ePort Vignette:
Academic report generation for statistics instructors

Xiaoyue Cheng, Di Cook, Lindsay Rutter, Amy Froelich

`ePort` version 0.0.1 , 2016-01-17

Contents

[1]This LaTeX vignette document is created using the R function `Sweave` on the R package `ePort`. It is automatically downloaded with the package and can be accessed with the R command `vignette("ePort")`. The vignette is written by Lindsay Rutter.

Introduction

Summary

The `ePort` package provides tools for course instructors to generate electronic reports regarding student performance. Instructors can produce reports immediately after homework assignment deadlines, and use them to better understand student performance throughout the teaching semester. The goal is to allow instructors to assess and improve upon their teaching approaches in a fast response cycle.

The tools in this package will be especially beneficial for users who supervise large introductory courses. These courses often consist of multiple topics (groups of learning outcomes) that are taught by multiple instructors across multiple sections (groups of students). To accomodate the various ways that student performance can be examined for such courses, the package can generate various reports that can compare within and between topics and sections.

At its simplest, a report can be generated for one topic and one section. This would allow course coordinators to determine how well a particular section performed on a particular topic.

Reports can also be generated for one topic across multiple sections, with output format that allows course coordinators to quickly determine how well and consistently the multiple sections performed on a particular topic. This could be particularly insightful in cases where discrepancies in student performance are discovered between sections, especially if different instructors and/or teaching methods are being used across the sections.

In addition, reports can be generated for one unit (group of topics), either within one section or between multiple sections. This allows coordinators to assess student performance across all the learning outcomes of the combined topics that form the unit, and to quantify the consistency of how students perform across sections.

In general, both short and long versions of reports can be generated. Short versions of reports provide brief summarizations of student performance without regard to individual problems, whereas long versions of reports provide detailed summarizations of student performance for each individual problem in the assignment. Hence, long reports can also be used to confirm the suitability of assigned problems. For instance, in some courses, problems that assess the same learning outcome and are intended to be of equal difficulty levels are grouped into a problem set, and each student is assigned a random subset from this set of problems. However, sometimes there may be an unexpected discrepancy in student performance between problems in a given problem set, indicating an unintended discrepancy in the clearness or difficulty level of the problems to which students are randomly assigned. This package will allow users to efficiently find and fix such issues.

Installation

R is a open source software project for statistical computing, and can be freely downloaded from the Comprehensive R Archive Network (CRAN) website. The link to contributed documentation on the CRAN website offers practical resources for an introduction to R , in several languages. After downloading and installing R , the installation of additional packages is straightforward. To install the `ePort` package from R , use the command:

```
install.packages("ePort")
```

The ePort package should now be successfully installed. Next, to render it accessible to the current R session, simply type:

```
library(ePort)
```

Help files

To access help pages with example syntax and documentation for the available functions of the ePort package, please type:

```
help(package="ePort")
```

To access more detailed information about a specific function in the ePort package, use the following help command on that function, such as:

```
help(mergeSection)
```

The above command will return the help file for the function. Notice that this help file includes freestanding example syntax to illustrate how function commands are executed. This is the case in help files for most functions. The provided example code can be pasted directly into an R session.

Online homework database

Amy Froelich, one of the authors of the ePort package, developed an online homework database that has been applied for years in a large multi-section introductory statistics course. The resulting student data from this database has been applied to the ePort package, and has been useful in discovering common patterns and/or problems in student learning in this course. In this section of the vignette, we will describe the configuration of this particular database, keeping in mind that a course supervisor who is intersted in applying the ePort package to their course can do so by constructing their own online homework database in a similar format to the one described below.

* How will others obtain this particular database?????????

Database structure

The online homework database consists of 184 learning outcomes, which are grouped into 26 topics. In total, the database contains 2000 questions. A given student does not receive all 2000 questions over the course of the semester. Instead, similar questions are grouped together to form 330 question sets, and a given student will receive a certain number (usually one) from each of these 330 question sets.

Topics

The topics in the database cover a broad range of material that includes curriculum from Advanced Placement Statistics and popular introductory statistics textbooks. The topics are not rigidly structured around a specific textbook, and are not self-contained. Hence, course supervisors can tailor their course by selecting a subset of and/or reordering topics from the database. The full list of topic numbers and descriptions are provided below in Table 1.

Table 1: Topic numbers and descriptions

Number	**Description**
01	Data
02	Descriptive Statistics for a Single Categorical Variable
03	Descriptive Statistics for a Single Quantitative Variable
04	Descriptive Statistics for a Contingency Table
05	Descriptive Statistics for a Single Quantitative Variable between Groups
06	Normal Distribution
07	Descriptive Statistics for a Scatterplot
08	Descriptive Linear Regression
09	Samples and Surveys
10	Experiments
11	Randomness and Probability
12	Introduction to Probability and Events
13	Introduction to Random Variables
14	Binomial and Poisson Distributions
15	Sampling Distribution for the Sample Proportion
16	Confidence Intervals for the Population Proportion
17	Hypothesis Tests for the Population Proportion
18	Sampling Distribution for the Sample Mean
19	Confidence Intervals for the Population Mean
20	Hypothesis Tests for the Population Mean
21	Inference for the Difference in Two Population Proportions
22	Inference for the Difference in Two Population Means
23	Inference for the Mean Difference (Paired Samples)
24	Goodness of Fit Tests
25	Inference for Contingency Tables
26	Inference for Simple Linear Regression

Learning outcomes

Each topic contains learning outcomes, which are a list of statements that describe what a student is expected to understand after learning the topic. Learning outcomes form the main structure of the electronic assessment model of `ePort`. The average topic contains about seven learning outcomes. As an example, the learning outcomes for Topic 03 are provided in List 1 below.

List 1: Learning outcomes for Topic 03

A. Use standardizing to determine how many standard deviations an observation is away from the mean value.

B. Use z-scores to compare observations for different quantitative variables.

C. Explain how standardizing affects the shape, center, and variability of the distribution of a quantitative variable.

D. Determine which quantitative variables could be modeled using the normal distribution by interpreting graphical representations of the variable.

E. Apply the 68-95-99.7 Rule to any quantitative variable with a normal distribution.

F. Find percentile or area values for any given observation from a normal distribution.

G. Find the value of an observation when given a percentile or area value from the normal distribution.

Question types

The majority of the questions in the database include real data examples that cover diverse application areas, excepting business. The questions are time-tested in that they have continuously been edited and improved upon with the use of evaluation after administration to students. The majority of these questions also include feedback, which can be correct/incorrect feedback or answer-specific feedback. There are seven types of questions, which are enumerated in Table 2.

Table 2: Seven types of questions in the database

Abbreviation	**Type**	**Possible Points**
TF	True/False	1
MC	Multiple Choice	1
MU	Multiple Answer	1
MA	Matching	Number of Matches
FB	Fill in the Blank	Number of Blanks
JS	Jumbled Sentence	Number of Blanks
CA	Calculation	1

Below, an example problem and solution is provided for each of the seven question types in the database.

Example TF:

Does regular exercise lead to higher VO_2 max? VO_2 max is the maximum amount of oxygen in millimeters, one can use in one minute per kilogram of body mass. A random sample of 20

college age women was selected. Each student was asked whether or not they exercised regularly (at least 30 minutes of aerobic exercise 3 times a week). The VO_2 max for each student was also taken. This is an observational study.

a. True
b. False

Correct answer: a

Example MC:

The z-score for a particular observation is z = -3.1. This means the observation is:

a. 3.1 standard deviations above the mean
b. 3.1 standard deviations below the mean
c. 3.1 units above the mean
d. 3.1 units below the mean

Correct answer: b

Example MU:

Which of the following characteristics of pie is/are quantitative variables? Choose ALL that apply.

a. Calorie count
b. Number of cups of flour used
c. Type of pie (pecan, blueberry, etc)
d. Brand of sugar used

Correct answers: a and b

Example MA:

An ultramarathon is a foot race that is longer than 26.2 miles. Doctors have found that people who run an ultramarathon are at increased risk for developing respiratory infections after the race. Doctors beleive that taking vitamin C the 10 days before and the 10 days after the race would reduce the incidence of respiratory infections in the ultramarathon runners. To test their hypothesis, 20 runners were randomly assigned into two groups of 10 runners each. One group was given the same dose of vitamin C, in pill form, for 10 days before and 10 days are the race and the other group was given a sugar pill. Ten days after the race, the two groups were studied to determine how many of the runners in each group developed a respiratory infection. Match the ordered terms (1-4) to the correct description (a-d).

1. Experimental units
2. Response variable
3. Factor

4. Treatments

a. 20 ultramarathon runners
b. The use of vitamin C by ultramarathon runners
c. Whether or not the runner developed a respiratory infection
d. Vitamin C, Sugar pill

Correct answer: a, c, b, d

Example FB:

Fill in the blank with the correct number: Assume the length of female humpback whales can be modeled with a normal distribution with a mean of 13.7 meters and a standard deviation of 0.5 meters. According to the Empirical Rule or 68-95-99.7 Rule, _____ percent of female humpback whales will have a length between 13.2 meters and 14.2 meters.

Correct matches: 68, 68%

Example JS:

All other things being equal, a _____ confidence interval for a population proportion will be wider than a _____ confidence interval for the same population proportion.

