# A SAS® Macro Implementing an Extension of McNemar's Test for Clustered Data

Michael L. Lieber and Cindy Ashley
Department of Biostatistics and Epidemiology
Cleveland Clinic Foundation
Cleveland, OH

## ABSTRACT

McNemar's test is often used to compare two proportions estimated from paired observations. Obuchowski (Statistics in Medicine, in press) has proposed a method extending this to the case where the observations are sampled in clusters. This method is simple to implement and makes no assumptions about the correlation structure. A SAS macro implementing Obuchowski's method is presented.

## INTRODUCTION AND PURPOSE

McNemar's test[1] is often used to test the null hypothesis that $p_1=p_2$, where two proportions are estimated from paired observations (and are therefore correlated). Obuchowski (Statistics in Medicine, in press) has proposed a method extending this to the case where the observations are sampled in clusters. This method is a direct extension of the work of Rao and Scott[2] who compared independent groups of clustered binary data. Like Rao and Scott's method, Obuchowski's method makes no assumptions about the correlation structure.

This work was motivated by a study in diagnostic radiology where two tests, PET and SPECT, were compared for the detection of hyperparathyroidism. Seventy-two glands in 21 patients were evaluated by both tests. Following the tests, all patients underwent surgery to determine definitively whether or not hyperparathyroidism was present. The objective of the study was to compare the sensitivity and specificity of the two tests. To do this, one must account for not only the correlation between the results of the PET and SPECT tests, but also the intra-cluster correlation between the test results of glands in the same patient.

Obuchowski's method is a particularly good alternative to other methods that have been proposed for this type of data when little is known about the correlation pattern of the data.

## MACRO CLUSTPRO

### MACRO CALL:

```
%CLUSTPRO(INFIL=<'datafile'>,
NUM_C=);
```

The macro CLUSTPRO calculates Obuchowski's chi-square statistic (with one degree of freedom) and the associated p-value for testing $H_o$: $p_1=p_2$.

### PARAMETER DEFINITIONS:

**INFIL** is the input data file. The required input variables per line (record) are i, j, $x_{ij}$, and $n_j$, where:

i= the test number (1 or 2)
j= the cluster number (1,2,...)
$x_{ij}$= the number of events using test i in cluster j
$n_j$= the total number of observations in cluster j

**NUM_C** is the total number of clusters. Often (but not always) a cluster will be a patient.

## EXAMPLE

Suppose that there are 51 glands confirmed at surgery not to have hyperparathyroidism, that these 51 glands are from 21 patients, and that two different testing modalities will be used to classify each gland as either diseased or nondiseased (for the raw data, refer to Appendix B or Obuchowski, Table 6). We wish to compare between modalities the percentage of glands correctly classified as nondiseased (i.e., the specificities of the two tests). In this case, since the two proportions are based on the same sample of patients (i.e., each gland generates two paired observations, one from each modality), they are correlated. Furthermore, each patient can be considered a cluster of data, in that there may be more than one observation (gland) per patient. From equation 1 in Obuchowski, the estimated specificities of the two tests are 40/51= 0.784 and 46/51= 0.902.

This situation describes a two-tailed hypothesis test with:
$H_o$: $p_1=p_2$
$H_a$: $p_1 \neq p_2$

The macro call would be:

```
%CLUSTPRO(INFIL='/home/mlieber/
data.dat', NUM_C=21);
```

with the corresponding output:

```
p_hat1 = 0.7843137255
p_hat2 = 0.9019607843

chi-square statistic =
2.8571429014

degrees of freedom = 1

p-value = 0.0909689455

S
0.0051048816        0.0009334091
0.0009334091        0.00016062272
```

(where S is the variance-covariance matrix for $p_1$ and $p_2$ using Obuchowski's method)

## ACKNOWLEDGEMENTS

The authors wish to thank Elizabeth Lieber for her valuable suggestions.

Comments are invited. To obtain an electronic copy of the macro, please contact the authors.

Mike Lieber
Cleveland Clinic Foundation
Department of Biostatistics and
Epidemiology/P-88
9500 Euclid Avenue
Cleveland, OH 44195
(216) 444-9367

e-mail: mlieber@bio.ri.ccf.org

## REFERENCES

1. Conover, W.J. *Practical Nonparametric Statistics*, 2nd edition, John Wiley and Sons, New York, 1980.

2. Rao, J.N.K. and Scott, A.J. 'A simple method for the analysis of clustered binary data', *Biometrics*, **48**, 577-585 (1992).



