Paper 087-2008

Any WAY you Want it: Getting the Right TYPEs of Observations Out of PROC SUMMARY or MEANS

Christianna S. Williams, Abt Associates Inc., Durham, NC

ABSTRACT

Have you used PROC MEANS or PROC SUMMARY and wished there was something intermediate between the NWAY option (which produces output only at the highest value of _TYPE_ -- the lowest level of summarization) and the default of getting all existing levels of combinations of the CLASS variables? Well, somebody at SAS® thought of this too, and brought us the TYPES and WAYS statements. This paper will explain how to use these two statements to customize what goes into your OUT= data set more efficiently than another PROC MEANS (or SUMMARY), a DATA step or even using a WHERE clause on the OUTPUT statement. Using these statements alone or together you can get exactly the combinations you need for further processing or reporting.

INTRODUCTION

Often the first step in analyzing a complex data set is descriptive analyses for key variables, which are frequently those variables that will become the dependent or outcome variables in later more complex analyses, such as linear regression or one of its many variants. In the early descriptive phase, commonly the purpose is to get a sense of the extent to which those outcome variables differ (or not) among subgroups within the sample, such as gender or race or whatever is of interest in the particular field. In a longitudinal study, one might also be interested in describing the pattern the outcome variables have over time and/or summarizing to the level of the individual to evaluate the variability among subjects.

The twin SAS procedures SUMMARY and MEANS are well suited to this type of stratified descriptive analyses, and typically the outcomes of interest are specified in the VAR statement and the classification variables (which define the subgroups of interest) are specified in the CLASS statement. However, in the case of multiple classification variables, the default output generated by these procedures may well include many observations that are not of interest. Specifically, PROC MEANS or SUMMARY will produce a data set with one observation at all possible levels of aggregation of the class variables, from the overall summary (across all observations in the input data set) down to the completely stratified case (i.e. summarizing only across observations with identical values on all the specified CLASS variables). For example, if there are 3 CLASS variables, each with 2 levels, (and assuming at least one statistic keyword is specified on the OUTPUT statement) by default the output data set will have 27 observations as follows: 1 for the overall (summarized across all levels of all class variables); 3*2=6 observations for which one CLASS variable is at a specified level but where data are summarized across the other two CLASS variables; 3*2*2 = 12 observations for which a level is specified for two CLASS variables but summarized across the third; and 2*2*2=8 observations for which a level is specified for all three CLASS variables. At the other extreme, if the NWAY option is specified on the PROC statement, the output data set will include only 8 observations – just those for which a level is specified for all of the class variables.

It may well be that you are not interested in all the combinations produced by the default but need something different from what NWAY provides – this is where the TYPES and WAYS statements come in. While other methods, such as using separate PROC SUMMARY/MEANS steps for each CLASS variable, post-processing in a DATA step, or using a WHERE clause for the OUT= data set in PROC SUMMARY/MEANS, may achieve the same result, they are likely less efficient (at least in terms of quantity of SAS code!). This is the job that TYPES and WAYS are built for, so they are quite flexibly suited to this task…all one has to do is figure out how they work, which is the purpose of this paper.

DEFINITIONS

Part of the challenge of using the WAYS and TYPES statements effectively (at least for me) is getting a handle on what these words mean in the SAS language (in contrast to the English language). So, let's start there.

WAYS

According to the online documentation for SAS 9.1 (1), the WAYS statement "specifies the number of ways to make unique combinations of class variables". Personally, I don't find this all that helpful. The documentation goes on to say that a "list" is the only required argument to the WAYS statement (in fact, it's the only available argument) and that this list "specifies one or more integers that define the number of class variables to combine to form all the unique combinations of class variables. For example, you can specify 2 for all possible pairs and 3 for all possible triples." This is decidedly more helpful, and it starts to make sense that we can request "two-way" and "three-way" combinations, but some very concrete examples will help. We'll come to that soon.

