

Manipulating Data with STATA

Rachael Walsh

Pennsylvania State University

rmw211@psu.edu

12:00-1:00pm

9/17/2009

Recoding Variables

When large scale datasets are reproduced for public consumption, many of the variables need to be recoded. For example, when performing a simple regression analysis, you do not want the value of female to be 2 in your gender variable, or your results will be incorrect. In this case, the gender variable has to be recoded into a dummy variable so that there is a reference group (=0) in order to make your results interpretable.

```
gen female = (sex==2)
```

Note: generate can be used in place of gen and female is the new variable name.

The double equals sign creates a dummy variable; in this case, sex is recoded so that male = 0 (or the reference group), and female = 1. You can verify this was done correctly using the **tab** command.

```
tab sex then tab female
```

The total of females in the sex variable should be equivalent to the total of 1 responses under female. As with the generate command, tabulate can be used in place of tab.

Now recode the race variable, creating dummy variables for Black, Hispanic, and Asian. Use the **tab** command to determine whether or not your recodes were successful. For each instance, the positive response of the dummy variable should match the value in the original race variable (i.e. 1,207 respondents should report being Black).

Reshaping Data & Variables

First you need to identify the type of data you have. If the data is person-level data, that is to say that there is one record per person, then it should look like this:

```
list pid pov*
```

	pid	pov1992	pov1993	pov1994	pov1996	pov1997	pov1998	pov1999
1.	1	1	1	1	1	1	1	1
2.	2	1	1	1	.	.	.	1
3.	3	1	1	2	1	4	.	4
4.	4	4	4	4	4	4	4	4
5.	5	1	1	1	1	2	1	1
6.	6	3	4	4	4	4	4	4
7.	7	3	4	4	4	4	4	4
8.	8	3	4	4	4	4	.	.
9.	9	3	4	4	2	3	3	2
10.	10	3	4	4	2	3	3	2

If you have person-period data, then each person has a record for each period and would look like the following

list pid pov year

	pid	pov	year
1.	1	1	1992
2.	1	1	1993
3.	1	1	1994
4.	1	1	1996
5.	1	1	1997
6.	1	1	1998
7.	1	1	1999
8.	2	1	1992
9.	2	1	1993
10.	2	1	1994
11.	2	.	1996
12.	2	.	1997
13.	2	.	1998
14.	2	1	1999

As you can see, there are now seven records per person because poverty status was determined in seven different periods.

Depending on your research question and the type of analysis, it may be necessary to convert person-level data to person-period data, and visa-versa. There are two relatively simple commands to do this. If you have person-level data and wish to convert it to person-period data:

reshape long *varlist*, i(pid) j(year)

In this example, the data should look like the second table, which you can check using the previous list command. The term following *i* indicates the variable basis of the reshape, while the *j* term identifies the time period or marker of the subsequent variables to be reshaped. In this case the variables were in annual increments, thus the term following the *j* command is year. This will vary from dataset to dataset depending on the variables being reshaped; it could be monthly increments, education level increments, etc.

To revert back to the person-level data, use the following:

reshape wide *varlist*, i(pid) j(year)

Again, you can use the previous list command to make sure your reshape was successful. In this case it should match the first output.

Example: This example is going to include reshaping data as well as forming a composite. To form a composite variable from two or more variables, you must first complete a factor analysis to ensure they correlate and can form an adequate measure. This can be done using different commands, determining how much information you need. There are different types of factor

analyses, and different fields recommend different methods. For more information on factor analysis commands in STATA see <http://www.stata.com/help.cgi?factor>.

reshape long foodst pov, i(pid) j(year)

list pid foodst pov year

	pid	foodst	pov	year
1.	1	1	1	1992
2.	1	1	1	1993
3.	1	1	1	1994
4.	1	1	1	1996
5.	1	1	1	1997
6.	1	1	1	1998
7.	1	1	1	1999
8.	2	1	1	1992
9.	2	1	1	1993
10.	2	1	1	1994
11.	2	.	.	1996
12.	2	.	.	1997
13.	2	.	.	1998
14.	2	1	1	1999
15.	3	1	1	1992

While we have the data in the person-period format, we are going to form a composite. The use of food stamps can be used as a determinant of poverty, so there is a likelihood that these two variables can be abbreviated into one variable. You can use either of the following paths to get there:

factor pov foodst, pcf

STATA Output: . factor foodst pov, pcf
(obs=22936)

