Notes explaining code in powerSAS

The code has two major parts:

 1) Using SAS as a calculator to implement the examples in the week 2 notes

 2) Using proc power to get power/sample size for common designs

The week 2 notes have 3 examples using the fundamental power equation. All custom calculations must be done in SAS data steps. We use a SAS data step to calculate what we need and any intermediate quantities. A proc print can then print the results.

There are many different ways to implement the calculations and show the results. For example, in the first data step, you could input alpha instead of defining. And, if you know about data step put statements, you could also use that to write the results to the log file. I show you one way. You’re welcome to use another way so long as it’s the correct calculation.

1) How big a difference is needed to get 80% power when n = 20 per group and s = 5?

This question provides you 1-β ( = 0.8), s (=5), and n (=20).

 The type I error is not specified, so we’ll use α = 0.05 by default.

We need to find δ, the true difference.

The first data step implements this calculation. We read values for power n and s from a data set that is included in the data step code. The next two lines calculate the df and the se for a 2-sample difference of means. We then set a value for alpha. (Note: This could be included in the input statement). Finally, we compute delta. The calculation of delta uses the quantile() function twice: once to calculate the quantile for the alpha level and once for the power. The quantile() function is described in Tquantile.sas: read the comments in the file.

A proc print then prints out the computed delta, along with the input quantities.

2) What n is needed for 80% power to detect a difference of 2 when s = 14.8?

As explained in class, the best way to implement this is to use an iterative calculation. The data n; data step computes n when the df (also a function of n) is specified. You need to get values of desired power, the specified delta, and the sd. I start with n = 31, for which df = 60 in a two-sample problem. The code computes df and then the “new” n. The results are printed.

This value of n is then used in a second data n; data step. Identical code except for the value of n. If the n doesn’t change very much, you can stop after the second iteration. If not, copy the code to run a 3rd iteration.

Note: You need to copy the printed value of n into the next data step. If you’re familiar with SAS macros (Possibly covered in 479, you’ll know how to save n into a macro variable then grab that variable in the next data step).

3) What is the power given n, delta and s?

This computation is implement in essentially the same way as the first one. Tb is the quantile associated with the power. We need to calculate the probability associated with that quantile. That’s the cdf() function, explained by comments in the Tquantile.sas code.

Using proc power.

SAS proc power provides sample size and power computations for common designs. powerSAS.sas illustrates those that we focus on in this class.

You provide a statement that tells proc power the type of analysis. The ones for this class are:

 twosamplemeans: difference of means from two independent samples

 onesamplemeans: test of a single mean = 0, used for paired data

 onewayanova test = contrast: test of a linear contrast = 0

Each “code word” is followed by specification of all the necessary quantities. The value to be computed is set to the SAS missing value code ( . ). You have to use the correct names for each of the quantities.

The first proc power computes the sample size per group (npergroup) to provides 80% power for a two-sample T test when the population (true) difference in means = 0.5 and population sd = 0.9. The type I error rate (alpha) is set to the usual 0.05; this is the default so alpha = can be omitted. The power is printed to the results file; a proc print is not needed.

NOTE: All the specifications are part of a single SAS statement. The semi-colon ( ; ) that ends a SAS statement goes after the last specification. If you add additional semi-colons, SAS will complain that you haven’t given it enough information. There are no commas between specifications.

NOTE: My practice is to put each specification on its own line. That’s for clarity and clarity only.

That way you can easily check that you don’t have a semi-colon until the very end.

The second proc power specifies a range of sample sizes. The printed output includes power for each sample size. The plot statement (a new statement, so after the first ;) requests a plot of power against the X variable you specify.

The third proc power is for a paired T-test (one sample comparison). The only change is that the sample size is written as ntotal = .

The third proc power is for a linear contrast. Parts (stddev = , power = , and npergroup = ) are familiar. The new parts are specifying the contrast coefficients and specifying the “true” value of that contrast.

contrast = specifies the coefficients. They go inside ( ) without commas.

 There are as many values as there are groups in the ANOVA.

 If you have a factorial treatment structure, you can use contrasts to evaluate marginal means.

 Details on doing this will be part of the discussion of factorial designs.

groupmeans = specifies means for all groups. Blocks of values are separated by vertical bars.

 You need as many blocks as there are coefficients in the contrast = specification.

 SAS allows you to specify more than one value within each block.

My approach is to determine the value of the contrast that a researcher “cares about”. You specify group means that give that value for the contrast. The old-new proc power uses 0.5 as the “cares about” value for that contrast. You specify group means for which group 2 - group 3 = 0.5. Since the coefficient for group 1 = 0, you can specify any value there. Since all that matters is the difference (group 2 – group 3), there are many possible specifications of means for which the contrast = 0.5. Here are some groupmeans = that give identical results:

groupmeans = 10 | 2 | 2.5

groupmeans = 0 | 5 | 5.5

groupmeans = 0 | 7.5 | 7 (the test is 2-sided, so there is no difference between +0.5 and -0.5)

The fourth proc power is for a contrast between group 1 and the average of groups 2 and 3.

You can have more than 1 statement in a single proc power;. That means you could include both contrast tests in the same proc.

The remaining calls to proc power are for optional things:

Computing power for the overall F test – this depends on the values of all group means

Numerical demonstrations that show why I find it hard to use overall F test power.

 It depends on the configuration of all the means, not just the “I care about” difference