BIOS 662

Homework 7 Solution

November, 2018

Question 1

First recast this as a two sample binary problem as in the class notes. Because bladder cancer is rare, the proportion of non-cases in the population is very close to 1. So, to a very good approximation, one person of 50 in the general population using the particular medication is the same as the proportion of controls who use the medication. That is, $\pi_2 = \Pr(\text{exposed}|\text{case}) = 1/50 = 0.02$. Also, as OR $= 3$, we have

$$\pi_1 = \frac{\pi_2 \text{OR}}{1 + \pi_2(\text{OR} - 1)} = 0.05769.$$

The power to detect $\pi_1 = 0.05769$ versus $\pi_2 = 0.02$ at the $\alpha = 0.05$ level of significance is 0.28. This result can be obtained using the formula in the class notes or by the following SAS code:

```
proc power;
   twosamplefreq
     refp   = 0.02
     pdiff  = 0.03769
     ntotal = 200
     power  = .;
```

```
The POWER Procedure
Pearson Chi-square Test for Two Proportions

               Fixed Scenario Elements

Distribution                          Asymptotic normal
Method                              Normal approximation
Reference (Group 1) Proportion                     0.02
Proportion Difference                           0.03769
Total Sample Size                                   200
Number of Sides                                       2
Null Proportion Difference                            0
Alpha                                              0.05
Group 1 Weight                                        1
Group 2 Weight                                        1


Computed Power

Power
0.280
```

We would not recommend conducting this study because the power is very low, so the chance of a type II error is too high. There is a probability of $1 - 0.28 = 0.72$ that we will fail to detect an OR as large as 3. Instead we would recommend 412 cases and 412 controls to have 80% power, or 551 cases and 551 controls to have 90% power. (Or a study with multiple controls per case, but even that would need many more than 100 cases to have reasonable power.) These results can be obtained using the formula in the class notes, as implemented in R in the function "ss_fleiss" defined in the class notes:

```
> ss_fleiss(0.02,0.05769,0.05,0.8)
[1] 411.4046

> ss_fleiss(0.02,0.05769,0.05,0.9)
[1] 550.2548
```

or by using proc power as follows:

```
proc power;
 twosamplefreq
   refp   = 0.02
   pdiff  = 0.03769
   ntotal = .
   power  = 0.8 0.9;
```

```
The POWER Procedure
Pearson Chi-square Test for Two Proportions

               Fixed Scenario Elements

Distribution                         Asymptotic normal
Method                             Normal approximation
Reference (Group 1) Proportion                    0.02
Proportion Difference                          0.03769
Nominal Power                                      0.9
Number of Sides                                      2
Null Proportion Difference                           0
Alpha                                             0.05
Group 1 Weight                                       1
Group 2 Weight                                       1


        Computed N Total

         Nominal    Actual        N
Index      Power     Power    Total
    1        0.8     0.801      824
    2        0.9     0.900     1102
```

Question 2

(a) Using either R or SAS, a key issue for any power/sample size simulation is to work out how to obtain the relevant p-value from the specific function or proc. In the SAS version below the dataset produced by proc reg for the first simulated dataset is printed to check where the p-value is. See the comment in the code. Similarly, in the R code the estimated coefficients and related test statistics are printed for the first simulated dataset.

Note that here for each dataset of size n we generate n values for X and for ε and then obtain n values for Y using $y_i = \alpha + \beta x_i + \varepsilon_i$. In this case any value can be used for α without affecting the results.

Because we want the sample size to achieve a specified power (0.9) we try various values of n until we find one that gives the appropriate power. The appropriate sample size is about 417. (I ran the simulation *before* using the formula in part (b) and will admit to being surprised at how well the simulation agrees with the result in (b).)

For a large sample size and/or large number of simulated datasets, the way the data are generated in the SAS example in the notes can cause out-of-memory errors. So I have included two different approaches to doing the simulation in SAS. The first one is like the example in the notes, with all the datasets being generated first and then the test of the regression coefficient being run on them. In the second approach the datasets are generated one at a time, with the test of the regression coefficient being run before the next dataset is generated. With this approach just one dataset of n observations is kept in memory at any time. The "end=eof", the retain statement and the three lines beginning with "if eof then do;" are to pass an updated random number seed to the next iteration. See pages 20-21 of the "Random Number Generation" overheads.

R code for the simulation and the corresponding output:

```
alpha <- 0
beta <- 0.2

mysim <- function(seed0,n,nsims){
  set.seed(seed0)
  rejects <- 0
  for (ii in 1:nsims){
    e <- rnorm(n,0,10)
    x <- rnorm(n,50,8)
    y <- alpha + beta*x + e
    fit<-summary.lm(lm(y~x))
    coeffs<-fit$coefficients
# The next statement shows the output for the regression on the first simulated
# dataset; looking at the output explains why coeffs[2,4] is used below
    if (ii==1) print(fit$coefficients)
    if (coeffs[2,4]<0.05) rejects <- rejects + 1
    }
  print(paste("Sample size:",n,", estimated power:",rejects/nsims))
  }
```

```
mysim(19,417,10000)
              Estimate Std. Error    t value     Pr(>|t|)
(Intercept) -1.0111480 3.21913633 -0.3141054 0.7535988457
x            0.2191917 0.06362334  3.4451458 0.0006289053
[1] "Sample size: 417 , estimated power: 0.902"
```

Using a different seed yields different estimates for the first dataset of the simulation and a very slightly different power estimate for this sample size:

```
> mysim(37,417,10000)
             Estimate Std. Error   t value   Pr(>|t|)
(Intercept) 2.4377952 3.36466099 0.7245292 0.46914917
x           0.1508311 0.06653812 2.2668369 0.02391407
[1] "Sample size: 417 , estimated power: 0.9005"
```

