

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

0. Downloading Data

You will have to download all of the data used from the internet before SAS can access the data.

If the file is accessed via a link, then right click on the file name and save it to a directory on your hard drive. It can be difficult to find the files, so if you are unfamiliar with PCs, I suggest that you download it directly to the W drive (main campus drive).

SAS cannot read a zipped file. Therefore, if any file is zipped, remember to extract it before you try to access it in SAS.

Please remember where your files are stored so that you can access them for the labs.

1. General Information for SAS


a) Output must be copy/pasted or 'snipped' when it is required. No credit will be given if the output is retyped.

b) To help in debugging SAS code, the SAS editor color codes the information. The colors can help you debug the code so I recommend that you memorize them or refer back to this listing. The following are some of the colors used by SAS:

blue or blue:	commands	black:	responses	green:	comments
blue green:	numbers	purple:	text/titles	black:	data

c) In the code that is presented, there are a large number of comments (in green). Please read them to understand what the variables represent. Comments are not required in your submitted code.

d) All command lines in SAS must end in a semicolon ';'.

e) To run a program, be sure that the editor window is active and then Run → Submit or just click on the . In addition to running all of the code, you may highlight part of the code and only run that part.

f) If the file does not run as you expect, please look in the Log screen; specifically at the information in red or green to see if that helps you find the problem.

g) SAS will append the output to the Result Viewer and Log screens which can cause great confusion when you are doing more than one problem in a session. If you are running the programming locally, that is, you are NOT using goremote, I would strongly suggest that you add the following code to the beginning of each program, this will clear both of these screens when you run the code:

```
ods html close;  
ods html;
```

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

- h) I recommend that you use a separate name for each data set and modification. This will help you differentiate the purpose that you are using the data for. In addition, if you make a mistake, you still have the original data set to fall back on. That is, if you need to modify the data set for any reason (like cleaning it or creating a new variable), you should use a new name. The names need to be one word (no spaces) and start with a letter. The only special character that can be used is the underscore '_' which can be used to indicate a space. Capitalization is important. Please change your data set names and variable names to match each situation. Points will be deducted if this is not done.
- i) When you submit the code in your lab report, please include all of the code for the question at the beginning of the question. This makes it easier to replicate your results.
- j) Do not print out the whole data set inside of the program! We will take off points if there is code for this in your submission.
- k) In SAS, if you input the file by using the command line, you need to input ALL of the variables; not just the variables that are used to answer the question.

2. Importing Data Sets, Cleaning, Manipulating, and Printing Them.

Hummingbirds and flowers. (Data Set: *helicon_m.txt*) Different varieties of the tropical flower *Heliconia* are fertilized by different species of hummingbirds. Over time, the lengths of the flowers and the form of the hummingbirds' beaks have evolved to match each other. Here are data on the lengths in millimeters of three varieties of these flowers on the island of Dominica:

<i>H. bihai</i>							
47.12	46.75	46.81	47.12	46.67	47.43	46.44	46.64
48.07	48.34	48.15	50.26	50.12	46.34	46.94	48.36
<i>H. caribaea red</i>							
41.90	42.01	41.93	43.09	41.47	41.69	39.78	40.57
39.63	42.18	40.66	37.87	39.16	37.40	38.20	38.07
38.10	37.97	38.79	38.23	38.87	37.78	38.01	
<i>H. caribaea yellow</i>							
36.78	37.02	36.52	36.11	36.03	35.45	38.13	37.1
35.17	36.82	36.66	35.68	36.03	34.57	34.63	

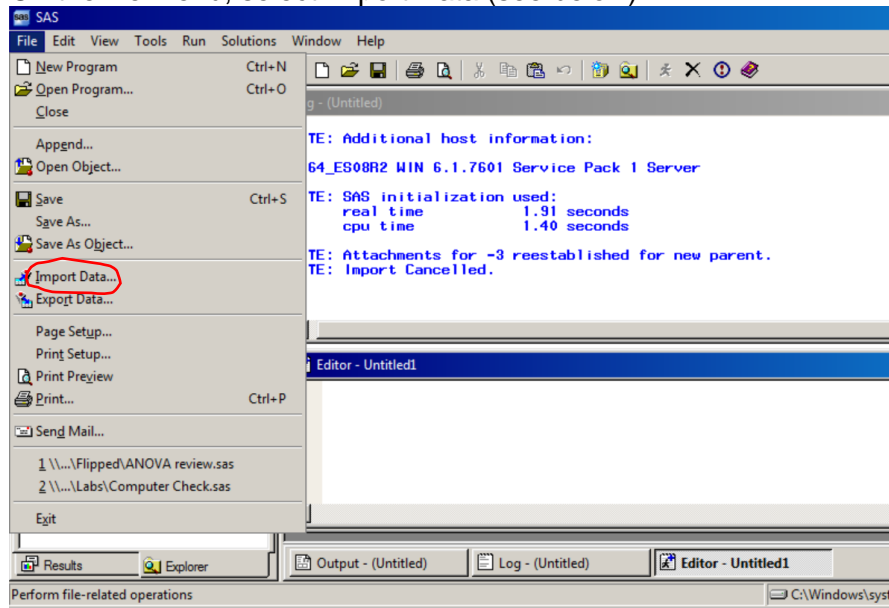
SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

