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.

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

http://www.stata-journal.com

The *Stata Journal* is indexed and abstracted in the following:

- CompuMath Citation Index®
- Current Contents/Social and Behavioral Sciences®
- RePEc: Research Papers in Economics
- Science Citation Index Expanded (also known as SciSearch®)
- Scopus™
- Social Sciences Citation Index®

The *Stata Journal*, electronic version (ISSN 1536-8734) is a publication of Stata Press. Stata, Mata, NetCourse, and Stata Press are registered trademarks of StataCorp LP.

The Stata Journal (2011)
11, Number 4, pp. 589–604

Managing the U.S. Census 2000 and World Development Indicators databases for statistical analysis in Stata

P. Wilner Jeanty
The Kinder Institute for Urban Research
Hobby Center for the Study of Texas
Rice University
Houston, TX
pwjeanty@rice.edu

Abstract. This article introduces a new Stata command, `labcenswdi`,[1] to automatically manage databases that provide variable descriptions on the second row in a dataset. While renaming all variables and converting them from string to numeric, `labcenswdi` automatically manages the variable descriptions including removing them from the second row to place them into Stata variable labels and saving them to a text file. The process yields a dataset ready for statistical analysis. I illustrate how this command can be used to efficiently manage datasets obtained from the U.S. Census 2000 and the World Development Indicators databases.

Keywords: dm0060, labcenswdi, U.S. Census 2000, World Development Indicators, databases, data management, panel data

1 Introduction

Researchers in nearly all fields use data emanating from the U.S. Census Bureau's Census 2000 Summary Files and the World Bank's World Development Indicators (WDI) database. One of the most apparent and appealing aspects of these databases is that the data come with variable descriptions. However, there are data-management impediments associated with using datasets from these databases for statistical analysis.

For example, consider the U.S. Census 2000 Summary Files. One problem arising in a dataset from this database is that the default variable names are on the first row and the variable descriptions or definitions are on the second row. As a result, when the data are imported into Stata, all numeric variables are treated as string, precluding any numerical calculations. Two commonly used official Stata commands come close in managing a dataset in which the first few lines contain headers or other markers: `infix` and `infile` (which require writing a dictionary file). However, the variable delimiter in datasets extracted from the U.S. Census 2000 Summary Files does not lend itself to the treatment of these two Stata commands.

1. `labcenswdi` was written when the author was a research economist in the Department of Agricultural, Environmental, and Development Economics at Ohio State University.

Before deleting the second row, users would want to attach the variable descriptions to the variables as Stata labels and eventually save them in a text file for reference. Yet any variable description more than 80 characters long will be truncated when attached to a variable as a label, causing loss of information. Thus managing the variable descriptions is key to prevent Stata from truncating them in an undesirable way. Another predicament is that the default variable names provided with the data are undecipherable once the second row is removed. For example, variable names such as `P019001`, `P019002`, and `P019003` in a Census 2000 dataset are meaningless. Consequently, users would undoubtedly want to rename the variables with more meaningful names.

Similarly, the World Bank's WDI database has recently and interestingly allowed its users to download long-form datasets with the series descriptors on the second row.[2] To surmount the challenges involving the management of WDI datasets, Jeanty (2010) and Baum and Cox (2007) are the first to provide adequate and efficient solutions. However, to obtain a dataset amenable to immediate panel-data analysis, both articles have focused on reshaping the WDI database from wide to long form, largely because reshaping WDI from wide to long form has been a prominent issue for the WDI and Stata users.

The advantage of having long-form WDI data at one's disposal remains largely unexploited. When the desired data structure is one that is needed for panel-data modeling, downloading and using long-form WDI datasets is more appealing and more efficient. The problem with the World Bank's delivered panel-data structure is that it presents the same predicaments encountered with data emanating from the U.S. Census 2000 Summary Files. In particular, data-management efforts required to handle the WDI missing-value symbols, the double dots (`..`), and the presence of the series descriptors on the second row make using this long-form structure inconvenient, tedious, and unattractive even for the more discerning Stata users.