Correct answer: 95%, 90%

Example CA:

The regression line for predicting the value of a variable y from the value of a variable x is as follows:

$y = 1.25 + 0.5x$

Use this equation to predict the value of y when the value of x is 188. Round your final answer to 2 decimal places.

Correct answer: 95.25

Question sets

A question set consists of a set of questions that all have the same format, and all cover the same component of a learning outcome. Question sets allow for different students to be presented with different subsets of similar questions. The number of questions from a given question set that are to be presented to each student is specified, and the questions are selected at random. There is at least one question set per learning outcome, and there is no connection between question sets.

Report generation outline

A diagram of the report generation process can be seen in Figure 1. Blue boxes represent external software (`Respondus` and `Blackboard`) that must be used along with `ePort`. Green boxes represent the three file types that must be provided as input in order for `ePort` to generate reports. Red boxes represent functions within `ePort` that are then used to produce the reports.

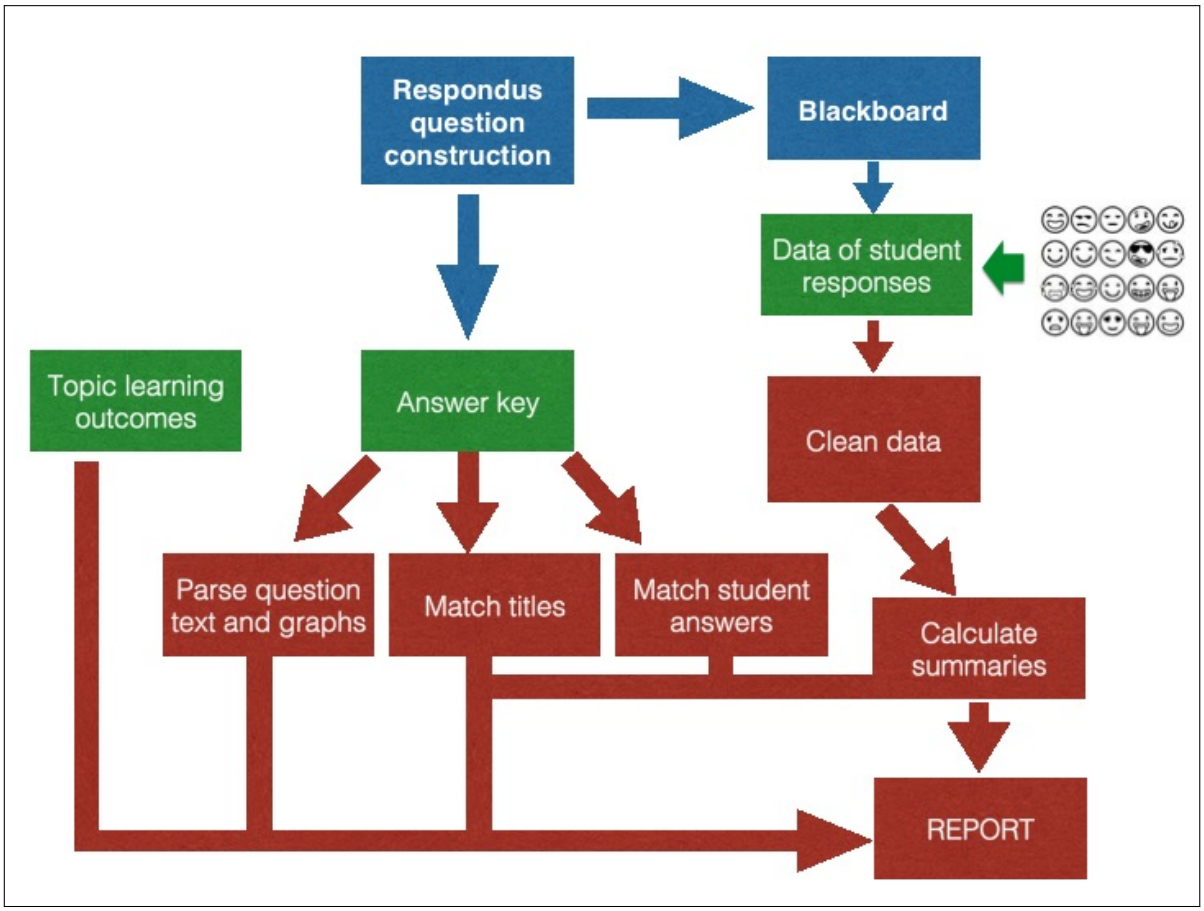


Figure 1: Overview of report generation

External software - Respondus

External software that must be used for `ePort` to generate the reports are represented in the blue boxes of Figure 1. As this figure shows, the database questions are constructed in the software `Respondus`. More information about `Respondus` can be found on their website.

It is important to code question titles in `Respondus` in a format that will allow all database questions to be identified and searched later in the `ePort` assessment model by topic, learning outcome, question type, and question set. An example of a coded question title is shown below:

Example question title: **T16.A.A.04-1.1.MC.1**

T16: Topic 16
A: Learning Outcome A for Topic 16
A: Question Set A for Topic 16
04-1: Question Set A for Topic 16 has 4 questions, and 1 is randomly assigned to each student
1: All questions in Question Set A for Topic 16 are worth 1 point
MC: All questions in Question Set A for Topic 16 are in multiple choice format
1: Question label of this particular question from Question Set A for Topic 16

External software - Blackboard

As seen in Figure 1, once the database has been constructed in `Respondus`, we can upload it to `Blackboard`. More information about `Blackboard` can be found on their website.

Typically, assignments on `Blackboard` can be made available to students on the first day the topic is introduced in lecture, and closed a set number of days after the topic is concluded in lecture. Each student can access their assignment over the course of as many sittings as needed, saving their work each time, but the can only submit their assignment once before the deadline. Once the deadline of the assignment has passed, students have access to answers and feedback for their set of questions.

Input files

The three necessary input files for `ePort` to generate reports are represented in green in Figure 1. A file that enumerates the learning outcomes is required; an example of this type of file has already been displayed in List 1. An answer key file is also required, and can be generated from the database in `Respondus`. The third required file consists of the answers that students submit when they complete their assignments on Blackboard.

Parsing input files

The functions needed to transform input files to final report files are represented in red in Figure 1. There are simple cleaning procedures that must be applied to the student response data files from Blackboard. One task that must then get done here is matching student answers from the cleaned student response data file to possible answer choices in the answer key from Respondus. This step simultaneously requires that the question set a particular student received at random is successfully matched to the corresponding question titles in the answer key from Respondus.

The final reports consist of data summaries, graphics, and analyses regarding how students performed on the assignments. As will later be explained in more detail, some report types have more verbose versions, which include individual summaries for each question of the topic. This means that it is not only the text, but also any graphs and equations from the questions that must be parsed in the answer key file. Doing so allows for the verbose report types to republish all components of a given question as was presented to students, such as relevant graphics, equations, and text, alongside the analysis of student performance for that question.

Example data

A directory that contains example data is automatically installed with the ePort package. The name of this directory is `extdata`. Understanding the location, layout, and content of the `extdata` directory will be necessary to continue with the examples provided in the vignette.

The absolute pathway to the `extdata` directory on your local computer can be determined by typing the following command into the R console:

```
system.file("inst/extdata/", package="ePort")
```

* Add in picture with layout of the example data file * Describe necessary format of each type of example data file * Warn that real data should not be added to this example data file. To view this file, simply open it in a Web Brower (Mozilla Firefox, Google Chrome, Microsoft Internet Explorer, Apple Safari).

Generating Reports

Currently, the `ePort` package offers six report types, depending on what the user is trying to compare and analyze about student performance. The same function `makeReport()` is used to generate each of these six report types; however, the input parameter `reportType` to the function will be different depending on which of the six report types the user wishes to run. Since it is most efficient for the user to hard-code in the `reportType` parameter, below is a reference for the parameter options (in quotes) for each of the six report types:

- One topic for one section - short version ("secTopicShort")
- One topic for one section - long version ("secTopicLong")
- One topic comparing multiple sections - short version ("crossSecTopicShort")
- One topic comparing multiple sections - long version ("crossSecTopicLong")
- One unit (group of topics) for one section ("secUnit")
- One unit (group of topics) comparing multiple sections ("crossSecUnit")

This information can also be obtained by running `help(makeReport)`, and will be demonstrated in the next two sections immediately below.

One topic for one section - short version

An instructor may be motivated to generate a short report for one topic and one section if they are seeking answers to the following types of questions:

1. Overall, how did this section of students do on this homework assignment?
2. Which students from this section scored poorly overall on this homework assignment?
3. Are there any learning outcomes for this topic that this section of students found easy or difficult?
4. Are there any question sets for this topic that this section of students found easy or difficult?

Code

We start by demonstrating how to generate the electronic report for one section one topic. This demonstration will use the example input files provided in the previously-described `extdata` directory, and will output the report to the `OutputFiles` subdirectory of the `extdata` directory. If you have not modified anything in the `extdata` directory, then the `OutputFiles` subdirectory should be empty, as we have not generated any example reports yet.

