Week 06/07 Class Activities

File: week-06-07-07oct07.doc

Directory (hp/compaq):
    C:\baileraj\Classes\Fall 2007\sta402\handouts
Directory: \\Muserver2\USERS\B\baileraj\Classes\sta402\handouts

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:** $formatw.

(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=ddmmmyyy: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 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 '---------------------------';
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
-------------------------------
EURDFDD8.
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>01/01/1960</td>
<td>01:00:00</td>
<td>$100.22</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 @ date MMDDYY10. @21 time TIME8. @31 money DOLLAR10.2;
data lines;
01/01/1960          01:00:00  $100.22
09/29/2003          09:49:59  $12693.79;

*ODS RTF file='D:\baileraj\Classes\Fall 2007\sta402\SAS-programs\week6-prtl1.rtf';
ODS RTF file="\Muserver2\USERS\B\BAILERAJ\public.www\classes\sta402\examples\week06-prtl1.rtf";

proc print;
title print of date and time w/o formatting – internal SAS representation;
  var date time money;
  run;
proc print;
title print of date and time w/ formatting;
  var date time;
  format date MMDDYY10. time TIME8. money DOLLAR10.2;
  run;

ODS RTF CLOSE;

Obs   date   time    money
  1      0   3600   100.22
  2  15977   35399  12693.79

<table>
<thead>
<tr>
<th>Obs</th>
<th>date</th>
<th>time</th>
<th>money</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>01/01/1960</td>
<td>1:00:00</td>
<td>$100.22</td>
</tr>
<tr>
<td>2</td>
<td>09/29/2003</td>
<td>9:49:59</td>
<td>$12,693.79</td>
</tr>
</tbody>
</table>

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.;
datalines;
001160090
002310220
003020008
004   080
005150070
;

ODS RTF file="\\Muserver2\USERS\B\BAILERAJ\public.www\classes\sta402\examples\week06-prt2.rtf";
* ODS RTF file='D:\baileraj\Classes\Fall 2007\sta402\SAS-programs\week6-prt2.rtf';

proc print data=demo;
  run;

ODS RTF CLOSE;

<table>
<thead>
<tr>
<th>Obs</th>
<th>ID</th>
<th>SBP</th>
<th>DBP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>001</td>
<td>160</td>
<td>90</td>
</tr>
<tr>
<td>2</td>
<td>002</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>003</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>4</td>
<td>004</td>
<td>.</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td>005</td>
<td>150</td>
<td>70</td>
</tr>
</tbody>
</table>

example: recoding and formatting

data country;
title 'Country data analysis';
infile "\\Muserver2\USERS\B\BAILERAJ\public.www\classes\sta402\hw\country.data";
  * infile "M:\public.www\classes\sta402\hw\country.data"; * using UDS;
  infile "D:\baileraj\classes\Fall 2007\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 logarea = '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 2007\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_lit1</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*
<table>
<thead>
<tr>
<th>categ_lit6</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_lit7</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>2</td>
<td>2.53</td>
<td>2</td>
<td>2.53</td>
</tr>
<tr>
<td>First quartile</td>
<td>20</td>
<td>25.32</td>
<td>22</td>
<td>27.85</td>
</tr>
<tr>
<td>Third quartile</td>
<td>19</td>
<td>24.05</td>
<td>41</td>
<td>51.90</td>
</tr>
<tr>
<td>Fourth quartile</td>
<td>19</td>
<td>24.05</td>
<td>60</td>
<td>75.95</td>
</tr>
<tr>
<td>Second quartile</td>
<td>19</td>
<td>24.05</td>
<td>79</td>
<td>100.00</td>
</tr>
</tbody>
</table>

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;

from the SAS LOG

<table>
<thead>
<tr>
<th>Order of operations</th>
<th>illustrated</th>
</tr>
</thead>
<tbody>
<tr>
<td>3*2**2 = 12</td>
<td></td>
</tr>
<tr>
<td>(3*2)**2 = 36</td>
<td></td>
</tr>
<tr>
<td>3-2/2 = 2</td>
<td></td>
</tr>
<tr>
<td>(3-2)/2 = 0.5</td>
<td></td>
</tr>
<tr>
<td>-2**2 = -4</td>
<td></td>
</tr>
<tr>
<td>(-2)**2 = 4</td>
<td></td>
</tr>
</tbody>
</table>

MORAL: Use PARENTHESES when concerned that operations need to be conducted in a specific order!!!!

