Furniture for Quantitative Scientists

by Tyson S. Barrett and Emily Brignone

Abstract A basic understanding of the distributions of study variables and the relationships among them is essential to inform statistical modeling. This understanding is achieved through the computation of summary statistics and exploratory data analysis. Unfortunately, this step tends to be under-emphasized in the research process, in part because of the often tedious nature of thorough exploratory data analysis. The `table1()` function in the **furniture** package streamlines much of the exploratory data analysis process, making the computation and communication of summary statistics simple and beautiful while offering significant time-savings to the researcher.

Introduction

A major, but often overlooked, aspect of the research process is the computation of summary statistics and exploratory data analysis. Through exploratory data analysis, researchers can begin to understand patterns and relationships in their data that are relevant for more complex modeling schemes. In addition, this exploration can inform future directions in their research area and can inspire better research questions. This is a valuable benefit of assessing summary statistics and exploring the data, for, as John Tukey (1980) stated, "Finding the question is often more important than finding the answer."

Generally, an important aspect of data exploration is the assessment of summary statistics. Yet, the computation and reporting of summary statistics can be tedious and messy, with many researchers computing and entering values into tables one variable at a time. This can be quite time-consuming and leaves the researcher susceptible to data-entry errors. Additionally, in the event of changes to the number of observations in the sample or to the nature of the stratifying variable, the whole table must be manually updated. These issues may lead researchers to conduct only minimal exploratory data analysis, potentially missing important relationships among study variables. Researchers may also opt to delegate the computation of summary statistics to research assistants. This is not always ideal, as assistants may lack the substantive expertise required to recognize important or theoretically interesting patterns in the data.

The presentation of sample summary statistics in conjunction with higher-level data analysis is also essential to the production of reproducible research.[1] In many fields there is a struggle to produce reproducible research (Open Science Collaboration, 2015; Chang and Li, 2015; Begley and Ioannidis, 2015). Thorough reporting of sample characteristics, including the distribution of sample characteristics stratified by key study variables, allows other researchers the opportunity to more carefully evaluate findings, replicate results, discover further research avenues, and critically compare findings across multiple studies. In the **furniture** package, as will be demonstrated, obtaining descriptive statistics for all study variables, either with or without a stratifying variable, is simple and can be done using just a few lines of code.

It is worth noting that there are other well-designed packages that help produce descriptive statistics. The package **tableone** is thoroughly developed and performs similar analyses to `table1()`. However, the syntax is more complex and lacks some of the flexibility that `table1()` offers. The **stargazer** and **psych** packages also offer informative summaries of the data for each variable, but are limited in their use for publication without manual entry, and offer few options for more involved data manipulation.

Data for example analyses

In order to demonstrate the utility of `table1()`, we descriptively explored and analyzed data from the National Health and Nutrition Examination Survey (NHANES) provided by the Centers for Disease Control and Prevention (National Center for Health Statistics, 2016). The data are attached in the package as "nhanes_2010" after having been cleaned using the **furniture**, **tidyverse** and **foreign** R packages (Barrett and Brignone, 2016; Wickham, 2016; R Core Team, 2016). The data contain information on general health, activity level, age, gender, drug use, and chronic conditions for 1,417 young adults (51.4% female) aged 18-30 (M = 23.3, SD = 3.96). Under the direction of the Centers for Disease Control and Prevention, the data were collected via a complex sampling strategy across the United States in 2013 and 2014.

[1] We use the definitions from Goodman et al. (2016): "methods reproducibility" refers to the ability to obtain the same results with the same methods and data and "results reproducibility" means being able to obtain similar results using the same methods with independent data.

