Conclusions

For batch production of dynamic `.pdf` or `.ps` documentation, the R/Sweave/LaTeX combination is powerful and nearly limitless in its capabilities. A major benefit to producing reports using this combination of tools is how closely Sweave integrates the processing power of R with the typesetting capability of LaTeX. The key to producing dynamic reports for a large number of recipients is the use of iterative control structures in R. This article provides the author's "homebrew" code. Other, more elegant, solutions are likely possible.

A next step after report generation is report distribution. In theory, given the appropriate server configuration, it should be possible to electronically distribute the reports to the appropriate recipient based upon, for example, an email address contained in the database. I would appreciate learning how others have addressed this and similar problems.

Acknowledgments

I would like to thank Bill Oliver for introducing me to R and for his gracious help. In addition, I would like to thank one of the reviewers for providing particularly insightful improvements to my code. Finally, I would like to express gratitude to the developers/contributors of R, Sweave, and LaTeX, without their efforts none of this would be possible.

Bibliography

F. Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. In W. Härdle and B. Rönz, editors, *Compstat 2002—Proceedings in Computational Statistics*, pages 575–580, Heidelberg, Germany, 2002a. Physika Verlag. ISBN 3-7908-1517-9. URL http://www.ci.tuwien.ac.at/~leisch/Sweave. 40

F. Leisch. Sweave, Part I: Mixing R and LaTeX. *R News*, 2(3):28–31, December 2002b. URL http://cran.r-project.org/doc/Rnews/Rnews_2002-3.pdf. 40

F. Leisch. Sweave, Part II: Package Vignettes. *R News*, 3(2):21–24, October 2003. URL http://cran.r-project.org/doc/Rnews/Rnews_2003-2.pdf. 40

F. Leisch. *Sweave User Manual*, 2005. 42

R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. URL http://www.R-project.org. 3-900051-07-0. 40

T. Tantau. *User's Guide to the Beamer Class*. Sourceforge.net, 2004. URL http://latex-beamer.sourceforge.net. 42

Damian Betebenner
Lynch School of Education
Educational Research, Measurement and Evaluation
Boston College
Damian.Betebenner@bc.edu

The Value of R for Preclinical Statisticians

Bill Pikounis and Andy Liaw
Merck & Co., Inc.

Participation on the R-help mailing list has shown R to be widely used across organizations of all types the world over. This article discusses one corner where R has become an indispensable tool for effective statistical practice: a preclinical pharmaceutical environment at the authors' place of employment.

Preclinical is defined here as a portion of the research and development process for prescription medicines. In this article that portion is defined to start with fundamental discovery, and to end up where a potential product is first put into human subjects in clinical trials. There is a wide variety of knowledge sought across this spectrum, including: biological targets believed to be important influences on a given disease; chemical entities that affect such targets; effectiveness and safety in animals; formulation of product itself. This brief listing is only intended to provide a broad overview. In reality, there are many more areas that could be included based on desired granularity. A complementary term of "non-clinical," which is also commonly used, is perhaps more appropriate here; however, we will stay with the phrase preclinical for brevity.

The hallmark diversity of preclinical research presents unlimited opportunities for statisticians. Data are collected everywhere in the process, and in large quantities. We have the liberty to choose methods of processing and analysis. This is a different environment than clinical biostatistics, where SAS is the dominant tool for various reasons. Below we present examples and discussion on how and why we use R and find it invaluable.

Daily operations

Our corporate-issued laptops run Windows XP and we have a small cluster of dual-processor Opterons that run 64-bit Linux for intensive jobs. The client-server configuration of VNC (TightVNC , 2004) provides a very stable way to bridge the two environments onto one monitor. An X-window session appears as just another window on the Windows desktop. Samba (Samba Team , 1992-2005) sets up file sharing so that a `/home` directory share under Linux appears as a network drive in Windows. The autocutsel (Witrant , 2004) utility eases cut-and-paste operations between the two operating systems.

