Training Session 2: Introduction to SAS, SAS Macros and Warehousing

Rodney Sparapani, PhD
The Center for Patient Care and Outcomes Research and the Division of Biostatistics
Medical College of Wisconsin

March 16, 2012
Training Outline

1. What is SAS?
2. Brief History of SAS
3. SAS Resources
4. Running SAS
5. The SAS Macro Language and The RASmacro Library
6. What is a Warehouse?
7. Warehouse Example
8. Conclusions

*If all else fails, read the instructions.*

- *Donald Knuth, renowned computer scientist*
The SAS Language

- Swiss Army Knife: data processing, statistical analysis, graphing/GIS, RDBMS access and more
- a combination of high-level, optimized PROCs and low-level DATAstep programming
- learn the SAS “way” of doing things
- use best-of-breed coding practices
- use short bits of PROC for optimal results interlaced with DATAstep code for flexibility and driven by SAS macros for reusability
- BY variable processing: sorting/merging/indexing/etc.
A Brief History of SAS

- 1966-8: Anthony Barr develops SAS language
- 1968: Barr and James Goodnight develop ANOVA and multiple regression procedures for SAS
- 1971-2: SAS popularity grows in academia, government and industry
- 1973: John Sall joins the project
- 1974: IBM creates Structured Query Language (SQL) Relational Data Bank Management System (RDBMS)
- 1976: SAS Institute is incorporated by Barr, Goodnight and Sall
- 1988: SAS v. 6 re-written in C for portability, adds support for UNIX, X11, SQL and RDBMS
- 1994+: GPL RASmacro library project
SAS Resources

- SAS v. 6/7/8 manuals in PCOR (please be considerate)
- The Little SAS Book: A Primer 4th Ed. (2008) $49.95 by LD Delwiche and SJ Slaughter
  SAS Press (available in PCOR and MCW book store)
- SAS Language and Procedures: Usage v. 6
- SAS v. 9.x manuals online http://support.sas.com/documentation/index.html plus “Knowledge Base/Tech Support”
- PCOR: http://www.mcw.edu/PCOR/Education/SAS.htm
- SAS-L mailing list http://www.listserv.uga.edu/archives/sas-l.html
A SAS Program

• a SAS program is a text file with a name ending in .sas: example1.sas

• copy some standard SAS settings:
  cp /opt/SAS/autoexec.sas 

• to manually submit a SAS batch job from the UNIX command line: sas example1.sas &

• generates a text log, .log, for notes and error messages and a text listing, .lst, for results:
  example1.log and example1.lst

• each SAS statement ends in a semicolon

• three styles of comments

  * comment statement starts w/ asterisk ;
  /* comment that is not a statement */
  %* macro comments are not echoed to the .log;
A SAS Script

- `/opt/local/bin/sas: sas example1.sas & (but, use emacs instead)
- `sas &` creates an interactive session
- `sas` saves 9 most recent versions of `.sas` program in `.USER sub-directory`
- `also, .log, .lst, .txt, .rtf, .ps, etc.`
- prevents multiple instances of the same SAS program from being run simultaneously
  very important for long SAS jobs which you could otherwise ruin by accidentally pressing F3 again
The SAS Macro Language

- SAS is a verbose language; lots of typing
- Macros allow you to write reusable code; macros generate the SAS code for you
- See that `options mprint; %annomac; are in your ˜/autoexec.sas`
- Macro variables start with a &
- Macro statements start with a %
- Two types of Macro statements
  - Use anywhere: `%global, %include, %let, %put`
  - In a macro definition only: `%macro/%mend, %local, %do/%end, %if-%then/%else`
- Macros, themselves, can be called anywhere:
  `%lowcase(), %upcase(), %_list(), etc.`
SAS Macro Language Resources

- RASmacro http://www.mcw.edu/PCOR/Education/SASMacros.htm
GPL RASmacro Library

- I started writing it in 1994 (my middle name is Allen)
- RASmacro names usually start with an underscore, i.e. /opt/sasmacro/_summary.sas and the documentation is self-contained
- Developed to generate table summaries of clinical trials: `%summary()`
- Required many building-block macros: `%count()`, `%level()`, `%list()`, `%nobs()`, `%printto()`, `%retain()`, etc.
- SPICE trial: table programs were 400-1200 lines
- with RASmacro: 40-120 lines
- approx. 90% reduction in the amount of code
- also, much faster: often, you can create a table in 15 minutes or less
The Summary Macro

%_summary(
    data=breast.le_cohort, stat=countpct pchi,
        col1=le, varorder=15 12-14, colorder=1 3,
        var12=e10nd1, format12=pos., order12=Yes\No,
        label12=Nodal\Involvement,
        var13=mdeath, format13=yesno.,
        var14=fup, format=2.,
        stat14=countpct mean_sd median_iqr ss3,
        var15=age_dx, format15=age.,
        label15=Age at\Diagnosis
    );
The Summary Macro

