Week 06/07 Class Activities

File: week-06-07-24sep05.doc
Directory (hp/compaq):
    C:\baileraj\Classes\Fall 2005\sta402\handouts

C:\Documents and Settings\John Bailer\My Documents\baileraj\Classes\Fall 2004\sta402\handouts\week-06-07-06oct04.doc

Week 6 TRANSFORMING SAS DATA SETS

- Creating SAS data sets with DATA steps: flow of execution, including the program data vector
- Creating variables in DATA steps with assignment statements
- Statements: DATA, SET, OUTPUT, RETURN, WHERE, IF, DROP, KEEP, LENGTH
- Subsetting observations and variables
- Using SAS functions and operators
- Working with SAS date values (also time and date-time)
- Introduction to missing values

Week 7 SAS PROGRAMMING

- Declarative vs. executables statements
- Statements: RETAIN, RENAME, LABEL, FORMAT, SUM
- Using formats in DATA steps
- Conditional execution
- DO groups
- Arrays
- More on missing values

FORMATTING and data recoding


Formats

Informats

Internal formats

General form of formats:
**Character:** Sformatw.
(e.g. $w. = standard character data, $HEXw. Convert to hexadecimal)
Numeric: formatw.d
(e.g. BESTw. = SAS System chooses; COMMAw.d; DOLLARw.d; Ew. = sci. not.; w.d)

Date: formatw.
(e.g. DATEw. =ddmmmyy or ddmmmyyyy; DATETIME=ddmmmyy:hh:mm:ss.ss;
DAYw. = day of month; EURDFDDw. = dd.mm.yy; JULIANw.=Julian date; MMDDYYw.;
TIMEw.d = hh:mm:ss.ss; WEEKDATEw. = day-name,month-name,yy or yyyy;
WORDDATEw. = mont-name dd,yyyy

Comments:
$ = character
w = total width
d = number of decimal places

example: illustrating the formats above

data char_format_show;
/
* character formatting illustrated first */
  charstring = “Hello there”;
  put charstring $11.;
  put charstring $15.;
  put charstring $5.;
run;
yields the following output on the SAS log
Hello there
Hello there
Hello

data numeric_format_show;
/
* character formatting illustrated first */
  test_num = 1277695.384;
  put ’BEST6. / BEST9. / BEST12.’;
  put test_num BEST6.;
  put test_num BEST9.;
  put test_num BEST12.;
  put ’-----------------------------’;
  put ’COMMA7. / COMMA10.1 / COMMA11.3’;
  put test_num COMMA9.;
  put test_num COMMA12.1;
  put test_num COMMA13.3;
  put ’-----------------------------’;
  put ’E7.’;
  put test_num E7.;
  put ’-----------------------------’;
put '7. / 10.1 / 11.3';
put test_num 8.;
put test_num 12.1;
put test_num 13.3;
put '-----------------------------';
put 'DOLLAR7. / DOLLAR10.2';
put test_num DOLLAR9.;
put test_num DOLLAR12.2;

run;

yields the following output on the SAS log

 1.28E6
 1277695.4
 1277695.384
-----------------------------
COMMA7. / COMMA10.1 / COMMA11.3
 1,277,695
 1,277,695.4
 1,277,695.384
-----------------------------
E7.
 1.3E+06
-----------------------------
7. / 10.1 / 11.3
 1277695
 1277695.4
 1277695.384
-----------------------------
DOLLAR7. / DOLLAR10.2
 $1277695
 $1277695.38

data date_format_show;
  start = 0;
  put start date9.;

today = 15977;   * days since Jan 1, 1960;

  put '-------------------------------';
  put 'DATE7. / DATE9.';
  put today date7.;
  put today date9.;
  put '-------------------------------';
  put 'DAY2. / DAY7.';
  put today day2.;
  put today day7.;
  put '-------------------------------';
  put 'EURDFDD8.';
  put today eurdfdd8.;
  put '-------------------------------';
  put 'MMDDYY8. / MMDDYY6.';
  put today mmdyy8.;
  put today mmdyy6.;
  put '-------------------------------';
  put 'WEEKDATE15. / WEEKDATE29.';
  put today weekdate15.;
put today weekdate29.;
put '-------------------------------';
put 'WORDDATE12. / WORDDATE18.';
put today worddate12.;
put today worddate18.;
run;

yields the following output on the SAS log

01JAN1960
------------------------------
DATE7. / DATE9.
29SEP03 29SEP2003
------------------------------
DAY2. / DAY7.
29 29
------------------------------
EUROFDD8.
29.09.03
------------------------------
MMDDYY8. / MMDDYY6.
09/29/03 092903
------------------------------
WEEKDATE15. / WEEKDATE29.
Mon, Sep 29, 03 Monday, September 29, 2003
------------------------------
WORDDATE12. / WORDDATE18.
Sep 29, 2003 September 29, 2003

data time_format_show;
    start=0;
    time_test = 1380442000;
    put start DATETIME13.;
    put time_test DATETIME17.;
run;

yields the following output on the SAS log

01JAN60:00:00
29SEP03:08:06:40

INFORMAT – input data according to a particular format

Suppose your data was in the following format ...

