RGIFT: An R Package for producing questionnaires in GIFT format

Virgilio Gómez-Rubio, María José Haro-Delicado and Franciso Parreño-Torres

November 17, 2023

Abstract

This paper introduces R package RGIFT to produce questionnaires in the GIFT format. This is a popular format used by several virtual learning environments. This package, combined with the R programming language, can be used to produce a number of questionnaires to test students' skills in an easy way.

1 Introduction

Zeileis and Grün [2009] describe the development of exams in an automated way in order to deal with a large number of students. In this work we describe a similar problem but provide a different solution based on the use of on-line questionnaires in a Course Management System.

This work has been triggered by the development of the Bolgone Process[1] and the need for a continuous assessment of the students during all their learning process. Lecturers are required to assess the students' developments through regular tests.

Most engineering degrees in the University of Castilla-La Mancha (UCLM) have a first-year course on Statistics. In order to conduct a continuous evaluation, which is at the heart of the Bologne Process, a number of tests are taken by the students in regular intervals. In particular, a test is taken at the end of each practical session in the computer lab.

The total number of first-year students attending these courses is between 300 and 400 in the Degrees of Computer Science and Industrial Engineering where we teach. UCLM provides all students and academic staff access to a Course Management System Moodle. Moodle is a complex system which allows lecturers to upload files with course notes, excercises and other interesting options to assess the students.

In particular, Moodle provides a suitable framework to develop tasks for the students. The most basic one is to provide a place to upload files so that lectures can grade them. Another interesting option includes on-line questionnaires. In this case, the student is presented with a number of different types of questions that he has to answer in a limited time. Moodle provides a grafical user interface to create these questions for this can be slow and, sometimes, cumbersome

[1] http://www.ond.vlaanderen.be/hogeronderwijs/bologna/

if a large number of questions need to be developed. Needless to say that once the student has finisehd and submitted his answers, he may get his mark immediately.

For this reason we have developed package `RGIFT`. With it a large number of differnt types of questions can be generated and used to create questionnaires for the students. In this paper we describe not only the different options in this package but also how to take advantage of the Moodle platform to produce questionnaires to assess students.

2 Creating questionnaires with GIFT

The GIFT format provides a simple but flexible way of producing questionnaires. Moodle can import and export questionnaires in GIFT format and there are several tools to convert from GIFT into other popular formats for on-line questionnaires. Questions must be written using UTF-8 text encoding.

Package RGIFT implements several functions to print questions in the GIFT format. The output can be redirected to a file using, for example, the `sink()` function. Similarly, Sweave [Leisch, 2002] can be used to embed the `R` code so that the ouput is directly included in the R output file.

The GIFT formats can deal with different types of questions whose formats are described in the following sections. Comments can be added to the questionnaire text file by using function `GIFTcomment()`. Also, questions can be grouped into categories by using function `GIFTcategory()`. Once this function is called, all question will be included in the same section until another category is set.

2.1 Types of questions

2.1.1 Multiple Choice Questions

Multiple Choice questions are those for which several possible answers are given and one must be chosen. Full mark is obtained only when the right answer is chosen and the student can get a penalty (i.e., negative mark) if a wrong answer is chosen rather than leaving the answer empty. Other variants include the option of having alternative "less right" answers which will not give the full mark. This can be controlled by assigning different weights to right and wrong answers.

The following `R` code shows how to produce a 3-question questionnaire using Multiple Choice Questions:

```
sink(file="MCQExample.txt", type="output")

GIFTcomment("Examples of Multiple Choice Questions")

GIFTcategory("MCQuestions")

#Question 1
GIFTMC("What's the mean of 1, 2, and 3?", c("1", "2", "3"), rightans=2,
   wwrong="-33.333")
```

```
#Question 2
GIFTMC("What's the command to compute the mean?",
   c("mean()", "sd()", "var()", "length()"), wwrong="-33.333")

#Question 3; Good answer plus a not so good answer
GIFTMC("How can we compute the mean of 1, 2, and 3?",
   c("mean(c(1,2,3))", "sd(c(1,2,3))", "(1+2+3)/3"), wright=c("100", "75"),
   rightans=c(1,3),  wwrong="-33.333")

#Question 4; Multiple valid answers
GIFTMC("How can we compute the mean of 1, 2, and 3?",
   c("mean(c(1,2,3))", "sd(c(1,2,3)", "(1+2+3)/3"), wright=c("50", "50"),
   rightans=c(1,3),  wwrong="-33.333")

sink()
```

```
> library(RGIFT)
> source("MCQExample.R")
```