* 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
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

```
libname class "\\Muserver2\USERS\B\BAIлерА\public.ww\classes\sta402\data";

data nitro;
    infile "\\Muserver2\USERS\B\BAIлерА\public.ww\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;
```
```sas
ods rtf file="\\userver2\users\b\baileraj\public.ww\classes\sta402\sas-programs\week-06-tab1.rtf";

proc print;
   run;

proc tabulate;
   class conc brood;
   var count;
   table conc*brood,count*(min q1 median q3 max);
   run;
ods rtf close;
```

<table>
<thead>
<tr>
<th>Nitrofen concentration</th>
<th>brood</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Min</td>
<td>Q1</td>
<td>Median</td>
<td>Q3</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>3.00</td>
<td>5.00</td>
<td>5.50</td>
<td>6.00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>10.00</td>
<td>11.00</td>
<td>12.00</td>
<td>14.00</td>
</tr>
<tr>
<td>80</td>
<td>3</td>
<td>10.00</td>
<td>12.00</td>
<td>15.00</td>
<td>15.00</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3.00</td>
<td>5.00</td>
<td>5.00</td>
<td>6.00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9.00</td>
<td>11.00</td>
<td>12.00</td>
<td>12.00</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>12.00</td>
<td>14.00</td>
<td>14.50</td>
<td>16.00</td>
</tr>
<tr>
<td>160</td>
<td>1</td>
<td>2.00</td>
<td>6.00</td>
<td>6.00</td>
<td>6.00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>8.00</td>
<td>10.00</td>
<td>12.00</td>
<td>12.00</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>10.00</td>
<td>11.00</td>
<td>11.00</td>
<td>12.00</td>
</tr>
<tr>
<td>235</td>
<td>1</td>
<td>2.00</td>
<td>4.00</td>
<td>6.00</td>
<td>6.00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.00</td>
<td>0.00</td>
<td>3.50</td>
<td>10.00</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.00</td>
<td>6.00</td>
<td>6.50</td>
<td>9.00</td>
</tr>
<tr>
<td>310</td>
<td>1</td>
<td>0.00</td>
<td>5.00</td>
<td>5.50</td>
<td>6.00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

* 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 '-----------------------------------------------';
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: n1=n2=10
Population 1: N(0,1)
Population 2: N(0,4)

/*

data twogroup;
array x{10} x1-x10;
array y{10} y1-y10;
do isim = 1 to 10000;
    /* generate samples X~N(0,1) Y~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;
    
Cumulative    Cumulative
      Reject05    Frequency     Percent     Frequency      Percent
 ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ
0        9443       94.43          9443        94.43
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**

/* 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 lambda_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>Iter</th>
<th>lambda</th>
<th>Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.2500</td>
<td>40.6116</td>
</tr>
<tr>
<td>1</td>
<td>1.1829</td>
<td>24.2475</td>
</tr>
<tr>
<td>2</td>
<td>0.9727</td>
<td>23.8587</td>
</tr>
<tr>
<td>3</td>
<td>1.0355</td>
<td>23.7914</td>
</tr>
<tr>
<td>4</td>
<td>1.0341</td>
<td>23.7908</td>
</tr>
<tr>
<td>5</td>
<td>1.0341</td>
<td>23.7908</td>
</tr>
<tr>
<td>6</td>
<td>1.0343</td>
<td>23.7908</td>
</tr>
<tr>
<td>7</td>
<td>1.0344</td>
<td>23.7907</td>
</tr>
<tr>
<td>8</td>
<td>1.0344</td>
<td>23.7907</td>
</tr>
<tr>
<td>9</td>
<td>1.0344</td>
<td>23.7907</td>
</tr>
<tr>
<td>10</td>
<td>1.0344</td>
<td>23.7907</td>
</tr>
<tr>
<td>11</td>
<td>1.0344</td>
<td>23.7907</td>
</tr>
<tr>
<td>12</td>
<td>1.0344</td>
<td>23.7907</td>
</tr>
<tr>
<td>13</td>
<td>1.0344</td>
<td>23.7907</td>
</tr>
</tbody>
</table>