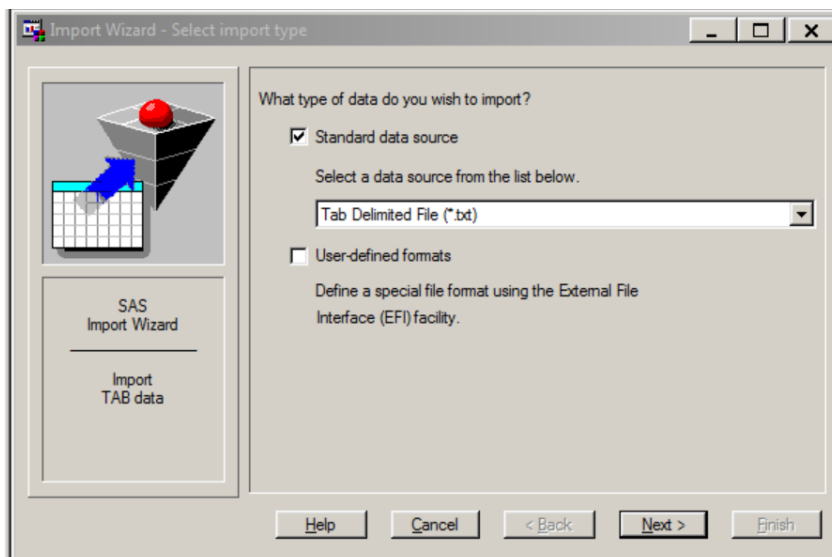
Import Data

There are two ways to import your data into SAS; using the GUI interface and using a command line. The advantages of the GUI interface is that you do not need to know what the variables are or whether they are categorical or numerical before you write the code. The disadvantage is that SAS might incorrectly define a categorical variable as numerical. If this becomes a problem, then code will be provided to change the data type into the correct one. The GUI interface is provided first with the code listed second separated by a horizontal line.

On the file menu, select Import Data (see below)



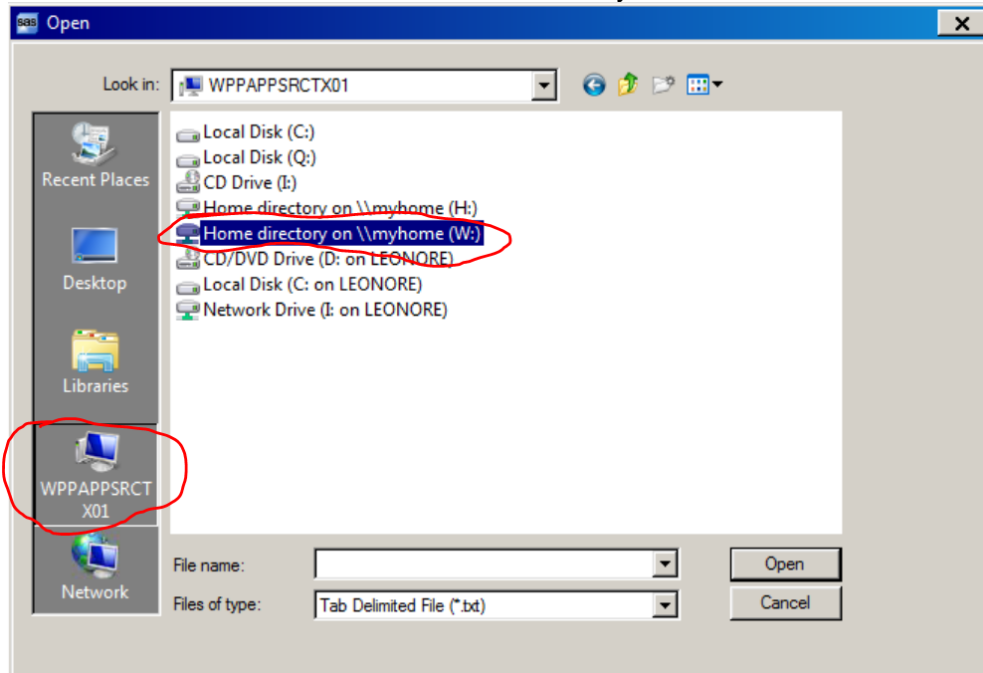
Select Tab Delimited File (.txt) see below and then select "Next >"



