

Lab 3: Plots, correlation, and regression, part 1
STT 422: Summer, 2004
Vince Melfi

Statistical techniques (such as correlation analysis or least squares regression) for exploring relationships between two quantitative variables are time-consuming when done by hand, but very easy in SAS. Today we'll explore **proc gplot**, **proc corr**, and **proc reg**, which, respectively, provide plotting, correlation, and least squares regression capabilities.

Plots

We have met **proc gplot** already. Today we'll investigate a few of its capabilities.¹ The data for this section come from Exercise 2.11 in the text, and are available in U:\msu\course\stt\422\summer04\metab.dat. The variables are **gender**, lean body mass (weight of a person, in kilograms, leaving out all fat), and metabolic **rate** in calories burned in 24 hours. Our goal is to get a plot of the metabolic rate versus lean body mass, with separate symbols for males and females. We'll sneak up on this in a few steps. Here is the program. Enter and run it, look at the **Output Window** to see the data, and look at the **Graph Window** to see what the **proc gplot** commands produced. As usual, some explanation follows.

```
data metab;
  infile 'u:\msu\course\stt\422\summer04\metab.dat';
  input gender $ mass rate;

proc print data = metab;
  var gender mass rate;

proc gplot data = metab;
  plot rate*mass;

proc gplot data=metab;
  plot rate*mass;
  symbol value = circle;

proc gplot data=metab;
  plot rate*mass=gender;
  symbol value = circle;

run;
```

Explanation

1. There's a new twist to the **input** statement. Since **gender** is a character variable (it's coded as "M" and "F" rather than 1 and 2), we must put a \$ after its name.

¹For *much* more, look at the SAS help page for **proc gplot**.

2. The first call to **proc gplot** is standard. Note that the default plotting symbol is a + sign.
3. The second call to **proc gplot** changes the plotting symbol to a circle. There are many plotting symbols to choose from. You can learn more by consulting the SAS help for the **Symbol Statement**.
4. The third call to **proc gplot** changes the plot statement to **plot rate*mass=gender**. The addition of **=gender** tells SAS to use different symbols for males and females.

Note that SAS just changes the color of the plotting symbol for males and females. On the screen this is fine, but if we're printing to a non-color printer, it isn't so good. We can change this behavior by adding the statement **goptions colors=(none)** in the **proc gplot** statement. Do this now and see what symbols SAS uses for males and females.

Correlation

The SAS procedure **proc corr** computes the correlation coefficient r between two variables. For example, we can compute the correlation coefficient in the previous example quite simply as follows.

```
data metab;
  infile 'u:\msu\course\stt\422\summer04\metab.dat';
  input gender $ mass rate;

proc corr data = metab;

run;
```

Explanation

There's not much to be said about the SAS code, so we'll concentrate on the output. The first part of the output contains the same sort of output we'd expect from **proc means**. The second part contains a *correlation matrix*. The upper left entry is the correlation between **mass** and **mass**, which is obviously 1. The upper right entry contains the correlation between **mass** and **rate**, which should be 0.86474. This is also what's given in the lower left entry, while the lower right entry contains the correlation between **rate** and **rate**. (There are also strange entries such as $< .0001$ that appear below the correlations. Ignore these for now. They are related to testing whether the correlation is equal to zero.)

In this example, the correlation matrix may seem excessive. In the next example the correlation matrix is much more clearly useful. The dataset **U:\msu\course\stt\422\summer04\cheese.dat** contains data on thirty samples of cheddar cheese. Variables include **taste**, which is the score on a taste test of the cheese; **acetic**, which is a measure of acetic acid, etc. A more complete description is in the Data Appendix of the text.

```
data cheese;
  infile 'U:\msu\course\stt\422\summer04\cheese.dat';
```

```

input taste acetic hysulf lactic;

proc corr data = cheese;

proc corr data = cheese;
  var taste acetic;

run;

```

Note that the correlation matrix is more useful in this context, since we can easily read off a lot of different correlations. As an example, the first row of the table lists the correlations between `cheese` and all the other variables. To make sure you understand the output, find the correlation between `lactic` and `hysulf`. Note also that if we want to limit the correlation matrix to less than all the variables, we can do this via the `var` statement as in the second call to `proc corr` in the above program.

