Producing Reports

Dr. Lendie Follett

2022-04-05

Titles and Labels

Adding titles

Titles can add context to your report (SAS-generated output) and help your audience follow along. The following syntax adds an n^{th} level title.

```
TITLE<n> "title-text";
```

- Replace `TITLE<n>` with `TITLE`, `TITLE2`, `TITLE3`, etc...
- Replace `title-text` with some descriptive title. For example,

```
title1 "Class Report";
title2 "All Students";

proc print data=s50dat.class_birthdate
run;
```

Class Report
All Students

Obs	Name	Sex	Age	Height	Weight	Birthdate
1	Alfred	M	14	69.0	112.5	16370
2	Alice	F	13	56.5	84.0	16756
3	Barbara	F	13	65.3	98.0	16451
4	Carol	F	14	62.8	102.5	16256
5	Henry	M	14	63.5	102.5	16406
6	James	M	12	57.3	83.0	16967
7	Jane	F	12	59.8	84.5	16873
8	Janet	F	15	62.5	112.5	15797
9	Jeffrey	M	13	62.5	84.0	16552
10	John	M	12	59.0	99.5	17036
11	Joyce	F	11	51.3	50.5	17169
12	Judy	F	14	64.3	90.0	16410
13	Louise	F	12	56.3	77.0	17021
14	Mary	F	15	66.5	112.0	15790
15	Philip	M	16	72.0	150.0	15665
16	Robert	M	12	64.8	128.0	16958
17	Ronald	M	15	67.0	133.0	15992
18	Thomas	M	11	57.5	85.0	17243
19	William	M	15	66.5	112.0	16087

Clearing Titles

- `Title` is a global statement; it will remain active for your whole SAS session.
- To clear all titles:

```
title; /*clear titles*/
```

- It's good practice to clear all titles at the beginning and end of each program.

Macro Variables in Titles and Footnotes

- Sometimes, it's useful to use macro variables (recall SAS topic 3) in titles.
- Allows you to dynamically change all titles automatically, rather than manually
- For example

```
%let age=13;

title1 "Class Report";
title2 "Age=&age";
footnote1 "School Use Only";

proc print data=s40dat.class_birthdate
    where age=&age;
run;

title;
```

Class Report
Age=13

Obs	Name	Sex	Age	Height	Weight	Birthdate
2	Alice	F	13	56.5	84	16756
3	Barbara	F	13	65.3	98	16451
9	Jeffrey	M	13	62.5	84	16552

Adding temporary labels

- Just like titles added helpful context, labels can too.

```
LABEL col-name="label-text";
```

- Note: a `LABEL` statement will occur within a `proc` step; it is *not* a global statement.
- Replace `col-name` with the name of the column you wish to provide a label for the proceure.
- Replace `label-text` with that label.

For example,

```
proc means data=sashelp.cars;
    where type="Sedan";
    var MSRP MPG_Highway;
    label MSRP="Manufacturer Suggested
          MPG_Highway="Highway Miles p
    /*the above labels only apply to
    this proc output*/
run;
```

Variable	Label	N	Mean	Std Dev	Minimum	Maximum
MSRP	Manufacturer Suggested Retail Price	262	29773.62	15584.59	10280.00	128420.00
MPG_Highway	Highway Miles per Gallon	262	28.6297710	4.4674591	17.0000000	46.0000000

Applying permanent labels

- The previous slide demonstrated how to add *temporary* labels
- If you put the same label statement in a `DATA` step, those labels are permanent

```
/*add a label statement to data statement*/
data cars_update;
    set sashelp.cars;
    keep Make Model Type MSRP AvgMPG;
    AvgMPG=mean(MPG_Highway, MPG_City);
    label MSRP="Manufacturer Suggested Retail Price"
          AvgMPG="Average Miles per Gallon";
run;

/*see permanent attributes*/
proc contents data=cars_update;
run;

/*Now all procs (except for proc print) will use the above labels.
Proc print requires a label argument. i.e.,

proc print data = cars_update label;
run;*/
```


Advanced Frequency Reports

Proc freq: so many options

- You remember `proc freq` as an easy way to get one-way frequency tables.
- However, there are several options to consider to enhance your report!

```
< ODS GRAPHICS ON; >
PROC FREQ DATA=input-table < options >;
        TABLES  col-name(s) < / options >;
RUN;
```

- `PROC FREQ` statement options:

  - ORDER=FREQ, FORMATTED, DATA
  - NLEVELS

- `TABLES` statement options:

  - NOCUM
  - NOPERCENT
  - PLOTS=FREQPLOT (must turn on ODS Graphics)
  - OUT=output-table

- Let's go to support.sas.com/documentation to figure out what these things do.