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1234567890123456789012345678901234567890</td>
<td>column guides</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>01/01/1960</td>
<td>01:00:00</td>
<td>$100.22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>09/29/2003</td>
<td>09:49:59</td>
<td>$12693.79</td>
<td></td>
</tr>
</tbody>
</table>

data test;
    input @1 date MMDDYY10. @21 time TIME8. @31 money DOLLAR10.2;
INFORMATS can be used to process input variable values also can be defined using a PROC FORMAT statement before a DATA step using INVALUE instead of value

/* example 8 from Cody and Pass (1995) */
/* set up informats for valid ranges of variables */
proc format;
  invalue sbpfmt 40-300=_SAME_
    OTHER = .;
  invalue dbpfmt 10-150=_SAME_
    OTHER = .;
run;

data demo;
  input @1 ID $3. @4 SBP sbpfmt3. @7 DBP dbpfmt3. ;

data demo;
  ID  SBP  DBP
  001  160  90
  002  .    .
  003  .    .
  004  .    .
  005  150  70
;

proc print data=demo;
  run;

example: recoding and formatting

data country;
  title 'Country data analysis';
  infile "\\Muserver2\USERS\B\BAILERAJ\public.www\classes\sta402\hw\country.data";
  input  name $ area popnsize pcturban lang $ liter lifemen lifewom pcGNP;
  logarea = log10(area);
  logpopn = log10(popnsize);
  loggnp  = log10(pcGNP);
  ienglish = (lang="English");
  drop area popnsize pcgnp;

  label name = 'Name of country';
  label pcturban = 'Percent of the population residing in urban setting';
  label lang = 'Primary language spoken';
  label liter = 'Percent of the population that is literate';
  label lifemen = 'Life expectancy for Men (years)';
  label lifewom = 'Life expectancy for Women (years)';
  label logare = 'Geographic area (log10-transformed)';
  label logpopn = 'Population size (log10-transformed)';
label loggnp = 'Per capita Gross National Product (log10-transformed)';
label ienglish = 'Indicator variable that primary language is English';

proc format;
  value Mlifefmt  LOW-54 = 'First quartile'
                 54<-63  = 'Second quartile'
                 63<-68  = 'Third quartile'
                 68<-HIGH = 'Fourth quartile';
  value Wlifefmt  LOW-56 = 'First quartile'
                 56<-67  = 'Second quartile'
                 67<-73  = 'Third quartile'
                 73<-HIGH = 'Fourth quartile';
  value Literfmt  LOW-53 = 'First quartile'
                53<-76  = 'Second quartile'
                76<-90  = 'Third quartile'
                90<-HIGH = 'Fourth quartile';
  value catlit 1 = 'First quartile'
              2 = 'Second quartile'
              3 = 'Third quartile'
              4 = 'Fourth quartile';
;

data country2; set country;

/* recoding option 1 */
/* - least attractive alternative */
  if 0 LE liter LE 53 then categ_lit1 = 1;
  if 53 LT liter LE 76 then categ_lit1 = 2;
  if 76 LT liter LE 90 then categ_lit1 = 3;
  if 90 LT liter then categ_lit1 = 4;

/* recoding option 2 */
if 0 LE liter LE 53 then categ_lit2 = 1;
else if 53 LT liter LE 76 then categ_lit2 = 2;
else if 76 LT liter LE 90 then categ_lit2 = 3;
else if 90 LT liter then categ_lit2 = 4;

/* recoding option 3 */
if 0 <= liter & liter <= 53 then categ_lit3 = 1;
else if 53 < liter & liter <= 76 then categ_lit3 = 2;
else if 76 < liter & liter <= 90 then categ_lit3 = 3;
else if 90 < liter then categ_lit3 = 4;

/* recoding option 4 */
if liter GE 0 AND liter LE 53 then categ_lit4 = 1;
else if liter GT 53 AND liter LE 76 then categ_lit4 = 2;
else if liter GT 76 AND liter LE 90 then categ_lit4 = 3;
else if liter GT 90 then categ_lit4 = 4;
/* recoding option 5 */
/* - may be more efficient than if-then-else */
select;
when (0 <= liter <= 53)  categ_lit5=1;
when (53< liter <= 76)  categ_lit5=2;
when (76<= liter <= 90)  categ_lit5=3;
when (90< liter)        categ_lit5=4;
when (liter=.)           categ_lit5=.;
end;

/* recoding option 6 */
categ_lit6 = 1*(0<liter<=53) + 2*(53<liter<=76) + 3*(76<liter<=90) +
4*(90<liter);
if liter=. then categ_lit6=.; * make sure missing=. not 0;

