The Stata Journal

The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go "beyond the Stata manual" in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

The *Stata Journal* is indexed and abstracted by *CompuMath Citation Index*, *Current Contents/Social and Behavioral Sciences*, *RePEc: Research Papers in Economics*, *Science Citation Index Expanded* (also known as *SciSearch*), *Scopus*, and *Social Sciences Citation Index*.

For more information on the *Stata Journal*, including information for authors, see the webpage

http://www.stata-journal.com

Subscriptions are available from StataCorp, 4905 Lakeway Drive, College Station, Texas 77845, telephone 979-696-4600 or 800-STATA-PC, fax 979-696-4601, or online at

http://www.stata.com/bookstore/sj.html

Subscription rates listed below include both a printed and an electronic copy unless otherwise mentioned.

U.S. and Canada		Elsewhere	
Printed & electronic		**Printed & electronic**	
1-year subscription	$115	1-year subscription	$145
2-year subscription	$210	2-year subscription	$270
3-year subscription	$285	3-year subscription	$375
1-year student subscription	$ 85	1-year student subscription	$115
1-year institutional subscription	$345	1-year institutional subscription	$375
2-year institutional subscription	$625	2-year institutional subscription	$685
3-year institutional subscription	$875	3-year institutional subscription	$965
Electronic only		**Electronic only**	
1-year subscription	$ 85	1-year subscription	$ 85
2-year subscription	$155	2-year subscription	$155
3-year subscription	$215	3-year subscription	$215
1-year student subscription	$ 55	1-year student subscription	$ 55

Back issues of the *Stata Journal* may be ordered online at

http://www.stata.com/bookstore/sjj.html

Individual articles three or more years old may be accessed online without charge. More recent articles may be ordered online.

http://www.stata-journal.com/archives.html

The *Stata Journal* is published quarterly by the Stata Press, College Station, Texas, USA.

Address changes should be sent to the *Stata Journal*, StataCorp, 4905 Lakeway Drive, College Station, TX 77845, USA, or emailed to sj@stata.com.

The *Stata Journal* (ISSN 1536-867X) is a publication of Stata Press. Stata, Stata, Stata Press, Mata, Mata, and NetCourse are registered trademarks of StataCorp LP.

The Stata Journal (2015)
15, Number 3, pp. 775–783

Creating summary tables using the sumtable command

Lauren J. Scott
Clinical Trials and Evaluation Unit
University of Bristol
Bristol, UK
Lauren.Scott@bristol.ac.uk

Chris A. Rogers
Clinical Trials and Evaluation Unit
University of Bristol
Bristol, UK
Chris.Rogers@bristol.ac.uk

Abstract. In many fields of statistics, summary tables are used to describe characteristics within a study population and are often used to compare characteristics of two or more groups. The `sumtable` command can be used to quickly and easily produce such summary tables, allowing for different types of data to be summarized within one table. In this article, we provide examples and advice for its use.

Keywords: st0403, sumtable, summarize, mean, median, standard deviation, interquartile range, percentage, Excel spreadsheet

1 Introduction

The `sumtable` command provides a simple method of producing summary tables of data from two or more groups. It builds this summary table within a Stata dataset and allows different types of summary measures (for example, means and standard deviations [SDs] and numbers and percentages) within one table. In this sense, it is perhaps more flexible than other Stata summary programs that are currently available.

The command removes any manual aspect of creating summary tables (for example, cutting and pasting) and therefore eliminates transposition errors. It also makes creating a summary table quick and easy, and it is especially useful if data are updated and tables subsequently need to change. The command creates an Excel spreadsheet that can be easily manipulated for reports or other documents. Although this command was written in the context of clinical trials, it would be equally useful in many other contexts.

The four types of data that can be summarized using this command are binary data; multicategory data; continuous data, where a mean and an SD are required; and continuous data, where a median and an interquartile range (IQR) are required.

2 The sumtable command

2.1 Description

This command creates summary tables by group—this may be treatment groups in a clinical trial or cohort groups in an observational study. The type of summary required

for each variable will depend on the data type. Three columns of summary data are produced for each group: for binary and multicategory data, the first two columns contain counts and percentages, and for continuous data, the first two columns contain either means and SDs or medians and IQRs. For all summary types, the third column for each group contains a count of the number of missing values for each summarized variable.

