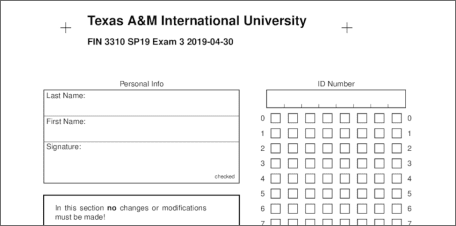
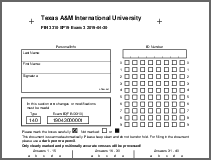
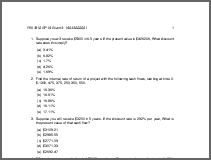
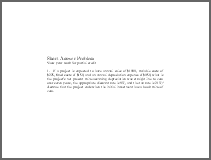
Experiences with using R/exams for written exams in finance classes with a moderate number of students at Texas A&M International University (TAMIU).



**Background**

While [R/exams](http://www.r-exams.org/) was originally written for large statistics classes, there is nothing specific to either large courses or statistics. In this article, I will describe how I use R/exams for my “Introduction to Finance (FIN 3310)” course. While occasionally I might have a class of 60 students, I generally have smaller sections of 20–25. These courses are taught and exams are given in-person (some material and homework assignments are delivered via the online learning management system, but I do not currently use R/exams to generate that).

Example pages from a typical exam are linked below:

[](http://www.r-exams.org/assets/posts/2019-04-23-tamiu_finance/fin3310exam31.pdf)  
[](http://www.r-exams.org/assets/posts/2019-04-23-tamiu_finance/fin3310exam31.pdf)  
[](http://www.r-exams.org/assets/posts/2019-04-23-tamiu_finance/fin3310exam31.pdf)

Because I have a moderate number of students, I grade all my exams manually. This blog post outlines how the exercises are set up and which R scripts I use to automate the generation of the PDF exams.

**Multiple-choice questions**

**Shuffling**

The actual questions in my exams are not different from many other [R/exams examples](http://www.r-exams.org/templates/), except that they are about finance instead of statistics. An example is included below with LaTeX markup and both the R/LaTeX (.Rnw) and R/Markdown (.Rmd) versions can be downloaded: [goalfinance.Rnw](http://www.r-exams.org/assets/posts/2019-04-23-tamiu_finance/goalfinance.Rnw), [goalfinance.Rmd](http://www.r-exams.org/assets/posts/2019-04-23-tamiu_finance/goalfinance.Rmd). I have 7 possible answers, but only 5 will be randomly chosen for a given exam (always including the 1 correct answer). To do so, many of my non-numerical questions take advantage of the exshuffle option here. (If you are one of my students, I *told* you this would be on the exam!)

Periodic updates to individual questions (stored as separate files) keep the exams fresh.

\begin{question}

The primary goal of financial management is to maximize the \dots

\begin{answerlist}

\item Present value of future operating cash flows

\item Net income, according to GAAP/IFRS

\item Market share

\item Value of the firm

\item Number of shares outstanding

\item Book value of shareholder equity

\item Firm revenue

\end{answerlist}

\end{question}

\exname{Goal of Financial Management}

\extype{schoice}

\exsolution{0001000}

\exshuffle{5}

**Numerical exercises**

Since this is for a finance class, there are numerical exercises as well. Every semester, I promise my students that this problem *will* be on the second midterm. The problem is simple: given some cash flows and a discount rate, calculate the net present value (NPV). Since the cash flows, the discount rate, and the answers are generated from random numbers, I would not gain much from defining more than five possible, the way I did in the qualitative question above. As you might expect, some of the incorrect answers are common mistakes students make when approaching NPV, so I have not lost anything relative to the carefully crafted questions and answers I used in the past to find out who really learned the material.

