DATA STEP KAPPA COMPUTATIONS

Peter Gaccione
Harvard School of Public Health

Abstract

This workshop leads users through the data manipulations necessary fo calculate the weighted kappa statistic (Spitzer, Cohen, Fleiss, and Endicott, 1967). As a generally accepted measure of rating reliability, kappa can be used to estimate an amount of agreement among raters. In this presentation, data from two physicians who rated films produced by magnetic resonance imaging (MRI) will be analyzed. In addition a SAS program utilizing data step statements will be shown, and its dissection will indicate the flexibility SAS allows in programming within the data step for calculations not readily available from procedure output. Emphasis will be placed on the program commands and how the data is set up to perform the computations. Output from the PRINT procedure will be used extensively.

Description of Problem and Statistic

Sometimes there are no existing standards for measurement of rating agreement in biomedical applications, yet accuracy and reliability are required for correct diagnosis. One method which is applied is to have more than one "scorer" assign value to the outcome, and to then use some measure of scorer or rater agreement as an indication of the reliability of the diagnosis.

Weighted kappa is appropriate for measuring agreement among raters who assign categorical scores to certain outcomes. The example given is for data collected from two radiologists scoring MRI pictures of 68 patients. Each patient had two images taken; one before an illuminating agent was ingested, and one afterwards. Each radiologist rated the "clarity" of the particular image, and the difference between post-agent score and pre-agent rating was taken. Positive scores corresponded to the post-agent image being clearer. Individual ratings ranged from 0 to 4, so that the range of differences was -4 to 4 or 9 categories. These differences as ratings were then compared.

Weighted kappa as used here take the form

$$K_w = \Sigma w F_o / \Sigma w F_e$$

where F_o and F_e are the observed and expected frequencies for the categories of the **diff1** by **diff2** (the variable names given to rater1's and rater2's scores) contingency table, and w is the weight applied to the DISAGREEMENT. If both raters concur, the weight is 0. If their scores are one category away (e.g. rater1 gives a value of 1 and rater2 gives a value of 2) then the weight is 1, and do on. K_w ranges from -1 to 1, with negative values indicating overall disagreement, 0 indicating chance agreement, and positive values exhibiting agreement.

Program Flow

As the formula above might suggest we can divide the program to compute K_w into three main parts: computing the numerator, computing the denominator, and computing kappa. Add to this reading in the data and some PROC PRINT statements for checking things along the way and a logical flow can take shape.

The first part of the program reads in the data, transfers the range of values for the raters' scores **diff1** and **diff2** from -4 to 4 to 1 to 9 for easier tabulation, and does a PROC FREQ to get the counts of the number of matches of **diff1** by **diff2.** These counts are the observed frequencies and will be used in computing F_o of the formula. Note that the output from the FREQ procedure only gives counts for the cells of the contingency table which have values. If the entire 9 by 9 table was desired (1 to 9 for

rater1 by 1 to 9 for rater2) there would be many 0's, which don't need to be considered here. This would be available by using the SPARSE option after the "/" in the tables statement. The program code is:

```
data freq; input diff1 diff2 @@;
diff1=diff1+5
diff2=diff2+5;  cards;
-2 -4 -2 -2 0 -1 0 -1 0 -1 0 -1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 2
0 2 0 2 0 2 1 -1 1 0 1 0 1 0 1 0 1 1 1 1 1
1 1 1 1 1 1 1 1 2 1 2 1 2 1 2 1 2 1 2 1 2
1 2 1 2 1 2 1 2 1 2 1 2 1 2 2 0 2 3 0 2 0
2 1 2 1 2 1 2 1 2 1 2 1 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 3 2 3 2 3 3 3 3 3
;
 proc freq; tables diff1*diff2/out=obs;
 proc print data=obs;
 var diff1 diff2 count;
```

The contingency table output is in Appendix 1, and the print output is as follows:

OBS	DIFF1	DIFF2	COUNT
1	3	1	1
2	3	3	1
3	5	4	4
4	5	5	10
5	5	6	2
6	5	7	4
7	6	4	1
8	6	5	4
9	6	6	6
10	6	7	14
11	7	5	3
12	7	6	6
13	7	7	7
14	7	8	3
15	8	8	2

We note that only 15 of the possible 81 cells have values, one of which is , for example, where rater1's value (**diff1**) is 3 and rater2's value (**diff2**) is 3. There was one image to which they gave a score of 3.

As mentioned above we already have the observed frequencies, but need the expected. We do this using the notion of independent events and the product of probabilities. If we think of the 2 by 2 **diff1** by **diff2** contingency table as alluded to above then each rater will have marginal totals for each of his scores. If **diff1** is the row variable from the above table we see these row totals to be 2 for his rating of 3, 20 for his rating of 5, 25 for his rating of 6, 19 for his rating of 7 and 2 for his rating of 8). Similar column totals exist for the column variable **diff2** representing rater2's scores. The next two parts of the program compute these totals (data set **one** holds the observed scores for each of the cells and data sets **diff1** and **diff2** will contain the marginal totals for variables **diff1** and **diff2**). While creating these data sets we also compute the pieces of the numerator of K_w and hold them for inclusion with the expected cell counts later on to compute the statistic. The program statements are as follows:

```
proc sort data=obs; by diff1 diff2;
data one diff1; set obs(drop=percent);
    by diff1;
if first.diff1 then count1=0;
retain obsscore 0 count1;
diff12=abs(diff1-diff2);
oscore=diff12*count;
obsscore=obsscore + oscore;
count1=count1+count;
    output one;
if last.diff1 then output diff1;
proc print data=one;
var diff1 diff2 diff12 count count1 oscore
   obsscore;
title 'data set one, observed for diff1';
proc print data=diff1; var diff1 count1;
title 'data set diff1, observed for diff1';

  proc sort data=obs; by diff2;
   data two diff2; set obs(drop=percent);
    by diff2;
   if first.diff2 then count2=0;
   retain obsscore 0 count2;
   diff12=abs(diff1-diff2);
   oscore=diff12*count;
   obsscore=obsscore + oscore;
count2=count2+count;
```

```
  output two;
if last.diff2 then output diff2;
proc print data=diff2; var diff2 count2;
title 'data set diff2, observed for diff2';
```