SAS Tutorial for Lab 1

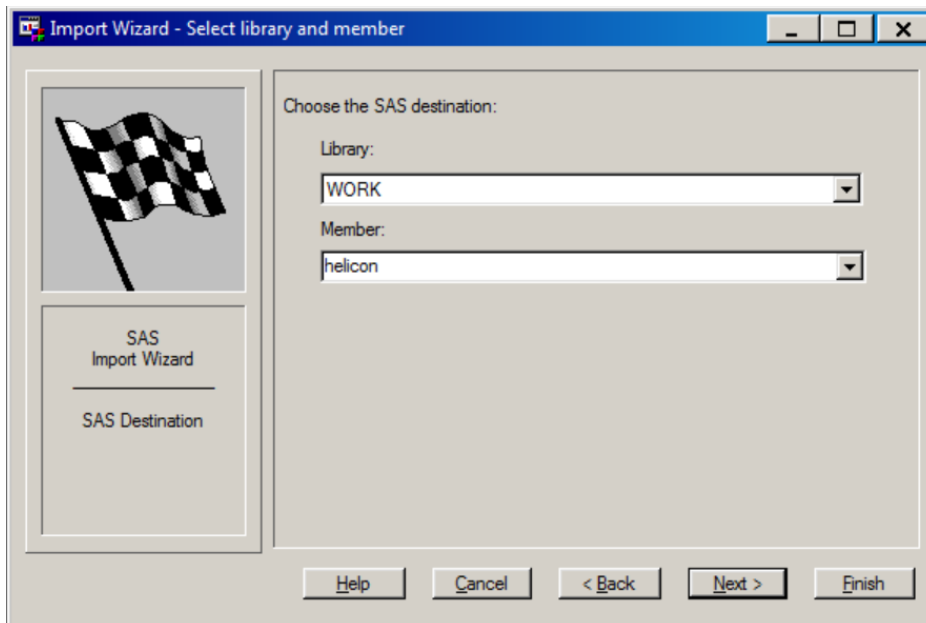
Author: Leonore Findsen, Cheng Li, Min Ren

Select the WPPAPPSRCT X01 and browse for your file on the W drive.



Then click Next >.

On the next screen select WORK (default) and put in the data name if you want to change it from the name of the file name.



Then select Finish.

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

The code that was used to import the data was (This was copied from the Log screen)

```
1/*****
2      *   PRODUCT:    SAS
3      *   VERSION:    9.4
4      *   CREATOR:    External File Interface
5      *   DATE:       27AUG17
6      *   DESC:       Generated SAS Datasheet Code
7      *   TEMPLATE SOURCE: (None Specified.)
8*****/
9      data WORK.helicon      ;
10     %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
11     infile 'W:\helicon_m.txt' delimiter='09'x
12 ! MISSEVER DSD lrecl=32767 firstobs=2 ;
13     informat Variety $5. ;
14     informat Length best32. ;
15     format Variety $5. ;
16     format Length best12. ;
17     input
18             Variety $
19             Length
20     ;
21     if _ERROR_ then call symputx('_EFIERR_',1); /* set ERROR
detection macro variable */
22     run;
```

SAS will generate errors if the data set is not clean; that is, it contains 'NA's. If you are concerned about the errors in the Log screen, ask your TA to be sure that everything is ok.

To report this code: File → Import Data → Tab Delimited File (.txt) → Next → browse for helicon_m → Library: Work, Member: helicon → Finish

```
data helicon;
  infile 'W:/helicon_m.txt' delimiter = '09'x firstobs = 2 ;
  length variety $ 10;
  input variety $ length;
  /*delimiter = '09'x means the file uses tabs as delimiters
  firstobs = 2 means we start reading the data from the second line,
  since the first line is the name of variables.*/
  /*The length command is required if the name in the first row
  is shorter than the name in the following categories. In USData,
  you will need to set the IncomeCategory to a length of 11 */
  /*Since variety is a categorical variable, a '$' follows it.*/
run;
```

Remember if you use this method, you will need to set the length of the variety to at least 11. The length command needs to be **before** the input line.

```
proc print data = helicon; run ;
```

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

I am only printing out the whole data set because it is small. **DO NOT DO THIS FOR THE DATA SET IN CLASS.** The first part of the output is below:

The SAS System

Obs	variety	length
1	bihai	47.12
2	bihai	46.75
3	bihai	46.81
4	bihai	47.12
5	bihai	46.67
6	bihai	47.43
7	bihai	46.44

I have highlighted the first five data points.