SAS code using the first approach:

```
%macro epower(beta=,seed=,nsims=,n=);
%let sd_x=8; %let sd_e=10;

data simdata;
  %do i = 1 %to &nsims;
    i=&i;
    do j=1 to &n;
   x=50+rannor(&seed)*&sd_x;
   e=rannor(&seed)*&sd_e;
   y=&beta*x+e;
   output;
end;
  %end;

ods listing close;
ods output "Parameter Estimates"=params;
proc reg data=simdata;
   model y=x;
   by i;
   run;
quit;
ods output close;
ods listing;

*** The following code shows what is in the regression output    ;
*** in "params" dataset when i=1 and thus why "if variable='x'" ;
*** is used below.                                               ;

proc print data=params;
   where i=1;
```

```
data params;
   set params;
if variable='x';

if Probt<0.05 then reject=1; else reject=0;

proc freq data=params;
   table reject;

%mend;

%epower(beta=0.2,seed=97461,nsims=10000,n=417);
```

SAS output from the first approach:

```
Obs i Model  Dependent Variable  DF  Estimate   StdErr  tValue  Probt

  1 1 MODEL1     y     Intercept  1  -1.16503  3.36072   -0.35 0.7290
  2 1 MODEL1     y     x          1   0.22517  0.06601    3.41 0.0007

The FREQ Procedure
                                    Cumulative    Cumulative
reject    Frequency     Percent     Frequency      Percent
-------------------------------------------------------------
     0         998        9.98           998         9.98
     1        9002       90.02         10000       100.00
```

SAS code using the second approach:

```
%let alpha=20; %let beta=0.20; %let sd_e=10;

data pvals; set _NULL_;

%macro rept(seed0=,reps=,sampsize=);
%do i=1 %to &reps;

proc iml;
   setnull=j(&sampsize,1,0);
create begindat from setnull[colname='zero'];
   append from setnull;
quit;

data tempsamp;
   set begindat end=eof;
   retain seed &seed0;

call rannor(seed,z);
call rannor(seed,etemp);
err=&sd_e*etemp;
x=8*z + 50;
y=&alpha + &beta*x + err;
```

```
if eof then do;
call symput('seed0',put(seed,best.));
end;

*** The following code is to check the data being generated ;
*** and look at where the p-value is in the output dataset. ;
%if &i=1 %then %do;

proc means data=tempsamp;
   var z x;

proc reg data=tempsamp outest=temp tableout;
   model y = x;

proc print data=temp;
   var _TYPE_ Intercept x;

%end;

proc reg data=tempsamp outest=regests tableout noprint;
   model y = x;

data regests;
   set regests;
if _TYPE_ NE 'PVALUE' then delete;
p_int=Intercept;
p_x=x;
keep p_int p_x;

data pvals;
   set pvals regests;

%end;
%mend rept;

%rept(seed0=421325,reps=1000,sampsize=417);

data pvals;
  set pvals;
if p_x le 0.05 then reject1=1;
   else reject1=0;

proc freq;
   table reject1;
```

Edited SAS output from the second approach, including PROC MEANS and PROC REG output for the first dataset generated:

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
z	417	-0.0558561	1.0325162	-3.0516687	2.8930635
x	417	49.5531515	8.2601299	25.5866506	73.1445081

The REG Procedure

Number of Observations Read 417
Number of Observations Used 417

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	18.53427	3.10077	5.98	<.0001
x	1	0.22700	0.06173	3.68	0.0003

Obs	_TYPE_	Intercept	x
1	PARMS	18.5343	0.22700
2	STDERR	3.1008	0.06173
3	T	5.9773	3.67756
4	PVALUE	0.0000	0.00027
5	L95B	12.4391	0.10566
6	U95B	24.6295	0.34833

The FREQ Procedure

reject1	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	95	9.50	95	9.50
1	905	90.50	1000	100.00

(b) To use the formula from the "Power and Sample Size, Part III" overheads we need s_X and s_Y. Here $s_X = 8$. We are not given s_Y but can calculate it. Because α and β are constants and X and ε are independent:

$$\text{Var}(Y) = \text{Var}(\alpha + \beta \cdot X + \varepsilon) = \beta^2 \text{Var}(X) + \text{Var}(\varepsilon) = 0.2^2 \cdot 8^2 + 10^2 = 102.56.$$

So $s_Y = \sqrt{102.56} = 10.13$.

Letting $\widehat{\beta}_1$ be the alternative we are interested in being able to detect (0.2) we have

$$r = \frac{s_X}{s_Y}\widehat{\beta}_1 = \frac{8}{10.13} \cdot 0.2 = 0.158$$

We should have the same power to test

$$H_A : \beta_1 = 0.2 \quad \text{against} \quad H_0 : \beta_1 = 0$$

as to test

$$H_A : \rho = 0.158 \quad \text{against} \quad H_0 : \rho = 0$$

Here

$$Z_0 = \frac{1}{2}\log\left(\frac{1+0}{1-0}\right) = 0$$

and

$$Z_1 = \frac{1}{2}\log\left(\frac{1+0.158}{1-0.158}\right) = 0.159$$

The sample size n to give us power 0.90 for testing $\rho = 0.159$ versus $\rho = 0$ is

$$n = \frac{(z_{1-\alpha/2} + z_{1-\beta})^2}{(Z_{\rho_1} - Z_{\rho_0})^2} + 3 = \frac{(1.96 + 1.28)^2}{(0.159 - 0)^2} + 3 = 416.9$$

Rounding up, that gives a sample size of $n = 417$.