Turning output into data, PROC TRANSPOSE, PROC EXPORT

Shannon Pileggi

STAT 330

OUTLINE

Overview

- For many procedures you can use an `OUTPUT` statement or an `OUT =` option to store printed output to a SAS data set
- If that option isn't available, you can all use the `ODS` system to obtain the output
- This can be helpful for assembling statistics from many pieces of output, either for a succinct table or for a figure

Data

SAS Code

```
PROC IMPORT OUT = WORK.ACS DATAFILE = "&path.acs.csv"
            DBMS = CSV REPLACE;
     GETNAMES = YES;
     DATAROW = 2;
     GUESSINGROWS = 1000 ;
RUN;
```

SAS Code

Obs	Sex	Age	MarStat	Income	HoursWk	Race	USCitizen	HealthInsurance	Language
1	female	31	not married	60	40	white	citizen	yes	other
2	male	31	not married	0.36	12	black	citizen	yes	native English
3	male	75	not married	0	.	white	citizen	yes	native English
4	female	80	not married	0	.	white	citizen	yes	native English
5	male	64	married	0	.	white	citizen	yes	native English
6	male	14	not married	.	.	white	citizen	yes	native English
7	male	78	married	0	.	white	citizen	yes	native English
8	male	35	not married	87	40	white	citizen	yes	other
9	female	70	married	0	1	white	citizen	yes	native English
10	female	18	not married	0	.	white	citizen	yes	native English

Storing output from PROC MEANS

SAS Code

```
PROC MEANS DATA = WORK.ACS ;
   OUTPUT OUT = meansresults ;
RUN ;
```

SAS Code

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	1000	40.1340000	23.1552199	0	130.0000000
Income	825	22.3962303	42.9056463	0	563.0000000
HoursWk	505	37.1128713	14.3210472	1.0000000	99.0000000

SAS Code

```
PROC PRINT DATA = meansresults ;
RUN ;
```

SAS Code

Obs	_TYPE_	_FREQ_	_STAT_	Age	Income	HoursWk
1	0	1000	N	1000	825	505
2	0	1000	MIN	0	0	1
3	0	1000	MAX	130	563	99
4	0	1000	MEAN	40.134	22.396230303	37.112871287
5	0	1000	STD	23.155219891	42.905646314	14.321047222

Storing output from PROC FREQ

SAS Code

```
PROC FREQ DATA = WORK.acs ;
   TABLES sex*marstat /
        OUT = freqresults ;
RUN ;
```

SAS Code

SAS Code

```
PROC PRINT DATA = freqresults ;
RUN ;
```

SAS Code

Frequency Percent Row Pct Col Pct	Table of Sex by MarStat		
	MarStat		
Sex	married	not married	Total
female	221 22.10 41.15 50.57	316 31.60 58.85 56.13	537 53.70
male	216 21.60 46.65 49.43	247 24.70 53.35 43.87	463 46.30
Total	437 43.70	563 56.30	1000 100.00

Obs	Sex	MarStat	COUNT	PERCENT
1	female	married	221	22.1
2	female	not married	316	31.6
3	male	married	216	21.6
4	male	not married	247	24.7

Storing output from PROC TTEST

PROC TTEST does not have an `OUTPUT` statement or an `OUT =` option! To store results, we need to use ODS.

1. *Trace* your results from a SAS procedure.
2. Identify the output that you want from your log window.
3. Modify your SAS procedure code to use ODS to store your results.

Step 1 - trace your results

SAS Code

```
ODS TRACE ON ;
PROC TTEST DATA = WORK.acs H0 = 40 ;
   VAR HoursWk ;
RUN ;
ODS TRACE OFF ;
```

SAS Code

The TTEST Procedure

Variable: HoursWk

N	Mean	Std Dev	Std Err	Minimum	Maximum
505	37.1129	14.3210	0.6373	1.0000	99.0000

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
37.1129	35.8608	38.3649	14.3210	13.4889	15.2634

DF	t Value	Pr > |t|
504	-4.53	<.0001

Step 2 - examine log

There are five pieces of output that I can grab! (4 shown)

From SAS log

```
Output Added:
-------------
Name:         Statistics
Label:        Statistics
Template:     Stat.TTest.Statistics
Path:         Ttest.HoursWk.Statistics
-------------

Output Added:
-------------
Name:         ConfLimits
Label:        Confidence Limits
Template:     Stat.TTest.ConfLimits
Path:         Ttest.HoursWk.ConfLimits
-------------
```

From SAS log

```
Output Added:
-------------
Name:         TTests
Label:        T-Tests
Template:     Stat.TTest.TTests
Path:         Ttest.HoursWk.TTest
-------------

Output Added:
-------------
Name:         SummaryPanel
Label:        Summary Panel
Template:     Stat.TTest.Graphics
Path:         Ttest.HoursWk.Summa
-------------
```

From SAS log

Step 3 - use ODS to store results

SAS Code

```
PROC TTEST DATA = WORK.acs H0 = 40 ;
  VAR HoursWk ;
  ODS OUTPUT ConfLimits = CIresults ;
RUN ;
```

