Statistical Methods and Computing, 22S:30/105
Instructor: Cowles
Lab 3
Mar. 1, 2006

1 Creating a simulated dataset

The following code will create a simulated dataset with 1000 observations drawn from a normal distribution with mean $\mu = 2$ and standard deviation $\sigma = 1$.

```
options linesize = 72 ;

data symm ;
seed = 32542 ;
retain seed ;
do i = 1 to 1000  ;
   y = 2 + rannor(seed) ;
   output ;
end ;
drop seed ;
run ;
```

We are going to treat this simulated dataset as a population – this is the complete set of items in which we are interested. To display just the first 10 records of this dataset,

```
proc print data = symm (obs=10) ;
run ;
```

2 Proc means

We can use *proc means* to get various summary statistics in a more compact format than *proc univariate* provides. The default statistics provided are

- n = number of observations
- mean
- std dev = standard deviation
- minimum
- maximum

```
proc means data = symm ;
var y ;
run ;
```

Output:

```
         Analysis Variable : Y


            N          Mean       Std Dev       Minimum       Maximum
         ------------------------------------------------------------
         1000     2.0142510     0.9967128    -1.3402900     5.3321347
         ------------------------------------------------------------
```

3 Drawing simple random samples from our population

We will use *proc plan* to draw a simple random sample of size 10 from our "population" of 1000 values. We can then use *proc means* to get summary statistics for our simple random sample by copying the selected record id's from the *proc plan* output into the code for *proc means*.

```
proc plan seed = 25;
factors a  =10 of 1000 ;
run ;

proc means data = symm ;
var y ;
where i in(625  253  996  487   44  236  578  298  251  668) ;
run ;
```

```
   N            Mean         Std Dev         Minimum         Maximum
 --------------------------------------------------------------------
  10       2.1224623       1.1918346      -0.0975871       3.9326700
 --------------------------------------------------------------------
```

Recall that, if you wish to get a different simple random sample from *proc plan*, you must specify a different value of *seed*.

4 Confidence intervals

We can request other descriptive statistics by specifying them as part of the *proc means* statement. One that you will need soon is the *confidence interval* for the mean. Putting "clm alpha = .05" on the end of the *proc means* statement produces a 95% confidence interval.

```
proc means data = symm  n mean stddev clm alpha = .05 ;
var y ;
where i in(625  253  996  487   44  236  578  298  251  668) ;
run ;
```

```
                                                Lower 95%       Upper 95%
   N            Mean         Std Dev          CL for Mean     CL for Mean
 --------------------------------------------------------------------
  10       2.1224623       1.1918346       1.2698752       2.9750494
 --------------------------------------------------------------------
```

5 Using formats to get SAS to print something other than the values a variable actually contains

We will be using the billionaire dataset again today. Its variable called `region` contains abbreviations ("A" for Asia, "E" for Europe, etc.). If we want SAS to print out the complete words instead of the abbreviations, so that tables and graphs are more understandable, we need to run a "proc format" *before* the data step. The data step must then refer to the formats defined in the format procedure.

```
options linesize = 75 ;
```

```
proc format ;
value $regfmt ’A’ = ’Asia’ ’E’ = ’Europe’ ’M’ = ’Middle East’
              ’O’ = ’Other’ ’U’ = ’US’ ;
run ;
```

Because the original data values in the region variable are characters rather than numbers, we have to use a dollar sign as the first character in the name of the format.

Note the format statement in the data step below. It tells SAS to apply the format you have defined here to a particular variable. When you use the format statement in a data step, you must put a period at the end of the format name.