/* recoding option 7 */
/* - creates character variable with the formatted levels as values */
categ_lit7 = put(liter,literfmt.);
run;

*ODS RTF file='D:\baileraj\Classes\Fall 2003\sta402\SAS-programs\week6-freq1.rtf';
ODS RTF file="\\Muserver2\USERS\B\BAILERAJ\public.www\classes\sta402\examples\week06-freq1.rtf";

proc freq;
  table categ_lit1-categ_lit7;
run;

ODS RTF CLOSE;

<table>
<thead>
<tr>
<th>categ_lit</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>25.97</td>
<td>20</td>
<td>25.97</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>24.68</td>
<td>39</td>
<td>50.65</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>24.68</td>
<td>58</td>
<td>75.32</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>24.68</td>
<td>77</td>
<td>100.00</td>
</tr>
</tbody>
</table>

* Frequency Missing = 2
<table>
<thead>
<tr>
<th>categ_lit2</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>25.97</td>
<td>20</td>
<td>25.97</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>24.68</td>
<td>39</td>
<td>50.65</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>24.68</td>
<td>58</td>
<td>75.32</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>24.68</td>
<td>77</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Frequency Missing = 2

<table>
<thead>
<tr>
<th>categ_lit3</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>25.97</td>
<td>20</td>
<td>25.97</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>24.68</td>
<td>39</td>
<td>50.65</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>24.68</td>
<td>58</td>
<td>75.32</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>24.68</td>
<td>77</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Frequency Missing = 2

<table>
<thead>
<tr>
<th>categ_lit4</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>25.97</td>
<td>20</td>
<td>25.97</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>24.68</td>
<td>39</td>
<td>50.65</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>24.68</td>
<td>58</td>
<td>75.32</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>24.68</td>
<td>77</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Frequency Missing = 2

<table>
<thead>
<tr>
<th>categ_lit5</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>25.97</td>
<td>20</td>
<td>25.97</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>24.68</td>
<td>39</td>
<td>50.65</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>24.68</td>
<td>58</td>
<td>75.32</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>24.68</td>
<td>77</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Frequency Missing = 2
TRANSFORMING SAS DATA SETS

* Creating SAS data sets with DATA steps: flow of execution, including the program data vector

LIBNAME pointer ‘directory-containing-SAS-data-sets’;
(see example below)

* Creating variables in DATA steps with assignment statements

lots of examples . . .

sqrt_total = sqrt(total);
conc2 = conc**2;
Iplastic = (condition="Plastic");
categ_lit6 = 1*(0<liter<=53) + 2*(53<liter<=76)
           + 3*(76<liter<=90) + 4*(90<liter);
if liter=. then categ_lit6=.; * make sure missing=. not 0;

Order of Operations/Precedence of operations …
1. ** (exponentiation first)
2. */ (multiplication and division second)
3. +- (addition and subtraction third)
4. –
etc.

data preced_test;
  x1a = 3*2**2;
  x1b = (3*2)**2;
  x2a = 3-2/2;
  x2b = (3-2)/2;
  x3a = -2**2;
  x3b = (-2)**2;
  put '-------------------------';
  put '|' Order of operations  |';
  put '|' illustrated         |';
  put '-------------------------';
  put '  3*2**2 = ' x1a;
  put '(3*2)**2 = ' x1b;
  put '  3-2/2 = ' x2a;
  put '(3-2)/2 = ' x2b;
  put ' -2**2 = ' x3a;
  put ' (-2)**2 = ' x3b;
run;

* Statements: DATA, SET, OUTPUT, RETURN, WHERE, IF, DROP, KEEP, LENGTH

DATA = begin new data block

SET = place contents of one (or more) data set(s) into new data set. Concatenates data sets if more than one data set named in the SET statement.

OUTPUT = writes an observation to an output data set

MORAL: Use PARENTHESES when concerned that operations need to be conducted in a specific order!!!!
RETURN = stops processing of current observation and proceeds to the beginning of the next observation.

WHERE = selects observations from a SAS data set that meet certain conditions

IF = continues processing only those observations that meet the condition specified in the expression (often used to subset a data set and define new variables)

DROP = names of variables to be omitted from an output data set

KEEP = names of the variables to write to the output data set

LENGTH = numeric constant that specifies a number of bytes for storing variables (often beneficial for reducing the storage of large data sets – e.g. don’t need 8 bytes to store a “Y” / “N” response)

