



AgEcon SEARCH
RESEARCH IN AGRICULTURAL & APPLIED ECONOMICS

The World's Largest Open Access Agricultural & Applied Economics Digital Library

This document is discoverable and free to researchers across the globe due to the work of AgEcon Search.

Help ensure our sustainability.

Give to AgEcon Search

AgEcon Search

<http://ageconsearch.umn.edu>

aesearch@umn.edu

*Papers downloaded from **AgEcon Search** may be used for non-commercial purposes and personal study only. No other use, including posting to another Internet site, is permitted without permission from the copyright owner (not AgEcon Search), or as allowed under the provisions of Fair Use, U.S. Copyright Act, Title 17 U.S.C.*

THE STATA JOURNAL

Editors

H. JOSEPH NEWTON
Department of Statistics
Texas A&M University
College Station, Texas
editors@stata-journal.com

NICHOLAS J. COX
Department of Geography
Durham University
Durham, UK
editors@stata-journal.com

Associate Editors

CHRISTOPHER F. BAUM, Boston College
NATHANIEL BECK, New York University
RINO BELLOCCO, Karolinska Institutet, Sweden, and
University of Milano-Bicocca, Italy
MAARTEN L. BUIS, WZB, Germany
A. COLIN CAMERON, University of California–Davis
MARIO A. CLEVES, University of Arkansas for
Medical Sciences
WILLIAM D. DUPONT, Vanderbilt University
PHILIP ENDER, University of California–Los Angeles
DAVID EPSTEIN, Columbia University
ALLAN GREGORY, Queen's University
JAMES HARDIN, University of South Carolina
BEN JANN, University of Bern, Switzerland
STEPHEN JENKINS, London School of Economics and
Political Science
ULRICH KOHLER, University of Potsdam, Germany

FRAUKE KREUTER, Univ. of Maryland–College Park
PETER A. LACHENBRUCH, Oregon State University
JENS LAURITSEN, Odense University Hospital
STANLEY LEMESHOW, Ohio State University
J. SCOTT LONG, Indiana University
ROGER NEWSON, Imperial College, London
AUSTIN NICHOLS, Urban Institute, Washington DC
MARCELLO PAGANO, Harvard School of Public Health
SOPHIA RABE-HESKETH, Univ. of California–Berkeley
J. PATRICK ROYSTON, MRC Clinical Trials Unit,
London
PHILIP RYAN, University of Adelaide
MARK E. SCHAFFER, Heriot-Watt Univ., Edinburgh
JEROEN WEESIE, Utrecht University
NICHOLAS J. G. WINTER, University of Virginia
JEFFREY WOOLDRIDGE, Michigan State University

Stata Press Editorial Manager

LISA GILMORE

Stata Press Copy Editors

DAVID CULWELL and DEIRDRE SKAGGS

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	
1-year subscription	\$ 79	1-year subscription	\$115
2-year subscription	\$155	2-year subscription	\$225
3-year subscription	\$225	3-year subscription	\$329
3-year subscription (electronic only)	\$210	3-year subscription (electronic only)	\$210
1-year student subscription	\$ 48	1-year student subscription	\$ 79
1-year university library subscription	\$ 99	1-year university library subscription	\$135
2-year university library subscription	\$195	2-year university library subscription	\$265
3-year university library subscription	\$289	3-year university library subscription	\$395
1-year institutional subscription	\$225	1-year institutional subscription	\$259
2-year institutional subscription	\$445	2-year institutional subscription	\$510
3-year institutional subscription	\$650	3-year institutional subscription	\$750

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.



Copyright © 2012 by StataCorp LP

Copyright Statement: The *Stata Journal* and the contents of the supporting files (programs, datasets, and help files) are copyright © by StataCorp LP. The contents of the supporting files (programs, datasets, and help files) may be copied or reproduced by any means whatsoever, in whole or in part, as long as any copy or reproduction includes attribution to both (1) the author and (2) the *Stata Journal*.

The articles appearing in the *Stata Journal* may be copied or reproduced as printed copies, in whole or in part, as long as any copy or reproduction includes attribution to both (1) the author and (2) the *Stata Journal*.

Written permission must be obtained from StataCorp if you wish to make electronic copies of the insertions. This precludes placing electronic copies of the *Stata Journal*, in whole or in part, on publicly accessible websites, file servers, or other locations where the copy may be accessed by anyone other than the subscriber.

Users of any of the software, ideas, data, or other materials published in the *Stata Journal* or the supporting files understand that such use is made without warranty of any kind, by either the *Stata Journal*, the author, or StataCorp. In particular, there is no warranty of fitness of purpose or merchantability, nor for special, incidental, or consequential damages such as loss of profits. The purpose of the *Stata Journal* is to promote free communication among Stata users.

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.