TYPES

With respect to the TYPES statement, the SAS documentation offers that this statement "identifies which of the possible combinations of class variables to generate." Additionally, the only (and required) argument is to specify one

or more "request(s)" which are "composed of one class variable name, several class variable names separated by asterisks, or ()". The latter rather cryptic syntax will yield the overall total. I think what is a little confusing about this is that the way in which the types of observations desired is specified in the TYPES statement doesn't correspond all that well to the content of the _TYPE_ automatic variable that identifies observations in the output data set. Now, we really need some examples. One other note before we get there: the SAS documentation also offers the following "Tip" regarding the TYPES statement: "If you do not need all types in the output data set, then use the TYPES statement to specify particular subtypes rather than applying a WHERE clause to the data set. Doing so saves time and computer memory." That sounds promising…now we just have to figure out how to use it.

THE AGITATION DATA SET

The data set that I'll use for all the examples has been modified from data collected in a study of the effect of an environmental lighting treatment on agitated behaviors in elderly nursing home residents with dementia(2). These types of behaviors, which include a range from wandering and pacing to physical aggression to crying uncontrollably, are unfortunately quite common in persons with dementia and are distressing for patients as well as those – whether family or paid staff – who care for them. The data set is in a library with the libref EX. What you need to know about the variables on this data set for these examples is summarized in Table 1.

Table 1. Description of variables in EX.AGITATION data set

Variable name	Description/Label	Coding information
RESID	Resident ID	5-digit resident identifier
GENDER	Resident gender	M=Male, F=Female
SITE	Study Site	1=North Carolina, 2=Oregon
COGSTAT	Severity of dementia	1=Mild/Moderate, 2=Severe
AGITATION	Agitation score(3)	Continuous, range 14-60

EXAMPLE 1: DEFAULT RESULTS

As noted in the introduction, the default output data set from PROC SUMMARY (or PROC MEANS) with three CLASS variables each with two levels (and assuming that there is at least one observation in the input data set for all possible combinations of the three CLASS variables) will have 27 observations. (Aside: if no output statistics are specified, the output data set will have 5 observations for each of the above combinations of CLASS variables – here 5*27=135 observations – one for each of 5 default statistics: N, MEAN, STD, MIN and MAX. I'll not deal further with this case in this paper).

I illustrate the code below. Note that for ALL the examples in this paper I am using PROC MEANS with the NOPRINT option; however, the results are absolutely identical if you use PROC SUMMARY (NOPRINT is the default). I chose to use PROC MEANS in this paper (even though PROC SUMMARY was originally developed for the purpose of creating output data sets as I am in all these examples) because virtually all the SAS documentation about these procedures – including for the TYPES statement and the WAYS statement – is now presented under the MEANS procedure.

Example 1. Code:

```
PROC MEANS DATA = ex.agitation NOPRINT;
CLASS gender site cogstat ;
VAR agitation ;
OUTPUT OUT=sum1 MEAN=  /WAYS;
RUN;
```

The output (a PROC PRINT of the SUM1 data set) is shown below (Example 1. Output). Because I used the MEAN= syntax on the OUTPUT statement, the name of the input analysis variable (AGITATION) is assigned to the MEAN of that variable in the output data set. Also, the WAYS option on the OUTPUT statement puts the _WAY_ variable on the output data set; the _TYPE_ variable (and _FREQ_ , which gives the number of observations in the input data set that are summarized in that observation of the output data set) are there by default. As noted above, there is one observation (_TYPE_ = 0; _WAY_=0) for the overall mean; 6 observations in which a level is specified for only one of the three CLASS variables (_TYPE_ = 1, 2 and 4; _WAY_ = 1); 12 observations in which a level is specified for two of the CLASS variables (_TYPE_ = 3,5,6; _WAY_ = 2); and 8 observations at the finest grain – where all three CLASS variables are at specific levels (_TYPE_ = 7; _WAY_ = 3). Note that the observations in this output data set for which one or more of the CLASS variables are missing have nothing to do with missing data in the input file. Instead, they indicate that those variables are ignored for the descriptive statistics presented in that row. For example, where GENDER is blank (missing) in the output data set, it indicates that the agitation MEAN shown is across both men and women. I use the WAYS options because the _WAY_ variable is more intuitive to me than the _TYPE_ variable – the values of _WAY_ correspond to the number of dimensions of the descriptive analysis for that observation; that is,

"one-way", "two-way" (think two-way table), "three-way", and so on. Luckily, however, you don't have to figure out the _TYPE_ variable to use the TYPES statement effectively, as we'll see.

