

Paper SBC-123

Data Manipulations Using Arrays and DO Loops

Patricia Hall and Jennifer Waller, Medical College of Georgia, Augusta, GA

ABSTRACT

Using public databases for data analysis presents a unique set of challenges. Although governmental databases contain enormous amounts of useful information, they are not necessarily optimal for efficient data analyses. Since these databases are often maintained by various federal, state, or local agencies, they are not always structurally compatible and must be manipulated to merge with other data sets and for data analysis. One example of an administrative data set is the publicly available Georgia's Physician Workforce database, which gives physician counts for each of 61 physician specialties across the columns and a separate row for each county/race combination resulting in multiple records for each county represented across several rows of data. To simplify analysis, and to maintain structural consistency with other county-level databases, it may be desired to collapse the data into one record per county without losing information, such as the racial delineations in this case. To solve this problem, ARRAY's and DO loops can be used in the DATA step in SAS® to create new variables for each race/specialty combination. The resulting structure will be a single record for each county which preserves all of the original information and allows simple incorporation of county-level data from other databases.

INTRODUCTION

With the increasing availability of public information on the Internet, it is easy to quickly access data needed for reference or analysis. Federal, state, and local agencies are making data available via simple downloads, or by online purchase. Moreover, many websites offer the ability to obtain customized data sets by allowing the user to choose particular demographic groups broken down into desired sub-categories. However, this convenience comes with certain challenges with respect to data analysis. Database structures vary a great deal since they often reflect the needs and thought processes of individual organizations. Medical data is often aggregated to protect the identity of individuals. Additionally, raw public data are not always in the form needed to import into statistical software. Often the data structure must be manipulated to get it into a format required by the particular software analysis package being used. Finally, even if the data can be imported easily, the structure of the data may not be optimal for the analysis at hand.

Frequently, data from several databases are needed for a single analysis. When this is the case, it is very important to examine the data sets for differences before attempting to combine them to avoid interface errors and invalid statistical inferences. In most cases, data preparation is necessary before merging to ensure a smooth integration process. For instance, one dataset may have a single record for each individual where another dataset may have multiple records per person. If all of the data is not structurally compatible, modifications must be done to one or more datasets before they are combined.

In our case, several public and administrative data sources were used to address questions for a single project. Many of the datasets had aggregate data at the county level. However, the primary database used in our analyses was the Georgia Physician's Workforce Database which gives physician counts for each of 61 medical specialties for each county broken down further by race. This incompatibility between databases presented a challenge since we needed to merge the physician workforce data, which had 4 records per county, with other data aggregated at the county level, to do much of our analyses. It was apparent that collapsing the physician workforce data into a single record per county was necessary; however, since we wanted to look at race as a factor in several of our analyses, we did not want to lose the racial delineation information. We had to find a way to reorganize our data into one record per county without losing this information. To solve this problem, ARRAY's and DO loops were used in the DATA step in SAS® to create new variables for each race/specialty combination. The resulting structure was a single record for each county which preserves all of the original information and allows simple integration with county-level data from other databases.

WHAT THE DATA SET LOOKED LIKE BEFORE IMPORTING

The Georgia Physician's Workforce data was received as an Excel spreadsheet in the format seen below. Each county (left column) was divided into up to four race divisions from White, Black, Asian, and Other. If a race group was not represented by the physicians in the county, the row was omitted. Counts for each of 61 physician specialties (column headings) were given for each county/race combination. Generic specialty names are used in this paper to simplify the explanation. (Not all columns are shown.)

The screenshot shows an Excel spreadsheet titled "Microsoft Excel - original data". The data is organized as follows:

County	Race	Specialty1	Specialty2	Specialty3	Specialty4	Specialty5	Specialty6	Specialty7	Specialty8	Specialty9	Specialty10	Specialty11	Specialty12	Specialty13
Appling	White				0						1		1	
	Black				0						0		1	
	Asian				2						0		0	
	Other				0						0		0	
Atkinson	Total				2						1		2	
	White													
Bacon	Black						0							
	Asian						1							
	Other						0							
	Total						1							
Baker	White													
	Black													

In order to bring the data into SAS, it had to be prepared for the import process by making the data conform to the standard structural convention of each horizontal row representing a record. First, it was necessary to unmerge the cells with the county name (column A) and create a duplicate county name field for each row. Column B also included merged cells, but since this column was unnecessary, it was deleted. The last row for each county (Total) was deleted since it gave redundant information. For counties that did not have a row for every race group (Black, White, Asian, Other), additional rows were created so that each county consistently had 4 records. After these manipulations, the data was in the following form, still in Excel.

