

22S:166  
Computing in Statistics

Other Software Packages  
Proc import  
A bit on SAS macro language

Lecture 25  
Nov. 19, 2008

Kate Cowles  
374 SH, 335-0727  
kcowles@stat.uiowa.edu

Reading data files into SAS from  
other software packages

- Import Wizard
  - point-and-click interactive reading
  - convenient if file only needs to be read once
  - can write `proc import` code to be copied into programs
- `proc import`
  - can be used instead of data step in SAS programs
  - much more convenient if file needs to be read in multiple programs, or program using file needs to be run repeatedly

Other software packages

- Microsoft Excel
  - spreadsheet
  - very convenient for entering data in flat-file format
  - clients very frequently bring data to statisticians in Excel format
  - NOT reliable and accurate for doing statistical analysis
- Microsoft Access
  - relational database management system

Importing from Other Sources

Types of files that the Import Wizard and/or `proc import` can read

Identifier	Input Data Source	Extension
ACCESS	Microsoft Access database	.MDB
DBF	dBASE file	.DBF
WK1	Lotus 1 spreadsheet	.WK1
WK3	Lotus 3 spreadsheet	.WK3
WK4	Lotus 4 spreadsheet	.WK4
EXCEL	Excel V 4 or 5 spreadsheet	.XLS
EXCEL4	Excel V 4 spreadsheet	.XLS
EXCEL5	Excel V 5 spreadsheet	.XLS
EXCEL97	Excel 97 spreadsheet	.XLS
DLM	delimited file (default is blank)	.*
CSV	delimited file (comma-sep vals)	.CSV
TAB	delimited file (tab-delimited )	.TXT

Restriction: The data sources available to you depend on the SAS/ACCESS products that you have licensed. If you do not have any SAS/ACCESS products licensed, then the only types of data source files available to you are .CSV, .TXT, and delimited files.

## Example

- from R or Splus
  - use `write.table` to write data out as a delimited file

Data frame that comes with R

```
> USArrests
      Murder Assault UrbanPop Rape
Alabama   13.2    236      58 21.2
Alaska   10.0    263      48 44.5
Arizona    8.1    294      80 31.0
Arkansas   8.8    190      50 19.5
California 9.0    276      91 40.6
.
```

R command to write out file as tab-delimited data file

```
> write.table(USArrests, file="C:\\My Documents\\166\\USArrests.txt",
  sep="\t", quote = FALSE, col.names=TRUE)
```

## Now in SAS....

File / Import Data

Import Wizard

Select a data source from the list below

Choose "Delimited File (\*.\*)"

Where is the file located?

Give full path name, e.g.

C:\My Documents\166\USArrests.txt

Choose SAS destination:

Library: (defaults to WORK)

Member: (fill in name of your choice; e.g. USArrest)

Question as to whether you want wizard to generate proc import statements so you can just run them next time

## What it generated

```
PROC IMPORT OUT= WORK.usarrest
  DATAFILE= "C:\My Documents\166\USArrests.txt"
  DBMS=DLM REPLACE;
  DELIMITER='00'x; * needed correction to DELIMITED='09'x ;
  GETNAMES=YES;
  DATAROW=2;
RUN;
```

## Example of reading Access database

```
PROC IMPORT OUT= WORK.courses
  DATATABLE= "Courses"
  DBMS=ACCESS97 REPLACE;
  DATABASE="c:\my documents\166\univ0_v7";
RUN;
```

## Overview of SAS Macro Programming

- purpose is to make SAS programming more efficient and to reduce coding errors
- macro variables
  - enable substitution of text into SAS programs
- macro programs
  - enable performing the same task on different inputs without rewriting code

## Example dataset

```
Data Set Name: BOOKS.YTDSALES      Observations:      695
Member Type:  DATA                Variables:         10
Engine:       V8                    Indexes:          0
Created:      7:36 Friday, October 19, 2001 Observation Length: 216
Last Modified: 7:36 Friday, October 19, 2001 Deleted Observations: 0
Protection:                               Compressed:      NO
Data Set Type:                               Sorted:         NO
Label:
```