Example 1. Output

Obs	gender	site	cogstat	_WAY_	_TYPE_	_FREQ_	agitation
1		.	.	0	0	432	21.3102
2		.	1	1	1	220	19.6636
3		.	2	1	1	212	23.0189
4		1	.	1	2	285	22.3544
5		2	.	1	2	147	19.2857
6		1	1	2	3	163	20.2209
7		1	2	2	3	122	25.2049
8		2	1	2	3	57	18.0702
9		2	2	2	3	90	20.0556
10	F	.	.	1	4	171	22.7193
11	M	.	.	1	4	261	20.3870
12	F	.	1	2	5	69	21.4638
13	F	.	2	2	5	102	23.5686
14	M	.	1	2	5	151	18.8411
15	M	.	2	2	5	110	22.5091
16	F	1	.	2	6	71	28.7746
17	F	2	.	2	6	100	18.4200
18	M	1	.	2	6	214	20.2243
19	M	2	.	2	6	47	21.1277
20	F	1	1	3	7	35	26.6571
21	F	1	2	3	7	36	30.8333
22	F	2	1	3	7	34	16.1176
23	F	2	2	3	7	66	19.6061
24	M	1	1	3	7	128	18.4609
25	M	1	2	3	7	86	22.8488
26	M	2	1	3	7	23	20.9565
27	M	2	2	3	7	24	21.2917

Before moving to the next example, I'll note that ALL the subsequent output data sets I create are some subset of the observations in this "default" data set, SUM1. Also, because I want to focus on the functioning of the TYPES and WAYS statements, as opposed to the many other capabilities of the MEANS procedure, I'm just using a single analysis variable on the VAR statement and requesting a single output statistic (the MEAN) on the OUTPUT statement. Nonetheless, all of these methods work the same if there are multiple analysis variables and/or multiple statistics requested. If the latter is the case, however, you need to specify names for the output variables containing the requested statistics (as opposed to the MEAN= syntax shown in these examples) or use the AUTONAME option on the OUTPUT statement (see references 1 and 4 for details).

EXAMPLE 2: NWAY OPTION

If all that is desired are the observations with the highest _WAY_ value (i.e. with all *n* CLASS variables at a specific level), you can specify the NWAY option (*Get it: "n-way")* on the PROC MEANS statement, as shown below

Example 2. Code

```
PROC MEANS DATA = ex.agitation NOPRINT NWAY;
CLASS gender site cogstat ;
VAR agitation ;
OUTPUT OUT=sum2 MEAN=  ;
RUN;
```

The output data set, SUM2, shown below, will be identical to the last 8 observations of SUM1, the output data set from Example 1, except that I've left the _WAY_ variable off (it would have a value of 3 for all observations).

Example 2. Output

Obs	gender	site	cogstat	_TYPE_	_FREQ_	agitation
1	F	1	1	7	35	26.6571
2	F	1	2	7	36	30.8333
3	F	2	1	7	34	16.1176
4	F	2	2	7	66	19.6061
5	M	1	1	7	128	18.4609
6	M	1	2	7	86	22.8488
7	M	2	1	7	23	20.9565
8	M	2	2	7	24	21.2917

EXAMPLE 3: ONE-WAY OUTPUT

If what we want is somewhere in between the results of Example 1 and Example 2, the WAYS and/or the TYPES statement can be very helpful. If, for instance, we just want to do our descriptive analyses for GENDER (regardless of SITE and COGSTAT), for SITE (regardless of GENDER and COGSTAT), and for COGSTAT (regardless of SITE and GENDER); that is, we want all the one-way analyses but only the one-way analyses, we can use the WAYS statement, as follows:

Example 3a. Code

```
PROC MEANS DATA = ex.agitation NOPRINT;
CLASS gender site cogstat ;
VAR agitation ;
WAYS 1;
OUTPUT OUT=sum3a MEAN=  ;
RUN;
```

The output data set, SUM3A, is shown below. We get the "one-way" analysis of each CLASS variable; that is, the description is broken out by one CLASS variable at a time, while the other two are ignored.