You will see some error messages because there are some missing data. We will be removing those values later.

Please name your data set and variables in context. In this case, we are interested in *Heliconia* flowers, so I named the data set helicon. The variables in the text file are Variety and Length so those are what I use in the input statement. Since capitalization is important for variable names, I often only use small letters. If you want to put a space in the name, use an underscore, '_' Note that there is a maximum length in SAS for both data set names and variable names so be careful.

If you want to copy tables (or parts of tables) from the SAS output, I suggest that you use the Snipping Tool. You can also use that tool to highlight your answer. This is the procedure that I used in the above table.

For large data sets, Do **NOT** print out the whole data set with all of the variables, only print out parts of it. The following code will print out the first 10 data points for variable Length only.

```
proc print data=helicon (obs = 10);  
  * the obs keyword has to be in parentheses;  
  var length;  
run;
```

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

Obs	length
1	47.12
2	46.75
3	46.81
4	47.12
5	46.67
6	47.43
7	46.44
8	46.64
9	48.07
10	48.34

If you want to print out more than one variable, separate them by spaces. The variables will be printed out in the order specified. For example, if I wanted to switch the order that the variables were printed out, I would use the code:

```
proc print data=helicon (obs = 10);  
  var length variety;  
run ;
```

Obs	length	variety
1	47.12	bihai
2	46.75	bihai
3	46.81	bihai
4	47.12	bihai
5	46.67	bihai
6	47.43	bihai
7	46.44	bihai
8	46.64	bihai
9	48.07	bihai
10	48.34	bihai

Lastly, in large data sets, it can be important to only print specific rows (and columns). The following code prints rows 2, 20, and 50.

```
** Create dataset printme with only desired rows;  
data printme;  
  set helicon;  
  if _n_ in (2, 20, 50);  
run;  
** Print it;  
proc print data = printme;  
  var length variety; run;
```

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

Obs	length	variety
1	46.75	bihai
2	43.09	red
3	36.66	yellow

The Obs column does not correspond to the columns in helicon, but to the new data set "printme".

Cleaning and Saving Data Sets

Since many datasets have missing values, it is important to process them before beginning the analysis. For this class, we will simply remove those rows. The following command will remove rows that are not complete and then view the cleaned data set. When you are displaying, be sure that you include the correct dataset name. Remember to **NEVER** print out large data files!

```
data helicon_cleaned;
  set helicon;
  if nmiss(of _NUMERIC_)=0 AND variety^= "NA";
/* only outputs the data if the numeric data is numeric, not text, and
variety is not NA) */
run;

proc print data=helicon_cleaned (obs = 10); run;
```

In the above code, the new data set is called 'helicon_cleaned' which is based on the old data set called 'helicon'.

Once you have cleaned the data set, you will want to save it so that you don't have to clean the data each time you use the data set. The procedure to save the new data set is as follows:

File → Export Data → In the 'Member', select the appropriate data set, in this case, it would be helicon_cleaned, be sure to select "Write variable labels as column names" (Fig. 1) → Next → Select the Tab Delimited Files (*.txt) (Fig. 2) → Next → Where do you want to save the file? → Browse - Browse on your computer for the appropriate location (I would suggest your W drive). As I do not have a W drive on my office computer, I am choosing another location (Fig. 3) → Finish