Within one table, there may be a requirement to present a combination of different summary types. For example, in a clinical trial, a table of baseline characteristics may consist of variables such as age (continuous; means and SDs), gender (binary; counts and percentages), height (continuous; means and SDs), and smoking status (multicategory; counts and percentages). The `sumtable` command enables the user to create such a table containing all of these variables.

2.2 Syntax

`sumtable` *sumvar groupvar*`, vartype(`*vartype_options*`)` [`vartext(`*text*`) dp1(`#`) dp2(`#`) first(1) last(1) exportname(`*text*`)`]

where *sumvar* is the variable to be summarized and *groupvar* is a group variable by which the *sumvar* data are summarized. The *groupvar* variable must be numeric.

2.3 Options

`vartype(`*vartype_options*`)` identifies the summary type. *vartype_options* may be `binary`, `contmean`, `contmed`, `categorical`, or `headerrow`. `vartype()` is required; one of the four summary types (or `headerrow`) must be specified. See section 2.4 below for details.

`vartext(`*text*`)` specifies the label to describe the variable that is being summarized. By default, the label in the summary table will be the variable label assigned to the variable in the dataset. If the variable has no variable label, the label in the summary table will be the variable name. For example, there may be a variable called `smoke` that the user wishes to label as `Smoking status` in the summary table. In this example, if the `smoke` variable had no variable label assigned, the text in the summary table would simply be `smoke`, unless `vartext("Smoking status")`, for example, is specified.

`dp1(`#`)` specifies the number of decimal places displayed for the first statistic in each group. For `binary` and `categorical` summaries, the first statistic is a count (so no decimal places are necessary). For `contmean` and `contmed` summaries, the first statistics are the mean and the median, respectively. The default is `dp1(0)`.

`dp2(`#`)` specifies the number of decimal places displayed for the second statistic in each group. For `binary` and `categorical` summaries, the second statistic is a percentage.

For `contmean` and `contmed` summaries, the second statistics are the SD and IQR, respectively. The default is `dp2(1)`.

`first(1)` sets up the temporary dataset that all subsequent summaries are appended to. This option should be specified for the first row of a table (that is, the first time any `sumtable` code is run for this summary table). Each time this option is specified, the temporary dataset is replaced.

`last(1)` deletes temporary datasets and produces an Excel document of the final summary table. This option should be specified for the last row of a table (that is, the last time any `sumtable` code is run for this summary table).

`exportname(`*text*`)` specifies the name or path name assigned to the Excel summary dataset that is produced from this command. This option should be used for only the last row of a table alongside the `last(1)` option. If this is not specified but `last(1)` is, the resultant summary table will be named `summarydatasetexcel.xls` and be saved in the user's current directory.

2.4 Types of summaries (vartype_options)

`binary` produces two main columns for each group: a count of the number of entries coded as 1s with the totals in each group (n/N) and the corresponding percentages calculated from the total number of entries excluding missing data. A third additional column is also created for each group containing missing counts. This `vartype()` option should be specified for numeric binary variables coded 0 and 1, where only the number of 1s is of interest. For example, for the variable `has the participant ever had a stroke`, a `1` could indicate a response of yes, and a `0` could indicate a response of no. Therefore, this `vartype()` option would summarize (by group) the number, total, and percentage of participants who previously had a stroke and the number of participants for whom this data item is missing.

`categorical` produces counts, totals, and corresponding percentages for each category of the specified variable of interest as well as a missing count for each group. This `vartype()` option should be specified for multicategory variables, binary variables that are not coded 0 and 1, or binary variables where reporting both categories is of interest. If, for example, `New York Heart Association class` is coded 1, 2, 3, or 4 for the four classes, this `vartype()` option would produce counts, totals, percentages, and missing counts (by group) of each of the four New York Heart Association classes. This `vartype()` option can be used to summarize categorical variables that are coded as strings or as numeric variables. `vartype(categorical)` will create multiple summary rows per variable (one for each category), while all other `vartype()` options create just one summary row per variable. If any of the variables summarized in the table are categorical, a column labeled `levellab` will be included in the resultant Excel summary table and will be blank for all variables other than those that are summarized using `vartype(categorical)`. If the categorical variable to be summarized is a string variable, the categories will be listed in alphabetical order. If the categorical variable is numeric and labeled in the Stata

dataset, the categories will be labeled and listed in number order. If the categorical variable is numeric and not labeled in the Stata dataset, the categories will be labeled as their numeric value.

