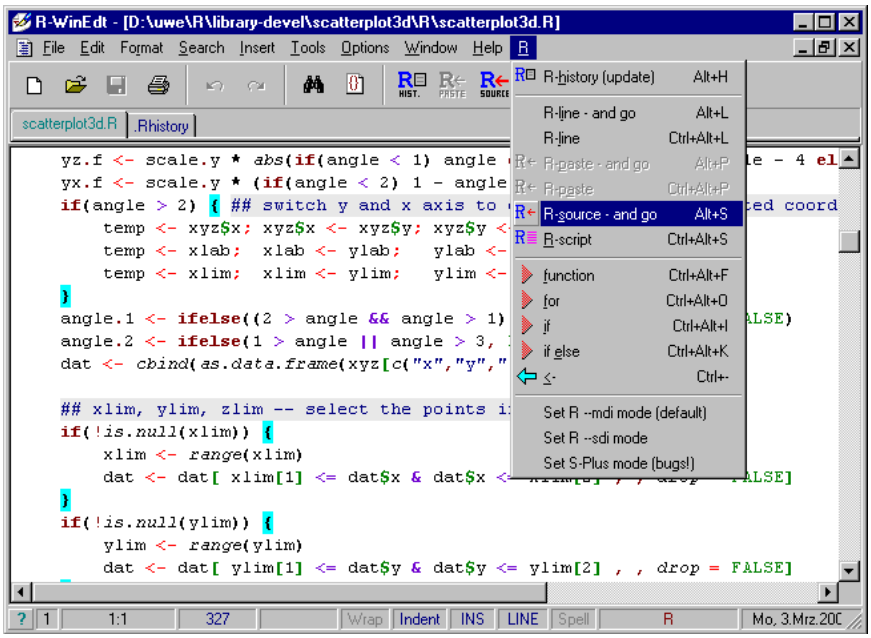

Figure 2: WinEdt running with the R-WinEdt plug-in

The recent version (1.5-0, at the time of writing) of R-WinEdt works with WinEdt 5.2 and later on any 32-bit version of Windows and requires R-1.2.0 or later. Of course, any version numbers are subject to change. In its most recent version, R-WinEdt is shipped in form of a binary package for R for Windows, which requires (for an automatical installation procedure) Duncan Temple Lang's `SWinRegistry` package of "The Omega Project for Statistical Computing"[6] in order to perform required tasks related to the Windows registry.

I received quite a lot questions related to the former (but still available) manual installation procedure of R-WinEdt, which indeed was uncommon in a way, because there was just a simple file `install.bat` that had to be invoked in the exactly correct directory.

For detailed installation instructions see R-WinEdt's (edited) `ReadMe.txt` file in the Appendix.

3.1 Interface

"Interface" is a Section title which implies more than what has been implemented. On the one hand, unfortunately, WinEdt does not support DCOM[7], but uses DDE[8] instead. I was not able to convince WinEdt's author to implement DCOM support in WinEdt. On the other hand, there is a DCOM server (Neuwirth and Baier, 2001) for R available, but no DDE interface.

[6] http://www.Omegahat.org/

[7] DCOM: *Distributed Component Object Model*

[8] DDE: *Dynamic Data Exchange*, basic technology of Microsoft's OLE (*Object Linking and Embedding*).

What I did instead is making use of some features of WinEdt's macro language in the following manner. When any information has to be send to R, it is constructed by the macro language. Then it is copied to the Windows clipboard and pasted into the R Console window of RGui using the following macro, which implements the awkward main part of the interface:

```
SetFocus("RGui");                              // set focus to R
PostMessage("RGui", $0104, $12, $20000001); // press ALT
Wait(150);                                     // RGui needs some time
PostMessage("RGui", $0105, $12, $20000001); // release ALT
PostMessage("RGui", $0102, 87, 1);           // Char: W -> Win. Menu
PostMessage("RGui", $0102, 49, 1);           // Char: 1 -> Window1
                                               //          = R Console
Wait(150);                                     // RGui needs some time
PostMessage("RGui", $0104, $12, $20000001); // press ALT
PostMessage("RGui", $0105, $12, $20000001); // release ALT
PostMessage("RGui", $0102, 69, 1);           // Char: E -> Edit Menu
PostMessage("RGui", $0102, 80, 1);           // Char: P -> Paste
```