In this article, I present a new Stata command, `labcenswdi`, that automatically completes these thorny data-management tasks required to make a U.S. Census 2000 dataset or a long-form WDI dataset ready for statistical analysis. Performed all at once, these tasks include, but are not limited to, variable renaming and conversion from string to numeric, removal of the variable descriptions from the second row to place them into Stata variable labels, and saving the variable descriptions to a text file.

The `labcenswdi` command is described in the next section. Section 3 illustrates `labcenswdi` on a Census 2000 dataset. Section 4 shows an example of data management using a long-form WDI dataset. Finally, section 5 concludes the article.

2. For more details on the WDI database, see Jeanty (2010) and the World Bank Group (2011).

2 The labcenswdi command

2.1 Description

`labcenswdi` automatically manages datasets obtained from databases that provide variable descriptions on the second row. Such databases include the U.S. Census 2000 Summary Files, the American Community Survey, and the WDI. While renaming variables with the user's specified variable names, `labcenswdi` manages the variable descriptions by removing them from the second row to place them into Stata variable labels, reducing their length to 80 characters or fewer, and saving them to a text file. The new variable names are supplied in *newvarlist* (see syntax below) if the user elects to replace the default variable names with more meaningful names.

When a dataset containing variable descriptions on the second row is imported into Stata using the `insheet` command, Stata understandably reads in all variables as string regardless of their content (string or numeric). `labcenswdi` will automatically attempt to convert all variables containing numerical content back to numeric. However, unless explicitly requested by the user, no conversion will take place for string variables containing nonnumeric characters such as 1000-separator commas.

Demoting the variables may also conserve memory. `labcenswdi` automatically attempts to demote both string and numeric variables. For example, storing an integer variable as *double* or storing a string variable having maximum length of five characters as a `str15` would be a waste of memory (see [D] **compress** and [D] **data types**). Demotions are accomplished by default when you type `labcenswdi` with option `nstr()`. But typed by itself, `labcenswdi` will display the default variable names along with the variable descriptions. As shown later, displaying the variable descriptions prior to managing the data presents several advantages.

2.2 Syntax

The syntax to display the variable descriptions and the default variable names is

```
labcenswdi
```

The syntax to manage the data and the variable descriptions is

```
labcenswdi [newvarlist], nstr(#) [truncby(("text1") [("text2")])
   truncwith(("text3") [("text4")]) repdes((# "text5") [(# "text6") ...])
   force comma saving(filename[, replace])]
```

2.3 Options

`nstr(#)` specifies the number of identifier (string) variables present in the dataset. These variables are assumed to be at the beginning of the dataset and will not be converted from string to numeric even if they have numeric content. `nstr()` is required.

`truncby(("`*text1*`") [ ("`*text2*`") ])` specifies the set or sets of characters by which the variable definitions should be truncated. This is important because Stata will unconditionally truncate all labels with length greater than 80 characters. Up to two sets of characters can be specified (see examples). Conspicuously, the two sets of characters must be different. If a set of characters contains space, surround it with quotes.

`truncwith(("`*text3*`") [ ("`*text4*`") ])` specifies the sets of characters with which "*text1*" and "*text2*" are to be replaced. If `truncwith()` is not specified, then `truncby()` returns the variable descriptions without "*text1*" and "*text2*".

`repdes((#  "`*text5*`") [ (#  "`*text6*`")  ... ])` specifies a list of original variable descriptions to be replaced with user-defined variable descriptions. Here # corresponds to the #th variable description to be replaced, and "*text5*", "*text6*", ... are the texts to replace. For example, specifying `repdes((1 "Workers 16 and plus in Agr. Sector"))` will replace the first variable description with "Workers 16 and plus in Agr. Sector". Prior to specifying this option, users are encouraged to use the first syntax to decipher the order of the original variable descriptions in a dataset. If more than one variable definition needs to be replaced, their corresponding order numbers should be specified in ascending order (see the examples in sections 3 and 4).

`repdes()` supersedes `truncby()` if applied to the same variable descriptions.