Example 3a. Output

Obs	gender	site	cogstat	_TYPE_	_FREQ_	agitation
1		.	1	1	220	19.6636
2		.	2	1	212	23.0189
3		1	.	2	285	22.3544
4		2	.	2	147	19.2857
5	F	.	.	4	171	22.7193
6	M	.	.	4	261	20.3870

Now, despite the fact that in Output 3a, the _TYPE_ variable indicates we are getting TYPES 1, 2 and 4 (refer back to the output from Example 1), we can use the TYPES statement to produce exactly the same result, but as alluded to above we don't need to know the _TYPE_ values. Instead, we directly specify the analyses we want. Here's the code:

Example 3b. Code

```
PROC MEANS DATA = ex.agitation NOPRINT;
CLASS gender site cogstat ;
VAR agitation ;
TYPES gender site cogstat;
OUTPUT OUT=sum3b MEAN=  ;
RUN;
```

This specifies that we want the one-way analyses for each of the three CLASS variables. The output is identical to that in Example 3a, so it is not repeated here.

EXAMPLE 4: ONE-WAY AND OVERALL OUTPUT

Commonly I want to get the one-way output as shown in Example 3, but I also want the overall description – that is, summarized across all the observations in the input data set, regardless of their values on the input data set. This is easily accomplished with a slight modification to the WAYS statement, as shown below:

Example 4a. Code

```
PROC MEANS DATA = ex.agitation NOPRINT;
CLASS gender site cogstat ;
VAR agitation ;
WAYS 0 1;
OUTPUT OUT=sum4a MEAN=  /WAYS;
RUN;
```

And, the output (Example 4a. Output), as we'd expect, has one more observation than that from Example 3.

Example 4a. Output

Obs	gender	site	cogstat	_WAY_	_TYPE_	_FREQ_	agitation
1		.	.	0	0	432	21.3102
2		.	1	1	1	220	19.6636
3		.	2	1	1	212	23.0189
4		1	.	1	2	285	22.3544
5		2	.	1	2	147	19.2857
6	F	.	.	1	4	171	22.7193
7	M	.	.	1	4	261	20.3870

I added the WAYS option to the code for this example (4a) so that we have the _WAY_ variable on the SUM4A data set, and we see that for the overall analysis (first observation), _WAY_ is 0, corresponding to our WAYS statement above. In contrast, we can get the same result using the _TYPES_ statement, but the request doesn't really jibe with the values of the _TYPE_ variable that we want. The code is shown in Example 4b, below:

Example 4b. Code

```
PROC MEANS DATA = ex.agitation NOPRINT;
CLASS gender site cogstat ;
VAR agitation ;
TYPES gender site cogstat ();
OUTPUT OUT=sum4b MEAN=  /WAYS;
RUN;
```

The output data set SUM4B is identical to SUM4A, and so is not replicated here. It is the symbol "()" on the TYPES statement in this example that gets us the overall analysis (_TYPE_ = 0, _WAY_ = 0) – that's one of those quirky things you just have to commit to memory.

EXAMPLE 5: FOCUSING ON ONE CLASS VARIABLE

The last two examples might lead one to believe that the TYPES and WAYS statements are redundant, in that we can achieve identical results with either one. Further, one might conclude that the TYPES statement is unnecessary because the same result can be obtained more simply with the WAYS statement. Well, one would be wrong. If you want a subset of analyses at one or more "WAY" levels, then the TYPES statement is indispensable. For instance, if you want to focus the descriptive analysis on a single CLASS variable, say GENDER, and you want all the analyses that are stratified by GENDER, either alone or in combination with the other CLASS variables, this cannot be achieved with just the WAYS statement (as we want a *subset* of one-way, two-way and three-way analyses). But, such an analysis is quite straightforward using TYPES:

Example 5. Code

```
PROC MEANS DATA = ex.agitation NOPRINT;
CLASS gender site cogstat ;
VAR agitation ;
TYPES gender gender*site gender*cogstat gender*site*cogstat;
OUTPUT OUT=sum5 MEAN= /WAYS;
RUN;
```

The output data set is printed at the top of the next page. It includes all those observations – and only those observations – at a specific level of the CLASS variable GENDER. Note that it includes observations with _WAY_ = 1, 2 and 3 – but not all of the observations for any of these WAYs.

