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 (2010)
10, Number 1, pp. 30–45

Using the world development indicators database for statistical analysis in Stata

P. Wilner Jeanty
Department of Agricultural, Environmental, and Development Economics
The Ohio State University
Columbus, OH
jeanty.1@osu.edu

Abstract. The World Bank's world development indicators (WDI) compilation is a rich and widely used database about the development of most economies in the world. However, after insheeting a WDI dataset, some data management is required prior to performing statistical analysis. In this article, I propose a new Stata command, `wdireshape`, for automating this data management. While reshaping a WDI dataset into structures amenable to panel data, seeming unrelated regression, or cross-sectional modeling, `wdireshape` renames the series and places the series descriptors into variable labels.

Keywords: dm0045, wdireshape, paverage, world development indicators, reshape, panel data, seeming unrelated regression

1 Introduction

The World Bank's world development indicators (WDI) compilation is a rich and widely used database about the development of most countries in the world, with time series going back to 1960. The 2009 WDI compilation consists of more than 800 indicators in over 90 tables organized in 6 sections, including world view, people, environment, economy, states and markets, and global links (The World Bank Group, 2009). The WDI database is available online for a paid subscription or on a CD-ROM as a purchase. However, after insheeting a WDI dataset, some data management is required prior to performing statistical analysis. Further, while the World Bank has made great strides in rendering WDI in several forms for download, seemingly unrelated regressions analysis, for example, cannot be carried out using any of such forms. The new Stata command `wdireshape`, presented in this article, automates the data management required to get the data ready for statistical analysis. In particular, `wdireshape` allows you to obtain data structures amenable to panel data, seeming unrelated regression, or cross-sectional modeling.

The next section presents the `wdireshape` command, and section 3 expands on data preparation and how to get WDI data into Stata. Section 4 outlines `wdireshape` syntax and options, while examples are presented in section 5. Section 6 concludes the article.

© 2010 StataCorp LP

2 The wdireshape command

`wdireshape` reshapes a Stata dataset obtained by insheeting a text (`.csv`) file downloaded from the WDI World Bank web site or extracted from the WDI CD-ROM. Depending on the option specified with `wdireshape`, the new dataset has a structure suitable for panel-data analysis, seemingly unrelated regression (SUR), or cross-sectional modeling. The panel-data structure is known as long form, and the SUR and cross-sectional structures are known as wide form. During the reshaping process, `wdireshape` places the WDI series descriptors into Stata variable labels and enables users to supply, in a varlist, names of their devising for the WDI series.

3 Data preparation

Before extracting a `.csv` file from the WDI web site or a recent CD release, users must choose a data orientation with series or country in rows and time in columns. The `.csv` file can be imported into Stata by typing

```
. insheet using filename.csv, names clear
```

`wdireshape` works on this insheeted dataset. Older CD releases, such as the WDI-2005 CD-ROM, produce `.csv` files that must be managed prior to insheeting. In particular, the years must be prepended with a letter, which can be done in a spreadsheet or by using the procedure suggested in Baum and Cox (2007).

4 Commands

4.1 Syntax for wdireshape

The syntax to check the number of indicators and their order of appearance in a WDI dataset is

`wdireshape, sername(`*varname*`)`

The syntax to reshape the dataset is

`wdireshape` *newvarlist*`, prepend(`*letter(s)*`) ctyname(`*varname*`) sername(`*varname*`)`
`ctycode(`*varname*`) sercode(`*varname*`) [byper(#) startyr(#) endyr(#)`
`byvar sur cros nstring(#)]`

(*Continued on next page*)

4.2 Options for wdireshape

Required options

`prepend(`*letter(s)*`)` specifies the letters with which the years are prepended. One or two letters from `a` to `z` should be used. When insheeting a WDI dataset downloaded from the World Bank's web site, the years are prepended with "`yr`".

`ctyname(`*varname*`)` specifies the variable containing the country names.

`sername(`*varname*`)` specifies the variable holding the series names.

`ctycode(`*varname*`)` specifies the variable containing the country code elements.

`sercode(`*varname*`)` specifies the variable containing the series code elements.