HTML output in Stata

Llorenç Quintó
Biostatistics Unit
Barcelona Centre for International Health Research
(CRESIB, Hospital Clínic—Universitat de Barcelona)
Barcelona, Spain
llorenc.quinto@cresib.cat

Sergi Sanz, Elisa De Lazzari, and John J. Aponte
Biostatistics Unit
Barcelona Centre for International Health Research
(CRESIB, Hospital Clínic—Universitat de Barcelona)
Barcelona, Spain

Abstract. In this article, we present a suite of basic commands that facilitate the production of HTML files in Stata. Creating HTML files in Stata allows for the programmatic production of formatted statistical reports that users can easily open without proprietary software, a feature long desired by many Stata statisticians.

Keywords: dm0066, htclose, htexample, htlist, htlog, htopen, htput, htsummary, HTML, logging

1 Introduction

One of the daily tasks of a professional statistician is producing a formatted output. In many cases, producing a statistical report is laborious and time consuming. Certainly, we all have suffered the misfortune of having to repeat a statistical report because of a last-minute change in the database or a previously undetected error. Stata do-files enable users to replicate the analysis and to get the new results by saving the output in a log file. However, on “handmade” tables, there is a high risk of introducing transcription mistakes or forgetting to update a table that was changed in the last version of the analysis.

Therefore, we asked how we could output results directly from Stata that were

- understood by researchers who are not used to Stata output;
- accessible without having the program; and
- reproducible and easy to update in case of any change in the data.

There have been several attempts to produce Stata output that other software can use to help statisticians produce reports. However, there is nothing that allows statisticians to produce a formatted output other than the basic logging mechanism with Stata Markup and Control Language, which users cannot open without Stata.

Hypertext Markup Language (HTML) has become the de facto standard on the Internet. An HTML page is a text file with a predefined set of tags that allows a web browser such as Microsoft Internet Explorer, Mozilla Firefox, or Chrome to render a formatted page. Statisticians usually know very little about HTML; however, this is a very simple language that is easily learned. A complete tutorial about HTML is beyond the scope of this article, but there are many tutorials on the web, for example, at <http://www.w3schools.com/html>.

In this article, we present a suite of commands that allows the programmer to not only produce the standard logging system but also produce output with HTML. First, we present the original idea, and then we present the commands used to open and close HTML files and the commands used to redirect the output to an HTML file. Several illustrative examples are also presented. The system presented here can be used with Stata version 10.0 or newer.

2 The initial idea

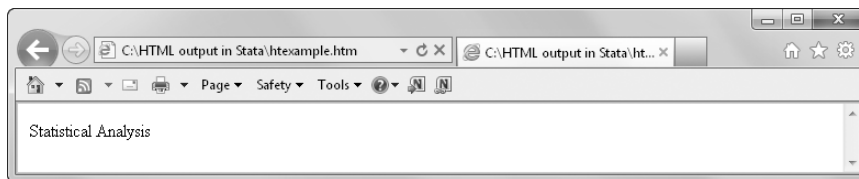
The Stata `file` command can generate text files. Given that HTML pages are text files, we explored what would happen if we used the extension `.html` (or `.htm`) to produce such files.

2.1 Example 1

To illustrate the initial idea, we execute the following code:

```
tempname handle
file open `handle' using htexample.html, text write
file write `handle' "Statistical Analysis"
file close `handle'
```

After running the code, we get a file called `htexample.html`. This is a log file in text format but with the extension `.html`, which makes it directly interpretable by any web browser. In fact, we do not need to close the file, because we use a `tempname` for `handle`. But our goal is not to explain the use of the command `file`; therefore, we include this line simply to clarify the procedure. The produced HTML file is syntactically incorrect (does not accomplish all standards for HTML files), but the browser tries to make its best rendering of the information.



This simple idea has enormous potential. It means that we can send text to an HTML page; therefore, we have confirmed that there is a communication path between Stata and our Internet browser. We just need to find a way to show results in an appropriate format.

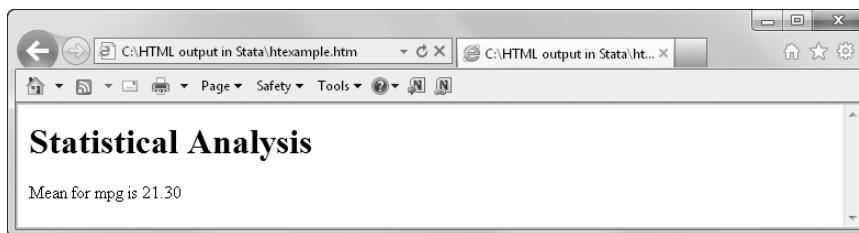
2.2 Example 2

Now modify example 1 to find the result of combining Stata commands and HTML tags:

```
tempname handle
file open `handle' using htexample.html, text write replace
file write `handle' "<h1> Statistical Analysis </h1>"
sysuse auto
summarize mpg
file write `handle' "Mean for mpg is `': display %8.2f r(mean)'"
file close `handle'
```

In the above code, we added tags from HTML (between the symbols < and >) that have a specific meaning. Thus the browser can interpret them properly. The tags in the example mean that the text between them should be displayed in title size. Then we added Stata commands to read in a file and calculate summary statistics. Finally, we wrote the mean of the variable `mpg` on the `htexample.html` file.

