*An Introduction to Stata for Health Researchers:*

Information to users of Stata 14 to 15

*An Introduction to Stata for Health Researchers, 4th ed.* was written for Stata 13. We refer to this edition as ISHR4.

There may be some discrepancies between ISHR4 and the release of your Stata program, and in the following, we point to the most important changes, organized by the structure of the book. We are only pointing, and you will need Stata's help facilities to get more information. Keep your Stata program updated.

To see all changes, for example, from release 13 to 14, type (it works only in Stata 14+):

 **. help whatsnew13to14**

Stata introduced a new file format in release 14. See Stata 14 and Unicode for more information.

If you want to create a dataset that a previous Stata version can read, you may need to use the **saveold** command; see this table:

|  |  |  |
| --- | --- | --- |
| Your version of Stata: | **save** generates datathat can be read by | **saveold** generates datathat can be read by |
| Stata 14, 15 | Stata 14, 15 | Stata 11, 12, 13\* |
| Stata 13 | Stata 13 | Stata 11, 12 |
| From Stata 14, **saveold** has a **version()** option. But **saveold** does not translate from Unicode to ASCII characters. This is a problem with extended ASCII characters like ü, é, and ø.  |

Sometimes, the syntax of a command changes with a new release of Stata. For example, the syntax of the **ci** and **cii** commands changed in Stata 14, but version control still allows the use of Stata 13 syntax:

 **. version 13: ci stenosis , binomial**

Aarhus, July 2018

Svend Juul and Morten Frydenberg

Changes from Stata 14 to Stata 15

|  |  |  |
| --- | --- | --- |
| **Section** | **Stata 15** | **Stata 13** (as described in ISHR4) |
| 1.5 | The **putdocx** and **putpdf** commands let you generate formatted documents. | Output management. |
| 15.2 | Improved **power** command matches several analyses. | **power** command. |
| 16.7 | Graph elements can be made more or less transparent. See **. help colorstyle** | Stata graphs are opaque: a new element will overlay an already existing element, hiding the latter. |
| 16.12 | **.eps** does not support Unicode. Use **.svg** (Scalable Vector Graphics) if a graph includes characters beyond plain ASCII. Alternatives: **.pdf** and **.png**. | **.eps** (Encapsulated PostScript) is the recommended export format. |

Changes from Stata 13 to Stata 14

|  |  |  |
| --- | --- | --- |
| **Section** | **Stata 14** | **Stata 13** (as described in ISHR4) |
|  | All text is Unicode encoded (UTF-8). If you only use plain ASCII, you have no problem; if you use extended ASCII, moving datasets and other files between Stata versions requires translation (see below). | All text is plain ASCII (0-9, a-z, A-Z, and some symbols) but, with some languages, also extended ASCII (characters like à, ü, é, æ, ø, å, and ð).  |
| 1.5 | The **putexcel** command lets you generate Excel worksheets from your data.  | Output management |
| 5.5 | A number of new string functions matching Unicode are included. In Unicode, the number of bytes used may be larger than the number of characters displayed. To get the number of characters, use **ustrlen()**; to get the number of bytes, use **strlen()**. See **. help string functions**. | A string function like **strlen()** counts the number of characters that in ASCII is the same as the number of bytes.  |
| 6.1 | **saveold** can save datasets in version 12, or 13 file format; for version 12, type **. saveold filename , version(12)** You can specify option **version(11)** as well. This option saves data in version 12 format, which can be read by Stata 11. **saveold** does not perform any translation from Unicode to ASCII characters. | In Stata 13, **saveold** saves a Stata 12 dataset. |

|  |  |  |
| --- | --- | --- |
| **Section** | **Stata 14** | **Stata 13** (as described in ISHR4) |
| 11.6 | The syntax for the **ci** command changed, and the output improved. Same examples as in ISHR4:Means (normal distribution): **. ci means crea**Proportions (binomial distribution): **. ci proportions stenosis** Rates (Poisson distribution): **. ci means died , poisson exposure(pyrs)**Variances: **. ci variances crea** | **ci** command. |
| 11.7 | The syntax for the immediate **cii** command changed, and the output improved. To open the **cii** dialog box, type **. db cii** | Immediate **cii** command. |
| 15.2 | Power analysis for time-to-event data is included in the **power** command. | Use **stpower** for power analysis of time-to-event data. |
| 15.6 | New **icd10** commands for classification of diseases. | No command matches the ICD-10 classification of diseases. |
| 16.12 | New file format for graphs (**.gph**). Stata 14 will understand an old **.gph**, but Stata 13 will not understand a version 14 **.gph**, and there is no translation facility (at least the documentation does not mention the issue at all). If you follow our repeated recommen­dation to save a do-file rather than a **.gph** for each graph, you have only a trivial translation problem at most. |  |
| 16.12 | **.eps** does not support Unicode. Use **.pdf** or **.png** if a graph includes characters beyond plain ASCII. | **.eps** (Encapsulated PostScript) is the recommended export format. |

Stata 14 and Unicode

If you never use anything but plain ASCII (characters 0-9, a-z, A-Z, and some symbols), you may skip this section. You need no precautions because for these characters, ASCII and Unicode (UTF-8) share codes. Just go on as usual. This will typically be the case if the language used is English or another language with no letters beyond A-Z.