EXAMPLE: libname plus set illustrated

```sas
libname class "\Muserver2\USERS\B\BAILERAJ\public.www\classes\sta402\data";

data nitro;
infile "\Muserver2\USERS\B\BAILERAJ\public.www\classes\sta402\SAS-programs\ch2-dat.txt" firstobs=16 expandtabs missover pad; * referencing M drive directly;
input animal conc brood1 brood2 brood3 total;
proc print;
run;

data class.nitrofen; set nitro;
run;

data nitrofen_A; set class.nitrofen;
brood=1; count=brood1; conc=conc; output;
brood=2; count=brood2; conc=conc; output;
brood=3; count=brood3; conc=conc; output;
keep brood count conc;
```
proc print;
  run;

proc tabulate;
  class conc brood;
  var count;
  table conc*brood,count*(min q1 median q3 max);
  run;
ODS RTF CLOSE;

* Subsetting observations and variables

data nitrofen2;  set class.nitrofen;
  * select all observations with all but highest concentration;
    if conc<310;
data nitrofen3; set class.nitrofen;
    where conc<310;
run;

From the SAS LOG file
NOTE: There were 50 observations read from the data set CLASS.NITROFEN.
NOTE: The data set WORK.NITROFEN2 has 40 observations and 7 variables.
NOTE: DATA statement used:
    real time           0.01 seconds
    cpu time            0.01 seconds

1143  data nitrofen3; set class.nitrofen;
1144    where conc<310;
1145  run;

NOTE: There were 40 observations read from the data set CLASS.NITROFEN.
WHERE conc<310;
NOTE: The data set WORK.NITROFEN3 has 40 observations and 7 variables.
NOTE: DATA statement used:
    real time           0.66 seconds
    cpu time            0.03 seconds

* Using SAS functions and operators

* Working with SAS date values (also time and date-time) – DISCUSSED ABOVE
* Introduction to missing values– DISCUSSED ABOVE

EXAMPLE: Using SAS data step to do Monte Carlo Integration

/*
Problem:   Estimate PI using Monte Carlo Integration
Strategy:  Equation of a circle with radius=1:  x^2 + y^2 = 1
            which can be written y = sqrt(1-x^2)
            Area of this circle = PI
            Area of this circle in the first quadrant = PI/4

            Generate Ux ~ Uniform(0,1) and Uy ~ Uniform(0,1)
            Check to see if Uy <= sqrt(1-Ux^2)
            The proportion of generated points when this
            Condition is true is an estimate of PI/4.
*/

data MCint;
    /* initialize seed */
    seed1 = 12345;
do isim = 1 to 10000;

    /* generate point in first quadrant */
    Ux = ranuni(seed1);
    Uy = ranuni(seed1);

    /* check to see if point under the circle */
    Under = (Uy <= sqrt(1-Ux**2));

    /* output simulation result */
    drop isim;
    output;
end;

* ODS TRACE ON;
ODS OUTPUT OneWayFreqs=data_freq;
proc freq;
    table Under;
    run;
ODS OUTPUT CLOSE;
* ODS TRACE OFF;

proc print data=data_freq; run;

data summary; set data_freq;
    if under = 1;
    PI_est = 4*Percent/100;
    prop_est = Percent/100;
    SE_PI_EST = 4*sqrt(prop_est * (1 - prop_est)/10000 );
    PI_CI_Low = PI_est - 2*SE_PI_EST;
    PI_CI_Up = PI_est + 2*SE_PI_EST;
    put '-----------------------------------------------';
    put '|' MC Integration estimate of PI               |';
    put '-----------------------------------------------';
    put '     PI (estimate) = ' PI_est;
    put '     SE [PI (estimate)] = ' SE_PI_EST;
    put '     Approx. 95% CI = ' PI_CI_Low ', ' PI_CI_Up;
    put '   ';
    put 'Based on 10,000 simulated points. ';
    put '-----------------------------------------------';
run;

From the SAS LOG file

<table>
<thead>
<tr>
<th>MC Integration estimate of PI</th>
</tr>
</thead>
<tbody>
<tr>
<td>PI (estimate) = 3.1056</td>
</tr>
<tr>
<td>SE [PI (estimate)] = 0.0166662792</td>
</tr>
<tr>
<td>Approx. 95% CI = 3.0722674415 , 3.1389325585</td>
</tr>
</tbody>
</table>

Based on 10,000 simulated points.

-----------------------------------------------

ODS RTF file=
EXAMPLE: Using SAS DATA programming to do a small MC simulation
Problem: Explore whether t-test really is robust to violations of the equal variance assumption.

Strategy: See if the t-test operates at the nominal Type I error rate when the unequal variance assumption is violated.

