Stat 302
Statistical Software and Its Applications
SAS Basics

Fritz Scholz

Department of Statistics, University of Washington

Winter Quarter 2013
Access the virtual lab on the terminal server ts.stat.washington.edu, as explained in the InstallInstructions.pdf, page 3, section 3.

Run SAS: → Start (lower left on task bar) → All Programs → SAS → SAS 9.3 (English)

You may also want to create a SAS start icon on your taskbar, just drag and drop the above SAS 9.3 (English) to the taskbar.
SAS language conventions

- Each statement needs a semi-colon ; at its end, to signal the end of the statement (the caboose).
- Unlike R the SAS language is not case sensitive.
- One statement can go over multiple lines, until terminated by ;
- Several statements (each terminated by ; ) can be on same line. Not recommended for readability. May do it for slides.
- Statements don’t have to start flushleft, indent for readability.
- Naming variables and data sets:
  - 32 characters or less
  - must begin with a letter or underscore _
  - other characters must be letters, numbers, or _
  - no dashes “-” or spaces “ ” or other characters
  - the period "." has special role, see later
Be liberal with comments in any SAS program. Comments are anything between * and ; or anything between \* and \*. The latter is the same as in the C programming language. Either can stretch over several lines.

* a comment stretching over 2 lines;

\* a comment running over 3 lines with a ; in between *

\*************************************
* a comment running *
* over 5 lines *
* with a ; in between *
************************************

Two common types of SAS steps:
- data
- proc

End each data or procedure statement with a `run;` statement.

Only two data types:
- numeric
- character (specified in the data step)

Give meaningful names to data and variables

Show program segmentation through indentation structure.

`title` statement before each procedure that produces output. It will show on the output. It organizes your work.

Run commands in the `active` Editor window by clicking...
* First SAS program;
data patient_data;
   *names the data set for future reference;
   input Age Sex $ Height Weight;
   * the $ after Sex designates Sex as a character variable;
datalines; *this initiates data reading from here;
   33 F 65 130
   48 M 71 160
run; * done with data reading;

title "Patient Data";
proc print data=patient_data;
run; * this prints the data;
We said “case” does not matter.

e.g., the variable Age can be referred to as age in the same program, each referring to same variable.

However, its first usage will reflect in output.

In the title statement case matters for the output.
### Basic Math

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition</td>
</tr>
<tr>
<td>−</td>
<td>Subtraction</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication</td>
</tr>
<tr>
<td>/</td>
<td>Division</td>
</tr>
<tr>
<td>**</td>
<td>Exponentiation</td>
</tr>
<tr>
<td>−</td>
<td>Negation</td>
</tr>
</tbody>
</table>

The usual precedence order applies $** < *, / < +, −$.

$5 − 3 \times 4 = −7$ and $−5 \times 2 = −25$, not 25 as in Excel or C.

SAS, Fortran and R do it correctly.
Example: Creating a New Variable in Data Step

* Second SAS program;
data patient_data;
  input Age Sex $ Height Weight;
  * Compute BMI (body mass index);
  BMI=(Weight/2.2)/(Height*.0254)**2;
  * New variables are defined prior to data entry;
datalines; *this initiates data reading from here;
  33 F 65 130
  48 M 71 160
run;

title "Patient Data with BMI";
proc print data=patient_data;
run;
* Second SAS program;
data patient_data;
   input age Sex $ Height Weight;
   * note lower case age;
   * Compute BMI (body mass index);
     BMI=(Weight/2.2)/(Height*.0254)**2;
datalines; *this initiates data reading from here;
33 F 65 130
48 M 71 160
run;

title "Patient Data with BMI";
proc print data=patient_data;
run;
Patient Data with BMI

<table>
<thead>
<tr>
<th>Obs</th>
<th>Age</th>
<th>Sex</th>
<th>Height</th>
<th>Weight</th>
<th>BMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>33</td>
<td>F</td>
<td>65</td>
<td>130</td>
<td>21.6784</td>
</tr>
<tr>
<td>2</td>
<td>48</td>
<td>M</td>
<td>71</td>
<td>160</td>
<td>22.3621</td>
</tr>
</tbody>
</table>
For each run of a SAS program the results will accumulate in the Results Viewer - SAS Output pane.
You can clear those results by: ⇒ Edit ⇒ Clear All
The same way you can clear other active panes.
Save as PDF from Results Viewer - SAS Output as follows:

- \( \Rightarrow \) File \( \Rightarrow \) Print, put slider on Select Printer to far left, select Adobe PDF \( \Rightarrow \) Print,
- select UDrive on the left panel, navigate to folder for saving the file, enter file name for saving \( \Rightarrow \) Save.

To use it in LaTeX on your physical machine you will have download it from your UDrive via FileZilla SFTP, or use whatever other method you have for interacting with the UDrive.

See the instructions on trimming and clipping unneeded white space from such graphics for use in LaTeX, as explained in LaTeXArticle.pdf, dealt with earlier.
How to Save and Retrieve Your SAS Programs

- First create a folder with name “My SAS Files” on your UDrive.
- In SAS make the Editor pane active ⇒ File ⇒ Save AS
- In the “Save As” window under “Save in” navigate to “My SAS Files” on your UDrive.
- Under “File name” give a name to the file in which you want to save your SAS program, say “My First SAS”, ⇒ Save.
- That will have saved that SAS code in the Editor pane in U:\My SAS Files\My First SAS.sas the .sas was automatically added.
- Now clear the Editor pane: ⇒ Edit ⇒ Clear All
- Load that program back in: ⇒ File ⇒ Open Program, and navigate to the saved program, select My First SAS.sas ⇒ Open. You will see your program again in the Editor pane.
Input the following data into SAS, call it `student_data`:

<table>
<thead>
<tr>
<th>Student</th>
<th>Quiz</th>
<th>Midterm</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>89</td>
<td>97</td>
<td>90</td>
</tr>
<tr>
<td>B</td>
<td>78</td>
<td>68</td>
<td>80</td>
</tr>
<tr>
<td>C</td>
<td>98</td>
<td>95</td>
<td>99</td>
</tr>
</tbody>
</table>

While reading the data, be sure to create a variable named `class_score`, which is calculated by adding 20% of quiz, 30% of midterm and 50% of final.

Print the data set to your screen.

Save your code as a `.txt` file using Notepad.
Terminating your Virtual Lab Session

- When you are done with your remote desktop session, don’t just click the × in the upper right corner of its window.
- That terminates the session, but any programs initiated in it (like SAS) will continue to run. You can resume them when you start another session of the terminal server.
- That ties up resources, like SAS, for which we have a limited number of licenses across campus.
- Thus close down SAS before exiting the server.
- Then log out from the server, don’t just hit the ×.
- You will notice that you will see different desktop layouts from session to session. That’s because the system assigns you randomly to one of two servers, and they look different.
- This also means that any file you leave on one virtual desktop won’t be on the other. To have access to your files either way, put them on the UDrive.