Table 1

The general form of `table1()`[2] is below, with several of the most common arguments. Note that the structure is similar to other "tidy" packages, with the `data.frame` as the first argument, unquoted variables, and the ability to pipe.

```
library(furniture)
table1(df,
       var1, var2, var3, etc.,
       splitby = ~stratifying_variable,
       test = TRUE,
       type = "pvalues",
       second = NULL,
       output = "text",
       FUN = NULL,
       FUN2 = NULL)
```

where

- `df` is the `data.frame` or a `tbl_df` from the tidyverse of functions,
- `var1`, etc. are unquoted variable names (any number of variables),
- `splitby` is a one-sided formula with a variable name,
- `test` is logical (i.e. `TRUE` or `FALSE`) indicating whether bivariate tests of association should be run,
- `type` allows the p-values and/or test statistics to be displayed, along with allowing a simplification of the output of the table,
- `second` is an optional vector of quoted variable names that, instead of means and SDs, are summarized by medians and the inter-quartile range (or other user-defined statistics),
- `output` allows several outputs for the table, including Latex and Markdown,
- `FUN` provides the user with the ability to apply user-defined functions to summarize numeric variables (any function that works with `tapply()`), and
- `FUN2`, similarly to `FUN`, allows the user to define a function to apply to all variables listed in the `second` argument above.

Notably, if no variables are listed, all variables in the `data.frame` will be summarized. Other options exist that can be used to better format and adjust the table, some of which we highlight in subsequent sections (e.g. `format_number` and `var_names`). However, in its simplest form, `table1()` only requires the data frame.

```
table1(df)
```

Yet, the function is designed for much more.

The function is built on fast base functions and is appropriate for even very large data sets ($n > 1,000,000$). It uses simple non-standard evaluation which allows the user to create and modify variables from within the function itself. For example, we could summarize levels of a dichotomous version of `var1` as seen below:

```
table1(df,
       ifelse(var1 > median(var1), 1, 0),
       splitby = ~stratifying_variable,
       test = TRUE)
```

Other simple modifications are also possible (e.g., `factor(var1)`, `var1*100`).

Exploratory data analysis with Table 1

The `table1()` function below is used on the NHANES data to explore differences in demographic and psychosocial factors between individuals who have and have not been informed by a doctor that he or she is overweight. The following code uses the `data.frame` named `d1`, selects 12 variables, stratifies by whether the individual was designated by a doctor as overweight (`splitby = ~overweight`), calls for tests of bivariate associations (`test = TRUE`), and is printed in the "text2" format (`output = "text2"`).

[2]The analyses for this paper used **furniture** 1.5.4 and R version 3.4.1.

```
table1(d1,
       gender, age, active, marijuana, illicit,
       down, sleeping, low_energy, appetite, feel_bad,
       dead, difficulty,
       splitby = ~overweight,
       test = TRUE,
       output = "text2")
```

The preceding code produces the following table. Note that, for space, several rows were excluded at the bottom of the table.

```
|=====================================================|
                           overweight
                Yes            No              P-Value
 -              ---            --              -------
 Observations   335            1082
 gender                                        0.004
    Male        139 (41.5%)    549 (50.7%)
    Female      196 (58.5%)    533 (49.3%)
 age                                           0.100
                23.65 (3.99)   23.24 (3.95)
 active                                        0.251
                151.36 (94.21) 166.83 (106.12)
 marijuana                                     0.333
    Yes         181 (59.9%)    535 (56.6%)
    No          121 (40.1%)    411 (43.4%)
 ...            ...            ...             ...
|=====================================================|
```

The table provides the n for each group, the means and standard deviations (SD) by group for each numeric variable, and the counts and percentages by group for each factor variable. The table can also provide bivariate statistical comparisons using the supplied stratifying variable as the independent variable. For categorical variables, a χ^2 test is performed. For numeric variables, either a t-test or an analysis of variance is performed (potentially adjusting for heterogeneity of variance), depending on the number of levels of the stratifying variable. Only the tests' p-values are shown by default. The user may also request the specific test statistics (`type = "full"`) or simply stars representing significance (`type = "stars"`).

