Using EdSurvey to Analyze TIMSS Data

Ting Zhang, Paul Bailey, and Michael Lee[*][†]

July 02, 2020

Overview of the EdSurvey Package

Data from large-scale educational assessment programs, such as the Trends in International Mathematics and Science Study (TIMSS), require special statistical methods to analyze because of their scope and complexity. The `EdSurvey` package gives users functions to perform analyses that account for both the complex sample survey design and the use of plausible values.

The `EdSurvey` package also seamlessly takes advantage of the `LaF` package to read in data only when required for an analysis. Users with computers that have insufficient memory to read in entire datasets can still do analyses without having to write special code to read in just the appropriate variables. This is all addressed directly in the `EdSurvey` package—behind the scenes and without any special tuning by the user.

Vignette Outline

This vignette will describe the basics of using the `EdSurvey` package for analyzing international educational assessment data using TIMSS data as an example. The vignette sections are organized as follows:

- Notes
  - Vignette notation
  - Software requirements
- Setting up the environment for analyzing NCES data
  - Installing and loading `EdSurvey`
  - Philosophy of conducting analyses using the `EdSurvey` package
  - Downloading data
  - Reading in data
  - Getting to know the data format
  - Removing special values
- Explore variable distributions with `summary2`
- Subsetting the data
- Retrieving data for further manipulation with `getData`

[*]This publication was prepared for NCES under Contract No. ED-IES-12-D-0002 with the American Institutes for Research. Mention of trade names, commercial products, or organizations does not imply endorsement by the U.S. Government.
[†]The authors would like to thank Dan Sherman and Yuqi Liao for reviewing this document.

    - Retrieving a set of variables in a dataset
    - Retrieving all variables in a dataset
    - Applying `rebindAttributes` to use `EdSurvey` functions with manipulated data frames
- Correlating variables with `cor.sdf`
    - Weighted correlations
    - Unweighted correlations
- Making a table with `edsurveyTable`
- Computing the percentages of students Benchmarks with `achievementLevels`
- Calculating percentiles with `percentile`
- Preparing an `edsurvey.data.frame.list`
    - Recoding variable names and levels using `recode.sdf` and `rename.sdf`
    - Combining several `edsurvey.data.frame` objects into a single object
    - Recommended workflow for standardizing variables in analyses across years or jurisdictions
- Estimating the difference in two statistics with `gap` analysis
    - Performing gap analysis and understanding the summary output
    - Gap analysis of achievement levels and percentiles
    - Gap analysis of jurisdictions
- Regression analysis with `lm.sdf`
- Multivariate regression with `mvrlm.sdf`
- Logistic regression analysis with `logit.sdf`
    - oddsRatio
    - waldTest
- Quantile regression analysis with `rq.sdf`
- Mixed models with `mixed.sdf`
- Endnotes
    - Memory usage
    - Factors and factor analysis
    - Summary and next steps
    - Additional resources
    - Methodology resources
    - Reference
- Appendix A
- Appendix B

Vignette Notation

This vignette displays examples using the notation for R console input and output. R console input will be displayed within a gray box:

```r
inputCode <- c(2,"neat")
```

R console output will be displayed next to a double hash mark (`##`). Here is an example where the user types `inputCode` into the console and the code output R gives after the double hash marks:

```r
inputCode
```

```
## [1] "2"    "neat"
```

Software Requirements

Unless you already have R version 3.5.0 or later, install the latest R version—which is available online at https://cran.r-project.org/. Users also may want to install RStudio desktop, which has an interface that many find easier to follow. RStudio is available online at https://www.rstudio.com/products/rstudio/download/.

Setting Up the Environment for Analyzing NCES Data

Installing and Loading EdSurvey

Inside R, run the following command to install `EdSurvey` as well as its package dependencies:

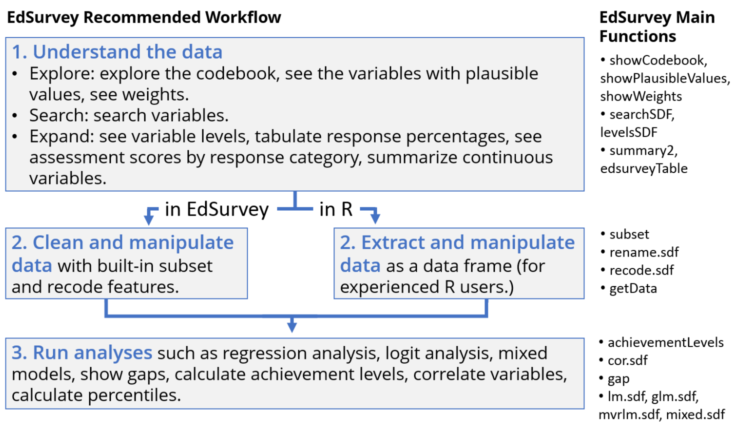
```r
install.packages("EdSurvey")
```

Once the package is successfully installed, `EdSurvey` can be loaded with the following command:

```r
library(EdSurvey)
```

Philosophy of Conducting Analyses Using the EdSurvey Package

Recognizing that researchers using R statistical software come with varying levels of experience, the `EdSurvey` package has provided multiple workflows to aid in this process of conducting survey analysis. The following graphic details the two recommended workflows:


EdSurvey Recommended Workflow
1. Understand the data
• Explore: explore the codebook, see the variables with plausible values, see weights.
• Search: search variables.
• Expand: see variable levels, tabulate response percentages, see assessment scores by response category, summarize continuous variables.
in EdSurvey
in R
2. Clean and manipulate data with built-in subset and recode features.
2. Extract and manipulate data as a data frame (for experienced R users.)
3. Run analyses such as regression analysis, logit analysis, mixed models, show gaps, calculate achievement levels, correlate variables, calculate percentiles.
EdSurvey Main Functions
• showCodebook, showPlausibleValues, showWeights
• searchSDF, levelsSDF
• summary2, edsurveyTable
• subset
• rename.sdf
• recode.sdf
• getData
• achievementLevels
• cor.sdf
• gap
• lm.sdf, glm.sdf, mvrlm.sdf, mixed.sdf

The workflow has three sections:

1. Understanding the data
2. Preparing the data for analysis
3. Running the analysis

The phase in which the two methods diverge is the second section. The `EdSurvey` package provides functions for users to clean and manipulate their data, but experienced R programmers might prefer to extract and manipulate their data using other R methods or supplementary packages to do so; each method is supported for performing `EdSurvey` analytical functions.

Downloading Data

If you do not have data ready in your computer, you can use `EdSurvey`'s download functions for each international study, including the following:

- TIMSS: Trends in International Mathematics and Science Study and TIMSS Advanced (`downloadTIMSS`, `downloadTIMSSAdv`)
- PIRLS: Progress in International Reading Literacy Study and ePIRLS (`downloadPIRLS`, `download_ePIRLS`)
- CIVED 1999 and ICCS: The Civic Education Study 1999 and International Civic and Citizenship Study (`downloadCivEDICCS`)
- ICILS: International Computer and Information Literacy Study (`downloadICILS`)
- PISA: The Program for International Student Assessment (`downloadPISA`)
- PIAAC: Program for the International Assessment of Adult Competencies (`downloadPIAAC`)
- TALIS: Teaching and Learning International Survey (`downloadTALIS`)

For TIMSS, use the `downloadTIMSS` function to download data to a directory that the user specifies; for example, `"C:/"`. Please note that the `downloadTIMSS` function currently works for TIMSS 2003, 2007, 2011, and 2015 data. One can also manually download desirable TIMSS data from the IEA Data Repository or at https://timssandpirls.bc.edu.

In this vignette, we will use the TIMSS 2015 U.S. data for fourth-grade students for most of the function illustrations. The following example shows how to download these data.

```r
downloadTIMSS(years = 2015, root = "C:/", cache=FALSE)
```

Each TIMSS dataset contains more than 300 files; hence, it often takes quite a long time to download. Setting `cache = TRUE` will cache the text version of files and save time for future use of the data, but the process can take several hours, so we leave it at the default level of `FALSE`.

The data will be automatically stored in a folder in the directory that you specified. For example, the 2015 TIMSS data will be saved in the "TIMSS2015" folder in the C drive. The R program assigns the folder name, but you can manually change it.

On the Mac OS, the user will need to set the root to a folder that exists. Here we use the user's home directory:

```r
# for Mac OS
downloadTIMSS(years = 2015, root = "~\")
```

Then, on the Mac OS, future calls to `readTIMSS` should have `"C:/TIMSS2015"` replaced with `"~/TIMSS2015"`.

Reading in Data

The next step to running an analysis is reading in the data. For each international study, a `read` function assists in reading and processing its data:

- TIMSS and TIMSS Advanced (`readTIMSS`, `readTIMSSAdv`)
- PIRLS and ePIRLS(`readPIRLS`, `read_ePIRLS`)
- CIVED 1999 and ICCS (`readCivEDICCS`)
- ICILS (`readICILS`)
- PISA (`readPISA`)
- PIAAC (`readPIAAC`)
- TALIS (`readTALIS`)

Each read function is unique given the differences across survey designs, but the functions typically follow a standard convention across functions for ease of use. To learn more about a particular read function, use `help(package = "EdSurvey")` to find the survey of interest and refer to its help documentation for guidance. This section shows how to read in TIMSS data using `EdSurvey`'s `readTIMSS` function.

The `readTIMSS` function creates an `edsurvey.data.frame` that stores information about the specific data files processed, including the location on disk and the file format/layout of those files, as well as information about the weights, achievement levels, omitted levels, and so forth for the user. A TIMSS `edsurvey.data.frame` includes information for all variables at multiple data levels: student, teacher, school, and/or home.

To load the TIMSS 2015 U.S. data for the fourth graders and create an `edsurvey.data.frame`, select the pathway to the TIMSS 2015 data folder, set `countries = c("usa")`, and `gradeLvl = "4"` and assign it the name `TIMSS15` with this call:

```
TIMSS15 <- readTIMSS(path = "C:/TIMSS2015/", countries = c("usa"), gradeLvl = "4")
```

The function uses the three-digit International Organization for Standardization country code to select countries to import (here, `"usa"`); additional countries can be added to the function call by inserting into the vector separated by a comma [e.g., `c("usa", "fin")`]. A list of country codes can be found at TIMSS 2015 User Guide for the International Database, exhibit 4.1. (pages 47—49). To select all countries, set `countries="*"`. Note that the `countries` argument used in `EdSurvey` (and this vignette) technically refers to "education systems" or "jurisdictions" because subnational entities also could participate in TIMSS and other international studies.

This function takes several minutes to run the first time; subsequent calls to `readTIMSS` are stored on the user's drive for easy access and near-instant retrieval.

Once read in, student, teacher, school, and/or home data from a TIMSS dataset can be analyzed and merged after loading the data into the R working environment. The `readTIMSS` function is built to connect with the student data file, but it silently holds file formatting for other levels of the data when read. For instance, when teacher-level variables are requested by `getData` or any `EdSurvey` analytic function, the teacher-level data will always be returned merged to the student-level data. If school-level variables are requested, only then will the school data be merged. If both teacher-level and school-level variables are requested, these two-level data will be merged with the student-level data.

Beginning in TIMSS 2015, a *Numeracy* dataset, which is designed to assess mathematics at the end of the primary school cycle for countries where most children are still developing fundamental mathematics skills, is also available. The *Numeracy* dataset is handled automatically for the user and is included within the fourth-grade dataset when one specifies `gradeLvl=4` in the `readTIMSS` function. Most *Numeracy* countries have a fourth-grade dataset in addition to their *Numeracy* dataset, but some do not. For countries that have both a *Numeracy* and a fourth-grade dataset, the two datasets are combined into one `edsurvey.data.frame` for that country. Data variables missing from either dataset are kept, with NA values inserted for the dataset records where that variable did not exist. Data variables common to both datasets are kept as a single data variable, with records retaining their original values from the source dataset. Consult the *TIMSS 2015 User Guide for the International Database* for further information.