If we work on data that can be comfortably explored on our laptops, we stay there. Generating reports for our research collaborators is helped by the right-click cutting and default pasting of a Windows metafile from the `windows` graphics device into a Microsoft Office document. The internal Data Editor (see `?edit`) is very useful for browsing a data frame. Once a workspace has been saved in a Windows folder, launching it from Explorer automatically sets the container working directory. The overall R GUI console is thankfully unobtrusive, and we well appreciate the separation of it from graphics device windows in SDI mode. We find it much more pleasant to switch amongst desktop windows where each stands equally on its own, rather than having to search inside a set of windows within the same application.

If an analysis task becomes too large (in memory or time) for the laptops, it is a simple matter to transfer the R workspace image to a Linux box and pick up right where we left off. Through the aforementioned network drive mapping of a Linux share, source code editing can be done in the same Windows editor, and file operations of saving, copying, opening, etc. are transparently done through the standard Windows GUI.

All these task-oriented details add up to increased productivity. More importantly, however, is the content of the environment itself. We point out here that some of what we say in the rest of the article is not necessarily unique to R, and such benefits can be found in the other implementation of the S language, namely S-PLUS. In his "Exegeses on Linear Models" talk (Venables , 2000), one exegesis from Bill Venables that we particularly admire, and paraphrase here, is that "the subject should dictate what the program should do and not the other way around." Our work environment streams a constant wealth of different data structures and types of scientific questions to address, and we can exercise our freedom to choose our strategies of analysis. Comprehensive data analyses are only limited by our own human capabilities, not by R. R enables us to seamlessly move from data management to exploration to formal evaluations to communications of results. No excuse remains to prevent the production of good data graphs for the purposes of valuable study or presentation. The implementation of modern methods allows us to use resampling, resistance/robustness, and dimension reduction methods on a routine basis, to name a few. We work with cutting edge science, which we firmly believe deserves cutting edge data analysis. R promotes this for us, with its virtues of flexibility, stability and efficiency so clearly practiced by its R Core caretakers.

Professor Brian D. Ripley, in "How Computing has Changed Statistics" (Ripley , 2004), aptly states of R that "the only barrier to understanding how it works, precisely, is skill." We interpret the meaning of "how it works" to be on multiple levels, and the level most important for us is that a true, continuous investment to learn its many functions, structures, and features pays off time and again. It helps us bring value to scientific research, gain trust from our collaborators, and stimulates the intellect.

Illustrations

Preclinical statisticians do not get nearly enough opportunities to provide input to design of experiments. When we do get such an opportunity, we are especially eager to make recommendations as quickly as possible. A recent query involved a crossover design with binary responses. An important part of the design plan was to estimate the number of subjects needed to see a meaningful difference between two conditions, where incidence rates were low. Sample size determination tends to be the predominant aspect of such study design requests.

Estimating sample size is a process of several approximations, but with some effort such approximations do not substantially diminish its value. In this example, tracking down software that could be readily used, or finding a literature reference where a recipe could be programmed, was not timely enough. So a simulation-based approach was taken.

```
calcPower <- function(nsubj, pyn, pny, pnnyy,
                      numsim=1000) {
  ## A simple approach to calculate the power of
  ## McNemar's matched-pair test for these inputs:
  ##  nsubj = the number of subjects/pairs of
  ##   measurements (msmts)
  ##   (note that msmt1 comes from condition 1 and
  ##    msmt2 comes from condition 2)
  ##  pyn = p(msmt1 = yes & msmt2 = no)
  ##  pny = p(msmt1 = no & msmt2 = yes)
  ##  pnnyy = p(msmt1 != msmt2)
  ##  numsim = Number of Simulations
  ##   (at least 1000 recommended)
  outcomes <- rmultinom(n=numsim, size=nsubj,
                        prob=c(pnn=pnnyy, pyy=0,
                          pny=pny, pyn=pyn))
  tscompares <-
    apply(outcomes, 2, function(x) {
```

```
      ts <- (((x[3] - x[4])^2) / (x[3] + x[4]))
      if (!is.finite(ts)) ts <- 0
      ts
    })
  mean(tscompares > qchisq(0.95, 1), na.rm=FALSE)
}
```

