

Using Arrays in SAS and Stata

Hsueh-Sheng Wu
CFDR Workshop Series
November 14, 2012

BGSU



Center for
Family and
Demographic Research

Outline of Presentation

- Why do we need arrays?
- The difference in arrays between SAS and Stata
- Syntax rules of arrays in SAS
- Syntax rules of arrays in Stata
- Three things that you can use arrays for
 - Recode the values of several variables
 - Create several new variables
 - Change the data structure from the wide format to the long format and vice versa
- Reminders of using arrays
- Conclusions

Why Do We Need Arrays?

- Arrays are used at the stage of constructing data.
- Arrays can be thought as a new way to refer to several variables. With arrays, you can specify what to be done on each of these variables or on a certain combination of variables.
- Every task that can be done with arrays can also done without arrays.
- Why do we want to use arrays?
 - Efficiency: With arrays, you do not need to write many lines of codes for repetitive tasks.
 - Accuracy: With fewer lines of codes, you can easily spot the possible errors in these codes.

Difference in Arrays between SAS and Stata

- SAS and Stata have different array commands. Specifically, SAS has “array” statement, but Stata uses “foreach” statement.
- While SAS and Stata both have the same logic of array, the syntax of array is completely different.
- Arrays are more important for SAS users than for Stata users especially when variables are to be constructed from multiple-records-per-person format.

Syntax Rules of Arrays in SAS

- All variables specified within an array must be of the same type.
- Variables specified within an array do not need to be already existing variables
- Syntax of specifying an ARRAY in SAS:

```
array array-name{n} <$> <length> array-elements;
```

Tell SAS that an array will be created

the name of the array

The dimension of the array

The type of variables in the array

The lengths of the variables in the array

The names of variables in the array

Syntax Rules of Arrays in SAS (Cont.)

- After defining an array, you can specify what to do with each of the elements in the array, which will be covered at the later part of the workshop.
- Examples:
 - (1) an one-dimension array, called “number”, contains five numeric variables from number1 through number5. The length of each of these variable is 3 digits.

```
array number{5} 3 number1-number5;
```

```
array number{*} 3 number1-number5;
```

```
array number{1:5} 3 number1-number5;
```

```
array number{5} 3 ;
```

Syntax Rules of Arrays in SAS (Cont.)

Table 1. An Example of An Array References and Variables

Array reference	Variable Name
number{1}	number1
number{2}	number2
number{3}	number3
number{4}	number4
number{5}	number5

Syntax Rules of Arrays in SAS (Cont.)

- Examples:

(2) an one-dimension array, called “character”, contains three string variables from string1 through string3. The length of each of these variable is 2 characters.

```
array character{3} $ 2 string1-string3;
```

(3) an two-dimension array, called “season”, contains 12 numeric variables that reflecting twelve months. The length of each of these variable is 1 digit.

```
Array season{4,3} 1   January   February   March  
                    April      May        June  
                    July       August     September  
                    October    November  December;
```


Syntax Rules of Arrays in Stata

- There is no array commands in Stata
- The `-foreach-` and `-forvalues-` commands have similar functions as one-dimension array in SAS
- Syntax of `foreach`

Tell Stata to invoke `foreach` command

Create an index name to refer to each variable specified

Specify whether the list of variables are generic variables, existing variables, or new variables

The list of variables

The open brace indicates the end of specifying the list of variables and need to be on the same line as `foreach`.

```
foreach lname {in|of varlist} variables {  
  commands referring to `lname'  
}
```

The close brace must appear on a line by itself and signal the end of the `foreach` command

All the data-construction commands that use the variables specified in the `foreach` command should be placed between the open brace and the close brace.

Syntax Rules of Arrays in Stata (Cont.)

Examples: An one-dimension array contains five existing numeric variables from number1 through number5. We use an index, i, to indicate the elements of this array.

```
(1) foreach i in number1 number2 number3 number4 number5 {  
    display "`i'"  
}
```

```
(2) foreach i of varlist number1 number2 number3 number4 number5 {  
    display "`i'"  
}
```

```
(3) foreach i of newlist new1 new2 new3 new4 new5 {  
    gen `i' = 1  
}
```

Examples of Using Arrays

Example 1: using arrays to recode variables

Your data set has five variables, including var1-var5. These variables are coded as 99 if respondents refused to answer, and you want to recode these refused respondents into missing.

