Cheatsheet: SAS ODS

Luke Johnston

2015-04-12

The Output Delivery System (ODS) in SAS allows results from a `proc` to be output into a data format so that if you wanted you could customize the results in a datastep and eventually output it into a file. This is an incredibly useful tool as you can make your analysis easier, make it easier to input the results into a manuscript or report, and make your research more reproducible and transparent. Below are some commands that will help you figure out what to output, how to customize the results, and how to save them to a file.

SAS ODS commands: Some useful or common ones

`ods trace on; ... ods trace off;`

When you run a chunk of code with the `ods trace on;` at the start, a `proc` in the middle, and `ods trace off;` at the end, the ODS objects within the `proc` are printed in the log file/window. You can then use that object to output just the object. The name of the ODS object changes depending on what statements are used in the `proc` command. For instance, in `proc anova` if the `lsmeans` statement is provided, `ods trace on;` will show an object that is named `LSMeans`.

Example code:

```
ods trace on;
proc means data=sashelp.fish;
run;
ods trace off;

/** Comment: The output from this code is: */
Output Added:
-------------
Name:       Summary
Label:      Summary statistics
Template:   base.summary
Path:       Means.Summary
-------------
```

ods output <ods-object-mame> = <custom-output-name>;

After extracting the name of the ODS object using `ods trace on;`, you can then use the name of the object to output the results specific to the object. For instance, the ODS object name for `proc means` was `Summary`. In order to output the `Summary` object, we would replace it with `<ods-object-name>` and make our own name up for the output dataset (eg: `meansDS`) in the `<custom-output-name>` space.

Example code:

```
proc means data=sashelp.fish;
    ods output Summary = meansDS;
run;

proc print data=meansDS;
run;
```

ods listing close; ... ods listing;

Sometimes when you are outputing an ODS object you also don't want SAS to output the normal results. In this case, you can prevent SAS from sending output to the 'listing' area, either as the `.lst` file or the output window on the SAS editor. That way, you can print only the results that *you* want to see, rather than what *SAS* wants you to see.

Example code:

```
/** Comment: Suppress printing of output */
ods listing close;
proc means data=sashelp.fish;
    ods output Summary = meansDS;
run;
ods listing;

/** Print the specific output */
proc print data=meansDS;
run;
```

data <new>; set <old>;

This command should be known to everyone given that, together with all `proc` commands, is the foundation to all SAS commands. This command is included

here to reinforce that the `ods output` data is in fact data and can be customized/wrangled. All the commands and code that you use to manage your dataset can also be used to manage your results data (eg. `if .. then ..;`, creating new variables, etc). So for instance, if you only want the probability from an ANOVA test, you can drop all other variables and just output the p-value.

Example code:

```
ods listing close;
proc glm data=sashelp.fish;
    class Species;
    model Height = Species / ss3;
    ods output ModelANOVA = modelFish;
run;
ods listing;

data modelFish;
    set modelFish;
    keep Dependent Source ProbF;
run;

proc print data=modelFish;
run;
```

||

This is the concatenate command. It basically allows you to combine multiple variables and characters together. This is especially useful for modifying ODS output objects so that they can be made into, for example, a table for a report. For instance, if you wanted to make a variable that shows the mean and standard deviation in the form of "mean (SD)", you can use the `||` command to achieve this.

Example code:

```
proc means data=sashelp.fish stackods;
    ods output Summary = meansDS;
run;

/** Comment: Edit the output results */
data meansDS (drop=Mean StdDev);
    set meansDS;
    /** In this case, you take the variable 'Mean',
```

```
        combine it using || with a space and a bracket ' ('
        add the variable StdDev, and lastly add the closing
        bracket */
    rawMeanSD = Mean||' ('||
        StdDev||')';
    /** If you print the rawMeanSD, there are lots of digits,
        so you can round it off using `round()` */
    roundMeanSD = round(Mean, 0.01)||' ('||
        round(StdDev, 0.01)||')';
    /** ... However, after rounding, there are extra spaces
        in the print out.  So you can use `strip()` to remove
        extra white space. */
    striproundMeanSD = round(Mean, 0.01)||' ('||
        strip(round(StdDev, 0.01))||')';
run;

proc print data=meansDS;
run;
```

`proc export ...;`

To make the most of customizing the output of a `proc` into a format that resembles a table, it's probably a good idea to output the results dataset into a file. You can use the `proc export` command to save the dataset into a file of your choice. There are in general four parts to the `proc export` command:

- `data=` is the output dataset.
- `dbms =` is the output format, eg 'XLS' or 'CSV'. Given the 'csv' format is simply plain text, I would recommend this option.
- `outfile =` is the name and location of the new output file, eg. 'means.csv'.
- `replace` tells SAS to overwrite any old file with the same name as the `outfile` name.

Example code:

```
/** Continuing with the dataset from the previous command */
proc export data=meansDS
    dbms = csv
    outfile = 'means.csv'
    replace;
run;
```