`force` specifies that nonnumeric character values of numeric variables be converted to missing values. If the numerical variables in your dataset include nonnumeric characters, such as "(D)", "NA", "-", "..", or "ND", you should specify the `force` option to replace them with missing values.

`comma` specifies that 1000-separator commas be removed from numbers displaying them. You need to specify this option if one or more variables include values with 1000-separator commas; otherwise, those variables will not be converted from string to numeric.

`saving(`*filename*`[ , replace ])` specifies that the original variable descriptions be saved to the text file *filename*. The optional `replace` specifies that *filename* be overwritten if it exists. When `saving()` is specified, at the end of the process, the filename (path included) is displayed as a link that shows the file contents when clicked on.

3 Using labcenswdi on a U.S. Census 2000 dataset

In this example, I use a dataset extracted from U.S. Census Bureau's the Census 2000 Summary File 1 database; see the appendix for more details on how to download data for all U.S. counties (U.S. Census Bureau 2000). This database includes data on people's ages, sexes, and races, their family and household groups, and whether their home is owned or rented. The American FactFinder website can be used to access large amounts of data, although large bandwidth and storage may be required.

3.1 Importing U.S. Census 2000 data into Stata

When you extract data from the U.S. Census Bureau's Census 2000 Summary File 1 database, you will be prompted to save a zip file on your computer. The zip file contains three text files. The first file is named `dc_dec_2000_sf1_u_data1.txt` (the only file you need) and it contains the data. The second file contains the names of the places, in our case, the counties. The third text file is a read-me file. The zip file must be uncompressed using special software packages such as the shareware Winzip or the freeware 7-zip. The data file can be directly loaded in Stata using the `insheet` command (see [D] **insheet**) by typing[3]

```
. insheet using dc_dec_2000_sf1_u_data1.txt, names clear delimiter(|)
(23 vars, 3220 obs)
```

One problem with importing the data into Stata is that you may have more variables than allowed by your Stata flavor. Another potential issue is that your dataset may require more memory than your current memory settings allow, because unlike SPSS and SAS, Stata stores the entire dataset in RAM.[4]

3.2 Managing U.S. Census 2000 data

After the data are loaded with `insheet`, `labcenswdi` can be used for data management. I use `labcenswdi` alone to display the list of variables in the dataset with their definitions. Viewing this list allows you to 1) decipher the correct number and order of the variables in the dataset, 2) appropriately choose variable names befitting the variable definitions if you elect to rename the default variables, and 3) identify which variable descriptions are potentially problematic to be used as Stata variable labels.

3. Note that the delimiter in datasets from these databases is the vertical bar or pipe (|). If you want to open the data file in Microsoft Excel, then in step 1 of the Text Import Wizard's three steps, select the file type "Delimited". In step 2, check the boxes in front of "Tab" and "Other", and in the box to the right of "Other", type the vertical bar (|). In step 3, select which variables you want to be treated as string (text) or numeric (general).
4. Those running Stata 12 should be less concerned about this issue. Stata's memory management is now completely automatic unless you are using the Linux operating system in which case some memory management may be required.

```
. labcenswdi
 The current dataset contains 23 variables defined as follows:
1) geo_id: Geography Identifier
2) geo_id2: Geography Identifier
3) sumlevel: Geographic Summary Level
4) geo_name: Geography
5) p019001: Households: Total
6) p019002: Households:  Households with one or more people under 18 years
7) p019003: Households:  Households with one or more people under 18 years;
> Family households
8) p019004: Households:  Households with one or more people under 18 years;
> Family households; Married-couple family
9) p019005: Households:  Households with one or more people under 18 years;
> Family households; Other family
10) p019006: Households:  Households with one or more people under 18 years;
> Family households; Other family; Male householder; no wife present
11) p019007: Households:  Households with one or more people under 18 years;
> Family households; Other family; Female householder; no husband present
12) p019008: Households:  Households with one or more people under 18 years;
> Nonfamily households
13) p019009: Households:  Households with one or more people under 18 years;
> Nonfamily households; Male householder
14) p019010: Households:  Households with one or more people under 18 years;
> Nonfamily households; Female householder
15) p019011: Households:  Households with no people under 18 years
16) p019012: Households:  Households with no people under 18 years; Family
> households
17) p019013: Households:  Households with no people under 18 years; Family
> households; Married-couple family
18) p019014: Households:  Households with no people under 18 years; Family
> households; Other family
19) p019015: Households:  Households with no people under 18 years; Family
> households; Other family; Male householder; no wife present
20) p019016: Households:  Households with no people under 18 years; Family
> households; Other family; Female householder; no husband present
21) p019017: Households:  Households with no people under 18 years; Nonfamily
> households
22) p019018: Households:  Households with no people under 18 years; Nonfamily
> households; Male householder
23) p019019: Households:  Households with no people under 18 years; Nonfamily
> households; Female householder
```