Getting to Know the Data Format

Information about an `edsurvey.data.frame` can be obtained in several ways. To get general data information, simply call `print` by typing the name of the `data.frame` object (i.e., `TIMSS15`) in the console.

```
TIMSS15
```

```
## edsurvey.data.frame for 2015 TIMSS (Mathematics and Science) in United
##   States
## Dimensions: 12119 rows and 2042 columns.
##
## There are 4 full sample weights in this edsurvey.data.frame:
##   'totwgt' with 150 JK replicate weights (the default).
##
##   'tchwgt' with 150 JK replicate weights.
##
##   'matwgt' with 150 JK replicate weights.
##
##   'sciwgt' with 150 JK replicate weights.
##
##
## There are 16 subject scale(s) or subscale(s) in this edsurvey.data.frame:
## 'mmat' subject scale or subscale with 5 plausible values (the default).
##
## 'ssci' subject scale or subscale with 5 plausible values.
##
## 'mdat' subject scale or subscale with 5 plausible values.
##
## 'mgeo' subject scale or subscale with 5 plausible values.
##
## 'mnum' subject scale or subscale with 5 plausible values.
##
## 'sear' subject scale or subscale with 5 plausible values.
##
## 'slif' subject scale or subscale with 5 plausible values.
##
## 'sphy' subject scale or subscale with 5 plausible values.
##
## 'mkno' subject scale or subscale with 5 plausible values.
##
## 'mapp' subject scale or subscale with 5 plausible values.
##
## 'mrea' subject scale or subscale with 5 plausible values.
##
## 'skno' subject scale or subscale with 5 plausible values.
##
## 'sapp' subject scale or subscale with 5 plausible values.
##
## 'srea' subject scale or subscale with 5 plausible values.
```

```
##
## 'mibm' subject scale or subscale with 5 plausible values.
##
## 'sibm' subject scale or subscale with 5 plausible values.
##
##
## Omitted Levels: 'Multiple', 'NA', 'OMITTED', 'OMITTED OR INVALID', 'OMITTED
##                 (BLANK ONLY)', 'BLANK(OMITTED)', 'BLANK(MISSING)',
##                 'BLANK(MISS)', 'MIS.', 'MISS.', 'N. REA.', 'N.REA.', 'N.
##                 ADM.', 'N. ADMIN.', 'NOT ADMIN', 'NOT APPL', 'NOT
##                 ADMINISTERED', 'NOT REACHED', 'NOT ADMIN.', 'NOT
##                 APPLICABLE', 'LOGICALLY NOT APPLICABLE', 'MISSING', and
##                 '(Missing)'
## Achievement Levels:
## Low International Benchmark: 400
## Intermediate International Benchmark: 475
## High International Benchmark: 550
## Advanced International Benchmark: 625
```

Some of the basic functions that work on a `data.frame`, such as `dim`, `nrow`, and `ncol`, also work on an `edsurvey.data.frame`.[1] They help check the dimensions of `TIMSS15`.

```r
dim(TIMSS15)
```

```
## [1] 12119  2042
```

```r
nrow(TIMSS15)
```

```
## [1] 12119
```

```r
ncol(TIMSS15)
```

```
## [1] 2042
```

Please note that calling the `dim` function for a TIMSS `edsurvey.data.frame` will result in the row count as if the teacher dataset was merged. This row count will be considered the "full data N" of the `edsurvey.data.frame`. even if no teacher data were included in an analysis. The column count returned by `dim` will be the count of unique column variables across all three data levels.

The `colnames` function can be used to list all variable names in the data:

To conduct a more powerful search of `TIMSS15` data variables, use the `searchSDF` function, which returns variable names and labels from an `edsurvey.data.frame` based on a character string such as `"book"`.

```r
searchSDF(string = "book", data = TIMSS15)
```

```
##   variableName                                 Labels
## 1       idbook                             Booklet ID
## 2      m051039         NUMBER OF BOOKS MARY CAN BUY (1)
## 3      m051533      HOW MANY BOOKS WILL FILL THE BOX (1)
```

[1] Use `?function` in the R console to view documentation on base R and `EdSurvey` package functions (e.g., `?readTIMSS` or `?lm.sdf`).

```
## 4       m061252 EXPRESSION TO FIND NUMBER OF BOOKS LUKE READ (D)
## 5        asbg04               GEN\\AMOUNT OF BOOKS IN YOUR HOME
## 6       asbh02a                      GEN\\HOW OFTEN\\READ BOOKS
## 7       asbh02f                 GEN\\HOW OFTEN\\BOOK DISCUSSION
## 8        asbh13                    GEN\\AMOUNT OF BOOKS AT HOME
## 9        asbh14       GEN\\AMOUNT OF BOOKS FOR CHILDREN AT HOME
## 10      mr51039                 NUMBER OF BOOKS MARY CAN BUY (1)
## 11      mr51533             HOW MANY BOOKS WILL FILL THE BOX (1)
## 12      mi51039                 NUMBER OF BOOKS MARY CAN BUY (1)
## 13      mi51533             HOW MANY BOOKS WILL FILL THE BOX (1)
## 14     acbg13aa         GEN\\BOOKS WITH DIFFERENT TITLES\\PRINT
## 15     acbg13ab       GEN\\BOOKS WITH DIFFERENT TITLES\\DIGITAL
## 17      atbs03i               SCI\\ASK STUDENTS\\READ TEXTBOOKS
```

The levels and labels for each variable searched via `searchSDF()` can also be returned by setting `levels = TRUE`. Let's use one of book variables "asbh02a" as an example:

```
searchSDF(string = "asbh02a", data = TIMSS15, levels = TRUE)
```

```
## Variable: asbh02a
## Label: GEN\HOW OFTEN\READ BOOKS
## Levels (Lowest level first):
##      1. OFTEN
##      2. SOMETIMES
##      3. NEVER OR ALMOST NEVER
##      9. OMITTED OR INVALID
```

The `|` (OR) operator can be used to search several strings simultaneously:

```
searchSDF(string="book|home|internet", data=TIMSS15)
```

```
##    variableName                                           Labels
## 1        idbook                                       Booklet ID
## 2       m051039                 NUMBER OF BOOKS MARY CAN BUY (1)
## 3       m051533             HOW MANY BOOKS WILL FILL THE BOX (1)
## 4       m061252 EXPRESSION TO FIND NUMBER OF BOOKS LUKE READ (D)
## 5        asbg03         GEN\\OFTEN SPEAK <LANG OF TEST> AT HOME
## 6        asbg04               GEN\\AMOUNT OF BOOKS IN YOUR HOME
## 7       asbg05a          GEN\\HOME POSSESS\\COMPUTER TABLET OWN
## 8       asbg05b       GEN\\HOME POSSESS\\COMPUTER TABLET SHARED
## 9       asbg05c                  GEN\\HOME POSSESS\\STUDY DESK
## 10      asbg05d                    GEN\\HOME POSSESS\\OWN ROOM
## 11      asbg05e         GEN\\HOME POSSESS\\INTERNET CONNECTION
## 12      asbg05f                  GEN\\HOME POSSESS\\MOBILE OWN
## 13      asbg05g               GEN\\HOME POSSESS\\GAMING SYSTEM
## 14      asbg05h            GEN\\HOME POSSESS\\<COUNTRY SPECIFIC>
## 15      asbg05i            GEN\\HOME POSSESS\\<COUNTRY SPECIFIC>
## 16      asbg05j            GEN\\HOME POSSESS\\<COUNTRY SPECIFIC>
## 17      asbg05k            GEN\\HOME POSSESS\\<COUNTRY SPECIFIC>
## 18      asbg10a     GEN\\USE COMPUTER TABLET FOR HOMEWORK\\HOME
## 19      asbg10b   GEN\\USE COMPUTER TABLET FOR HOMEWORK\\SCHOOL
## 20      asbg10c    GEN\\USE COMPUTER TABLET FOR HOMEWORK\\OTHER
## 21      asbghrl                  Home Resources for Learning/SCL
## 22      asdghrl                  Home Resources for Learning/IDX
## 23      asdg05s                    Number of Home Study Supports
## 24      asbh02a                       GEN\\HOW OFTEN\\READ BOOKS
## 25      asbh02f                  GEN\\HOW OFTEN\\BOOK DISCUSSION
## 26      asbh09a               GEN\\HOW OFTEN DOES CHILD HOMEWORK
## 27       asbh13                     GEN\\AMOUNT OF BOOKS AT HOME
## 28       asbh14        GEN\\AMOUNT OF BOOKS FOR CHILDREN AT HOME
## 29      asbh17a               GEN\\HOME\\BORN IN COUNTRY\\FATHER
## 30      asbh17b               GEN\\HOME\\BORN IN COUNTRY\\MOTHER
## 31       asbh19 GEN\\HOW OFTEN DOES CHILD SPEAK LANGUAGE AT HOME
## 32      mr51039                 NUMBER OF BOOKS MARY CAN BUY (1)
## 33      mr51533               HOW MANY BOOKS WILL FILL THE BOX (1)
## 34      mi51039                 NUMBER OF BOOKS MARY CAN BUY (1)
## 35      mi51533               HOW MANY BOOKS WILL FILL THE BOX (1)
## 36     acbg13aa          GEN\\BOOKS WITH DIFFERENT TITLES\\PRINT
## 37     acbg13ab        GEN\\BOOKS WITH DIFFERENT TITLES\\DIGITAL
## 39      atbm07a            MAT\\HOW OFTEN MATH HOMEWORK ASSIGNED
## 40      atbm07b                       MAT\\TIME SPENT ON HOMEWORK
## 41     atbm07ca               MAT\\HOMEWORK\\CORRECT ASSIGNMENTS
## 42     atbm07cb                  MAT\\HOMEWORK\\DISCUSS HOMEWORK
## 43     atbm07cc              MAT\\HOMEWORK\\MONITOR COMPLETENESS
## 44      atbs03i                 SCI\\ASK STUDENTS\\READ TEXTBOOKS
## 45      atbs06a          SCI\\HOW OFTEN SCIENCE HOMEWORK ASSIGNED
## 46      atbs06b                       SCI\\TIME SPENT ON HOMEWORK
## 47     atbs06ca                            SCI\\HOMEWORK\\CORRECT
## 48     atbs06cb                  SCI\\HOMEWORK\\DISCUSS HOMEWORK
## 49     atbs06cc              SCI\\HOMEWORK\\MONITOR COMPLETENESS
```

A vector of strings is used to search for variables that contain multiple strings, such as both "book" and "home"; each string is present in the variable label and can be used to filter the results:

```
searchSDF(string=c("book","home"), TIMSS15)
```

```
##   variableName                                    Labels
## 1       asbg04         GEN\\AMOUNT OF BOOKS IN YOUR HOME
## 2       asbh13              GEN\\AMOUNT OF BOOKS AT HOME
## 3       asbh14 GEN\\AMOUNT OF BOOKS FOR CHILDREN AT HOME
```

We can also use `levelsSDF()`to return the levels and labels for a particular variable:

```
levelsSDF(varnames = "itsex", data = TIMSS15)
```

```
## Levels for Variable 'itsex' (Lowest level first):
##     1. FEMALE (n=5071)
##     2. MALE (n=4958)
##     9. OMITTED OR INVALID* (n=0)
##     NOTE: * indicates an omitted level.
```

The variables associated with plausible values and weights can be seen from the `showPlausibleValues` and `showWeights` functions, respectively, when the `verbose` argument is set to `TRUE`. Please see the full names of the plausible values for each subject and each subscale in Appendix A of this vignette. Appendix B explains how to decide which weights to use in TIMSS.

Here we hide the (lengthy) results, but the user can easily see them by running the same code.

Removing Special Values

The `EdSurvey` package uses listwise deletion to remove special values (e.g., “Not Administered,” “Omitted,” “Invalid”, and “Not Reached”) in all analyses by default. To use a different method, set `omittedLevels = FALSE` in a `EdSurvey` function (e.g., `getData` and `lm.sdf`). You can remove levels that you want to remove with a call to `subset`, which is discussed in the “Subsetting the Data” section of this vignette.

Explore Variable Distributions With `summary2`

The `summary2` function produces both weighted and unweighted descriptive statistics for a variable. This functionality is particularly useful for gathering response information for survey variables when conducting data exploration. For TIMSS data and other datasets that have a default weight variable, `summary2` produces weighted statistics by default. If the specified variable is a set of plausible values, and the `weightVar` option is non-`NULL`, `summary2` statistics account for both plausible values pooling and weighting.

```r
summary2(TIMSS15, "mmat")
```

```
## Estimates are weighted using weight variable 'totwgt'
##   Variable     N Weighted N     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 1     mmat 10029    3757302 224.3781 485.1651 542.9994 539.1556 595.8614 815.8607
##         SD NA's Zero-weights
## 1 81.49701    0            0
```

By specifying `weightVar = NULL`, the function prints out unweighted descriptive statistics for the selected variable or plausible values:

```r
summary2(TIMSS15, "mmat", weightVar = NULL)
```

```
## Estimates are not weighted.
##   Variable     N     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.       SD NA's
## 1 asmmat01 10029 197.9503 484.6619 542.6149 538.4445 594.5530 821.3848 81.48399    0
## 2 asmmat02 10029 254.4527 482.9293 541.3637 537.2122 594.3503 805.0983 82.06935    0
## 3 asmmat03 10029 220.8828 483.9967 540.9457 538.0488 595.6088 815.8132 81.67018    0
## 4 asmmat04 10029 229.6011 484.4857 541.1721 538.0045 595.5815 816.8415 81.87465    0
## 5 asmmat05 10029 219.0036 484.3985 541.6757 537.8948 594.4392 820.1658 81.92304    0
```

For a categorical variable, the `summary2` function returns the weighted number of cases, the weighted percent, and the weighted standard error. For example, the variable `asbg03` (HOW OFTEN SPEAK OF THE LANGUAGE OF THE TEST AT HOME) returns the following output:

```r
summary2(TIMSS15, "asbg03")
```

```
## Estimates are weighted using weight variable 'totwgt'
##               asbg03    N Weighted N Weighted Percent Weighted Percent SE
## 1          (Missing)   86   41244.74         1.097722           0.5373254
## 2             ALWAYS 6479 2431900.50        64.724645           1.1423340
## 3      ALMOST ALWAYS 1158  433069.47        11.526075           0.4554113
## 4          SOMETIMES 1888  692416.84        18.428564           0.8506162
## 5              NEVER  209   75956.31         2.021565           0.1508753
## 6 OMITTED OR INVALID  209   82714.31         2.201428           0.1872072
```

Note that by default, the `summary2` function includes omitted levels; to remove those, set `omittedLevels = TRUE`:

```
summary2(TIMSS15, "asbg03", omittedLevels = TRUE)
```

```
## Estimates are weighted using weight variable 'totwgt'
##          asbg03    N Weighted N Weighted Percent Weighted Percent SE
## 1        ALWAYS 6479 2431900.50        66.932861           1.1062444
## 2 ALMOST ALWAYS 1158  433069.47        11.919311           0.4478264
## 3     SOMETIMES 1888  692416.84        19.057293           0.8876543
## 4         NEVER  209   75956.31         2.090535           0.1574219
```

Subsetting the Data

A subset of a dataset can be used with `EdSurvey` package functions. In this example, a summary table is created with `edsurveyTable` after filtering the sample to include only female students (`itsex == "FEMALE"`) and whose parents' involvement (reported by teacher) is very high or high `(atbg06f == 1 | atbg06f == 2)`. Because `atbg06f` is a teacher variable, `weightVar = "tchwgt"` is used to ensure teacher weights are used in the analysis.

```
TIMSS15F <- subset(TIMSS15, itsex == "FEMALE" & (atbg06f == 1 | atbg06f == 2))
es2 <- edsurveyTable(formula = ssci ~ itsex + atbg06f, data = TIMSS15F, weightVar = "tchwgt")
```

```
es2
```

Table 1: es2

itsex	atbg06f	N	WTD_N	PCT	SE(PCT)	MEAN	SE(MEAN)
FEMALE	VERY HIGH	667	213350.5	31.7189	4.65619	577.1958	6.073735
FEMALE	HIGH	1483	459278.5	68.2811	4.65619	564.5101	5.007309

Retrieving Data for Further Manipulation With getData

Retrieving a Set of Variables in a Dataset

Although the `EdSurvey` package allows for rudimentary data manipulation and analysis directly on an `edsurvey.data.frame` connection, the function `getData()` can extract a dataset of variables for manipulation and analyses in the same manner as other `data.frame` objects. This object, referred to as a `light.edsurvey.data.frame`, can then be used with packaged `EdSurvey` analytical functions.

