Automatic Generation of Exams in R

Bettina Grün
Johannes Kepler
Universität Linz

Achim Zeileis
Universität Innsbruck

Abstract

This first introduction to the R package **exams** is a (slightly) modified version of Grün and Zeileis (2009), published in the *Journal of Statistical Software*. It describes how to produce PDF files from exercises in Sweave format. Meanwhile, **exams** has been considerably extended by Zeileis, Umlauf, and Leisch (2014) and beyond to also produce HTML output or e-learning exams for **Moodle**, **OLAT**/**OpenOLAT**, etc. This has resulted in some small changes that are not fully backward-compatible and which are marked with "UPDATE" in the text below.

Package **exams** provides a framework for automatic generation of standardized statistical exams which is especially useful for large-scale exams. To employ the tools, users just need to supply a pool of exercises and a master file controlling the layout of the final PDF document. The exercises are specified in separate Sweave files (containing R code for data generation and LaTeX code for problem and solution description) and the master file is a LaTeX document with some additional control commands. This paper gives an overview of the main design aims and principles as well as strategies for adaptation and extension. Hands-on illustrations – based on example exercises and control files provided in the package – are presented to get new users started easily.

Keywords: exams, multiple choice, arithmetic problems, Sweave, LaTeX, R.

1. Introduction

The introductory statistics lecture at the Wirtschaftsuniversität Wien (WU Wien, `http://www.wu.ac.at/`) is attended each semester by about 1,000–1,500 students (mostly first-year business students). Several lecturers from the Department of Statistics and Mathematics teach this course in parallel. In order to ensure an efficient, consistent, and transparent organization, the course format and all its teaching materials (presentation slides, collections of exercises, exams, etc.) were re-designed in a collaborative effort during 2006/7. Among many other aspects – such as specification of a topic list or definition of learning outcomes, etc. – this re-design encompassed several technological challenges. Hence, the **exams** package was designed to address these challenges and thus facilitate the discussions about the content of the new course. More specifically, **exams** aims to provide software infrastructure for:

- *Scalable exams:* Automatic generation of a large number of different exams in order to provide an individual test to each student.
- *Associated self-study materials:* Collections of exercises and solutions from the same pool of examples.

- *Joint development:* Development and maintenance of a large pool of exercises in a multi-author and cross-platform setting.

Specifically, at WU Wien about 10–15 lecturers were working in small teams of 2–4 people on different chapters for the presentation slides. For each chapter, the corresponding team would also provide suitable exercise templates that could be used for self-study materials, exams, and solutions.

The pool of exercises does not only need to contain different types of exercises, but also variants of the same type to avoid that students learn the solutions "by heart". Correction should be fast and easy. This restricts the suitable types of exercises to those which either have a single number as result which only needs to be checked to assess the correctness, multiple-choice questions, or potentially questions which require only a short text answer. These requirements on maintenance, variation, and correction of exercises led to the following design principles for package **exams**:

- *Maintenance:* Each exercise template is a single file (also just called "exercise").
- *Variation:* Exercises are dynamic documents, containing a problem/solution along with a data-generating process (DGP) so that random samples can be drawn easily.
- *Correction:* Solutions for exercises are either multiple-choice answers (logical vectors), numeric values (e.g., a test statistic or a confidence interval), short text answers (e.g., the appropriate null hypothesis corresponding to a given problem), or combinations of these.

Thus, the DGP of an exercise controls the distribution of possible solutions and can be utilized to make them (approximately) evenly distributed and difficult to "guess" or "learn by heart". In addition to the variability within an exercise, one can add further variation by providing several exercise templates for the same type of problem. Depending on the flexibility of the DGP, the pool of exercises can thus be rather small or needs to be somewhat larger.

Mixing problems/solutions and DGPs for exam generation poses challenges that are similar to those of making data analysis reproducible. Thus, **exams** employs many ideas from literate data analysis (Rossini 2001; Leisch and Rossini 2003), literate programming (Knuth 1992) and reproducible research (de Leeuw 2001). Specifically, it makes extensive use of Sweave (Leisch 2002) for mixing DGPs written in the R system for statistical computing (R Development Core Team 2008) and problem/solution descriptions written in the typesetting system LaTeX (Knuth 1984; Lamport 1994). Thus, the implementation in **exams** is based on (1) independent Sweave files for each exercise interweaving R and LaTeX code, (2) different master LaTeX files controlling the appearance and (3) an implementation of a minimal markup for communication with R plus R functions tying everything together.