<>=

discountrate <- round(runif(1, min = 6.0, max = 15.0), 2)

r <- discountrate / 100.0

cf0 <- sample(10:20, 1) \* -100

ocf <- sample(seq(200, 500, 25), 5)

discounts <- sapply(1:5, function(i) (1 + r) \*\* i)

npv <- round(sum(ocf / discounts) + cf0, 2)

notvm <- round(sum(ocf) + cf0, 2)

wrongtvm <- round(sum(ocf / (1.0 + r)) + cf0, 2)

revtvm <- round(sum(ocf \* (1.0 + r)) + cf0, 2)

offnpv <- round(npv + sample(c(-200.0, 200.0), 1), 2)

@

\begin{question}

Assuming the discount rate is \Sexpr{discountrate}\%, find the

net present value of a project with the following cash flows, starting

at time 0: \$\Sexpr{cf0}, \Sexpr{ocf[1]}, \Sexpr{ocf[2]}, \Sexpr{ocf[3]},

\Sexpr{ocf[4]}, \Sexpr{ocf[5]}.

\begin{answerlist}

\item \$\Sexpr{wrongtvm}

\item \$\Sexpr{notvm}

\item \$\Sexpr{npv}

\item \$\Sexpr{revtvm}

\item \$\Sexpr{offnpv}

\end{answerlist}

\end{question}

\exname{Calculating NPV}

\extype{schoice}

\exsolution{00100}

\exshuffle{5}

**Leveraging R’s flexibility**

While there are other methods of generating randomized exams out there, few are as flexible as R/exams, in large part because we have full access to R. The next example also appears on my second midterm; it asks students to compute the internal rate of return for a set of cash flows. Numerically, this is a simple root-finding problem, but systems that do not support more advanced mathematical operations (such as Blackboard’s “Calculated Formula” questions) can make this difficult or impossible to implement directly.

<>=

cf0 <- sample(10:16, 1) \* -100

ocf <- sample(seq(225, 550, 25), 5)

npvfunc <- function(r) {

discounts <- sapply(1:5, function(i) (1 + r) \*\* i)

npv <- (sum(ocf / discounts) + cf0) \*\* 2

return(npv)

}

res <- optimize(npvfunc, interval = c(-1,1))

irr <- round(res$minimum, 4) \* 100.0

wrong1 <- irr + sample(c(1.0, -1.0), 1)

wrong2 <- irr + sample(c(0.25, -0.25), 1)

wrong3 <- irr + sample(c(0.5, -0.5), 1)

wrong4 <- irr + sample(c(0.75, -0.75), 1)

@

\begin{question}

Find the internal rate of return of a project with the following cash flows,

starting at time 0: \$\Sexpr{cf0}, \Sexpr{ocf[1]}, \Sexpr{ocf[2]},

\Sexpr{ocf[3]}, \Sexpr{ocf[4]}, \Sexpr{ocf[5]}.

\begin{answerlist}

\item \Sexpr{wrong1}\%

\item \Sexpr{wrong2}\%

\item \Sexpr{irr}\%

\item \Sexpr{wrong3}\%

\item \Sexpr{wrong4}\%

\end{answerlist}

\end{question}

\exname{Calculating IRR}

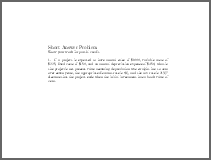
\extype{schoice}

\exsolution{00100}

\exshuffle{5}

**Short-answer questions**

While exams2nops() has some support for open-ended questions that are scored manually, it is very limited. I create and format those questions in the traditional manner (Word or LaTeX) and save the resulting file as a PDF. Alternatively, a custom exams2pdf() could be used for this. Finally, I add this PDF file on to the end of the exam using the pages option of exams2nops() (as illustrated in more detail below). As an example, the short-answer questions from the final exam above are given in the following PDF file:

[](http://www.r-exams.org/assets/posts/2019-04-23-tamiu_finance/exam3sa.pdf)

To facilitate automatic inclusion of these short-answer questions, a naming convention is adopted in the scripts below. The script looks for a file named examNsa.pdf, where N is the exam number (1 for the first midterm exam, 2 for the second, and 3 for the final exam) and sets pages to it. So for the final exam, it looks for the file exam3sa.pdf. If I want to update my short-answer questions, I just replace the old PDF with a new one.