`contmean` specifies that continuous variables be summarized by means and SDs (usually normally distributed variables). For example, the variable `height` is usually normally distributed, so this `vartype()` option could be used to calculate the mean and SD (by group) of the height of the study population. As with other `vartype()` options, missing counts are also calculated for each group.

`contmed` specifies that continuous variables be summarized by medians and IQRs (usually nonnormally distributed or discrete variables). For example, in surgical trials, the variable `operation length` is usually skewed, so `vartype(contmed)` could be specified to display the median and IQR (by group) of operation length in the study population. As with other `vartype()` options, missing counts are also calculated for each group.

`headerrow` breaks up a summary table. This `vartype()` option is not necessary, but it may be useful to split the final summary table into sections. For example, in a demography table in a cardiac clinical trial, it may be useful to divide the table into sections such as `baseline characteristics`, `cardiac history`, and `drugs on admission`. For `vartype(headerrow)`, there is no variable to be summarized, so only the group variable should be specified. Section titles can be entered using the `vartext()` option. `dp1()` and `dp2()` do not apply if this `vartype()` option is specified. An example of the code used to specify the top row of the example demography table is `sumtable` *groupvar*`, vartype(headerrow) vartext("Baseline characteristics") first(1)`.

3 Output

After a series of commands that identify the variable to be summarized, the group variable, and the options about how the summary statistics will be formatted, the result is an Excel document with a label column, a level label column (if categorical variables have been summarized), and three columns for each group and overall (as described in section 2.1). Variables are ordered in the summary table in the order that they are run in Stata. The first column for the first group in a table is labeled `stat1_grp1` (if the groups are 1, 2, 3, etc.), the second column is labeled `stat2_grp1`, and the third is labeled `miss_grp1`. These labels are assigned using the group values (not the number of groups); for example, if there were two groups with group values 3 and 29, the columns for the second group would be labeled `stat1_grp29`, `stat2_grp29`, and `miss_grp29`.

4 Example 1

This example is a step-by-step guide to using the `sumtable` command. In this example, we use `nlsw88.dta`. This dataset contains several variables, including education and career information for 2,246 individuals.

We first set the current directory to the directory where the resultant Excel summary table is to be saved (using the `cd` command), and then we read the `nlsw88.dta` into Stata using the command

```
. sysuse nlsw88
(NLSW, 1988 extract)
```

The differences between college graduates and noncollege graduates, in variables such as marital status, wages, or number of hours worked, may be of interest. In this dataset, the numeric variable `collgrad` identifies graduate status and is coded `1` for college graduates and `0` for noncollege graduates. There are 532 individuals who are college graduates and 1,714 individuals who are not college graduates.

If the average wage of college graduates versus that of noncollege graduates is of interest, for example, the first row in the summary table could be added with the following code:

```
. sumtable wage collgrad, vartype(contmed) first(1)
```

This is the simplest code that could be specified for the first line of a table. Here no labels or decimal place options have been specified, so these would all revert to default. The `first(1)` option must be specified here because this is the first line in the table. This command would create a dataset with one row containing median wage split by college-graduate status. Because the `last(1)` option has not been specified, this code would not create an Excel output dataset, but it would create a temporary Stata dataset called `dummy1234.dta`, which would be appended to when the next line of code identifying the next variable to be summarized is run. The Stata dataset created from the above code is as follows:

```
. use dummy1234
(NLSW, 1988 extract)
. list
```

1.

label	level	levellab	stat1_~0	stat2_grp0	stat1_~1
hourly wage	.		6	(4.0, 8.1)	10

stat2_grp1	stat1_~1	stat2_all	miss_g~0	miss_g~1	miss_all
(6.6, 12.4)	6	(4.3, 9.6)	0	0	0

In this output, `grp0` refers to noncollege graduates, and `grp1` refers to college graduates, because this is how they are coded in `nlsw88.dta`. The label for this variable is `hourly wage` because this is the label for this variable in `nlsw88.dta`. This table shows that the median wage is higher for college graduates than for noncollege graduates in this dataset. This table also shows that there are no missing wage data in either group.