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

Estimation Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Gauss-Newton</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterations</td>
<td>13</td>
</tr>
<tr>
<td>Subiterations</td>
<td>5</td>
</tr>
<tr>
<td>Average Subiterations</td>
<td>0.384615</td>
</tr>
<tr>
<td>R</td>
<td>8.253E-6</td>
</tr>
<tr>
<td>PPC</td>
<td>8.915E-9</td>
</tr>
<tr>
<td>RPC(lambda)</td>
<td>3.603E-8</td>
</tr>
<tr>
<td>Objective</td>
<td>5.51E-10</td>
</tr>
<tr>
<td>Objective</td>
<td>23.7907</td>
</tr>
<tr>
<td>Observations Read</td>
<td>25</td>
</tr>
<tr>
<td>Observations Used</td>
<td>25</td>
</tr>
<tr>
<td>Observations Missing</td>
<td>0</td>
</tr>
</tbody>
</table>

NOTE: An intercept was not specified for this model.
Sum of Mean Approx
Source DF Squares Square F Value Pr > F
Model 1 -23.7907 -23.7907 -24.00 .
Error 24 23.7907 0.9913
Uncorrected Total 25 0

Parameter Estimate Std Error Approximate 95% Confidence Limits
lambda 1.0344 0.000228 1.0339 1.0349

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

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
0.25 40.611594

Iteration History
Iter Calls NegLogLike Diff MaxGrad Slope
1 2 25.0547568 15.55684 9.516393 -380.915
2 4 23.8231484 1.231608 1.308295 -0.67921
3 5 23.7908217 0.032327 0.359401 -0.05109
4 6 23.7880391 0.002783 0.01845 -0.00532
5 7 23.7880316 7.492E-6 0.000274 -3.31E-9
6 9 23.7880316 1.655E-9 1.785E-8 -3.31E-9

NOTE: GCONV convergence criterion satisfied.

Fit Statistics
-2 Log Likelihood 47.6
AIC (smaller is better) 49.6
AICC (smaller is better) 49.7
BIC (smaller is better) 50.8

Parameter Estimates
Parameter Estimate Error DF t Value Pr > |t| Alpha Lower Upper Gradient
lambda 1.0497 0.2099 25 5.00 <.0001 0.05 0.6173 1.4820 1.785E-8

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 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

          N          31  Sum Weights          31
          Mean     31.7096774  Sum Observations     983
          Std Deviation     5.39254763  Variance     29.0795699
          Skewness     0.08079739  Kurtosis    -0.6630813
          Uncorrected SS   32043  Corrected SS    872.387097
          Coeff Variation   17.0059996  Std Error Mean   0.96853014

Basic Statistical Measures

Location          Variability
          Mean     31.70968        Std Deviation     5.39255
          Median     32.00000        Variance     29.07957
          Mode     36.00000        Range     22.00000
          Interquartile Range     9.00000