With these five required options specified, `wdireshape` will attempt to reshape the entire dataset at once, which is the default. Due to memory issues, reshaping large datasets at once may not be successful. In such a case, Stata will prompt the user to specify the `byvar` or `byper(#)` option, or to increase the amount of memory allocated to Stata. You can reset the size of memory only if you are using Stata/MP, Stata/SE, or Stata/IC.

Optional options

`byper(#)` requires `wdireshape` to reshape the dataset 1 year, 5 years, or 10 years at a time, as long as the time span contains no gaps. One of these three values should be used with the `byper(#)` option. If either 5 or 10 is specified, `wdireshape` will account for the fact that the last subperiod may not be of 5 or 10 years. Also, Stata will check whether the current memory size is enough to reshape the data 5 or 10 years at a time.

`startyr(#)` specifies the first year of the time period.

`endyr(#)` specifies the last year of the time period.

Note 1: The `byper(#)`, `startyr(#)`, and `endyr(#)` options must be combined.

`byvar` specifies that the dataset be reshaped one variable at a time, as proposed by Kossinets (2006). The `byvar` option may not be combined with `byper(#)`, `startyr(#)`, and `endyr(#)`.

`sur` requests a wide form suitable for (SUR) analysis (see [R] **sureg**). By default, the dataset is reshaped in long form for panel-data analysis (see [XT] **xtreg**). When this option is specified, in the reshaped dataset, the country names are postfixed to the user-supplied variable names and are represented by `c1`, `c2`, etc. `describe` the reshaped dataset if you want to know which countries `c1`, `c2`, ..., `c`*n* represent. In Stata 10 or higher, you can just look at the variable labels in the variable window. The SUR-reshaped structure displays the years in rows and the variables, for each country, in columns.

`cros` requests a wide form suitable for cross-sectional analysis. The `cros`-reshaped structure displays the country names in rows and the variables, for each year, in columns. Obviously, the `cros` option may not be combined with `sur`.

Note 2: When the `sur` or `cros` option is specified, Stata will complain if the resulting number of variables exceeds its limits, which are 99 for Small Stata, 2,047 for Stata/IC, and 32,767 for Stata/MP and Stata/SE.

`nstring(#)` indicates that the dataset contains the WDI missing-value symbols, the double dots (`..`), and that they should be removed from the reshaped dataset. # represents the number of identifier variables in the dataset. For example, `nstring(4)` must be specified when the dataset includes names and code elements for both countries and series as identifier variables. When the `nstring(#)` option is specified, if an error occurs for any reason, the dataset to be reshaped needs to be reloaded before running `wdireshape` again. Otherwise, Stata will abort with a type-mismatch error.

Note 3: In the case of large datasets, reshaping 10 years at a time—as long as the time period is at least 10 years and there is enough memory—would be much faster than reshaping variable by variable. However, when the time period contains gaps, `byper(#)` will not work.

4.3 Syntax for paverage

The syntax to calculate *p*-year averages of the variables in a panel dataset is

`paverage` *varlist*`, p(#)` `indiv(`*varname*`)` `yr(`*varname*`)`

4.4 Description of paverage

`paverage` (pronounced p-average) calculates series of averages in a panel dataset. In the process, the labels of the original variables, if present, are attached to the average variables. The time period must be a multiple of the subperiod over which averages need to be calculated. When analyzing a panel dataset with a long time period, using series of averages is a common way to reduce business-cycle effects and measurement error.

4.5 Options for paverage

`p(#)` indicates the number of years for which averages need to be calculated. # ranges from 2 to 10. For example, specifying `p(5)` will create a five-year average dataset.

`indiv(`*varname*`)` specifies the variable name containing the country names, individuals, or firms.

`yr(`*varname*`)` specifies the variable name holding the years.

5 Examples

This section illustrates `wdireshape`, and all examples use data downloaded from the WDI web site.