If means and SDs are insufficient for a numeric variable, for example in the case of a highly skewed numeric variable, medians and the interquartile range (IQR) can be produced through the use of the "second" argument. This argument accepts quoted names of numeric variables, and for those variables, returns median and IQR in place of means and SDs by default.

```
table1(d1,
       gender, age, active, marijuana,
       splitby = ~overweight,
       test = TRUE,
       output = "text2",
       second = c("age", "active"))
```

```
|=================================================|
                          overweight
                Yes          No            P-Value
 -              ---          --            -------
 Observations   335          1082
 gender                                     0.004
    Male        139 (41.5%)  549 (50.7%)
    Female      196 (58.5%)  533 (49.3%)
 age                                        0.1
                23.00 [7.0]  23.00 [7.0]
 active                                     0.251
                120.00 [90.0] 135.00 [142.5]
 marijuana                                  0.333
    Yes         181 (59.9%)  535 (56.6%)
    No          121 (40.1%)  411 (43.4%)
|=================================================|
```

The medians and IQRs are differentiated from the means and SDs by the square brackets that are applied to the IQRs. This keeps the table clean but provides information on the types of statistics being presented.

Beneficially, the user can define a function to use in place of the default means/SDs and medians/IQRs. for numeric variables. This is through the use of the `FUN` and `FUN2` arguments. `FUN` applies to any variable not listed in the second argument. It can be as simple as `FUN = min`—asking for the minimum of each variable–or as complicated of a function as tapply will accept. For an example of a multi-statistic function, see below.

```
table1(d1,
       gender, age, active,
       splitby = ~overweight,
       test = TRUE,
       output = "text2",
       FUN = function(x) paste0(min(x, na.rm=TRUE), ", ", max(x, na.rm=TRUE)))
```

The preceding code allows the anonymous function [i.e. `function(x) paste0(min(x,...))`], to produce the minimum and maximum, be applied to each of the numeric variables (i.e. in this case, age and active). This code produces the following table.

```
|================================================|
                           overweight
                 Yes         No          P-Value
 -               ---         --          -------
 Observations    335         1082
 gender                                  0.004
    Male         139 (41.5%) 549 (50.7%)
    Female       196 (58.5%) 533 (49.3%)
 age                                     0.1
                 18, 30      18, 30
 active                                  0.251
                 30, 505     20, 840
|================================================|
```

In the same way, `FUN2` allows the user to specify another function to be applied to the indicated numeric variables within the same table. It also allows for simple formatting changes to these statistics as well (e.g. can insert brackets, commas, or change the rounding of the numbers). In essence, this makes `table1()` useful for a wide variety of situations.

We can also simplify and condense the table. As demonstrated below, `type = "simple"` produces only percentages for the categorical variables (instead of counts and percentages) and `type = "condense"` displays the summary only for the reference category of dichotomous variables and removes much of the white space. Together, they provide a much more succinct table.

```
table1(d1,
       gender, age, active, marijuana, illicit, down,
       splitby = ~overweight,
       test = TRUE,
       output = "text2",
       type = c("simple", "condense"))
```

```
|=====================================================|
                            overweight
                 Yes            No              P-Value
 -               ---            --              -------
 Observations    335            1082
 gender: Female  58.5%          49.3%           0.004
 age             23.65 (3.99)   23.24 (3.95)    0.1
 active          151.36 (94.21) 166.83 (106.12) 0.251
 marijuana: No   40.1%          43.4%           0.333
 illicit: No     89.7%          88.4%           0.584
 down                                           <.001
    No           69.7%          81.1%
    Several Days 21.4%          14.8%
    Majority     5.6%           2.8%
    Everyday     3.3%           1.3%
|=====================================================|
```