The package **exams** that emerged from the developments at WU Wien is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=exams`. To utilize it for exam generation, users need to generate a pool of exercises and supply a master LaTeX file. The package provides several examples for both types of files along with documentation (like this paper) and hands-on examples to get new users started.[1] Hence, the

[1] In version 2 of the **exams** package a new function `exams_skeleton()` has been added to facilitate starting new exams projects. See Section 2.3 of Zeileis *et al.* (2014) or `vignette("exams2", package = "exams")` for more details.

remainder of this paper is organized as follows: Section 2 explains the structure of the Sweave files defining the exercises and demonstrates how the final output file is derived from the original R and LaTeX code. In Section 3, the structure of the master LaTeX file for constructing the final PDF file from the exercise files is presented. Finally Section 4 illustrates how to use the package in practice and how **exams** can be customized. Experience with the development of exercises and the application of the package are discussed in Section 5. Section 6 concludes the paper with a summary.

2. Exercises

Each exercise is contained in a separate Sweave file and typically represents an exemplary application of a statistical procedure. It needs to contain (1) a question and a solution description which are encapsulated in LaTeX environments of the corresponding names and (2) some meta-information about the exercise such as type and the solution. To allow for variability, the data for the exercise can be generated within the Sweave file in a code chunk (typically suppressed in the final document). Meta-information for the examples needs to be provided, e.g., for computing with the solution within R or for generating lists with solutions.

In Table 1, the Sweave file for a simple exercise asking students to compute a one-sample t test statistic is shown for illustration. The four different blocks (R code with DGP, `question` environment, `solution` environment, meta-information) can be easily distinguished. The LaTeX file generated from a call to `Sweave()` is shown in Table 2, and the final compiled PDF output is provided in Table 3.

The R code for the DGP is used as in any other Sweave document: The code chunk is framed by `<<>>=` and `@` and options in the header control formatting (typically suppressing the display in the final document, as in this example). For further information about the application of Sweave and a detailed description of the different options see Leisch (2012a,b).

The core of an exercise are two blocks for the question and the solution in their accordingly named environments. The values generated by the DGP are typically included using `\Sexpr{}` statements. Furthermore, question/solution might contain graphics and/or R output created by Sweave. To include the LaTeX output (see Table 2) in some master LaTeX file, this needs to define the `question` and `solution` environment. This mechanism can be used to control the display, e.g., to suppress the solution in an exam but to display it in associated self-study material (see Section 3 for further details).

Finally, some meta-information needs to be provided. All meta-information commands are in LaTeX style but are actually commented out and hidden in the final output file. There are four commands[2]:

- `\extype{}`: type of question. It may be either `mchoice` for multiple-choice questions, `num` for questions with a numeric result or `string` for questions with a (short) text answer.
- `\exsolution{}`: solution. It contains the numeric solution, a string of zeros and ones for multiple-choice questions or a character string. For confidence intervals, it may also specify two numeric solutions of type `\exsolution{`*lower*`}{`*upper*`}`.