Now the page `htexample.html` looks like this (from now on, if you already have the page opened, you must refresh it to see the result of the last run):



With these two examples, we see how we can produce reports in HTML, but we are not completely satisfied. We need a simple and efficient way to send results to the HTML page while we view them in the Stata Results window. Let us describe the `ht` package, which is a suite of commands that will greatly facilitate this process.

3 The ht package

The `ht` package is a set of commands designed to produce HTML files from Stata; these files might contain text, HTML tags, tables, standard output from Stata, and linked objects. Any web browser can then open these HTML files. Stata graphics can be included as linked objects after you export them to an image format (for example, the `.png` format) by using the command `graph export`.

3.1 Opening and closing HTML output

htopen syntax

htopen using *filename* [, replace append notag]

htopen description

htopen opens an HTML output file. If *filename* does not end with `.html`, `.htm`, or `.css` (used for formatting structured content of HTML files), then `.html` is used. By default, an HTML file is produced with some header tags, such as `<!DOCTYPE ...>` and `<HTML>`. It uses the “DTD HTML 4.01 Transitional” version of HTML, which is interpreted correctly by most browsers. With options `append` or `notag`, the header tags are not included. Only one HTML file can be open at a time; if you try to open a new file when one is already open, you will get an error message. The *filename* is affected by the limitations of the macro substitution length.

htopen options

`replace` specifies that *filename* be overwritten if it already exists. If neither `replace` nor `append` is specified, the file is assumed to be new; if the specified file already exists, an error message is issued, and HTML output is not started.

`append` specifies that results be appended to an existing file. If the file does not already exist, a new file is created.

`notag` omits the header tags (`<!DOCTYPE ...>` and `<HTML>`).

htclose syntax

htclose [, notag]

htclose description

htclose closes a previously opened HTML file. By default, the closing tag `</HTML>` is produced. An error message is produced when no HTML output file has been previously opened.

htclose option

`notag` omits the HTML tag to close the document (`</HTML>`).

3.2 Sending output to the HTML file

htput and htlog syntax

`htput` *expression*

`htlog` *stata_cmd*

htput and htlog description

`htput` puts the text in the HTML file. Any local or global macro is evaluated, and the final macro substitution is sent to the HTML file. The text can also include HTML tags that the Internet browser will interpret correctly. This opens a wide range of possibilities to define font properties (size, color, etc.) and paragraphs as well as to insert images.

`htlog` sends the output from any Stata command to the HTML file.

htput and htlog remarks

These commands redirect to the HTML file but preserve the state of any previous log file. This allows output to be sent from the same program to the normal log and HTML files. The `htput` command is easy to use because it does not require quotation marks, and the `htlog` command is useful to send the output of commands to the HTML file.

An error message is produced when there is no HTML output opened previously.

3.3 Example 3

Using these new commands, we can reproduce example 1, as follows:

```
htopen using htexample, replace
htput <h1> Statistical Analysis </h1>
htclose
```

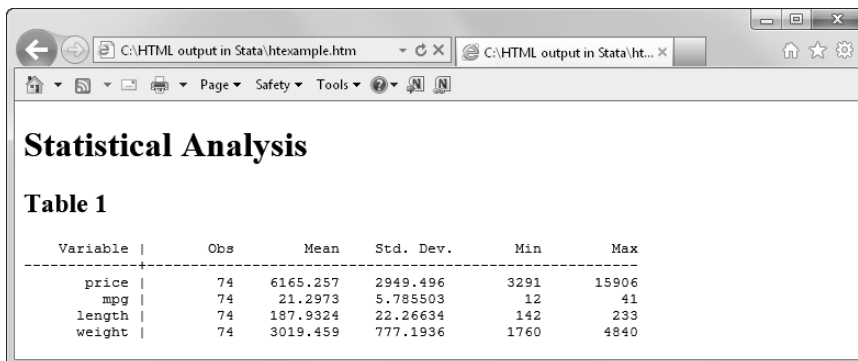
We can also reproduce example 2:

```
htopen using htexample, replace
htput <h1> Statistical Analysis </h1>
sysuse auto
summarize mpg
htput Mean for mpg is `': display %8.2f r(mean)`
htclose
```


3.4 Example 4

In this example, we will demonstrate the `htlog` command, which sends any command output to the HTML page.

```
htopen using htexample, replace
htput <h1> Statistical Analysis </h1>
sysuse auto
htput <h2> Table 1 </h2>
htlog summarize price mpg length weight
htclose
```