Finally, with the rise of "piping" (Wickham, 2016), we have integrated functionality that allows the table to be part of a bigger pipeline. When in a pipeline, the function auto-detects the pipe, prints the table and invisibly returns the original data so that it can continue to be used in subsequent functions. For example,

```
d1 %>%
  filter(age > 20) %>%
  table1(gender, age, active, marijuana, illicit, down,
         splitby = ~overweight,
         test = TRUE,
         output = "text2",
         type = c("simple", "condense")) %>%
  lm(overweight ~ age + gender, data = .)
```

In the end, these simple tables provide several pieces of important information. Without needing any advanced modeling, we already have an idea of several relationships among the study variables. For example, several demographic and psychosocial factors are related to the designation by a doctor as being overweight, including trouble sleeping and feeling down. These insights can inform model building and follow-up visualizations. `table1()`, in conjunction with other summary techniques (e.g. the base function `summary` in R), provides a quick and broad understanding of the patterns and relationships in the data.

Easy communication of descriptive statistics

In addition to the power of exploratory data analysis in informing statistical modeling, `table1()` is an important tool in scientific communication. For this reason, it was equipped with several output formatting features.[3] Five that are particularly useful are discussed below.

1. `output` allows for two regular console outputs (i.e., "text" and "text2") and all `knitr::kable` options, including "latex", "markdown" and "html."
2. `var_names` allows the user to provide a list of names that will replace the variable names in the table.
3. `format_number` provides formatting of numbers with commas (e.g., 22,000 instead of 22000) when set to `TRUE`.
4. `type`, in addition to the "simple" and "condense" options discussed previously, provides three options for the presentation of statistical tests: 1) "pvalues", which is default, and displays the p-values for the tests of association, 2) "stars" which provides the common star notation for p-values, and 3) "full" which provides the test statistics and the p-values.[4]

These, among other options, allow for the production of quality tables that can be easily published via various mediums including peer-reviewed journals and online webpages.

The example below illustrates the simplicity of communicating the basic statistics of the NHANES sample in Table 1.

```
  table1(d1,
         gender, age, active, marijuana, illicit,
         down, sleeping, dead,
         splitby = ~overweight,
         var_names = c("Gender", "Age", "Activity (minutes)", "Marijuana",
                       "Other Illicit Drug", "Feeling Down",
                       "Trouble Sleeping", "Wish Were Dead"),
         format_number = TRUE,
         type = "simple",
         test = TRUE,
         output = "latex")
```

With only a few lines of code, simple yet important information about the sample, study variables, and their bivariate relationships to a key variable are elegantly produced for easy dissemination. With minor touching up, this table can be ready for professional and peer-reviewed publications.[5]

[3] For a list of all argument options, see Barrett and Brignone (2016).

[4] Only the "pvalues" option works when `"simple"` or `"condense"` are also included.

[5] The splitby variable name at the top of Table 1—"Overweight"—was added afterwards. This information does not print automatically in the `kable` output.

Table 1: Latex table produced from `table1()` with a number of formatting options.

	Overweight		
	Yes	No	P-Value
Observations	335	1,082	
Gender			0.004
– Male –	41.5%	50.7%	
– Female –	58.5%	49.3%	
Age			0.1
	23.65 (3.99)	23.24 (3.95)	
Activity (minutes)			0.251
	151.36 (94.21)	166.83 (106.12)	
Marijuana			0.333
– Yes –	59.9%	56.6%	
– No –	40.1%	43.4%	
Other Illicit Drug			0.584
– Yes –	10.3%	11.6%	
– No –	89.7%	88.4%	
Feeling Down			<.001
– No –	69.7%	81.1%	
– Several Days –	21.4%	14.8%	
– Majority –	5.6%	2.8%	
– Everyday –	3.3%	1.3%	
Trouble Sleeping			<.001
– No –	53%	64.7%	
– Several Days –	29.6%	23.1%	
– Majority –	7.6%	6.8%	
– Everyday –	9.9%	5.4%	
Wish Were Dead			0.02
– No –	94.4%	97.9%	
– Several Days –	3.3%	1.3%	
– Majority –	1.3%	0.5%	
– Everyday –	1%	0.3%	