6 Using labels to get SAS to print more descriptive variable names

```
data billion ;
infile ’c:\temp\billion.dat’ ;
input wealth age region $ ;
format region $regfmt. ;
label wealth = ’Wealth in Billion $’
         age = ’Age in Years’ ;
run ;
```

Now enter and run the following code to see how the formats and labels affect the output of the “print” and “freq” procedures.

```
proc print data = billion (obs = 20);
run ;

proc print label data = billion (obs = 20);
run ;

proc freq data = billion ;
tables region ;
run ;
```

7 Formats for numeric variables

Formats can also be used to group numeric data. Add a line to your format procedure and change one line in the data step as follows:

```
proc format ;
value $regfmt ’A’ = ’Asia’ ’E’ = ’Europe’ ’M’ = ’Middle East’
              ’O’ = ’Other’ ’U’ = ’US’ ;
value amtfmt low-<5 = ’<5’ 5-<10 = ’5-<10’ 10-<20 = ’10-<20’ 20-high = ’20+’ ;
run ;

data billion ;
infile ’c:\temp\billion.dat’ ;
input wealth age region $ ;
format region $regfmt. wealth amtfmt. ;
label wealth = ’Wealth in Billion $’
         age = ’Age in Years’ ;
run ;
```

To see the effect of adding this format:

```
proc print data = billion ;
run ;

proc freq data = billion ;
tables wealth ;
run ;
```

8 Using `proc tabulate` to summarize the distributions of quantitative variables in different groups

Gulanick (*Heart and Lung*, 1991) studied patients who were recovering from heart surgery. She was interested in whether different combinations of supervised exercise or teaching would affect patients’ self-efficacy (or confidence) to perform physical activity.

Patients were randomly assigned to one of three groups. Group 1 received teaching, treadmill exercise testing, and exercise training three times per week. Group 2 received only teaching and exercise testing. Group 3 received only routine care without supervised exercise or teaching. After 4 weeks, each patient was scored on self-efficacy.

Self-efficacy was measured on a continuous scale and scores were assumed to be distributed normally in each of the populations of interest. Her results are in the dataset “gulanick.dat.” We wish to produce a table that shows the number of observations and the mean and standard deviation of scores within each of the three groups.

```
proc format ;
value grpfmt 1 = ’Teaching and Training’ 2 = ’Teaching’ 3 = ’Neither’ ;
run ;

data gulan ;
infile ’c:\temp\gulanick.dat’ ;
input score group ;
format group grpfmt. ;
run ;

proc tabulate data = gulan ;
class group ;  * class statement identifies qualitative variables ;
var score ;    * var statement identifies quantitative variables ;
tables group , score * (n mean std) ;
run ;
```

9 More on `proc means`

The following code will produce means for this dataset of the values in the variables “wealth” and “age.”

```
proc means data = billion n mean ;
var wealth age ;
title ’Average Age and Wealth of 1992 Billionaires’ ;
run ;
```

Since our dataset contains an observation for every billionaire in the world in 1992, if the population of interest is billionaires in 1992, is this a population mean or a sample mean?

If we want a separate mean for each region, we must first make sure that the dataset is sorted in order by region, and then run "proc means" with an additional "by" statement.

```
proc sort ;
by region ;
run ;

proc means data = billion n mean ;
var wealth age ;
by region ;
title ’Average Age and Wealth of 1992 Billionaires’ ;
title2 ’By Region’ ;
run ;
```

10 Assignment for the rest of the lab period

(You may work with another person to do this, but I need at least 1 sets of results for each of 2 kinds of samples from each person.)

1. Do the following:
   (a) Draw a simple random sample of size 10 from our simulated "population." Use a different seed each time so you get different samples.
   (b) Calculate the sample mean and sample standard deviation from the sample and record it.
2. Sample means from a skewed distribution.
   (a) Use the code below to simulate a dataset from a skewed distribution.

```
data skewed ;
seed = 325 ;
retain seed ;
do i = 1 to 1000  ;
   y = 2 + rangam(seed, 2) ;
   output ;
end ;
drop seed ;
run ;
```

   (b) Use *proc univariate* to verify that you got a skewed distribution.

```
proc univariate plot data = skewed ;
var y ;
run ;
```

   (c) Draw a simple random sample of size 10 from the skewed population. Calculate and record the sample mean and standard deviation from each one.

Symmetric, n=10	$\bar{x}$	s
Skewed, n=10	$\bar{x}$	s