Example 5. Output

Obs	gender	site	cogstat	_WAY_	_TYPE_	_FREQ_	agitation
1	F	.	.	1	4	171	22.7193
2	M	.	.	1	4	261	20.3870
3	F	.	1	2	5	69	21.4638
4	F	.	2	2	5	102	23.5686
5	M	.	1	2	5	151	18.8411
6	M	.	2	2	5	110	22.5091
7	F	1	.	2	6	71	28.7746
8	F	2	.	2	6	100	18.4200
9	M	1	.	2	6	214	20.2243
10	M	2	.	2	6	47	21.1277
11	F	1	1	3	7	35	26.6571
12	F	1	2	3	7	36	30.8333
13	F	2	1	3	7	34	16.1176
14	F	2	2	3	7	66	19.6061
15	M	1	1	3	7	128	18.4609
16	M	1	2	3	7	86	22.8488
17	M	2	1	3	7	23	20.9565
18	M	2	2	3	7	24	21.2917

EXAMPLE 6: USING TYPES AND WAYS IN COMBINATION

Sometimes the problem is most simply solved by using both the TYPES statement and the WAYS statement. This will most likely be the case if the set of descriptive analyses you need is somewhat idiosyncratic. In this example, I'm requesting the overall analysis, all three one-way analyses (both specified in the WAYS statement) and in addition I want a single two-way analysis.

Example 6. Code

```
PROC MEANS DATA = ex.agitation NOPRINT;
CLASS gender site cogstat ;
VAR agitation ;
WAYS 0 1;
TYPES gender*site ;
OUTPUT OUT=sum6 MEAN= /WAYS;
RUN;
```

The output data set is printed below. This example also illustrates that when both TYPES and WAYS statements are specified, the result includes observations that satisfy either request (think of them as joined by OR, not AND).

Example 6. Output

Obs	gender	site	cogstat	_WAY_	_TYPE_	_FREQ_	agitation
1		.	.	0	0	432	21.3102
2		.	1	1	1	220	19.6636
3		.	2	1	1	212	23.0189
4		1	.	1	2	285	22.3544
5		2	.	1	2	147	19.2857
6	F	.	.	1	4	171	22.7193
7	M	.	.	1	4	261	20.3870
8	F	1	.	2	6	71	28.7746
9	F	2	.	2	6	100	18.4200
10	M	1	.	2	6	214	20.2243
11	M	2	.	2	6	47	21.1277

CONCLUSIONS

I'll close with just a few miscellaneous notes/caveats/observations about what I've presented here.

WAYS vs. TYPES

Although I've shown one situation in which the WAYS statement alone is not sufficient to request desired output (Example 5), and such cases may arise with some frequency, I don't think there are any combinations of analyses that can't be requested with only the TYPES statement. (Aside: I didn't show you the TYPES statement that would achieve the result from Example 6 without the WAYS statement, but I bet you can figure it out). So, you could just forget the WAYS statement and get cozy with TYPES. However, I still think this would be a loss because of the intuitive syntax of the WAYS statement and the fact that for many of the most common specifications of desired

descriptive analyses, the WAYS statement is a little simpler than the TYPES statement specifying the same requests (such as Example 4a).

A FEW COMMENTS ON EFFICIENCY

The focus here was not on processing efficiency, and for the data set used in these examples, which had only 432 observations, programming efficiency is much more relevant than processing time. I did conduct a small amount of testing on a data set with 4.32 million observations, and the efficiency gains for these examples was modest compared to using a WHERE clause on the OUTPUT statement or an additional DATA step to select desired observations (the latter I'm sure because even if the input data set is huge, the output data sets are still very small – and their size depends on the number of CLASS variables not the number of observations in the input data set). Clearly there are substantial efficiency gains over having separate PROC MEANS steps for different descriptive analyses that could be achieved in a single step with WAYS and/or TYPES. I did not examine memory use or the dependence of these results on the number of CLASS variables (or the number of levels those CLASS variables have); thus, I'm just taking the SAS documentation's word that WAYS and TYPES have the edge there. Finally, I'll restate that I believe there are clear advantages to using these statements to enhance the clarity and simplicity of your SAS code.