So this main part can be described as follows. After the focus of programs is switched from WinEdt to R (and both applications have been started, of course), key presses are simulated in a manner like one would use the menu without any mouse:

1. Press `Alt`.
2. RGui (and Windows) needs some time to open the menu, (time depends of the operating system, it takes most time on Windows XP).
3. Release `Alt`.
4. Press `W` – opens the menu called `Windows`.
5. Press `1` – selects "window 1" which is identical to the R Console; now we are sure to have focussed the R Console (instead of, e.g., an R Graphics window).
6. Again, some time is needed to switch to the R Console window.
7. Press and release `Alt` to open the menu again.
8. Press `E` to open the `Edit` menu, and
9. press `P` to `Paste` the contents of the clipboard into the R Console window.

This macro code is written for working with RGui running in `--mdi` mode, a version working with RGui's `--sdi` mode is somewhat simpler:

```
SetFocus("R Console");
Wait(150);
PostMessage("R Console", $0104, $12, $20000001);
Wait(150);
PostMessage("R Console", $0105, $12, $20000001);
PostMessage("R Console", $0102, 69, 1);
PostMessage("R Console", $0102, 80, 1);
```

Obviously, the "interface" is an awkward one, but it is extremely simple with just 11 (7) lines of macro code. There is one "bug" in this macro: If RGui is minimized, it is not possible to interact with it. For a discussion whether it is desirable to improve the interface see Section 4.

The question arises, what kind of information can be pasted into the R session. Two examples are given, the first one shows a macro that pastes a selected region of R code from R-WinEdt (both shortcuts and icons can be set up to invoke macros):

```
CMD("Copy");                    // copy selected text into a buffer
ReplaceInString("1", ">", 0, 1, 1, 9); // add a newline at the end
CopyToClipboard('%!9', 1);      // copy buffer into the clipboard
LetReg(9,"");                   // clear the buffer
Exe("%b\send2R.edt")            // execute the main interface
End;
```

The second example shows a macro that saves the current file and `source()`s it into R:

```
CMD("Save");            // save the current file
GetChildName(9);        // save the ...\path\filename into buffer1
DosToUnix("%!9", 8);    // translate "\" to "/" in buffer1
      // copy a "newline" in buffer2:
ReplaceInString("1", ">", 0, 1, 1, 9);
      // copy 'source("buffer1")buffer2' into the clipboard:
CopyToClipboard('source("%!8")%!9', 0);
LetReg(9,"");           // clean up buffers 1 and 2
LetReg(8,"");
Exe("%b\send2R.edt")    // execute the main interface
End;
```

For details on WinEdt's macro language I refer to its manual and the two versions of "The WinEdt Hacker's Guide" by Alexander and Marquardt (1998), and Alexander (1999).

A complete list of all available macros is given below. The "template" macros insert templates into the current document of WinEdt. One can jump from one missing entry (marked with "`..`" in the examples below) to the other in order to fill in by pressing `Ctrl+Space`. For each of the marked (+) items, there are two macros. One macro submits information to R and goes back to WinEdt at once (e.g. in order to submit several discontinued regions of code), whereas the other one is intended to change the focus permanently to R after submission:

+ Copy and paste a selected region of code.
+ Copy and paste the current line.
+ Save and `source()` the whole file.
- Save the R history in file `.Rhistory` and open it in R-WinEdt.
- Insert function template: `function( .. ){ .. }`.
- Insert loop template: `for( .. ){ .. }`.
- Insert if template: `if( .. ){ .. }`.
- Insert ifelse template: `if( .. ){ .. } else{ .. }`.