The screenshot shows an Excel spreadsheet titled "Microsoft Excel - data_modified". The data is organized as follows:

County	Race	Specialty1	Specialty2	Specialty3	Specialty4	Specialty5	Specialty6	Specialty7	Specialty8	Specialty9	Specialty10	Specialty11	Specialty12	Specialty13	Sp
Appling	White				0						1		1	2	
Appling	Black				0						0		1	0	
Appling	Asian				2						0		0	0	
Appling	Other				0						0		0	1	
Atkinson	White														
Atkinson	Black														
Atkinson	Asian														
Atkinson	Other														
Bacon	White							0						2	
Bacon	Black							1						1	
Bacon	Asian														
Bacon	Other							0						1	
Baker	White														
Baker	Black														
Baker	Asian														
Baker	Other														

At this point, the data was imported into SAS.

Below is the data set after being imported into SAS. As shown, there are four records per county: one for each of the four race groups (White, Black, Asian, Other).

	County	Race	Specialty1	Specialty2	Specialty3	Specialty4	Specialty5	Specialty6	Specialty7	Specialty8	Specialty9	Specialty10	Specialty11
1	Appling	White	.	.	.	0	.	.	.	.	.	1	.
2	Appling	Black	.	.	.	0	.	.	.	.	.	0	.
3	Appling	Asian	.	.	.	2	.	.	.	.	.	0	.
4	Appling	Other	.	.	.	0	.	.	.	.	.	0	.
5	Atkinson	White	.	.	.	.	.	.	.	.	.	.	.
6	Atkinson	Black	.	.	.	.	.	.	.	.	.	.	.
7	Atkinson	Asian	.	.	.	.	.	.	.	.	.	.	.
8	Atkinson	Other	.	.	.	.	.	.	.	.	.	.	.
9	Bacon	White	.	.	.	.	.	0	.	.	.	.	.
10	Bacon	Black	.	.	.	.	.	1	.	.	.	.	.
11	Bacon	Asian	.	.	.	.	.	.	.	.	.	.	.
12	Bacon	Other	.	.	.	.	.	0	.	.	.	.	.
13	Baker	White	.	.	.	.	.	.	.	.	.	.	.
14	Baker	Black	.	.	.	.	.	.	.	.	.	.	.
15	Baker	Asian	.	.	.	.	.	.	.	.	.	.	.
16	Baker	Other	.	.	.	.	.	.	.	.	.	.	.
17	Baldwin	White	1	.	.	1	.	1	.	.	.	1	.
18	Baldwin	Black	0	.	.	0	.	0	.	.	.	0	.
19	Baldwin	Asian	1	.	.	1	.	0	.	.	.	0	.
20	Baldwin	Other	0	.	.	0	.	1	.	.	.	0	.
21	Banks	White	.	.	.	.	.	.	.	.	.	.	.
22	Banks	Black	.	.	.	.	.	.	.	.	.	.	.
23	Banks	Asian	.	.	.	.	.	.	.	.	.	.	.
24	Banks	Other	.	.	.	.	.	.	.	.	.	.	.
25	Barrow	White	.	1	.	1	.	.	.	.	.	.	.
26	Barrow	Black	.	0	.	0	.	.	.	.	.	.	.
27	Barrow	Asian	.	0	.	0	.	.	.	.	.	.	.
28	Barrow	Other	.	.	.	.	.	.	.	.	.	.	.
29	Bartow	White	.	.	.	2	.	2	1	.	.	1	.
30	Bartow	Black	.	.	.	0	.	0	0	.	.	1	.
31	Bartow	Asian	.	.	.	0	.	0	0	.	.	0	.
32	Bartow	Other	.	.	.	0	.	0	0	.	.	0	.
33	Ben Hill	White	.	.	.	1	.	.	.	.	.	0	.