A FEW MORE COMMENTS ON MISSING DATA

The input data set I've used here (EX.AGITATION) has no missing data on either the CLASSification variables (GENDER, SITE and COGSTAT) or the analysis VARiable (AGITATION). As noted above, the observations on the output data set that are missing for one or more of the CLASS variables have nothing to do with missingness – they just indicate that that variable or variables are being ignored. What if you do have some missing data on your input data set (usually the case in real life)?

If the outcome or analysis variables are missing, then the computation of the requested statistics will, of course, exclude those observations, though the _FREQ_ variable on the output data set, which is the number of observations on the input file with the specified levels of the CLASS variables, will no longer reflect the actual sample size for that descriptive analysis. (Use the N output statistic in such case, which can vary among analysis variables if they have different patterns of missingness).

The situation is a little different for missing CLASS variables. If one or more CLASS variables are missing on a given observation, then that observation will be excluded from ALL the analyses (including the _FREQ_ variable) even if the given analysis doesn't depend in any way on that variable. For example, if GENDER were missing for 10 observations (and none of the other CLASS or requested analysis VARiables were missing), then, in the overall analysis, as well as the one-way analysis for COGSTAT and SITE and the two-way COGSTAT*SITE analysis, none of which depend on GENDER, these 10 observations would be excluded. Now, this may be what you want (all the analyses are based on the same sample), but it might not be – if, for example, you really want the one-way analysis of SITE to be *ignoring* GENDER, you might also want it to be ignoring if GENDER is missing. The MISSING option on the PROC statement doesn't really get around this; it simply treats missing values as another category of the CLASSification variables – which can get confusing on the output data set because missing now can mean that the particular CLASS variable was missing on the input data set OR that it is being ignored in the given analysis (i.e. the meaning of missing on the output data sets that we've discussed previously), although the _WAY_ and/or _TYPE_ variable could allow you to distinguish these two possibilities. My advice, if you have some missing data on one or more of your CLASS variables and you want PROC MEANS to ignore that missingness when the analysis doesn't pertain to the CLASS variable(s) with missing values, is to forego the TYPES and WAYS statements, and use one of the other methods (separate MEANS steps, using a clever WHERE clause on the OUTPUT statement, or some additional processing of the output data set) to achieve the desired result. Oh, well!

PARTING WORDS

This has been by no means (☺) an exhaustive treatment of PROC MEANS. PROC MEANS (and PROC SUMMARY) can do *lots* of other things. See, for example, a great NESUG 2005 paper by Marge Scerbo and Mic Lajiness (4), which clearly outlines many of the possibilities available in PROC MEANS (and PROC FREQ). In this relatively brief paper, I've focused on only a single feature (that I'll confess I learned about only pretty recently though I've been using these PROCs for many years) of this very versatile procedure that has increased the simplicity of the programming for many of the descriptive analyses I do. I hope it has opened up some new possibilities for you!

REFERENCES

1. SAS OnlineDoc 9.1, Base SAS Procedures Guide, The MEANS procedure.
2. Sloane, P. D., Noell-Waggoner, E., Hickman, S., Mitchell, M., Williams, C., Preisser, J., Barrick, A. L., Zimmerman, S., Brawley, E. Implementing a lighting intervention in public areas of long-term care facilities. Alzheimer's Care Quarterly, 2005, 6(4) 280-293.
3. Werner, P, Cohen-Mansfield, J, Koroknay, V, & Braun, J. The impact of a restraint-reduction program on nursing home residents. Geriatric Nursing, 1994, 15(3), 142-146.
4. Scerbo, M., Lajiness, M. 2005. *Freq n' Means.* NESUG 2005 Proceedings. Available online at: http://www.nesug.org/html/Proceedings/nesug05/pm/pm7.pdf

ACKNOWLEDGMENTS
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.

CONTACT INFORMATION
I welcome comments, suggestions and questions at:

Christianna S. Williams, PhD
Christianna_Williams@abtassoc.com