In this demonstration, we will create a report for `Topic 06` and `Section AB`. Like any individual report, we will require three input files: an answer key file, a data file, and a learning outcome file. There should be two example answer key files in the subdirectory `KeyFiles` (`Topic06.Questions.htm` and `Topic03.Questions.htm`), and we will use the `Topic06.Questions.htm` file. ????????????? Additionally, there should be four example data files in the subdirectory `DataFiles` (`Topic03.AB.csv`, `Topic03.CD.csv`, `Topic06.AB.csv`, and `Topic06.CD.csv`), ?????????????????????????????????????? and we will use the `Topic06.AB.csv` file. Lastly, there should be two example learning outcome files in the subdirectory `LOFiles` (`Topic03.Outcomes.txt` and `Topic06.Outcomes.txt`), and we will use the `Topic06.Outcomes.txt` file.

The block of code we will use to generate our `Topic 06 Section AB` is shown in the code block below. If this is your first time reading through the vignette, it is recommended that you do **not** run this code block all at once just yet. Instead, this code block is designed to provide you with an overview of the procedure, and is something you can refer back to once you have completed the vignette at least once, should you want the code in one condensed location:

```
key_htm = system.file("inst/extdata/KeyFiles/Topic06.Questions.htm", package="ePort")

refineKey(key_htm)

keyPath = gsub("htm$","txt",key_htm)

dataPath = system.file("inst/extdata/DataFiles/Topic06/Topic06.AB.csv", package="ePort")

rewriteData(dataPath)

loPath = system.file("inst/extdata/LOFiles/Topic06.Outcomes.txt", package="ePort")

outPath = system.file("inst/extdata/OutputFiles", package="ePort")

makeReport(keyFile=keyPath, dataFile=dataPath, loFile=loPath, outFile=outPath)
```

Now that we have seen the entire code required for this procedure, we briefly explain each step of the above code block. Here, we recommend that you follow along by actively running each piece of code, as will be demonstrated below.
The first line from the code block is where we save the absolute pathway of the answer key. Here, we save it to a string variable called `key_htm`.

```
key_htm = system.file("inst/extdata/KeyFiles/Topic06.Questions.htm", package="ePort")
```

Second, we must parse and clean this .htm answer key file, and convert it to plain text format. We do this by calling the `refineKey()` function of `ePort` on the .htm answer key file. By running the line below, we will create the cleaned .txt file:

```
refineKey(key_htm)
```

By default, the `refineKey()` function will place the cleaned .txt file into the same directory as the original .html file, and with the same name. Hence, if the you navigate to the `extdata` directory and its `KeyFiles` subdirectory, you should now see the new and cleaned .txt file we just created, `Topic06.Questions.txt`.

Our third step is to save the absolute pathway of this new and cleaned .txt answer key. Below, we save it to a string variable called `keypath`.

```
keyPath = gsub("htm$","txt",key_htm)
```

Now that we have the absolute path of our cleaned .txt answer key, our fourth step is to define the absolute pathway of our data file. We save this to a variable called `dataPath`.

```
dataPath = system.file("inst/extdata/DataFiles/Topic06/Topic06.AB.csv", package="ePort")
```

After this, our fifth step is to prime the data file to be compatible with steps later down the pipeline of generating the reports. We do this by running the `rewriteData()` function of ePort on the data file. This function changes certain non-meaningful character issues that would otherwise cause a problem when running the reports. For more details about the specific process, please run a help command on the function. Below, we rewrite the data file:

```
rewriteData(dataPath)
```

Upon running the above code snippet, you do not obtain a new file. Instead, the original file is overwritten. For this reason, you might inadvertantly run the function `rewriteData()` on a certain data file more than once, not remembering whether or not you have indeed converted it. This should not be a problem to run the function any number of times; you would simply receive a message that reads something like: "Note: Topic06.AB.csv was already successfully converted to usable format."

Our sixth step is to save the absolute path of our learning outcome file. Below, we save this to a variable called `loPath`.

```
loPath = system.file("inst/extdata/LOFiles/Topic06.Outcomes.txt", package="ePort")
```

After that, the seventh step is to specify our desired output directory. This is the absolute path of where the reports should be saved. Below, we create a variable called `outPath` to specify that we want to output our report to the `OutputFiles` subdirectory.

```
outPath = system.file("inst/extdata/OutputFiles", package="ePort")
```

Now that we have primed our three input files (cleaned answer key, data file, and learning outcomes file) and specified our output directory, our last step is to generate the report using the `makeReport()` function.

```
makeReport(keyFile=keyPath, dataFile=dataPath, loFile=loPath, outFile=outPath)
```

Upon running this code, you will receive the following message and menu asking for your input:

Please enter integer (1-6) corresponding to desired report type below.

Note: If running many reports, it is more efficient to exit now and hard-code the reportType parameter. See help(makeReport).

1: One topic for one section - short version ("secTopicShort")
2: One topic for one section - long version ("secTopicLong")
3: One topic comparing multiple sections - short version ("crossSecTopicShort")
4: One topic comparing multiple sections - long version ("crossSecTopicLong")
5: One unit (group of topics) for one section ("secUnit")
6: One unit (group of topics) comparing multiple sections ("crossSecUnit")

Since we wish to generate the type of report that is the short version of one topic for one section, then we can type the number `1` into the menu console. If we do so, then the report will be generated. However, upon seeing the menu option, we can also escape out of the command, and rerun the last line of code, only this time hard-code specifying the `reportType` parameter as `secTopicShort`. This is shown below:

```
makeReport(keyFile=keyPath, dataFile=dataPath, loFile=loPath, outFile=outPath,
reportType = "secTopicShort")
```

The advantage to specifying by hard-coding the `reportType` parameter is especially pertinent for users who wish to run these reports serially for each of many sections. This will be made more clear in a later section of the vignette (Running sequentially on batch of sections).

Output

Whether we specify the type of report we wish to generate by hard-coding or selecting from the menu, we should have successfully generated the report. We can find our report in the `OutputFiles` subdirectory of the `extdata` example directory. Indeed, we see that we have our short report (`Stat101_hwkTopic06_ABshort.pdf`) for `Topic 06` and `Section AB`. At this point, it may be helpful for you to open the short report you just generated.

This short report contains tabular and plotting overviews of student performance overall, by learning outcome, and by question set. We present a few examples of the types of output that can be found in this report. However, be sure to also view all of the output in the short report you just generated. A tabular summary of student scores is provided in the report, as seen in Figure 2. The report also contains a similar overview of student performance in the form of a histogram, which is not included in this vignette.

Mean	Std.dev	Min	Q1	Median	Q3	Max
6.24	2.38	0.00	5.00	6.00	8.00	12.00
(25%)	(9.5%)	(0%)	(20%)	(24%)	(32%)	(48%)

Figure 2: Summary statistics of student scores. The mean score on the assignment was low (only 25%), and even the top-scoring student did not perform so well (48%).

In addition to understanding how students from this section performed overall for this topic, instructors and course coordinators may wish to determine how students performed across the different learning outcomes. Indeed, the report provides a fluctuation diagram for this purporse. Figure 3 plots square objects, each of which contains an area that corresponds to the count of students who obtained that score for that learning outcome. Learning outcomes are ordered on the horizontal axis by mean score, with the highest mean score positioned on the left side of the axis. The mean score for each learning outcome is represented with a blue horiziontal line.

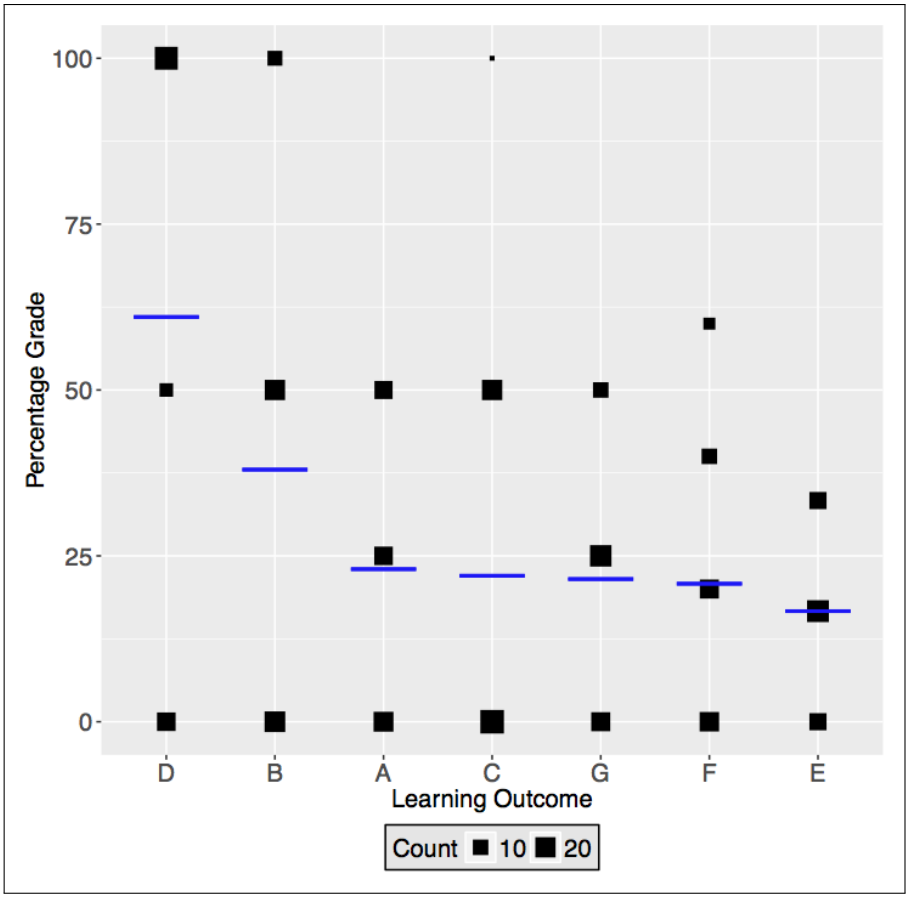


Figure 3: Fluctuation diagram of percentage correct by learning outcome. For this section, learning outcome D had the highest score, and learning outcome E had the lowest score. However, for each learning outcome, there remained a large proportion of the total of 50 students who scored poorly, and even did not earn any points at all (0%).