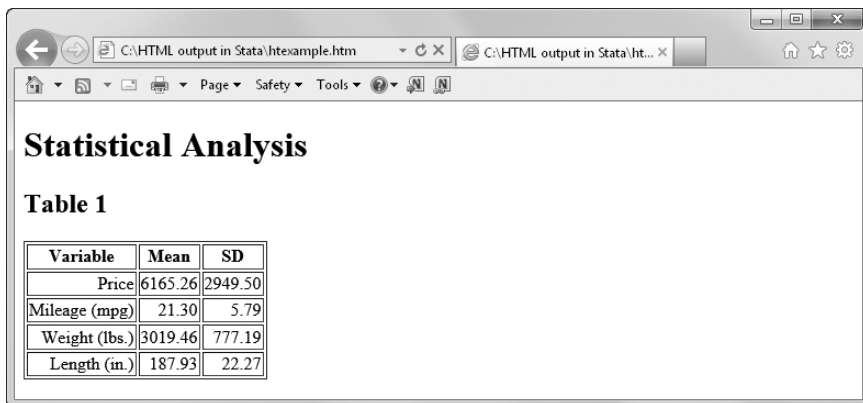


Variable	Obs	Mean	Std. Dev.	Min	Max
price	74	6165.257	2949.496	3291	15906
mpg	74	21.2973	5.785503	12	41
length	74	187.9324	22.26634	142	233
weight	74	3019.459	777.1936	1760	4840

3.5 Example 5

HTML syntax is useful for making a table with descriptive results similar to that included in a report or sent for publication.

```
htopen using htexample, replace
htput <h1> Statistical Analysis </h1>
sysuse auto
htput <h2> Table 1 </h2>
htput <table border=1>
htput <tr>
htput <th>Variable</th>
htput <th>Mean</th>
htput <th>SD</th>
htput </tr>
foreach var of varlist price mpg weight length {
    local lab: var lab `var'
    summarize `var'
    local mean: display %8.2f r(mean)
    local sd: display %8.2f r(sd)
    htput <tr align=right>
    htput <td>`lab'</td>
    htput <td>`mean'</td>
    htput <td>`sd'</td>
    htput </tr>
}
htput </table>
htclose
```



Statistical Analysis

Table 1

Variable	Mean	SD
Price	6165.26	2949.50
Mileage (mpg)	21.30	5.79
Weight (lbs.)	3019.46	777.19
Length (in.)	187.93	22.27

3.6 Predesigned tables

This section includes two ado-files developed to make tables. Both of them use the command `htput` and require an HTML file opened with `htopen`.

The first ado-file concerns lists, which are widely used tables where the cell contents are observations and variables are from the dataset. We have developed the ado-file `htlist` to produce lists in HTML format.

htlist syntax

```
htlist [varlist] [if] [in] [, display nodisplay nolabel noobs novarlab
      varname table(string) align(string)]
```

htlist description

`htlist` displays the values of variables in the HTML file. If `varlist` is not specified, the values of all the variables are displayed. `htlist` requires the HTML output to be set up with `htopen`.

htlist options

`display` or `nodisplay` determines the style of output. By default, `htlist` determines whether to use a table (`nodisplay`) or to `display` output based on the number of variables to be displayed.

`nolabel` causes the numeric codes rather than the label values to be displayed.

`noobs` suppresses the listing of the observation numbers.

`novarlab` suppresses the printing of the default variable label.

`varname` specifies to include the variable name in the output.

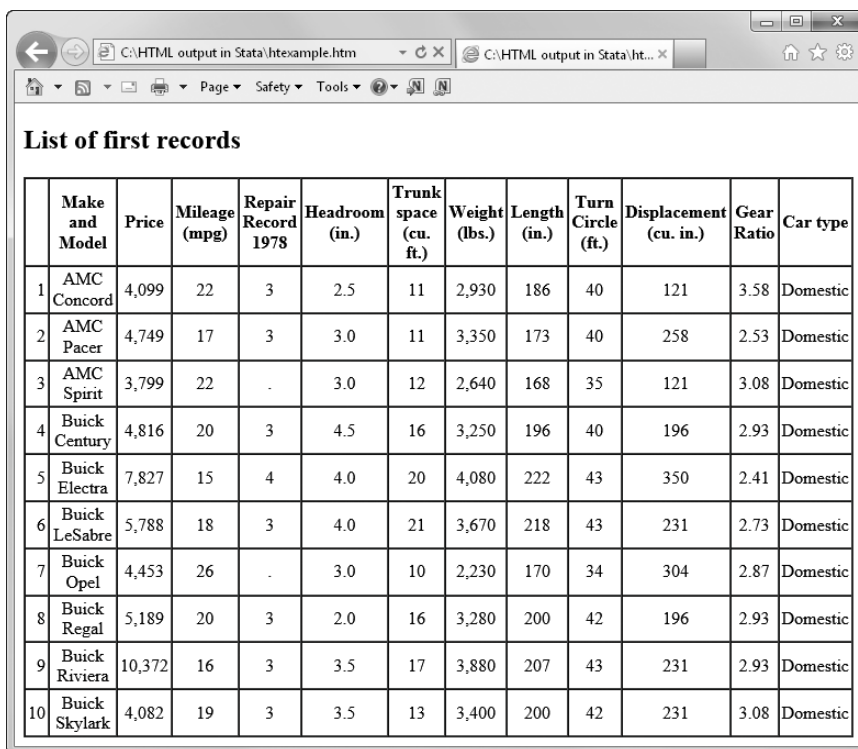
`table(string)` includes the HTML options specified in `string` in the table. The default is `table(BORDER=1 CELLSPACING=0 CELLPADDING=2)`.