5.1 Example 1: Reshape everything at once for panel-data analysis

To import the data, type

```
. set memory 20m
(20480k)
. insheet using wdi_exmpl1.csv, names clear
(46 vars, 577 obs)
```

I take a look at the raw dataset by listing a few observations for three countries and two years of the time period, 1961 and 2002.

```
. list countryname seriesname yr1961 yr2002 in 1/13, sepby(countryname)
> string(30) noobs
```

countryname	seriesname	yr1961	yr2002
Afghanistan	Foreign direct investment, net..	..	..
Afghanistan	GDP (current US$)	1240000000	4040000000
Afghanistan	Trade (% of GDP)	12.55061	..
Algeria	Foreign direct investment, net..	..	1.904728
Algeria	Foreign direct investment, net..	..	1070000000
Algeria	Foreign direct investment, net..	..	..
Algeria	GDP (current US$)	2430000000	55900000000
Algeria	Trade (% of GDP)	113.7483	61.70864
Angola	Foreign direct investment, net..	..	15.43182
Angola	Foreign direct investment, net..	..	1670000000
Angola	Foreign direct investment, net..	..	0.2648883
Angola	GDP (current US$)	..	10800000000
Angola	Trade (% of GDP)	..	143.2107

Prior to reshaping the data, I use the first syntax of `wdireshape` to rigorously check the number of variables (series/indicators) and their order of appearance. I recommend doing this check before reshaping the data.

```
. wdireshape, sername(seriesname)
 The current dataset contains 5 variables in the following order:
 1) Foreign direct investment, net inflows (% of GDP)
 2) Foreign direct investment, net inflows (BoP, current US$)
 3) Foreign direct investment, net outflows (% of GDP)
 4) GDP (current US$)
 5) Trade (% of GDP)
```

Supplying five new variable names that befit the series descriptors, I now use the second syntax of `wdireshape` to reshape the data. I specify `nstring(4)`[1] to remove the WDI missing-value symbols, the double dots (`..`), from the reshaped dataset. Removing the double dots is paramount to perform numerical operations on the reshaped dataset: Stata will regard as string data any columns containing them.

```
. wdireshape fdingdp fdincur fdiout gdp trade,  prepend(yr) ctyname(countryname)
> sername(seriesname) sercode(seriescode) ctycode(country_code) nstring(4)
Reshaping your dataset (everything at once), please wait
Your dataset has been reshaped and is ready for panel data analysis
```

As output, Stata first announces that the entire dataset is being reshaped at once. When the conversion process is complete, another message asserts that the dataset has been reshaped and is ready for panel-data analysis, which is the default.

I use `describe` (see [D] **describe**) to get an overview of the reshaped dataset:

```
. describe
Contains data
  obs:         5,166
 vars:             9
 size:       428,778 (98.0% of memory free)
```

variable name	storage type	display format	value label	variable label
cid	float	%9.0g		Country ID
countrycode	str3	%9s		Country code
countryname	str30	%30s		Country name
year	int	%9.0g		
fdingdp	double	%12.0g		Foreign direct investment, net inflows (% of GDP)
fdincur	double	%12.0g		Foreign direct investment, net inflows (BoP, current US$)
fdiout	double	%12.0g		Foreign direct investment, net outflows (% of GDP)
gdp	double	%12.0g		GDP (current US$)
trade	double	%12.0g		Trade (% of GDP)

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

1. If the `nstring(#)` option is specified and an error occurs for any reason (e.g., invalid syntax), the raw dataset to be reshaped needs to be reloaded before running `wdireshape` again. Otherwise, Stata will abort with a type-mismatch error.

Another look is provided by `list`ing the first few observations:

```
. list countryname year fdingdp fdincur fdiout gdp trade in 1/10
```

	countryname	year	fdingdp	fdincur	fdiout	gdp	trade
1.	Afghanistan	1961	.	.	.	1.240e+09	12.55061
2.	Afghanistan	1962	.	.	.	1.230e+09	14.22764
3.	Afghanistan	1963	.	.	.	9.610e+08	26.03551
4.	Afghanistan	1964	.	.	.	8.000e+08	26.94445
5.	Afghanistan	1965	.	.	.	1.010e+09	32.67108
6.	Afghanistan	1966	.	.	.	1.400e+09	27.14286
7.	Afghanistan	1967	.	.	.	1.670e+09	20.98273
8.	Afghanistan	1968	.	.	.	1.370e+09	24.11003
9.	Afghanistan	1969	.	.	.	1.410e+09	25.07887
10.	Afghanistan	1970	.	.	.	1.750e+09	21.72811