Since the analysis at hand required the merging of this data with other datasets with a single record per county, the issue of multiple records per county had to be addressed. Collapsing the four records into one would not be difficult, but the information at the race level would be lost. The challenge was to manipulate the data to get one record per county without losing any information. To do this, a new variable was created for each race/specialty combination. For example, for the first physician specialty (Specialty1), the variables Specialty1_white, Specialty1_black, Specialty1_asian, and Specialty1_other were created for each county; for the second physician specialty (Specialty2), Specialty2_white, Specialty2_black, Specialty2_asian, and Specialty2_other were created, etc. The result was a single record for each county with 4 race-specific variables for each physician specialty category. The total number of new variables was 61 (the original number of physician specialties) X 4 = 244 race-specific physician specialty categories.

CREATING SEPARATE DATASETS AND NEW RACE-SPECIFIC VARIABLES

To create the race-specific variables for each of the 61 physician specialties, the original data set had to be filtered into subsets based on race, one at a time. To begin the process of creating 4 separate race-specific datasets, an IF statement was used to create a subset of the original dataset which only included the records for the white race.

```
data white;
  set all;
  if race='White';
```

Using the filtered dataset “white,” each of the 61 physician specialty variables (Specialty1, Specialty2, Specialty3, ..., Specialty61) was renamed with the race-specific suffix “_white” (i.e., Specialty1_white, Specialty2_white, Specialty3_white, ..., Specialty61_white) by creating two arrays: an array called “old” consisting of the original 61 specialty names (the column headings of the original data) and an array called “new” with new variable names with the race-specific suffix. Below is the SAS code.

```
array old Specialty1 Specialty2 Specialty3 ... Specialty61;

array new Specialty1_white Specialty2_white Specialty3_white ...
      Specialty61_white;
```

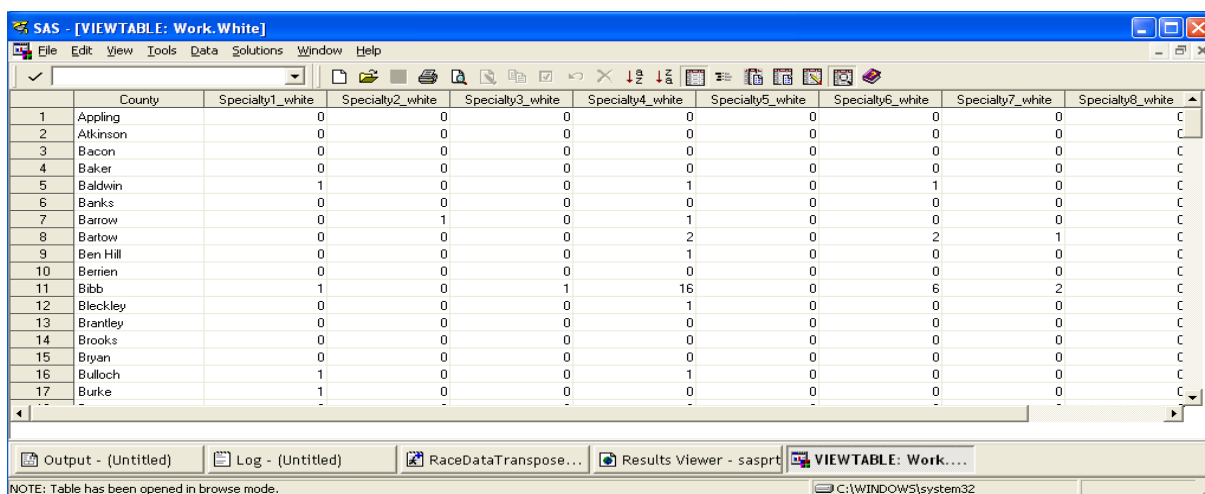
A DO OVER loop was then used to rename the variables in the array named “old” with the new variable names in the array named “new.” So, Specialty1 became Specialty1_white, Specialty2 became Specialty2_white, Specialty3 became Specialty3_white, etc. An IF statement was used to fill any empty fields with a zero.

```
do over old;
  new=old;
  if new=. then new=0;
end; run;
```

Finally, a KEEP statement was used to retain only the new variable names in the dataset.

```
keep County Specialty1_white Specialty2_white Specialty3_white ...
      Specialty61_white; run;
```

The resulting dataset is shown below.



