cox.zph function provides formal tests and graphical diagnostics for proportional hazards

```
> cox.zph(mayomodel)
                   rho chisq      p
edtrt          -0.1602 3.411 0.0648
log(bili)       0.1507 2.696 0.1006
log(protime)   -0.1646 2.710 0.0997
age            -0.0708 0.542 0.4617
platelet       -0.0435 0.243 0.6221
GLOBAL              NA 9.850 0.0796

## graph for variable 1 (edtrt)
> plot(cox.zph(mayomodel)[1])
```

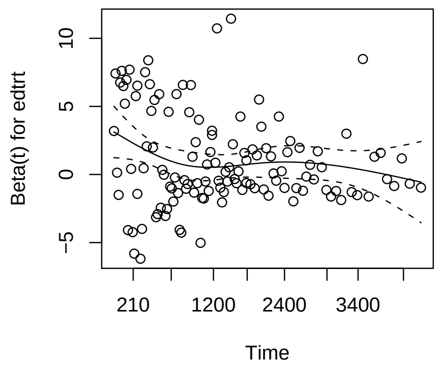

Figure 3: Testing proportional hazards

There is a suggestion that edtrt and log(protime) may have non-proportional hazards, and Figure 3 confirms this for edtrt. The curve shows an estimate of β at each point in time, obtained by smoothing the residuals on the graph. It appears that edtrt is quite strongly predictive for the first couple of years but has little predictive power after that.

Additional features

Computation for the Cox model extends straightforwardly to situations where z can vary over time, and when an individual can experience multiple events of the same or different types. Interpretation of the parameters becomes substantially trickier, of course. The coxph() function allows formulas to contain a cluster() term indicating which records belong to the same individual and a strata() term indicating subgroups that get their own unspecified baseline hazard function.

Complementing coxph is survreg, which fits linear models for the mean of survival time or its logarithm with various parametric error distributions. The parametric models allow more flexible forms of censoring than does the Cox model.

More recent additions include penalised likelihood estimation for smoothing splines and for random effects models.

The survival package also comes with standard mortality tables for the US and a few individual states, together with functions for comparing the survival of a sample to that of a population and computing person-years of followup.

Thomas Lumley
Department of Biostatistics
University of Washington, Seattle

useR! 2004

The R User Conference

by John Fox
McMaster University, Canada

More than 200 R users and developers converged on the Technische Universität Wien for the first R users' conference — userR! 2004 — which took place in Vienna between May 20 and May 22. The conference was organized by the Austrian Association for Statistical Computing (AASC) and sponsored by the R Foundation for Statistical Computing, the Austrian Science Foundation (FWF), and MedAnalytics (http://www.medanalytics.com/). Torsten Hothorn, Achim Zeileis, and David Meyer served as chairs of the conference, program committee, and local organizing committee; Bettina Grün was in charge of local arrangements.

The conference program included nearly 100 presentations, many in keynote, plenary, and semi-plenary sessions. The diversity of presentations — from bioinformatics to user interfaces, and finance to fights among lizards — reflects the wide range of applications supported by R.

A particularly interesting component of the program were keynote addresses by members of the R core team, aimed primarily at improving programming skills, and describing key features of R along with new and imminent developments. These talks reflected a major theme of the conference — the close relationship in R between use and development. The keynote addresses included:

- Paul Murrell on grid graphics and programming
- Martin Mächler on good programming practice

- Luke Tierney on namespaces and byte compilation
- Kurt Hornik on packaging, documentation, and testing
- Friedrich Leisch on S4 classes and methods
- Peter Dalgaard on language interfaces, using `.Call` and `.External`
- Douglas Bates on multilevel models in R
- Brian D. Ripley on datamining and related topics

Slides for the keynote addresses, along with abstracts of other papers presented at userR! 2004, are available at the conference web site, `http://www.ci.tuwien.ac.at/Conferences/useR-2004/program.html`.

An innovation of useR! 2004 was the deployment of "island" sessions, in place of the more traditional poster sessions, in which presenters with common interests made brief presentations followed by general discussion and demonstration. Several island sessions were conducted in parallel in the late afternoon on May 20 and 21, and conference participants were able to circulate among these sessions.

The conference sessions were augmented by a lively social program, including a pre-conference reception on the evening of May 19, and a conference trip, in the afternoon of May 22, to the picturesque Wachau valley on the Danube river. The trip was punctuated by a stop at the historic baroque Melk Abbey, and culminated in a memorable dinner at a traditional 'Heuriger' restaurant. Of course, much lively discussion took place at informal lunches, dinners, and pub sessions, and some of us took advantage of spring in Vienna to transform ourselves into tourists for a few days before and after the conference.

Everyone with whom I spoke, both at and after useR! 2004, was very enthusiastic about the conference. We look forward to an even larger useR! in 2006.

R Help Desk

Date and Time Classes in R

Gabor Grothendieck and Thomas Petzoldt

Introduction

R-1.9.0 and its contributed packages have a number of datetime (i.e. date or date/time) classes. In particular, the following three classes are discussed in this article:

- *Date*. This is the newest R date class, just introduced into R-1.9.0. It supports dates without times. Eliminating times simplifies dates substantially since not only are times, themselves, eliminated but the potential complications of time zones and daylight savings time vs. standard time need not be considered either. Date has an interface similar to the POSIX classes discussed below making it easy to move between them. It is part of the R base package. Dates in the Date class are represented internally as days since January 1, 1970. More information on Date can be found in `?Dates`. The percent codes used in character conversion with Date class objects can be found in `strptime` (although `strptime` itself is not a Date method).

- *chron*. Package **chron** provides dates and times. There are no time zones or notion of daylight vs. standard time in chron which makes it simpler to use for most purposes than date/time packages that do employ time zones. It is a contributed package available on CRAN so it must be installed and loaded prior to use. Datetimes in the **chron** package are represented internally as days since January 1, 1970 with times represented by fractions of days. Thus 1.5 would be the internal representation of noon on January 2nd, 1970. The **chron** package was developed at Bell Labs and is discussed in James and Pregibon (1993).

- *POSIX classes*. POSIX classes refer to the two classes POSIXct, POSIXlt and their common super class POSIXt. These support times and dates including time zones and standard vs. daylight savings time. They are primarily useful for time stamps on operating system files, data bases and other applications that involve external communication with systems that also support dates, times, time zones and daylight/standard times. They are also useful for certain applications such as world currency markets where time zones are important. We shall refer to these classes collectively as the POSIX classes. POSIXct datetimes are represented as seconds since January 1, 1970 GMT while POSIXlt datetimes are represented by a list of 9 components plus an optional tzone attribute. POSIXt is a common superclass of