`align(string)` specifies the alignment of the output. `align()` supersedes the alignment in the `nodisplay` option. `string` can be CENTER (the default), RIGHT, or LEFT.

Example 6

The following code produces a list of all variables for the first 10 observations of `auto.dta` and prints it in the HTML file:

```
htopen using htexample, replace
htput <h2> List of first records </h2>
sysuse auto
htlist in 1/10
htclose
```



The screenshot shows a web browser window with the title "List of first records". The browser's address bar shows the file path "C:\HTML output in Stata\htexample.htm". The table displayed has the following structure:

	Make and Model	Price	Mileage (mpg)	Repair Record 1978	Headroom (in.)	Trunk space (cu. ft.)	Weight (lbs.)	Length (in.)	Turn Circle (ft.)	Displacement (cu. in.)	Gear Ratio	Car type
1	AMC Concord	4,099	22	3	2.5	11	2,930	186	40	121	3.58	Domestic
2	AMC Pacer	4,749	17	3	3.0	11	3,350	173	40	258	2.53	Domestic
3	AMC Spirit	3,799	22	.	3.0	12	2,640	168	35	121	3.08	Domestic
4	Buick Century	4,816	20	3	4.5	16	3,250	196	40	196	2.93	Domestic
5	Buick Electra	7,827	15	4	4.0	20	4,080	222	43	350	2.41	Domestic
6	Buick LeSabre	5,788	18	3	4.0	21	3,670	218	43	231	2.73	Domestic
7	Buick Opel	4,453	26	.	3.0	10	2,230	170	34	304	2.87	Domestic
8	Buick Regal	5,189	20	3	2.0	16	3,280	200	42	196	2.93	Domestic
9	Buick Riviera	10,372	16	3	3.5	17	3,880	207	43	231	2.93	Domestic
10	Buick Skylark	4,082	19	3	3.5	13	3,400	200	42	231	3.08	Domestic

Although in example 5 we succeeded in making a table, we had to use a minimum of HTML syntax. We can imagine that the program to make more complicated tables would also be much more complicated. To facilitate this, we created another ado-file to make tables containing summary statistics.

htsummary syntax

```
htsummary varname1 [varname2] [if] [in] [, head close nototal freq row
rowtotal log format(string) median add(#) recode(rule) anova kw chi
exact test pval(real) method(string) missing color(real)]
```

htsummary description

htsummary makes a row-table of summary statistics for *varname1* in HTML format. Columns for the table can be specified by *varname2*. By default, mean and standard deviation (SD) are reported. Use options to choose appropriate summary statistics.

htsummary options

head displays the header of the table. **head** is only used for the first row of the table.

close ends the table. **close** is only used for the last row of the table.

nototal specifies not to include a column for totals. **nototal** is only used for the first row of the table.

freq requests calculation of frequencies and percentages. **freq** is suitable for discrete (or qualitative) variables. If **freq** is not specified, the default summary mean and SD will be reported.

row requests calculation of row percentages instead of the default column percentages. **row** is only allowed when the option **freq** is also specified.

rowtotal requests calculation of row totals for discrete variables. **rowtotal** is only allowed when the option **freq** is also specified.

log specifies that *varname1* be analyzed on a logarithmic scale (so calculate the geometric mean instead of the default arithmetic mean).

format(string) sets the format for *varname1*.

median calculates the median (centile 50) and interquartile range (centile 75–centile 25) instead of the default arithmetic mean and standard deviation.

add(#) adds # to *varname1*. **add()** is very useful for the logarithmic transformation of variables containing 0.

recode(rule) specifies that the variable be recoded to be analyzed according to *rule*; see [D] **recode**.

anova compares groups in columns by analysis of variance (Student's *t* if the number of columns is two).

kw compares groups in columns by Kruskal–Wallis (Wilcoxon if the number of columns is two).