Panel-data analysis

I now use the Stata `xtsum` command to show that the data are xt ready (see [XT] **xt**).

```
. xtsum fdingdp fdincur fdiout gdp trade
```

Variable		Mean	Std. Dev.	Min	Max	Observations
fdingdp	overall	1.858811	4.390006	-82.81054	72.32173	N = 3136
	between		2.099348	-.0070139	10.36776	n = 110
	within		3.924819	-85.64466	69.48761	T-bar = 28.5091
fdincur	overall	4.19e+08	2.61e+09	-4.55e+09	4.93e+10	N = 3642
	between		1.48e+09	-4375758	1.30e+10	n = 111
	within		2.15e+09	-1.26e+10	3.67e+10	T-bar = 32.8108
fdiout	overall	.1266303	.5728872	-7.195931	7.155485	N = 2474
	between		.2665292	-.0696363	2.175378	n = 114
	within		.4976684	-7.423015	6.91092	T-bar = 21.7018
gdp	overall	2.41e+10	8.02e+10	1.16e+07	1.45e+12	N = 4171
	between		5.29e+10	4.35e+07	3.64e+11	n = 121
	within		5.63e+10	-2.94e+11	1.11e+12	T-bar = 34.4711
trade	overall	67.14495	38.96681	1.530677	228.8752	N = 3985
	between		35.73635	15.20354	163.0827	n = 121
	within		17.71395	-18.7896	174.2657	T-bar = 32.9339

`xtsum` provides much more information than `summarize`. For example, the values for n in the "Observations" column indicate the number of countries for which data are available on each variable. Because the data are xt ready, panel-data models can be fit using the Stata `xtreg` command (see [XT] **xtreg**).

Time series of averages across countries

Time series of averages across countries can be obtained using the Stata `collapse` command.

```
. collapse fdingdp fdincur fdiout gdp trade, by(year) cw // mean is the default
```

Now suppose that I want to graph the time series of averages for investment (net inflows and outflows). I type the following lines of code to produce figure 1. Here I use two y axes because the two variables are of different scales.[2]

```
. set scheme sj
. label variable fdingdp "Investment, net inflows (% of GDP)"
. label variable fdiout "investment, net outflows (% of GDP)"
. scatter fdingdp year, s(t) c(l) yaxis(1)  ||
> scatter fdiout year, s(D) c(l) yaxis(2) legend(cols(1)) xlab(1970(5)2002)
```

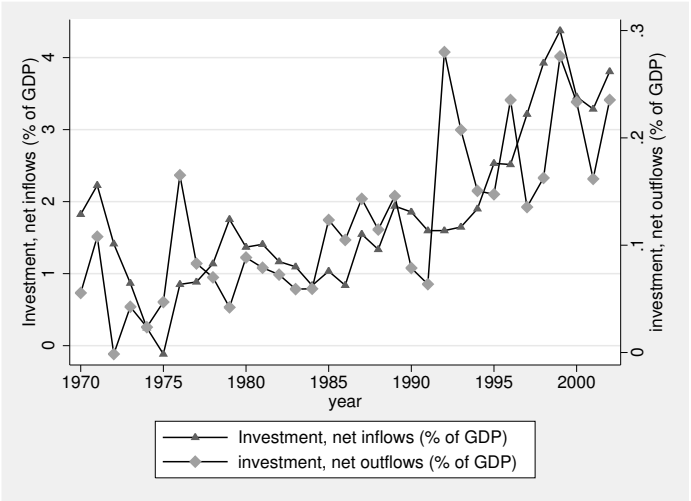

Figure 1. Time series of averages of investment (net inflows and outflows)

Country averages across years

To obtain country averages across years, I load and reshape the dataset one more time. But this time, I elect to reshape one variable at a time by specifying the `byvar` option.

