STAT 2005 – Programming Languages for Statistics

Tutorial 7 SAS, ggplot2

2020

LIU Ran

Department of Statistics, The Chinese University of Hong Kong

1 Installation

- Academic Suite Edition for Windows platform: Make an appointment with Mr.Julian through email: julianwong@cuhk.edu.hk and he will help you. Details can be found in the announcement on Blackboard.
- SAS University Edition: Go this link to get the SAS University Edition.

2 Introduction

2.1 Steps

Compared with R, SAS has very clear structure and process to analysis data.

1. `DATA` step: Convert raw data into a SAS dataset and modify it by writing functions.
2. `PROC` step: Perform analysis on SAS datasets, produce formated reports and so on.

Remark 2.1. We will basically consider these two steps in this course. However, there are lots of many other steps you can use in SAS.

2.2 Basic SAS Syntax

- SAS is case **in**sensitive.

```
DATA abc;
daTa Abc;
Data aBC;
```

Remark 2.2. SAS remembers the case of the first occurrence of each variable name and uses that case when printing results.

- Each SAS command must be terminated by a semicolon(;).

```
data abc;
```

- One command can be written in more than one line.

```
data
abc;
```

- Two or more commands can be written in the same line.

```
data abc; data xyz;
```

- The command can start at any column.

```
        data abc;
```

- Enter `RUN;` statement at the end of a unit of work, i.e., after each `DATA` step and `PROC` step.

```
data new;
set existing;
run;
proc print data=new;
run;
```

2.3 Comment in SAS

1. `*text of comment;`
2. `/*text of comment*/`

```
data d1; *I am a comment.  The SAS complier will ignore me.;
data d2; /*I am also a comment.  The SAS complier will also ignore me.*/
```

2.4 Variable Name

- Length $\leq$ 32 characters.
- Must start with an alphabetic character (`a - z`, `A - Z`) or an underscore (`_`).
- Contain only alphabetic characters (`a - z`, `A - Z`), underscores (`_`), and numbers (`0 - 9`).
- Avoid using keywords in SAS as variable names, such as `DATA`, `PROC`, `_ALL_`, `_NULL_`, `_N_`.

Remark 2.3. The beginning of a variable name cannot be numbers.

2.5 Data Types

- Numeric
  - Can be either positive or negative, and can contain plus sign(+), minus sign(-), decimal point(.), or `e`(scientific notation).
  - A missing numeric value is denoted as a single dot (`.`).
- Character
  - Can contain numbers, letters, or special characters such as $ # !.
  - A missing character value is denoted as a blank space ( ).
  - A $ is added after a variable name to mark this variable as character variable.

3 Exercises

1. Verify whether the following variable names are valid:
   (a) `oneTwoThree`
   (b) `123abc`
   (c) `abc123`
   (d) `_123abc`
   (e) `single-element`
   (f) `array.element`
   (g) `document_subarray`
   (h) `The_Chinese_University_of_Hong_Kong`

(a) Valid (b) Invalid (c) Valid (d) Valid (e) Invalid (f) Invalid (g) Valid (h) Invalid

2. Which of the following SAS statements are incorrect?

   (a) `DATA studentsID $; /* input character data */`

   (b) `DATA My.lib.cars_data;`

   (c) `*** COMMENT ***;`

   (d) `INFILE C:\DATA.TXT`

   (e) `LIBNAME A.B 'C:\";`

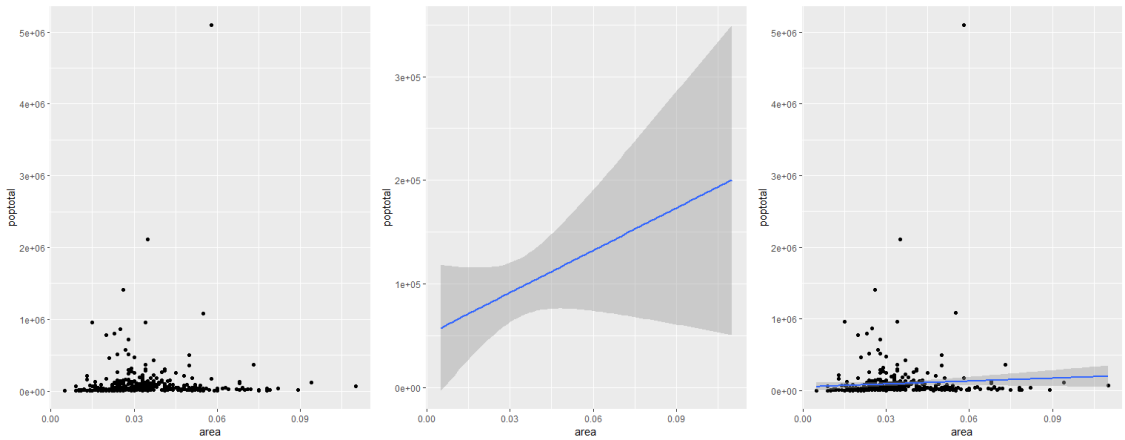
(a) incorrect (b) incorrect (c) correct (d) incorrect (e) incorrect

4 ggplot2(optional)

ggplot2 is based on The Grammar of Graphics. You provide the data, tell ggplot2 how to map variables to aesthetics(coordinates), what graphical commands to use, and it takes care of the details.

The main difference is that, unlike base graphics, ggplot works with dataframes and not individual vectors. All the data needed to make the plot is typically be contained within the dataframe supplied to the ggplot() itself or can be supplied to respective geoms.

```r
library(ggplot2)
data("midwest", package = "ggplot2")  # load the built-in data in ggplot2 package

# input the dataframe, and can also define the mapping between variables and aesthetics globally
# the following commands will use this data and mapping
p<-ggplot(midwest, aes(x=area, y=poptotal))

# simple scatterplot
p1 <- p + geom_point()

# add a smoothed line
p2 <- p + geom_smooth(method="lm")

# scatterplot and a smoothed line
p3 <- p + geom_point() + geom_smooth(method="lm")

# combine the three plots
# cannot use par(mfrow = ) for ggplot
library(cowplot)
plot_grid(p1, p2, p3, ncol = 3)

# just use the function of cowplot instead of loading the whole package
# cowplot::plot_grid(p1, p2, p3, ncol = 3)
```



Remark 4.1. When we load the whole package, some new methods may be loaded into the old generic function.

Remark 4.2. Add data and define the mapping; go through different pipelines to create graphs. (similiar to the `%>%`)

Remark 4.3. More commands: cheatsheets.