The default number of decimal places for `stat1` (the median in this example) is zero. However, if more detail is desired, this could be displayed to one decimal place instead. To do this, we could extend the above code to create the output as follows:

```
. sysuse nlsw88
(NLSW, 1988 extract)
. sumtable wage collgrad, vartype(contmed) first(1) dp1(1)
. use dummy1234
(NLSW, 1988 extract)
. list
```

1.

label	level	levellab	stat1_~0	stat2_grp0	stat1_~1
hourly wage	.		5.6	(4.0, 8.1)	9.7

stat2_grp1	stat1_~l	stat2_all	miss_g~0	miss_g~1	miss_all
(6.6, 12.4)	6.3	(4.3, 9.6)	0	0	0

This output shows that the addition of `dp1(1)` in the code has changed the table to include one decimal place in the columns labeled `stat1`.

For clarity, it may be necessary to also include the currency in the label (for example, a pound or dollar sign). To do this, we could extend the code as follows:

```
. sysuse nlsw88
(NLSW, 1988 extract)
. sumtable wage collgrad, vartype(contmed) first(1) dp1(1)
> vartext("Hourly wage ($)")
. use dummy1234
(NLSW, 1988 extract)
. list
```

1.

label	level	levellab	stat1_~0	stat2_grp0	stat1_~1
Hourly wage ($)	.		5.6	(4.0, 8.1)	9.7

stat2_grp1	stat1_~l	stat2_all	miss_g~0	miss_g~1	miss_all
(6.6, 12.4)	6.3	(4.3, 9.6)	0	0	0

For this example, we may decide that wage is the only variable that we want summarized in this table. In this case, an Excel spreadsheet summarizing just the hourly wage could be produced as follows:

```
. sysuse nlsw88
(NLSW, 1988 extract)
. sumtable wage collgrad, vartype(contmed) first(1) dp1(1)
> vartext("Hourly wage ($)") last(1)
```

label	stat1_grp0	stat2_grp0	stat1_grp1	stat2_grp1	stat1
Hourly wage ($)	5.6	(4.0, 8.1)	9.7	(6.6, 12.4)	6.3

Note: The Excel spreadsheet is incomplete because of size restraints on the page.

Because this is the Excel spreadsheet, the `level` column has now been dropped. Additionally, because there are no categorical variables summarized in this table, the `levellab` column is empty and is therefore dropped. When the `last(1)` option is

specified, the temporary dataset `dummy1234.dta` is deleted and replaced by the Excel spreadsheet.

Finally, this Excel spreadsheet is likely to need a name to identify it. The default name is `summarydatasetexcel.xls`; however, it could instead be named `Wages by graduate status.xls`, for example. This could be done as follows:

```
. sumtable wage collgrad, vartype(contmed) first(1) dp1(1)
> vartext("Hourly wage ($)") last(1) exportname("Wages by graduate status")
```

This code would export the exact same Excel table as above but simply save it with a different name.

5 Example 2

In this example, we use Stata's `auto.dta` and demonstrate how to use several `sumtable` commands to create a summary table with multiple variable summaries. Summaries will be grouped by different car types by using `foreign` as the group variable. `foreign` is a numeric variable coded `0` = Domestic and `1` = Foreign. There are 74 entries in this dataset, comprising 52 domestic cars and 22 foreign cars.