If you always or sometimes use a language that requires extended ASCII for characters like à, é, ü, æ, ø, å, č, þ, and ð, you may need to do something, especially if you exchange data and do-files with others who use a different Stata version. Here we assume that you just replaced an older version of Stata with version 14 or 15.

Prior to version 14, Stata used ASCII encoding of characters. The codes for plain ASCII are 0-127; for extended ASCII, they are 128-255. There are several extended ASCII encoding schemes, for example, **Latin 1** and **Windows-1252** for Western European languages, **Latin 2** for some Central and Eastern European languages, and **Latin 4** for the Cyrillic alphabet.[[1]](#footnote-1) Thus, the same extended ASCII code may display different characters dependent on the encoding scheme used when the file is displayed.

From version 14, Stata uses UTF-8 or Unicode (UTF-8) encoding of characters. This gives access to thousands of characters and symbols, including Arabic, Cyrillic, Chinese, and other alphabets. The Unicode *code point* is the number we use with **uchar(#)** and other functions, but behind it is a more complex encoding (UTF-8) where each character is defined by one to four bytes.[[2]](#footnote-2) For the characters 0-9, a-z, and A-Z represented in plain ASCII and in the **Latin 1** extended ASCII encoding, the Unicode *code point* is the same as the ASCII code.

**Converting existing Stata files to Unicode**

Once and for all: Make a backup copy of all of your Stata datasets, do-files, ado-files, and help files, and put the backup in a different place. Most likely, you will never need to touch these copies again; it is just a safeguard.

Before converting a file, Stata must "know" its encoding scheme. Specify the encoding scheme, for example, Windows-1252:

 **. unicode encoding set Windows-1252**

To examine the need for translation of Stata files (.dta, .do, .sthelp, etc.) in the current folder, type

 **. unicode analyze \***

In this context, \* means all Stata-related files. You may drop the analysis and go directly to translation by typing

 **. unicode translate \***

The **unicode translate** command overwrites the old version by the new version, but Stata stored a backup elsewhere, so you can restore the old version of the translated files by typing

 **. unicode restore \***

An unofficial substitute for **unicode analyze** is **whichencoding**. It gives a nice overview of the encoding of the datasets and do-files in the current directory. Install it with

 **. ssc install whichencoding**

The unofficial **ascii2unicode** command has some advantages compared with **unicode translate**. It stores two versions with different names, one with the old and one with the new encoding. Install it with

 **. ssc install ascii2unicode**

**Converting Unicode files back to ASCII**

A dataset generated by Stata 14 is in Unicode encoding from its birth, and if a file includes characters beyond plain ASCII, you cannot restore it correctly with an ASCII character set. However, you need to do that if you want to share your data with someone using Stata version 13 or earlier. The **saveold** command does not make the back-translation, and currently (August 2018), Stata has no official command that does it. However, the unofficial **unicode2ascii** command combines a translation with a **saveold** command. Install it with

 **. ssc install unicode2ascii**

The official **unicode convertfile** command translates text files back to ASCII – but not datasets. **unicode2ascii** translates text files but also files like do-files, ado-files, and sthlp-files back to ASCII.

**Other relevant commands**

The unofficial **asciiplot** command displays the codes for plain and extended ASCII characters. With Stata version <14, the encoding is determined by the setup of your computer. With Stata version 14+, **asciiplot** shows the Unicode code points, which are the same as the ASCII Latin 1 codes. Install **asciiplot** with

 **. ssc install asciiplot**

If you have Stata version 14+, the unofficial **grtext** command is helpful if you want to insert, for example, in graphs special characters or symbols beyond plain and extended ASCII. Install **grtext** with

 **. ssc install grtext**

1. The **Windows-1252** encoding scheme includes the same characters as **Latin 1** – and some more characters. [↑](#footnote-ref-1)
2. For example, the character **á** has the extended ASCII code **225** in the Latin 1 and Windows-1252 encodings. The Unicode *code point* is also **225**, and the **uchar(225)** function will generate this character. The ordinary user hardly needs to know that the UTF-8 encoding of **á** is **195 161** (or hexadecimal **C3 A1**). [↑](#footnote-ref-2)