Proc freq: two-way frequency tables

- We know how to get a frequency table of one variable.
- Often, we want to know how two categorical variables are distributed *jointly*.
- In that case, we need a two-way frequency table!
- Adding an asterisk between two variable names is all we need to do!

```
PROC FREQ DATA=input-table < options >;
        TABLES  col-name1*col-name2 < / options >;
RUN;
```

For example, suppose we were curious about what types of vehicles different countries tend to produce:

```
proc freq data = sashelp.cars;
tables Origin*Type;
label Origin = "Country of Origin"
      Type = "Type of Vehicle";
run;
```

The FREQ Procedure

Frequency
Percent
Row Pct
Col Pct

Table of Origin by Type

Origin(Country of Origin)	Type(Type of Vehicle)						
	Hybrid	SUV	Sedan	Sports	Truck	Wagon	Total
Asia	3 0.70 1.90 100.00	25 5.84 15.62 41.67	94 21.96 59.49 35.88	17 3.97 10.76 34.69	8 1.87 5.06 33.33	11 2.57 6.96 36.67	158 36.92
Europe	0 0.00 0.00 0.00	10 2.34 8.13 16.67	78 18.22 63.41 29.77	23 5.37 18.70 46.94	0 0.00 0.00 0.00	12 2.80 9.76 40.00	123 28.74
USA	0 0.00 0.00 0.00	25 5.84 17.01 41.67	90 21.03 61.22 34.35	9 2.10 6.12 18.37	16 3.74 10.88 66.67	7 1.64 4.76 23.33	147 34.35
Total	3 0.70	60 14.02	262 61.21	49 11.45	24 5.61	30 7.01	428 100.00

Case Study

Where do storms happen, and when? Are some basins hit harder in certain months?

Save this case study as t5_cs_reports.sas.

Advanced Numerical Summaries

Proc means

- We've used `proc means` before, when we explored data.
- We can extend our previous use to include various statistics and groupings within the data.

```
PROC MEANS DATA=input-table <stat-list> <options>;
          VAR col-name(s);
          CLASS col-name(s);
          WAYS n;
          OUTPUT OUT=output-table <statistic=col-name>;
RUN;
```

- The `var` statement is used to identify the *numeric column(s)* you want to summarize.
- The `class` statement allows you to list one or more column by which to group the data.
- If you're grouping by more than one variable, the `ways` statement can help control combinations.
- The `output` statement creates a new SAS data set of the summarised output.

Case Study

Continue on with t5_cs_reports.sas that we started earlier. How do wind patterns vary by time of year and location?

Visualization

Visualization

- Data visualization is the technique of communicating data via graphical representation
- THIS IS AN IMPORTANT SKILL
- There are many types of plots available to us

- How do we choose?
  - What is your research question?
  - Does your research question involve one (univariate) or multiple (multivariate) variables?
  - Does your research question involve discrete or continuous variables?

First, some definitions

Definition: A **discrete** variable has distinct categories and can be stored as either character or numeric.

Definition: A **continuous** variable is numeric and can theoretically take on any value in an interval on the real line.

For example, MSRP is a continuous variable while Origin is a discrete variable. Note, Cylinders - though numeric - could be thought of as discrete.

Make	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders
Acura	SUV	Asia	All	36945	33337	3.5	6
Acura	Sedan	Asia	Front	23820	21761	2.0	4
Acura	Sedan	Asia	Front	26990	24647	2.4	4
Acura	Sedan	Asia	Front	33195	30299	3.2	6
Acura	Sedan	Asia	Front	43755	39014	3.5	6

Univariate plots: Discrete

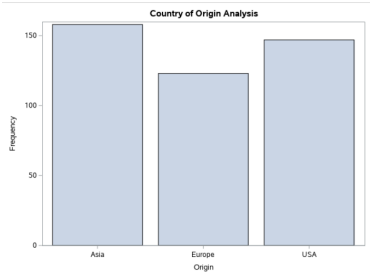
- bar chart: a chart displaying groups of data with bars having a length proportional to the value corresponding to each (discrete) group
- There are two types of bar charts
    1. height of the bar represents the number of cases per group
    2. height of bar represents some summary statistic within each group.
- We will focus on the first type in our univariate discussion
- `PROC SGPLOT` is the procedure we will use to do most of our plotting. In the case of bar charts,

```
proc sgplot data = input-data;
vbar discrete-col-name;
run;
```