Test case: \( n_1 = n_2 = 10 \)
- Population 1: \( N(0,1) \)
- Population 2: \( N(0,4) \)

```sas
/*
data twogroup;
array x{10} x1-x10;
array y{10} y1-y10;
do isim = 1 to 10000;
  /* generate samples \( X \sim N(0,1) \) \( Y \sim N(0,4) \) - normal case */
  do isample = 1 to 10;
    x{isample} = rannor(0);
    y{isample} = 2*rannor(0);
  end;
  /* calculate the t-statistic */
  xbar = mean(of x1-x10);
  ybar = mean(of y1-y10);
  xvar = var(of x1-x10);
  yvar = var(of y1-y10);
  s2p = (9*xvar + 9*yvar)/18;
  tstat = (xbar-ybar)/sqrt(s2p*(2/10));
  Pvalue = 2*(1-probt(abs(tstat),18));
  Reject05 = (Pvalue <= 0.05);
  keep xbar ybar xvar yvar s2p tstat Pvalue Reject05;
  output;
end; /* end of the simulation loop;*/
/*
proc print;
  run;
*/
proc freq;
  table Reject05;
  run;
```

<table>
<thead>
<tr>
<th>Reject05</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>9443</td>
<td>94.43</td>
<td>9443</td>
<td>94.43</td>
</tr>
</tbody>
</table>
So, a nominal error rate of 5% was specified and we rejected 5.57% of the time in the 10000 simulated samples.

**EXAMPLE: Using Maximum Likelihood Estimation using PROC NLIN or NLMIXED**

```plaintext
/* Problem: Obtain MLEs for the exponential rate parameter */
Test case: Generated a sample with 25 observations from an exponential distribution with lambda=1

Statistical details . . .
Exponential density: \( f(t) = \lambda \exp(-\lambda t) \)

Likelihood: \( L(\lambda) = \lambda^n \exp(-\lambda \sum t_i) \)

Log-likelihood:
\[
LL(\lambda) = \sum \left[ \log(\lambda) - \lambda t_i \right]
\]

Neg-Log-likelihood:
\[
NLL(\lambda) = \sum \left[ -\log(\lambda) + \lambda t_i \right]
\]

MLE = value of \( \lambda \) that max \( LL(\lambda) \) or Min \( NLL(\lambda) \)

Technical details . . .
PROC NLIN is used to fit non-linear regression model, Say \( g(t,\lambda) \), using least squares. In other words, \( \lambda \) is estimated to minimize \( [Y - g(t,\lambda)]^2 \).

The trick to using NLIN for MLE is to set \( Y=0 \) and \( g(t,\lambda) = NLL(\lambda) \).

*/

data gen_exp;

/* set up a dummy response for use with NLIN */
dummy=0;

/* generate the exponential sample for testing */
do i = 1 to 25;
    time = ranexp(0);
    output;
end;
```
proc means data=gen_exp;
  var time;
  output out=m_out;
run;

data mle_exact; set m_out;
  if _STAT_='MEAN';
  lambda_MLE = 1/TIME;
proc print data=mle_exact;
  run;
/*
  Obs    _TYPE_    _FREQ_    _STAT_      time       MLE
  1        0        25       MEAN     0.95268    1.04967
*/

proc nlin method=dud data=gen_exp;
  parameter lambda=0.25;
  negloglin = -log(lambda) + lambda*time;
  if negloglin<0 then negloglin = 1e-6;
  model dummy = sqrt(negloglin);
run;

Iterative Phase

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>TYPE</em></th>
<th><em>FREQ</em></th>
<th><em>STAT</em></th>
<th>time</th>
<th>lambda</th>
<th>MLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>25</td>
<td>MEAN</td>
<td>0.95268</td>
<td>1.04967</td>
<td></td>
</tr>
</tbody>
</table>

NOTE: Convergence criterion met but a note in the log indicates a possible problem with the model.

Estimation Summary

Method             Gauss-Newton
Iterations         13
Subiterations      5
Average Subiterations 0.384615
R                 8.253E-6
PPC                8.915E-9
RPC(lambda)        3.603E-8
Object             5.51E-10
### Objective

23.7907

### Observations Read

25

### Observations Used

25

### Observations Missing

0

**NOTE:** An intercept was not specified for this model.

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Sum of Squares</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>1</td>
<td>-23.7907</td>
<td>-23.7907</td>
<td>-24.00</td>
<td>.</td>
</tr>
<tr>
<td>Error</td>
<td>24</td>
<td>23.7907</td>
<td>0.9913</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected Total</td>
<td>25</td>
<td></td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Approx Parameter</th>
<th>Estimate</th>
<th>Std Error</th>
<th>Approximate 95% Confidence Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>lambda</td>
<td>1.0344</td>
<td>0.000228</td>
<td>1.0339 1.0349</td>
</tr>
</tbody>
</table>

/* alternative code using NLMIXED where likelihood is directly entered */
/* added: 6 Oct 04 */

```latex
proc nlmixed data=gen_exp;
  parms lambda=0.25;
  ll = log(lambda) - lambda*time;
  model time ~ general(ll); /* could also use gamma(lambda,1) in model; */
run;
```

### Specifications

- **Data Set:** WORK.GEN_EXP
- **Dependent Variable:** time
- **Distribution for Dependent Variable:** General
- **Optimization Technique:** Dual Quasi-Newton
- **Integration Method:** None

