tion with the subjects of its analyses. Academics rarely contact authors they critique – indeed, post-publication data repositories aim to make it easy to replicate a study without questioning the authors. Good journalists do the opposite: we contact the subjects of our investigations early and often.

And of course, R let us flexibly change our analysis every time CNBC and Cramer changed their story. When they argued for a different selection, R made it easy to create a new data subset. When they argued for a different holding strategy, R's indexing facility let us start and stop our analysis on different dates. In fact, when I begged CNBC for their own analysis of Cramer's performance, they said something that should warm the hearts of all you folks who've made R the powerful environment it is. CNBC told me not to expect a timely response from them because it was obvious that Pat and I had spent *months* on our analysis. In truth, Pat put in less than a week's work.

Acknowledgments

Essential services for this project included: R coding by Pat Burns, of Burns Statistics in London; Excel macros by Edgar Online analyst Elias-John Kies and from Tufts University undergraduate Madison Mcgaffin; wise statistical pointers from Tim Hesterberg, of Insightful Corp. in Seattle . . . none of whom bear responsibility for any woebegone errors.

Bibliography

B. Alpert. Shorting Cramer. *Barron's*, 2007.

B. Efron and R. Tibshirani. *Introduction to the Bootstrap*. Chapman and Hall, 1993.

J. Engelberg, C. Sasseville, and J. Williams. Attention and Asset Prices: the Case of Mad Money. Technical report, Kellogg School of Management, 2007. `http://papers.ssrn.com`.

N. Nayda. You! Me! Let's Get Ready to Make Mad Money!!! Technical report, Elmont Virginia Elementary School, 2006. Science fair project.

Bill Alpert
Barron's, Dow Jones & Co., U.S.A.
`william.alpert@barrons.com`

Need A Hint?

Sanford Weisberg and Hadley Wickham

Suppose you have created an object in R, for example from a regression fit using `lm` or `loess`. You know that auxiliary functions exist that do useful computations on the object, but you can't remember their names. You need a hint on what to do next.

The `hints` function in the **hints** package does just this, finding a list of appropriate functions to jog your memory. For example, Figure 1 shows a list of hints for a `lm` object.

The output lists methods for generic functions like `print` specific to the class you specify, as well as searching the documentation to find all mentions of the class. You can then use the usual help mechanism to learn more about each of these methods and functions.

The `hints` function has three arguments:

```
hints(x, class=class(x), all.packages=FALSE)
```

If specified, the argument `x` can be any R object. For example, `x` might have been created by `x <- lm(y z)`. `hints` determines the S3 class of the object, and then looks for functions that operate on that class. The S3 class of an object is a character vector, and may consist of multiple strings, as, for example, a generalized linear model which has class `c("glm", "lm")`. If `x` is not given, then you can specify the `class` you want hints about as character vector.

The hints function will look for methods and functions in all currently loaded packages. For example, the hints for `lm` would be different if either the **car** or the **alr3** packages have been loaded, since both of these add methods and functions for `lm` objects. Similarly, `hints(class="lda")` would return methods only if the package **MASS** were loaded, since all the relevant methods and functions are in that package. You can get hints for all your packages by setting `all.packages=TRUE`, but note that this works by `require`'ing all available packages so may be time consuming.

The **hints** package also includes an `xtable` method so, for example, `xtable(hints(m1))` would have produced a version of Figure 1, but in LaTeX format.

The function isn't foolproof, as it depends on the quality of documentation written by others. It may find irrelevant functions if the name of the class appears in the documentation for the irrelevant function. It can miss functions, too. For example, the function `coeftest` in the **lmtest** package can be used with `lm` objects by applying the function `coeftest.default`.

```
> hints(class = "lm")

Functions for lm in package 'base'
kappa                          Estimate the Condition Number
base-defunct                   Defunct Functions in Base Package
Functions for lm in package 'methods'
setOldClass                    Specify Names for Old-Style Classes
Functions for lm in package 'stats'
add1                           Add or Drop All Possible Single Terms to a Model
alias                          Find Aliases (Dependencies) in a Model
anova.lm                       ANOVA for Linear Model Fits
case.names.lm                  Case and Variable Names of Fitted Models
cooks.distance.lm              Regression Deletion Diagnostics
dfbeta.lm                      Regression Deletion Diagnostics
dfbetas.lm                     Regression Deletion Diagnostics
drop1.lm                       Add or Drop All Possible Single Terms to a Model
dummy.coef.lm                  Extract Coefficients in Original Coding
effects                        Effects from Fitted Model
family.lm                      Accessing Linear Model Fits
formula.lm                     Accessing Linear Model Fits
hatvalues.lm                   Regression Deletion Diagnostics
influence.lm                   Regression Diagnostics
labels.lm                      Accessing Linear Model Fits
logLik                         Extract Log-Likelihood
model.frame.lm                 Extracting the "Environment" of a Model Formula
model.matrix.lm                Construct Design Matrices
plot.lm                        Plot Diagnostics for an lm Object
predict.lm                     Predict method for Linear Model Fits
print.lm                       Fitting Linear Models
proj                           Projections of Models
residuals.lm                   Accessing Linear Model Fits
rstandard.lm                   Regression Deletion Diagnostics
rstudent.lm                    Regression Deletion Diagnostics
summary.lm                     Summarizing Linear Model Fits
variable.names.lm              Case and Variable Names of Fitted Models
vcov                           Calculate Variance-Covariance Matrix for a Fitted Model
                               Object
case.names                     Case and Variable Names of Fitted Models
dummy.coef                     Extract Coefficients in Original Coding
influence.measures             Regression Deletion Diagnostics
lm                             Fitting Linear Models
lm.influence                   Regression Diagnostics
lm.fit                         Fitter Functions for Linear Models
model.frame                    Extracting the "Environment" of a Model Formula
model.matrix                   Construct Design Matrices
stats-defunct                  Defunct Functions in Package stats
Functions for lm in package 'unknown'
confint.lm                     NA
deviance.lm                    NA
extractAIC.lm                  NA
simulate.lm                    NA
```