SAS Code

SAS Code

```
PROC PRINT DATA = CIresults;
RUN ;
```

SAS Code

The TTEST Procedure

Variable: HoursWk

N	Mean	Std Dev	Std Err	Minimum	Maximum
505	37.1129	14.3210	0.6373	1.0000	99.0000

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
37.1129	35.8608	38.3649	14.3210	13.4889	15.2634

DF	t Value	Pr > |t|
504	-4.53	<.0001

Obs	Variable	Mean	LowerCLMean	UpperCLMean	StdDev	LowerCLStdDev	UpperCLStdDev
1	HoursWk	37.1129	35.8608	38.3649	14.3210	13.4889	15.2634

Obs	UMPULowerCLStdDev	UMPUUpperCLStdDev
1	13.4802	15.2531

Storing output from PROC MEANS

SAS Code

```
PROC MEANS DATA = WORK.ACS ;
   OUTPUT OUT = meansresults ;
RUN ;
```

SAS Code

SAS Code

```
PROC PRINT DATA = meansresults ;
RUN ;
```

SAS Code

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	1000	40.1340000	23.1552199	0	130.0000000
Income	825	22.3962303	42.9056463	0	563.0000000
HoursWk	505	37.1128713	14.3210472	1.0000000	99.0000000

Obs	_TYPE_	_FREQ_	_STAT_	Age	Income	HoursWk
1	0	1000	N	1000	825	505
2	0	1000	MIN	0	0	1
3	0	1000	MAX	130	563	99
4	0	1000	MEAN	40.134	22.396230303	37.112871287
5	0	1000	STD	23.155219891	42.905646314	14.321047222

On your own: How can we get our meansresults to look like the original output?

PROC TRANSPOSE

SAS Code

```
PROC TRANSPOSE
   DATA = DataSetName OUT = TransposedData NAME = NewVariableName;
   ID IdentifyingVariable ;
   BY GroupingVariable ;
   VAR var1 var2 var3 ;
RUN;
```

SAS Code

- `DATA=` specifies the SAS data set name you want to transpose
- `OUT=` name of new transposed data set
- `NAME=` names a variable in *TransposedData* that contains `VAR` variables
- `ID` a variable in *DataSetName* that names multiple variables in *TransposedData*
- `BY` transposes by groups (data must be pre-sorted)
- `VAR` lists variables to transpose

Transpose summary statistics

The SAS System

Obs	_TYPE_	_FREQ_	_STAT_	Age	Income	HoursWk
1	0	1000	N	1000	825	505
2	0	1000	MIN	0	0	1
3	0	1000	MAX	130	563	99
4	0	1000	MEAN	40.134	22.396230303	37.112871287
5	0	1000	STD	23.155219891	42.905646314	14.321047222

SAS Code

```
PROC TRANSPOSE
    DATA = meansresults
    OUT = statstransposed
    NAME = variable ;
    ID _stat_ ;
    VAR age income hourswk;
RUN ;
```

SAS Code

Obs	variable	N	MIN	MAX	MEAN	STD
1	Age	1000	0	130	40.134	23.155219891
2	Income	825	0	563	22.396230303	42.905646314
3	HoursWk	505	1	99	37.112871287	14.321047222

Discussion

Data we have

Student	ID	Test1	Test2
Andrew	6545	94	91
Beth	1252	51	65
Charlie	1167	95	97

Data we have

Data we want

exams	Andrew	Beth	Charlie
Test1	94	51	95
Test2	91	65	97

Data we want

SAS Code

```
PROC TRANSPOSE
    DATA = datawehave
    OUT = datawewant
    NAME =  1 ;
    ID      2 ;
    VAR     3 ;
run;
```

SAS Code

Fill in the SAS statements.

1.

2.

3.

Reshaping data

- Re-shaping data is commonly needed for longitudinal studies
- Long format is required for longitudinal data analysis

  Reshape data wide to long:
  https://stats.idre.ucla.edu/sas/modules/reshaping-data-wide-to-long-using-a-data-step/
- Wide format might be more useful for creating graphics or for paired analysis

  Reshape data long to wide:
  https://stats.idre.ucla.edu/sas/modules/how-to-reshape-data-long-to-wide-using-proc-transpose/

PROC EXPORT

SAS Code

```
PROC EXPORT
    DATA = DataSetName
    OUTFILE = "Computer Location/mydata.ext"
    DBMS = identifier
    REPLACE ;
RUN;
```

SAS Code

`DATA=` specifies the SAS data set name you want to export

`OUTFILE=` takes computer location, data file name, and extension of the data file

`DBMS=` specifies type of data (e.g., `CSV`, `TAB`, `DLM`)

`REPLACE` option overwrites an existing file called *mydata.ext*

Example Code

SAS Code

```
DATA statstransposed2 ;
   SET statstransposed ;
   FORMAT mean 4.1 std 4.1 ;
RUN ;

PROC EXPORT
   DATA = statstransposed2
   OUTFILE = "&path.summarystats.csv"
   DBMS = CSV
   REPLACE ;
RUN;
```

SAS Code

- any format assigned to variables in the DATA step will be applied in the exported file for `.csv` or `.txt` files
- exporting to `.xls` or `.xlsx` does not apply formats