-----Alphabetic List of Variables and Attributes-----

#	Variable	Type	Len	Pos	Format	Informat	Label
6	author	Char	50	115			First Author
8	cost	Num	8	8	DOLLAR9.2		Wholesale Cost
4	datesold	Num	4	32	MMDDYY8.	MMDDYY8.	Date Book Sold
9	listpric	Num	8	16	DOLLAR9.2		List Price
7	publishr	Char	50	165			Publisher
2	saleid	Num	8	0	8.		Sale ID
3	saleinit	Char	3	62			Sales Person Initial
10	salepric	Num	8	24	DOLLAR9.2		Sale Price
1	section	Char	26	36			Section
5	title	Char	50	65			

## Macro variables

- **%let** keyword defines a macro variable and assigns it a value
- use **&** before macro variable name when referencing variable
- use **%eval** keyword to convert a macro variable's value to numeric
- when referencing macro variables in character literals, use double quotes

## Macro variables example

```
%let repmonth=4;
%let repyear=2001;
%let repmword=%sysfunc(mdy(&repmonth,1,&repyear),monname9.)

data month&repmonth;
  set books.ytdsales;
  mosale=month(datesold);
  label mosale='Month of Sale';
run;

proc tabulate data=month&repmonth;
  title "Sales During &repmword &repyear";
  where mosale=&repmonth and year(datesold)=&repyear;
  class section;
  var salepric listpric cost;
  tables section all='**TOTAL**',
    (salepric listpric cost)*(n*f=4. sum*f=dollar9.2);
run;

* proc gchart data=month&repmonth ;
proc chart data=month&repmonth
  (where=(mosale < %eval(&repmonth+1) and
    year(datesold)=&repyear));
  title "Sales Through &repmword &repyear";
  pie section / sumvar=salepric noheading ;
run;
```

## Output

Sales During April 2001

	Sale Price		List Price		Wholesale Cost	
	N	Sum	N	Sum	N	Sum
Section						
Internet	145	\$4,579.71	145	\$4,680.75	145	\$3,318.77
Networks and Communication	55	\$1,633.01	55	\$1,665.25	55	\$1,177.46
Operating Systems	132	\$4,016.45	132	\$4,108.40	132	\$2,916.03
Programming Languages	60	\$1,835.07	60	\$1,878.00	60	\$1,330.98
Web Design	131	\$4,015.50	131	\$4,114.45	131	\$2,910.87
<b>**TOTAL**</b>	<b>523</b>	<b>\$16079.73</b>	<b>523</b>	<b>\$16446.85</b>	<b>523</b>	<b>\$11654.09</b>

```

Networks and Com
*****
***      .      ***
**      .      ** Internet
** $.5725.76      **
*      .9.54%      *
**      .      14911.01      **
**      .      24.84%      **
*      .      *
*      .      *
Operating System * 16660.07      *
*      .      *
*      .      *
*      .      *
**      .      15555.92      **
** $7178.46      25.91%      **
* .. 11.96%      *
**      .      **
**      .      ** Web Design
***      .      ***
Programming Lang *****

```

## Using built-in SAS macro variables

```

title "Sales Report";
title2 "As of &sysptime &sysday &sysdate";
title3 "Using SAS Version: &sysver";
proc means data=books.ytdsales n sum;
  var salepric;
run;

```

## Output

```

Sales Report
As of 06:38 Friday 19OCT01
Using SAS Version: 8.00

The MEANS Procedure

Analysis Variable : salepric Sale Price

      N      Sum
      ---      ---
6959      210588.23

