Some Power Examples

Consider the usual linear model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\mathbf{X}$ is nxr and $\boldsymbol{\beta}$ is rx1.

When H_0: $\mathbf{C}\boldsymbol{\beta}=\mathbf{h}$ is false ($\mathbf{C}$ is qxr with linearly independent rows) , the F statistic has a non-central F distribution with q and n-r degrees of freedom, and non-centrality parameter

$$\phi = \frac{(\mathbf{C}\beta - \mathbf{h})'(\mathbf{C}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{C}')^{-1}(\mathbf{C}\beta - \mathbf{h})}{\sigma^2}.$$

For a one-factor design, this may be written as

$$\phi = n\frac{\sum_{k=1}^{r} \frac{n_k}{n}(\mu_k - \mu_.)^2}{\sigma^2} = n\frac{\sum_{k=1}^{r} f_k(\mu_k - \mu_.)^2}{\sigma^2}.$$

Suppose we have four treatments, and that the four population treatment means are equally spaced, one-quarter of a standard deviation apart. We'd like to be able to detect the differences among treatment means with probability 0.80, using the conventional significance level of α=0.05. We'll use equal sample sizes. Let

Non-centrality parameter = Sample size x Effect size

Without loss of generality, we'll let the four population treatment means be 0, 0.25, 0.50 and 0.75. Using R as a calculator, and remembering that the `var` function divides by the number of observations minus one, we'll calculate the effect size as

```
> 3 * var(c(0,.25,.5,.75)) / 4
[1] 0.078125
```

The program fpow2.sas uses `put` statements to write on the log file.

```
****************  fpow2.sas *********************************************/
options linesize = 79 pagesize = 100  noovp formdlim='-';   /*             */
data fpower;             /* Replace alpha, q, r, effsize and wantpow below  */
     alpha = 0.05;       /* Signif. level for testing H0: C Beta = h        */
     q = 3;              /* Numerator df = # rows in C matrix               */
     r = 4;              /* There are r beta parameters                     */
     effsize = 0.078125; /* Effect size is ncp/n                            */
     wantpow = .80; /* Find n to yield this power                           */
     power = 0; n = r+2; oneminus = 1-alpha;  /* Initializing ...           */
/**************************************************************************/
     do until (power >= wantpow);
        n = n+1 ;
        ncp = n * effsize;
        df2 = n-r;
        power = 1-probf(finv(oneminus,q,df2),q,df2,ncp);
     end;
     put ' ';
     put ' *************************************************************';
     put '   With ' r ' beta parameters, testing H0 of ' q ' linear';
     put '   restrictions on the betas and an effect size of ' effsize ',';
     put '   A sample size of ' n 'is needed to have probability ' ;
     put '    ' wantpow ' of rejecting H0 at significance level alpha = ' alpha;
     put ' *************************************************************';
     put ' ';
     put ' ';
```

The log file includes

```
*************************************************************
   With 4  beta parameters, testing H0 of 3  linear.
   restrictions on the betas and an effect size of 0.078125 ,
   A sample size of 144 is needed to have probability
   0.8  of rejecting H0 at significance level alpha = 0.05
 *************************************************************
```

Now do the same thing with R

```
fpow2 <- function(r,q,effsize,wantpow=0.80,alpha=0.05)
#############################################################################
# Power for the general multiple regression model, testing H0: C Beta = h   #
#       r       is the number of beta parameters                            #
#       q       Number rows in the C matrix = numerator df                  #
#       effsize is ncp/n, a squared distance between C Beta and h           #
#       wantpow is the desired power, default = 0.80                        #
#       alpha   is the significance level, default = 0.05                   #
#############################################################################
    {
    pow <- 0 ; nn <- r+1 ; oneminus <- 1 - alpha
    while(pow < wantpow)
        {
        nn <- nn+1
        phi <- nn * effsize
        ddf <- nn-r
        pow <- 1 - pf(qf(oneminus,q,ddf),q,ddf,phi)
        }#End while
    fpow2 <- nn
    fpow2  # Returns needed n
    }      # End of function fpow2
```

One may paste this function definition into the R window. Then,

```
> fpow2(r=4,q=3,effsize=0.078125)
[1] 144
```