Question 1 provides a very simple question about the mean of a set of numbers. In addition to the question, a vector of possible answers is given and the index of the right answer is available in `rightans`. Also, a negative weight is assigned to all the wrong answers (`wwrong="-33.333"`). Full marks will have a weight of 100.00 and secondary right answers can be given a lower positive weight. Question 2 is a similar answer where the right answer is the first one in the vector of possible answers (this is the default).

Question 3 is an example of a Multiple Choice question where there is a right answer which gives full mark, another not so good answer (which gives a 50%) and wrong answers (with a penalty). As one of the answers provides the full mark, Moodle only allows a single answer to be chosen.

Finally, Question 4 is an example of a trully Multiple Choice question where multiple answers must be chosen. Note that this is so because no single answer provides a full mark and that in order to get a 100% two answers must be chosen.

2.1.2 True-False Questions

True-False questions are those in which a statement is made and the student has to mark it either as true or false. The following code shows some examples using function `GIFTTF()`:

```
sink(file="TFQExample.txt", type="output")

GIFTcomment("Examples of True-False Questions")

GIFTcategory("TFQuestions")

#Question 1
GIFTTF("The mean of 1, 2, and 3 is 3?", TRUE)
```

```
#Question 2
GIFTTF("The command to compute the mean is sd()", FALSE)

#Question 3
GIFTTF("R is the best programming language ever!!", TRUE)

sink()
```

```
> source("TFQExample.R")
```

2.1.3 Short Answer Questions

Short Answer questions are those for which the students has to write down the answer. This can be produced using function `GIFTSA()` as follows:

```
sink(file="SAQExample.txt", type="output")

GIFTcomment("Examples of Short Answer Questions")

GIFTcategory("SAQuestions")

#Question 1
GIFTSA("The mean of 1, 2, and 3 is", c("Two", "2"),
  wright = c("100", "100"))

#Question 2
GIFTSA("Compute the mean of 1, 2 and 3",
   c("mean(c(1,2,3))", "sum(c(1,2,3))/3", "sum(c(1,2,3))/length(c(1,2,3))", "(1+2+3)/3"),
   wright=c("100", "75", "75", "50"))

sink()
```

```
> source("SAQExample.R")
```

Similarly to the Multiple Choice questions, different weights for right and worngs answers can be set. Multiple answers are possible as well.

2.1.4 Matching Questions

Matching Questions provide two groups of words and/or sentences that need to be matched. This can be produced as follows using function `GIFTM()`:

```
sink(file="MQExample.txt", type="output")

GIFTcomment("Examples of Matching Questions")

GIFTcategory("MQuestions")

#Question 1
```

```
GIFTM("Match the following operations to their respective R commands:",
   c("mean", "variance", "standard deviation"), c("mean()", "var()", "sd()"))

#Question 2
GIFTM("Match each vector with its mean:",
   c("c(1,2,3)", "c(4,5,6)", "c(7,8,9)"),
   as.character(c(mean(1:3), mean(4:6), mean(7:9))) )

sink()
```

```
> source("MQExample.R")
```

2.1.5 Missing Word Questions

In Missing Words questions the students need to fill in an empty space in a sentence with a word from a list. These can be defined using function `GIFTMW()`:

```
sink(file="MWQExample.txt", type="output")

GIFTcomment("Examples of Missing Word Questions")

GIFTcategory("MWQuestions")

#Question 1
GIFTMW("With command ", " the mean of a vector of values can be computed",
c("mean()", "sd()", "var()"), rightans=1)

#Question 2
GIFTMW("With command ", " the variance of a vector of values can be computed",
c("mean()", "sd()", "var()"), rightans=3)

#Question 3
GIFTMW("With command ", " the standard deviation  of a vector of values can be computed",
c("mean()", "sd()", "var()"), rightans=2)

sink()
```

```
> source("MWQExample.R")
```