[2]Note that version 2 provides many additional commands, especially for e-learning output and other custom interfaces.

```
<<echo=FALSE, results=hide>>=
## DATA GENERATION
n <- sample(120:250, 1)
mu <- sample(c(125, 200, 250, 500, 1000), 1)
y <- rnorm(n, mean = mu * runif(1, min = 0.9, max = 1.1),
           sd = mu * runif(1, min = 0.02, max = 0.06))

## QUESTION/ANSWER GENERATION
Mean <- round(mean(y), digits = 1)
Var <- round(var(y), digits = 2)
tstat <- round((Mean - mu)/sqrt(Var/n), digits = 3)
@

\begin{question}
A machine fills milk into $\Sexpr{mu}$ml packages. It is suspected that the
machine is not working correctly and that the amount of milk filled differs
from the setpoint $\mu_0 = \Sexpr{mu}$. A sample of $\Sexpr{n}$ packages
filled by the machine are collected. The sample mean $\bar{y}$ is equal to
$\Sexpr{Mean}$ and the sample variance $s^2_{n-1}$ is equal to
$\Sexpr{Var}$.

Test the hypothesis that the amount filled corresponds on average to the
setpoint. What is the absolute value of the $t$~test statistic?
\end{question}

\begin{solution}
The $t$~test statistic is calculated by:
\begin{eqnarray*}
  t & = & \frac{\bar y - \mu_0}{\sqrt{\frac{s^2_{n-1}}{n}}}
  = \frac{\Sexpr{Mean} - \Sexpr{mu}}{\sqrt{\frac{\Sexpr{Var}}{\Sexpr{n}}}}
  = \Sexpr{tstat}.
\end{eqnarray*}
The absolute value of the $t$~test statistic is thus equal to
$\Sexpr{fmt(abs(tstat), 3)}$.
\end{solution}

%% META-INFORMATION
%% \extype{num}
%% \exsolution{\Sexpr{fmt(abs(tstat), 3)}}
%% \exname{t statistic}
%% \extol{0.01}
```

Table 1: A simple Sweave exercise: `tstat.Rnw`.

```
\begin{question}
A machine fills milk into $500$ml packages. It is suspected that the
machine is not working correctly and that the amount of milk filled differs
from the setpoint $\mu_0 = 500$. A sample of $226$ packages
filled by the machine are collected. The sample mean $\bar{y}$ is equal to
$517.2$ and the sample variance $s^2_{n-1}$ is equal to
$262.56$.

Test the hypothesis that the amount filled corresponds on average to the
setpoint. What is the absolute value of the $t$~test statistic?
\end{question}

\begin{solution}
The $t$~test statistic is calculated by:
\begin{eqnarray*}
  t & = & \frac{\bar y - \mu_0}{\sqrt{\frac{s^2_{n-1}}{n}}}
  = \frac{517.2 - 500}{\sqrt{\frac{262.56}{226}}}
  = 15.958.
\end{eqnarray*}
The absolute value of the $t$~test statistic is thus equal to
$15.958$.
\end{solution}

%% META-INFORMATION
%% \extype{num}
%% \exsolution{15.958}
%% \exname{t statistic}
%% \extol{0.01}
```

Table 2: LaTeX output of `Sweave("tstat.Rnw")`.

- `\exname{}`: a name or description. This is some short informative text, only used for printing of solutions within R.

- `\extol{}`: optional tolerance limit(s). For numeric solutions a tolerance limit can be specified; by default it is assumed to be 0. This can be useful for the automatic evaluation of numeric solutions if some deviation from the correct answer is allowed (e.g., due to rounding errors). If a single tolerance value is supplied, a symmetric interval around the true value is used. *(Version 1 of the **exams** package also supported asymmetric tolerance intervals when two tolerance values were provided. However, this has not been implemented for any of the version 2 interfaces.)* UPDATE

1. **Problem**
   A machine fills milk into 500ml packages. It is suspected that the machine is not working correctly and that the amount of milk filled differs from the setpoint $\mu_0 = 500$. A sample of 226 packages filled by the machine are collected. The sample mean $\bar{y}$ is equal to 517.2 and the sample variance s^2_{n-1} is equal to 262.56.

   Test the hypothesis that the amount filled corresponds on average to the setpoint. What is the absolute value of the t test statistic?

   Solution
   The t test statistic is calculated by:

$$t = \frac{\bar{y} - \mu_0}{\sqrt{\frac{s^2_{n-1}}{n}}} = \frac{517.2 - 500}{\sqrt{\frac{262.56}{226}}} = 15.958.$$

   The absolute value of the t test statistic is thus equal to 15.958.

Table 3: Display of processed exercise from `exams2pdf("tstat.pdf")`.

The meta-information is read from the LaTeX file (e.g., as in Table 2) and hence all values can be dynamically computed using `\Sexpr{}`. Typically, this will be necessary for the correct solution, but similarly the descriptive name, the tolerance allowed, or even the type of the question could be data-driven and determined by the DGP.