	County	Specialty1_white	Specialty2_white	Specialty3_white	Specialty4_white	Specialty5_white	Specialty6_white	Specialty7_white	Specialty8_white
1	Appling	0	0	0	0	0	0	0	0
2	Atkinson	0	0	0	0	0	0	0	0
3	Bacon	0	0	0	0	0	0	0	0
4	Baker	0	0	0	0	0	0	0	0
5	Baldwin	1	0	0	1	0	1	0	0
6	Banks	0	0	0	0	0	0	0	0
7	Barrow	0	1	0	1	0	0	0	0
8	Bartow	0	0	0	2	0	2	1	0
9	Ben Hill	0	0	0	1	0	0	0	0
10	Berrien	0	0	0	0	0	0	0	0
11	Bibb	1	0	1	16	0	6	2	0
12	Bleckley	0	0	0	1	0	0	0	0
13	Brantley	0	0	0	0	0	0	0	0
14	Brooks	0	0	0	0	0	0	0	0
15	Bryan	0	0	0	0	0	0	0	0
16	Bulloch	1	0	0	1	0	0	0	0
17	Burke	1	0	0	0	0	0	0	0

This race-specific renaming process for the specialty variables was repeated for the remaining three race categories (Black, Asian, and Other). The result was 4 separate datasets (White, Black, Asian, and Other), each with 61 specialty names with the race-specific suffix in the variable name. The four datasets were then merged to create a single (wide) dataset with all 244 (61 X 4) race-specific variable names for the 61 specialties.

MERGING THE 4 DATASETS

The four separate race-specific datasets needed to be merged into one to complete the process. First, each separate dataset was sorted by county using PROC SORT. Then, the four datasets were merged by county using a simple MERGE statement.

```
proc sort data=white; by county; run;
proc sort data=black; by county; run;
proc sort data=black; by county; run;
proc sort data=black; by county; run;

data allrace;
  merge white black asian other;
  by county;
run;
```

Below is an abbreviated version of the merged data.

County	Specialties1-61 (white)					Specialties1-61 (black)					Specialties1-61 (asian)					Specialties1-61 (other)				
	Specialty 1 white	Specialty 2 white	Specialty 3 white	...	Specialty 61 white	Specialty 1 black	Specialty 2 black	Specialty 3 black	...	Specialty 61 black	Specialty 1 asian	Specialty 2 asian	Specialty 3 asian	...	Specialty 61 asian	Specialty 1 other	Specialty 2 other	Specialty 3 other	...	Specialty 61 other
Appling	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Atkinson	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Bacon	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Baker	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Baldwin	1	0	0	...	0	0	0	0	...	0	1	0	0	...	0	0	0	0	...	0
Banks	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Barrow	0	1	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Bartow	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Ben Hill	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Berrien	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Bibb	1	0	1	...	0	1	0	1	...	0	0	0	1	...	0	0	0	0	...	0
Bleckley	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Brantley	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0
Brooks	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0	0	0	0	...	0

The result is a single record per county which preserves the original racial delineations and related information. This allows for integration with other county level data.

CONCLUSION

Integrating data from public databases can present challenges due to differences in the structure of data from various sources. When combining data with differing amounts of detail, multiple records in one dataset may need to be collapsed into one to facilitate merging with other data. In this case, information lost as a result of the process of collapsing can be preserved by using a sub-setting IF to filter the data into groups which allow the subsequent use of ARRAYS and DO Loops to create new variables which retain the potentially lost information.

The complete code related to this paper is given in the Appendix.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Patricia Hall
Department of Biostatistics
Medical College of Georgia
Augusta, GA 30912-4900
(706) 721-2947
pthall@mcg.edu

Jennifer L. Waller
Department of Biostatistics
Medical College of Georgia
Augusta, GA 30912-4900
(706) 721-0814
jwaller@mcg.edu

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. In the USA and other countries, ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

