Arrays and Macros in SAS and Stata

Hsueh-Sheng Wu

CFDR Workshop Series

Fall 2009

BGSU Center for Family and Demographic Research

Outline of Presentation

- When do we need Arrays and Macros?
- Arrays in SAS
  - Relation between array elements and variables
  - Examples of using array in SAS
- Arrays in Stata
  - Differences in using -in- option and -of varlist- option in the -foreach- command
  - Example of using -foreach- command in Stata
- What are Macros?
  - Similarities and differences in using macros between SAS and Stata
- Macro variables in SAS and Stata
- Using macros in SAS and in Stata
- Reminders of using arrays and macros
- Conclusions

When Do We Need Arrays and Macros?

- If a SAS or Stata program involves repetitive actions on a group of variables

- Examples:
  - Encode missing values for n variables
  - Create n new variables

Arrays in SAS

- A set of variables grouped together for the duration of a data step by being given a name in an ARRAY statement
- Syntax:

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

  array number {5} 3 variable1-variable5;

  array character {3} $ 2 string1-string3;
- Two-dimension array

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

Arrays in SAS (Cont.)

- Explanation of these arrays
  - array-name – number, character, and season.
  - These arrays have 5, 3 and 12 elements, respectively.
  - Only the array, character, has character variables as its elements
  - The lengths of elements are 3, 2, and 1 in these arrays respectively
  - The elements in these three arrays are variable1 through variable3, string1 through string5, and the twelve months.

Relation between Array Elements and Variables

Tabe1. 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

Example of Using Array in SAS

Example 1: using array to recode variables

You 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 2. SAS commands in recoding variables

Without using Array	Using Array
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 = .;	

Example of using Array in SAS (Cont.)

Example 2: using array to generate new variables

You data set has five variables measuring temperature in the Fahrenheit scale, fahren1 through fahren5. You want to convert each of these variables into the Celsius scale.

Table 3. SAS commands in creating new variables

Without using Array	Using Array
Celsius1 = (Fahren1 - 32)/1.8;	array Fahren{5} Fahren1-Fahren5;
Celsius2 = (Fahren2 - 32)/1.8;	array Celsius{5} Celsius1-Celsius5;
Celsius3 = (Fahren3 - 32)/1.8;	do i = 1 to 5;
Celsius4 = (Fahren4 - 32)/1.8;	Celsius{i} = (Fahren{i} - 32)/1.8;
Celsius5 = (Fahren5 - 32)/1.8;	end;

Arrays in Stata

- There is no array commands in Stata
- The -foreach- command has similar functions as one-dimension array in SAS
- Syntax of foreach
  - foreach *lname* {**in**|**of varilist**} variables **{**
    commands referring to `lname'
    }
- The open brace must appear on the same line as the foreach;
- Nothing may follow the open brace except, of course, comments; the first command to be executed must appear on a new line;
- The close brace must appear on a line by itself

Differences in Using -in- option and -of varlist- option in the -foreach- command

- An example of a SAS array

  array number {5} 3 variable1-variable5;
- foreach i **in** variable1-variable5 {

  Stata commands

  }
  - There is only one variable called “variable1-variable5”
- foreach i **of varlist** variable1-variable5 {

  Stata commands

  }
  - There are five variables, including variable1 through variable5

Stata Commands in Recoding Variables

Table 4. 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		

Stata Commands in Generating New Variables

Table 5. Stata commands in creating new variables

Without using foreach command	foreach command with in option	foreach command with of varlist option

gen n_Fahren1 = (Fahren1 - 32)/1.8;
 	
gen n_Fahren2 = (Fahren2 - 32)/1.8;
 	
gen n_Fahren3 = (Fahren3 - 32)/1.8;
 	
gen n_Fahren4 = (Fahren4 - 32)/1.8;
 	
gen n_Fahren5 = (Fahren5 - 32)/1.8;
	foreach var in Fahren1 Fahren2 Fahren3 Fahren4 Fahren5 {
	
	gen n_`var'=(`var'-32)/1.8
	}
		foreach var of varlist Fahren1- Fahren5 {
		
		gen n_`var'=(`var'-32)/1.8
		}

What Are Macros?