The user does not have to go from the Sweave file (Table 1) to the LaTeX file (Table 2) to the PDF file (Table 3) "by hand". The function `exams2pdf()` from package **exams** carries out all of these steps automatically. *(In version 1 of the package the function `exams()` was provided. While this is still contained in the package, it is recommended to use the newer and more flexible `exams2pdf()` for new projects.)*

UPDATE

```
R> library("exams")
R> set.seed(1090)
R> tstat_ex <- exams2pdf("tstat.Rnw")
```

After loading the package, `exams2pdf()` is called with the Sweave file name. The final PDF file is displayed on screen by default (for further options see Section 4). The PDF viewer used depends on the operating system: On Windows, the application specified in the system's file associations is used. On other systems (e.g., Linux or Mac OS), the PDF viewer specified in the options (see `getOption("pdfviewer")`) is started. `exams2pdf()` returns a list of exams which is a list of exercises (see Zeileis *et al.* 2014, for details). To extract an "`exams_metainfo`" object for pretty printing in R, the function `exams_metainfo()` can be used:

```
R> exams_metainfo(tstat_ex)

exam1
```

```
1. t statistic: 15.958 (15.948--15.968)
```

3. Combining exercises: The master LaTeX file

To produce the PDF for an exam, the exercises need to be weaved, tied together in the master LaTeX file and subsequently processed to PDF. More precisely, `exams2pdf()` takes the following approach:

1. Collect all Sweave files for the exercises, the master LaTeX file(s) and potentially additionally specified input files.
2. Copy all files to a (temporary, by default) directory.
3. Run `Sweave()` for each exercise.
4. Produce a copy of the master LaTeX file(s) in which certain control structures are substituted by dynamically generated LaTeX commands (e.g., for including the exercises).
5. Run `texi2dvi()` for each master LaTeX file.
6. Store the resulting PDF file(s) in an output directory or display it on the screen (for a single file only, by default).

Thus, the only thing needed to combine several independent exercises to an exam is a master LaTeX file. Its basic structure is rather straightforward but for more elaborate layouts it can become quite complex. Package **exams** provides various examples of master LaTeX files in the `inst/tex/` directory of the source package. The simplest one is `plain.tex` (see Table 4) which is described here to provide a first overview.

The structure of `plain.tex` is rather straightforward: First, the document class is defined and then necessary add-on packages are loaded.[3] Then the environments for `question` and `solution` are defined (as part of an itemize or enumerate list). Finally, the document starts and consists only of an enumerate list. In Step 4 of the algorithm described above, the control command `%% \exinput{exercises}` is replaced by (a sequence of) `\input{`*filename*`}`, e.g., `\input{exercise1.tex}`[4] for the first exercise etc. Optionally, the `%% \exinput{exercises}` comment can also be omitted in the template. Then, the user can place his/her own `\input{exercise1.tex}`, `\input{exercise2.tex}`, ... in the master LaTeX file. This requires knowing the correct number of exercises but adds the flexibility of including arbitrary text/formatting between the exercises.

Thus, all appearance options can be controlled in the LaTeX code of the master file, e.g., by changing the definitions of the `question`/`solution` environments or by modifying the code around the `\exinput{}` command. As an example, one might want to show only the questions and hide the solutions in the PDF. This is easily obtained by changing the definition of the `solution` environment to `\newenvironment{solution}{\comment}{\endcomment}`.

[3] Note that `Sweave.sty` is included for displaying R output. `texi2dvi()` ensures that this style file will be found by including R's `texmf` directory in the `TEXINPUTS` path.

[4] Leading zeros are added if `n` has more than one digit, e.g., `exercise01.tex` if there are between ten and 99 exercises.

```
\documentclass[a4paper]{article}

\usepackage{a4wide,color,Sweave,url,amsmath,booktabs,longtable}
\newenvironment{question}{\item \textbf{Problem}\newline}{}
\newenvironment{solution}{\textbf{Solution}\newline}{}
\newenvironment{answerlist}{\renewcommand{\labelenumi}{(\alph{enumi})}
  \begin{enumerate}}{\end{enumerate}}

\begin{document}
\begin{enumerate}
%% \exinput{exercises}
\end{enumerate}
\end{document}
```

Table 4: A simple master LaTeX file: `plain.tex`.