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;
  end;
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 36 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 2007\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>N</td>
<td>4000</td>
<td>Sum Weights</td>
</tr>
<tr>
<td>Mean</td>
<td>31.6703065</td>
<td>Sum Observations</td>
</tr>
<tr>
<td>Std Deviation</td>
<td>0.9319187</td>
<td>Variance</td>
</tr>
<tr>
<td>Skewness</td>
<td>-0.0194903</td>
<td>Kurtosis</td>
</tr>
<tr>
<td>Uncorrected SS</td>
<td>4015506.26</td>
<td>Corrected SS</td>
</tr>
<tr>
<td>Coeff Variation</td>
<td>2.94256293</td>
<td>Std Error Mean</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Basic Statistical Measures</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td></td>
<td>Variability</td>
</tr>
<tr>
<td>Mean</td>
<td>31.67031</td>
<td>Std Deviation</td>
</tr>
<tr>
<td>Median</td>
<td>31.67742</td>
<td>Variance</td>
</tr>
<tr>
<td>Mode</td>
<td>31.83871</td>
<td>Range</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Interquartile Range</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Quantiles (Definition 5)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantile</td>
<td>Estimate</td>
</tr>
<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>
</tbody>
</table>
Quantiles (Definition 5)

<table>
<thead>
<tr>
<th>Quantile</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<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

<table>
<thead>
<tr>
<th>#</th>
<th>Boxplot</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>80</td>
<td></td>
</tr>
<tr>
<td>223</td>
<td>9</td>
</tr>
<tr>
<td>439</td>
<td>22</td>
</tr>
<tr>
<td>714</td>
<td>43</td>
</tr>
<tr>
<td>823</td>
<td>78</td>
</tr>
<tr>
<td>774</td>
<td>48</td>
</tr>
<tr>
<td>301</td>
<td>30</td>
</tr>
<tr>
<td>96</td>
<td>9</td>
</tr>
<tr>
<td>28</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

* may represent up to 18 counts

Normal Probability Plot

---

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);

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;

ods rtf;
proc transpose;
  run;
proc print;
  id _name_;
  run;
ods rtf close;

_NAME_   COL1
norm_area_left  0.04998
norm_area_left2 0.04998
norm_area_right 0.95002
alpha 0.0500
z_lower -1.64485
z_upper 1.64485
df 6.00000
t_area_left 0.07554
t_area_left2 0.07554
t_area_right 0.92446
t_lower -1.94318
t_upper 1.94318