Additionally, instructors and course coordinators may wish to obtain an overview of how students from this section performed across the different learning sets within the learning outcomes. The report does generate such a tool, which is shown here in Figure 4.

LO	Qset	#	Mean	Std.dev	Min	Median	Max
D	I	2	64	48.49	0.00	100.00	100.00
D	J	4	58	49.86	0.00	100.00	100.00
A	B	4	48	50.47	0.00	0.00	100.00
A	A	4	44	50.14	0.00	0.00	100.00
B	E	1	40	49.49	0.00	0.00	100.00
B	F	1	36	48.49	0.00	0.00	100.00
F	M	5	36	48.49	0.00	0.00	100.00
F	Q	5	30	46.29	0.00	0.00	100.00
G	S	5	26	44.31	0.00	0.00	100.00
G	U	5	26	44.31	0.00	0.00	100.00
C	H	1	24	43.14	0.00	0.00	100.00
E	L	9	23.33	25.42	0.00	33.33	66.67
C	G	1	20	40.41	0.00	0.00	100.00
F	N	5	18	38.81	0.00	0.00	100.00
G	R	5	18	38.81	0.00	0.00	100.00
G	T	5	16	37.03	0.00	0.00	100.00
F	O	5	14	35.05	0.00	0.00	100.00
E	K	9	10	15.43	0.00	0.00	33.33
F	P	5	6	23.99	0.00	0.00	100.00
A	C	4	0	0.00	0.00	0.00	0.00
A	D	4	0	0.00	0.00	0.00	0.00

Figure 4: Summary statistics of the question sets. Rows are sorted by mean scores, which are marked red if less than 80 percent. We see that, in this section, students scored a mean of less than 80 percent in all question sets; however, some questions sets were more challenging than others.

An instructor may wish to know which students from this section are performing poorly overall on this assignment. To assist this, the report also includes a list of student names and their corresponding scores, for all cases of students who received a score below 80% on the assignment, as is demonstrated in Figure 5.

	% Correct		% Correct		% Correct		% Correct
w.introstat40	48.00	w.introStat59	32.00	w.introstat43	24.00	w.introstat23	16.00
w.introStat50	48.00	w.introstat69	32.00	w.introStat51	24.00	w.introstat29	16.00
w.introstat26	36.00	w.introstat31	28.00	w.introStat53	24.00	w.introstat36	16.00
w.introstat39	36.00	w.introstat35	28.00	w.introstat66	24.00	w.introstat46	16.00
w.introstat42	36.00	w.introstat44	28.00	w.introstat67	24.00	w.introStat56	16.00
w.introStat52	36.00	w.introstat48	28.00	w.introstat70	24.00	w.introstat68	16.00
w.introstat65	36.00	w.introStat54	28.00	w.introstat21	20.00	w.introstat24	12.00
w.introstat33	32.00	w.introStat58	28.00	w.introstat28	20.00	w.introstat27	12.00
w.introstat34	32.00	w.introstat61	28.00	w.introstat38	20.00	w.introstat30	12.00
w.introstat37	32.00	w.introstat62	28.00	w.introstat47	20.00	w.introstat64	4.00
w.introstat45	32.00	w.introstat63	28.00	w.introStat57	20.00	w.introstat25	0.00
w.introstat49	32.00	w.introstat32	24.00	w.introstat60	20.00		
w.introStat55	32.00	w.introstat41	24.00	w.introstat22	16.00		

Figure 5: The 50 students whose percentages are less than 80%.

Along the same lines, the report automatically generates a .txt file that lists the e-mails of all students who scored below 80% on this assignment. The list is repeated, once in a comma-separated format, and once in a semicolon-separated format. These list formats are intended to provide instructors with a streamlined method to contact students, if needed, to inform them of their low-scoring performance and possibly to recommend seeking additional support and resources to improve their performance in future assignments. You can open the example .txt file we created in this vignette; it should be located in the `OutputFiles` subdirectory of the `extdata` example directory as (`Topic06_AB_students_below80.txt`).

One topic for one section - long version

An instructor may be motivated to generate a long report for one topic and one section if they are seeking answers to the following types of questions:

1. Based on how students from this section performed on this topic, are there any noticeable problems with questions in certain question sets not being equally difficult?
2. Which students from this section scored poorly on each learning outcome for this homework assignment?
3. For each question in the assignment (particularly the difficult ones), what was the distribution of student responses?

Code

The short report we created for one section one topic (`Stat101hwk_Topic06_AB-short.pdf`) may be helpful for users who want to view a brief and overall summarization of student performance. However, we can also generate a more verbose report summarization for one section one topic that would include additional details, such as separate analysis for each problem in the assignment. We have already defined our three input file pathways and one output file pathway. Hence, if we wish to generate this longer version of the report summarization for one section one topic, all we need to rerun is the `makeReport()` function, only this time specifying the `reportType` parameter with the option of "secTopicLong".

```
makeReport(keyFile = keyPath, dataFile = dataPath, loFile = loPath, outFile = outPath, reportType = "
```

We can find our output report in the `OutputFiles` subdirectory of the `extdata` example directory. Indeed, we see that we now have a much longer report, called `Stat101hwk_Topic06_ABlong.pdf`.

Output

The short report featured tools that identified students from the section who performed poorly overall. The long report provides a more detailed output that informs instructors on how each student did not only overall, but also for each learning outcome. This is shown below in Figure 6.

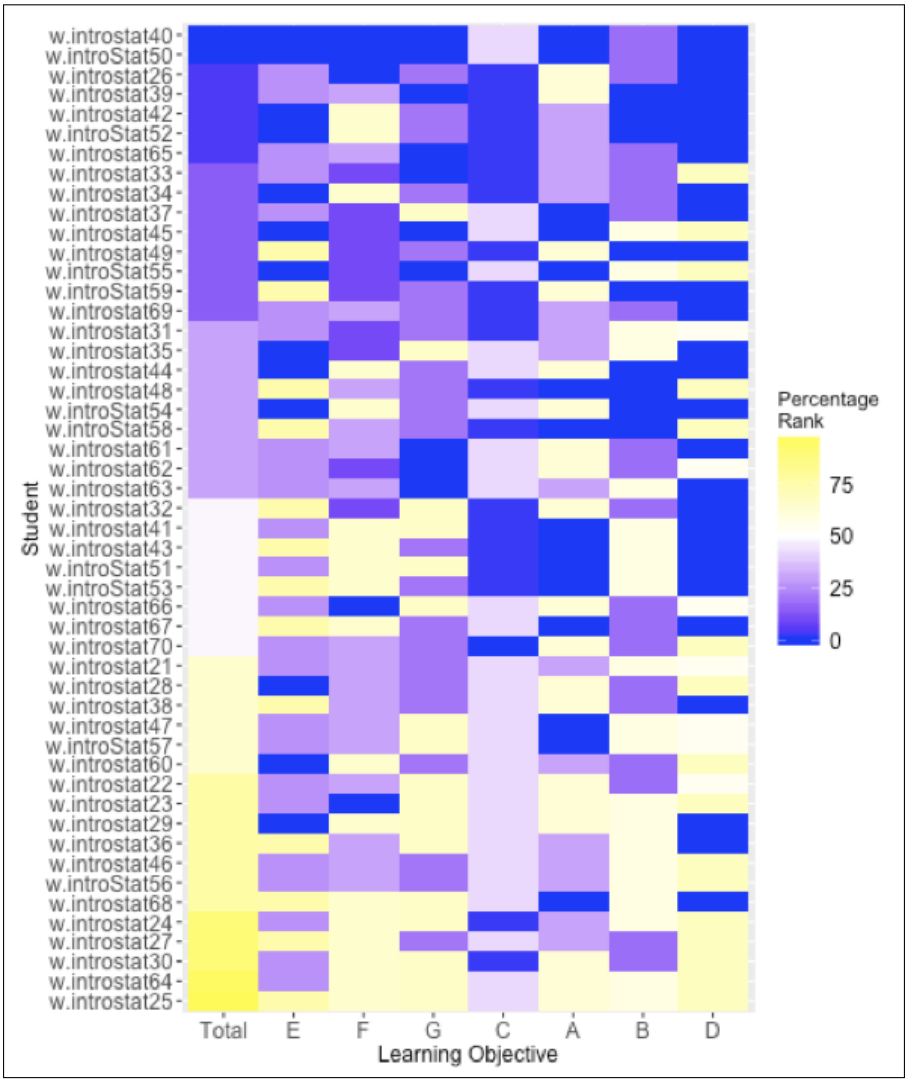


Figure 6: Heat map of the student ranks. Blue represents the top ranks, while yellow represents the bottom ranks. We see that the student (w.introstat25) who performed the best in the section overall, also performed above average compared to the class for most learning outcomes, except for learning outcome C. Moreover, some students (for instance, w.introstat68) performed inconsistently across learning outcomes.