For those using processors other than latex or markdown, the table can exported to a spreadsheet program, such as Microsoft's Excel, via the `export` argument. Here, all that needs to be provided is a string that will be the outputted file name (e.g., "myfile"). This will export the table as a formatted CSV to a new folder in the working directory called "Table1." From there, the table can be imported into a word processor, such as Microsoft's Word. Another, more manual option is simply copying-and-pasting from the console output into a spreadsheet. Using the "text to columns" feature common in spreadsheet programs, this table can become ready to import into a word processor. Regardless of the approach, the common errors of manually inputting values into a table are greatly reduced.

Additional features

In addition to `table1()`, the **furniture** package contains a data cleaning function (i.e. `washer()`) and an operator (i.e. `%xt%`). `washer()` is used to replace values in a vector with other values in a way that avoids more complex `ifelse` statements. The `%xt%` operator can be used for simple cross-tabulations among two categorical variables. These, although not discussed much here, are helpful in getting the data ready for more in depth analysis in `table1()`. In fact, both were used to get the data in the format found in the package. More information can be found in the package documentation and at `tysonstanley.github.io`.

Summary

Ultimately, each function in the **furniture** package is designed to simplify important data exploration with the long-term goal of increasing both methods and results reproducibility. With the `table1()` function, the computation and communication of descriptive statistics is made simple and beautiful, streamlining the exploratory data analysis process. We hope that the user-friendly syntax and polished output enables researchers to efficiently perform more thorough exploration of their data and to more

easily communicate their findings.

Bibliography

T. Barrett and E. Brignone. *furniture: Furniture for Applied Quantitative Researchers*, 2016. URL https://CRAN.R-project.org/package=furniture. R package version 1.5.0. [p142, 146]

C. G. Begley and J. P. A. Ioannidis. Reproducibility in science: Improving the standard for basic and preclinical research. *Circulation Research*, 116(1):116–126, 2015. ISSN 15244571. doi: 10.1161/CIRCRESAHA.114.303819. [p142]

A. C. Chang and P. Li. Is Economics Research Replicable? Sixty Published Papers from Thirteen Journals Say "Usually Not". *Finance and Economics Discussion Series*, 083:1–26, 2015. ISSN 19362854. doi: 10.17016/FEDS.2015.083. [p142]

S. N. Goodman, D. Fanelli, and J. P. A. Ioannidis. What does research reproducibility mean? *Science Translational Medicine*, 8(341):1–6, 2016. ISSN 1946-6234. doi: 10.1126/scitranslmed.aaf5027. [p142]

National Center for Health Statistics. National Health and Nutrition Examination Survey Data. Technical report, U.S. Department of Health and Human Services, Centers for Disease Control and Prevention, Hyattsville, MD, 2016. URL http://www.cdc.gov/nchs/nhanes/. [p142]

Open Science Collaboration. Estimating the reproducibility of psychological science. *Science*, 349 (6251):aac4716–aac4716, 2015. ISSN 0036-8075. doi: 10.1126/science.aac4716. URL http://science.sciencemag.org/content/349/6251/aac4716. [p142]

R Core Team. *foreign: Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...*, 2016. URL https://CRAN.R-project.org/package=foreign. R package version 0.8-67. [p142]

J. Tukey. We Need Both Exploratory and Confirmatory. *The American Statistician*, 34(1):79–88, 1980. [p142]

H. Wickham. *tidyverse: Easily Install and Load 'Tidyverse' Packages*, 2016. URL https://CRAN.R-project.org/package=tidyverse. R package version 1.0.0. [p142, 146]

Tyson S. Barrett
Utah State University
2810 Old Main Hill, Logan, UT, 84322
U.S.A
tyson.barrett@usu.edu

Emily Brignone
VA Pittsburgh Healthcare System
Pittsburgh, PA 15240
U.S.A
emilybrignone@gmail.com