From http://support.sas.com/onlinedoc/912/docMainpage.jsp

“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>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>LOGPDF Function</td>
<td>Computes the logarithm of a probability density (mass) function</td>
<td></td>
</tr>
<tr>
<td>LOGSDF Function</td>
<td>Computes the logarithm of a survival function</td>
<td></td>
</tr>
<tr>
<td>PDF Function</td>
<td>Computes probability density (mass) functions</td>
<td></td>
</tr>
<tr>
<td>POISSON Function</td>
<td>Returns the probability from a Poisson distribution</td>
<td></td>
</tr>
<tr>
<td>PROBBETA Function</td>
<td>Returns the probability from a beta distribution</td>
<td></td>
</tr>
<tr>
<td>PROBBNML Function</td>
<td>Returns the probability from a binomial distribution</td>
<td></td>
</tr>
<tr>
<td>PROBBNRM Function</td>
<td>Computes a probability from the bivariate normal distribution</td>
<td></td>
</tr>
<tr>
<td>PROBCHI Function</td>
<td>Returns the probability from a chi-squared distribution</td>
<td></td>
</tr>
<tr>
<td>PROBF Function</td>
<td>Returns the probability from an $F$ distribution</td>
<td></td>
</tr>
<tr>
<td>PROBGAM Function</td>
<td>Returns the probability from a gamma distribution</td>
<td></td>
</tr>
<tr>
<td>PROBHYPR Function</td>
<td>Returns the probability from a hypergeometric distribution</td>
<td></td>
</tr>
<tr>
<td>PROBMC Function</td>
<td>Computes a probability or a quantile from various distributions for multiple comparisons of means</td>
<td></td>
</tr>
<tr>
<td>PROBNEGB Function</td>
<td>Returns the probability from a negative binomial distribution</td>
<td></td>
</tr>
<tr>
<td>PROBNORM Function</td>
<td>Returns the probability from the standard normal distribution</td>
<td></td>
</tr>
<tr>
<td>PROBT Function</td>
<td>Returns the probability from a $t$ distribution</td>
<td></td>
</tr>
<tr>
<td>SDF Function</td>
<td>Computes a survival function</td>
<td></td>
</tr>
<tr>
<td>Quantile</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BETAINV Function</td>
<td>Returns a quantile from the beta distribution</td>
<td></td>
</tr>
<tr>
<td>CINV Function</td>
<td>Returns a quantile from the chi-squared distribution</td>
<td></td>
</tr>
<tr>
<td>FINV Function</td>
<td>Returns a quantile from the $F$ distribution</td>
<td></td>
</tr>
<tr>
<td>GAMINV Function</td>
<td>Returns a quantile from the gamma distribution</td>
<td></td>
</tr>
<tr>
<td>PROBIT Function</td>
<td>Returns a quantile from the standard normal distribution</td>
<td></td>
</tr>
<tr>
<td>QUANTILE Function</td>
<td>Computes the quantile from a specified distribution</td>
<td></td>
</tr>
<tr>
<td><strong>Random Number</strong></td>
<td><strong>Function</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>TINV Function</strong></td>
<td>Returns a quantile from the ( t ) distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANBIN Routine</strong></td>
<td>Returns a random variate from a binomial distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANCAU Routine</strong></td>
<td>Returns a random variate from a Cauchy distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANEXP Routine</strong></td>
<td>Returns a random variate from an exponential distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANGAM Routine</strong></td>
<td>Returns a random variate from a gamma distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANNOR Routine</strong></td>
<td>Returns a random variate from a normal distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANPERK Routine</strong></td>
<td>Randomly permutes the values of the arguments, and returns a permutation of ( k ) out of ( n ) values</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANPERM Routine</strong></td>
<td>Randomly permutes the values of the arguments</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANPOI Routine</strong></td>
<td>Returns a random variate from a Poisson distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANTBL Routine</strong></td>
<td>Returns a random variate from a tabled probability distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANTRI Routine</strong></td>
<td>Returns a random variate from a triangular distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL RANUNI Routine</strong></td>
<td>Returns a random variate from a uniform distribution</td>
<td></td>
</tr>
<tr>
<td><strong>CALL STREAMININIT Routine</strong></td>
<td>Specifies a seed value to use for subsequent random number generation by the RAND function</td>
<td></td>
</tr>
<tr>
<td><strong>NORMAL Function</strong></td>
<td>Returns a random variate from a normal distribution</td>
<td></td>
</tr>
<tr>
<td><strong>RANBIN Function</strong></td>
<td>Returns a random variate from a binomial distribution</td>
<td></td>
</tr>
<tr>
<td><strong>RANCAU Function</strong></td>
<td>Returns a random variate from a Cauchy distribution</td>
<td></td>
</tr>
<tr>
<td><strong>RAND Function</strong></td>
<td>Generates random numbers from a specified distribution</td>
<td></td>
</tr>
<tr>
<td><strong>RANEXP Function</strong></td>
<td>Returns a random variate from an exponential distribution</td>
<td></td>
</tr>
<tr>
<td><strong>RANGAM Function</strong></td>
<td>Returns a random variate from a gamma distribution</td>
<td></td>
</tr>
</tbody>
</table>
### RANNOR Function
Returns a random variate from a normal distribution

### RANPOI Function
Returns a random variate from a Poisson distribution

### RANTBL Function
Returns a random variate from a tabled probability distribution

### RANTRI Function
Returns a random variate from a triangular distribution

### RANUNI Function
Returns a random variate from a uniform distribution

### UNIFORM Function
Returns a random variate from a uniform distribution

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

### COMB Function
Computes the number of combinations of n elements taken r at a time

### PERM Function
Computes the number of permutations of n items taken r at a time