Creativity1.sas: Graphical display of data - Explanation of code

Lines from creativity1.sas are repeated here so you know what each explanation refers to.

Because SAS generally produces lots of results, including many you don't want/need, these explanations include both what code to write and where to find desired results.

Do not copy and paste code from a pdf file. That works 98% of the time but not 100% of the time. The problem are those characters or character combinations that pdf files make "look pretty". Quotes are one example. They do not copy and paste correctly.

The .sas file has the run; commands after each proc to make each proc stand-alone.

Goals of code:

- Display the creativity data (dotplot, histogram, boxplot)
  - As one group or for each treatment
- Saving results to copy to Word

Confusing things:
Copying code from the pdf file. Copy it from the .sas file or load the entire file into the program editor window.
Forgetting to submit/run after correcting errors in the code.
Forgetting to submit the run; command.
Forgetting to submit the ods rtf close; to finish saving results in a Word rtf file.

Comments: `/* anything */`
Any text between `/*` and `*/` is a comment and ignored. This can span many lines. The SAS program editor window will show the comment in green. If code you want to run is in green, it probably won't be run. Figure out why your comment didn't end where it was supposed to.

Reading a .csv file `proc import`
Some proc are set up so that follow on statements (lines with ; at their end) specify what needs to be done. proc import wants its information in the proc statement. The necessary pieces are where the data comes from and the name of the SAS data set that will store the data.
`datafile='name in quotes'` tells where the data comes from. The file extension (e.g. `.csv`) is necessary because that's how SAS knows how the information is stored. The file name and extension are a string of characters inside quotes. Those may be double quotes (`"`) or single quotes (`'`). They must match, i.e. if the string starts with a double quote, it **must** end with a double quote.
`out=name` specifies the name of the SAS data set. SAS will store this in the WORK library, so the full name will be WORK.NAME. The data set name can be the same as the file name or different. Names of data sets only become important when you save more than one.

`replace` is an option I recommend. Without it, SAS is very conservative. It checks whether the data set you specify already exists. If it does, then proc import stops; the old data set is not overwritten. This is a problem when you have corrected data and really do want to overwrite the old version. replace gives SAS permission to overwrite the data set.

I generally check the line in the log that says "The data set WORK.CREATIVITY has 47 observations and 2 variables". I check that SAS read the number of observations I expected and produced the number of variables I expected.

Saving a copy of results in a rtf file: `ods rtf file='creativity.rtf';`
This line is optional. It tells SAS to save a copy of the output in the RTF format file called creativity.rtf. This file is easily readable by WORD, and makes it easy to copy selected results to another document.

If you want to copy a figure from SAS results, that is most easily done by creating a WORD copy of the results (the ods rtf file = ; command) and copying from the WORD document.

When you are done, you need to submit `ods rtf close;` (at the end of the code file), but can be anywhere if you want to turn on and off to selectively save results. When you close the output, SAS will (should) start Word so you can see the file. To copy parts of the output to another file, just select the part you want, copy and paste into your file.

Saving a copy of results to a pdf file:
This would be `ods pdf file='creativity.pdf';` with a matching `ods pdf close;` at the end of the code (or the end of where you get results you want to save. Just like saving to an rtf file but with pdf instead.

Although Acrobat (but not Acrobat Reader, the last time I checked) can extract figures from a pdf document, I find it easier to work from a WORD document.

graphs in SAS:
SAS graphics are better then they were 10 years ago, but I (and many others) use other programs for graphics. I show how you can use SAS to draw homework quality graphs. If you add options, you can get publication quality graphs.

There are three different sorts of SAS graphics: base plots, which date from the days of lineprinters (really coarse graphs), SAS/GRAPHICS functions and SAS/StatisticalGraphics (SG functions). The SG functions are the first to be both easy to produce and result in reasonable plots. I only demonstrate SG graphics.

There three major procs to draw SG plots: proc sgscatter, proc sgpanel, and proc sgplot. These provide different capabilities, especially for drawing multiple plots in one figure. Today, we focus on proc sgplot, which produces a variety of different sorts of plots.

The basic structure of a proc sgplot; is to start the procedure (proc sgplot;) then request one or

more displays. If you request multiple displays, they are overlain on top of each other to produce one plot. The multiple displays can be of the same information or different information. I only illustrate one application of multiple plotting requests. The explanations are divided by the type of plots and corresponding plot request.

Most additional options apply to any type of plot. We will use group = regularly to separate the data by the levels of the variable specified after group = .

Options in SAS are specified after a / and before the ; that ends the statement. If you use multiple options, you only need one /.