2. Issuing the `tsline fdingdp fdiout` command would do the job, but scales are not taken into account.

```
. insheet using wdi_exmpl1.csv, names clear
(46 vars, 577 obs)
. wdireshape fdingdp fdincur fdiout gdp trade, prepend(yr) ctyname(countryname)
> sername(seriesname) sercode(seriescode) ctycode(countrycode) byvar nstring(4)
Reshaping your dataset one variable at a time, please wait
Your dataset has been reshaped and is ready for panel data analysis
```

I now run the Stata `collapse` command to obtain country averages across years.

```
. collapse  fdingdp fdincur fdiout gdp trade, by(countryname) cw
```

Now suppose that I want to know which countries have their annual average net outflow investment between 0.1 and 0.5 percent, given their annual average net inflow investment. The names of these countries are displayed in figure 2, which I obtained by typing the following:

```
. label var fdingdp "Investment, net inflows (% of GDP)"
. label var fdiout "investment, net outflows (% of GDP)"
. generate pos=3
. replace pos=6 if countryname== "South Africa"
(1 real change made)
. replace pos=9 if countryname=="Niger"
(1 real change made)
. scatter fdingdp fdiout  if fdiout>=.1 & fdiout<=.5, mlabel(countryname)
> xscale(log) xscale(range(.6)) mlabv(pos) yscale(range(-1))
```

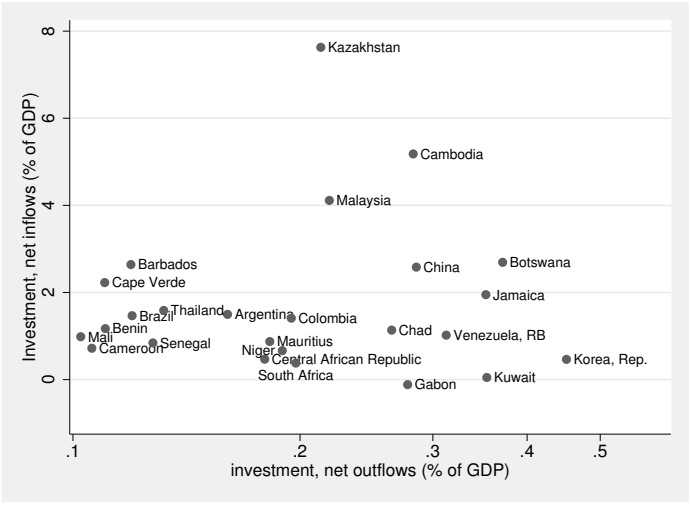

Figure 2. Countries with annual average net outflow investment between 0.1 and 0.5 percent of gross domestic product (GDP)

If the time period is a multiple of 5 or 10, 5-year or 10-year averages can be calculated using the `paverage` command described in section 4.3. `paverage` is available from the

Statistical Software Components archive. Baum (2006) also illustrates some tools for operating on and manipulating panel data.

5.2 Example 2: Reshape five years at a time for SUR analysis

To import the data and check the number of indicators and their order of appearance, I type

```
. insheet using wdi_country_time.csv, names clear
(50 vars, 1157 obs)

. wdireshape, sername(seriesname)

 The current dataset contains 13 variables in the following order:

 1) Agricultural land (% of land area)
 2) Agricultural machinery, tractors per 100 sq. km of arable land
 3) Fertilizer consumption (100 grams per hectare of arable land)
 4) GDP (constant 2000 US$)
 5) GDP (current US$)
 6) GDP per capita growth (annual %)
 7) Irrigated land (% of cropland)
 8) Permanent cropland (% of land area)
 9) Population density (people per sq. km)
 10) Population growth (annual %)
 11) Rural population density (rural population per sq. km of arable land)
 12) Trade (% of GDP)
 13) Urban population growth (annual %)
```