Table 1. SAS Commands for Recoding Variables

<u>Without using Arrays</u>	<u>Using Arrays</u>
if var1 = 99 then var1 = .;	array v{5} var1-var5;
if var2 = 99 then var2 = .;	do k=1 to 5;
if var3 = 99 then var3 = .;	if v{k}= 99 then v{k}=.;
if var4 = 99 then var4 = .;	end;
if var5 = 99 then var5 = .;	

Stata Commands in Recoding Variables

Table 2. Stata Commands in Recoding Variables

Without using foreach command	foreach command with in option	foreach command with of varlist option
replace var1 = . if var1 ==99	foreach var in var1 var2 var3 var4 var5 {	foreach var of varlist var1-var5 {
replace var2 = . if var2 ==99	replace `var' =. if `var' ==99	replace `var' =. if `var' ==99
replace var3 = . if var3 ==99	}	}
replace var4 = . if var4 ==99		
replace var5 = . if var5 ==99		

Create New Variables

Example 2: You want to generate five new variables, new1-new5. These variables all have a value of 1 for all respondents.

- SAS example

Table 3. SAS Commands in Generating New Variables

Without using Arrays	Using Arrays
<code>new1=1;</code>	<code>array v{5} new1-new5;</code>
<code>new2=1;</code>	<code>do i=1 to 5;</code>
<code>new3=1;</code>	<code>v{i}= 1;</code>
<code>new4=1;</code>	<code>end;</code>
<code>new5=1;</code>	

Sample Data

- Stata example

Table 4. Stata Commands in Generating New Variables

Without using Arrays	Using Arrays
gen new1=1	foreach i of newvar1 new1 new2 new3 new4 new5 {
gen new2=1	gen `i' =1
gen new3=1	}
gen new4=1	
gen new5=1	

Change Data Structure

- There are two types of data structure: wide and long. The wide format means one-record-per-person, and the long format means multiple-record-per-person.
- Different analyses may need different data formats.
- Stata has a built-in command called “reshape” to change the data structure, but SAS needs to use array to do so.

Change Data Structure (Cont.)

- Data in wide and long format

Table 5. Data in Wide Format

name	marriage at Wave 1	marriage at Wave 2	marriage at Wave 3
John	1	0	1
Mary	0	0	0
Tom	0	1	0

Table 6. Data in Long Format

name	wave	marriage
John	1	1
John	2	0
John	3	1
Mary	1	0
Mary	2	0
Mary	3	0
Tom	1	0
Tom	2	1
Tom	3	0

Change Data Structure (Cont.)

- SAS command: change data from wide to long

```
libname in 'c:\temp\array';  
data in.long2;  
set in.wide;  
array status[3] marriage1 marriage2 marriage3;  
do wave=1 to 3;  
Marriage = status[wave];  
output;  
keep name wave marriage;  
end;  
Run;
```

Change Data Structure (Cont.)

- SAS command: change data from long to wide

```
DATA in.wide3 ;
```

```
SET in.long3 ;
```

```
BY name ;
```

```
KEEP name wave marriage marriage1 marriage2 marriage3;
```

```
RETAIN marriage1 marriage2 marriage3 ;
```

```
ARRAY status(3) marriage1 marriage2 marriage3;
```

```
IF first.name THEN DO;
```

```
DO i = 1 to 3 ;
```

```
status(i) = . ;
```

```
END;
```

```
END;
```

```
status(wave) = marriage;
```

```
IF last.name THEN OUTPUT ;
```

```
RUN;
```

Change Data Structure (Cont.)

- Stata commands
 - Change the data from wide to long
 - **reshape long marriage, i(name) j(wave)**
 - Change the data from wide to long
 - **reshape wide marriage, i(name) j(wave)**

Reminders of Using Arrays and Macros

- Arrays save you time by substituting the repetitive patterns in your programming codes with arrays
- Learn arrays with the following steps:
 - Write a few lines of code without arrays or macros
 - Write an ARRAY statement to represent the repetitive patterns in the code
 - Substitute the array for the repeating patterns in the code
- When writing arrays, you need to look at them from the perspective of SAS or Stata in terms of how they will be interpreted.

Conclusions

- Using arrays can save time and efforts. For example, with arrays, you don't need to keep retyping the names of variables.
- By redefining arrays, you can run the same analyses on different sets of variables.
- Pay attention to new commands in SAS and Stata. Arrays used to be very powerful in changing the data from one-record-per-individual to multiple-record-per-individual and vice versa. However, Stata now has reshape command. If SAS develops similar commands in the future, you may not need array to switch between different structure of data.
- Be patient. You will run into many errors when you start writing arrays or macros, but practice makes perfect.
- CFDR programming support is available. Please call Hsueh-Sheng Wu @ 372-3119 or send an e-mail to wuh@bgsu.edu.