Use the following screen shots as guides

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

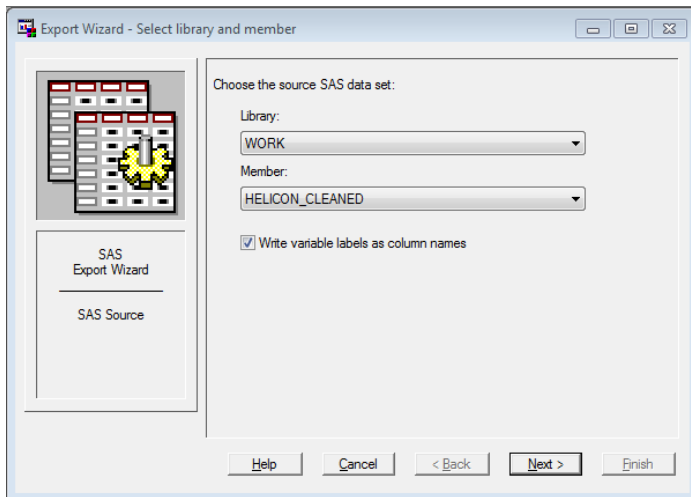


Fig. 1

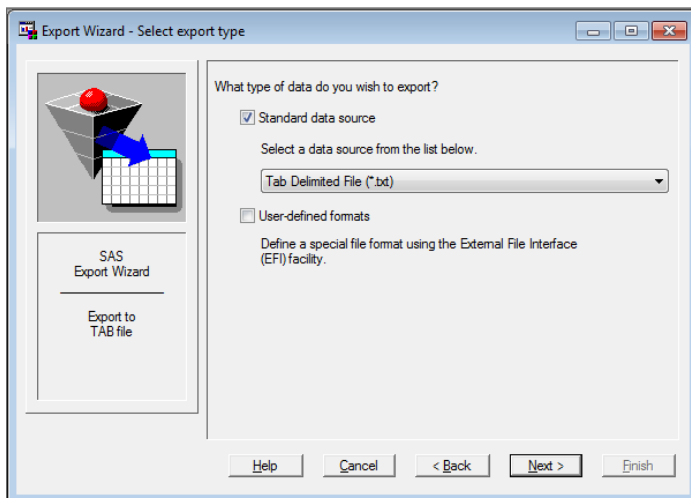


Fig. 2

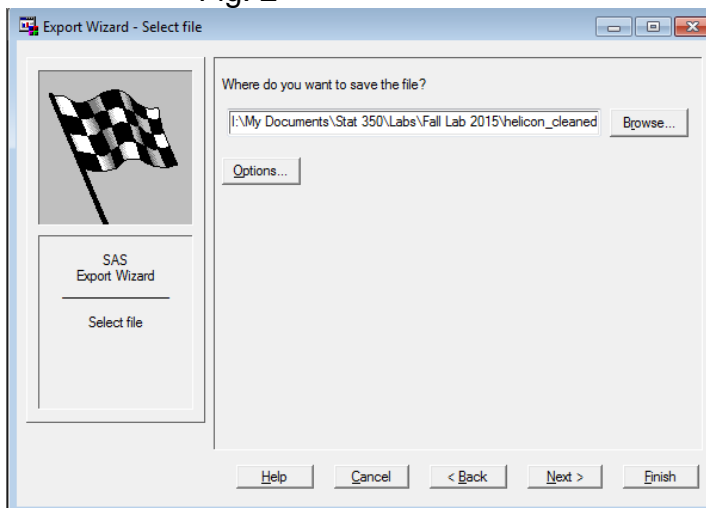


Fig. 3

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

Manipulating Data

For readability, you might want to change the shortened name or abbreviation to the full version. Remember to **NEVER** print out large data files! This is done by the following code:

```
data helicon_new;  
  length variety $ 15;  
  set helicon_cleaned;  
  if variety = 'red' then variety = 'Carabaea_Red';  
  if variety = 'yellow' then variety = 'Carabaea_Yellow';  
run;  
proc print data=helicon_new (obs = 43); run;
```

Some of the output is shown below:

36	Carabaea_Red	38.23
37	Carabaea_Red	38.87
38	Carabaea_Red	37.78
39	Carabaea_Red	38.01
40	Carabaea_Yellow	36.78
41	Carabaea_Yellow	37.02
42	Carabaea_Yellow	36.52
43	Carabaea_Yellow	36.11

If the length statement is not used, then the name would be truncated. *Always use a length statement if you are replacing an abbreviation with a full name or increasing the length of a categorical variable.* This statement needs to be placed right after the data statement.

In addition, you might want to create a new variable based on mathematical operations from old variable(s). You can use the sample code below to convert the lengths of the beaks from millimeters to inches. Remember to **NEVER** print out large data files! The conversion factor is 1/25.4.

```
data helicon_new;  
  set helicon_new;  
  length_inches = length/25.4;  
run;  
proc print data=helicon_new (obs = 10); run;
```

SAS Tutorial for Lab 1

Author: Leonore Findsen, Cheng Li, Min Ren

You will see a new column in the data set called length_inches:

Obs	variety	length	length_inches
1	bihai	47.12	1.85512
2	bihai	46.75	1.84055
3	bihai	46.81	1.84291
4	bihai	47.12	1.85512
5	bihai	46.67	1.83740
6	bihai	47.43	1.86732
7	bihai	46.44	1.82835

Note that it is not necessary to change the name of the new data set. However, I strongly recommend that you do change it since (1) in case there's a mistake, you won't overwrite the original data, and (2) you can compare the new with the original. In addition, you should never re-use variable names. That is, your modified variable should always have a distinct name from any other variable in your data set.