This simple-minded function evaluates how McNemar's test would behave for a given configuration of underlying parameters, thereby providing an estimate of its power to detect that difference. Recall that small occurrence rates were expected. A grid of potential configurations can be evaluated:

```
powerGrid <-
  expand.grid(pyn=seq(0.01, 0.10, by=.01),
              pny=c(0.001, 0.002, 0.005,
                    seq(0.01, 0.09, by=.01)))
powerGrid$pnnyy <- (1 - powerGrid$pyn -
                    powerGrid$pny)
powerGrid <- subset(powerGrid, pyn > pny)
powerGrid <- powerGrid[order(powerGrid$pyn,
                             powerGrid$pny),]
powerGrid$power <-
  apply(powerGrid, 1,
        function(x) calcPower(100, x[1],
                              x[2], x[3]))
```

From the resulting data frame `powerGrid`, the context of the study objectives and available resources, a variety of alternatives were explored and recommended to the researcher. Ensuing discussions then produced a design that was later successfully implemented.

We realize that potential influences of period and sequence were ignored in the above calculations, as well as perhaps a more suitable parametric model approach to the postulated analysis. But as we mentioned previously, we felt the approximations in the approach were reasonable enough to address the questions at hand.

Similar sample size scenarios we have encountered include (1) survival data with simple Type I censoring; (2) single population binomial parameter estimation; (3) inspection of device performance within its specified tolerances; and (4) comparability of old and new drug formulations. In all these scenarios the principal recipe of simulating power remains the same as the example above; the key differences include the postulated underlying distribution and the methodology of estimation or testing.

One could argue the drawback of computational time that might be needed to generate sufficient grids of sample size and power values. In the above example, a `nsim=10000` or `100000` would provide better estimates of power and the computational time is not prohibitive; we are talking minutes and at most hours to get the needed results. Here is where our small cluster of Linux servers, or simply scheduling something to run overnight, takes advantage of what computers are really good at.

Molecular Modeling

As another illustration, one particular area where our group has been involved in is molecular modeling. Specifically, we are referring to the problem of predicting biological activities of small organic molecules from their chemical "descriptors". The biological activities can be quantitative (e.g., percent inhibition against a target enzyme) or qualitative ("active" vs. "inactive"). The chemical descriptors are properties/features of the molecules computed from their structures. There are two possible goals. One is simply prediction: Given a collection of molecules with unknown biological activities, predict which ones are likely to be active. The other possible goal is interpretation: What chemical properties or substructures are biologically relevant?

Our computational chemistry colleagues have traditionally used techniques such as *k*-nearest neighbors, partial least squares, and neural networks, etc. for such problems, mostly using tools written in-house. A couple of years ago, we started to convince our colleagues that more modern, powerful tools such as random forests (Breiman, 2001; Svetnik et. al. , 2003) can be readily accessible through R. We started working on linking Breiman and Cutler's Fortran code to R after we got tired of using the Fortran code alone, since every little change in the data or parameters required recompiling the source code. It would have been impossible to convince our colleagues to use the tool in that form, no matter how powerful the tool may be. As a result of making random forests available in R, it has become an important component of the methodologies utilized by our colleagues.

Currently, R is installed on the main computer system for the computational chemists, who are invariably Unix based. Whatever R functionalities are required, the development group (who are responsible for research and implementation of new methodologies) would wrap them in shell or Perl scripts. These scripts are then used by the applications group to support specific projects.

Delivery of Tools

Merck preclinical statisticians are outnumbered at least ten to one by potential researchers to collaborate with, and we have global sites that we strive to serve since they have no access to local statisticians. As briefly alluded to in the previous section, the availability of R to communicate with other software offers great potential to serve our customers.