**Midterm exams**

I keep the questions for each exam in their own directory, which makes the script below that I use to generate exams 1 and 2 straightforward. For the moment, the script uses all the questions in an exam’s directory, but I if want to guarantee that I have exactly (for example) 20 multiple choice questions, I could set nsamp = 20 instead of however many Rnw (or Rmd) files it finds.

Note that exlist is a list with one element (a vector of file names), so that R/exams will randomize the order of the questions as well. If I wanted to specify the order, I could make exlist a list of N elements, where each element was exactly one file/question. The nsamp option could then be left unset (the default is NULL).

When I generate a new set of exams, I only need to update the first four lines, and the script does the rest. Note that I use a modified version of the English en.dcf in order to adapt some terms to TAMIU’s terminology, e.g., “ID Number” instead of the exams2nops() default “Registration Number”. Since the TAMIU student ID numbers have 8 digits, I use the reglength = 8 argument, which sets the number of digits in the ID Number to 8.

library("exams")

## configuration: change these to make a new batch of exams

exid <- 2

exnum <- 19

exdate <- "2019-04-04"

exsemester <- "SP19"

excourse <- "FIN 3310"

## options derived from configuration above

exnam <- paste0("fin3310exam", exid)

extitle <- paste(excourse, exsemester, "Exam", exid, sep = " ")

saquestion <- paste0("SA\_questions/exam", exid, "sa.pdf")

## exercise directory (edir) and output directory (odir)

exedir <- paste0("fin3310\_exam", exid, "\_exercises")

exodir <- "nops"

if(!dir.exists(exodir)) dir.create(exodir) ## in case it was previously deleted

exodir <- paste0(exodir, "/exam", exid, "\_", format(Sys.Date(), "%Y-%m-%d"))

## exercise list: one element with a vector of all available file names

exlist <- list(dir(path = exedir, pattern = "\.Rnw$"))

## generate exam

set.seed(54321) # not the actual random seed

exams2nops(file = exlist,

n = exnum, nsamp = length(exlist[[1]]), dir = exodir, edir = exedir,

name = exnam, date = exdate, course = excourse, title = extitle,

institution = "Texas A\\&M International University",

language = "en\_tamiu.dcf", encoding = "UTF-8",

pages = saquestion, blank = 1, logo = NULL, reglength = 8,

samepage = TRUE)

**Final exam**

My final exam is comprehensive, so I would like to include questions from the previous exams. I do not want to keep separate copies of those questions just for the final, in case I update one version and forget to update the other, so I need a script that gathers up questions from the first two midterms and adds in some questions specific to the final.

The script below uses a feature that has long been available in R/exams but was undocumented up to version 2.3-2: You can set edir to a directory and all its subdirectories will be included in the search path. I have specified some required questions that I want to appear on every student’s exam; each student will also get a random draw of other questions from the first two midterms in addition to some questions that only appear on the final. How many of each is controlled by exnsamp, which is passed to the nsamp argument. They add up to 40 currently, so exams2nops()’s 45 question limit does not affect me.

library("exams")

## configuration: change these to make a new batch of exams

exnum <- 19

exdate <- "2019-04-30"

exsemester <- "SP19"

excourse <- "FIN 3310"

## options derived from configuration above

extitle <- paste(excourse, exsemester, "Exam", exid, sep = " ")

## exercise directory (edir) and output directory (odir)

exedir <- getwd()

exodir <- "nops"

if(!dir.exists(exodir)) dir.create(exodir)

exodir <- paste0(exodir, "/exam3", "\_", format(Sys.Date(), "%Y-%m-%d"))