I choose to reshape the data five years at a time and invoke the `sur` option to request a structure amenable to SUR analysis. Although a memory size of four megabytes is enough to load the data, a memory error message would have occurred had I not specified `byp(5)`.

```
. wdireshape agland tractsk fertilha gdpcnst gdpcur gdppg irrigpct croplnd
> popdens popg ruraldens trade urbpg, prepend(yr) ctyname(countryname)
> sername(seriesname) sercode(seriescode) ctycode(countrycode) startyr(1961)
> endyr(2006) byper(5) sur nstring(4)

Reshaping your dataset 5 years at a time
Now reshaping period  1961 - 1965
Now reshaping period  1966 - 1970
Now reshaping period  1971 - 1975
Now reshaping period  1976 - 1980
Now reshaping period  1981 - 1985
Now reshaping period  1986 - 1990
Now reshaping period  1991 - 1995
Now reshaping period  1996 - 2000
Now reshaping period  2001 - 2006

Your dataset has been reshaped and is ready for SUREG analysis
```

The dataset reshaped for SUR contains 1,158 variables, because the raw dataset had 13 variables and 89 countries. As mentioned above, this structure displays the years in rows and the variables, for each country, in columns. I now display a few observations on the variables `year` and GDP per capita growth pertaining to four Latin American (LA) countries.

```
. list year gdppgc11 gdppgc9 gdppgc32 gdppgc60 in 1/5
```

	year	gdppgc11	gdppgc9	gdppgc32	gdppgc60
1.	1961	7.0261211	-.15207509	1.5345942	4.0316689
2.	1962	2.0897761	3.2527942	.78809956	8.1733652
3.	1963	-2.107187	4.0718175	6.630273	6.466808
4.	1964	.49288856	2.464101	1.852948	7.5981292
5.	1965	.17106316	3.2549077	1.5911081	5.5632783

As should be emphasized, the country names are concatenated to the user-supplied variable names and are represented by `c1`, `c2`, `c3`, and so on. You can decipher them by typing `describe` or by looking at the variable labels from the variable window if you are using Stata 10 or higher. I now `describe` the variable GDP per capita growth for the four LA countries.

```
. describe gdppgc11 gdppgc9 gdppgc32 gdppgc60
              storage  display     value
variable name   type   format      label      variable label
------------------------------------------------------------------------------
gdppgc11        double %10.0g                 GDP per capita growth (annual %)
                                                - Brazil
gdppgc9         double %10.0g                 GDP per capita growth (annual %)
                                                - Bolivia
gdppgc32        double %10.0g                 GDP per capita growth (annual %)
                                                - Guatemala
gdppgc60        double %10.0g                 GDP per capita growth (annual %)
                                                - Nicaragua
```

To show that the reshaped dataset is ready for SUR analysis, I estimate four SURs, one for each LA country. Given the data at hand, I want to investigate whether the agricultural sector contributes to the GDP per capita growth in these countries. First, I `describe` for the first country, which is Brazil, the other variables used in the analysis. Then I estimate the regressions with the Stata `sureg` command. Here I specify the `dfk` and `corr` options to request a small-sample adjustment and the Breusch–Pagan test for independent equations.

```
. describe tractskc11 fertilhac11 croplndc11 popgc11
              storage  display     value
variable name   type   format      label      variable label
------------------------------------------------------------------------------
tractskc11      double %10.0g                 Agricultural machinery, tractors
                                                per 100 sq. km of arable land -
                                                Brazil
fertilhac11     double %10.0g                 Fertilizer consumption (100 grams
                                                per hectare of arable land) -
                                                Brazil
croplndc11      double %10.0g                 Permanent cropland (% of land
                                                area) - Brazil
popgc11         double %10.0g                 Population growth (annual %) -
                                                Brazil
```

```
. foreach num of numlist  11 9 32 60 {
  2.    local eqn "`eqn´ (gdppgc`num´ L.gdppgc`num´ tractskc`num´ fertilhac`num´
> croplndc`num´ popgc`num´) "
  3. }
. sureg `eqn´, dfk corr
Seemingly unrelated regression
```

Equation	Obs	Parms	RMSE	"R-sq"	chi2	P
gdppgc11	41	5	3.083649	0.4231	24.85	0.0001
gdppgc9	41	5	3.174336	0.2522	16.53	0.0055
gdppgc32	41	5	1.719363	0.5391	41.80	0.0000
gdppgc60	41	5	5.245668	0.3524	23.91	0.0002