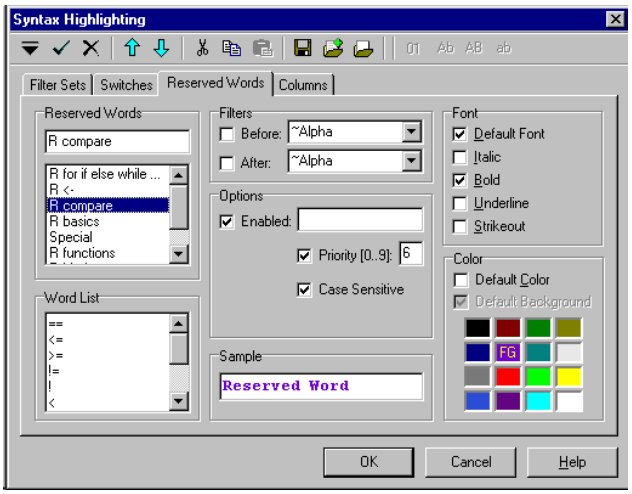

Figure 3: R-WinEdt – configuring the highlighting scheme

3.2 Highlighting

Setting up highlighting schemes is easy in WinEdt, and corresponding rules are highly configurable (cf. Figure 3). The very complete R-WinEdt highlighting scheme distinguishes between the following main groups of "highlighting-classes" (given in descending order):

- Comments: `#` and all stuff thereafter.
- Assignment operators: `<-`, `<<-`, `=`, `->`, and ";".
- Control structure keywords: `if`, `else`, `ifelse`, `while`, `for`, `break`, `repeat`, `next`.
- Logical operators: `==`, `<=`, `>=`, `!=`, `!`, `<`, `>`, `&`, `&&`, `|`, `||`.
- `function`, `library`, `require`
- Special values: `NULL`, `NA`, `NaN`, `Inf`, `F`, `T`, `FALSE`, `TRUE`, and the functions `.Internal` and `.C`.
- Matrix operators: `%%`, `%*%`, `%/%`, `%in%`, `%o%`, `%x%`.
- Contents within quotes and double quotes.
- Extraction of list and slot elements: `$`, `@`.
- Parenthesis, brackets and braces: `()`, `[]`, `{}`.
- A list of known R functions, currently those implemented in all shipped and recommended packages.
- Content within brackets in `library(content)`, `require(content)`.

3.3 Auto-completion

A "Command Completion Wizard" by Holger Danielsson[9] for WinEdt is available at http://www.WinEdt.org/Plugins/complete.php. The wizard is quite easy to use, but not yet integrated into R-WinEdt, mainly because I do not know anything about its licensing.

4 The future

Aiming at the future of R-WinEdt: Is it desirable to improve the plug-in? From my point of view it should be redesigned in order to get Sweave support (Section 4.2). For the computer scientist's point of view the interface needs to be redesigned as well, but not for me being mainly an "user".

4.1 Interface

In Section 3.1 the awkward recent "interface", which works surprisingly stable except for the "RGui is minimized" bug, has been described. Three ways of improved interfaces can be considered:

- Convincing WinEdt's author to integrate DCOM support.
- Writing a DDE interface for R.
- Using socket connections.

For me, the current interface works stable enough, so I will not make much effort. Benefits of possible improvements have to be compared to all the work that is required to implement these improvements. Nevertheless, contributions will be highly appreciated!

4.2 Sweave

Since WinEdt is focused on LaTeX, it seems to be natural to combine this LaTeX functionality with the capabilities of the R-WinEdt plug-in. Literate statistical analysis (for a discussion related to ESS see Rossini, 2001) with R would benefit from combining those capabilities. More specifically, full support of Sweave[10] by Leisch (2002) is desirable for convenient state of the art literate statistical analysis with R:

R-WinEdt was initially written on a Pentium I machine (133MHz, 64MB RAM, Windows NT 4.0), where speed and memory efficiency was still an issue. Unfortunately, it was not possible to combine both R and LaTeX functionality without getting a big penalty in WinEdt's starting times and memory consumption. Today, that is no longer a problem to worry about, so a redesigned plug-in could easily provide a combined interface.

The only issue to think about is the configuration of an adequate highlighting scheme for combined documents such as Sweave files. Thus highlighting must be sensitive to code chunks in a way. And, of course, an installation should not corrupt users' adaptions.

[9] dani@fbg.schwerte.de, he seems to be disappeared

[10] Sweave combines typesetting with LaTeX and statistical data analysis with R into integrated statistical documents.

5 Conclusion

The recent version of R-WinEdt provides the "Windows way" of editing R code – convenient, flexible, and powerful – but it is not quite as powerful as ESS for the Emacs. I think the plug-in should be redesigned in order to implement Sweave support, whereas the implementation of a better interface is of less priority.