The structure described so far is completely static, except for the `\exinput{}` command. In fact, `exercises` is not the only argument allowed; at the moment two further arguments can be used to modify aspects of the master LaTeX file dynamically:

- `\exinput{exercises}`: Inclusion of exercises.
  Replaced by: `\input{`*filename*`}` (one for each exercise).
  Example: `\input{exercise1.tex}`.
- `\exinput{questionnaire}`: Inclusion of questionnaires, e.g., for cover sheets.
  Replaced by: `\exnum{...}`, `\exmchoice{...}`, or `\exstring{...}`, respectively (one for each exercise).
  Example: `\exnum{}{}{}{}{1}{5}{9}{5}{8}`.
- `\exinput{header}`: Further commands and definitions.
  Replaced by: `\`*command*`{`*value*`}` (one for each header command).
  Example: `\Date{2016-11-09}`.

For the latter two, the master LaTeX file has to define the commands that are used in the replacement step, e.g., `\exnum` has to be a command that takes nine arguments (corresponding to six and three digits before and after the decimal point for numeric solutions), `\exmchoice` has to take one required and one or more optional arguments such as `\exmchoice{X}[][X][][]` (corresponding to the logical multiple-choice answers) and `\exstring` has to take only one argument (corresponding to the solution string). *(The implementation of* `mchoice` *was slightly different in version 1 of the package where only a fixed number of five multiple choices was supported.)* For the header, arbitrary commands can be set up, see Section 4 for details.

UPDATE

Two master LaTeX files that illustrate all of the commands above are provided in **exams**: `exam.tex` and `solution.tex`. `exam.tex` can be used to generate exams including a cover sheet for students to fill in their names as well as their answers to the problems (hiding

solutions for the exercises, obviously). `solution.tex` produces PDF files containing a cover sheet similar to `exam.tex` but with the correct answers already filled in. Furthermore, the `solution` environments are displayed for each exercise.

4. Application and customization

In the simple case where only a single Sweave exercise is processed running `exams2pdf()` essentially corresponds to first calling `Sweave()` and then `texi2dvi()` on the file after including it in a master LaTeX file. This is quite convenient, especially for non-experts (in R and/or LaTeX), but beyond that not much simplification is gained by `exams2pdf()`. The main advantages of the function, however, are its flexibility and customizability: It controls the (dynamic) combination of the Sweave and LaTeX files and allows for

- construction of exams with stratified sampling of exercises,
- automatic generation of multiple copies (potentially of multiple layouts) with suitable names and storage,
- inclusion of a suitable cover page with answer fields,
- collection of meta-information for problems and solutions in an R object.

In the following, we illustrate how `exams2pdf()` can be employed to exploit all of these features. First, the interface of `exams2pdf()` is briefly described before using it to generate a small set of exams with corresponding solutions. Function `exams2pdf()` has the following arguments:

```
exams2pdf(file, n = 1L, nsamp = NULL, dir = ".", template = NULL,
          inputs = NULL, header = list(Date = Sys.Date()), name = NULL,
          control = NULL, encoding = "", quiet = TRUE, transform = NULL,
          edir = NULL, tdir = NULL, sdir = NULL, verbose = FALSE,
          points = NULL, ...)
```

where `file` specifies a list/vector of exercise Sweave files (see Section 2), `template` is the name of the master LaTeX file (see Section 3), and `n` is the number of random replications. The remaining arguments control details of the processing, e.g., the directories for input/output files. A technical manual is available on the help page `?exams2pdf`, a brief hands-on introduction to selected arguments is given below:

`file`: This is either a character vector containing the file names of Sweave exercises or a list of such vectors, e.g.,

```
R> myexam <- list("boxplots",
+                 c("confint", "ttest", "tstat"),
+                 c("anova", "regression"),
+                 "scatterplot",
+                 "relfreq")
```

Exams generated from `myexam` always have five exercises: `"boxplots"`, `"scatterplot"`, and `"relfreq"` are always included but the second exercise is randomly drawn from `"confint"`, `"ttest"`, `"tstat"`. Similarly, the third exercise is randomly chosen from `"anova"` and `"regression"`. This stratified sampling strategy is useful if there are several exercises related to the same topic, or several exercises for the same statistical techniques with different "stories". If only a single vector (rather than a list) of file names is specified, each exercise is always included in the final output file. The extension `.Rnw` can be omitted from the file names and the corresponding files should either be in the local directory, the `edir` directory or in the `exercises` directory of the installed package.