Variables are extracted from an `edsurvey.data.frame` and returned as a `light.edsurvey.data.frame` by specifying a set of variable names in `varnames` or by entering a formula in `formula`.[2]

To access and manipulate data for `itsex` and `atbg06f` variables in `TIMSS15`, call `getData`. Note that in the following code, the `head` function is used. This reveals only the first few rows of the resulting data.

[2]Use `?getData` for details on default `getData` arguments.

```
gddat <- getData(data = TIMSS15, varnames = c("itsex", "atbg06f"),
                                 omittedLevels = TRUE, addAttributes = TRUE)
head(gddat)
```

```
##    itsex atbg06f
## 1   MALE  MEDIUM
## 2   MALE     LOW
## 3   MALE  MEDIUM
## 4   MALE     LOW
## 5 FEMALE     LOW
## 6 FEMALE  MEDIUM
```

By default, setting `omittedLevels` to `TRUE` removes special values, such as multiple entries or `NA`s. `getData` tries to help by dropping the levels of factors for regression, tables, and correlations that are not typically included in analyses.

The `addAttributes = TRUE` ensures that the analysis functions shown so far can continue to be used with the resulting dataset, `gddat`.

Retrieving All Variables in a Dataset

The `getData` function gives users flexibility for efficiently managing computer resources. We recommend importing only the necessary variables, as some international assessment data files can be exceptionally large and take up several gigabytes of space on a disk. It is unfathomable to load all of such data files in the memory to perform either data exploration or analysis. That said, users who wish to extract all the data in an `edsurvey.data.frame` such as the `TIMSS15` can do so by defining the `varnames` argument as `names(TIMSS15)`, which will query all variables. Setting the arguments `omittedLevels` and `defaultConditions` to `FALSE` ensures that values that would normally be removed are included:

```
lsdf0 <- getData(data = TIMSS15, varnames = colnames(TIMSS15), addAttributes = TRUE,
                 omittedLevels = FALSE, defaultConditions = FALSE)
dim(lsdf0)
dim(TIMSS15)
```

Applying `rebindAttributes` to Use `EdSurvey` Functions With Manipulated Data Frames

A helper function that pairs well with `getData` is `rebindAttributes`. This function allows users to reassign the attributes from a survey dataset to a data frame that might have had its attributes stripped during the manipulation process. Once attributes have been rebinded, all variables—including those outside the original dataset—are available for use in `EdSurvey` analytical functions.

For example, a user might want to run a linear model using `mmat`, the default weight `totwgt`, the variable `itsex`, and the categorical variable `atbg06f` recoded into a binary variable. To do so, we can return a portion of the `TIMMS15` survey data as the `gddat` object with special values excluded. Next, use the `base` R function `ifelse` to conditionally recode the variable `atbg06f` by collapsing the levels `"VERY HIGH"` and `"HIGH"` into one level (`"High involvement"`) and all other levels into `"Medium to low involvement"`.

```
gddat <- getData(data = TIMSS15, varnames = c("itsex", "atbg06f", "totwgt", "mmat"),
                 omittedLevels = TRUE)
gddat$parInvolve <- ifelse(gddat$atbg06f %in% c("VERY HIGH",
                                                "HIGH"),
                           "High involvement", "Medium to low involvement")
```

From there, apply `rebindAttributes` from the attribute data `TIMSS15` to the manipulated data frame `gddat`. The new variables are now available for use in `EdSurvey` analytical functions:

```r
gddat <- rebindAttributes(gddat, TIMSS15)
lm2 <- lm.sdf(formula = mmat ~ itsex + parInvolve, data = gddat)
summary(lm2)
```

```
##
## Formula: mmat ~ itsex + parInvolve
##
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## Plausible values: 5
## jrrIMax: 1
## full data n: 12119
## n used: 10570
##
## Coefficients:
##                                         coef      se        t    dof  Pr(>|t|)
## (Intercept)                         562.2805  5.1130 109.9704 45.050 < 2.2e-16 ***
## itsexMALE                             7.4231  2.1219   3.4983 63.487 0.0008612 ***
## parInvolveMedium to low involvement -45.7216  6.2427  -7.3240 54.088 1.216e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared: 0.078
```

Additional details on the features of the `getData` function appear in the vignette titled *Using the `getData` Function in EdSurvey*.

Correlating variables with cor.sdf

The `EdSurvey` package features multiple correlation methods for data exploration and analysis that fully accounts for the complex sample design in TIMSS data by using the `cor.sdf` function.[3] This includes the following correlation procedures:

- Pearson product-moment correlations for continuous variables
- Spearman rank correlation for ranked variables
- polyserial correlations for one categorical and one continuous variable
- polychoric correlations for two categorical variables
- correlations among plausible values of the subject scales and subscales (marginal correlation coefficients, which uses the Pearson type)

Weighted Correlations

In the following example, `asbg10a` ("computer/tablet for homework at home"), `asbs06a` ("usually do well in science"), and the total student weight (`totwgt`) are read in to calculate the correlation using the Pearson method. Similar to other `EdSurvey` analytical functions, the data are removed automatically from memory after the correlation is run.

[3]Use `?cor.sdf` for details on default `cor.sdf` arguments.

```
cor_pearson <- cor.sdf(x = "asbg10a", y = "asbs06a", data = TIMSS15,
                       method = "Pearson", weightVar = "totwgt")
```

It is important to take note of the order of levels to ensure that correlations are functioning as intended. Printing a correlation object will provide a condensed summary of the correlation details and the order of levels for each variable:

```
cor_pearson
```

```
## Method: Pearson
## full data n: 12119
## n used: 9612
##
## Correlation: -0.01725057
##
## Correlation Levels:
##   Levels for Variable 'asbg10a' (Lowest level first):
##     1. EVERY DAY OR ALMOST EVERY DAY
##     2. ONCE OR TWICE A WEEK
##     3. ONCE OR TWICE A MONTH
##     4. NEVER OR ALMOST NEVER
##   Levels for Variable 'asbs06a' (Lowest level first):
##     1. AGREE A LOT
##     2. AGREE A LITTLE
##     3. DISAGREE A LITTLE
##     4. DISAGREE A LOT
```

Variables in `cor.sdf` can be recoded and reordered. Variable levels and values can be recoded given the desired specifications. Recoding can be useful when a level is very thinly populated (so that it might merit combination with another level) or when changing the value label to something more appropriate for a particular analysis. In the example that follows, `asbg10a` and `asbs06a` are correlated using the Pearson method, with the levels `"ONCE OR TWICE A MONTH"` and `"NEVER OR ALMOST NEVER"` of the variable `asbg04` being recoded to `"INFREQUENTLY"` within a list of lists in the `recode` argument:

```
cor_recode <- cor.sdf(x = "asbg10a", y = "asbs06a", data = TIMSS15,
                      method = "Pearson", weightVar = "totwgt",
                      recode = list(asbg10a = list(from = c("ONCE OR TWICE A MONTH",
                                                             "NEVER OR ALMOST NEVER"),
                                                    to = c("INFREQUENTLY"))))
cor_recode
```

```
## Method: Pearson
## full data n: 12119
## n used: 9612
##
## Correlation: -0.0151273
##
## Correlation Levels:
##   Levels for Variable 'asbg10a' (Lowest level first):
##     1. EVERY DAY OR ALMOST EVERY DAY
##     2. ONCE OR TWICE A WEEK
##     3. INFREQUENTLY
##   Levels for Variable 'asbs06a' (Lowest level first):
##     1. AGREE A LOT
##     2. AGREE A LITTLE
##     3. DISAGREE A LITTLE
##     4. DISAGREE A LOT
```

Changing the order of levels might be useful to modify a variable that is out of order or when reversing the orientation of a series. Note that reordering a level will not change the magnitude of the correlation coefficient but only the direction—from negative to positive in this example. The `reorder` argument also is suitable when implemented in conjunction with recoded levels. The variables `asbg10a` and `asbs06a` are correlated using the Pearson method in the following example, with the variable `asbg10a`'s values `"EVERY DAY OR ALMOST EVERY DAY","ONCE OR TWICE A WEEK","ONCE OR TWICE A MONTH","NEVER OR ALMOST NEVER"` being reordered to `"NEVER OR ALMOST NEVER", "ONCE OR TWICE A MONTH", "ONCE OR TWICE A WEEK", "EVERY DAY OR ALMOST EVERY DAY"` within a list:

```
cor_reorder <- cor.sdf(x = "asbg10a", y = "asbs06a", data = TIMSS15,
                      method = "Pearson", weightVar = "totwgt",
                      reorder = list(asbg10a = c("NEVER OR ALMOST NEVER",
                                                 "ONCE OR TWICE A MONTH",
                                                 "ONCE OR TWICE A WEEK",
                                                 "EVERY DAY OR ALMOST EVERY DAY")))

cor_reorder
```

```
## Method: Pearson
## full data n: 12119
## n used: 9612
##
## Correlation: 0.01725057
##
## Correlation Levels:
##   Levels for Variable 'asbg10a' (Lowest level first):
##     1. NEVER OR ALMOST NEVER
##     2. ONCE OR TWICE A MONTH
##     3. ONCE OR TWICE A WEEK
##     4. EVERY DAY OR ALMOST EVERY DAY
##   Levels for Variable 'asbs06a' (Lowest level first):
##     1. AGREE A LOT
##     2. AGREE A LITTLE
##     3. DISAGREE A LITTLE
##     4. DISAGREE A LOT
```

Note: As an alternative, recoding can also be completed within `getData`. To see additional examples of recoding and reordering, use `?cor.sdf` in the R console.

The Marginal Correlation Coefficient among two sets of plausible values can be calculated using the `cor.sdf` function with the Pearson method. The mathematics and science subject scales `mmat` and `ssci` are correlated in this example:

```
cor3_mcc <- cor.sdf(x = "mmat", y = "ssci", data = TIMSS15, method = "Pearson")
cor3_mcc
```

```
## Method: Pearson
## full data n: 12119
## n used: 10029
##
## Correlation: 0.8420299
```

Use the `showPlausibleValues` function to return the plausible values of an `edsurvey.data.frame` to be used in calculating the correlation coefficients between subject scales or subscales.