### Dimensions

- **Observations Used:** 25
- **Observations Not Used:** 0
- **Total Observations:** 25
- **Parameters:** 1

### Parameters

- **lambda**
  - **NegLogLike:** 40.611594
  - **Estimate:** 0.25

### Iteration History

<table>
<thead>
<tr>
<th>Iter</th>
<th>Calls</th>
<th>NegLogLike</th>
<th>Diff</th>
<th>MaxGrad</th>
<th>Slope</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>25.0547568</td>
<td>15.55684</td>
<td>9.516393</td>
<td>-380.915</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>23.8231484</td>
<td>1.231608</td>
<td>1.308295</td>
<td>-0.67921</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>23.7908217</td>
<td>0.032327</td>
<td>0.359401</td>
<td>-0.01845</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>23.7880391</td>
<td>7.492E-6</td>
<td>0.000274</td>
<td>-3.31E-9</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>23.7880316</td>
<td>7.492E-6</td>
<td>0.000274</td>
<td>-3.31E-9</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>23.7880316</td>
<td>1.655E-9</td>
<td>1.785E-8</td>
<td>-3.31E-9</td>
</tr>
</tbody>
</table>

**NOTE:** GCONV convergence criterion satisfied.

### Fit Statistics
EXAMPLE: Using SAS DATA programming to do a percentile bootstrap CI

/* Problem: Construct a 90% confidence interval for mean using simple-percentile bootstrap */

data in_data;
  input mpg @@;
datalines;
28 27 34 31 29 27 24 23 36 37 31 38 36 36 36 34 38 32 38 25 38 26 22 32 36 27 27 44 32 28 31;
proc univariate;
  var mpg;
  run;
/* The UNIVARIATE Procedure

Variable: mpg

Moments

<table>
<thead>
<tr>
<th>N</th>
<th>31</th>
<th>Sum Weights</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>31.7096774</td>
<td>Sum Observations</td>
<td>983</td>
</tr>
<tr>
<td>Std Deviation</td>
<td>5.39254763</td>
<td>Variance</td>
<td>29.0795699</td>
</tr>
<tr>
<td>Skewness</td>
<td>0.08079739</td>
<td>Kurtosis</td>
<td>-0.6630813</td>
</tr>
<tr>
<td>Uncorrected SS</td>
<td>32043</td>
<td>Corrected SS</td>
<td>872.387097</td>
</tr>
<tr>
<td>Coeff Variation</td>
<td>17.0059996</td>
<td>Std Error Mean</td>
<td>0.96853014</td>
</tr>
</tbody>
</table>

Basic Statistical Measures

<table>
<thead>
<tr>
<th>Location</th>
<th>Variability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>31.70968</td>
</tr>
<tr>
<td>Median</td>
<td>32.0000000</td>
</tr>
<tr>
<td>Mode</td>
<td>36.0000000</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Lcm = 31.7096774 - tinv(.05,30)*0.96853014;
Ucm = 31.7096774 + tinv(.05,30)*0.96853014;
*/

data tmp;
  Lcm = 31.7096774 - tinv(1-.05,30)*0.96853014;
Ucm = 31.7096774 + tinv(1-.05,30)*0.96853014;
put "90% t-based CI = " lcm ucm;
run;

90% t-based CI = 30.065829076 33.353525724

data boot_data;
array mpg{31} mpg1-mpg31;
array bmpg{31} bmpg1-bmpg31;
input mpg1-mpg31;
do i=1 to 4000;
  do ii = 1 to 31;
    ipick = int(31*ranuni(0)+1);
    bmpg(ii) = mpg(ipick);
  end;
  boot_mean = mean(of bmpg1-bmpg31);
  keep boot_mean;
  output boot_data;
end;
datalines;
28 27 34 31 29 27 24 23 36 37 31 38 36 34 38 32 38 25 38 26 22 32 36 27 27 44 32 28 31
run;

ODS RTF file='D:\baileraj\Classes\Fall 2003\sta402\SAS-programs\week6-boot.rtf'

proc univariate data=boot_data plot;
  var boot_mean;
  run;

ODS RTF close;

<table>
<thead>
<tr>
<th>Moments</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N=4000</td>
<td>Sum Weights=4000</td>
</tr>
<tr>
<td>Mean</td>
<td>31.6703065</td>
<td>Sum Observations=126681.226</td>
</tr>
<tr>
<td>Std Deviation</td>
<td>0.9319187</td>
<td>Variance=0.86847246</td>
</tr>
<tr>
<td>Skewness</td>
<td>-0.0194903</td>
<td>Kurtosis=-0.0973244</td>
</tr>
<tr>
<td>Uncorrected SS</td>
<td>4015506.26</td>
<td>Corrected SS=3473.02135</td>
</tr>
<tr>
<td>Coeff Variation</td>
<td>2.94256293</td>
<td>Std Error Mean=0.01473493</td>
</tr>
</tbody>
</table>
### Basic Statistical Measures