Oftentimes, one of the aims in an online homework assignment is to ensure that any two students in the course are not likely to receive the exact same set of problems on any given assignment, thereby encouraging students to complete their own work. However, it is important to ensure that all possible combinations of problem subsets that a student could receive are equally difficult. Hence, it is necessary to confirm that all problems that can be randomly selected from each given learning set are equally difficult. The long report does indicate this information to users, as is seen below in Figure 7. Please note that this image is abridged for demonstration purposes; if you were to view the same image in the long report you just generated, then you would see that indeed it is a comprehensive table that consecutely lists the pertinent information for all 89 questions in this assignment.

ID	LO	Qset	Name	Type	FullPt	QinSet	N	CrtPct	Count	NA's	Mean	Std	Flag
1	A	A	1	MC	1	1	4	40.00	10	40	0.40	0.52	
2	A	A	2	MC	1	1	4	46.15	13	37	0.46	0.52	*
3	A	A	3	MC	1	1	4	30.00	10	40	0.30	0.48	*
4	A	A	4	MC	1	1	4	52.94	17	33	0.53	0.51	*
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
21	D	I	feet	TF	1	1	2	65.00	20	30	0.65	0.49	
22	D	I	lowtemp	TF	1	1	2	63.33	30	20	0.63	0.49	

Figure 7: Comparison of student performance in this section on each question of Topic 06, grouped by learning set. If any pair of questions in a given learning set have a difference in mean scores of more than 15%, then both questions are flagged. Question set A contained four problems, with mean scores that ranged from 30.00% to 52.94%. Three of these problems (IDs: 2, 3, and 4) were flagged as having a mean score that was at least 15% different than the mean score of another problem in the same question set. In contrast, question set I contained two problems, with mean scores that ranged from 63.33% to 65.00%. Hence, neither of these two problems were flagged as demonstrating a concerningly large difference of at least 15% in difficulty level from within the same question set.

As per the figure above, an instructor can quickly determine if there are problems for which students tend to show poor performance, and if there question sets for which students tend to show disparate performance between the individual questions. In either situation, an instructor may wish to examine these problems more closely, and determine how students are answering these problems incorrectly. For this reason, the long report also includes a summary of student responses for each individual problem in the assignment, allowing instructors to determine the most common mistakes students in this section are making for these difficult problems.

If you open the long report we generated (Stat101hwk_Topic06_AB-long.pdf), then you can view the summary for each of the 89 problems in this assignment. Each problem summary is printed on a separate page, beginning on Page 12 and ending on Page 100 of the report. As we saw in Figure 7, the first problem in the assignment had a mean score of 40%. We can ascertain what types of errors students from this section made on this problem by viewing the summary for the first problem (on Page 12 of the report), which provided below in Figure 8.

(1) Question "T06.A.A.04.1.1.MC.1" is given on the right. This question was selected from the question set with a frequency of 0.25. The question was administered to 10 out of the total of 50 students. The average score was 0.4 out of 1.

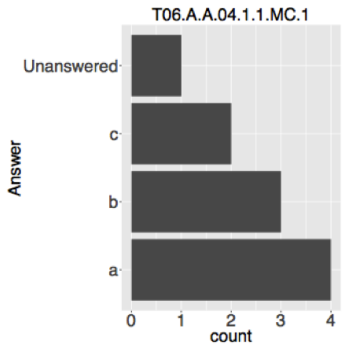
The z-score for a particular observation is z = 1.5. This means the observation is

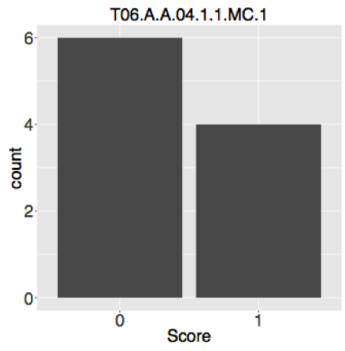
*a. 1.5 standard deviations above the mean.

b. 1.5 standard deviations below the mean.

c. 1.5 units above the mean.

d. 1.5 units below the mean.





Answer	Count
a	4
b	3
c	2
Unanswered	1

Summary	Value
Mean	0.40
Std.dev	0.52
Min	0.00
Median	0.00
Max	1.00

Figure 8: A detailed summary of student performance for the first question in the assignment. We see that 10 of the 50 students in this section received this problem at random out of a learning set that contains four similar problems. Only 4 of these 10 students correctly selected answer choice A. The most frequently incorrect response from students was answer choice B, followed by answer choice C. No student incorrectly responded with answer choice D, and one student did not respond.

One topic comparing multiple sections - short version

An instructor may be motivated to generate a short report for one topic across multiple sections if they are seeking answers to the following types of questions:

1. For this topic, is there consistency across the sections in terms of overall performance, and in terms of performance by learning outcome and question set?
2. Combining students from all sections, are there any disparities in student performance between questions from the same question set?
3. Combining students from all sections, are there any learning outcomes or question sets for which students consistently tend to perform poorly?

Code

Now that we have successfully generated reports for both sections of `Topic 06` separately, we may wish to generate a report that compares the performance between these two sections (AB and CD). In this case, we use the same `makeReport()` function as before. However, for our parameter `dataFile`, we will input a list `dataList` that lists the data files for both sections we wish to compare. Of course,

we must also correctly specify the `reportType` parameter to indicate that we wish to generate a short report that compares sections for a given topic:

```
dataFolder = system.file("inst/extdata/DataFiles/Topic06", package = "ePort")
dataList = list.files(path = dataFolder, full.names = TRUE)[1:2]
makeReport(keyFile = keyPath, dataFile = dataList, loFile = loPath, outFile = outPath, reportType = "
```

Please ensure that the variable `dataList` contains the full pathway to the two data files (Topic06.AB.csv and Topic06.CD.csv). If your variable contains additional pathways, then you have more than just the two necessary input data files in your DataFiles/Topic06 folder. In this case, you will need to move these extraneous files elsewhere or hardcode the two full pathways into the `dataList` variable.

Output

You should be able to examine the report we just generated (Stat101hwk_Topic06_crossSection-short.pdf) in our example `OutputFiles` folder. One of the most immediate information that a course coordinator may wish to glean is whether or not there are discrepancies in student performance for a given topic across multiple sections. This could be especially true if there are different instructors and/or different teaching methods being employed across the different sections. In the report we produced, we can indeed quickly compare how the students performed overall for Topic 06 for both sections AB and CD, in a plot that has been copied here in Figure 9.

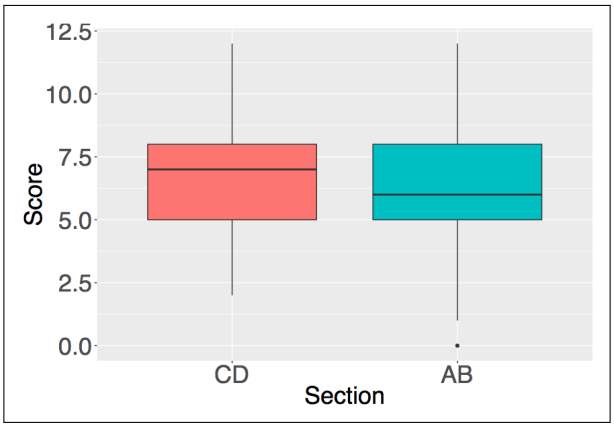


Figure 9: Boxplot of student performance in sections AB and CD for Topic 06. The minimum, IQR, and maximum are similar between the two sections. However, section CD has a larger median than section AB, which has some low-scoring outliers.

One topic comparing multiple sections - long version

An instructor may be motivated to generate a long report for one topic across multiple sections if they are seeking answers to the following types of questions:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! FILL THIS IN !!!!!!!!!!!!!!!!!!!!!!!!!

1. Based on how students from this section performed on this topic, are there any noticeable problems with questions in certain question sets not being equally difficult?

2. Which students from this section scored poorly on each learning outcome for this homework assignment?

3. For each question in the assignment (particularly the difficult ones), what was the distribution of student responses?

Code

We received a brief comparative summary between the sections of Topic 06 in the previous section. If we would like to see a more detailed version of this report that includes comparative information for each question in the topic of interest, then we can simply run the previous line of code, only now specifying the `reportType` parameter to indicate that we wish to generate a long report that compares sections for a given topic:

```
makeReport(keyFile = keyPath, dataFile = dataList, loFile = loPath, outFile = outPath, reportType = "
```

Output

The code above should have generated the long edition of the report (Stat101hwk_Topic06_crossSection-long.pdf). While the short report allowed us to compare how students performed overall between sections AB and CD in Topic 06, the long report will allow us to make more detailed comparisons between the sections at the level of learning outcomes and question sets. For instance, the long report includes comparative information at the level of learning outcomes, as is illustrated in Figure 10.