`chi` compares groups in columns by chi-squared test. `chi` is only allowed when the option `freq` is also specified.

`exact` compares groups in columns by Fisher's exact test. `exact` is only allowed when the option `freq` is also specified.

`test` automatically determines the appropriate test for comparison. `test` will use ANOVA for means, Kruskal–Wallis for medians, a chi-squared test for frequencies when less than 20% of cells have an expected frequency ≤ 5 , and Fisher's exact test for frequencies when at least 20% of cells have an expected frequency ≤ 5 .

`pval(real)` displays *real* as a *p*-value. `pval()` is useful when the *p*-value comes from methods other than those performed by former options.

`method(string)` displays a footnote with *string* as the method for acquiring the *p*-value from the option `pval()`.

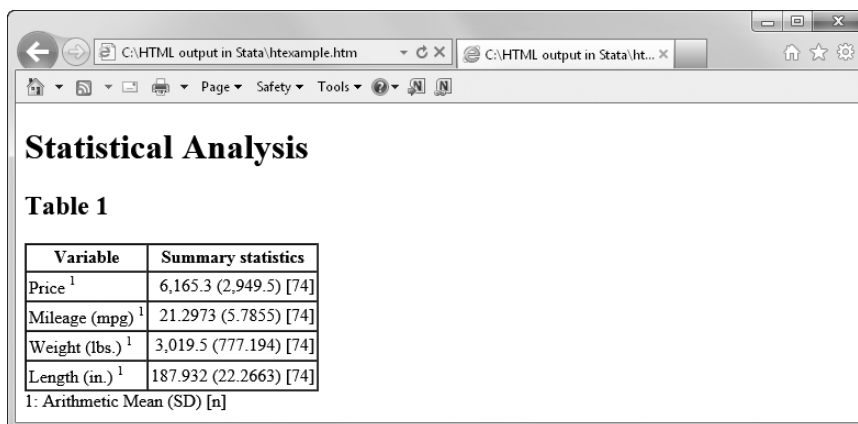
`missing` defines a category for missing data. This is only for descriptive analysis and not for comparison tests. `missing` is only allowed when the option `freq` is also specified.

`color(real)` changes the row-background color when *p*-value $<$ *real*.

Example 7

With `htsummary`, we can reproduce example 5 with a much simpler syntax:

```
htopen using htexample, replace
htput <h1> Statistical Analysis </h1>
sysuse auto
htput <h2> Table 1 </h2>
htsummary price, head
htsummary mpg
htsummary weight
htsummary length, close
htclose
```



Variable	Summary statistics
Price ¹	6,165.3 (2,949.5) [74]
Mileage (mpg) ¹	21.2973 (5.7855) [74]
Weight (lbs.) ¹	3,019.5 (777.194) [74]
Length (in.) ¹	187.932 (22.2663) [74]

1: Arithmetic Mean (SD) [n]

Example 8

htsummary has options to describe different statistics from two or more groups and to perform association tests.

```
htopen using htexample, replace
htput <h1> Statistical Analysis </h1>
sysuse auto
htput <h2> Table 1 </h2>
htsummary price foreign, head format(%8.2f) test
htsummary mpg foreign, format(%8.2f) test
/* generate a new categorical variable */
recode mpg (min/25 = 0 "Low/Medium") (25/max = 1 "High"), gen(mympg)
label var mympg "Mileage (level)"
htsummary mympg foreign, freq rowtotal row test
htsummary weight foreign, median format(%8.2f) test
htsummary length foreign, log format(%8.2f) test close
htclose
```

Statistical Analysis

Table 1

Variable	Car type			p-value
	Domestic	Foreign	Total	
Price ¹	6072.42 (3097.10) [52]	6384.68 (2621.92) [22]	6165.26 (2949.50) [74]	0.6802 ²
Mileage (mpg) ¹	19.83 (4.74) [52]	24.77 (6.61) [22]	21.30 (5.79) [74]	0.0005 ²
Mileage (level) ³	Low/Medium	15 (25%)	60 (100%)	0.1021 ⁴
	High	7 (50%)	14 (100%)	
	Total	52 (70%)	22 (30%)	
Weight (lbs.) ⁵	3360.00 (940.00) [52]	2180.00 (630.00) [22]	3190.00 (1360.00) [74]	< 0.0001 ⁶
Length (in.) ⁷	195.09 (20.61) [52]	168.01 (13.68) [22]	186.61 (22.42) [74]	< 0.0001 ²

1: Arithmetic Mean (SD) [n]
2: t-test
3: n (row percentage)
4: Fisher's exact test
5: Median (IQR) [n]
6: Wilcoxon Rank Sum test
7: Geometric Mean (SD) [n]

In the above code, there are some details to note:

