

Chapter 1

Frequently Asked Questions

1.1 For All Zelig Users

How do I cite Zelig?

We would appreciate if you would cite Zelig as:

Imai, Kosuke, Gary King and Olivia Lau. 2006. “Zelig: Everyone’s Statistical Software,” <http://GKing.Harvard.Edu/zelig>.

Please also cite the contributors for the models or methods you are using. These citations can be found in the contributors section of each model or command page.

Why can’t I install Zelig?

You must be connected to the internet to install packages from web depositories. In addition, there are a few platform-specific reasons why you may have installation problems:

- **On Windows:** If you are using the very latest version of R, you may not be able to install Zelig until we update Zelig to work on the latest release of R. If you wish to install Zelig in the interim, check the Zelig release notes (Section ??) and download the appropriate version of R to work with the last release of Zelig. You may have to manually download and install Zelig.
- **On Mac:** If the latest version of Zelig is not yet available at CRAN but you would like to install it on your Mac, try typing the following at your R prompt:

```
install.packages("Zelig", repos = "http://gking.harvard.edu", type = "source")
```

- **On Mac or Linux systems:** If you get the following warning message at the end of your installation:

```
Installation of package VGAM had non-zero exit status in ...
```

this means that you were not able to install VGAM properly. Make sure that you have the g77 Fortran compiler. For PowerPC Macs, download g77 from <http://hpc.sourceforge.net>). For Intel Macs, download the xcode Apple developer tools. After installation, try to install Zelig again.

Why can't I install R?

If you have problems installing R (rather than Zelig), you should check the R FAQs for your platform. If you still have problems, you can search the archives for the R help mailing list, or email the list directly at r-help@stat.math.ethz.ch.

Why can't I load data?

When you start R, you need to specify your working directory. In linux R, this is done pretty much automatically when you start R, whether within ESS or in a terminal window. In Windows R, you may wish to specify a working directory so that you may load data without typing in long directory paths to your data files, and it is important to remember that *Windows* R uses the *Linux* directory delimiter. That is, if you right click and select the "Properties" link on a Windows file, the slashes are backslashes (\), but Windows R uses forward slashes (/) in directory paths. Thus, the Windows link may be `C:\Program Files\R\R-2.5.1\`, but you would type `C:/Program Files/R/R-2.5.1/` in Windows R.

When you start R in Windows, the working directory is by default the directory in which the R executable is located.

```
# Print your current working directory.  
> getwd()
```

```
# To read data not located in your working directory.  
> data <- read.table("C:/Program Files/R/newwork/mydata.tab")
```

```
# To change your working directory.  
> setwd("C:/Program Files/R/newwork")
```

```
# Reading data in your working directory.  
> data <- read.data("mydata.tab")
```

Once you have set the working directory, you no longer need to type the entire directory path.

Where can I find old versions of Zelig?

For some replications, you may require older versions of Zelig.

- **Windows** users may find old binaries at <http://gking.harvard.edu/bin/windows/contrib/> and selecting the appropriate version of R.

- **Linux** and **MacOSX** users may find source files at <http://gking.harvard.edu/src/contrib/>

If you want an older version of Zelig because you are using an older version of R, we strongly suggest that you update R and install the latest version of Zelig.

Some Zelig functions don't work for me!

If this is a new phenomenon, there may be functions in your namespace that are overwriting Zelig functions. In particular, if you have a function called `zelig`, `setx`, or `sim` in your workspace, the corresponding functions in Zelig will not work. Rather than deleting things that you need, R will tell you the following when you load the Zelig library:

```
Attaching package: 'Zelig'
The following object(s) are masked _by_ '.GlobalEnv':
  sim
```

In this case, simply rename your `sim` function to something else and load Zelig again:

```
> mysim <- sim
> detach(package:Zelig)
> library(Zelig)
```

Who can I ask for help? How do I report bugs?

If you find a bug, or cannot figure something out, please follow these steps: (1) Reread the relevant section of the documentation. (2) Update Zelig if you don't have the current version. (3) Rerun the same code and see if the bug has been fixed. (4) Check our list of frequently asked questions. (5) Search or browse messages to find a discussion of your issue on the `zelig` listserv.

If none of these work, then if you haven't already, please (6) subscribe to the Zelig listserv and (7) send your question to the listserv at zelig@lists.gking.harvard.edu. Please explain exactly what you did and include the full error message, including the `traceback()`. You should get an answer from the developers or another user in short order.

How do I increase the memory for R?