`n`: Number of randomly generated exams.

`nsamp`: Number of elements sampled from each list element of `file`. Sampling without replacement is used (if possible).

`dir`: Output directory for storing the resulting PDF files (and meta-information). If only a single PDF file is generated, this can be omitted (and the result is displayed directly on the screen), otherwise it has to be specified.

`template`: A (vector of) master LaTeX file(s). If more than one `template` is specified, one PDF output file is created for each in each of the `n` runs. The extension `.tex` can be omitted in the file name and the corresponding files should either be in the local directory (or provided with the full path) or in the `tex` directory of the installed package.

`header`: Additional LaTeX commands for replacement of `\exinput{header}` in the master LaTeX file. It has to be a `list()` of *command* `=` *value* pairs, where *value* can either be a static string or a function computing a string from the index i of the ith exam.

Further arguments: `inputs` can specify a list of files needed during the LaTeX compilation, e.g., private `.sty` files etc. `name` is the (vector of) prefix(es) for the final PDF files. `quiet = TRUE` suppresses output when calling `Sweave()` and `texi2dvi()`. `edir` is the path to the exercise directory (defaulting to the current working directory). `tdir` is the temporary (by default) directory into which all files are copied and where `Sweave()` and `texi2dvi()` are called. `sdir` is the directory (tempoarary by default) in which supplementary files (such as graphics) are stored. `verbose = TRUE` displays some progress information and `control` allows a few further control options (see also below).

For the subsequent example, we use a temporary output directory (but the reader could easily change `odir` to some local directory).

```
R> odir <- tempfile()
```

Assume that we want a different ID for each exam. To accomplish this, we define a function which is used to substitute a different `\ID{myexam`*ii*`}` command in the header of each exam. (The LaTeX command `\ID{}` has to be defined in the master LaTeX files.)

```
R> getID <- function(i)
+    paste("myexam", gsub(" ", "0", format(i, width = 2)), sep = "")
R> getID(1)
```

```
[1] "myexam01"
```

Using these arguments, a set of exams can be easily produced:

```
R> set.seed(1090)
R> ex <- exams2pdf(myexam, n = 5, nsamp = c(1, 2, 1, 1, 1), dir = odir,
+   template = c("exam", "solution"),
+   header = list(ID = getID, Date = Sys.Date()))
```

This takes the exam `myexam` (for which the corresponding Sweave files are all provided in the `exercises` directory of the package) and produces five exams (from the `exam.tex` LaTeX file) with associated solutions (from the `solution.tex` LaTeX file). The templates are both provided in the `tex` directory of the package and allow for specification of `\ID{}` and `\Date{}`. All output files are stored in `odir`

```
R> list.files(odir)
```

```
 [1] "exam1.pdf"     "exam2.pdf"     "exam3.pdf"     "exam4.pdf"
 [5] "exam5.pdf"     "solution1.pdf" "solution2.pdf" "solution3.pdf"
 [9] "solution4.pdf" "solution5.pdf"
```

and can now be easily inspected by the reader. If in addition to the PDF exams and solutions, further information about the exams is needed the `ex` list returned could be stored for future reference. This contains all LaTeX code for questions/solutions along with the metainformation about the correct answers etc. The latter can also be extracted and printed within R:

```
R> sol <- exams_metainfo(ex)
R> print(sol, 1)
```

```
exam1
    1. Parallel boxplots: 1, 2, 4
    2. t statistic: 4.52 (4.51--4.53)
    3. 2-sample t-test: 2, 3
    4. Prediction: 267.66 (267.65--267.67)
    5. Scatterplot: 1, 4
    6. Relative frequencies: 1, 4, 5
```

```
R> print(sol, "exam5")
```

```
exam5
    1. Parallel boxplots: 1, 2, 4
    2. t statistic: 48.203 (48.193--48.213)
    3. 2-sample t-test: 1, 5
    4. Prediction: 205 (204.99--205.01)
    5. Scatterplot: 1, 2
    6. Relative frequencies: 1, 5
```

For numeric results, the solution is displayed, possibly including tolerance limits (if non-zero). For multiple-choice answers, the true statements (coded with 1–5) are listed and the false ones are omitted. Consequently, if a multiple-choice question has no true statements, no numbers are displayed.