## APPENDIX A

### MACRO %CLUSTPRO

```
/*   infil=input datafile -
should contain i, j, xij, and nj
(NOTE: Path to file should be
enclosed in single quotes.)
num_c= number of clusters
(patients) */
```

```
%macro clustpro (infil=,
num_c=);


/*  Dataset created from data
file  */

data tmp;
  infile &infil;
  input i j xij nj;
run;


/*  number of tests initialized
to 2  */

%let num_t=2;


/* Input data set sorted by i  */

proc sort data=tmp;
  by i;
run;


/*  Length of num_t and num_c
computed (used later in format-
ting upper loop values  */

%let lent=%length(&num_t);
%let lenc=%length(&num_c);


/*  "." added to end of lengths
(for formatting purposes)  */

data _null_;
  call symput
     ('fort',&lent||'.');
  call symput
     ('forc', &lenc||'.');
run;


/*  xij, nj put into macro
variables and trimmed  */

data _null_;
  set tmp;

  call symput ('x'||left
     (put(i, &fort))||left(put
     (j, &forc)), xij);
  call symput ('n'||left
     (put(j, &forc)), nj);
run;

%do k=1 %to &num_c;
    %let n&k=%eval(&&n&k);

    %do l=1 %to &num_t;
        %letx&l&k=%eval
          (&&x&l&k);
    %end;
%end;


/* sums of xij and nj computed */

proc means data=tmp noprint;
  by i;
  var xij nj;
  output out=a sum=sumx sumn;
run;


/*  sum of njs put into macro
variable  (used later)  */

data _null_;
  set a;
  call symput ('sumn', sumn);
run;


/*  &sum=sum of p_hats.  Will
change with each iteration of
loop.  */

%let sum=0;


/*  p_hat1, p_hat2 computed, put
into macro variables  */
```

```
%do k=1 %to &num_t;

  data b&k;
    set a;
    if i=&k;
    p_hat&k=sumx/sumn;
  run;

  data _null_;
    set b&k;
    tmp=&k;
    call symput
       ('p_hat'||left(put
       (tmp, 1.)), p_hat&k);
  run;
  %let sum=%sysevalf
     (&sum+&&p_hat&k);
%end;


/*  p bar computed  */

%let p_bar=%sysevalf
   (&sum/&num_t);


/*  part1= j/(j-1) (Used in
computing the matrix S)  */

%let part1=%sysevalf(&num_c*
   (1/(&num_c-1)));


/*  part3= (sum of njs)^2  (Used
in computing the matrix S)  */

%let part3=%eval(&sumn*&sumn);


/*  Part 2 computed  */

/*  Each iteration of part 2
initialized to 0  */

%do l=1 %to &num_t;
    %do m=1 %to &num_t;
        %let tcmp&l&m=0;
    %end;
%end;

/*  l represents i, m
represents i' */

%do l=1 %to &num_t;
    %do m=1 %to &num_t;


/* Since S is symmetric, tcmp&l&m
is only calculated if tcmp&m&l
hasn't been calculated  */

        %if &l > &m %then %do;
            %let tcmp&l&m=
               &&tcmp&m&l;
        %end;

        %else %do;


/*  Each iteration of the loop
represents an iteration of the
summation from 1 to j  */

           %do k=1 %to &num_c;
              %letcomp&l&m&k=
                %sysevalf
                ((&&x&l&k-
                (&&n&k*&p_bar))
                *(&&x&m&k-
               (&&n&k*&p_bar)));
              %let tcmp&l&m=
                %sysevalf
                (&&tcmp&l&m+
                &&comp&l&m&k);
          %end;
        %end;
    %end;
%end;


/*  Sii' calculated  */

%do l=1 %to &num_t;
    %do m=1 %to &num_t;
        %let s&l&m= %sysevalf
          (&&tcmp&l&m*&part1
           &part3);
    %end;
%end;
```

```
/*  Chi-square value calculated,
put into data set variable  */

%let phatcalc=%sysevalf
  (&p_hat1 - &p_hat2);
%let numer=%sysevalf
  (&phatcalc*&phatcalc);
%let denom1=%sysevalf
  (&s11+&s22);
%let denom2=%sysevalf(2*&s12);
%let denom=%sysevalf
  (&denom1-&denom2);
%let chisq=%sysevalf
  (&numer/&denom);

data final;
  chi=&chisq;
run;


/*  p-value calculated with i-1
degrees of freedom, put into
macro variable  */

data _null_;
  set final;

  pvalue=1-probchi
    (chi, %eval(&num_t-1));
  call symput ('pvalue', pvalue);
run;


/*  Degrees of freedom put into
macro variable  */

%let dof=%eval(&num_t-1);


/*  Calculations output to output
window  */

data _null_;
  file print;

%do l=1 %to &num_t;
    put @5 "p_hat&l = &&p_hat&l";
%end;
put;

put @5 "chi-square statistic
  = &chisq";
put;

put @5 "degrees of freedom
  = &dof";
put;

put @5 "p-value = &pvalue";
put;
put;
put @5 "S";
put;

%do l=1 %to &num_t;
    %do m=1 %to &num_t;
        %let fmt=%eval
          (%sysevalf(14*&m)-9);
        put @&fmt "&&s&l&m" @;
    %end;
    put;
%end;

put;
put;

run;

%mend clustpro;
```

## APPENDIX B

**'DATA.DAT'**

```
1 1 0 3
1 2 2 3
1 3 3 3
1 4 1 1
1 5 2 3
1 6 4 4
1 7 3 3
1 8 2 2
1 9 2 2
1 10 1 1
1 11 2 3
1 12 2 2
1 13 3 3
1 14 2 2
```

```
1 15 0 2
1 16 2 3
1 17 2 3
1 18 2 3
1 19 2 2
1 20 1 1
1 21 2 2
2 1 2 3
2 2 3 3
2 3 3 3
2 4 1 1
2 5 3 3
2 6 4 4
2 7 3 3
2 8 2 2
2 9 1 2
2 10 1 1
2 11 2 3
2 12 2 2
2 13 3 3
2 14 2 2
2 15 2 2
2 16 2 3
2 17 2 3
2 18 3 3
2 19 2 2
2 20 1 1
2 21 2 2
```