I list a few observations on a few variables to give you a sense of what the data look like.

```
. list geo_id2 sumlevel p019001 in 1/5
```

	geo_id2	sumlevel	p019001
1.	Geography Identifier	Geographic Summary Level	Households: Total
2.	01001	050	16003
3.	01003	050	55336
4.	01005	050	10409
5.	01007	050	7421

As you can see, the U.S. Census Bureau delivers the data with the variable definitions in the second row. As remarked in the *Introduction*, Stata has a command to handle this issue, but as you will see, it cannot manage the variable definitions before attaching them as variable labels.

The identifier variable `geo_id` includes values such as 05000US01001, 05000US01003, and so on, and the variable `sumlevel` includes values that do not vary across counties. Because of this, these variables are not germane to any analysis. Thus I drop them before proceeding with the data management. But I keep `geo_id2`, the variable holding the county FIPS code, because it is essential for combining datasets with the `merge` command (see [D] **merge**). I must be mindful of that when specifying the `nstr()` option to manage the data, because the dataset now contains only two identifier variables.

```
. drop geo_id sumlevel
```

Because I dropped two variables, I now run `labcenswdi` alone one more time to check the new order of the variables in the dataset. This is heartily recommended.

```
. labcenswdi
 The current dataset contains 21 variables defined as follows:
1) geo_id2: Geography Identifier
2) geo_name: Geography
3) p019001: Households: Total
4) p019002: Households:  Households with one or more people under 18 years
5) p019003: Households:  Households with one or more people under 18 years;
> Family households
6) p019004: Households:  Households with one or more people under 18 years;
> Family households; Married-couple family
7) p019005: Households:  Households with one or more people under 18 years;
> Family households; Other family
8) p019006: Households:  Households with one or more people under 18 years;
> Family households; Other family; Male householder; no wife present
9) p019007: Households:  Households with one or more people under 18 years;
> Family households; Other family; Female householder; no husband present
10) p019008: Households:  Households with one or more people under 18 years;
> Nonfamily households
11) p019009: Households:  Households with one or more people under 18 years;
> Nonfamily households; Male householder
12) p019010: Households:  Households with one or more people under 18 years;
> Nonfamily households; Female householder
13) p019011: Households:  Households with no people under 18 years
14) p019012: Households:  Households with no people under 18 years; Family
> households
15) p019013: Households:  Households with no people under 18 years; Family
> households; Married-couple family
16) p019014: Households:  Households with no people under 18 years; Family
> households; Other family
17) p019015: Households:  Households with no people under 18 years; Family
> households; Other family; Male householder; no wife present
18) p019016: Households:  Households with no people under 18 years; Family
> households; Other family; Female householder; no husband present
19) p019017: Households:  Households with no people under 18 years; Nonfamily
> households
20) p019018: Households:  Households with no people under 18 years; Nonfamily
> households; Male householder
21) p019019: Households:  Households with no people under 18 years; Nonfamily
> households; Female householder
```

Variable descriptions exceeding 80 characters will be truncated when attached to variables as labels. As a result, you will need to find an effective way to truncate the variable descriptions without losing information. We will replace the set of characters `"Households: Households with one or more people under 18 years"` with `"HH w/ 1+ person <18"`, and we will replace `"Households: Households with no people under 18 years"` with `"HH w/o people <18"`. However, four variable descriptions—the 8th, 9th, 17th, and 18th—are so long that the chosen strategy is not good enough. Thus these long variable descriptions have to be replaced. This is easily accomplished using the `repdes()` option. To be more meaningful, the first and second variable descriptions are also replaced with new ones.