5. Discussion

Infrastructure vs. content

Package **exams** provides the technological framework for the generation of structured exams, especially for large-lecture courses. Given a relatively simple structure of exams (sequence of "stand-alone" exercises with multiple-choice/numeric/short text solutions), it is designed to be as flexible as possible. Thus, users should not have to worry about implementation details and can focus on the specification of the content and the development of the pool of exercises. While the package's structure can aid the design and development of the exercises, the package can, of course, not assure that "good" exercises (from an educational point of view) are generated. As this is beyond the scope of **exams**, some brief pointers to the relevant literature on statistical education and assessment are given here: Gal and Garfield (1997) and Garfield and Chance (2000) discuss issues such as topics covered and skills developed in statistics courses as well as suitable ways of assessment. The development of effective multiple-choice questions which force the student to understand underlying statistical concepts is crucial and often not straightforward. Strategies for good multiple-choice questions, especially if they are also used for self-study materials, are suggested by Klinke (2004).

Strategies for setting up exercises

As pointed out above, **exams** does not really address the problem of designing "good" exercises. However, the infrastructure supports several commonly-used strategies for setting up different types of exercises which in turn can support the decisions about the content. For example, in the case of multiple-choice two typical strategies are: Either only one option is correct or any one option can be correct or false. In the former case, the evaluation typically penalizes incorrect answers (to avoid random guessing) while in the latter case the number of potential patterns is sufficiently large that no penalization is necessary. In both cases, it is good practice to check the DGP employed for generating the answers and look at the resulting distribution of answer patterns. For exercises with numerical solutions, on the other hand, a typical problem in practice is how to set the tolerance. While the instructor will want to catch typical mistakes (e.g., wrong standardization, etc.), certain imprecisions (e.g., rounding errors in intermediate steps) might be tolerable. The simpler and somewhat more rigid approach to this problem is to practice with the students how to avoid the imprecisions and just require a certain accuracy of the solution (as in the t test example in Tables 1 and 2). A more flexible approach would be to derive the tolerance dynamically (as part of the DGP) assuring that common small imprecisions lead to results within tolerance limits while typical mistakes are not.

Experiences at WU Wien

During 2007, **exams** was employed by about 10–15 lecturers at the Department of Statistics

and Mathematics of WU Wien for jointly developing materials for the basic statistics lecture. With package **exams** we were able to address the technological requirements and facilitate the development process. All lecturers involved were familiar with R and LaTeX (which are both available on all standard platforms) and could thus contribute to the pool of exercises. To do so, they just needed to know the structure of the exercise Sweave files while different master LaTeX files (for the department's exams, exercise collections, etc.) have been written by the authors of the **exams** package. As an additional tool we decided to use **Subversion** (SVN, Pilato, Collins-Sussman, and Fitzpatrick 2004) for version control in order to provide all lecturers involved with access to all resources. In combination with the package this approach proved to be rather successful in addressing the needs of multi-author and cross-platform development. Since Spring 2008, **exams** is used at WU Wien for generating collections of exercises and trial exams (both available prior to the actual exam) as well as the exams and associated solutions (which are e-mailed to the students individually after correction of the test). Exams containing numeric answers are still corrected "by hand" but for exams consisting entirely of multiple/single-choice answers automatic scanning using optical character recognition (OCR) is used.[5]

6. Summary

Package **exams** provides a framework for automatic generation of simple (statistical) exams and associated self-study materials. It is based on independent exercises in Sweave format which can be compiled in exams (or other collections of exercises) by providing one (or more) master LaTeX template(s). Because contributing to the pool of exercises just requires knowledge of Sweave and minimal markup for meta-information, **exams** facilitates joint development of lecture materials. An extension of the package to e-learning exams, e.g., for **Moodle** or **OLAT**/**OpenOLAT** is described in Zeileis *et al.* (2014).