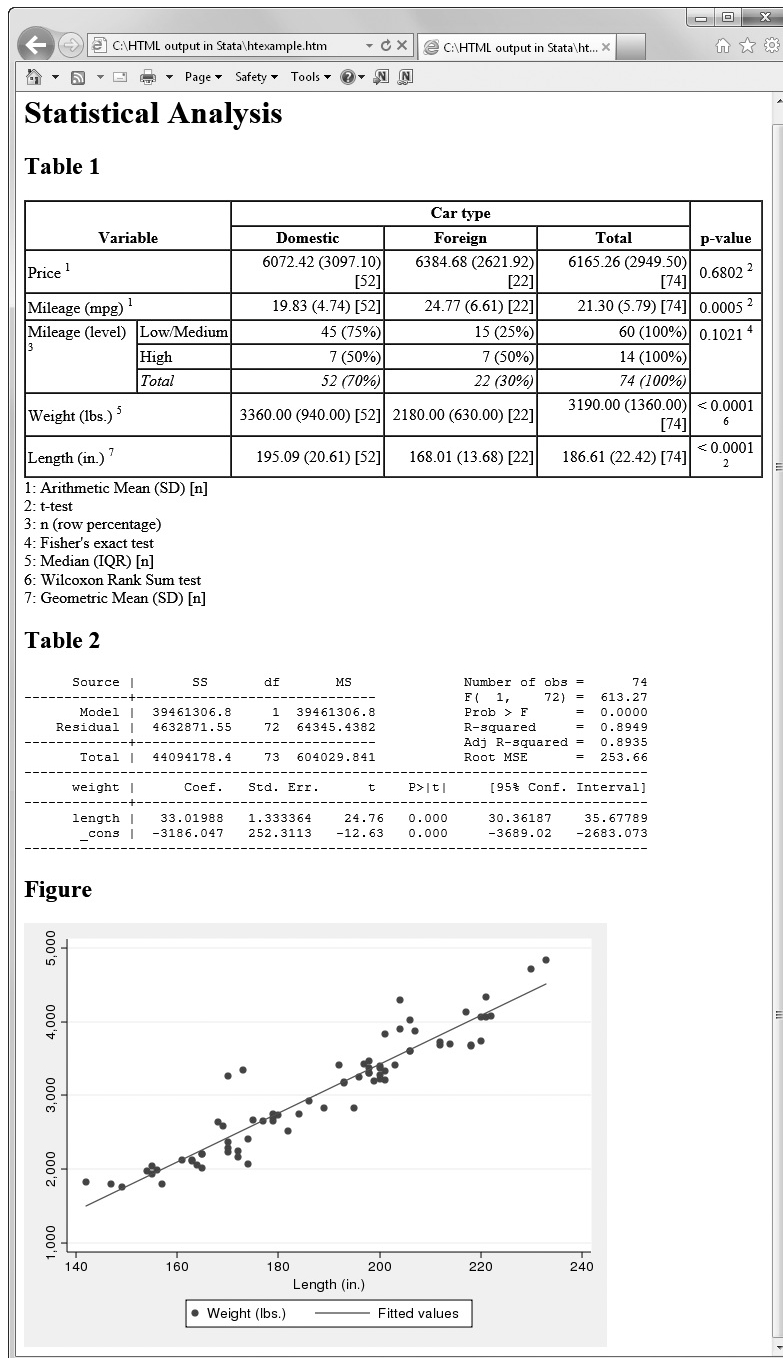
- `htsummary` produces rows of a table, so the first time it is used (first table row), it should contain the `head` option; the last time it is used, it should contain the `close` option.
- We can insert other Stata commands among the command lines that produce an HTML table provided that these commands do not send any output to the HTML file. In the example, we created the variable `mympg` before adding the frequencies to the table.
- The `test` option determines the appropriate association test for each variable, but we also have the options `anova`, `kw`, `chi`, and `exact` if we prefer to specify the test we want to perform.

Example 9

Remember that with `htlog`, we can send any result from Stata to the HTML page. To complete the example, we see how to append results from a linear regression and a chart.

```
htopen using htexample, append
htput <h2> Table 2 </h2>
htlog regress weight length
twoway (scatter weight length) (lfit weight length), name(htexample, replace)
graph export htexample.png, replace
htput <h2> Figure </h2>
htput 
htclose
```

The HTML page produced by examples 8 and 9 is as follows:



4 Technical notes

These ado-files need some information to work. Specifically, `htopen` opens an HTML file—the filename (and path) we need for the rest of the ado-files to use to send output. In the design of these programs, we tried to minimize syntax; for this reason, we decided to save the HTML file identification in a global macro called `HTfile`. The alternative would be to use an option to pass the name in the ado-file, but that complicates the syntax because we would have to repeat this option each time we send a new line to the HTML file.