To avoid losing information about the original variable descriptions, you can save them to a text file for reference by using the `saving()` option. An alternative is to keep the original data file in a safe place. You can open and view the content of the text file by simply clicking on the link provided by `labcenswdi` after completing the task. When the text file is opened, Stata will display the order of the variables, the variable types, the variable names,[5] and the variable definitions. Recall that we must instruct `labcenswdi` that we now have only two identifier variables in the datasets by specifying `nstr(2)`. Note that I chose more meaningful names for the variables, including `fips` to hold the county FIPS codes and `county` to hold the county names.

```
. labcenswdi fips county tot_hhs hhu18 fhhu18 fhhu18m fhhu18o fhhu18om fhhu18of
> nfhhu18 nfhhu18m nfhhu18f hhn18 fhhn18 fhhn18m fhhn18o fhhn18om fhhn18of
> nfhhn18 nfhhn18m nfhhn18f, nstr(2) saving(fem_hh)
> truncby(("Households:  Households with one or more people under 18 years")
> ("Households:  Households with no people under 18 years"))
> truncwith(("HH w/ 1+ person <18") ("HH w/o people <18"))
> repdes((1 "County Fips Code") (2 "County Name")
> (8 "HH w/ 1+ person <18; Family HH; Other family; Male householder")
> (9 "HH w/ 1+ person <18; Family HH; Other family; Female householder")
> (17 "HH w/o people <18; Family HH; Other family; Male HHer no wife present")
> (18 "HH w/o people <18; Family HH; Other family;
> Female HHer; no husband present"))
 Labeling, renaming, and/or conversion of variables done successfully
Note: For future reference, original variable descriptions saved to text file:
> C:\data/fem_hh.txt
```

As you can see, `Stata` responds with a message indicating that the task is complete and that the original variable definitions have been saved to the text file that I specified with the `saving()` option.[6] To give you a sense of what `labcenswdi` has done, I `describe` the data, which are now ready for statistical analysis.

5. The default variable names will not be displayed if you renamed the variables.
6. To save on space, the text file content is not shown.

```
. describe
Contains data
  obs:         3,219
 vars:            21
 size:       405,594
```

variable name	storage type	display format	value label	variable label
fips	str5	%9s		County Fips Code
county	str51	%51s		County Name
tot_hhs	long	%10.0g		Households: Total
hhu18	long	%10.0g		HH w/ 1+ person <18
fhhu18	long	%10.0g		HH w/ 1+ person <18; Family households
fhhu18m	long	%10.0g		HH w/ 1+ person <18; Family households; Married-couple family
fhhu18o	long	%10.0g		HH w/ 1+ person <18; Family households; Other family
fhhu18om	long	%10.0g		HH w/ 1+ person <18; Family HH; Other family; Male householder
fhhu18of	long	%10.0g		HH w/ 1+ person <18; Family HH; Other family; Female householder
nfhhu18	int	%10.0g		HH w/ 1+ person <18; Nonfamily households
nfhhu18m	int	%10.0g		HH w/ 1+ person <18; Nonfamily households; Male householder
nfhhu18f	int	%10.0g		HH w/ 1+ person <18; Nonfamily households; Female householder
hhn18	long	%10.0g		HH w/o people <18
fhhn18	long	%10.0g		HH w/o people <18; Family households
fhhn18m	long	%10.0g		HH w/o people <18; Family households; Married-couple family
fhhn18o	long	%10.0g		HH w/o people <18; Family households; Other family
fhhn18om	long	%10.0g		HH w/o people <18; Family HH; Other family; Male HHer no wife present
fhhn18of	long	%10.0g		HH w/o people <18; Family HH; Other family; Female HHer; no husband present
nfhhn18	long	%10.0g		HH w/o people <18; Nonfamily households
nfhhn18m	long	%10.0g		HH w/o people <18; Nonfamily households; Male householder
nfhhn18f	long	%10.0g		HH w/o people <18; Nonfamily households; Female householder

```
Sorted by:
     Note:  dataset has changed since last saved
```

