Week 1 [24-26 Aug 05] Class Activities

FILE:
Hp/compaq: C:\baileraj\Classes\Fall 2005\sta402\handouts\week-01-22aug05.doc

[BASED on C:\Documents and Settings\John Bailer\My Documents\baileraj\Classes\Fall 2003\handouts\day1-27aug03.doc and ..\day2

0. Roster

STA 402 (n=4)
Bruestle, Ryan D
DePalma, Glen R
Durst, Adam M
Lucas, Catherine D

STA 502 (n=13)
Agyei, Festus O
Al-Shaikh, Enas S
Fadel, William F
Fenchel, Matthew C
Ghattas, Adam E
Hackett, Adam L
Kang, Wooyoung
Mo, Min
Schuurmann, James G
Stanev, Stefan S
Stasek, David J
Wei, Xingtao
Wu, Jian

1. Information cards

Please follow the link from the syllabus page to complete the following information -

FALL 2004
STA 402 or STA 502
Name
Dept./major
Undergrad. School
Prev. stat. classes
Prev. math. Classes
Prev. computer or prog. Classes
Why are you taking this class?
Something that will help me get to know you better
2. Syllabus [http://www.users.muohio.edu/baileraj]

or http://www.users.muohio.edu/baileraj/classes/sta402/sta402-syllabus-f05-22aug05.htm (in particular)

What this class is …

Statistical programming = coding required to conduct an analysis of interest

What this class isn’t …

Statistical computing = efficient coding of statistical procedures (e.g. numerical analytic methods, random number generation, etc.)

Data Management = the structuring and storage of data files

Why SAS first and then S-Plus/R?

Why not do both together?

3. Tour of the Class web pages PLUS finding SAS/R/S-Plus on campus

Point a browser at http://www.users.muohio.edu/baileraj.

SAS
* Students are allowed to install SAS on their home machines using the instructions in the Installing SAS Version 9.1.3 for Windows (see http://www.muohio.edu/quantapps and follow the “SAS System” link and the “Installing SAS 9.1.3 for Windows at Miami University” link)
* NOTE: You need the Novell Netware Client installed on their machine and be logged in before installation begins.
* Labs containing SAS besides the Math/Stats lab are the Learning Technology Center Lab in 200 Gaskill, the School of Business labs in the basement of Laws Hall. Psychology labs in Benton have SAS installed, but some machines still have version 6.1.2.
* You can also run SAS on unixgen.muohio.edu (more to come)

R/S-Plus
* getting R for yourself - http://www.r-project.org/
* R/S-Plus available on labs in Bachelor and in Upham

4. Safe computing
Practice safe computing. Be wary of attachments. If you send me code with questions, then please send this code in the body of the email message – NOT as an attachment.

BACKUP your work. BACKUP your work. BACKUP your work. Keep your work on the server on your UDS (your “M” drive). Keep a copy on your local computer. Keep a computer on a flash memory device. Expect your hardware to fail. How much would you lose if you had a catastrophic failure?

5. Example SAS programs

We start our exploration of SAS with a couple of examples …

Files can be found on the web at http://www.users.muohio.edu/baileraj and then following class links or Finding the server \Muserver2\USERS\B\BAIлерA\public.ww\classes\sta402\data

ch2.sas
code051403.txt
KS-fish_4-15-02__pre-post-delta_.txt
remerge-with-addresses-09jan01.sas

BASIC CONCEPTS – week 1

* Review basic concepts of statistical computing and research data management
* Introduce SAS data sets
* Review the form of SAS Statements and SAS names
* Introduce SAS procedures
* Review the structure of SAS programs
* Describe SAS data libraries and what they can contain
* Show documenting SAS programs using comments
* Illustrate running SAS programs and basic debugging

Programming practice

0. Document your programs!
   * file location
   * date
* author
* revision (based on previous program?)
* purpose of the program?
* input variables? output variables?

1. Use meaningful variable names (e.g. X1 vs. dose_ppm)
   * SAS variable names can contain underscores and up to 32 characters.
   * WARNING: S-Plus/R does NOT allow underscores in variable names. In fact, underscores can be used as assignment operators in S+/R. SAS does not allow periods in variable names while S+/R does. Beware when going back and forth between packages.

2. DON’T ONLY USE CAPITALS IN PROGRAM STATEMENTS

3. Indent program statements that naturally go together

```plaintext
compare
PROC GLM;
CLASS SMOKE;
MODEL Y = X|SMOKE;
LSMEANS SMOKE;
RUN;

with
proc glm;
   class SMOKE;
   model Y = X | SMOKE;
   lsmeans SMOKE;
run;
```

**Example 1.1: A SAS program that fits a multiple regression**

/* mreg-country-23jan05.sas

Directory: C:\baileraj\Classes\Spring '05\programs\regression-examples
[Laptop]
Author: John Bailer
Purpose: multiple regression example where average life expectancy of women is modeled as a function of country characteristics

Input data file -------------------------------
\\Casnov5\MST\MSTLab\Baileraj\country.data

Input variables -------------------------------
Name = country name (Character variable)
Area = country area
Popsize = population size
Pcturban = % residents in urban setting
Lang = primary language (Character variable)
Liter = % literate
Lifemen = average life expectancy men
Lifewom = average life expectancy women
PcGNP = per capita gross national product

Created Variables -----------------------------------
  logarea = log10(area);
  logpopn = log10(popnsize);
  loggnp  = log10(pcGNP);
  ienglish = (lang="English");

Data Source: Extracted from World Almanac
*/

data country;
  title 'country data analysis';
  infile "\Casnov5\MST\MSTLab\Baileraj\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;

proc print;
  run;

proc reg data=country;
  title "LITER and LOGGNP as predictors of Life expectancy of women";
  model lifewom = liter loggnp/ tol vif collinoint;
  output out=new p=yhat r=resid;
run;

LAB: “Fix” the following code. The variable “Y” was the yield of some rice (t/ha) as a function of minimum temperature “mint” (deg. C).