The `cor.sdf` function features multiple methods for data exploration and analysis using correlations. The following example shows the differences in correlation coefficients among the Pearson, Spearman, and polychoric methods using a `subset` of the `edsurvey.data.frame` data, where `itsex == 1` (saved as the `sdf_dnf` object), `asbg10a` ("computer/tablet for homework at home"), `asbs06a` ("usually do well in science"), and the full sample weight (`totwgt`):

```r
sdf_dnf <- subset(TIMSS15, itsex == 1)
cor_pearson <- cor.sdf(x = "asbg10a", y = "asbs06a", data = sdf_dnf,
                    method = "Pearson", weightVar = "totwgt")
cor_spearman <- cor.sdf(x = "asbg10a", y = "asbs06a", data = sdf_dnf,
                    method = "Spearman", weightVar = "totwgt")
cor_polychoric <- cor.sdf(x = "asbg10a", y = "asbs06a", data = sdf_dnf,
                    method = "Polychoric", weightVar = "totwgt")
```

```r
cbind(Correlation = c(Pearson = cor_pearson$correlation,
                    Spearman = cor_spearman$correlation,
                    Polychoric = cor_polychoric$correlation))
```

```
##             Correlation
## Pearson    -0.006546623
## Spearman   -0.005747158
## Polychoric -0.010325973
```

Plausible values for subject scales and subscales can also be correlated with contextual variables using the `cor.sdf` function. In this case, the five plausible values for `ssci`, the variable `asbs06a`, and the total student weight `totwgt` are read in to calculate the correlation coefficients using the Pearson, Spearman, and polyserial methods:

```r
cor_pearson2 <- cor.sdf(x = "ssci", y = "asbs06a", data = sdf_dnf,
                    method = "Pearson", weightVar = "totwgt")
cor_spearman2 <- cor.sdf(x = "ssci", y = "asbs06a", data = sdf_dnf,
                    method = "Spearman", weightVar = "totwgt")
cor_polyserial2 <- cor.sdf(x = "ssci", y = "asbs06a", data = sdf_dnf,
                    method = "Polyserial", weightVar = "totwgt")
```

```r
cbind(Correlation = c(Pearson = cor_pearson2$correlation,
                      Spearman = cor_spearman2$correlation,
                      Polyserial = cor_polyserial2$correlation))
```

```
##            Correlation
## Pearson     -0.1545121
## Spearman    -0.1361754
## Polyserial  -0.1693877
```

Unweighted Correlations

The `cor.sdf` function also features the ability to perform correlations without accounting for weights. The `cor.sdf` function automatically accounts for the default sample weights of the dataset in `weightVar = "default"`, but can be modified by setting `weightVar = NULL`. The following example shows the correlation coefficients of the Pearson and Spearman methods of the variables `asbg10a` and `asbs06a` while excluding weights:

```
cor_pearson_unweighted <- cor.sdf(x = "asbg10a", y = "asbs06a", data = TIMSS15,
                                  method = "Pearson", weightVar = NULL)
cor_pearson_unweighted
```

```
## Method: Pearson
## full data n: 12119
## n used: 9612
##
## Correlation: -0.009911209
##
## Correlation Levels:
##   Levels for Variable 'asbg10a' (Lowest level first):
##     1. EVERY DAY OR ALMOST EVERY DAY
##     2. ONCE OR TWICE A WEEK
##     3. ONCE OR TWICE A MONTH
##     4. NEVER OR ALMOST NEVER
##   Levels for Variable 'asbs06a' (Lowest level first):
##     1. AGREE A LOT
##     2. AGREE A LITTLE
##     3. DISAGREE A LITTLE
##     4. DISAGREE A LOT
```

```
cor_spearman_unweighted <- cor.sdf(x = "asbg10a", y = "asbs06a", data = TIMSS15,
                                   method = "Spearman", weightVar = NULL)
cor_spearman_unweighted
```

```
## Method: Spearman
## full data n: 12119
## n used: 9612
##
## Correlation: -0.002913911
##
## Correlation Levels:
##   Levels for Variable 'asbg10a' (Lowest level first):
##     1. EVERY DAY OR ALMOST EVERY DAY
##     2. ONCE OR TWICE A WEEK
##     3. ONCE OR TWICE A MONTH
##     4. NEVER OR ALMOST NEVER
##   Levels for Variable 'asbs06a' (Lowest level first):
##     1. AGREE A LOT
##     2. AGREE A LITTLE
##     3. DISAGREE A LITTLE
##     4. DISAGREE A LOT
```

Making a Table With `edsurveyTable`

Summary tables can be created in the `EdSurvey` package using the `edsurveyTable` function. A call to `edsurveyTable`[4] with five plausible values of the Science achievement scale (`ssci`) and two variables: "sex of students" (`itsex`) and "usually do well in science" (`asbs06a`), creates a table that shows the number, percentage, and TIMSS science achievement scale scores of fourth-grade students by gender and self-efficacy of science. Percentages add up to 100 within each gender.

By default, the `totwgt` weight variable and its associated replicate weights are selected for the analysis. In most of the `EdSurvey` analytical functions, users can define the weight variable through the argument `weightVar`. For instance, "`weightVar='tchwgt'`" means the analysis is at the teacher level. The program automatically applies the associated replicate weights to a weight variable.

```
es1 <- edsurveyTable(formula = ssci ~ itsex + asbs06a, data = TIMSS15,
                     jrrIMax = 1, varMethod = "jackknife")
```

This `edsurveyTable` is saved as the object `es1`, and the resulting table can be displayed by printing the object

```
es1
```

```
##
## Formula: ssci ~ itsex + asbs06a
##
## Plausible values: 5
## jrrIMax: 1
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## full data n: 12119
## n used: 9762
##
##
## Summary Table:
##   itsex           asbs06a    N      WTD_N       PCT   SE(PCT)     MEAN SE(MEAN)
##  FEMALE       AGREE A LOT 2759 1036100.69 55.655932 1.0011557 552.2677 2.857932
##  FEMALE    AGREE A LITTLE 1647  617484.88 33.169263 0.8744095 542.6399 2.432625
##  FEMALE DISAGREE A LITTLE  373  139961.25  7.518260 0.4468298 525.9240 5.481524
##  FEMALE    DISAGREE A LOT  189   68070.88  3.656545 0.2853314 491.3390 5.374267
##    MALE       AGREE A LOT 2927 1096249.78 61.382277 1.0212163 557.5826 2.831807
##    MALE    AGREE A LITTLE 1377  506094.91 28.337755 0.8665385 545.0932 2.496928
##    MALE DISAGREE A LITTLE  296  112855.44  6.319111 0.3780806 531.2213 5.904808
##    MALE    DISAGREE A LOT  194   70738.48  3.960857 0.2707493 494.6857 6.576870
```

Note that we used the argument `jrrIMax` to indicate the maximum number of plausible values to be included in the computation of the sampling portion of the standard error estimates. `jrrIMax` is also applicable to most of `EdSurvey` analytical functions including functions for means, percentiles, achievement levels/benchmarks, and regression coefficients.

The default estimation option, `jrrIMax = 1`, calculates the sampling variance using the first plausible value of a set with the jackknife variance estimation method, as seen in the next example. If jrrIMax is omitted, the default value of 1 will be selected. An alternative is to set `jrrIMax = Inf` to use all plausible values of a set. This setting leads to longer computing time but more accurate error estimates.

[4]Use `?edsurveyTable` for details on default `edsurveyTable` arguments.

The `full data n: 12119` represents the total sample size of the student- and teacher-level data. The final sample size of student-level data used for this analysis is indicated in `n used: 9762` (missing data were excluded by default).

The function also features variance estimation by setting the `varMethod` argument. As shown in the previous example, the default `varMethod = "jackknife"` indicates that the call used the jackknife method for variance estimation. By setting `varMethod = "Taylor"`, the same `edsurveyTable` call used in the previous example can return results using Taylor series variance estimation. Note users should use the Taylor series method with caution for international assessment data because most international assessments, including TIMSS, were not designed to be analyzed with this approach. This method is proper for NAEP data and some longitudinal datasets. See the help documentation for `lm.sdf` for details on the variance calculation.

If the percentages do not add to up to 100 at the desired level, an adjustment can be made in the `pctAggregationLevel` argument to change to the level needed to add up to 100. By default, `pctAggregationLevel = 1`, indicating that the PCT column in the output will be aggregated by each level of the first variable in the call; in our previous example this is `itsex`. Setting `pctAggregationLevel = 0` aggregates the PCT column in the output by each level of all variables in the call:

```
es2t <- edsurveyTable(formula = ssci ~ itsex + asbs06a, data = TIMSS15,
                      jrrIMax = 1, varMethod = "jackknife",
                      pctAggregationLevel = 0)
```

```
es2t$data
```

Table 2: es2t

itsex	asbs06a	N	WTD_N	PCT	SE(PCT)	MEAN	SE(MEAN)
FEMALE	AGREE A LOT	2759	1036100.69	28.405338	0.5735363	552.2677	2.857932
FEMALE	AGREE A LITTLE	1647	617484.88	16.928728	0.5043188	542.6399	2.432625
FEMALE	DISAGREE A LITTLE	373	139961.25	3.837124	0.2339475	525.9239	5.481524
FEMALE	DISAGREE A LOT	189	68070.88	1.866205	0.1438968	491.3390	5.374267
MALE	AGREE A LOT	2927	1096249.78	30.054362	0.6700557	557.5826	2.831807
MALE	AGREE A LITTLE	1377	506094.91	13.874903	0.4199227	545.0932	2.496928
MALE	DISAGREE A LITTLE	296	112855.44	3.094001	0.1877033	531.2213	5.904808
MALE	DISAGREE A LOT	194	70738.48	1.939339	0.1302186	494.6857	6.576870

The calculation of means and standard errors requires computation time that the user may not want to wait for. If you wish to simply see a table of the levels and the N sizes, you can set the `returnMeans` and `returnSepct` arguments to `FALSE` to omit those columns as follows:

```
es1b <- edsurveyTable(formula = ssci ~ itsex + asbs06a, data = TIMSS15, jrrIMax = Inf,
                      returnMeans = FALSE, returnSepct = FALSE)
```

In this `edsurveyTable`, the resulting table can be displayed by printing the object:

```
es1b
```

```
##
## Formula: ssci ~ itsex + asbs06a
##
## Plausible values: 5
## jrrIMax: 5
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## full data n: 12119
## n used: 9762
##
##
## Summary Table:
##   itsex           asbs06a    N      WTD_N       PCT
##  FEMALE       AGREE A LOT 2759 1036100.69 55.655932
##  FEMALE    AGREE A LITTLE 1647  617484.88 33.169263
##  FEMALE DISAGREE A LITTLE  373  139961.25  7.518260
##  FEMALE    DISAGREE A LOT  189   68070.88  3.656545
##    MALE       AGREE A LOT 2927 1096249.78 61.382277
##    MALE    AGREE A LITTLE 1377  506094.91 28.337755
##    MALE DISAGREE A LITTLE  296  112855.44  6.319111
##    MALE    DISAGREE A LOT  194   70738.48  3.960857
```

For more details on the arguments in the `edsurveyTable` function, look at the examples using `?edsurveyTable`.

Computing the Percentages of Students by Benchmarks With achievementLevels

The `achievementLevels` function[5] computes the percentages of students by benchmarks/achievement levels defined by an assessment. For TIMSS, each dataset's unique set of international benchmark cutpoints (defined as *Low*, *Intermediate*, *High*, and *Advanced*) is provided in the `EdSurvey` package. They can be accessed using the `showCutPoints` function:

```
showCutPoints(data = TIMSS15)
```

```
## Achievement Levels:
##   Low International Benchmark:  400
##   Intermediate International Benchmark:  475
##   High International Benchmark:  550
##   Advanced International Benchmark:  625
```

The `achievementLevels` function applies appropriate weights and the variance estimation method for each `edsurvey.data.frame`, with several arguments for customizing the aggregation and output of the analysis results. Namely, by using these optional arguments, users can

- choose to generate the percentage of students performing at each international benchmark (*discrete*) or at or above each benchmark (*cumulative*),
- calculate the percentage distribution of students by benchmark (*discrete* or *cumulative*) and selected characteristics (specified in `aggregateBy`), and
- compute the percentage distribution of students by selected characteristics within a specific benchmark.

The `achievementLevels` function can produce statistics by both discrete and cumulative benchmarks. By default, the `achievementLevels` function produces the results only by discrete benchmarks; when the

[5]Use `?achievementLevels` for details on default `achievementLevels` arguments.

`returnCumulative` argument is set to `TRUE`, the function generates results by both discrete and cumulative benchmarks.

To compute the overall results by benchmarks, use the TIMSS dataset's subject scale plausible values in the `achievementVars` argument. That is, `mmat` (the five plausible values for the mathematics scale), or `ssci` (the five plausible values for the science scale).

```
aLev1 <- achievementLevels(achievementVars = c("mmat"),
                           data = TIMSS15, returnCumulative = TRUE)
```

Here we are interested in just the discrete portion, which we can print in a neatly formatted table:

```
aLev1$discrete
```

Table 3: aLev1$discrete

Level	N	wtdN	Percent	StandardError
Below Low International Benchmark	550.6	197165.7	5.247535	0.4773946
At Low International Benchmark	1630.8	605075.8	16.103995	0.6380872
At Intermediate International Benchmark	3237.2	1202291.4	31.998795	0.7411637
At High International Benchmark	3206.8	1219063.8	32.445189	0.7584189
At Advanced International Benchmark	1403.6	533705.4	14.204486	0.7226353

In the next example, the plausible values for `mmat` are used to calculate the benchmarks, which are aggregated by the variable `itsex` using `aggregateBy`.

```
aLev2 <- achievementLevels(achievementVars = c("mmat", "itsex"),
                           aggregateBy = "itsex",
                           data = TIMSS15, returnCumulative = TRUE)
```

```
aLev2$discrete
```

Table 4: aLev2$discrete

Level	itsex	N	wtdN	Percent	StandardError
Below Low International Benchmark	FEMALE	277.2	99525.12	5.223096	0.5390420
At Low International Benchmark	FEMALE	872.0	323536.87	16.979273	0.7455260
At Intermediate International Benchmark	FEMALE	1694.2	627525.62	32.932657	1.0178026
At High International Benchmark	FEMALE	1604.0	614797.91	32.264705	0.9557248
At Advanced International Benchmark	FEMALE	623.6	240095.77	12.600269	0.7648260
Below Low International Benchmark	MALE	273.4	97640.61	5.272681	0.5452870
At Low International Benchmark	MALE	758.8	281538.91	15.203355	0.7431733
At Intermediate International Benchmark	MALE	1543.0	574765.82	31.037873	0.8795551
At High International Benchmark	MALE	1602.8	604265.87	32.630903	0.9219013
At Advanced International Benchmark	MALE	780.0	293609.67	15.855187	0.9572671