## exercises: required and from previous midterms

exrequired <- c("goalfinance.Rnw", "pvcalc.Rnw", "cashcoverageortie.Rnw",

"ocfcalc.Rnw", "discountratecalc.Rnw", "npvcalc.Rnw", "npvcalc2.Rnw",

"stockpriceisabouttopaycalc.Rnw", "annuitycalc.Rnw", "irrcalc.Rnw")

exlist1 <- dir(path = "fin3310\_exam1\_exercises", pattern = "\.Rnw$")

exlist2 <- dir(path = "fin3310\_exam2\_exercises", pattern = "\.Rnw$")

exlist3 <- dir(path = "fin3310\_exam3\_exercises", pattern = "\.Rnw$")

## final list and corresponding number to be sampled

exlist <- list(

exrequired,

setdiff(exlist1, exrequired),

setdiff(exlist2, exrequired),

exlist3

)

exnsamp <- c(10, 5, 10, 15)

## generate exam

set.seed(54321) # not the actual random seed

exams2nops(file = exlist,

n = exnum, nsamp = exnsamp, dir = exodir, edir = exedir,

name = "fin3310exam3", date = exdate, course = excourse, title = extitle,

institution = "Texas A\\&M International University",

language = "en\_tamiu.dcf", encoding = "UTF-8",

pages = "SA\_questions/exam3sa.pdf", blank = 1, logo = NULL,

reglength = 8, samepage = TRUE)

**Grading**

R/exams can read scans of the answer sheet, automating grading for the multiple-choice questions. For a variety of reasons, I do not use any of those features. If I had to teach larger classes I would doubtless find a way to make it convenient, but for the foreseeable future I will continue to use the function below to produce the answer key for each exam, which I grade by hand. This is not especially onorous, since I have to grade the short-answer questions by hand anyway.

Given the serialized R data file (.rds) produced by exam2nops() the corresponding exams\_metainfo() can be extracted. This comes with a print() method that displays all correct answers for a given exam. Starting from R/exams version 2.3-3 (current R-Forge devel version) it is also possible to split the output into blocks of five for easy reading (matching the blocks on the answer sheets). As an example the first few correct answers of the third exam are extracted:

fin3310exam3 <- readRDS("fin3310exam3.rds")

print(exams\_metainfo(fin3310exam1), which = 3, block = 5)

## 19043000003

## 1. Discount Rate: 1

## 2. Calculating IRR: 4

## 3. Present Value: 3

## 4. Calculating stock prices: 5

## 5. Calculating NPV: 3

##

## 6. Annuity PV: 3

## 7. Goal of Financial Management: 1

## 8. Finding OCF: 5

## ...

It can be convenient to display the correct answers in a customized format, e.g., with letters instead of numbers and omitting the exercise title text. To do so, the code below sets up a function exam\_solutions(), applying it to the same exam as above.

exam\_solutions <- function(examdata, examnum) {

solutions <- LETTERS[sapply(examdata[[examnum]],

function(x) which(x$metainfo$solution))]

data.frame(Solution = solutions)

}

split(exam\_solutions(fin3310exam3, examnum = 3), rep(1:8, each = 5))

## $`1`

## Solution

## 1 A

## 2 D

## 3 C

## 4 E

## 5 C

##

## $`2`

## Solution

## 6 C

## 7 A

## 8 E

## ...

The only downside of the manual grading approach is that I do not have students’ responses in an electronic format for easy statistical analysis, but otherwise grading is very fast.

**Wrapping up**

Using the system above, R/exams works well even for small courses with in-person, paper exams. I do not need to worry about copies of my exams getting out and students memorizing it, or any of the many ways students have found to cheat over the years. By doing all the formatting work for me, R/exams helps me avoid a lot of the finicky aspects of adding, adjusting, or removing questions from an existing exam, and generally keeps exam construction focused on the important part: writing questions that gauge students’ progress.