<table>
<thead>
<tr>
<th>Location</th>
<th>Variability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.93192</td>
</tr>
<tr>
<td>Median</td>
<td>0.86847</td>
</tr>
<tr>
<td>Mode</td>
<td>6.35484</td>
</tr>
<tr>
<td>Range</td>
<td>1.25806</td>
</tr>
</tbody>
</table>

### Quantiles (Definition 5)

<table>
<thead>
<tr>
<th>Quantile</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% Max</td>
<td>34.8387</td>
</tr>
<tr>
<td>99%</td>
<td>33.8387</td>
</tr>
<tr>
<td>95%</td>
<td>33.1935</td>
</tr>
<tr>
<td>90%</td>
<td>32.8710</td>
</tr>
<tr>
<td>75% Q3</td>
<td>32.2903</td>
</tr>
<tr>
<td>50% Median</td>
<td>31.6774</td>
</tr>
<tr>
<td>25% Q1</td>
<td>31.0323</td>
</tr>
<tr>
<td>10%</td>
<td>30.4516</td>
</tr>
<tr>
<td>5%</td>
<td>30.1613</td>
</tr>
<tr>
<td>1%</td>
<td>29.5161</td>
</tr>
<tr>
<td>0% Min</td>
<td>28.4839</td>
</tr>
</tbody>
</table>

**Histogram**

```
34.75++  # 0
.*        0
*****     80  |
*********** 439 |
************************* 714 +-----+ 4
**************************************** 774 +-----+
********************************************** 823 |
*********************************************** 493 |
*********************************************** 301 |
******                          96  |
*****                            28 0
****                             9 0
*                                1 0
```

* may represent up to 18 counts

**Boxplot**

```
|                                               |
|                                               |
```

**Normal Probability Plot**

```
34.75+  |
|       |
|       |
|       |
|       |
|       |
|       |
|       |
```

---

---
EXAMPLE: Why you don’t need statistical tables any longer...

data prob_fcn_examples;

/* SAS Functions that provide Z ~ N(0,1) table values */
norm_area_left = probnorm(-1.645);  * area below -1.645 under N(0,1);
norm_area_left2 = cdf("Normal", -1.645);

norm_area_right = 1 - probnorm(-1.645);  * area above -1.645 under N(0,1);

alpha = 0.05;
z_lower = probit(alpha);
z_upper = probit(1 - alpha);

/* SAS Functions that provide T ~ t(df) table values */
df = 6;
t_area_left = probt(-1.645, df);  * area below -1.645 under t(df=6);
t_area_left2 = cdf("T", -1.645, df);

t_area_right = 1 - probt(-1.645, df);  * area above -1.645 under t(df=6);

alpha = 0.05;
t_lower = tinv(alpha, df);
t_upper = tinv(1 - alpha, df);

/* can do the same type of calculations for Chisquare: probchi, cinv
   F: profb, finv
   plus for a host of other continuous and discrete RVs */

put _all_;  * output results to SAS log;

proc print;
run;
“Functions and Call Routines”

<table>
<thead>
<tr>
<th>Probability</th>
<th>CDF Function</th>
<th>Computes cumulative distribution functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LOGCDF Function</td>
<td>Computes the logarithm of a left cumulative distribution function</td>
</tr>
<tr>
<td></td>
<td>LOGPDF Function</td>
<td>Computes the logarithm of a probability density (mass) function</td>
</tr>
<tr>
<td></td>
<td>LOGSDF Function</td>
<td>Computes the logarithm of a survival function</td>
</tr>
<tr>
<td></td>
<td>PDF Function</td>
<td>Computes probability density (mass) functions</td>
</tr>
<tr>
<td></td>
<td>POISSON Function</td>
<td>Returns the probability from a Poisson distribution</td>
</tr>
<tr>
<td></td>
<td>PROBBETA Function</td>
<td>Returns the probability from a beta distribution</td>
</tr>
<tr>
<td></td>
<td>PROBBNML Function</td>
<td>Returns the probability from a binomial distribution</td>
</tr>
<tr>
<td></td>
<td>PROBBNRM Function</td>
<td>Computes a probability from the bivariate normal distribution</td>
</tr>
<tr>
<td></td>
<td>PROBCHI Function</td>
<td>Returns the probability from a chi-squared distribution</td>
</tr>
<tr>
<td></td>
<td>PROBF Function</td>
<td>Returns the probability from an F distribution</td>
</tr>
<tr>
<td></td>
<td>PROBGAM Function</td>
<td>Returns the probability from a gamma distribution</td>
</tr>
<tr>
<td></td>
<td>PROBHYPR Function</td>
<td>Returns the probability from a hypergeometric distribution</td>
</tr>
<tr>
<td></td>
<td>PROBMC Function</td>
<td>Computes a probability or a quantile from various distributions for multiple</td>
</tr>
</tbody>
</table>