A second look is provided by listing the first few observations on a few variables:

```
. list fips tot_hhs nfhhn18m nfhhn18f in 1/10
```

	fips	tot_hhs	nfhhn18m	nfhhn18f
1.	01001	16003	1604	1988
2.	01003	55336	6771	8124
3.	01005	10409	1262	1722
4.	01007	7421	868	956
5.	01009	19265	1988	2416
6.	01011	3986	553	691
7.	01013	8398	1050	1453
8.	01015	45307	5901	7954
9.	01017	14522	1788	2481
10.	01019	9719	1127	1372

If you are less concerned about converting the variable descriptions into Stata variable labels and the provided default variable names are appealing to you, then your best alternative is the official Stata command `destring`. First, load the data:

```
. insheet using dc_dec_2000_sf1_u_data1.txt, names clear delimiter(|)
(23 vars, 3220 obs)
```

Second, remove the row with the variable definitions by typing

```
. drop if _n == 1
(1 observation deleted)
```

Now run the `destring` command:

```
// because all the variables with numerical contents start with p
. destring p*, replace
```

(*output omitted*)

In other instances, you would type the first and the last variables with numerical contents separated by a dash (see the next example), assuming there are no intervening string variables.

Remember to use options `force` and `ignore()` when there are variables containing nonnumeric characters (see [D] **destring**).

4 Using labcenswdi on a WDI dataset

4.1 Downloading and importing long-form WDI data

To run `labcenswdi` on a WDI dataset to be used for panel-data analysis, you must select countries or time in rows and series in columns when downloading the dataset from the World Bank's website. For this example, I downloaded `wdi_time_series.csv`, a dataset with time in rows and series in columns. To import the data into Stata, I type

```
. insheet using wdi_time_series.csv, names clear
(16 vars, 4095 obs)
```

Before managing these data, I list all the variables along with their descriptions by typing `labcenswdi`.

```
. labcenswdi
 The current dataset contains 16 variables defined as follows:
1) countryname:
2) countrycode:
3) timename: .
4) aglndtraczs: Agricultural machinery, tractors per 100 sq. km of arable land
5) agconfertzs: Fertilizer consumption (100 grams per hectare of arable land)
6) nygdpmktpkd: GDP (constant 2000 US$)
7) nygdpmktpcd: GDP (current US$)
8) nygdppcapkdzg: GDP per capita growth (annual %)
9) aglndagrizs: Agricultural land (% of land area)
10) aglndirigzs: Irrigated land (% of cropland)
11) aglndcropzs: Permanent cropland (% of land area)
12) enpopdnst: Population density (people per sq. km)
13) sppopgrow: Population growth (annual %)
14) enrurdnst: Rural population density (rural population per sq. km of arable
> land)
15) netrdgnfszs: Trade (% of GDP)
16) spurbgrow: Urban population growth (annual %)
```

Notice that descriptions were not provided for the first three variables. Also, in contrast to the U.S. Census variable descriptions, there is no risk of having truncated variable labels in this example. Next I list some observations on a few variables to show you what the data structure looks like.

```
. list countryname countrycode timename aglndtraczs in 1/10, string(20)
```

	countryname	count~de	timename	aglndtraczs
1.			.	Agricultural machine..
2.	Afghanistan	AFG	1961	0.156862745
3.	Afghanistan	AFG	1962	0.194805195
4.	Afghanistan	AFG	1963	0.258064516
5.	Afghanistan	AFG	1964	0.256410256
6.	Afghanistan	AFG	1965	0.384615385
7.	Afghanistan	AFG	1966	0.510529675
8.	Afghanistan	AFG	1967	0.637429883
9.	Afghanistan	AFG	1968	0.637429883
10.	Afghanistan	AFG	1969	0.699745547

4.2 Managing the data

The WDI default variable names make as little sense as those of the U.S. Census 2000 Summary File 1, so users will want to replace them with names of their own choosing. Knowing that the dataset contains 16 variables and given their order of appearance, I choose 16 variable names that befit the variable descriptions. I need to specify the `force` option to remove the WDI missing-value symbols, the double dots (`..`). When a WDI dataset extracted in long form is read into Stata with the `insheet` command, the presence of the double dots and the variable descriptions on the second row will cause `labcenswdi` to treat every variable as string, even those with numerical contents. In this context, the `force` option provides an alternative to the official Stata command `destring` and its options `replace` and `force`.