Factor analysis/correlation	Number of obs	=	22936
Method: principal-component factors	Retained factors	=	1
Rotation: (unrotated)	Number of params	=	1

Factor	Eigenvalue	Difference	Proportion	Cumulative
Factor1	1.42835	0.85670	0.7142	0.7142
Factor2	0.57165	.	0.2858	1.0000

LR test: independent vs. saturated: chi2(1) = 4648.97 Prob>chi2 = 0.0000

Factor loadings (pattern matrix) and unique variances

Variable	Factor1	Uniqueness
foodst	0.8451	0.2858
pov	0.8451	0.2858

predict poverty

Note: Poverty is the name of the new variable formed through the factoring sequence. If there were more than one factors, you would have to provide a name for each factor to make each new variable. They would be put in sequential order with no commas, just one space (i.e. for three factors, predict p1 p2 p3).

Notice the execution of the predict command adds the variable poverty to the bank of variables on the bottom left side of the screen.

If you wanted additional information with respect to your factors and factor loadings you could also use the following commands:

alpha pov foodst, asis item cas

Notice the execution of the predict command adds the variable poverty to the bank of variables on the bottom left side of the screen.

Now that we have a new variable, lets go back to the person-level data format. To revert back, use the following command:

reshape wide pov foodst poverty, i(pid) j(year)

Again, notice the variables on the left. They are now ordered by year instead of variable, and there is a new poverty variable at each year of data collection.

Merging Datasets: STATA 11 (<http://www.stata.com/help.cgi?merge>)

**We are going to use code from an older version of STATA to show some of the highlights of STATA 11.

There are several different ways to merge data in STATA, including 1:1, m:1, 1:m, m:m, and sequential, in which m stands for many. A one-to-one merge would be used to merge datasets with generally the same people (coded with the same identifying variable). If both datasets contain only one record per person (a person-level dataset), the one-to-one merge can be used using the following:

merge 1:1 pid using merging

When you are merging datasets that are about the same sample, but coded differently, you will need to use the one-to-many merge, or 1:m. For example, you may have your master dataset coded by school, but the merging data is coded by student and for each student the school id is available. Thus the combining variable is the school, but in the first dataset each school is only listed once and in the merging dataset each school could be associated with several students.

merge 1:m schoolid using student dataset

If your master dataset is the student dataset instead of the school dataset:

merge m:1 schoolid using school dataset

Due to the complex nature and inherent error, it is not recommended to perform many-to-many merges (m:m) or sequential merges. If the data you wish to merge does not fall into the aforementioned categories, go to the STATA help link above to read more on these merges.

Merging Data: Older Versions of STATA

In older versions of STATA merges were slightly different. In order to merge two datasets, both datasets have to be sorted using the variables that are to be merged.

sort pid pov* foodst*

*Note: the * indicates to STATA to use all variables with matching letters prior to *.*

merge pid using “merging file name”

Note: When you use an older version of coding, STATA will provide you with a help link identifying and explaining the new code.

list pid foodst* _merge

See next page for output.

```
list pid foodst* _merge
```

	pid	foo~1992	foo~1993	foo~1994	foo~1996	foo~1997	foo~1998	foo~1999	foo~2000	foo~2001	_merge
1.	1	1	1	1	1	1	1	1	1	.	3
2.	2	1	1	1	.	.	.	1	1	.	3
3.	3	1	1	2	1	2	.	2	2	.	3
4.	4	2	2	2	2	2	2	2	1	.	3
5.	5	1	1	1	1	2	2	2	.	.	3
6.	6	1	2	2	2	2	2	2	2	.	3
7.	7	1	2	2	2	2	2	2	2	.	3
8.	8	1	2	2	2	2	3
9.	9	1	2	2	2	2	2	2	2	.	3
10.	10	1	2	2	2	2	2	2	2	.	3

You can tell your merge in this case was successful because you now have two additional years of poverty information, 2000 and 2001. Also, the new variable `_merge` that was created indicates how each person-level record was varied. In this case, the data collection was complete for at least the first 10 cases, and given the addition of two variables per person, the `_merge` total is three. The `pid` and `foodst1992-1999` were all included in the master dataset. The merging dataset added two, totally 3. Let's say person id # 9 was not interviewed in 2000, but were interviewed in 2001. Their merge total would be 2.

Notice that the merging dataset had more than just the `pid` and `pov2000 pov2001 foodst2000 foodst2001`; however, because `race` and `sex` are fixed variables, there was no need to copy them and duplicate existing data.