We have recently embarked on a COM-driven[1] framework to take advantage of existing infrastruc-

[1]COM stands for Common Object Model, a Microsoft-driven "standard" for communications between software components.

ture. For instance, virtually everyone at Merck runs the same version of Windows XP and the same version of Microsoft Excel. Use of R(D)COM (Baier and Neuwirth , 2004) allows construction of an application where the user interface is entirely in Excel; namely, input data storage, selection of data and choices through Userforms, and formatted output. The underlying work of the R engine is invisible to the user as it provides calculations and the raw output to Excel for the formatting and the organized presentation. This framework leverages Excel strengths of formatting, familiarity, and ubiquity, and R provides numerical reliability and breadth of data analytic functionality. While we are currently in early stages, the framework has demonstrated reliability for widespread distribution.

Summary

There are several specific aspects that make R uniquely suitable for us and our colleagues:

- Availability of modern methodologies;
- Flexibility for implementing new methods;
- Facilities to package added functionalities;
- Seamless integration with other software;
- Liberty to (try to) install on any platform.

The first point is discussed above. We add that colleagues in our department rely on R for their support of genomic and proteomic research, where access to (or the ability to implement) cutting-edge methodologies is crucial. The second through fourth points are important for us as tool developers. The fact that R is a full-featured language enables us to follow the spirit of "turn ideas into software, quickly and faithfully" (Chambers , 1998). The packaging facility in R lets us easily create, maintain, and distribute tools and associated documentation. The same cannot really be said about most other statistical languages or packages. The COM framework discussed above is but one of many options for integration of R with processes or GUIs, etc. The last point is important not because R is free (as in beer), but because we are not limited to run the software on whatever platform a vendor chooses to support. As long as we can get R to compile and pass all its tests, we are comfortable using it. As an example, when we bought our first 64-bit machine (a dual Opteron 244 with 16GB of RAM), the main motivation was to overcome the memory limitation of a 32-bit platform. Because we can build R from source, we readily built a 64-bit version of R on that box.

This article focused on preclinical statistics use of R. Nearly four years ago we organized our first in-house introductory course on R, and about 20 people attended, mostly from preclinical statistics and other basic research departments that provide quantitative analysis support to researchers. As a sign of the increasing acceptance of R at Merck three years later, attendance more than doubled and mostly included clinical statisticians and SAS programmers. Results based on the use of R by preclinical statisticians have been included in regulatory responses and filings, and no issues have arisen. We are not aware of R use in the traditional core workflows of clinical trials production work to date, but we do expect the use of R within Merck to continue to increase in all areas, including clinical.

In our small corner of the world that is preclinical drug research & development, R is a guide towards better statistical practice and helps expands our influence on scientific research. It is indispensable. If R— the software and community — did not exist, we would be wishing that something like it would come along.

Footnote: The first author recently changed employment from Merck to Centocor. The preclinical/nonclinical experiences with researchers there have been quite similar to date.

Bibliography

T. Baier and E. Neuwirth (2004) *R (D)COM Server V1.35* URL http://cran.r-project.org/contrib/extra/dcom/RSrv135.html 47

L. Breiman (2001) Random forests *Machine Learning*, 45, 5–32. 46

J. M. Chambers (1998) *Programming with Data* New York: Springer-Verlag. 47

B. D. Ripley (2004) How Computing Has Changed Statistics (and is changing...) *Symposium in Honour of David Cox's 80th birthday, Neuchatel, July 2004* URL http://www.stats.ox.ac.uk/~ripley/Cox80.pdf 45

Samba URL http://www.samba.org/ 45

V. Svetnik, A. Liaw, C. Tong, C. Culberson, R. P. Sheridan, and B. P. Feuston (2003) Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling *J. Chem. Inf. Comput. Sci.* 43, 1047–1058. 46

TightVNC URL http://www.tightvnc.com/ 45

W. N. Venables (2000) Exegeses on Linear Models *Paper presented to the S-PLUS User's Conference Washington, DC, 8-9th October, 1998* URL http://www.stats.ox.ac.uk/pub/MASS3/Exegeses.pdf 45

M. Witrant (2004) *autocutsel* URL http://savannah.nongnu.org/projects/autocutsel/ 45