APPENDIX

```
/*IMPORT DATA INTO SAS*/
proc import out= work.race
  datafile= "path to sasdatasetname"
  DBMS=EXCEL REPLACE;
  SHEET="Sheet$";
  GETNAMES=YES;
  MIXED=NO;
  SCANTEXT=YES;
  USEDATE=YES;
  SCANTIME=YES;
run;

/*CREATE FILTERED DATASET WITH ONLY WHITE PHYSICIAN RECORDS*/
data white;
  set sasdatasetname;
  if race='White';

/*RENAME OLD PHYSICIAN SPECIALTY VARIABLES WITH RACE-SPECIFIC SUFFIX "_white"
USING ARRAYS*/
array old Specialty1 Specialty2 Specialty3 .... Specialty61;

array new Specialty1_white Specialty2_white Specialty3_white ....
Specialty61_white;

/*ASSIGNING NEWLY CREATED VARIABLES TO VALUES OF OLD VARIABLE NAMES*/
/*POPULATING EMPTY FIELDS WITH ZEROES*/
do over old;
  new=old;
  if new=. then new=0;
end;

/*KEEPING ONLY NEWLY CREATED VARIABLES (DROPPING OLD VARIABLE NAMES)*/
keep County Specialty1_white Specialty2_white Specialty3_white ....
Specialty61_white;
run;

/*CREATE FILTERED DATASET WITH ONLY BLACK PHYSICIAN RECORDS*/
data black;
  set sasdatasetname;
  if race='Black';

/*RENAME OLD PHYSICIAN SPECIALTY VARIABLES WITH RACE-SPECIFIC SUFFIX "_black"
USING ARRAYS*/
array old Specialty1 Specialty2 Specialty3 .... Specialty61;

array new Specialty1_black Specialty2_black Specialty3_black ....
Specialty61_black;
```

```

/*ASSIGNING NEWLY CREATED VARIABLES TO VALUES OF OLD VARIABLE NAMES*/
/*POPULATING EMPTY FIELDS WITH ZEROES*/
do over old;
  new=old;
  if new=. then new=0;
end;

/*KEEPING ONLY NEWLY CREATED VARIABLES (DROPPING OLD VARIABLE NAMES)*/
keep County Specialty1_black Specialty2_black Specialty3_black ....
Specialty61_black;
run;

/*CREATE FILTERED DATASET WITH ONLY ASIAN PHYSICIAN RECORDS*/
data asian;
  set sasdatasetname;
  if race='Asian';

/*RENAME OLD PHYSICIAN SPECIALTY VARIABLES WITH RACE-SPECIFIC SUFFIX "_asian"
USING ARRAYS*/
array old Specialty1 Specialty2 Specialty3 .... Specialty61;

array new Specialty1_asian Specialty2_asian Specialty3_asian ....
Specialty61_asian;

/*ASSIGNING NEWLY CREATED VARIABLES TO VALUES OF OLD VARIABLE NAMES*/
/*POPULATING EMPTY FIELDS WITH ZEROES*/
do over old;
  new=old;
  if new=. then new=0;
end;

/*KEEPING ONLY NEWLY CREATED VARIABLES (DROPPING OLD VARIABLE NAMES)*/
keep County Specialty1_asian Specialty2_asian Specialty3_asian ....
Specialty61_asian;
run;

/*CREATE FILTERED DATASET WITH ONLY OTHER RACE PHYSICIAN RECORDS*/
data other;
  set sasdatasetname;
  if race='Other';

/*RENAME OLD PHYSICIAN SPECIALTY VARIABLES WITH RACE-SPECIFIC SUFFIX "_other"
USING ARRAYS*/
array old Specialty1 Specialty2 Specialty3 .... Specialty61;

array new Specialty1_other Specialty2_other Specialty3_other ....
Specialty61_other;

```

```

/*ASSIGNING NEWLY CREATED VARIABLES TO VALUES OF OLD VARIABLE NAMES*/
/*POPULATING EMPTY FIELDS WITH ZEROES*/
do over old;
  new=old;
  if new=. then new=0;
end;

/*KEEPING ONLY NEWLY CREATED VARIABLES (DROPPING OLD VARIABLE NAMES)*/
keep County Specialty1_other Specialty2_other Specialty3_other ....
Specialty61_other;
run;

/*SORTING EACH NEWLY CREATED RACE-SPECIFIC DATASET BY COUNTY*/
proc sort data=white; by county; run;
proc sort data=black; by county; run;
proc sort data=asian; by county; run;
proc sort data=other; by county; run;

/*MERGING ALL FOUR RACE-SPECIFIC DATASETS INTO ONE*/
data allrace;
merge white black asian other;
by county;
run;

run;
quit;
run;

```