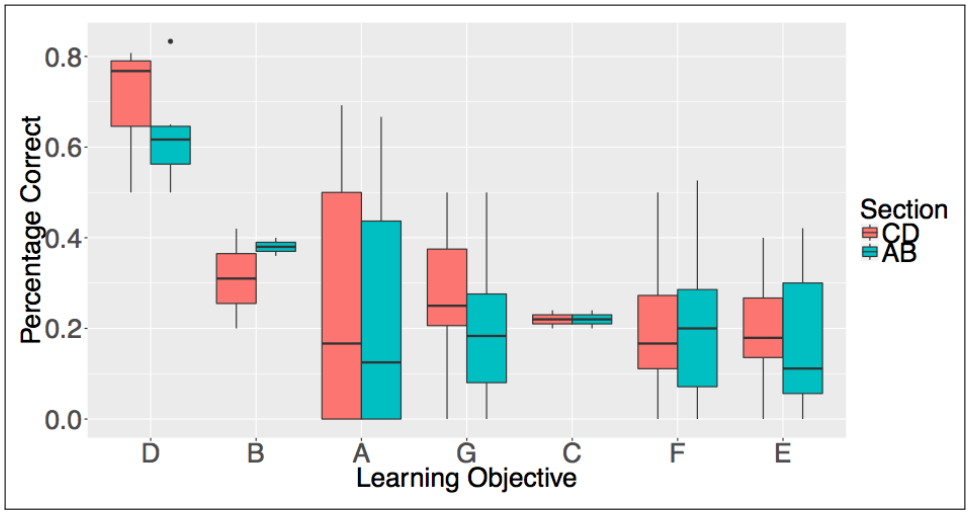


Figure 10: Each pair of boxplots compares the student performace between sections AB and CD for each learning outcome. For each learning outcomes, the two sections were fairly comparable in their performances, although section CD outperformed in learning outcome D, and section AB outperformed for learning outcome B. Both sections separately showed superior performance in learning outcome D than they did for all other learning outcomes.

The long edition of the report breaks it down further into comparing student performances between the sections for each question set. This is demonstrated below in Figure 11.

Qset	LO	#Qn	Overall	CD	AB
I	D	2	72.00	80.00	64.00
J	D	4	61.00	64.00	58.00
B	A	4	51.00	54.00	48.00
A	A	4	46.00	48.00	44.00
E	B	1	41.00	42.00	40.00
M	F	5	29.00	22.00	36.00
U	G	5	29.00	32.00	26.00
F	B	1	28.00	20.00	36.00
S	G	5	26.00	26.00	26.00
H	C	1	24.00	24.00	24.00
R	G	5	24.00	30.00	18.00
Q	F	5	23.00	16.00	30.00
L	E	9	22.67	22.00	23.33
G	C	1	20.00	20.00	20.00
N	F	5	19.00	20.00	18.00
O	F	5	18.00	22.00	14.00
T	G	5	16.00	16.00	16.00
K	E	9	14.67	19.33	10.00
P	F	5	11.00	16.00	6.00
C	A	4	0.00	0.00	0.00
D	A	4	0.00	0.00	0.00

Figure 11: Mean percentage scores for each question set by section. The question sets are sorted by decreasing overall mean scores. Both question sets with the highest mean scores are from learning outcome D. We discover that learning outcome A had two question sets (A and B) with high mean scores, but two question sets (C and D) with low mean scores.

The long edition of the report also offers information that does not explicitly compare the sections of interest, but instead, combines data from all the sections of interest to generate more power in any analysis that we could instead do separately using just the data from one section. For instance, we previously demonstrated certain tools that are available in the individual section reports that can allow instructors to confirm that all students will receive equally-challenging randomly-selected subsets of problems for their assignments; this was shown in Figure 7. A course coordinator could again explore this same motivation, only now using the data combined from students across both sections, as is displayed in Figure 12. Please note that this image is abridged for demonstration purposes; if you were to view the same image in the long report you just generated, then you would see that indeed it is a comprehensive table that consecutely lists the pertinent information for all 89 questions in this assignment.

ID	LO	Qset	Name	Type	FullPt	QinSet	N	CrtPct	Count	NA's	Mean	Std	Flag
1	A	A	1	MC	1	1	4	46.15	26	74	0.46	0.51	
2	A	A	2	MC	1	1	4	54.17	24	76	0.54	0.51	*
3	A	A	3	MC	1	1	4	36.36	22	78	0.36	0.49	*
4	A	A	4	MC	1	1	4	46.43	28	72	0.46	0.51	
⋮													
21	D	I	feet	TF	1	1	2	73.91	46	54	0.74	0.44	
22	D	I	lowtemp	TF	1	1	2	70.37	54	46	0.70	0.46	

Figure 12: Comparison of student performance combined from sections AB and CD on each question of Topic 06, grouped by learning set. If any pair of questions in a given learning set have a difference in mean scores of more than 15%, then both questions are flagged. Question set A contained four problems, and when we only used student performance information from the 50 students in section AB, three IDs (2, 3, and 4) were flagged as having a mean score that was at least 15% different than the mean score of another problem in the same question set (see Figure 7). However, now that we have used student performance information from all 100 students in both setions AB and CD, only two IDs (2 and 3) remain flagged. In contrast, question set I contained two problems, and neither of these problems were flagged when we only used student performance information from the 50 students in section AB. This remains the case now that we have used student performance information from all 100 students in both setions AB and CD.

As per the figure above, an instructor can quickly determine if there are problems for which students tend to perform poorly, and if there question sets for which students tend to show inconsistent performance between the individual questions. In either situation, an instructor may wish to examine these problems more closely, and determine how students are answering these problems incorrectly. For this reason, the long report also includes a summary of student responses for each individual problem in the assignment, allowing instructors to determine the most common mistakes students in this section are making for these difficult problems.

If you open the long report we generated (Stat101hwk_Topic06_crossSectionlong.pdf), then you can view the summary for each of the 89 problems in this assignment. Each problem summary is printed on a separate page, beginning on Page 12 and ending on Page 100 of the report. As we saw in Figure 12, the first problem in the assignment had an overall mean score of 46% across both sections. We can ascertain what types of errors students made on this problem, regardless of section, by viewing the summary for the first problem (on Page 12 of the report), which provided below in Figure 13. Of course, now that we are collecting student responses from the 26 students who received this problem in sections AB and CD, our figure will provide us with more power about the distribution of student responses for this problem than when only considered the 10 students in section AB, as was done in Figure 8.

(1) Question "T06.A.A.04.1.1.MC.1" is given on the right. This question was selected from the question set with a frequency of 0.25. The question was administered to 26 out of the total of 100 students. The average score was 0.46 out of 1.

(Back to the question summary Table 7.)

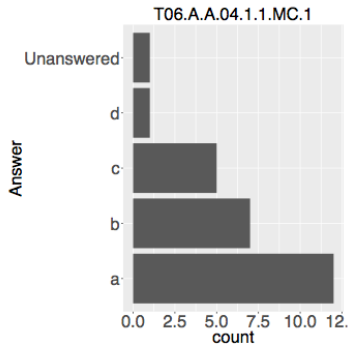
The z-score for a particular observation is z = 1.5. This means the observation is

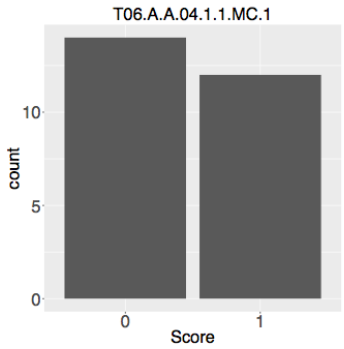
*a. 1.5 standard deviations above the mean.

b. 1.5 standard deviations below the mean.

c. 1.5 units above the mean.

d. 1.5 units below the mean.





Answer	Count
a	12
b	7
c	5
d	1
Unanswered	1

Summary	Value
Mean	0.46
Std.dev	0.51
Min	0.00
Median	0.00
Max	1.00

Figure 13: A detailed summary of student performance for the first question in the assignment. Only 12 of 26 students correctly selected answer choice A. The most frequently incorrect response from students was answer choice B, followed by answer choice C, and then answer choice D. One student did not respond. This is a similar distribution to what we saw in Figure 8, when we had less power by only looked at the 10 students from section AB.

One unit (group of topics) for one section

An instructor may be motivated to generate a long report for one topic across multiple sections if they are seeking answers to the following types of questions:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! FILL THIS IN !!!!!!!!!!!!!!!!!!!!!!!!!!

1. Based on how students from this section performed on this topic, are there any noticeable problems with questions in certain question sets not being equally difficult?
2. Which students from this section scored poorly on each learning outcome for this homework assignment?
3. For each question in the assignment (particularly the difficult ones), what was the distribution of student responses?