I would like to invite the reader to submit contributions and other suggestions to any of these topics.

Appendix

Readme.txt

```
================================================================================
Using WinEdt as an editor for R      *** R-WinEdt 1.5-0 ***       13.07.2003
================================================================================

Overview:
  Using WinEdt as an editor for R (only Rgui under Windows).

Features:
  - Syntax-Highlighting  (and highlighted printing)
    (function index of all "recommended" packages, including:
        base  boot  class  cluster  ctest  eda  foreign  grid  KernSmooth
        lattice  lqs  MASS  methods  mgcv  modreg  mva  nlme  nls  nnet
        rpart  spatial  splines  stepfun  survival  tcltk  tools  ts
     last update: 13.09.2002, R-1.6.0 (beta)).
  - Many R functions at the same time in the same editor.
  - Buttons / Shortcuts for simple access (one click) to:
    - Save document and source(.) into R.
    - Submit script to R (WinEdt as scripting window)
    - Paste selected code from WinEdt directly to R (e.g. from history)
      or just evaluate the current line
    - Open (and update) .Rhistory (so you can reuse your last commands).
  - Templates, e.g.: for(_ in _){_}
  - Faster starting than with standard configuration (of WinEdt).
  - Features of WinEdt (e.g. Delimiter Check, advanced searching, bookmarks,
    macros, ...).
  - Independent of another WinEdt running with "LaTeX-configuration".
  - Works with RGui running in MDI or SDI style and also with S-PLUS
    (not supported!).

Installation:
  - Install R (rw1020 or later)
  - Install WinEdt 5 (V. 5.2, Build: 20001213 or later)
    Note: WinEdt is Shareware (http://www.WinEdt.com).
  - Now, there are two ways, the first one (a) is recommended:

  a) Installation as an R package:
    - Install the Omegahat package 'SWinRegistry' (required), available at:
        http://www.Omegahat.org/SWinRegistry
      At the time of writing "SWinRegistry_0.3-1_binary.zip" is current.
      The R GUI can be used:
```

```
    "Packages" -> "Install package(s) from local zip files..."
  - Install this archive (RWinEdt_.....zip)
    The R GUI can be used:
    "Packages" -> "Install package(s) from local zip files..."
  - All further steps will be done automatically by typing
        > library(RWinEdt)
    in the R Console. (R-WinEdt will be started, and an additional menu
    "R-WinEdt" will be created)
  - If you are running RGui in multiple windows style (MDI), the default
    mode should be unchanged. For RGui in single window style (SDI), or if
    you want to use R-WinEdt with S-PLUS, you can change the R-WinEdt mode
    permanently(!) using the Menu, e.g.: R -> Set R --sdi mode


  Examples:
  ###### Recommended procedure:
  ## Load the package:
   library(RWinEdt)    # within R!
  ## Then create a new document and write an R function.
  ## Click on the symbol "R source" or press ALT+S.
  ## You will be asked to specify a filename (e.g. "R-prog1.R").
  ## If RGui is running, it will be focussed and source(.) will be called.
  ######
   file.show(".Rhistory")
  ## Mark some lines, click the "R paste" button or use Alt+P as shortcut:
  ## Marked lines will be executed in RGui.
  ######
  ## The following way to edit function is possible, but not recommended:
   my.legend <- legend
   fix(my.legend)

b) Alternative: Manual installation procedure
  - Unzip the archive and copy the sub-directory PlugIn into directory
    ....\winedt\plugins
    (e.g. "c:\program files\winedt team\winedt\plugins\PlugIn")
  - In your Windows-Explorer double-click on "install.bat".
  - If you are running RGui in multiple windows style (MDI), the default
    mode should be unchanged. For RGui in single window style (SDI), or if
    you want to use R-WinEdt with S-PLUS, you can change the R-WinEdt mode
    permanently(!) using the Menu, e.g.:  R -> Set R --sdi mode
  - To invoke R-WinEdt from startmenu or desktop create a shortcut to
    WinEdt as follows:
    "c:\program files\winedt team\winedt\winedt" -C="R-WinEdt" -e=r.ini
  - In R use something like (for example in your .Rprofile):
     options(editor="\"c:/program files/winedt team/winedt/winedt\"
        -c=\"R-WinEdt-edit\" -e=r.ini -V")
     options(pager="\"c:/program files/winedt team/winedt/winedt\"
        -C=\"R-WinEdt\" -e=r.ini -V")

  Used WinEdt parameters (There is a difference between -c and -C !):
      -c="name": New instance of WinEdt called "name" will be started.
      -C="name": If an instance "name" of WinEdt is already running, it
                 will be used. So you can run WinEdt as LaTeX and R
                 editor at the same time.
```

```
        -e="name": Using "name" as initialization-file.
        -V       : Running in "virgin"-mode

    Examples:
    ###### Recommended procedure:
    ## Open WinEdt with something like:
     "c:\program files\winedt team\winedt\winedt" -C="R-WinEdt" -e=r.ini
    ## i.e. ideally the shortcut you have already created.
    ##
    ## Then create a new document and write an R function.
    ## Click on the symbol "R source" or press ALT+S.
    ## You will be asked to specify a filename (e.g. "R-prog1.R").
    ## If RGui is running, it will be focussed and source(.) will be called.
    ######
     options(pager="\"c:/program files/winedt team/winedt/winedt\"
        -C=\"R-WinEdt\" -e=r.ini -V")
     file.show(".Rhistory")
    ## Mark some lines, click the "R paste" button or use Alt+P as shortcut:
    ## Marked lines will be executed in RGui.
    ######
    ## The following way to edit function is possible, but *not* recommended:
     options(editor="\"c:/program files/winedt team/winedt/winedt\"
        -c=\"R-WinEdt-edit\" -e=r.ini -V")
     my.legend <- legend
     fix(my.legend)

Known bugs:
  - If RGui is minimized, it is impossible to send commands from WinEdt to
    RGui. Workaround: Don't minimize RGui! ;-)
  - Works in S-PLUS mode only, if the command window is opened as "window 1"
  - Doesn't work with all versions of S-PLUS (tested with version 4.5).
```