- Macros can be entire programs or just pieces of programming codes.
- By creating and executing macros, you are writing SAS or Stata programs for customary tasks.
- SAS and Stata use different names and syntaxes for writing macros, which often creates confusions.

Similarities and Differences in Using Macros between SAS and Stata

Similarities:

- Ability to use macros to represent one or more variables as well as a series of programming codes
- To execute macros, you need to define them first.

Differences:

- Naming is different. A variable or variables are represented by “a macro variable” in SAS and “macro” in Stata. A series of programming codes are represented by “a macro” in SAS and “program” in Stata.
- The procedure of using “macros” is different. In SAS, you need to define it and then execute it. In Stata, you need to define it, load it, and then execute it.

A Sample Data

Table 6. A sample data

make_model	price	mpg	foreign
Pontiac Firebird	4,934	18	Domestic
Volvo 260	11,995	17	Foreign
Toyota Corolla	3,748	31	Foreign
Chevrolet Nova	3,955	19	Domestic
Fiat Strada	4,296	21	Foreign
Pont. Sunbird	4,172	24	Domestic

Using Macro Variables in SAS

- Syntax

  %LET macro-variable-name = value;

- An Example of a macro variable

  %let var_list = price mpg;

- An example of using a macro variable in a SAS codes.

Table 7. Using macro variables in SAS codes

Without using macro variables	Using macro variables
title “using a macro variable in SAS";	title “using a macro variable in SAS";
proc means data=auto n mean min max maxdec=1;	proc means data=auto n mean min max maxdec=1;
var price mpg;	var &var_list;
run;	run;

Using Macro Variables in Stata

- Syntax

  local macro_name [=exp]

  global macro_name [=exp |

- An example of using a macro variable

Table 8. Using macro variables in Stata codes		
Without using macro variables		Using single macro variables
display “using a macro variable in Stata"		display “using a macro variable in Stata"
sum price mpg		local var_list = “price mpg”
		global var_list = “price mpg”
		display “using a macro variable in Stata"
		sum `var_list’ /*use local macro*/
		sum $var_list /*use global macro*/

Writing Macros in SAS

- Macros start with a %macro statement and ends with a %mend statement.
- Syntax

  %MACRO macro-name (parameter-1, parameter-2,… parameter-n);

  macro-text

  %MEND macro-name;
- Execute the macro

  %macro-name

Using Macros in SAS, Without Specifying Parameters

- Writing a macro:

```
%MACRO printit;
PROC PRINT DATA = auto;
TITLE ‘Listing the values of four variables';
VAR make model price mpg foreign;
RUN;
%MEND printit;
```

- Run the macro

```
%printit
```

- Run the macro after sorting the data by price

```
PROC SORT DATA = models;
BY Price;
%printit
```

Using Macros in SAS, With Specifying Parameters

- Writing a macro:

```
%MACRO s_reg (dep =, ind1= );
PROC  REG DATA = auto;
TITLE ‘‘using macros in regression analysis with specifying parameters”;
MODEL  &dep = &ind1;
RUN;
%MEND s_reg;
```

- Run the macro
- %s_reg (dep=price, ind1=mpg)

Using Macros in Stata, Without Specifying Parameters

- Writing a program:

  use auto.dta, clear
  program printit
  display “Listing the values of four variables”
  list make_model price mpg foreign
  end

- Run the macro

  printit

- Run the macro after sorting the data by price

  sort price
  printit

Using Macros in Stata, With Specifying Parameters

- Writing a macro:

  display "using macros in regression analysis with specifying parameters"

  program s_reg

  reg `1’ `2’

  end

Run the macro

  s_reg price mpg

Reminders of Using Arrays and Macros

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

Conclusions

- Using arrays and macros can save time and efforts. For example, with arrays and macro variables, you don't need to keep retyping the names of variables. Similarly, you don't need to retype codes for analyzing data if you use macros.
- By redefining arrays and macro variables, you can run the same analyses on different sets of variables.
- When putting a series of analyses into a macro, you can conduct these analyses with one command line.
- 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, SAS now has Proc Transpose command and Stata has reshape command for this purpose.
- Be patient. You will run into many errors when you start writing arrays or macros, but practice makes perfect.