2.1.6 Numerical Questions

With numerical questions students are asked to compute something and write the numeric answer down. Rather than matching the exact number it is better to allow for some error around the exact value so that valid answers are those found in an short interval. This questions are defined as following using function `GIFTNQ()`:

```
sink(file="NQExample.txt", type="output")
```

```
GIFTcomment("Examples of Numerical Questions")

GIFTcategory("NQuestions")

#Question 1
GIFTNQ("What's the mean of vector c(.4, .4, .5, .3)",
   as.character(mean(c(.4, .4, .5, .3))), .01)

#Question 2
GIFTNQ("What's the mean of vector c(.4, .4, .5, .3)",
   c(0.39, 0.41))

sink()
```

```
> source("NQExample.R")
```

In principle, the range for valid answers is the answer plus/minus the error (as in Question 1 above). If the answer is a vector of length two, then that is the range of valid answers (as in Question 2 above).

2.1.7 Essays

Essays are simply a question followed by a text box where the student is asked to write some text. This can be done, for example, to ask students to write some code to perform a particular task. The answers can be downloaded later and run through `R` to test the students' code automatically. This questions can be created with command `GIFTE()`:

```
sink(file="EExample.txt", type="output")

GIFTcomment("Examples of Essays")

GIFTcategory("EQuestions")

#Question 1
GIFTE("Write a few words about the R programming language.")

#Question 2

#Question 3

sink()
```

```
> source("EExample.R")
```

2.1.8 Description

Finally, Descriptions provide some text with no answer required. These descriptions are useful to include hints or tips in the questionnaires or to mark

the beginning of different blocks of questions. Descriptions can be added with command `GIFTD()`:

```
sink(file="DExample.txt", type="output")

GIFTcomment("Examples of Descriptions")

GIFTcategory("DQuestions")

#Description 1
GIFTD("R is a language for data analysis.")

sink()
```

```
> source("DExample.R")
```

2.2 Formatting text

In principle, questions and answers will be shown in plain text. However, other text formats can be used with Moodle. A full description of them is available at `http://docs.moodle.org/21/en/Formatting_text`. The text format is specified before the question text inside square brakets. The options are:

moodle is for Moodle Auto-Format text. This is used in text boxes in Moodle (for example, to send e-mails , add descriptions, etc.).

html is for HTML text. This will allow the use of most HTML tags to customise the text, add external links, etc.

plain is for plain text. The question will be shown as it is written and special tags (for example, in HTML) will be shown as they are written with no other effect.

markdown is for Markdown text. This is another simple format to enhance the text. A complete description of this format is available at

`http://docs.moodle.org/22/en/Markdown`

In addition, it is necessary to take care of a number of special characters (such as $, {, } and others) as they may not be displayed correctly. Function `RGIFTparse()` will look for these specials characters in a string and substitute them by an equivalent expression which will not interfere with the GIFT format. The following example will take a string with some of these special characters and return a valid one which can be passed to any of the functions to produce the questions:

```
> GIFTparse("Take care with $, {, } and ~.")

[1] "Take care with \\$, \\{, \\} and \\~."
```

The double backslash is need because text is printed using the `cat()` function but a single backslash will be printed.

