Statistical Computing with SAS

P6110: Lecture Notes

Jihui Lee

Department of Biostatistics

Columbia University

COLUMBIA UNIVERSITY | MAILMAN SCHOOL of PUBLIC HEALTH BIOSTATISTICS

Chapter 1. A Dive into SAS

1.1. Statistical Analysis System (SAS)

- Initially started as an agricultural research project at North Carolina State University in 1966.
- Primarily leased to other agricultural departments to analyze the effect soil, weather and seed varieties had on crop yields in the early 1970s.
- SAS Institute Inc. was founded as a private company in 1976.
- "Best Company to Work For" in Fortune's annual rankings each year since 1997.
- Used at more than 75,000 sites in 147 countries.
- Some components
  - Base SAS: Basic procedures and data management
  - SAS/STAT: Statistical analysis
  - SAS/GRAPH: Graphics and presentation
  - SAS/OR: Operations research

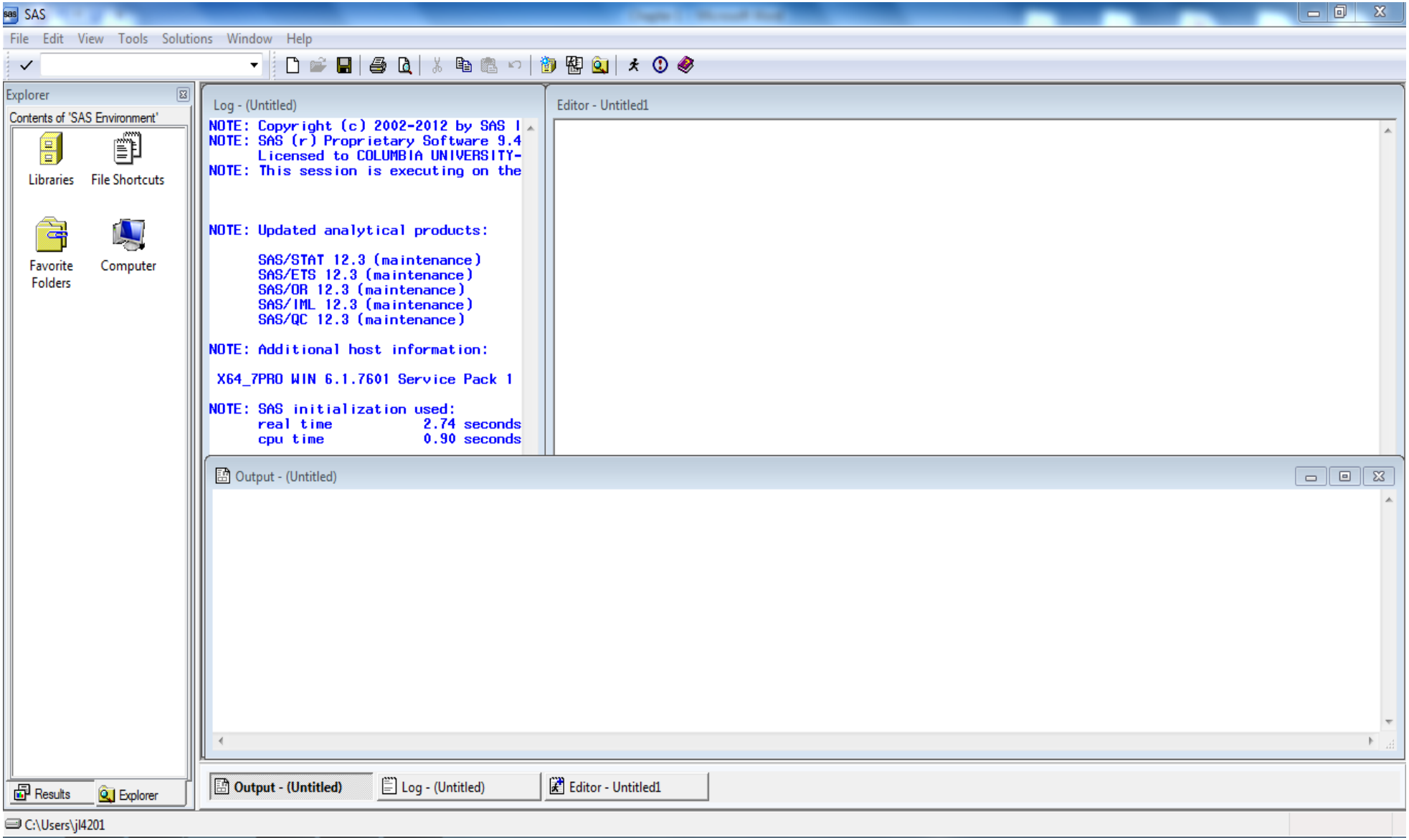
- SAS/ETS: Econometrics and time series analysis
- SAS/IML: Interactive matrix language
- SAS/QC: Quality control
- SAS/INSIGHT: Data mining

1.2. Why SAS?

- History: Long history, wide range of procedures
- Popularity: Useful in job market
- Reliability: Quality control and customer service by experts
- Big data: Great for (large) data manipulation
- Documentation: Neat and well-structured results
- Help: Detailed help documentations and web pages

1.3. SAS Windows

- Editor: Write, edit, and submit SAS programs
- Log: Notes about the SAS session including errors and warnings related to the submitted SAS programs
- Output: Any printable results
  - HTML (Result Viewer; Default for SAS 9.3 or later) vs listing
  - Tools → Options → Preferences → Results → Select 'Create listing'
- Results: Tree list of contents for the Output window
- Explorer: Access to SAS data files and libraries