Figure 1: Hints for the `lm` class.

Hints can't figure this out because there is no explicit mention of `lm` in the function or the documentation, and so it misses the function. If the regression had been done using `glm` rather than `lm`, `hints` would have found `coeftest.glm`.

The explanations of what the methods and functions do may be more generic than one might want, if the title of the help page is too generic. In some cases, no explanation is found. For example, `simulate.lm` is shown in Figure 1, but its description is missing. The help page for `simulate` mentions the `lm` class, but no page is available for `simulate.lm`, and so the `hints` function doesn't know where to get documentation. Finally, the hints function can only find hints for S3 objects, not for S4. Nevertheless, this simple function can be a useful tool, if you are willing to take a hint.

Sanford Weisberg
University of Minnesota
`sandy@stat.umn.edu`
Hadley Wickham
Iowa State University
`h.wickham@gmail.com`

Psychometrics Task View

Patrick Mair, Reinhold Hatzinger

Psychometrics is concerned with the design and analysis of research and the measurement of human characteristics. Psychometricians have also worked collaboratively with those in the field of statistics and quantitative methods to develop improved ways to organize and analyze data. In our task view we subdivide "Psychometrics" into the following methodological areas: Item Response Theory (IRT), Correspondence Analysis (CA), Structural Equation Models (SEM) and related methods such as Factor Analysis (FA) and Principal Component Analysis (PCA), Multidimensional Scaling (MDS), Classical Test Theory (CTT), and other approaches related to psychometrics.

Since much functionality is already contained in base R and there is considerable overlap between tools for psychometry and tools described in other views, particularly in `SocialSciences`, we only give a brief overview of packages that are closely related to psychometric methodology. Recently, `Journal of Statistical Software (JSS)` published a special volume on Psychometrics in R in which some new R packages were published. For an overview see de Leeuw and Mair (2007).

Item Response Theory (IRT)

The **eRm** package fits extended Rasch models, i.e. the ordinary Rasch model for dichotomous data (RM), the linear logistic test model (LLTM), the rating scale model (RSM) and its linear extension (LRSM), the partial credit model (PCM) and its linear extension (LPCM) using conditional ML estimation.

The package **ltm** also fits the simple RM. Additionally, functions for estimating Birnbaum's 2- and 3-parameter models based on a marginal ML approach are implemented as well as the graded response model for polytomous data, and the linear multidimensional logistic model.

Item and ability parameters can be calibrated using the package **plink**. It provides various functions for conducting separate calibration of IRT single-format or mixed-format item parameters for multiple groups using the Mean/Mean, Mean/Sigma, Haebara, and Stocking-Lord methods. It includes symmetric and non-symmetric optimization and chain-linked rescaling of item and ability parameters.

The package **plRasch** computes maximum likelihood estimates and pseudo-likelihood estimates of parameters of Rasch models for polytomous (or dichotomous) items and multiple (or single) latent traits. Robust standard errors for the pseudo-likelihood estimates are also computed.

A multilevel Rasch model can be estimated using the package **lme4** with functions for mixed-effects models with crossed or partially crossed random effects.

Other packages of interest are: **mokken** in the JSS special issue as a package to compute non-parametric item analysis, the **RaschSampler** allowing for the construction of exact Rasch model tests by generating random zero-one matrices with given marginals, **mprobit** fitting the multivariate binary probit model, and **irtoys** providing a simple interface to the estimation and plotting of IRT models. Simple Rasch computations such a simulating data and joint maximum likelihood are included in the **MiscPsycho** package.

Gaussian ordination, related to logistic IRT and also approximated as maximum likelihood estimation through canonical correspondence analysis is implemented in various forms in the package **VGAM**.

Two additional IRT packages (for Microsoft Windows only) are available and documented on the JSS site. The package **mlirt** computes multilevel IRT models, and **cirt** uses a joint hierarchically built up likelihood for estimating a two-parameter normal ogive model for responses and a log-normal model for response times.