Least Squares Regression

The procedure `proc reg` computes the least squares line (and much more). We'll investigate this procedure using the `cheese` dataset from above. Here is the small SAS program to fit a line to the data $y=\text{taste}$ and $x=\text{hysulf}$.

```

data cheese;
  infile 'u:\msu\course\stt\422\summer04\cheese.dat';
  input taste acetic hysulf lactic;

proc reg data=cheese corr;
  model taste = hysulf;
  plot taste*hysulf;

run;

```

Explanation

1. First we read in the dataset as before.
2. We specify two options to the `proc reg` statement. First, `data=cheese` tells SAS which dataset the variables come from. (This is useful if we have more than one SAS dataset open.) Second, `corr` tells SAS that we'd like to see the correlation matrix for the variables in the model.
3. We specify the x and y variables via the `model` line. The generic format is `model y = x`. Since in our case the y variable is `taste` and the x variable is `hysulf`, we specify the model as `model taste = hysulf`.
4. The line `plot taste*hysulf` tells SAS to draw a scatter plot of the data along with the least squares line.

5. Look at the **Graph Window** first. Here you'll see the scatter plot of the data along with the least squares line. At the top is the equation of the least squares line, which in this case is

```
taste = -9.7868 + 5.7761 hysulf.
```

6. Now look at the **Output Window**. Here you'll find the correlation matrix, then a lot of output related to the least squares regression.

The Analysis of Variance table is printed first. From this, for example, we find that the F statistic for testing whether the slope parameter for **hysulf** is zero is equal to 37.29, and the p-value is less than 0.0001.

The table of **Parameter Estimates** gives the intercept -9.78684 and slope 5.77609 of the least squares line. (Which doesn't tell us anything more than we learned from the graph window!) In addition, the standard errors and t statistics are given, along with p-values. In this case, the t statistic for testing whether the slope parameter for **hysulf** is zero (6.11) is the square root of the F statistic. In addition, the t statistic for testing whether the intercept is zero is -1.64 , with a p-value of 0.1116.

Multiple regression

The following program fits a multiple regression model with **taste** as the response and **hysulf**, **lactic**, and **acetic** as predictors. Look through the output to be sure you can interpret it.

```
data cheese;
  infile 'u:\msu\course\stt\422\summer04\cheese.dat';
  input taste acetic hysulf lactic;

proc reg data=cheese;
  model taste = hysulf acetic lactic;

run;
```

The quit statement

If you look closely, you'll see that even though the `proc reg` procedure has done what you've asked, the Editor window still says "PROG REG Running." I don't know why this happens, but it doesn't seem to cause any problems. Adding a **quit** statement (with a semicolon) at the end of the program avoids this.

Exercise

The file `U:\msu\course\stt\422\summer04\anscombe.dat` contains eight variables. In order they are `x1 y1 x2 y2 x3 y3 x4 y4`. You can read these data into SAS and call the resulting dataset `anscombe` via

```
data anscombe;
  infile 'u:\msu\course\stt\422\summer04\anscombe.dat';
  input x1 y1 x2 y2 x3 y3 x4 y4;
```

For the following questions, only report numbers accurate to three significant digits. For example, if a standard deviation is computed to be 5.43256, report it as 5.43. If a correlation is reported to be 0.00023423, report it to be 0.000234.

1. Compute the means and standard deviations of the variables and report them here. Compare the means and standard deviations of the x variables. What do you notice? Compare the means and standard deviations of the y variables. What do you notice?
2. Compute the correlations between x_1 and y_1 , between x_2 and y_2 , between x_3 and y_3 and between x_4 and y_4 . Report them here. What do you notice?
3. Run separate regressions of y_1 on x_1 , y_2 on x_2 , y_3 on x_3 , and y_4 on x_4 . Report the equation of the least squares lines for each regression. What do you notice?
4. Based only on your answers to the previous questions, you might be tempted to conclude that
 - (a) There is no difference between x_1 and x_2 and x_3 and x_4 since they have the same means and standard deviations.
 - (b) There is no difference in the relationships between x_1 and y_1 , between x_2 and y_2 , between x_3 and y_3 , and between x_4 and y_4 , since the correlations and least squares lines are the same in all four cases.

Present convincing evidence to refute these “conclusions.”