Acknowledgments

We are indebted to our colleagues at the Department of Statistics and Mathematics at WU Wien – in particular Regina Tüchler and Josef Leydold – for testing and challenging the code and making suggestions for improvement. We would like to thank two anonymous referees and one associate editor for their valuable comments which led to several improvements. This research was partially supported by the Austrian Science Foundation (FWF) under Hertha-Firnberg grant T351.

References

de Leeuw J (2001). "Reproducible Research: The Bottom Line." *Technical Report 2001031101*, Department of Statistics Papers, University of California, Los Angeles. URL http://repositories.cdlib.org/uclastat/papers/2001031101/.

[5] We also have an R implementation for scanning multiple/single-choice exams. However, this is somewhat geared towards the exam sheets used at WU Wien and Universität Innsbruck and hence not part of the **exams** package. Readers interested in this code should contact the package authors.

Gal I, Garfield JB (eds.) (1997). *The Assessment Challenge in Statistics Education*. IOS Press, Netherlands.

Garfield JB, Chance B (2000). "Assessment in Statistics Education: Issues and Challenges." *Mathematical Thinking and Learning*, **2**(1/2), 99–125.

Grün B, Zeileis A (2009). "Automatic Generation of Exams in R." *Journal of Statistical Software*, **29**(10), 1–14. URL http://www.jstatsoft.org/v29/i10/.

Klinke S (2004). "Q&A – Variable Multiple Choice Exercises with Commented Answers." In J Antoch (ed.), *COMPSTAT 2004 – Proceedings in Computational Statistics*, pp. 1323–1328. Physica Verlag, Heidelberg.

Knuth DE (1984). *The TEXbook*, volume A of *Computers and Typesetting*. Addison-Wesley, Reading, Massachusetts.

Knuth DE (1992). *Literate Programming*, volume 27 of *CSLI Lecture Notes*. Center for the Study of Language and Information, Stanford, California.

Lamport L (1994). *LATEX: A Document Preparation System*. 2nd edition. Addison-Wesley, Reading, Massachusetts.

Leisch F (2002). "Dynamic Generation of Statistical Reports Using Literate Data Analysis." In W Härdle, B Rönz (eds.), *COMPSTAT 2002 – Proceedings in Computational Statistics*, pp. 575–580. Physica Verlag, Heidelberg.

Leisch F (2012a). "Sweave FAQ." URL http://www.stat.uni-muenchen.de/~leisch/Sweave/.

Leisch F (2012b). Sweave *User Manual*. URL http://www.stat.uni-muenchen.de/~leisch/Sweave/.

Leisch F, Rossini AJ (2003). "Reproducible Statistical Research." *Chance*, **16**(2), 46–50.

Pilato CM, Collins-Sussman B, Fitzpatrick BW (2004). *Version Control with **Subversion***. O'Reilly. Full book available online at http://svnbook.red-bean.com/.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Rossini AJ (2001). "Literate Statistical Practice." In K Hornik, F Leisch (eds.), *Proceedings of the 2nd International Workshop on Distributed Statistical Computing, Vienna, Austria*. ISSN 1609-395X, URL http://www.ci.tuwien.ac.at/Conferences/DSC-2001/Proceedings/.

Zeileis A, Umlauf N, Leisch F (2014). "Flexible Generation of E-Learning Exams in R: Moodle Quizzes, OLAT Assessments, and Beyond." *Journal of Statistical Software*, **58**(1), 1–36. URL http://www.jstatsoft.org/v58/i01/.

Affiliation:

Bettina Grün
Institut für Angewandte Statistik
Johannes Kepler Universität Linz
Altenbergerstraße 69
4040 Linz, Austria
E-mail: Bettina.Gruen@jku.at
URL: http://ifas.jku.at/gruen/

Achim Zeileis
Department of Statistics
Faculty of Economics and Statistics
Universität Innsbruck
Universitätsstr. 15
6020 Innsbruck, Austria
E-mail: Achim.Zeileis@R-project.org
URL: http://eeecon.uibk.ac.at/~zeileis/