```

13

## Using call symput to assign a value from a data step variable to a macro variable

- embedded `put` statement also formats the value before assigning it
- `retain` statement used in following example
  - initializes a variable at the beginning of a data step
  - tells SAS to carry its value forward as it sequentially processes records in the dataset

## Example of call symput

```

data temp;
  set books.ytdsales end=lastobs;
  retain sumintwb 0;
  if section in ('Internet','Web Design') then
    sumintwb=sumintwb + salepric;
  if lastobs then
    call symput('INTWEBSL',put(sumintwb,dollar10.2));

run;

proc chart data=temp;
  title "Internet and Web Design Sales: &intwebsl";
  title2 "As of &enddate";
  hbar section / sumvar=salepric;
  format salepric dollar10.2;
run;

```

## Output

```
Internet and Web Design Sales: $107320.40      1:
As of &enddate

Section                                     Sale Price
                                     Freq      Sum
-----
Internet ***** 1777 $53,998.95
Networks and Com ***** 649 $19,472.97
Operating System ***** 1877 $56,964.04
Programming Lang ***** 900 $26,830.81
Web Design ***** 1756 $53,321.45

                                     $30,000.00
                                     Sale Price
```

## Example of macro function

```
%macro daily;
proc means data=books.ytdsales(where=(datesold=today()))
    maxdec=2 sum;
    title "Daily Sales Report for &sysdate";
    class section;
    var salepric;
run;
%if &sysday=Friday %then %do;
    proc means data=books.ytdsales
        (where=(today()-6 le datesold le today()))
        sum maxdec=2;
        title "Weekly Sales Report Week Ending &sysdate";
        class section;
        var salepric;
    run;
%end;
%mend daily;
```

## Writing macro programs

- like subroutines or functions
- macro function is defined by the following structure

```
%macro macro-name
.
. < statements to be executed by macro
.
%mend macro-name
```
- code inside macro is essentially just SAS code
- but special macro keywords are used to control conditional and iterative processing
- macro must be defined before it can be called

## Calling the macro

- call a macro using %macroname

Example

```
%daily
```

# Output

Daily Sales Report for 19OCT01

2

The MEANS Procedure

The MEANS Procedure

Analysis Variable : salepric Sale Price

Analysis Variable : salepric Sale Price

Section	N Obs	Sum
Internet	7	212.66
Networks and Communication	5	123.76
Operating Systems	6	224.91
Programming Languages	3	81.36
Web Design	2	58.90

Section	N Obs	Sum
Internet	46	1391.95
Networks and Communication	15	420.37
Operating Systems	36	1171.83
Programming Languages	24	719.03
Web Design	35	1049.40

## Example of macro to do iterative processing

- the following macro copies the book sales data into 12 separate datasets, one for each month of the year

```
%macro makesets;
  data
    %do i=1 %to 12;
      month&i
    %end;
  ;
  set books.ytdsales;
  mosale=month(datesold);
  if mosale=1 then output month1;
  %do i=2 %to 12;
    else if mosale=&i then output month&i;
  %end;
run;
%mend makesets;
```

```
%makesets
```

## Passing parameters to macros

- parameters may be passed to a macro program
  - by position
  - by keyword
- parameters are named in parentheses after macro name in macro definition
- values are listed in parentheses after macro name in macro call

## Example of macro program with positional parameters

```
options mprint mlogic;

%macro listparm(start,stop,opts);
  title "Books Sold by Section Between &start and &stop";
  proc means data=books.ytdsales &opts;
    where "&start"d le datesold le "&stop"d;
    class section;
    var salepric;
  run;
%mend listparm;

*----First call to LISTPARM, all 3 parameters specified;
%listparm(01JUN1998,15JUN1998,n sum)

*----Second call to LISTPARM, first 2 parameters specified and;
*----third parameter is null;
%listparm(01SEP1998,15SEP1998,)
```