5.1 The commands

```
. sysuse auto
(1978 Automobile Data)
. sumtable foreign, vartype(headerrow) vartext("CAR DETAILS") first(1)
. sumtable price foreign, vartype(contmed) dp1(0) dp2(0)
. sumtable weight foreign, vartype(contmean) dp1(0) dp2(0)
. sumtable length foreign, vartype(contmean) dp1(1) dp2(1)
. sumtable rep78 foreign, vartype(categorical) vartext("Repairs since 1978")
> dp1(0) dp2(1) last(1) exportname("Details by car groups")
```

5.2 The summary table

label	levellab	s	p1	stat1_all	stat2_all	miss_grp0	miss_grp1	miss_all
CAR DETAILS								
Price		4	140)	5007	(4195, 6342)	0	0	0
Weight (lbs.)		3		3019	777	0	0	0
Length (in.)		1		187.9	22.3	0	0	0
Repairs since 1978	1	2		2/69	2.9%	4	1	5
	2	8		8/69	11.6%	4	1	5
	3	2		30/69	43.5%	4	1	5
	4	9		18/69	26.1%	4	1	5

Note: The Excel spreadsheet is incomplete because of size restraints on the page.

5.3 Comments

For binary and multicategory variables, counts and percentages are always calculated after excluding any missing data. We can see in the `stat1_all` column that 69 entries included a record of the number of repairs since 1978. Therefore, there are five entries with missing data, as we can see in the `miss_all` column.

All variables in `auto.dta` have variable labels assigned and can therefore be accurately labeled in the summary table without having to specify a new label in the summary table code. However, if a different label is required, the Stata variable label can be overwritten, as has been demonstrated here; the variable `rep78` is labeled `Repair Record 1978` in `auto.dta`, but this label has been replaced with the new label `Repairs since 1978` in the summary table, as stated in the last line of code. The label in the Stata dataset remains unchanged.

This Excel summary table is saved in the user's current directory with the name `Details by car groups.xls`, as specified in the last line of code. If the user wishes to save it in another directory, a path name can be specified with the `exportname()` option.

This table would need some editing before use in a report or an article. Because different summary types are reported within the same table, one must decide how the columns will be labeled. For the above example, there are more rows that are summarized by counts and percentages than any other summary type, so the user might choose to label the columns as such and change the label text to describe the summary type for all variables not described by counts and percentages. The changes in the label-column text could be done manually at this stage or could be written into the code using the `vartext()` option. For the final table, the missing counts would not usually be displayed as part of the table, but these counts could be used to create footnotes to the table to document missing data. The missing counts are not necessary in this table because there are no missing data for the continuous variables, and the amount of missing data for repairs since 1978 is clear from the overall count in each group and the totals at the top of the table. However, we include a footnote with the table for illustrative purposes.

Car details	Group 0 (n=52)		Group 1 (n=22)		Overall (n=74)	
	n	%	n	%	n	%
Price (median, IQR)	4783	(4184, 6234)	5759	(4499, 7140)	5007	(4195, 6342)
Weight (lbs.) (mean, SD)	3317	695	2316	433	3019	777
Length (in.) (mean, SD)	196.1	20.0	168.5	13.7	187.9	22.3
Repairs since 1978 *						
1	2/48	4.2%	0/21	0.0%	2/69	2.9%
2	8/48	16.7%	0/21	0.0%	8/69	11.6%
3	27/48	56.3%	3/21	14.3%	30/69	43.5%
4	9/48	18.8%	9/21	42.9%	18/69	26.1%
5	2/48	4.2%	9/21	42.9%	11/69	15.9%

Missing counts: *4 in group 0, 1 in group 1.

6 Conclusion

`sumtable` was written to simplify generating summary tables. Although there are already commands in Stata that summarize data, as well as commands that output summary tables, to our knowledge, there are currently no commands that have the flexibility to combine different data types and to output them in a usable table format.

`sumtable` makes creating summary tables quick and easy. It removes the need to cut and paste, eliminating transposition errors, and it is especially beneficial when producing reports and summaries that regularly need updating.

7 Acknowledgments

We thank the financial support from the National Institute for Health Research Bristol Biomedical Research Unit in Cardiovascular Disease. We also thank the Bristol Clinical Trials and Evaluation Unit statistics team, especially Katie Pike, for their support with this work.

About the authors

Lauren Scott is a medical statistician working at the University of Bristol Clinical Trials and Evaluation Unit. Her work involves the design, data management, and analysis of randomized controlled trials and observational studies.

Chris Rogers is a reader in medical statistics and codirector of the Clinical Trials and Evaluation Unit at the University of Bristol. Her key research interest is the design and analysis of clinical trials in surgery.