Note that each level of the `itsex` variable aggregates to 100 for the results by discrete achievement levels. The object `aLev2` created in this call to `achievementLevels` is a `list` with two `data.frames`: one for the discrete results and the other for the cumulative results. In the previously described code, only the discrete levels are shown using `aLev2$discrete`. To show the cumulative results, change the specified `data.frame`, as follows:

```
aLev2$cumulative
```

Table 5: aLev2$cumulative

Level	itsex	N	wtdN	Percent	StandardError
Below Low International Benchmark	FEMALE	277.2	99525.12	5.223096	0.5390420
At or Above Low International Benchmark	FEMALE	4793.8	1805956.17	94.776904	0.5390420
At or Above Intermediate International Benchmark	FEMALE	3921.8	1482419.30	77.797631	1.0389487
At or Above High International Benchmark	FEMALE	2227.6	854893.69	44.864974	1.2501151
At Advanced International Benchmark	FEMALE	623.6	240095.77	12.600269	0.7648260
Below Low International Benchmark	MALE	273.4	97640.61	5.272681	0.5452870
At or Above Low International Benchmark	MALE	4684.6	1754180.26	94.727319	0.5452870
At or Above Intermediate International Benchmark	MALE	3925.8	1472641.35	79.523963	1.0153240
At or Above High International Benchmark	MALE	2382.8	897875.53	48.486090	1.1988298
At Advanced International Benchmark	MALE	780.0	293609.67	15.855187	0.9572671

The `aggregateBy` argument sums the percentage of students by discrete achievement level up to 100 at the most disaggregated level specified by the analytical variables. For example, when `itsex` and `asbg05a` ("computer or tablet own at home") are used for analysis, each unique value pair of the two variables (i.e., MALE + YES, MALE + NO, FEMALE + YES, or FEMALE + NO) sums to 100.

```
achievementLevels(achievementVars = c("mmat", "itsex", "asbg05a"),
                  aggregateBy = c("itsex", "asbg05a"),
                  data = TIMSS15)
```

```
##
## AchievementVars: mmat, itsex, asbg05a
## aggregateBy: itsex, asbg05a
##
## Achievement Level Cutpoints:
## 400 475 550 625
##
## Plausible values: 5
## jrrIMax: 1
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## full data n: 12119
## n used: 9811
##
##
## Discrete
##                                   Level  itsex asbg05a      N      wtdN   Percent
##        Below Low International Benchmark FEMALE     YES  203.6  73042.80  5.054613
##           At Low International Benchmark FEMALE     YES  662.4 245583.77 16.994570
##  At Intermediate International Benchmark FEMALE     YES 1326.8 489013.76 33.840097
##          At High International Benchmark FEMALE     YES 1216.4 468540.04 32.423301
##      At Advanced International Benchmark FEMALE     YES  437.8 168891.63 11.687420
##        Below Low International Benchmark FEMALE      NO   58.0  20775.97  4.940626
##           At Low International Benchmark FEMALE      NO  185.8  68575.46 16.307575
##  At Intermediate International Benchmark FEMALE      NO  333.8 124564.87 29.622129
##          At High International Benchmark FEMALE      NO  370.0 138042.45 32.827162
##      At Advanced International Benchmark FEMALE      NO  180.4  68554.15 16.302508
##        Below Low International Benchmark   MALE     YES  194.4  69290.44  4.976729
##           At Low International Benchmark   MALE     YES  567.2 209483.47 15.045977
##  At Intermediate International Benchmark   MALE     YES 1190.4 444130.71 31.899322
##          At High International Benchmark   MALE     YES 1214.2 456863.75 32.813862
##      At Advanced International Benchmark   MALE     YES  567.8 212520.51 15.264110
##        Below Low International Benchmark   MALE      NO   61.0  21352.66  5.217336
##           At Low International Benchmark   MALE      NO  159.2  59524.67 14.544334
##  At Intermediate International Benchmark   MALE      NO  322.6 117179.30 28.631738
##          At High International Benchmark   MALE      NO  359.4 135219.67 33.039743
##      At Advanced International Benchmark   MALE      NO  199.8  75987.37 18.566850
##  StandardError
##      0.5353872
##      0.7593686
##      1.0216259
##      1.0258938
##      0.8293207
##      0.8412233
##      1.8032787
##      2.0383255
##      1.8848608
##      1.3453530
##      0.5725209
##      0.8273639
##      0.9741350
##      0.9446379
##      0.9559848
##      0.8190289
##      1.3570142
##      1.7523308
##      1.9087685
##      1.8225177
```

Note: When more than one contextual variable is used in the analysis, it is not appropriate to aggregate the results by only one contextual variable. The same variables used in the analysis must be used in the argument `aggregateBy()`, but their order does not impact the percentage results. For example, if one wants to know the distribution of achievement level percentages for each combination of "gender" and "computer or tablet own at home" (e.g., FEMALE + YES), both "gender" and "computer or tablet own at home" variable must be used in the argument `aggregateBy()`. If only one variable (such as gender) is used, the results reflect only the distribution of achievement level percentages for each level of gender.

The `achievementLevels` function can also compute the percentage of students by selected characteristics within a specific achievement level. The object `aLev3` presents the percentage of students by gender within each achievement level (i.e., within each discrete and cumulative level).

```r
aLev3 <- achievementLevels(achievementVars = c("mmat", "itsex"),
                           aggregateBy = "mmat",
                           data = TIMSS15, returnCumulative = TRUE)
aLev3$discrete
```

```
##                                        Level  itsex      N      wtdN  Percent
## 1        Below Low International Benchmark FEMALE  277.2  99525.12 50.47790
## 2        Below Low International Benchmark   MALE  273.4  97640.61 49.52210
## 3           At Low International Benchmark FEMALE  872.0 323536.87 53.47047
## 4           At Low International Benchmark   MALE  758.8 281538.91 46.52953
## 5  At Intermediate International Benchmark FEMALE 1694.2 627525.62 52.19414
## 6  At Intermediate International Benchmark   MALE 1543.0 574765.82 47.80586
## 7          At High International Benchmark FEMALE 1604.0 614797.91 50.43197
## 8          At High International Benchmark   MALE 1602.8 604265.87 49.56803
## 9      At Advanced International Benchmark FEMALE  623.6 240095.77 44.98657
## 10     At Advanced International Benchmark   MALE  780.0 293609.67 55.01343
##    StandardError
## 1       2.646888
## 2       2.646888
## 3       1.349679
## 4       1.349679
## 5       1.116163
## 6       1.116163
## 7       1.004937
## 8       1.004937
## 9       1.657732
## 10      1.657732
```

The percentage of students within a specific achievement level can be aggregated by one or more variables. For example, the percentage of students' ownership of a computer/tablet is aggregated by `itsex` within each achievement level:

```r
aLev4 <- achievementLevels(achievementVars = c("mmat", "itsex", "asbg05a"),
                           aggregateBy = c("itsex", "mmat"),
                           data = TIMSS15,
                           returnCumulative = TRUE)
aLev4$discrete
```

```
##                                      Level  itsex asbg05a      N      wtdN  Percent
## 1        Below Low International Benchmark FEMALE     YES  203.6  73042.80 77.85521
## 2        Below Low International Benchmark FEMALE      NO   58.0  20775.97 22.14479
## 3           At Low International Benchmark FEMALE     YES  662.4 245583.77 78.17175
## 4           At Low International Benchmark FEMALE      NO  185.8  68575.46 21.82825
## 5  At Intermediate International Benchmark FEMALE     YES 1326.8 489013.76 79.69863
## 6  At Intermediate International Benchmark FEMALE      NO  333.8 124564.87 20.30137
## 7          At High International Benchmark FEMALE     YES 1216.4 468540.04 77.24259
## 8          At High International Benchmark FEMALE      NO  370.0 138042.45 22.75741
## 9      At Advanced International Benchmark FEMALE     YES  437.8 168891.63 71.12850
## 10     At Advanced International Benchmark FEMALE      NO  180.4  68554.15 28.87150
## 11       Below Low International Benchmark   MALE     YES  194.4  69290.44 76.44315
## 12       Below Low International Benchmark   MALE      NO   61.0  21352.66 23.55685
## 13          At Low International Benchmark   MALE     YES  567.2 209483.47 77.87254
## 14          At Low International Benchmark   MALE      NO  159.2  59524.67 22.12746
## 15 At Intermediate International Benchmark   MALE     YES 1190.4 444130.71 79.12396
## 16 At Intermediate International Benchmark   MALE      NO  322.6 117179.30 20.87604
## 17         At High International Benchmark   MALE     YES 1214.2 456863.75 77.16206
## 18         At High International Benchmark   MALE      NO  359.4 135219.67 22.83794
## 19     At Advanced International Benchmark   MALE     YES  567.8 212520.51 73.66194
## 20     At Advanced International Benchmark   MALE      NO  199.8  75987.37 26.33806
##    StandardError
## 1       3.014389
## 2       3.014389
## 3       1.945471
## 4       1.945471
## 5       1.340267
## 6       1.340267
## 7       1.368631
## 8       1.368631
## 9       2.092690
## 10      2.092690
## 11      2.844876
## 12      2.844876
## 13      1.875236
## 14      1.875236
## 15      1.258082
## 16      1.258082
## 17      1.220979
## 18      1.220979
## 19      1.999234
## 20      1.999234
```

```
aLev4$cumulative
```

```
##                                              Level  itsex asbg05a      N       wtdN
## 1                   Below Low International Benchmark FEMALE     YES  203.6   73042.80
## 2                   Below Low International Benchmark FEMALE      NO   58.0   20775.97
## 3             At or Above Low International Benchmark FEMALE     YES 3643.4 1372029.20
## 4             At or Above Low International Benchmark FEMALE      NO 1070.0  399736.92
## 5  At or Above Intermediate International Benchmark FEMALE     YES 2981.0 1126445.43
## 6  At or Above Intermediate International Benchmark FEMALE      NO  884.2  331161.47
## 7            At or Above High International Benchmark FEMALE     YES 1654.2  637431.67
## 8            At or Above High International Benchmark FEMALE      NO  550.4  206596.60
## 9                 At Advanced International Benchmark FEMALE     YES  437.8  168891.63
## 10                At Advanced International Benchmark FEMALE      NO  180.4   68554.15
## 11                  Below Low International Benchmark   MALE     YES  194.4   69290.44
## 12                  Below Low International Benchmark   MALE      NO   61.0   21352.66
## 13            At or Above Low International Benchmark   MALE     YES 3539.6 1322998.43
## 14            At or Above Low International Benchmark   MALE      NO 1041.0  387911.01
## 15 At or Above Intermediate International Benchmark   MALE     YES 2972.4 1113514.97
## 16 At or Above Intermediate International Benchmark   MALE      NO  881.8  328386.34
## 17           At or Above High International Benchmark   MALE     YES 1782.0  669384.26
## 18           At or Above High International Benchmark   MALE      NO  559.2  211207.04
## 19                At Advanced International Benchmark   MALE     YES  567.8  212520.51
## 20                At Advanced International Benchmark   MALE      NO  199.8   75987.37
##     Percent StandardError
## 1  77.85521     3.0143888
## 2  22.14479     3.0143888
## 3  77.43851     0.7518633
## 4  22.56149     0.7518633
## 5  77.28047     0.8869351
## 6  22.71953     0.8869351
## 7  75.52255     1.1326017
## 8  24.47745     1.1326017
## 9  71.12850     2.0926905
## 10 28.87150     2.0926905
## 11 76.44315     2.8448758
## 12 23.55685     2.8448758
## 13 77.32720     0.7187741
## 14 22.67280     0.7187741
## 15 77.22546     0.7592508
## 16 22.77454     0.7592508
## 17 76.01532     0.9869434
## 18 23.98468     0.9869434
## 19 73.66194     1.9992339
## 20 26.33806     1.9992339
```

Finally, users can set unique cutpoints that override the standard values in the `EdSurvey` package using the `cutpoints` argument. In the example that follows, not using the standard cutpoints of `c(400,475,550,625)`, we set `cutpoints = c(405,475,550,625)`, resulting in a higher threshold to reach the *Basic* category but leaving *Proficient* and *Advanced* unchanged:

```
aLev5 <- achievementLevels(achievementVars = c("mmat", "itsex"),
                           aggregateBy = "itsex",
                           data = TIMSS15,
                           cutpoints = c(405,475,550,625),
                           returnCumulative = TRUE)

aLev5$discrete
```

```
##              Level  itsex      N     wtdN   Percent StandardError
## 1  Below Level 1 FEMALE  308.4 111138.5  5.832566     0.5756298
## 2     At Level 1 FEMALE  840.8 311923.5 16.369803     0.7350955
## 3     At Level 2 FEMALE 1694.2 627525.6 32.932657     1.0178026
## 4     At Level 3 FEMALE 1604.0 614797.9 32.264705     0.9557248
## 5     At Level 4 FEMALE  623.6 240095.8 12.600269     0.7648260
## 6  Below Level 1   MALE  302.2 108515.6  5.859940     0.5936202
## 7     At Level 1   MALE  730.0 270663.9 14.616097     0.6970205
## 8     At Level 2   MALE 1543.0 574765.8 31.037873     0.8795551
## 9     At Level 3   MALE 1602.8 604265.9 32.630903     0.9219013
## 10    At Level 4   MALE  780.0 293609.7 15.855187     0.9572671
```

```
aLev3$discrete
```

```
##                                          Level  itsex      N      wtdN  Percent
## 1        Below Low International Benchmark FEMALE  277.2  99525.12 50.47790
## 2        Below Low International Benchmark   MALE  273.4  97640.61 49.52210
## 3           At Low International Benchmark FEMALE  872.0 323536.87 53.47047
## 4           At Low International Benchmark   MALE  758.8 281538.91 46.52953
## 5  At Intermediate International Benchmark FEMALE 1694.2 627525.62 52.19414
## 6  At Intermediate International Benchmark   MALE 1543.0 574765.82 47.80586
## 7          At High International Benchmark FEMALE 1604.0 614797.91 50.43197
## 8          At High International Benchmark   MALE 1602.8 604265.87 49.56803
## 9      At Advanced International Benchmark FEMALE  623.6 240095.77 44.98657
## 10     At Advanced International Benchmark   MALE  780.0 293609.67 55.01343
##    StandardError
## 1       2.646888
## 2       2.646888
## 3       1.349679
## 4       1.349679
## 5       1.116163
## 6       1.116163
## 7       1.004937
## 8       1.004937
## 9       1.657732
## 10      1.657732
```

Changing the cutpoint for a particular benchmark will result in different distributions of student achievement. Notice that labels for the levels based on user-defined cutpoints are distinct from those based on TIMSS-defined cutpoints; instead, labels are based on the range of values in the `cutpoints` argument.