The `repdes()` option is used to provide descriptions to three variables (`countryname`, `countrycode`, and `timename`) with no descriptions. Note that descriptions provided with the `repdes()` option will be used as final labels to the associated variables. Considering the first three variables as identifiers, I specify `nstr(3)`. But that does not make the variable holding the years a string variable. In fact, it is already read as numeric.

```
. labcenswdi country code year tractsk fertilha gdpcnst gdpcur gdppg agland
> irrigpct croplnd popdens popg ruraldens trade urbpg, nstr(3) force
> repdes((1 "Country Names") (2 "Country Code") (3 "Year"))
 Labeling, renaming, and/or conversion of variables done successfully
```

I now describe the data to show you that the missing labels have been added and the string variables with numerical contents have been converted from string to numeric.

```
. describe
Contains data
  obs:         4,094
 vars:            16
 size:       544,502
```

variable name	storage type	display format	value label	variable label
country	str24	%24s		Country Names
code	str3	%9s		Country Code
year	int	%8.0g		Year
tractsk	double	%10.0g		Agricultural machinery, tractors per 100 sq. km of arable land
fertilha	double	%10.0g		Fertilizer consumption (100 grams per hectare of arable land)
gdpcnst	double	%10.0g		GDP (constant 2000 US$)
gdpcur	double	%10.0g		GDP (current US$)
gdppg	double	%10.0g		GDP per capita growth (annual %)
agland	double	%10.0g		Agricultural land (% of land area)
irrigpct	double	%10.0g		Irrigated land (% of cropland)
croplnd	double	%10.0g		Permanent cropland (% of land area)
popdens	double	%10.0g		Population density (people per sq. km)
popg	double	%10.0g		Population growth (annual %)
ruraldens	double	%10.0g		Rural population density (rural population per sq. km of arable land)
trade	double	%10.0g		Trade (% of GDP)
urbpg	double	%10.0g		Urban population growth (annual %)

```
Sorted by:
     Note:  dataset has changed since last saved
```

Using a few variables, I now show that the dataset is ready for panel-data analysis by fitting a fixed-effects model. But first, I must `xtset` the data before using any `xt` commands. Specifying `xtset` on the data requires a numeric identifier variable to identify the panels (see [XT] **xtset**). This is done using the Stata command `egen` and its `group()` function.

```
. egen cid = group(country)
```

You can also type

```
. encode country, gen(cid)
```

to create a value label and label the `cid` variable so that when listed, it actually displays the countries.

```
. xtset cid year
       panel variable:  cid (strongly balanced)
        time variable:  year, 1961 to 2006
                delta:  1 unit

. xtreg gdppg tractsk fertilha trade croplnd popg, fe
Fixed-effects (within) regression               Number of obs      =      3014
Group variable: cid                             Number of groups   =        86

R-sq:  within  = 0.0109                         Obs per group: min =         5
       between = 0.0121                                        avg =      35.0
       overall = 0.0002                                        max =        42

                                                F(5,2923)          =      6.45
corr(u_i, Xb)  = -0.7376                        Prob > F           =    0.0000

------------------------------------------------------------------------------
       gdppg |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
     tractsk |  -.0070858   .0018163    -3.90   0.000    -.0106471   -.0035245
    fertilha |   .0002492   .0002118     1.18   0.239    -.0001661    .0006645
       trade |   .0155643   .0066784     2.33   0.020     .0024694    .0286593
     croplnd |  -.4745714   .1329247    -3.57   0.000    -.7352069   -.2139358
       popg |  -.3488353    .134395    -2.60   0.009    -.6123539   -.0853168
       _cons |   2.990199   .5713367     5.23   0.000     1.869936    4.110462
-------------+----------------------------------------------------------------
     sigma_u |  2.8434961
     sigma_e |  5.2993572
         rho |  .22354908   (fraction of variance due to u_i)
------------------------------------------------------------------------------
F test that all u_i=0:     F(85, 2923) =     3.69            Prob > F = 0.0000
```