Code

```
dataFolder = system.file("inst/extdata/DataFiles/Topic03_06", package="ePort")
dataList = list.files(path = dataFolder, full.names = TRUE)
```

```
for (file in dataList){
  rewriteData(file)
}
dataTable = setDir(dataFolder)
mergedData = mergeData(dataTable)
# Add this to makeReport if we are using this file!!!!!!!!!!!!!!!!!!
for (sctn in unique(dataTable$section)) {
  merged = subsetData(mergedData, dataTable, choice = sctn)
  knit("/Users/lindz/ePort/inst/Rnw/hw-topic.Rnw", output = paste0('Stat101hwk_Unit1_Section',sctn, '
}
```

Output

An instructor might want to know how their section of students is performing throughout the course duration, and in partiular, if there are any topics for which their students perform well or poorly. The report we just generated (Stat101hwk_Unit1_SectionAB.pdf) provides tools that could inform such instructors. For instance, we can obtain a quick summary of how students performed for this section AB for both Topic 03 and Topic 06, as is provided in Figure 14.

	Mean	Std.dev	Min	Median	Max
Topico3	23.80	18.74	0.00	17.14	88.57
Topico6	24.96	9.51	0.00	24.00	48.00

Figure 14: Overall summary of student performance for Topic 03 and Topic 06. Students from this section showed a higher mean and median for Topic 06, but a larger standard deviation for Topic 03.

We can also compare this section of student performances between these two topics, not only at an overall level, but also at the individual student level, see Figure 15.

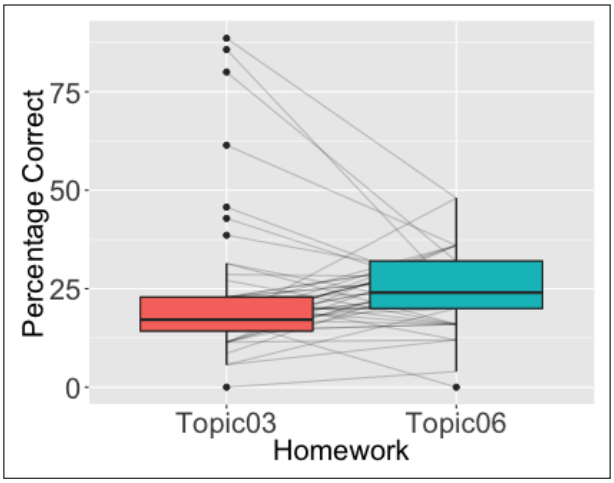


Figure 15: Boxplots with parallel coordinate plot overlaid. Each boxplot represents the five-number summary of scores in each topic, and each parallel coordinate line represents scores of an individual student. We see that many of the students who did not perform as well in Topic 06 were outliers for high scores for Topic 03.

While the figure above can allow us to see if there are trends in how individual students scored between the topics, if we find patterns of interest, we may wish to determine the identity of individual students. This is provided below in Figure 16.

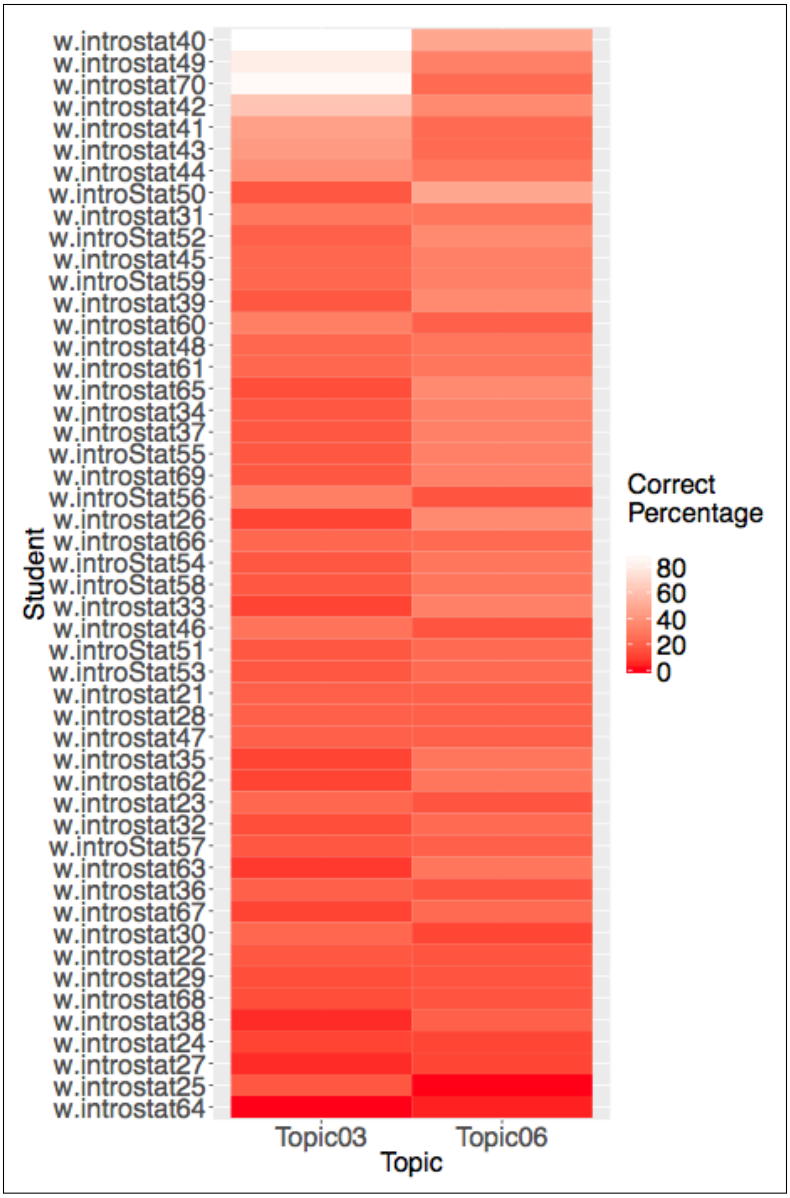


Figure 16: Heat map of the scores for both topics of interest for students in the section. Some students (such as w.introstat64) performed poorly in both topics. Other students (such as w.introstat40, w.introstat49, and w.introstat70) performed much better in Topic 03 than they did in Topic 06.

One unit (group of topics) comparing multiple sections

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! FILL THIS IN !!!!!!!!!!!!!!!!!!!!!!!!!

1. Based on how students from this section performed on this topic, are there any noticeable problems with questions in certain question sets not being equally difficult?
2. Which students from this section scored poorly on each learning outcome for this homework assignment?
3. For each question in the assignment (particularly the difficult ones), what was the distribution of student responses?

Code

```
merged = subsetData(mergedData, dataTable)
knit("/Users/lindz/ePort/inst/Rnw/hw-topic-section.Rnw", output = 'Stat101hwk_Unit1_allSections.tex')
```

Output

From the report we just generated (Stat101hwk_Unit1_allSections.pdf), we can simultaneously examine how students performed across different sections and different topics. As is the case with previous reports, we are provided some basic summaries, as provided in Figure 17.

	Topic03	Topic06
AB	23.80	24.96
CD	32.71	27.04

Figure 17: Mean scores for students in both sections and topics. For both topics, section CD had a higher mean.

In this report, we can also compare score distributions for each combination of section and topic, see Figure 18.

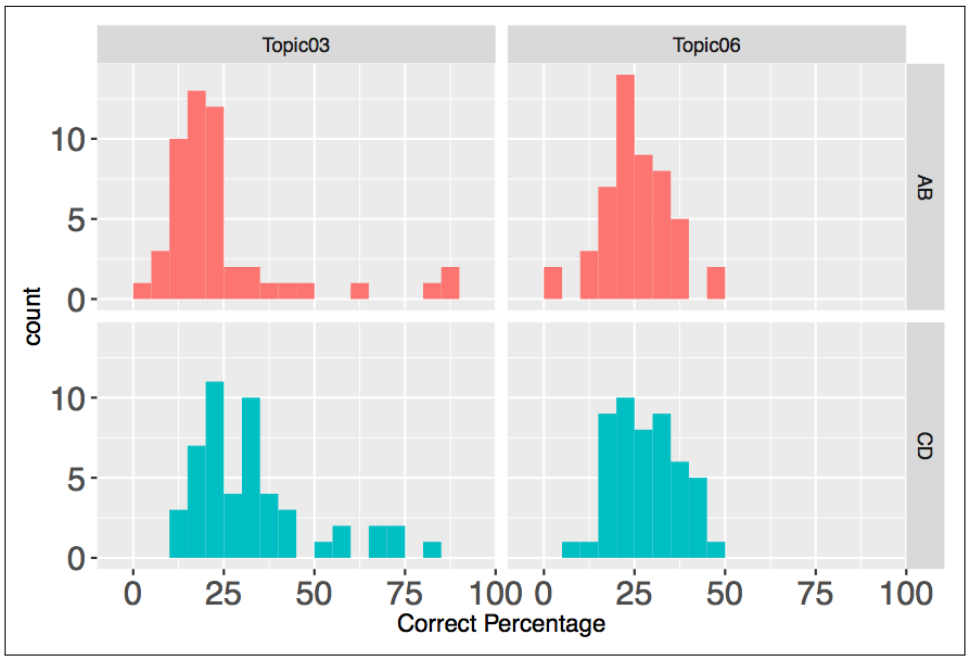


Figure 18: Histogram of student scores for sections AB and CD and topics 03 and 06. Both sections had large-scoring outliers in Topic 03.