SAS
File Edit View Tools Solutions Window Help
Explorer
Contents of 'SAS Environment'
Libraries
File Shortcuts
Favorite Folders
Computer
Results
Explorer
Log - (Untitled)
NOTE: Copyright (c) 2002-2012 by SAS I
NOTE: SAS (r) Proprietary Software 9.4
Licensed to COLUMBIA UNIVERSITY-
NOTE: This session is executing on the
NOTE: Updated analytical products:
SAS/STAT 12.3 (maintenance)
SAS/ETS 12.3 (maintenance)
SAS/OR 12.3 (maintenance)
SAS/IML 12.3 (maintenance)
SAS/QC 12.3 (maintenance)
NOTE: Additional host information:
X64_7PRO WIN 6.1.7601 Service Pack 1
NOTE: SAS initialization used:
real time 2.74 seconds
cpu time 0.90 seconds
Editor - Untitled1
Output - (Untitled)
Output - (Untitled)
Log - (Untitled)
Editor - Untitled1
C:\Users\jl4201

1.4. Layout of SAS Programs

Step	Role	Example

DATA step	Read, create, and manipulate data	`data Distance;`
		`  Miles = 26.22;`
		`  Kilometers = 1.61 * Miles;`
		`run;`

PROC step	Perform analysis and generate output	`proc print data = Distance; run;`

- Every statement ends with a semicolon (;).
- A statement can be more than one line.
- A statement can be on the same line as other statements.
- A statement can start in any column.
- NOT case sensitive! (except “quoted” strings)
- Color-coded!
- Check SAS log for important notes.

Example

Editor

```
* Create a SAS dataset named 'distance';
data Distance;
     Miles = 26.22;
     Kilometers = 1.61 * Miles; * Convert miles into kilometers;
run;

* Print the results;
proc print data= Distance;
run;
```

Log

```
NOTE: Copyright (c) 2002-2012 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software 9.4 (TS1M0)
    Licensed to COLUMBIA UNIVERSITY-HEALTH SCIENCES
CAMPUS-T&R SFA, Site 70080790.
NOTE: This session is executing on the X64_7PRO  platform.
NOTE: Additional host information:
 X64_7PRO WIN 6.1.7601 Service Pack 1 Workstation
NOTE: SAS initialization used:
     real time          0.57 seconds
     cpu time           0.51 seconds

1    * Create a SAS dataset named 'distance';
2    data Distance;
3        Miles = 26.22;
4        Kilometers = 1.61 * Miles;
5    run;
NOTE: The data set WORK.DISTANCE has 1 observations and 2
variables.
NOTE: DATA statement used (Total process time):
     real time          0.01 seconds
     cpu time           0.01 seconds

6
7    * Print the results;
8    proc print data= Distance;
NOTE: Writing HTML Body file: sashtml.htm
9    run;
NOTE: There were 1 observations read from the data set
WORK.DISTANCE.
NOTE: PROCEDURE PRINT used (Total process time):
     real time          0.60 seconds
     cpu time           0.54 seconds
```

Output

Result Viewer

Obs	Miles	Kilometers
1	26.22	42.2142

Listing

Obs	Miles	Kilometers
1	26.22	42.2142

Example

SAS Code

```
data Exam;
     input Name $ Exam1 Exam2 Exam3;
     cards;
     Emma 95 75 85
     Noah 89 . 99
     Liam 88 98 78
     Olivia . 70 80
     ;
run;
proc print data=Exam; run;
```

Output

Obs	Name	Exam1	Exam2	Exam3
1	Emma	95	75	85
2	Noah	89	.	99
3	Liam	88	98	78
4	Olivia	.	70	80

```
proc print data=Exam (obs=3);
     title 'Exam Dataset';
run;
```

Exam Dataset

Obs	Name	Exam1	Exam2	Exam3
1	Emma	95	75	85
2	Noah	89	.	99
3	Liam	88	98	78

```
proc print data=Exam noobs;
     var name exam2;
run;
```

Name	Exam2
Emma	75
Noah	.
Liam	98
Olivia	70

```
proc print data=Exam
style(data)={background=yellow};
     var name / style(data)={font_style=italic
font_weight=bold};
     var exam1-exam2;
run;
```

Obs	Name	Exam1	Exam2
1	*Emma*	95	75
2	*Noah*	89	.
3	*Liam*	88	98
4	*Olivia*	.	70

1.5. SAS Statements: Rules

- SAS variable/dataset names
  - Must be 32 characters or fewer in length.
  - Contain letters, numbers, and underscores (_).
  - Start with a letter or an underscore.
  - If no name is given for a dataset, SAS creates default names of the form 'data*n*' where *n* is an integer. (e.g. data1, data2, …)
- Missing values are represented by a period (.).
- Comments can be added in two ways: `* Comment;` or `/* Comment */`

1.6. Useful Tips

- Include a header for every program: Title, purpose, author, date
- Name your dataset and variables in a concise yet informative way.
- Select the part of programs you want to run before clicking the running button. Otherwise, SAS runs the whole program.
- Write neat and straightforward programs.
  - One SAS statement on one line
  - Include detailed comments: Easy to understand the logistics of SAS codes.
  - Avoid too many loops. (e.g. DO, IF-ELSE)
- Test each part of your program: Check datasets, output, and *log*.
- Macros can be useful when some codes should be repeatedly executed.
- Take advantage of HELP menu and worldwide network of SAS users.