Books Sold by Section Between 01SEP2001 and 15SEP2001 2

The MEANS Procedure

Analysis Variable : salepric Sale Price

Section	N		Mean	Std Dev
	Obs	N		
Internet	77	77	30.0097403	6.7365196
Networks and Communication	24	24	28.7466667	4.2080094
Operating Systems	81	81	29.9080864	5.0282968
Programming Languages	41	41	29.6634146	4.7974602
Web Design	82	82	29.8243902	5.1119165

  

Section	N		Minimum	Maximum
	Obs	N		
Internet	77	77	15.9600000	42.9500000
Networks and Communication	24	24	19.9500000	35.9500000
Operating Systems	81	81	15.9500000	39.9500000
Programming Languages	41	41	15.9600000	39.9500000
Web Design	82	82	18.9500000	42.9500000

## Output

Books Sold by Section Between 01JUN2001 and 15JUN2001 2

The MEANS Procedure

Analysis Variable : salepric Sale Price

Section	N		Sum
	Obs	N	
Internet	84	84	2533.78
Networks and Communication	26	26	820.8350000
Operating Systems	71	71	2092.13
Programming Languages	46	46	1368.66
Web Design	66	66	2002.66

## Passing parameters by keyword

- enables setting defaults in macros

## Example

```
options mprint mlogic;

%macro keyparm(start=01JAN2001,stop=31DEC2001,
  opts=N SUM MIN MAX);
  title "Books Sold by Section Between &start and &stop";
  proc means data=books.ytdsales &opts;
    where "&start"d le datesold le "&stop"d;
    class section;
    var salepric;
  run;
%mend keyparm;

*----First call to KEYPARM: specify all keyword parameters;
%keyparm(start=01JUN2001,stop=15JUN2001,opts=n sum)

*----Second call to KEYPARM: specify start and stop,;
*----opts is null: should see default stats for PROC MEANS;
%keyparm(start=01SEP2001,stop=15SEP2001,opts=)

*----Third call to KEYPARM: use defaults for start and stop,;
*----specify opts;
%keyparm(opts=n sum)
```

# Output

Books Sold by Section Between 01JUN2001 and 15JUN2001

The MEANS Procedure

Analysis Variable : salepric Sale Price

Section	N		Sum
	Obs	N	
Internet	84	84	2533.78
Networks and Communication	26	26	820.8350000
Operating Systems	71	71	2092.13
Programming Languages	46	46	1368.66
Web Design	66	66	2002.66

Books Sold by Section Between 01SEP2001 and 15SEP2001

23

The MEANS Procedure

Analysis Variable : salepric Sale Price

Section	N		Mean	Std Dev
	Obs	N		
Internet	77	77	30.0097403	6.7365196
Networks and Communication	24	24	28.7466667	4.2080094
Operating Systems	81	81	29.9080864	5.0282968
Programming Languages	41	41	29.6634146	4.7974602
Web Design	82	82	29.8243902	5.1119165

Analysis Variable : salepric Sale Price

Section	N		Minimum	Maximum
	Obs	N		
Internet	77	77	15.9600000	42.9500000
Networks and Communication	24	24	19.9500000	35.9500000
Operating Systems	81	81	15.9500000	39.9500000
Programming Languages	41	41	15.9600000	39.9500000
Web Design	82	82	18.9500000	42.9500000

Books Sold by Section Between 01JAN2001 and 31DEC2001

24

The MEANS Procedure

Analysis Variable : salepric Sale Price

Section	N		Sum
	Obs	N	
Internet	1777	1777	53998.95
Networks and Communication	649	649	19472.97
Operating Systems	1877	1877	56964.04
Programming Languages	900	900	26830.81
Web Design	1756	1756	53321.45

## Options for macro processing

- `mprint` (`nomprint`)
  - specifies whether SAS statements that are generated by macro execution are displayed
- `mlogic` (`nomlogic`)
  - specifies whether SAS traces execution of the macro language processor. If `mlogic` is specified, trace information is written in SAS log