One other feature of this report is a tabulation of missing homework for both sections and both topics (see Figure 19). If there are patterns found for this, course coordinators may wish to determine what is causing students to tend to miss homework assignments for a certain section or certain topic.

	Total	Missing.Topic03	Missing.Topic06
AB	50	0	0
CD	50	0	0

Figure 19: Missing homework summary for both sections and both topics of interest. In both sections, the fifty students submitted homework assignments for both Topic 03 and 06.

Additional tools

Splitting files

In some cases, one homework assignment may span across two topics. In order to generate separate reports for each of the two topics, we will first need to split the data file. In our example, we will split the Topic 03 data files for both sections AB and CD. We have a special folder in our example data to complete this process; if you examine the folder called `Topic03_Split`, which is located in our example `DataFiles` folder, then you will see that it contains the two files, Topic03.AB.csv and Topic03.CD.csv.

If you open either of these files, then you will notice there is a 'Question ID' column and a 'Question' column (which contains Question RsQ). The Question ID refers to the question order that each student

completed; in this example assignment, there are 31 Question IDs. Hence, each student received 31 problems, and should have Question IDs for each of these problems consecutively increasing from Question ID 1 to Question ID 31. In contrast, the Question RsQ refers to the absolute ID from Respondus. In this assignment, in total, there were 86 problems in Respondus, from which each student received their randomly-selected subset of 31 problems. Hence, the Question RsQ should range from 1 to 86.

Because of this, if an instructor wants to split the data files, there are two parameters they can use to designate the cut location. If you run a help function on the `ePort` function `splitFile()`, then you see that these cut types are called 'RsQ' and 'ID'. Let us say that we need to split these data files into one file that only contains the first 9 ID Questions and another file that only contains the last 22 ID Questions. In that case, we can choose a cut type of 'ID' with a value of 9, or a cut type of 'RsQ' with a value of 27. We show this process below using the cut type of 'ID'.

```
dataFolder = system.file("inst/extdata/DataFiles/Topic03_Split", package="ePort")
dataList = list.files(path=dataFolder,full.names=TRUE)[1:2]
for(file in dataList){
  rewriteData(file)
  splitFile(file, 9, "ID")
}
```

Upon running the above code, if you return to the `Topic03_Split` folder, then you will see that four new files have been created, `Topic03.AB.part1.csv`, `Topic03.AB.part2.csv`, `Topic03.CD.part1.csv`, and `Topic03.CD.part2.csv`. You can verify that the 'part1' files contain the first 9 Question IDs, while the 'part2' files contain the last 22 Question IDs.

Merging files

In some cases, one topic might encompass two homework assignments. In order to generate a report for the overall topic, we will first need to combine the two data files. Such a scenario exists in the online database, in which chapter 9 and 12 homework assignments together form Topic 11.

In our example, we will combine the Chapter 9 and 12 homework assignments to form the Topic 11 data file for both sections AB and CD. We again have a special folder in our example data to complete this process. It is located in our example `DataFiles` folder in a subfolder called `Topic03_Merge`. Upon entering this folder, you should see that it contains four files, Ch12.AB.csv, Ch12.CD.csv, Ch9.AB.csv, and Ch9.CD.csv. Notice in the code below that we save the Topic number to a string variable called `topicNum`.

```
dataFolder = system.file("inst/extdata/DataFiles/Topic03_Merge", package="ePort")
dataList = list.files(path = dataFolder, full.names=TRUE)[1:4]
topicNum = "Topic11"
for (i in dataList) rewriteData(i)
for(i in c('AB', 'CD')){
  tmp = dataList[grep(i, basename(dataList))]
  combineFiles(tmp[2], tmp[1], paste(dirname(tmp[1]),"/",paste(topicNum, i, "csv", sep='.'),sep=""))
}
```

Running this code should produce two new files (Topic11.AB.csv and Topic11.CD.csv) into the `Topic03_Merge` folder. If we explore these files, we see that they have a maximum Question ID

value of 15 and a maximum RsQ value of 32. This can verify that this is correct if we explore the original files for the Chapter 9 and 12 homework assignments. The Chapter 9 homework data file had a maximum Question ID value of 10 and a maximum RsQ value of 23, while the Chaper 12 homework data file had a maximum Question ID value of 5, and a maximum RsQ value of 9. Hence, the total number of Question ID and RsQ values in the combined files appear to have been added correctly given the total number of Question ID and RsQ values from the individual files.

Deidentifying files

It should be noted that data files and the corresponding reports they generate contain student names. At times, it may be necessary to anonymize student names in a given data file, so that it (and any reports that can be generated from it) do not contain confidential information. You can most easily deidentify student names by transferring all data files that you wish to deidentify into a common folder that is otherwise empty. It is important to ensure that no extraneous files are in this common folder.

In this vignette, such an example folder has already been set up for you. The data files for sections AB and CD of Topic 03 are located in the `DataFiles/Topic03_Deidentified` folder of the `extdata` folder. If you examine the content of these files, then you will notice that the column called `Username` contains the names of students. Of course, these are not real student names, but instead are fictitious student names generated for example purposes.

As is demonstrated below, we first save the pathway to the directory that contains nothing but the files we wish to deidentify. Then, we call the `getNameList` function of `ePort`. We choose the `save = TRUE` option so that a dictionary file `nameCode.csv` will be generated. This dictionary file contains a list of the original student names and their corresponding deidentified codes. Lastly, we call the `encodeName` function, which will use the dictionary file we just created to translate the original data files into deidentified data files. If you complete this process, you will notice that the original data files have been overwritten.

```
dataFolder = system.file("inst/extdata/DataFiles/Topic03_Deidentified", package="ePort")
getNameList(dataFolder, section=NULL, semester=NULL, secblind=TRUE, save=TRUE)
encodeName(dataFolder, dict=paste(dataFolder, "nameCode.csv", sep='/'))
```

Running reports sequentially

If a course has multiple sections, and we wish to create an individual report for each of the many sections, then one inconvenient way to accomplish this would be to run the example code above, separately for each section at a time, with new data files each time. However, a more convenient way to accomplish the task would be to run all the reports at once. We are still using the same key and learning outcome files, although we would now need two data files (one with the answers from students in `Section AB` and one with the answers from students in `Section CD`).

We can hard code the two data files needed for the two sections into a vector called `dataListPath` as shown below:

```
dataListPath = c(system.file("inst/extdata/DataFiles/Topic06/Topic06.AB.csv", package="ePort"),
system.file("inst/extdata/DataFiles/Topic06/Topic06.CD.csv", package="ePort"))
```

Next, as demonstrated below, we can generate a report for both sections listed in our `dataListPath` object using a for loop. Our two data files of interest in the for loop (`Topic06.AB.csv` and `Topic06.CD.csv`) are from the same Topic, and so they share the same key file and learning outcome file. Hence, we do not have to run the `refineKey()` function on the key file for this Topic, as we have already completed this step earlier. However, our two data files of interest do not share the same data file. We have already executed the `rewriteData()` function for the `Topic06.AB.csv` data file, but we have not yet done so for the `Topic06.CD.csv` file. Hence, we must include the `rewriteData()` function in our for loop to ensure that both data files have this priming step completed.

```
for (i in dataListPath){
  rewriteData(i)
  makeReport(keyFile=keyPath, dataFile=i, loFile=loPath, outFile=outPath, reportType="secTopicShort")
}
```

We can also, however, simply

```
dataFolder = system.file("inst/extdata/DataFiles/Topic06", package="ePort")
#namelist = list.files(path=dataFolder, pattern = "^[^.]*\.[^.]*\.[^.]*£", full.names=FALSE)
#http://stackoverflow.com/questions/9949176/match-string-with-exactly-2-of-a-given-character-e-g-2-li

#dataListPath = c(system.file("inst/extdata/DataFiles/Topic06/Topic06.AB.csv", package="ePort"), syst
#topic = gsub('.Questions.txt','',gsub('Topic','',basename(key)))
#namelist = list.files(path=dataPath,full.names=TRUE)
#namelist = namelist[grep(paste('Topic',topic,'\\.',sep=''),basename(namelist))]
#namelist
```

Report Options

As this document may be used for course coordinators to evaulate a large amount of data (if inputting many sections and topics), there are additional tools that may be used.
- analyze effects of sections and topics on students performance, we consider the generalized linear mixed model, with the response (homework score), and the predictors (section and topic) - A mixed effects model is considered since we have multiple measurements on each student, while the students may have different ability to study. The fixed effects are topic (t), section (s), and their interaction (ts). The random effect is student (u). - clustering analysis - During a semester of 16 weeks, students may not keep the same pace in studying. Some students are smart and working very hard all the time. Some students start full of energy, and gradually lose their passions. Some students do not take the course seriously until they are challenged by some difficult content. We are interested in finding some featured behaviors along with time, by clustering the students into groups. The result of hierarchical clustering on the scaled scores is shown in Figure 5.

Conclusions

The `ePort` package offers various plotting tools that can assist those studying genealogical lineages in the data exploration phases. As each plot comes with its advantages and disadvantages, we recommend for users to explore several of the available visualization tools.
This vignette briefly introduced some of the capabilities of the `ePort` package. Inevitably, new approaches will necessitate new features in subsequent versions and might reveal unforeseen bugs. Please send comments, suggestions, questions, and bug reports to `amyf@iastate.edu`.