Finally, as text must be encoded in UTF-8 function `iconv()` can be used to convert text strings from the local character encoding into UTF-8. A simple example in Spanish (that requires Spanish locales to be installed) could be:

```
#Example of use of text in UTF-8 encoding
library(RGIFT)

#Get Local enconding:
#Option 1: Set locale
#locale<-"es_ES.iso-8859-15"
#Option 2: Get locale from OS
locale <- Sys.getlocale("LC_CTYPE")

# Set locales according to OS
sysname <- Sys.info()['sysname']

if(sysname == "Darwin") { # Mac OS X
  from_locale <- "es_ES.ISO8859-1"
  to_locale <- "es_ES.UTF-8"
} else { # Other OS's (Linux)
  from_locale <- "es_ES.iso88591"
  to_locale <- "es_ES.utf8"
}

Sys.setlocale(locale = from_locale)
txt <- "\xbfEs Madrid la capital de Espa\xf1a?"

Sys.setlocale(locale = to_locale)

txt2 <- iconv(txt, "latin1", "UTF-8")

sink("utf8.txt")
GIFTTF(txt2, TRUE)
sink()

Sys.setlocale(locale = locale)


> source(file("UTF8.R", encoding="ISO-8859-1"))
```

Note that the previous text has been written in an environment with an ISO-8859-15 encoding. A good discussion on the use of different encodings in `R` can be found in the manual page of function `Encoding()`.

2.3 Including images

Questions sometimes require the use of images that display data or summary results. In principle, these images should be among the files uploaded to Moodle. Images can be uploaded to the course site in Moodle in a ZIP file and then

uncompressed there. Alternatively, the images can be created and stored in the same server but outside Moodle (for example, the lecturer's web site) so that they are linked from the question text using HTML tags.

A convenient approach is to define an image repository and create all images in JPEG or PNG format. Then the images can be uploaded to the image repository and the specfific link to the image included in the question text. A simple example would look as follows:

```
> #Image repository
> repos<-"http://www.uclm.es/profesorado/vgomez/imgtests/"
> #Create image
> imgfile<-"q1.png"
> png(imgfile, width=480, height=240)
> par(mfrow=c(1,4))
> #Sample several random variables and plot histogram
> hist(rbinom(100, 10, .25), main="(a)", xlab="x")
> hist(rnorm(100, 10*.25, sqrt(10*.25*.75)), main="(b)", xlab="x")
> hist(rexp(100, 1/(10*.25)), main="(c)", xlab="x")
> hist(runif(100, 8, 12), main="(d)", xlab="x")
> dev.off()
> #Create question in HTML format
> qtxt<-paste("[html]What\'s the histogram for a Binomial ",
+    "variable with n=10 and prob=0.25?\n<p>\n<img src=\'",
+    repos, imgfile, "\'>\n<p>", sep="")
> #Print question
> sink(file="imgqtion.txt", type="output")
> GIFTMC(qtxt, c("(a)", "(b)", "(c)", "(d)"), rightans=1, wwrong="-33.333")
> sink()
>
```

Note that different images must have different names but that this can be easily implemented in `R` .

2.4 Dealing with equations

In some cases equations need to be displayed in the question text or in the answers. Moodle supports many different ways of including equations (see `http://docs.moodle.org/22/en/Mathematics`). However, when producing questions, it is necessary to include the equations within the question text. Moodle allows the use of in-line LaTeX syntax for equations provided that these are included using LaTeX math-mode (i.e., using the two-dollar signs before and after the equation). For example:

```
> sink(file="eqqtion.txt")
> qtxt<-paste("[html]The expectation of a random variable which is \n",
+    "Binomial with $$n=10$$ and $$\\pi=0.7$$ is ", sep="")
> GIFTMC(qtxt, c("0.7", "1", "7", "0.07"), rightans=3)

[html]The expectation of a random variable which is
Binomial with $$n=10$$ and $$\pi=0.7$$ is
{
```

```
~%0%0.7
~%0%1
=7
~%0%0.07
}

> sink()
```

3 Design and use of questionnaires in Moodle

The previous examples can be used to create a set of different questions which will possibly belong to different categories. The use of categories is encouraged since it allows questions to be grouped into different topics. Moodle allows some customization on the group of questions.

For example, a questionnaire can be made of, say, 15 questions taken at random from different categories. The number of questions from each category can be set as well, so that all students will have the same number of questions (taken at random) from each category. This is helpful to design questionnaires that include questions about different sections in the syllabus.