From [http://support.sas.com/onlinedoc/912/docMainpage.jsp](http://support.sas.com/onlinedoc/912/docMainpage.jsp)
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROBNEGB Function</td>
<td>Returns the probability from a negative binomial distribution</td>
</tr>
<tr>
<td>PROBNORM Function</td>
<td>Returns the probability from the standard normal distribution</td>
</tr>
<tr>
<td>PROBT Function</td>
<td>Returns the probability from a t distribution</td>
</tr>
<tr>
<td>SDF Function</td>
<td>Computes a survival function</td>
</tr>
<tr>
<td>Quantile</td>
<td></td>
</tr>
<tr>
<td>BETAINV Function</td>
<td>Returns a quantile from the beta distribution</td>
</tr>
<tr>
<td>CINV Function</td>
<td>Returns a quantile from the chi-squared distribution</td>
</tr>
<tr>
<td>FINV Function</td>
<td>Returns a quantile from the F distribution</td>
</tr>
<tr>
<td>GAMINV Function</td>
<td>Returns a quantile from the gamma distribution</td>
</tr>
<tr>
<td>PROBIT Function</td>
<td>Returns a quantile from the standard normal distribution</td>
</tr>
<tr>
<td>QUANTILE Function</td>
<td>Computes the quantile from a specified distribution</td>
</tr>
<tr>
<td>TINV Function</td>
<td>Returns a quantile from the t distribution</td>
</tr>
<tr>
<td>Random Number</td>
<td></td>
</tr>
<tr>
<td>CALL RANBIN Routine</td>
<td>Returns a random variate from a binomial distribution</td>
</tr>
<tr>
<td>CALL RANCAU Routine</td>
<td>Returns a random variate from a Cauchy distribution</td>
</tr>
<tr>
<td>CALL RANEXP Routine</td>
<td>Returns a random variate from an exponential distribution</td>
</tr>
<tr>
<td>CALL RANGAM Routine</td>
<td>Returns a random variate from a gamma distribution</td>
</tr>
<tr>
<td>CALL RANNOR Routine</td>
<td>Returns a random variate from a normal distribution</td>
</tr>
<tr>
<td>CALL RANPERK Routine</td>
<td>Randomly permutes the values of the arguments, and returns a permutation of k out of n values</td>
</tr>
<tr>
<td>CALL RANPERM Routine</td>
<td>Randomly permutes the values of the arguments</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>CALL RANPOI Routine</td>
<td>Returns a random variate from a Poisson distribution</td>
</tr>
<tr>
<td>CALL RANTBL Routine</td>
<td>Returns a random variate from a tabled probability distribution</td>
</tr>
<tr>
<td>CALL RANTRI Routine</td>
<td>Returns a random variate from a triangular distribution</td>
</tr>
<tr>
<td>CALL RANUNI Routine</td>
<td>Returns a random variate from a uniform distribution</td>
</tr>
<tr>
<td>CALL STREAMINIT Routine</td>
<td>Specifies a seed value to use for subsequent random number generation by the RAND function</td>
</tr>
<tr>
<td>NORMAL Function</td>
<td>Returns a random variate from a normal distribution</td>
</tr>
<tr>
<td>RANBIN Function</td>
<td>Returns a random variate from a binomial distribution</td>
</tr>
<tr>
<td>RANCAU Function</td>
<td>Returns a random variate from a Cauchy distribution</td>
</tr>
<tr>
<td>RAND Function</td>
<td>Generates random numbers from a specified distribution</td>
</tr>
<tr>
<td>RANEXP Function</td>
<td>Returns a random variate from an exponential distribution</td>
</tr>
<tr>
<td>RANGAM Function</td>
<td>Returns a random variate from a gamma distribution</td>
</tr>
<tr>
<td>RANNOR Function</td>
<td>Returns a random variate from a normal distribution</td>
</tr>
<tr>
<td>RANPOI Function</td>
<td>Returns a random variate from a Poisson distribution</td>
</tr>
<tr>
<td>RANTBL Function</td>
<td>Returns a random variate from a tabled probability distribution</td>
</tr>
<tr>
<td>RANTRI Function</td>
<td>Returns a random variate from a triangular distribution</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>RANUNI Function</td>
<td>Returns a random variate from a uniform distribution</td>
</tr>
<tr>
<td>UNIFORM Function</td>
<td>Returns a random variate from a uniform distribution</td>
</tr>
</tbody>
</table>

A couple of mathematical functions that might be of interest as well …

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMB Function</td>
<td>Computes the number of combinations of $n$ elements taken $r$ at a time</td>
</tr>
<tr>
<td>PERM Function</td>
<td>Computes the number of permutations of $n$ items taken $r$ at a time</td>
</tr>
</tbody>
</table>