<table>
<thead>
<tr>
<th>Age at</th>
<th>LE (8.8%)</th>
<th>Total (100.0%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>66-</td>
<td>339 (10.7%)</td>
<td>3162 (100.0%)</td>
</tr>
<tr>
<td>71-</td>
<td>332 (10.1%)</td>
<td>3279 (100.0%)</td>
</tr>
<tr>
<td>76-90</td>
<td>409 (7.0%)</td>
<td>5805 (100.0%)</td>
</tr>
<tr>
<td>Missing</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Diagnosis

Pearson Chisq 43.9
DF 2
p-value <0.0001
## The Summary Macro

<table>
<thead>
<tr>
<th>Fup</th>
<th>1080 (8.8%)</th>
<th>12246 (100.0%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean (SD)</td>
<td>54.7 (11.2)</td>
<td>53.6 (12.7)</td>
</tr>
<tr>
<td>Median (IQR)</td>
<td>60.0 (0.0)</td>
<td>60.0 (5.0)</td>
</tr>
<tr>
<td>Q1, Q3</td>
<td>60.0, 60.0</td>
<td>55.0, 60.0</td>
</tr>
<tr>
<td>Min, Max</td>
<td>12.0, 60.0</td>
<td>12.0, 60.0</td>
</tr>
<tr>
<td>ANOVA DF</td>
<td>1, 12244</td>
<td>9.9 (0.0016)</td>
</tr>
<tr>
<td>ANOVA F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>4 (5.7%) (0.4%)</td>
<td>70 (100.0%) (</td>
</tr>
<tr>
<td>13</td>
<td>5 (7.4%) (0.5%)</td>
<td>68 (100.0%) (</td>
</tr>
<tr>
<td>14</td>
<td>6 (9.1%) (0.6%)</td>
<td>66 (100.0%) (</td>
</tr>
<tr>
<td>15</td>
<td>2 (2.6%) (0.2%)</td>
<td>77 (100.0%) (</td>
</tr>
</tbody>
</table>
data carrier;
set %_list(carrier1999-carrier2008);
by bene_id;

array _dgns(9) dgnscd1-dgnscd9;

do i=1 to dgnscnt;
    if _dgns(i) %_prefix('042'-'044', 'V08')
        then aids=1;
    else if _dgns(i) %_in('36511'-'36514')
        then glaucoma=1;
end;
run;
RASmacro Examples: Use Anywhere

data carrier;
  set %_list(carrier1999-carrier2008);
  by bene_id;

  array _dgns(9) dgnscd1-dgnscd9;

  do i=1 to dgnscnt;
    select(_dgns(i));
    %_when(’040’–’047’, ’V08’) aids=1;
    %_when(’36511’–’36519’, by=2)
      glaucoma=1;
    otherwise;
    end;
  end;
run;
RASmacro Examples: Macro Definition

%macro check (data=%upcase(&syslast), var=);
%local i;
%let var=%_list(&var);
%do i=1 %to %_count(&var);
  %local var&i;
  %let var&i=%scan(&var, &i, %str( ));
  %put VAR&i=&&var&i;
data check; %*CHECK DATASET created;
  ...
data check; set &data; %* &DATA=? ;
%end;
%mend check;
%check (var=dgnscde dgnscdl1-dgnscdl10); *call macro;
What is a Warehouse?

- A Warehouse summarizes long or complex calculations for convenience
- For example: to identify breast cancer patients, you need to summarize breast cancer procedures/surgeries, breast cancer diagnoses and non-breast cancer diagnoses from inpatient and outpatient claims
- Warehouses have unique keys; generally, one record per patient which allows them to be easily merged with unique and/or non-unique datasets
- Therefore, Warehouses usually contain arrays so that a lot of information can be conveniently summarized in one record per patient (the arrays themselves are referred to as Warehouses as well)
Warehouse Example

- Medicare eligibility and enrollment
- Beneficiary Summary File (BSF): one file per year with 12 variables (one per month) for each of 4 Medicare programs, i.e. parts A/B/C/D, and one record per file for each patient
- SEER-Medicare helpfully combines each of the yearly files into one big file with one record for each patient
- But, how do you determine if a patient is enrolled in Part D from June 2006 until December 2008?
- Create a warehouse that records the number of consecutive months enrolled in each Medicare program for each month and year (one record per patient): same number of variables, but the information is more usefully organized
Warehouse Example: Part D Enrollment

- SES/Part D study: CCW.DENOM created by /ses/sas/ccw/denom.sas
- SEER-Medicare: CANCER.DENOM created by /seermedicare/sas/new/denom.sas
- Part D enrollment is represented by the array D200601--D200812
- Each element of the array contains the number of consecutive months enrolled
- D200601=0 means that they were not enrolled in January 2006, while D200601=12 means that they were enrolled for the whole year (missing means either there was no record for that year and/or deceased)
- D200606=31 enrolled June 2006 to December 2008
Conclusions

- SAS is a best-of-breed tool for data processing and data analysis
- embrace the “SAS way”: sorting/merging/retaining/etc.
- SAS was developed by statisticians for statisticians
- many SAS resources are now freely available
- the SAS Macro Language simplifies complex tasks
- the RASmacro Library provides the building blocks for SAS macro programming
- Warehouses are convenient summaries of long/complex calculations