Calculating Percentiles With percentile

The `percentile` function calculates the percentiles of a numeric variable in the range 0 to 100 for a survey dataset. For example, to compare the TIMSS mathematics subject scale at the 10th, 25th, 50th, 75th, and 90th percentiles, include these as integers in the `percentiles` argument:

```
pct1 <- percentile(variable = "mmat", percentiles = c(10, 25, 50, 75, 90), data = TIMSS15)
pct1
```

```
## Percentile
## Call: percentile(variable = "mmat", percentiles = c(10, 25, 50, 75,
##     90), data = TIMSS15)
## full data n: 12119
## n used: 10029
##
##  percentile estimate       se       df confInt.ci_lower confInt.ci_upper nsmall
##          10 431.6736 3.478940 42.69991         422.3862         440.0570 1033.6
##          25 485.1651 2.119434 37.40158         477.6587         492.5979 2550.2
##          50 542.9994 2.154467 34.38474         536.5916         549.5875 4940.4
##          75 595.8614 2.494173 37.81299         589.6863         602.2926 2467.0
##          90 640.3610 2.238016 14.68479         633.1420         648.5989  984.2
```

Preparing an `edsurvey.data.frame.list`

Although most `EdSurvey` functions involve analyses of one dataset, an `edsurvey.data.frame.list` appends two or more `edsurvey.data.frame` objects into one list for analysis, which can be useful for comparisons across years (trend analysis) or jurisdictions. For example, TIMSS assessment datasets from different countries or years can be combined into an `edsurvey.data.frame.list` to make a single call to analysis functions for ease of use in comparing, formatting, and/or plotting output data.

To prepare an `edsurvey.data.frame.list` for analysis, it is necessary to ensure that variable information is consistent across each `edsurvey.data.frame`. When comparing groups across years, it is not uncommon that variable names and labels change across data years. Two useful functions in determining these inconsistencies are `searchSDF` and `levelsSDF`, which return variable names and labels based on a character string for a given `edsurvey.data.frame`.

Recoding Variable Names and Levels Using `recode.sdf` and `rename.sdf`

To assist in the process of standardizing data for `edsurvey.data.frames`, `light.edsurvey.data.frames`, and `edsurvey.data.frame.lists`, the functions `recode.sdf()` and `rename.sdf()` are particularly handy.

Similar to the `recode` argument from the `cor.sdf()` section earlier in this vignette (and featured in many other functions), `recode.sdf()` accepts the levels of a variable as a vector from their current values to their new recoded value. The following example collapses the lowest two levels of parent involvement `atbg06f` to `"LOW"` and the highest two levels to `"high"`. We then reassign the `edsurvey.data.frame` with revised variable information to a new object: "TIMSS15R".

```
TIMSS15R <- recode.sdf(TIMSS15,
                  recode=list(atbg06f=list(from=c("VERY LOW", "LOW"),
                                           to=c("LOW")),
                              atbg06f=list(from=c("VERY HIGH", "HIGH"),
                                           to=c("HIGH"))
                              )
                  )
searchSDF("atbg06f", TIMSS15R, levels = TRUE)
```

```
## Variable: atbg06f
## Label: GEN\CHARACTERIZE\PARENTAL INVOLVEMENT
## Levels (Lowest level first):
##      2. HIGH
##      3. MEDIUM
##      4. LOW
##      9. OMITTED OR INVALID
```

In addition, we can change the name of variables using `rename.sdf()`. The recoded variable `"atbg06f"` can be changed to a value that more effectively describes its contents, such as `"parentinvolvement"`:

```
TIMSS15R <- rename.sdf(TIMSS15R, "atbg06f", "parentinvolvement")
searchSDF("parentinvolvement", TIMSS15R, levels = TRUE)
```

```
## Variable: parentinvolvement
## Label: GEN\CHARACTERIZE\PARENTAL INVOLVEMENT
## Levels (Lowest level first):
##      2. HIGH
##      3. MEDIUM
##      4. LOW
##      9. OMITTED OR INVALID
```

NOTE: The functions `rename.sdf()` and `recode.sdf()` do not permanently overwrite the variable information from your data source; they recode it only for the current connection to the data in R. The original file formatting can be retrieved by calling the original object (e.g., "TIMSS15") or reconnecting to the data source via `readNAEP()`.

Combining Several `edsurvey.data.frame` Objects Into a Single Object

Once variables in each `edsurvey.data.frame` have been standardized, users can combine them into an `edsurvey.data.frame.list`. In the following example, three TIMSS 2015 datasets from the United States (usa), Finland (fin), and Hong Kong-China (hkg) for fourth-grade students were read in, appended into an `edsurvey.data.frame.list`, and then automatically assigned with unique labels.

```r
TIMSS15USA<- readTIMSS(path = "C:/TIMSS2015/", countries = c("usa"), gradeLvl = "4")
TIMSS15FIN<- readTIMSS(path = "C:/TIMSS2015/", countries = c("fin"), gradeLvl = "4")
TIMSS15HKG<- readTIMSS(path = "C:/TIMSS2015/", countries = c("hkg"), gradeLvl = "4")
```

```r
#Form an "edsurvey.data.frame.list"
TIMSS15C <- edsurvey.data.frame.list(list(TIMSS15USA, TIMSS15FIN, TIMSS15HKG))
```

Note that the previous example uses the same data connection (`TIMSS15`) in its formation of an `edsurvey.data.frame.list`; another method to create this data object is via `readTIMSS`, which automatically appends all jurisdictions specified in the `countries` argument into an `edsurvey.data.frame.list`:

```r
TIMSS15C <- readTIMSS("C:/TIMSS2015/", countries = c("usa", "fin", "hkg"), gradeLvl = "4")
```

This `edsurvey.data.frame.list` can now be analyzed with other EdSurvey analytical functions.

Recommended Workflow for Standardizing Variables in Analyses Across Years or Jurisdictions

Although the `EdSurvey` package features several methods to resolve inconsistencies across `edsurvey.data.frames`, the following approach is recommended:

1. Connect to each dataset using a read function such as `readNAEP()`.
2. Recode each discrepant variable name and level using `recode.sdf()` and `rename.sdf()`.
3. Combine datasets into one `edsurvey.data.frame.list` object.
4. Analyze trends using the `edsurvey.data.frame.list` object.

NOTE: It also is possible to retrieve and recode variables with the `getData` function; further details and examples of this method are discussed in the vignette titled *Using the `getData` Function in EdSurvey*.

Estimating the Difference in Two Statistics with Gap Analysis

Performing Gap Analysis and Understanding the Summary Output

The gap function in EdSurvey calculates the difference between two statistics for two groups in a population using the jackknife replication method. The function can perform the following comparisons with statistics as mean scores, achievement level percentages, percentiles, or student group percentages:

- Between groups by time point or jurisdiction
- Of the same group across time points or jurisdictions
- Between groups across time points or jurisdictions

Due to the scope of this vignette, we focus on gap analyses of jurisdictions. More details about conducting trend analysis across time points can be found in the vignette titled *Using EdSurvey for Trend Analysis*.

Please note that although it is typical and appropriate to test two statistics (e.g., two groups or two years) at a time, multiple comparison procedures should be applied when testing more than two groups to correct your results. Consult the vignette titled *Calculating Adjusted* p-*Values From EdSurvey Results* for the basics of adjusting p-values to account for multiple comparisons.

The following example demonstrates the `gap` function, comparing the difference between the `itsex` variables by year 2015:

```r
gapResult <- gap(variable = 'mmat', data = TIMSS15,
                 groupA=itsex %in% "FEMALE", groupB = itsex %in% "MALE")
gapResult
```

```
## Call: gap(variable = "mmat", data = TIMSS15, groupA = itsex %in% "FEMALE",
##     groupB = itsex %in% "MALE")
##
## Labels:
##  group          definition nFullData nUsed
##      A itsex %in% "FEMALE"     12119  5071
##      B   itsex %in% "MALE"     12119  4958
##
## Percentage:
##      pctA    pctAse     pctB    pctBse   diffAB      covAB diffABse diffABpValue
##  50.71408 0.5701627 49.28592 0.5701627 1.428164 -0.3250855 1.140325    0.2127084
##     dofAB
##  127.5956
##
## Results:
##  estimateA estimateAse estimateB estimateBse   diffAB    covAB diffABse diffABpValue
##   535.7482    2.312608  542.6617    2.501978 -6.91345 4.068738 1.862948  0.000319999
##     dofAB
##  114.6827
```

The gap output contains three blocks: labels, percentage, and results.

- The first block, labels, shows the definition of groups A and B, along with a re-minder of the full data n count (nFullData) and the n count of the number of individuals who are in the two subgroups with valid scores (nUsed).
- The second block, percentage, shows the percentage of individuals who fall into each category, with omitted levels removed.
- The third block, results, shows the estimated outcomes from the two groups listed in the columns estimateA and estimateB. The diffAB column shows the estimated difference between the two groups, and the diffABse column shows the standard error of the difference. t-test significance results show in difABpValue with the degrees of freedom following. When sampling survey respondents through cluster sampling designs (e.g., a design involving sampling students from the same classrooms or schools), these respondents have more in common than randomly selected individuals. Therefore, EdSurvey calculates a covariance between groups from the same assessment sample (same assessment and year), which appears in the covAB column. Even when selecting respondents through a simple random sampling, little harm occurs in estimating the covariance because it will be close to zero.

Users can choose to return only the data frame detailing the results using

```r
gapResult$results
```

```
##   estimateA estimateAse estimateB estimateBse   diffAB    covAB diffABse
## 1  535.7482    2.312608  542.6617    2.501978 -6.91345 4.068738 1.862948
##   diffABpValue    dofAB
## 1  0.000319999 114.6827
```

Gap Analysis of Achievement Levels and Percentiles

Gap analysis can be performed across achievement levels and percentiles by specifying the values in the `achievementLevel` or `percentiles` arguments, respectively. Using our previous `edsurvey.data.frame` object (`TIMSS15`), setting `achievementLevel=c("Low International Benchmark", "Intermediate International Benchmark", "High International Benchmark", "Advanced International Benchmark"))` will perform comparisons between groups for each achievement level value.

```
gapALResult <- gap(variable = "mmat", data = TIMSS15,
                   groupA=itsex%in%"FEMALE", groupB = itsex%in%"MALE",
                   achievementLevel = c("Low International Benchmark",
                                        "Intermediate International Benchmark",
                                        "High International Benchmark",
                                        "Advanced International Benchmark"))
gapALResult$results
```

```
##                                   achievementLevel estimateA estimateAse estimateB
## 1          At or Above Low International Benchmark  94.77690    0.539042  94.72732
## 2 At or Above Intermediate International Benchmark  77.79763    1.038949  79.52396
## 3         At or Above High International Benchmark  44.86497    1.250115  48.48609
## 4             At Advanced International Benchmark  12.60027    0.764826  15.85519
##   estimateBse       diffAB     covAB  diffABse diffABpValue     dofAB
## 1   0.5452870  0.04958509 0.1623821 0.5129716  0.923124124 149.27161
## 2   1.0153240 -1.72633244 0.6731292 0.8740930  0.049623634 203.01255
## 3   1.1988298 -3.62111596 0.7929669 1.1891370  0.002813883 130.52596
## 4   0.9572671 -3.25491808 0.2810054 0.9691792  0.001222924  76.69773
```

Similarly, setting `percentiles = c(10, 25, 50, 75, 90)` will perform comparisons between groups for each percentile value.

```
gapPCTResult <- gap(variable = "mmat", data = TIMSS15,
                   groupA=itsex%in%"FEMALE", groupB = itsex%in%"MALE",
                   percentiles = c(10, 25, 50, 75, 90))
gapPCTResult$results
```

```
##   percentiles estimateA estimateAse estimateB estimateBse     diffAB    covAB
## 1          10  430.7472    2.548751  433.0292    4.288548  -2.281968 5.539868
## 2          25  482.8148    3.273067  487.8001    3.749624  -4.985253 6.810713
## 3          50  539.5734    4.495574  546.7166    2.533361  -7.143178 5.616811
## 4          75  591.5397    2.458030  600.2116    3.541086  -8.671900 3.834350
## 5          90  634.9166    3.001938  646.1702    3.911206 -11.253630 4.230028
##   diffABse diffABpValue     dofAB
## 1 3.715916  0.540805540  83.95686
## 2 3.339345  0.139226859  83.71839
## 3 3.923580  0.071097611 123.23413
## 4 3.303408  0.009600676 143.38729
## 5 3.981094  0.006348545  61.04594
```

Gap Analysis of Jurisdictions

Comparisons of district, state, and national jurisdictions can be performed using the `gap` function. The following code compares U.S. students with Finland and Hong Kong students in their science achievement using the `edsurvey.data.frame.list` `TIMSS15C`, which combined the TIMSS 2015 fourth-grade data from the United States, Finland, and Hong Kong-China:

```
TIMSSCGap <- gap(variable = "ssci", data = TIMSS15C)
TIMSSCGap$results
```

```
##         country estimateA estimateAse    diffAA covAA diffAAse diffAApValue    dofAA
## 1 United States  545.9069    2.261519        NA    NA       NA           NA       NA
## 2       Finland  553.8131    2.268841  -7.90625     0 3.203452  0.014881281 130.3391
## 3 Hong Kong SAR  556.5472    2.955640 -10.64028     0 3.721595  0.004774799 171.9679
##   sameSurvey
## 1         NA
## 2      FALSE
## 3      FALSE
```

The gap in average science scores between the United Sates and Finland (`diffAA`) is `-7.90625`, with a standard error of (`diffAAse`) `3.203452`. The degrees of freedom (`dofAA`) are `130.3391`, and the *p*-value (`diffAApValue`) is `0.014881281`.