You can add a title to any SAS proc that produces output by adding a title statement. This is a second statement, not an option to another statement. It can go anywhere between the proc statement and the terminating run; statement. The title goes in quotes, either single or double, so long as they match. That means if you want the title to be Angela's data, you need to use double quotes so the apostrophe is part of the title, not the end of the title. This will do that: `title ''Angela's data'';` The title is "sticky". Once set, it applies to the proc step containing the title (must go before the run; that ends that proc step) and all subsequent proc steps, until changed. If you want to reset the title to the default of nothing, write `title '';` in the **next** proc step.

Histograms, for one group: `proc sgplot; histogram score; run;` This will produce a histogram of all values of the score variable, i.e., combining data from the two treatments.

Histograms with multiple groups. Add the group = option to separate data by groups. For the histogram, SAS puts the two histograms on top of each other. Adding a transparency = option stops one group from "wiping out" the other.

```
proc sgplot;
  histogram score / group = treatment transparency=0.5;
  title 'Histogram of scores for the two treatments';
  run;
```

Dot plot, for multiple groups: `proc sgplot; scatter x=treatment y=score / jitter; run;` The scatter statement draws scatter plots of the data. x= is followed by the name of the variable to plot on the X axis; y= is followed by the name of the variable to plot on the Y axis. `scatter x=treatment y=score` will plot the data for each treatment in a column, i.e., along the Y axis. If you wrote `scatter x=score y=treatment`, you will get a horizontal dot plot, in which the values are plotted along the X axis. The X variable can be continuous (will see later) or categorical (e.g. the treatment name). When one of the variables is categorical, you get dot plots.

By default two observations with the same score in the same treatment will be overplotted. You will only see one dot. That's true even if there are 100 observations with the same score and treatment. The /jitter option separates the points so you see each value. SAS has a very intelligent jitter. The

results look better than those from jitter options in many other software programs.

Dot plots of one group of data
We have to trick scatter into drawing only one column of observations. We do this by creating a "grouping" variable where all observations have the same value. The scatter plot looks nicer if the grouping variable is categorical. I use the value "all" for the grouping value. You can choose any other word(s). I suggest not using a number so SAS knows the variable is categorical.

There are two ways to add a new variable to a SAS data set. Good data management practice is to make changes like this in code, not by modifying the data set. SAS allows data set manipulations only in a data step. We used proc import to read the .csv file. We can't add the group variable in that proc. We need to write a follow on data step that reads observations from an already existing data set and adds the variable we want. That looks like this:

```
data creativity2;
  set creativity;
  group = 'all';
  run;
```

Here's what's going on, step-by-step:
`data creativity2;` Create a new SAS data set called creativity2
`set creativity;` by reading observations from the creativity data set
This replaces input ; that is used to read observations from a text file
`group = 'all';` create the variable called group with the value of "all". "all" must be in quotes because it is a character string, not variable name.
`run;` Then run the whole proc.
Note: the group = statement comes after the set statement. This is crucial when the new variable is a transformation of variables in the pre-existing data set.

If this isn't working and you can't figure out why, an alternative is to modify the data set in Excel. Add a column (I will call it group) with "all" in each data row.

The hard part is creating this data set. After that, you'll recognize the code. You use same sort of proc sgplot, only the X variable is the newly created variable (here, group with one value) not treatment (two values).

proc sgplot; dot ;
There is a dot statement in proc sgplot. It is meant for categorical values and does not draw what most folks call a dot plot. I mention this only to avoid confusion if you search "sgplot dot plot".

Boxplots for one group of observations:
When you only have one group of observations (so no grouping variable), the easiest way to get a boxplot is use proc sgplot. The vbox command creates vertical box plots. vbox is followed by the name of the variable to plot. Values of that variable are plotted vertically, i.e., along the y axis of

the plot. If you want a horizontal box plot, use the hbox statement.

As is, you get a single box plot for all observations. When you add /group = , you get side-by-side box plots for each group. So:
`proc sgplot; vbox score; run;` will plot all observations in a single box plot.
`proc sgplot; vbox score / group=treatment; run;` will give separate box plots for each group

Interpreting SAS's box plots:
SAS displays a "5 + 1" box plot, possibly augmented by individual extreme values. The central box shows the 25'th to the 75'th percentiles. The horizontal line in the middle of the box is the median (50'th percentile). The whiskers usually run up to the maximum value and down to the minimum value. Extreme values (you can find out what SAS defines to be extreme by looking in the documentation) are plotted individually above the ends of the whiskers. That's the 5 number summary usually shown in a box plot, with individual extreme values. The diamond, usually inside the box, shows the mean of the data.

Closing the rtf output: `ods rtf close;`
This tells SAS to stop saving results in the RTF file. SAS will then open up WORD so you can look at the results. You can cut and paste figures from this document or save the entire thing as a .docx file (use WORD save as, not save).

N.B.: If you forget to close the rtf destination, you won't get any output file.