Univariate plots: Discrete

A bar chart could be used to count the number of vehicles being made in each country of origin:

```
/*How many cars per country of origin?
title 'Country of Origin Analysis';
proc sgplot data = sashelp.cars;
vbar origin;
run;
```



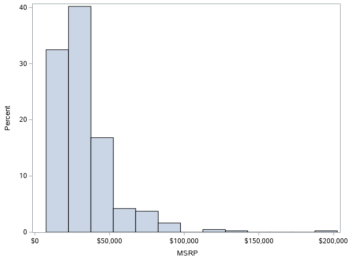
Univariate plot: Continuous

- histogram: a type of bar chart used to show the frequency, or relative frequency, with which data points take on continuous values. It shows the shape of the distribution of continuous data points.
- We can use similar syntax to what we used to make a bar chart:

```
proc sgplot data = input-data;
histogram continuous-col-name;
run;
```

For example,

```
/*What does the distribution of prices
proc sgplot data = sashelp.cars;
histogram MSRP;
run;
```



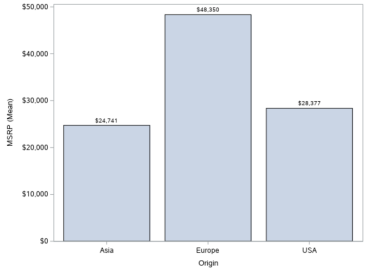
Multivariate plot: Continuous x discrete

- Goal: show a simple summary statistic of a numeric variable within each level of a categorical variable
- We can add options after `/`in the `vbar` statement.
- `response` and stat are some of the options SAS recognizes
- `datalabel` tells SAS to label the value of the bar height

```
/*Display mean for each species*/
proc sgplot data = input-data;
vbar discrete-col / response = numeric-col < stat = mean > < datalabel >;
run;
```


Multivariate plot: Continuous x discrete

```
/*What origins have the highest prices
title 'Price by origin';
proc sgplot data = sashelp.cars;
vbar origin / response = MSRP stat = m
run;
```



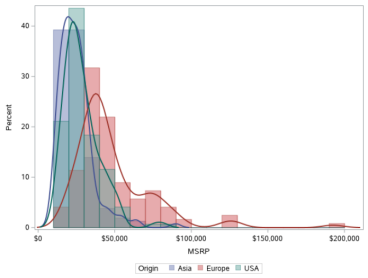
Multivariate plot: Continuous x discrete

- The bar chart in the previous slide doesn't show much detail in how MSRP varies with origin.
- It would be nice if we could show the same kind of variability that we saw in the histogram, but for each origin separately:

```
/*ITERATION 1: add group = option afte
proc sgplot data=sashelp.cars;
histogram MSRP / group=origin;
run;

/*ITERATION 2: hard to see overlapping
proc sgplot data=sashelp.cars;
histogram MSRP / group=origin transpar
run;

/*ITERATION 3: overlay densities with
proc sgplot data=sashelp.cars;
histogram MSRP / group=origin transpar
density MSRP / type=kernel group=origi
run;
```



Multivariate plot: Continuous x discrete

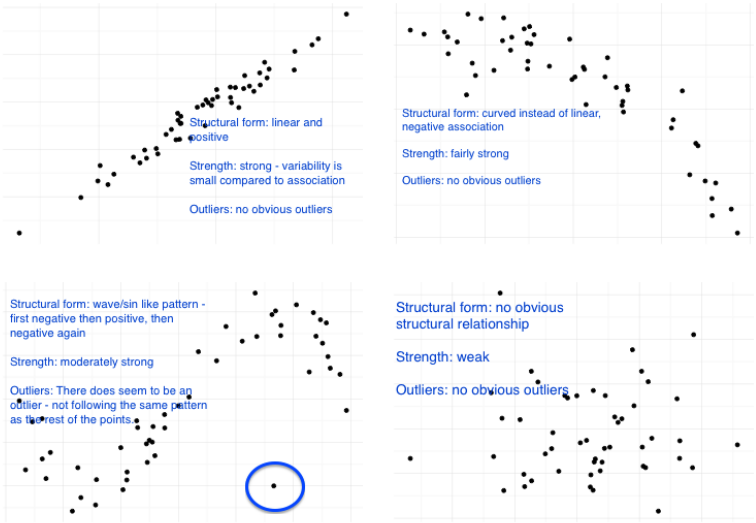
- An alternative to the overlaid histograms is the grouped boxplot!
- box plot (a.k.a box and whisker plot): a graphical representation of the quantiles and outliers of a continuous variable.
  - The Interquartile Range (IQR) is the quantity:(75th percentile minus the 25th percentile)
  - Outliers are typically considered as values exceeding 1.5*IQR above the 75th or below the 25th percentile
  - The whiskers in a box plot typically correspond to the max and min non-outlier values *Thus, a boxplot is a simple summary of the distribution of a continuous variable.

```
proc sgplot data = sashelp.cars;
vbox MSRP / category = origin ;
run;
/*what happens when "vbox" is changed
to "hbox"?*/
```