As before, if you are not interested in labeling the variables with the variable descriptions and you want to keep the default variable names, your first option is the official Stata `destring` command:

```
. insheet using wdi_time_series.csv, names clear
(16 vars, 4095 obs)
. drop if _n == 1
(1 observation deleted)
. destring aglndtraczs-spurbgrow, replace force
  (output omitted)
```

Because the WDI data file is comma-delimited, your second option is to write a dictionary file to use with the `infile` command to apply the variable descriptions as Stata labels. However, this option is also unappealing when compared with `labcenswdi`. Not only does `labcenswdi` do all the work for you, but also you would run into the same label-length problem described herein.

5 Conclusion

In this article, I introduced a new user-written Stata command, `labcenswdi`, to automatically rename variables, convert string variables with numerical content into their numeric equivalents, and manage variable definitions, including removing them from the second row to attach them as labels to variables. I showcased `labcenswdi` using

a dataset from the U.S. Census 2000 Summary File and a WDI dataset extracted in long form. As illustrated, `labcenswdi` provides U.S. Census 2000 data users with much flexibility in handling variable descriptions. The command enables WDI users and Stata users to take advantage of the already available long-form data structure. When the desired data structure is that of a panel, there is no need to download wide-form WDI datasets to be reshaped to long form.

More importantly, `labcenswdi` works not only on datasets from the U.S. Census 2000 and WDI databases but also on datasets emanating from any database delivering variable definitions on the second row. One example that is not demonstrated here is the U.S. Census Bureau's American Community Survey database. It is worth noting that when managing a WDI dataset, `labcenswdi` supplements rather than replaces the user-written command `wdireshape` (Jeanty 2010). The use of the former is much more efficient when obtaining a panel-data setting is the ultimate goal.

Because `labcenswdi` was written prior to the release of the U.S. Census 2010, an updated version to automatically manage data from the U.S. Census 2010 is under way.

6 Acknowledgments

I thank the students in the Department of Agricultural, Environmental, and Development Economics at Ohio State University whose questions led to the development of `labcenswdi`.

7 References

Baum, C. F., and N. J. Cox. 2007. Stata tip 45: Getting those data into shape. *Stata Journal* 7: 268–271.

Jeanty, P. W. 2010. Using the world development indicators database for statistical analysis in Stata. *Stata Journal* 10: 30–45.

The World Bank Group. 2011. World Development Indicators (WDI) Online. http://data.worldbank.org/data-catalog/world-development-indicators.

U.S. Census Bureau. 2000. American FactFinder. http://factfinder.census.gov.

About the author

P. Wilner Jeanty is a research scientist at the Kinder Institute for Urban Research and the Hobby Center for the Study of Texas at Rice University in Houston, Texas.

A Steps to downloading data from the U.S. Census 2000 Summary Files

To download for all U.S. counties data from the U.S. Census 2000 Summary Files, follow these steps:

1. Point your browser to http://factfinder2.census.gov.
2. Under *Your Selections* on the left-hand side, expand the *Topics* menu and then
   a. expand "Year" and click on *2000*;
   b. expand "Dataset" and click on *2000 SF1 100% Data*;
   c. expand "Product Type" and click on *Detailed Table*; and
   d. expand "Housing", expand "Occupancy Characteristic", and then click on *Household Type*.
3. In the *Search Results* table, select **P019 Households by Presence of People Under 18 Years By Household Type [19]**.
4. To download the data, click on **Download** at the top of this table. You will be prompted to save a zip file. This may take a while, depending on your Internet connection speed.