By default, the `gap` function treats the first data in an `edsurvey.data.frame.list` as the reference data. We can use the `referenceDataIndex` argument to change the reference to another data. For example, set `referenceDataIndex` argument = 2 to make the second data as the reference data.

```
TIMSSCGap2 <- gap(variable = 'ssci', data = TIMSS15C, referenceDataIndex = 2)
TIMSSCGap2$results
```

```
##         country estimateA estimateAse    diffAA covAA diffAAse diffAApValue    dofAA
## 1 United States  545.9069    2.261519  7.906250     0 3.203452   0.01488128 130.3391
## 2       Finland  553.8131    2.268841        NA    NA       NA           NA       NA
## 3 Hong Kong SAR  556.5472    2.955640 -2.734028     0 3.726050   0.46441390 130.2009
##   sameSurvey
## 1      FALSE
## 2         NA
## 3      FALSE
```

Comparison of Two Student Groups Between Jurisdictions

The following example compares the gender gap of U.S. fourth-grade students with fourth-grade students from Finland and Hong Kong, respectively.

```
TIMSSCGap3 <- gap(variable = 'ssci', data = TIMSS15C,
                  groupA = itsex %in% "FEMALE",
                  groupB = itsex %in% "MALE")
```

To look only at the results of the gender gap between jurisdictions,

```
TIMSSCGap3$results[,c("country", "diffAB", "diffABAB", "diffABABse",
                      "dofABAB", "diffABABpValue")]
```

```
##         country    diffAB  diffABAB diffABABse   dofABAB diffABABpValue
## 1 United States -4.007687        NA         NA        NA             NA
## 2       Finland 11.827152 -15.83484   3.033440 151.01164   5.823087e-07
## 3 Hong Kong SAR -9.655327   5.64764   4.293422  47.25219   1.947171e-01
```

The student gender gap in average science scores between the United Sates and Finland (`diffABAB`) is `-15.83484` with a standard error (`diffABABse`) of `3.033440`. The degrees of freedom (`dofABAB`) are `151.01164`, and the *p*-value (`diffABABpValue`) is `5.823087e-07`.

Regression Analysis with lm.sdf

After the data are read in with the `EdSurvey` package, a linear model can be fit to fully account for the complex sample design used for the TIMSS data by using `lm.sdf`.

```
lm1 <- lm.sdf(formula = ssci ~ itsex + asbs06a, data = TIMSS15)
summary(lm1)
```

```
##
## Formula: ssci ~ itsex + asbs06a
##
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## Plausible values: 5
## jrrIMax: 1
## full data n: 12119
## n used: 9762
##
## Coefficients:
##                                coef       se         t    dof  Pr(>|t|)
## (Intercept)                552.7576   2.7903 198.0987 111.43 < 2.2e-16 ***
## itsexMALE                    4.3619   1.8570   2.3489 136.00   0.02027 *
## asbs06aAGREE A LITTLE      -10.9774   2.2953  -4.7826 129.29 4.639e-06 ***
## asbs06aDISAGREE A LITTLE -26.4161   3.8750  -6.8170 130.86 3.086e-10 ***
## asbs06aDISAGREE A LOT      -61.9360   4.5770 -13.5320 116.56 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared: 0.0285
```

After the regression is run, the data are automatically removed from memory. The data are read in and analyzed by the `lm.sdf` function—in this case, `itsex`, `asbs06a`, the five plausible values for `ssci`.

Note that the option `jrrIMax` is omitted in the following example; therefore, the default jackknife variance estimator is used.[6] Also, note that an explicit weight variable is not set, so the `lm.sdf` function uses the weight that it deems appropriate based on the variables specified.

By default, `lm.sdf` uses "treatment contrasts," where one level is dropped from the regression. This cannot be changed, but the omitted and comparison group can be changed with the `relevels` argument. In the following example, "MALE" is omitted from the analysis for the variable `dsex`:

```
lm1m <- lm.sdf(formula = ssci ~ itsex + asbs06a, data = TIMSS15,
               relevels = list(itsex = "MALE"))
summary(lm1m)
```

[6]Use `?lm.sdf` for details on default `lm.sdf` arguments.

```
##
## Formula: ssci ~ itsex + asbs06a
##
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## Plausible values: 5
## jrrIMax: 1
## full data n: 12119
## n used: 9762
##
## Coefficients:
##                              coef       se        t    dof  Pr(>|t|)
## (Intercept)              557.1195   2.6722 208.4860 120.04 < 2.2e-16 ***
## itsexFEMALE               -4.3619   1.8570  -2.3489 136.00   0.02027 *
## asbs06aAGREE A LITTLE    -10.9774   2.2953  -4.7826 129.29 4.639e-06 ***
## asbs06aDISAGREE A LITTLE -26.4161   3.8750  -6.8170 130.86 3.086e-10 ***
## asbs06aDISAGREE A LOT    -61.9360   4.5770 -13.5320 116.56 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Multiple R-squared: 0.0285
```

Note that the coefficient on `itsex` changed from positive in the previous run to negative of the exact same magnitude, whereas none of the other coefficients (aside from the intercept) changed — the expected result. The change results from the switch of the reference gender from “FEMALE” in the first regression model to “MALE” in the second regression model.

Multivariate Regression With `mvrlm.sdf`

A multivariate regression model can be fit to fully account for the complex sample design used for NCES data by using `mvrlm.sdf`. This function implements an estimator that correctly handles multiple dependent variables that are numeric (such as plausible values), which allows for variance estimation using the jackknife replication method.

The vertical line symbol `|` separates dependent variables on the left-hand side of formula. In the following example, a multivariate regression is fit with two subject scales as the outcome variables (`mmat` and `ssci`) by two predictor variables signifying gender and parent involvement (`itsex` and `atbg06f`):

```r
mvrlm1 <- mvrlm.sdf(mmat | ssci ~ itsex + atbg06f, data = TIMSS15)
summary(mvrlm1)
```

```
##
## Formula: mmat | ssci ~ itsex + atbg06f
##
## jrrIMax:
## Weight variable: 'totwgt'
## Variance method:
## JK replicates: 150
## full data n: 12119
## n used: 10570
##
```

```
## Coefficients:
##
## mmat
##                     coef       se        t    dof  Pr(>|t|)
## (Intercept)     572.2616   9.0737  63.0684 13.822 < 2.2e-16 ***
## itsexMALE         7.6224   2.0461   3.7253 70.216 0.0003908 ***
## atbg06fHIGH     -15.3979  10.4394  -1.4750 23.947 0.1532476
## atbg06fMEDIUM   -43.9984  10.1167  -4.3491 16.032 0.0004950 ***
## atbg06fLOW      -63.7597  11.2764  -5.6543 24.897 7.034e-06 ***
## atbg06fVERY LOW -86.0765  10.0850  -8.5351 17.127 1.412e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ssci
##                     coef       se        t    dof  Pr(>|t|)
## (Intercept)     579.9583   7.8615  73.7721 17.512 < 2.2e-16 ***
## itsexMALE         4.0814   2.2291   1.8310 70.900    0.0713 .
## atbg06fHIGH     -13.8273   9.0967  -1.5200 32.693    0.1381
## atbg06fMEDIUM   -42.4909   8.5172  -4.9888 23.235 4.665e-05 ***
## atbg06fLOW      -65.0156   9.9487  -6.5351 34.294 1.685e-07 ***
## atbg06fVERY LOW -82.8187   9.6430  -8.5885 30.081 1.365e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual correlation matrix:
##
##      mmat ssci
## mmat 1.00 0.83
## ssci 0.83 1.00
##
## Multiple R-squared by dependent variable:
##
##   mmat   ssci
## 0.0988 0.1012
```

The `mvrlm.sdf` documentation provides examples to compare the regression outputs. See `?mvrlm.sdf` for an overview of additional details that can be accessed through components of the returned object. In addition, the vignette titled *Statistical Methods Used in EdSurvey* goes into further detail by describing estimation of the reported statistics.

Logistic Regression With logit.sdf

The `EdSurvey` package also features generalized linear models—such as logit and probit—that account for the large-scale complex sampling design. The models allow dichotomized variables, including plausible values, to be the outcome variables. EdSurvey offers three binomial regression functions, namely `glm.sdf`, `logit.sdf`, and `probit.sdf`, for complex sampling design data with plausible values. `glm.sdf` is an umbrella function that currently fits logit and probit models. Alternatively, users can choose `logit.sdf` or `probit.sdf` functions for binomial outcomes.

The following example demonstrates how to use `logit.sdf` to predict students' school belongingness with variables `asbg04` ("amount of books in home") and `asbg12a` ("how often being made fun of"). The dependent variable `asbg11c` ("Agree belong at school") is recoded from `"AGREE A LOT"` and `"AGREE A LITTLE"` to

"AGREE"; and from "DISAGREE A LITTLE" and "DISAGREE A LOT" to "DISAGREE". I(asbg11c=="AGREE") is used to specify the outcome level of the regression.

```
logit1 <- logit.sdf(I(asbg11c=="AGREE") ~ asbg04 + asbg12a, data = TIMSS15,
                    recode=list(asbg11c=list(from=c("AGREE A LOT",
                                                    "AGREE A LITTLE"),
                                             to=c("AGREE")),
                                asbg11c=list(from=c("DISAGREE A LITTLE",
                                                    "DISAGREE A LOT"),
                                             to=c("DISAGREE"))))
summary(logit1)
```

```
##
## Formula: asbg11c ~ asbg04 + asbg12a
## Family: binomial (logit)
##
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## full data n: 12119
## n used: 9445
##
## Coefficients:
##                                  coef       se         t     dof  Pr(>|t|)
## (Intercept)                  0.514522  0.095187  5.405407 146.725 2.560e-07 ***
## asbg0411-25 BOOKS            0.475385  0.088771  5.355185 102.424 5.268e-07 ***
## asbg0426-100 BOOKS           0.531433  0.081191  6.545455  95.956 2.911e-09 ***
## asbg04101-200 BOOKS          0.686595  0.107625  6.379488 133.399 2.699e-09 ***
## asbg04MORE THAN 200          0.400948  0.110178  3.639076 133.595 0.0003899 ***
## asbg12aONCE OR TWICE A MONTH 0.397545  0.118141  3.365005 156.071 0.0009634 ***
## asbg12aA FEW TIMES A YEAR    0.699042  0.110698  6.314866  88.133 1.066e-08 ***
## asbg12aNEVER                 1.022492  0.085854 11.909643 101.050 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notice that testing asbg11c=="AGREE" ensures that the sign of the coefficients indicates increased agreement with “Agree belong at school.” Also, note that these coefficients are in unmodified logistic units and are not linearized.

Most components of logit.sdf also apply to probit.sdf and glm.sdf. For more information about all EdSurvey generalized linear models, please refer to the R help file.

oddsRatio

The oddsRatio helper function allows for the conversion of coefficients from an EdSurvey logit regression model to odds ratios. Odds ratios are useful for understanding the real likelihood of an event occurring based on a transformation to the log odds returned in a logistic model.

In EdSurvey, odds ratios can be returned by specifying the logistic model object (e.g., logit1)

```
oddsRatio(logit1)
```

```
##                                    OR      2.5%    97.5%
## (Intercept)                  1.672839 1.360745 1.984934
## asbg0411-25 BOOKS            1.608634 1.328746 1.888522
## asbg0426-100 BOOKS           1.701368 1.430622 1.972114
## asbg04101-200 BOOKS          1.986938 1.567802 2.406074
## asbg04MORE THAN 200          1.493239 1.170775 1.815704
## asbg12aONCE OR TWICE A MONTH 1.488167 1.143572 1.832761
## asbg12aA FEW TIMES A YEAR    2.011824 1.575323 2.448325
## asbg12aNEVER                 2.780113 2.312292 3.247934
```

waldTest

The `waldTest` function allows the user to test composite hypotheses—hypotheses with multiple coefficients and the intercept involved—even when the data include plausible values. Because there is no likelihood test for plausible values or residuals, the Wald test fills the role of the likelihood ratio test, ANOVA, and F-test.

Wald tests can be run by specifying the model and coefficients. In the following two examples, the first one tests whether the coefficient for level "11-25" of `asbg04` is equal to zero, whereas the second jointly tests all the coefficients including the intercept corresponding to the levels of two predictors `asbg04` and `asbg12a`.