References

J. Alexander. *The WinEdt Hacker's Guide*, 11 1999. URL http://www.WinEdt.org/.

J. Alexander and C. Marquardt. *The WinEdt Hacker's Guide*, 11 1998. URL http://www.WinEdt.com/.

K. Hornik. *The R FAQ*, Version 1.7-3. R–Project, 2003. URL http://CRAN.R-project.org/manuals.html.

R. Ihaka and R. Gentleman. R: A Language for Data Analysis and Graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

F. Leisch. *Sweave User Manual*. Institut für Statistik und Wahrscheinlichkeitstheorie, Technische Universität Wien, Vienna, Austria, 2002. URL http://www.ci.tuwien.ac.at/~leisch/Sweave. R Version 1.6.0.

E. Neuwirth and T. Baier. Embedding R in Standard Software, and the other way round. In Kurt Hornik and Friedrich Leisch, editors, *Proceedings of the 2nd International Workshop on Distributed Statistical Computing, March 15–17*,

Vienna, 2001. Technische Universität Wien. URL http://www.ci.tuwien.ac.at/Conferences/DSC-2001/Proceedings/.

A. Rossini. Literate Statistical Analysis. In Kurt Hornik and Friedrich Leisch, editors, *Proceedings of the 2nd International Workshop on Distributed Statistical Computing, March 15–17*, Vienna, 2001. Technische Universität Wien. URL http://www.ci.tuwien.ac.at/Conferences/DSC-2001/Proceedings/.

A. Rossini, R. M. Heiberger, R. Sparapani, M. Mächler, and K. Hornik. Emacs Speaks Statistics: a multi-platform, multi-package development environment for statistical analysis. *Journal of Computational and Graphical Statistics*, 2003. Forthcoming.

R. M. Stallmann. *The Emacs Editor*. Boston, 1999. URL http://www.gnu.org. Version 20.7.

W. N. Venables, D. M. Smith, and the R Development Core Team. *An Introduction to R*, Version 1.7.0. R–Project, 2003. URL http://CRAN.R-project.org/manuals.html.

Affiliation

Uwe Ligges
SFB 475 "Reduction of Complexity for Multivariate Data Structures"
Fachbereich Statistik
Universität Dortmund
Germany
E-mail: ligges@statistik.uni-dortmund.de