Regarding the output given to the student, the right answers and final score can be shown once the questionnaire has been finished and the answers submitted. However, in certain cases it may be better not to show the right answers immediately and wait until the end of the exam (to make sure that all students have finished).

Although setting up a Moodle environment may seem complex (as it will require a web server and other software), a virtual appliance could be used instead. This will provide a pre-installed system will all the required software. We have tried TurnkeyLinux (`http://www.turnkeylinux.org/moodle`) to start a Moodle server in a few minutes. Among other features, this includes LDAP access so that external users can be authenticated and students can login using their university username and password. Hence, lecturers can set up a complete Moodle system in the local network.

4 Real Example

We have used RGIFT to produce a number of tests for the practicals of a first-year course in Statistics in the degrees in Computer Science and Industrial Engineering in the campus of Albacete (University of Castilla-La Mancha). These practicals are organised in seven two-hour sessions. Sessions one to six are used to train the students in the uses of `R` for descriptive and inferential statistics. In these sessions the last 30 minutes are used to answer a questionnaire about the contents of the practical. These questionnaires are made of 15 different questions which are randomly selected from the pool of question for that practical. Each group of questions is assigned to a category depending on the practical and, sometimes, the topic (for example, linear regression or ANOVA). The students where allowed to repeat the test up to three times and the highest score was kept. The students where given the right answers and mark after they completed each test.

The last session is an overall exam of the course practicals. Here, the students have two hours to complete the questionnaire. As this exam needs to cover all topics, the categories where used to design a questionnaire where two questions where taken (at random) from the questions in previous practicals. The students were only given the total mark when they submitted the exam and the actual list of right ans wrong answers was available after all the students had taken the exams a few days later.

In all cases, our aim was to provide each student with a similar (but different) list of questions so that all of them faced tests of similar complexity but in a way that prevented cheating. The code to produce these questions is available in files `test_exam-en.R` (in English) and `test_exam-es.R` (in Spanish) in directory `RGIFT/doc/exams` and can be easily view using, for example, `vi(file=system.file("doc/exams/test_exam-es.R", package="RGIFT"))`.

Also, we provided similar tests on the theoretical parts of the course so that they could take the tests at home and have immediate feedback on his knowledge of the topics in the course. Moodle keeps a record of the students' activities. In particular, for questionnaires the complete list of answer, socres, etc. is available so that the lecturers can provide a complete analysis of the responses. This is of interest in order to find what parts of the syllabus the students have not understood well or particularly difficult questions.

5 Conclusions

In this paper we have introduced package RGIFT which provides a convenient and easy way of producing different types of questions for on-line questionnaires. This can be fully imported into the Course Management System Moodle in order to develop on-line assessment, self study materials and on-line courses.

6 Acknowledgements

This work is part of project "MatemáTICas con software libre" funded by Consejería de Educación y Ciencia, Junta de Comunidades de Castilla-La Mancha for 2010-2012.

References

GIFT. GIFT format, 2011. URL `http://docs.moodle.org/22/en/GIFT`.

Friedrich Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. In Wolfgang Härdle and Bernd Rönz, editors, *Compstat 2002 — Proceedings in Computational Statistics*, pages 575–580. Physica Verlag, Heidelberg, 2002. URL `http://www.stat.uni-muenchen.de/ leisch/Sweave`. ISBN 3-7908-1517-9.

Moodle. Moodle, version 2.2, 2011. URL `http://moodle.org`.

Achim Zeileis and Bettina Grün. Automatic generation of exams in r. *Journal of Statistical Software*, 29(i10), 2009. URL `http://econpapers.repec.org/RePEc:jss:jstsof:29:i10`.

Carrera, E. (2002). Teaching statistics in secondary school. An overview: From the curriculum to reality. In B. Philips (Ed.), Proceedings of the Sixth International Conference on Teaching of Statistics. Cape Town: IASE. CD ROM.

Gal, I. (2002). Adult?s statistical literacy. Meanings, components, responsabilities. International Statistical Review, 70(1), 1-25