```
#Test the 2nd coefficient: level "11-25 BOOKS".
#Note the 1st coefficient is the intercept of the model in the "logit1" object.
waldTest(model = logit1, coefficients = 2)
```

```
## Wald test:
## ----------
## H0:
## asbg0411-25 BOOKS = 0
##
## Chi-square test:
## X2 = 28.7, df = 1, P(> X2) = 8.5e-08
##
## F test:
## W = 28.7, df1 = 1, df2 = 75, P(> W) = 9e-07
```

```
#Test all coefficients of the model excluding the intercept.
waldTest(model = logit1, coefficients = 2:8)
```

```
## Wald test:
## ----------
## H0:
## asbg0411-25 BOOKS = 0
## asbg0426-100 BOOKS = 0
## asbg04101-200 BOOKS = 0
## asbg04MORE THAN 200 = 0
## asbg12aONCE OR TWICE A MONTH = 0
## asbg12aA FEW TIMES A YEAR = 0
## asbg12aNEVER = 0
##
## Chi-square test:
## X2 = 249.7, df = 7, P(> X2) = 0.0
##
## F test:
## W = 32.8, df1 = 7, df2 = 69, P(> W) = 0
```

To learn more about conducting Wald tests, consult the vignette titled *Methods and Overview of Using EdSurvey for Running Wald Tests* at the AIR website.

Quantile Regression Analysis with `rq.sdf`

The `rq.sdf` function computes an estimate on the tau-th conditional quantile function of the response, given the covariates, as specified by the formula argument. Similar to `lm.sdf`, the function presumes a linear specification for the quantile regression model (i.e., the formula defines a model that is linear in parameters). Note that Jackknife is the only applicable variance estimation method used by the function.

To conduct quantile regression at a given tau value (by default, tau is set as 0.5), specify using the `tau` argument (in this example `tau = 0.8`); all other arguments are otherwise consistent with `lm.sdf`, except for `returnVarEstInputs`, `returnNumberOfPSU`, and `standardizeWithSamplingVar`, which are not available.

```r
rq1 <- rq.sdf(ssci ~ itsex + asbs06a, data=TIMSS15, tau = 0.8)
summary(rq1)
```

```
##
## Formula: ssci ~ itsex + asbs06a
##
## tau: 0.8
## jrrIMax: 1
## Weight variable: 'totwgt'
## Variance method: jackknife
## JK replicates: 150
## full data n: 12119
## n used: 9762
##
## Coefficients:
##                             coef       se        t     dof  Pr(>|t|)
## (Intercept)             619.9363   3.3929 182.7144  85.506 < 2.2e-16 ***
## itsexMALE                 5.8125   3.1849   1.8250  95.759 0.0711133 .
## asbs06aAGREE A LITTLE   -15.9569   4.6860  -3.4053 126.335 0.0008866 ***
## asbs06aDISAGREE A LITTLE -31.8919   4.9019  -6.5060 103.352 2.809e-09 ***
## asbs06aDISAGREE A LOT   -60.2836   6.8520  -8.7980  46.328 1.908e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For further details on quantile regression models and how they are implemented in R, see the vignette from the `quantreg` package (accessible by `vignette("rq", package="quantreg")`), on which the `rq.sdf` function is built.

Mixed Models With `mixed.sdf`

The `EdSurvey` package features the functionality of estimating mixed-effects models accounting for plausible values and survey weights. The `EdSurvey` package fits a weighted mixed model, also known as a weighted multilevel or hierarchical linear model using the `WeMix` package.

The following two examples illustrate how to model the random intercept of mathematics achievement at the school level with students' gender as a covariate.

```
#Use all plausibel values

mix1 <- mixed.sdf(mmat ~ itsex + (1|idschool), data = TIMSS15USA,
                  weightVar=c("totwgt","schwgt"), weightTransformation=FALSE)
summary(mix1)
```

```
## Call:
## mixed.sdf(formula = mmat ~ itsex + (1 | idschool), data = TIMSS15USA,
##     weightVars = c("totwgt", "schwgt"), weightTransformation = FALSE)
##
## Formula: mmat ~ itsex + (1 | idschool)
##
## Plausible Values: 5
## Number of Groups:
##   Group Var Observations Level
## 1  idschool          250     2
## 2  Residual        10029     1
##
## Variance terms:
##                         variance Std. Error Std.Dev.
## idschool.(Intercept) 1758.268   184.4192 41.93171
## Residual             4671.666   105.1603 68.34959
##
## Fixed Effects:
##             Estimate Std. Error  t value
## (Intercept) 536.0425     3.7673 142.2880
## itsexMALE     6.0908     1.5000   4.0606
##
## Intraclass Correlation= 0.273
```

```
# uses only one plausible value
mix2 <- mixed.sdf(asmmat01 ~ itsex + (1|idschool), data = TIMSS15USA,
                  weightVar=c("totwgt","schwgt"), weightTransformation=FALSE)
summary(mix2)
```

```
## Call:
## mixed.sdf(formula = asmmat01 ~ itsex + (1 | idschool), data = TIMSS15USA,
##     weightVars = c("totwgt", "schwgt"), weightTransformation = FALSE)
##
## Formula: asmmat01 ~ itsex + (1 | idschool)
## Number of Groups:
##   Group Var Observations Level
## 1  idschool          250     2
## 2  Residual        10029     1
##
## Variance terms:
##                         variance Std. Error Std.Dev.
## idschool.(Intercept) 1713.408   173.7081 41.39334
## Residual             4657.934   105.2967 68.24906
##
## Fixed Effects:
##             Estimate Std. Error  t value
## (Intercept) 536.8467     3.6683 146.3488
## itsexMALE     5.7265     1.4416   3.9723
##
## lnl=-21290221.51
## Intraclass Correlation= 0.269
```

For further guidance and use cases for mixed-effects models in `EdSurvey`, see the vignette titled *Methods Used for Estimating Mixed-Effects Models in EdSurvey*. For examples of how NCES recommends using weighted mixed-effects models, as well as their summary of the mathematical background and description of hierarchical linear model's insufficiency in this case, see Appendix D in the NCES working paper on analysis of TIMSS data at *Using TIMSS to Analyze Correlates of Performance Variation in Mathematics*.

Endnotes

Memory Usage

Because many NCES databases have hundreds of columns and hundreds of thousand rows, the `EdSurvey` package allows users to subset data and run regressions without storing it in the global environment. Alternatively, the `getData` function retrieves `light.edsurvey.data.frames` into the global environment, which can be costly to memory usage.

This package uses the `LaF` package to read in only the necessary data when needed for an analysis. Instead of storing all the data in memory, only some "header" information is stored as well as a link to the file in question. When the user calls a function, only the data needed for that function are read in. It works seamlessly and reduces the memory requirements for a user's machine.

Factors and Factor Analysis

R uses the concept of *factors* for data storage. This is a separate concept from factor analysis. In the case of the R storage method, it is simply a way of enforcing that only valid data labels are used.

Summary and Next Steps

This vignette covered the basics of the `EdSurvey` package, such as setting up the R environment for analysis, creating summary tables with `edsurveyTable`, running linear regression models with `lm.sdf`, correlating variables with `cor.sdf`, and retrieving data for manipulation with the `getData` function. Other aspects of the package relating to memory usage also were considered.

For a full list of `EdSurvey` functions and documentation, use the R help viewer:

```r
help(package = "EdSurvey")
```

Appendix A.

TIMSS 2015 Overall Scales and Subscales of Plausible Values

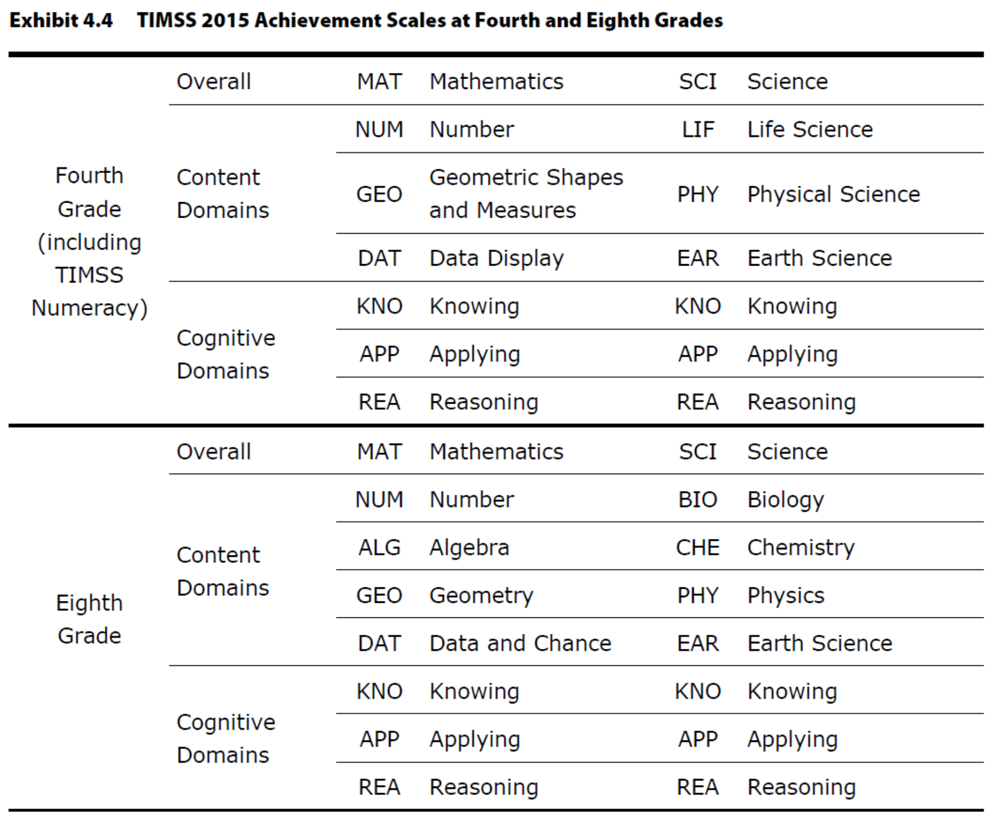


Exhibit 4.4 TIMSS 2015 Achievement Scales at Fourth and Eighth Grades

Fourth Grade (including TIMSS Numeracy)	Overall	MAT	Mathematics	SCI	Science
	Content Domains	NUM	Number	LIF	Life Science
		GEO	Geometric Shapes and Measures	PHY	Physical Science
		DAT	Data Display	EAR	Earth Science
	Cognitive Domains	KNO	Knowing	KNO	Knowing
		APP	Applying	APP	Applying
		REA	Reasoning	REA	Reasoning
Eighth Grade	Overall	MAT	Mathematics	SCI	Science
	Content Domains	NUM	Number	BIO	Biology
		ALG	Algebra	CHE	Chemistry
		GEO	Geometry	PHY	Physics
		DAT	Data and Chance	EAR	Earth Science
	Cognitive Domains	KNO	Knowing	KNO	Knowing
		APP	Applying	APP	Applying
		REA	Reasoning	REA	Reasoning

Source: *TIMSS 2015 User Guide for the International Database*, p. 56.

Appendix B.

How to Decide Which Weights to Use in TIMSS

Users should carefully consider the purpose of the analysis and the research question when selecting the sampling weights to be used. Generally speaking,

1. For analysis of **student background and achievement** data, total student weight (`totwgt`) is appropriate for single-level student-level analyses in survey software that adjusts standard errors to reflect the study design. The `totwgt` is the product of the final weighting components for schools, classrooms, and students, including the nonparticipation adjustments. Although `totwgt` has desirable properties, it may have drawbacks for some analyses; thus IEA also provides student house weight (`houwgt`) and student senate weight (`senwgt`), which are linear transformations of `totwgt` that result in some number with desirable properties. For further explanation of these student weights, see page 64 in the *TIMSS 2015 User Guide for the International Database*.

2. For analysis of **teacher background** data: The teachers in the TIMSS International Database do not constitute representative samples of teachers in the participating countries. Rather, they are the teachers of nationally representative samples of students. In other words, we can generalize the

estimates only to students, not their teachers. Therefore, analyses with teacher data should be made with students as the units of analysis and reported in terms of students who are taught by teachers with a particular attribute.

Teacher data are analyzed by linking the students to their teachers. The student-teacher linkage data files are used for this purpose, and the `EdSurvey` package makes use of them automatically.

The three teacher weight variables (overall teacher weight—`tchwgt`—and subject-wise teacher weights—`matwgt` and `sciwgt`) are specifically designed for using teacher background data in student-level analyses and are based on `totwgt`. Although `tchwgt` is used for analyses using all teachers, `matwgt` and `sciwgt` are used for analyses of mathematics and science teachers, respectively.

3. For analysis with **school background** data: Because TIMSS has representative samples of schools, it is possible to compute reasonable statistics with schools as units of analysis using the school weight (`schwgt`). However, the school samples were designed to optimize the student samples and the student-level results. For this reason, it is preferable to analyze school-level variables as attributes of the students, rather than as elements in their own right. Therefore, analyzing school data should be done by merging the students to their schools and then conducting a student-level analysis using `totwgt`. The `EdSurvey` package automatically takes care of the merging.