* SAS code to fit quadratic regression;
 DATA RICE; INPUT Y MINT @@;
    MINTBAR = 23.1975; X = MINT - MINTBAR; X2 = X*X;
 DATALINES;
  2.8   27.2   3.1   28.1   2.3   29.2   3.0   23.4
  2.1   25.7   2.3   26.2
  3.6   26.3   2.4   25.9   3.1   26.0
  3.8   26.2   2.4   26.4   4.4   23.4
  3.5   24.5   3.0   23.7   2.6   23.1
  4.2   21.7   3.1   22.5   4.8   23.1
  3.1   23.2   3.3   23.6   4.5   23.7
  2.9   23.9   3.8   24.4   3.1   24.0
  3.2   23.9   3.7   23.5   3.3   22.9
  4.7   20.0   3.5   23.7   3.4   22.5
  3.2   23.0   3.2   22.4   4.5   21.2
  5.0   19.2   6.2   17.4   6.2   19.0
  6.0   19.0   6.1   18.8   7.3   18.0
PROC REG; MODEL Y = X X2 ; RUN;
SAS programs are typically stored as text files with “.sas” extensions.

Running a SAS program generates two additional files – a “log” file (containing messages describing what happened during the execution of the SAS commands) and a listing file (“listing” or “lst” or “lis” if produced when running SAS in a batch mode). Typically, we run SAS interactively.

SAS programs consist of a series of DATA blocks and/or PROCEDURE blocks.

1. DATA blocks are used to enter and manipulate data …
   * naming variables
   * typing variables (numeric, character)
   * identifying data source (datalines, infile, set from libname)

2. PROCEDURE blocks are used to conduct specific procedures including
   * descriptive statistics for variables (PROCs UNIVARIATE, MEANS, FREQ, etc.)
   * modeling procedures (PROCs REG, GLM, GENMOD, LOGISTIC, PHGLM, etc.)
   * some data and output management (IMPORT, EXPORT, CONTENTS).

The general syntax for a SAS procedure is
PROC name;
   subcommand 1;
   subcommand 2;
run;

Subcommands may be required in some PROCs, e.g.

    proc plot;
       plot TOTAL*CONC;
    run;
or may be optional;

```sas
proc print;
  var TOTAL CONC;  * print only two variables from current data set;
run;
```

```sas
proc print;
  run;  * prints all variables in the current data set;
```

---

### Introduce SAS data sets and libraries

SAS data can be entered as part of a SAS program using “datalines” [or “cards”] option

```sas
data;
  input x y @@;
  datalines;
  1 2 3 2 5 7
;
```

SAS data can be entered from an external file using “infile”

```sas
data d1;
  infile 'C:\Documents and Settings\John Bail\My Documents\baileraj\Classes\Fall 2003\SAS-
  programs\ch2-dat.txt' expandtabs missover pad firstobs=16;
  * infile
  "\\Muserver2\USERS\B\BAILERAJ\public.ww\classes\sta402\SAS-programs\ch2-
  dat.txt"
  firstobs=16 expandtabs missover pad ;  * referencing M drive
directly;
  input @9 animal 2.
    @17 conc 3.
    @25 brood1 2.
    @33 brood2 2.
    @41 brood3 2.
    @49 total 2.;
```

SAS data can be entered from an existing SAS library using “libname” and “set” (Note that SAS libraries may contain both data sets and formats)

```sas
Libname insas 'd:\baileraj\grants\odhs-oct98\Datasets'; **** changed 5jul00;
data mdsmar98;  set insas.summar98;  matcher = provider;
```

Data can be constructed in a SAS data set and “output” for later use

```sas
data junk;
```
do kk = 0 to 10 by 0.01;
  x = kk;
  y = 3 + 2*kk + rannor(0);
  output;
end;

proc plot;
  plot y*x;
  run;

proc reg;
  model y=x;
  run;

Data can also be imported into SAS from other applications such as Excel (import wizard or PROC IMPORT)

**Documenting SAS programs**

Comments can be incorporated in SAS using a couple of strategies.

1. Statements beginning with an asterisk. For example,

   proc glm;   * fitting ancova model with covariate x and classification variable SMOKE;
               class SMOKE;
               model SBP = x | SMOKE;
               run;

2. Statements beginning with /* and ending with */. For example,

   proc glm;   /* fitting ancova model with covariate x and classification variable SMOKE */
               class SMOKE;
               model SBP = x | SMOKE;
               run;

               * I use both forms although the /* ---- */ is nice for commenting out blocks of code

3. You can use TITLE statements to annotate output. For example,

   proc glm;
   title ‘fitting ancova model with covariate x and classification variable SMOKE’;
               class SMOKE;
               model SBP = x | SMOKE;
Illustrate running SAS programs and basic debugging

Let’s explore this with the program

```sas
data junk;
  do kk = 0 to 10 by 0.01;
    x = kk;
    y = 3 + 2*kk + rannor(0);
  output;
end;

proc plot;
  plot y*x;
run;

proc reg;
  model y=x;
run;
```