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
gdppgc11						
gdppgc11						
L1.	.2014618	.1446664	1.39	0.164	-.0820791	.4850028
tractskc11	-.0842038	.0301914	-2.79	0.005	-.1433779	-.0250297
fertilhac11	.009484	.0040085	2.37	0.018	.0016275	.0173405
croplndc11	-20.48911	10.52829	-1.95	0.052	-41.12418	.1459484
popgc11	-1.500581	2.382403	-0.63	0.529	-6.170006	3.168843
_cons	24.21315	11.5525	2.10	0.036	1.570678	46.85563
gdppgc9						
gdppgc9						
L1.	.0428409	.1508581	0.28	0.776	-.2528357	.3385174
tractskc9	-.1529211	.1650501	-0.93	0.354	-.4764134	.1705712
fertilhac9	-.0086263	.0561642	-0.15	0.878	-.1187061	.1014536
croplndc9	78.50515	30.84602	2.55	0.011	18.04806	138.9622
popgc9	17.4214	5.810168	3.00	0.003	6.033678	28.80912
_cons	-46.62524	16.28789	-2.86	0.004	-78.54892	-14.70157
gdppgc32						
gdppgc32						
L1.	.3834769	.1242297	3.09	0.002	.1399912	.6269626
tractskc32	-.318205	.1216817	-2.62	0.009	-.5566968	-.0797131
fertilhac32	.0112325	.0029539	3.80	0.000	.005443	.017022
croplndc32	-6.652823	1.92693	-3.45	0.001	-10.42954	-2.876109
popgc32	11.52029	4.294384	2.68	0.007	3.103453	19.93713
_cons	3.227747	11.92218	0.27	0.787	-20.1393	26.59479
gdppgc60						
gdppgc60						
L1.	-.1455299	.1551685	-0.94	0.348	-.4496546	.1585947
tractskc60	-.9258023	.2191462	-4.22	0.000	-1.355321	-.4962838
fertilhac60	.0129039	.0078384	1.65	0.100	-.0024592	.0282669
croplndc60	36.16019	12.29522	2.94	0.003	12.06201	60.25838
popgc60	7.464455	4.188569	1.78	0.075	-.7449899	15.6739
_cons	-67.79043	29.24217	-2.32	0.020	-125.104	-10.47684

(*Continued on next page*)

```
Correlation matrix of residuals:

           gdppgc11   gdppgc9  gdppgc32  gdppgc60
gdppgc11    1.0000
 gdppgc9   -0.3330    1.0000
gdppgc32    0.0860   -0.0576    1.0000
gdppgc60   -0.2389    0.1804   -0.2081    1.0000

Breusch-Pagan test of independence: chi2(6) =    10.436, Pr = 0.1074
```

As the results indicate, the null hypothesis that the coefficients in each respective regression are zero is rejected at the one percent significance level. However, the Breusch–Pagan test fails to reject at the conventional significance level the null hypothesis that the correlation of the residuals across equations is zero.

5.3 Example 3: Reshape 10 years at a time for cross-sectional analysis

In this example, the same dataset used in example 2 is reshaped 10 years at a time. I specify the `cros` option to request a structure appropriate for cross-sectional analysis.

```
. drop _all

. quietly set memory 2m

. insheet using wdi_country_time.csv, names
(50 vars, 1157 obs)

. wdireshape agland tractsk fertilha gdpcnst gdpcur gdppg irrigpct croplnd
> popdens popg ruraldens trade urbpg, prepend(yr) ctyname(countryname)
> sername(seriesname) sercode(seriescode) ctycode(countrycode) start(1961)
> end(2006) byp(10) cros nstring(4)

Reshaping your dataset 10 years at a time
Now reshaping period  1961 - 1970
Now reshaping period  1971 - 1980
Now reshaping period  1981 - 1990
Now reshaping period  1991 - 2000
Now reshaping period  2001 - 2006

Your dataset has been reshaped and is ready for cross sectional or change analysis
```