Multivariate plot: Continuous x continuous

- The most common way to display two continuous variables jointly is with a scatterplot.
- scatterplot: a plot displaying the joint distribution of two continuous variables by mapping the value of one variable to the x-axis and the value of the other variable to the y-axis.
- When interpreting a scatterplot (your own or otherwise), you want to make note of the following key features:
  - Structural form
  - Shape - linear, curved?
  - Direction - positive, negative?
  - Strength - how closely does the collection of points follow the structural form?
  - Outliers - points not seeming to follow the same structural form

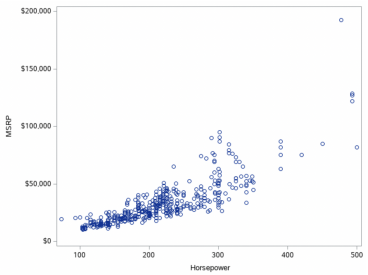
Multivariate plot: Continuous x continuous



Multivariate plot: Continuous x continuous

- A `scatter` statement can be added within a `proc sgplot` step to obtain a scatterplot
- Importantly, a scatterplot requires both an x dimension (`x=`) and a y dimension (`y=`)

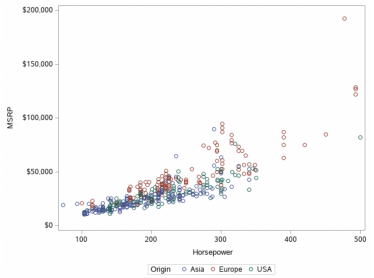
```
proc sgplot data = sashelp.cars;
scatter x = Horsepower y = MSRP ;
run;
```



Multivariate plot: Continuous x continuous

- Want to take into consideration an important discrete variable as well? We can again use color to our advantage!
- As before, the `group =` argument after the `/` allows us to select a discrete variable by which to color.

```
proc sgplot data = sashelp.cars;
scatter x = Horsepower y = MSRP / grou
run;
```



Case Study

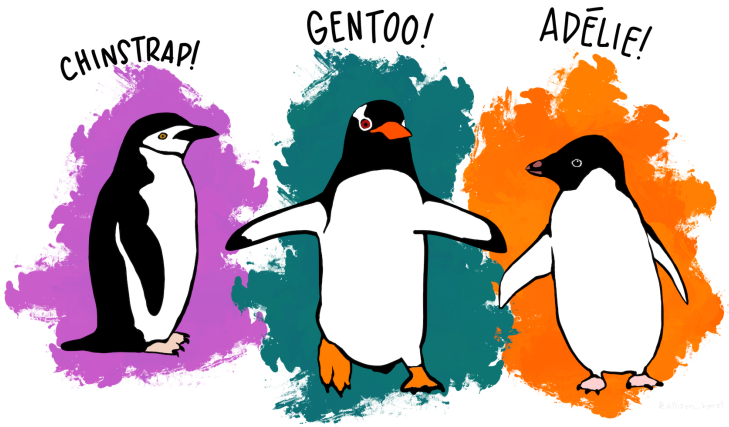
Create high quality visualizations to help an audience understand how storm intensity varies across both seasons and locations.

Continue with t5_cs_reports.sas.

Your Turn

1. Open another script and save it as t5_yt_pressure.sas.
2. You'll be working off of storm_summary, so create the appropriate libname statement.
3. Create a new SAS data set called work.storm_yearly. This data set should contain everything from the SAS data set s40dat.storm_final, plus a new column called `year` that contains only the year of the storm. Use the `year()` function to create the new variable.
4. Create a report that calculates the average of minimum pressure for each basin, at a yearly level. Make sure you save the means (for each year, basin combination) in a new, temporary SAS data set. Call the output data set work.pressure_yearly.
5. Create a bar chart illustrating how minimum pressure varies over location and time. Use `xaxis` and `yaxis` to set meaningful axis labels. Add a meaningful title.
6. Create a line chart illustrating the same thing. Use `xaxis` and `yaxis` to set meaningful axis labels. Add a meaningful title. Which of the two plots did a better job displaying the data?
7. Create the last plot again, but without displaying data from the "North Indian" basin. Recall topic 3 notes, `where` statements.

Case Study



1. Clean
2. Validate
3. Analyze