# STA/OPR 9750
# Summarizing data with SAS

Víctor Peña

Baruch College

## 1. PROC MEANS AND PROC UNIVARIATE

PROC MEANS and PROC UNIVARIATE are useful to summarize quantitative variables. PROC MEANS has a somewhat concise output, and PROC UNIVARIATE is substantially more verbose. Suppose that you have read in the hsb2 dataset and named it "hsb2." You can get summaries of the quantitative variables directly with the commands:

```
PROC MEANS data=hsb2;
RUN;


PROC UNIVARIATE data=hsb2;
RUN;
```

If you just want summaries for a subset of quantitative variables, you can indicate that with "VAR." For example, if you want to show summaries for "math" only (math scores):

```
PROC MEANS data=hsb2;
     VAR math;
RUN;
```

If you want to stratify your summaries by levels of categorical variables, you should add "CLASS." For example, if you want stratified summaries for race and ses (socioeconomic status), this will do it:

```
PROC MEANS data=hsb2;
     VAR math;
     CLASS race ses;
RUN;
```

You can get summaries for subsets of observations pretty easily. For example, you can find the summary statistics for students who scored higher than 60 in math with this command

```
PROC MEANS data=hsb2;
     WHERE math > 60;
RUN;
```

In these examples, I used PROC MEANS, but you can use the same options to refine the output of PROC UNIVARIATE.

# 2. PROC FREQ

PROC FREQ is useful for summarizing categorical variables. It's pretty easy to use. The following code will produce a one-way table for race

```
PROC FREQ data = hsb2;
     TABLES race;
RUN;
```

And the following code will produce a contingency table for race and ses:

```
PROC FREQ data = hsb2;
     TABLES race*ses;
RUN;
```

PROC FREQ has some options, which can be specified after adding "/" in the TABLES statement.

- MISSING: adds missing data as a separate category in the table
- NOROW/NOCOLUMN: don't display row/column percentages, respectively
- NOPERCENT: don't display overall %
- NOFREQ: don't display raw counts.

For example, this code will create a two-way table of race and ses with raw counts only:

```
PROC FREQ data = hsb2;
     tables race*ses / NOCOL NOROW NOPERCENT;
RUN;
```

# 3. REORDERING LEVELS OF CATEGORICAL VARIABLES

In our previous examples with the hsb2 dataset, the tables that include ses are somewhat hard to read: the levels are displayed in alphabetical order, instead of a more intuitive "low-middle-high." We can get SAS to print out the results in the more intuitive order following this procedure:

1. Convert ses to a quantitative variable, whose levels are 1, 2, and 3 for low, middle, and high socioeconomic status.
2. Apply categorical labels to the new quantitative variable.

The labels can be created with PROC FORMAT:

```
PROC FORMAT;
     value sesFmt  1 = 'low'
                   2 = 'middle'
                   3 = 'high';
RUN;
```

The following code creates a new categorical variable, sesnew, which takes on the values 1, 2, and 3, as we described above. The instruction "FORMAT sesnew sesFmt.;" applies the categorical labels 'low', 'middle' and 'high' to sesnew.

```
DATA hsbnew;
      SET hsb2 ;
      FORMAT sesnew sesFmt.;
      IF ses = 'low' THEN sesnew = 1;
      ELSE IF ses = 'middle' THEN sesnew = 2;
      ELSE sesnew = 3;
RUN;
```

When we run PROC FREQ on sesnew, SAS will display the categories in the order "low-middle-high" because it looks at the values of the variables, which in this case are 1, 2, and 3. The labels "low", "middle", and "high" are ignored and only used for printing.

# 4. PROC TABULATE

You can use PROC TABULATE to create tables with summary statistics.

In my view, the most efficient way of learning how to use PROC TABULATE is by playing around with some examples.

The tutorial "Anyone can learn PROC TABULATE" can be useful, too.

### Some examples

Table for with sample size, mean, standard deviation and median for math scores:

```
PROC TABULATE data = hsbnew;
      VAR math;
      TABLE math*(N MEAN STDDEV MEDIAN);
RUN;
```

The following code works too:

```
PROC TABULATE data = hsbnew;
      VAR math;
      TABLE math*N math*MEAN math*STDDEV math*MEDIAN;
RUN;
```

You can stratify by levels of a categorical variables easily. For example, you can get means and standard deviations by ses as follows:

```
PROC TABULATE data = hsbnew;
      CLASS ses;
      VAR math;
      TABLE math*(MEAN STDDEV), ses;
RUN;
```

The order in the TABLE statement is "<rows>, <columns>." That is, the output of the code above will have different columns with summaries by socioeconomic status. You can transpose the table by changing the order of ses and math.

If you want to stratify further by race, you can use either of these 2 codes:

```
* Break down math scores by race (rows);
PROC TABULATE data = hsbnew;
     CLASS ses race;
     VAR math;
     TABLE math*race*(MEAN STDDEV), ses;
RUN;

* Break down math scores by race (columns);
PROC TABULATE data = hsbnew;
     CLASS ses race;
     VAR math;
     TABLE math*(MEAN STDDEV), ses*race;
RUN;
```

Yet another option is creating separate tables by the levels of race:

```
* 3D table with gender;
PROC TABULATE data = hsbnew;
     CLASS gender ses race;
     VAR math;
     TABLE race,gender, math*(N MEAN STDDEV), ses;
RUN;
```

You can get rid of the variable headers in the tables as follows:

```
PROC TABULATE data = hsbnew;
     CLASS ses race;
     VAR math;
     TABLE (math=' ')*(N MEAN STDDEV), (ses=' ')*(race=' ');
RUN;
```

Compare this with the output of

```
PROC TABULATE data = hsbnew;
     CLASS ses race;
     VAR math;
     TABLE math*(MEAN STDDEV), ses*race;
RUN;
```

It looks substantially cleaner.

# 4. PROC SGPLOT

The commands for plotting with PROC SGPLOT are pretty intuitive. I think that the code in “lecture4.sas” is probably enough to understand how it works. The tutorial “Using PROC SGPLOT for quick, high-quality graphs” can be useful, too. If you have any questions, let me know.