As can be seen, `wdireshape` accounts for the fact that the subperiod 2001–2006 is not of 10 years. Because the raw dataset consists of 13 variables observed on a period of 46 years, the reshaped dataset contains at least a total of 598 variables. Recall that the `cros`-reshaped structure places the country names in rows and the variables, for each year, in columns. For the years 1961 and 2006, I `list` a few observations on the variables population density (`popdens`) and urban population growth (`urbpg`):

```
. list countryname popdens1961 popdens2006 urbpg1961 urbpg2006 in 1/10
```

	countryname	popd~1961	popd~2006	urbpg1961	urbpg2006
1.	Afghanistan	15.623018	.	5.6376534	5.2102131
2.	Algeria	4.6212609	14.001507	6.4459141	2.4988708
3.	Angola	4.0977862	13.147816	5.8804117	4.0143405
4.	Argentina	7.6554608	14.294807	2.3680321	1.1362566
5.	Bangladesh	402.52815	1108.8965	6.4879591	3.5009976
6.	Barbados	538.07209	628.00706	.60466256	1.3126057
7.	Belize	4.1316061	13.030856	2.8929328	2.2994995
8.	Benin	21.300515	78.586803	8.3754272	3.9628651
9.	Bolivia	3.1597392	8.617589	3.0227506	2.470662
10.	Botswana	1.0341344	3.1018031	6.8385594	.9156059

I now `describe` these variables:

```
. describe countryname popdens1961 popdens2006 urbpg1961 urbpg2006
```

variable name	storage type	display format	value label	variable label
countryname	str24	%24s		Country name
popdens1961	double	%10.0g		1961 - Population density (people per sq. km)
popdens2006	double	%10.0g		2006 - Population density (people per sq. km)
urbpg1961	double	%10.0g		1961 - Urban population growth (annual %)
urbpg2006	double	%10.0g		2006 - Urban population growth (annual %)

Now if I want to calculate change in population density from 1961 to 2006, I type

```
. generate popdens_ch = popdens2006 - popdens1961
(2 missing values generated)
```

In summary, `wdireshape` helps reduce data-management tasks when WDI users need to

- Conduct a panel-data analysis.

  Run `wdireshape` without specifying the `sur` or `cros` option.

- Analyze a time series of averages across countries.

  First, run `wdireshape`, and then run the Stata `collapse` command by the variable containing the years.

- Analyze a series of averages across years (pure cross-sections).

  First, run `wdireshape`, and then run the Stata `collapse` command by the variable holding the country names.

- Analyze a series of p-year averages (with $p = 2, 3, \ldots, 10$).

  First, run `wdireshape`, and then run `paverage`, available from the Statistical Software Components archive.

- Conduct a SUR analysis.

  Run `wdireshape` with the `sur` option.

- Perform a change or cross-sectional analysis.

  Run `wdireshape` with the `cros` option.

- Operate on and manipulate WDI as a panel dataset of countries.

  Run `wdireshape`, and then apply Baum's 2006 suggestions provided in chapter 3.

6 Conclusions

In this article, I introduced a new Stata command, `wdireshape`, enabling Stata users to efficiently manage WDI datasets. While allowing users to supply variable names of their choosing for the series, `wdireshape` reshapes the data for panel data, SUR, or cross-sectional modeling. In the process, the WDI series descriptors are placed into Stata variable labels.

7 Acknowledgments

I thank Christopher Baum, the participants of the 2008 Summer North American Stata Users Group meeting in Chicago, and an anonymous reviewer for useful comments and suggestions. This work was partly funded by the grant "Conflict, Poverty, and Environmental and Food Security" from the Mershon Center at The Ohio State University. Support from the Swank Program is also greatly acknowledged.

Inspiration for the `byvar` option comes from Stata code posted on the web by Kossinets (2006).

8 References

Baum, C. F. 2006. *An Introduction to Modern Econometrics Using Stata*. College Station, TX: Stata Press.

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

Kossinets, G. 2006. Code to reshape a WDI dataset in Stata.
http://www.columbia.edu/acis/eds/data_tools/tips/reshape_manyvar.do.

The World Bank Group. 2009. World Development Indicators (WDI) Online.
http://publications.worldbank.org/WDI/.

About the author

P. Wilner Jeanty is a postdoctoral scholar in the Department of Agricultural, Environmental, and Development Economics at The Ohio State University.