`htsummary` needs to retain information regarding the table, and it is also saved in global macros: `HTsummary` indicates whether we are actually making a table, that is, whether we have executed the command with the option `head`; `HTptot` indicates whether the table has a column for totals; `HTcolor` indicates whether we specified the option `color()`; and `HTsup` and `HT#` are used for superscripts and footnotes.

We decided to use names starting with `HT` for these global macros so that we would have a common nomenclature for this set of programs, for others not presented in this article, and for those in the future. `htclose` removes all global macros whose names begin with `HT`; users should keep this in mind if they use global macros in their programs.

To avoid unexpected errors due to the above macros, you should, when writing do-files, enclose all code between `htopen` and the corresponding `htclose` in a `capture noisily` block; that is, each `htopen` statement should be immediately followed by the command

```
capture noisily {
```

and the corresponding `htclose` statement should be immediately preceded by the command

```
}
```

This practice ensures that if any command in the `capture noisily` block fails, then Stata will transfer control to the `htclose` command following the `capture noisily` block. The global macros will be cleared, and we can run the corrected program without failing because of the uncleared `HTfile` macro.

As mentioned before, we tried to design these ado-files to work in the easiest possible way: we only need to identify the name (and path) of the HTML file at the time it is opened by using `htopen`. However, the real procedure entails more complex actions not shown. The programs presented here use the command file to send text to the HTML file. In these commands, files are referred to by a file `handle`. When we open a file, we specify the file `handle` that we want to use. From that point on, we refer to the file by its `handle`. This means that what we really need to know is not the name of the HTML file but the name of the `handle` we used when it was opened.

We thought of several options to avoid engaging the user with the inner mechanisms of these ado-files: to use the filename (specified by `using filename`) as the name of

the `handle`, to use a default name for the `handle` with the possibility of changing it if required by the user, etc. Finally, we decided to use a `tempname` for the `handle`, which ensures that it does not conflict with the other `handle` that users may eventually use throughout their programs. We know that `handle` obtained from `tempname` is automatically closed when the ado-file terminates; therefore, we need to open it every time we send text to the HTML file. We admit that this is not the most efficient solution from an informatics point of view, but using temporary names for the file `handle` offers considerable advantages, including the following:

1. We must not forget that programs can be stopped because of errors or because the user presses Break. By using a `handle` obtained from `tempname`, we avoid possible errors due to the management of `handle` that users might not understand, because they could not be aware of using that `handle` or even the command file.
2. The contents of the files might not be fully written until the file is closed. By using a `handle` obtained from `tempname`, we can see the results in the HTML file while they are being written by `htput`, `htlog`, `htlist`, or `htsummary` because the file is automatically closed when the ado-file terminates.

5 Conclusion

We have presented the basic methodology to produce HTML files in Stata. HTML is a very rich and flexible language. The commands presented here allow the programmatic production of formatted output within a statistical analysis. These commands are just one example of the enormous potential of using HTML syntax to produce reports. From the initial idea of this work, custom styles using cascade style sheets and reports organized by menus are possible. This type of output increases productivity because there is no need to copy and paste from one software file to another: the statistical report can be produced entirely with Stata programs.

About the authors

Llorenç Quintó has been a medical statistician at the Barcelona Centre for International Health Research (CRESIB), Hospital Clínic—Universitat de Barcelona since 1994. His work involves statistical analyses and methodological support for scientists at CRESIB, Hospital Clínic and at Centro de Investigação em Saúde de Manhiça (CISM) in Mozambique, in research projects that are mainly focused on malaria, imported diseases, HIV/AIDS and sexually transmitted infections, and bacterial and viral infections. He also organizes and teaches courses in biostatistics with Stata that CRESIB offers in Barcelona and Mozambique.

Sergi Sanz has been a medical statistician at the Barcelona Centre for International Health Research (CRESIB), Hospital Clínic—Universitat de Barcelona since 2001. He coordinates projects and performs similar tasks as his colleagues at the Biostatistics Unit. Since 2009, he has worked as associate professor in the Department of Public Health at Universitat de Barcelona.

Elisa De Lazzari has been a medical statistician at the Barcelona Centre for International Health Research (CRESIB), Hospital Clínic—Universitat de Barcelona since 2011. She does

similar tasks as her colleagues Llorenç Quintó and Sergi Sanz. During the previous 10 years, her work in the Infectious Diseases Unit (Hospital Clínic) mostly involved statistical analyses of and methodological support for HIV/AIDS research projects.

John J. Aponte is an associate research professor at the Barcelona Centre for International Health Research (CRESIB), Hospital Clínic—Universitat de Barcelona. He is a medical statistician with a lot of experience in the design and analysis of clinical trials that evaluate malaria prevention tools. He collaborates actively with the Centro de Investigação em Saúde de Manhiça (CISM) in projects to evaluate the RTS,S malaria vaccine and Mefloquine as an alternative drug for IPTp to prevent malaria during pregnancy in malaria-endemic areas.