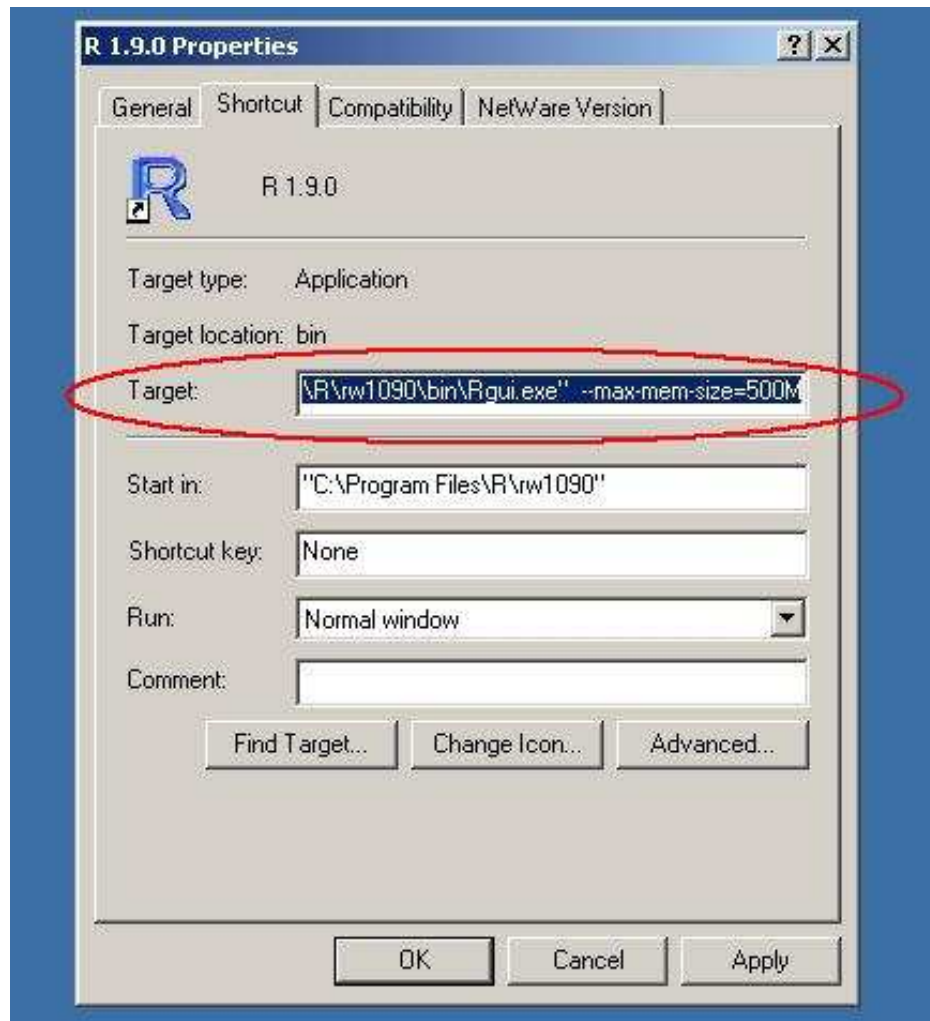
Windows users may get the error that R has run out of memory.

If you have R already installed and subsequently install more RAM, you may have to reinstall R in order to take advantage of the additional capacity.

You may also set the amount of available memory manually. Close R, then right-click on your R program icon (the icon on your desktop or in your programs directory). Select "Properties", and then select the "Shortcut" tab. Look for the "Target" field and after the closing quotes around the location of the R executable, add

```
--max-mem-size=500M
```

as shown in the figure below. You may increase this value up to 2GB or the maximum amount of physical RAM you have installed.



If you get the error that R cannot allocate a vector of length x, close out of R and add the following line to the “Target” field:

```
--max-vsize=500M
```

or as appropriate.

You can always check to see how much memory R has available by typing at the R prompt

```
> round(memory.limit())/2^20, 2)
```

which gives you the amount of available memory in MB.

Why doesn't the pdf print properly?

Zelig uses several special L^AT_EX environments. If the pdf looks right on the screen, there are two possible reasons why it's not printing properly:

- Adobe Acrobat isn't cleaning up the document. Updating to Acrobat Reader 6.0.1 or higher should solve this problem.
- Your printer doesn't support PostScript Type 3 fonts. Updating your print driver should take care of this problem.

R is neat. How can I find out more?

R is a collective project with contributors from all over the world. Their website (<http://www.r-project.org>) has more information on the R project, R packages, conferences, and other learning material.

In addition, there are several canonical references which you may wish to peruse:

Venables, W.N. and B.D. Ripley. 2002. *Modern Applied Statistics with S*. 4th Ed. Springer-Verlag.

Venables, W.N. and B.D. Ripley. 2000. *S Programming*. Springer-Verlag.

1.2 For Zelig Contributors

Where can I find the source code for Zelig?

Zelig is distributed under the GNU General Public License, Version 2. After installation, the source code is located in your R library directory. For Linux users who have followed our installation example, this is `~/R/library/Zelig/`. For Windows users under R 2.5.1, this is by default `C:\Program Files\R\R-2.5.1\library\Zelig\`. For Macintosh users, this is `~/Library/R/library/Zelig/`.

In addition, you may download the latest Zelig source code as a tarball'ed directory from <http://gking.harvard.edu/src/contrib/>. (This makes it easier to distinguish functions which are run together during installation.)

How can I make my R programs run faster?

Unlike most commercial statistics programs which rely on precompiled and pre-packaged routines, R allows users to program functions and run them in the same environment. If you notice a perceptible lag when running your R code, you may improve the performance of your programs by taking the following steps:

- Reduce the number of loops. If it is absolutely necessary to run loops in loops, the inside loop should have the most number of cycles because it runs faster than the outside loop. Frequently, you can eliminate loops by using vectors rather than scalars. Most R functions deal with vectors in an efficient and mathematically intuitive manner.

- Do away with loops altogether. You can vectorize functions using the `apply`, `mapply()`, `sapply()`, `lapply()`, and `replicate()` functions. If you specify the function passed to the above `*apply()` functions properly, the R consensus is that they should run significantly faster than loops in general.
- You can compile your code using C or Fortran. R is not compiled, but can use bits of precompiled code in C or Fortran, and calls that code seamlessly from within R wrapper functions (which pass input from the R function to the C code and back to R). Thus, almost every regression package includes C or Fortran algorithms, which are locally compiled in the case of Linux systems or precompiled in the case of Windows distributions. The recommended Linux compilers are gcc for C and g77 for Fortran, so you should make sure that your code is compatible with those standards to achieve the widest possible distribution.

Which compilers can I use with R and Zelig?

In general, the C or Fortran algorithms in your package should compile for any platform. While Windows R packages are distributed as compiled binaries, Linux R compiles packages locally during installation. Thus, to ensure the widest possible audience for your package, you should make sure that your code will compile on gcc (for C and C++), or on g77 (for Fortran).