Data steps "**data one diff1**" and "**data two diff2**" do the same things. The variable **count1** will be the counter for our marginal totals. The variable **diff12** will be the DISAGREEMENT weight w of the kappa statistic. As described previously for our application it will be the number of categories rater1's value is away from rater2's value. The statement "**diff12=abs(diff1-diff2);**" creates these weights. Then for each cell a w times F_o must be calculated, and this is what variable **oscore** does. Variable **obsscore** keeps a running tab of the **oscore**'s within each of rater1's (or rater2's) rated scores so as to compute the ΣwF_o of the numerator of K_w. The output from the first **proc print** is in Appendix 2; the next two **proc print** statements show marginal totals for the two variables as follows:

data set diff1, observed for diff1

OBS	DIFF1	COUNT1
1	3	2
2	5	20
3	6	25
4	7	19
5	8	2

data set diff2, observed for diff2

OBS	DIFF2	COUNT2
1	1	1
2	3	1
3	4	5
4	5	17
5	6	14
6	7	25
7	8	5

So we see the marginal totals for rater2 are one rating of 1, 1 rating of 3, 5 ratings of 4, 17 ratings of 5, 14 ratings of 6, 25 ratings of 7, and 5 ratings of 8. We also note that 35 out of the 81 possible cells have observed values, so we will have to find 35 expected values to match them. This is what the next part of the program does. Variables **row1-row9** and **column1-column9** are created to hold the marginal totals and at the same time get observations having the value of **diff1**, the value of **diff2**, the row total for that **diff1**, and the column total for that **diff2**. Thus the expected value for that cell can be found by multiplying the two mariginal totals together and dividing by the total n, the variable **totaln**. The following statements compute the expected values.

```
data three;
set diff1(keep=diff1 count1)
   diff2(keep=diff2 count2);
proc sort; by diff1 diff2;
proc print;

data four; set three;
array row row1-row9;
array column column1-column9;
retain row1-row9 column1-column9;
   do i=1 to 9;
 if diff1=i then row[i]=count1;
if diff2=i then column[i]=count2;
  end;
totaln=sum(of column1-column9);
 do i=1 to 9;
   do j=1 to 9;
   expect=row[i]*column[j]/totaln;
   diff1=i; diff2=j;
   if expect ne . then output;
   end;
  end;
 drop row1-row9 column1-column9
count1 count2 i j;
proc sort; by diff1 diff2;
proc print;  var diff1 diff2 row3 row5 row6
row7 row8 colymn1 column3-column8
expect;
```

Some relevant lines of the output show the values of **diff1** and **diff2** matched along with the marginal totals, the total sample size, and the expected values for that match of **diff1** by **diff2**. That output is shown in Appendix 3.

The final piece to the program puts the numerator and denominator together to compute kappa. Data sets **four** and **one** are merged, **four** containing the expected values (variable **expect**) for the cells and one containing the observed values (variable **obsscore**). A running tab for the expected cell counts is kept (**expscore**) in the same fashion as **obsscore** was kept in data set **one**, and then a running kappa is kept as the summations of the observed and expected increase until all 35 cells are included. The program statements are :

```
data freqkapp; merge four one;
by diff1 diff2; if last.diff2;
   retain expscore 0 kappa obsscore;
   diff12=abs(diff1-diff2);
    escore=diff12*expect;
    expscore=expscore+escore;
    kappa=1-(obsscore/expscore);
    drop count1  totaln oscore escore;
    if count ne . then output;
    proc print;
```

The output is shown in Appendix 4, and the final observation shows kappa=0.3937.

Interpretation

Two approaches are possible for determining the strength of the relationship. Cohen (1968) gives a formula for the standard error (σ_k) of K_w. A p-value for testing the null hypothesis of no agreement can be found by $z = K_w/\sigma_k$. The other approach is to view the absolute magnitude of the value, and Landis and Koch (1977) have suggested strengths for K_w values:

<0	Poor
0.00-0.20	Slight
0.21-0.40	Fair
0.41-0.60	Moderate
0.61-0.80	Substantial
0.81-1.00	Almost Perfect

For this application we note that there is almost moderate agreement among the raters given the "penalty" of disagreement of 1 unit.

Discussion

The usage of the data step in SAS to compute statistics not readily available from procedural output can sometimes be a daunting task, particularly for an investigator who is not an experienced programmer. However, if the problem is well thought out and modularized, and a program is created with checks along the way, the desired values can be computed straightforwardly, and the experimenter can gain added insight into the construction and correct usage of the created statistics.

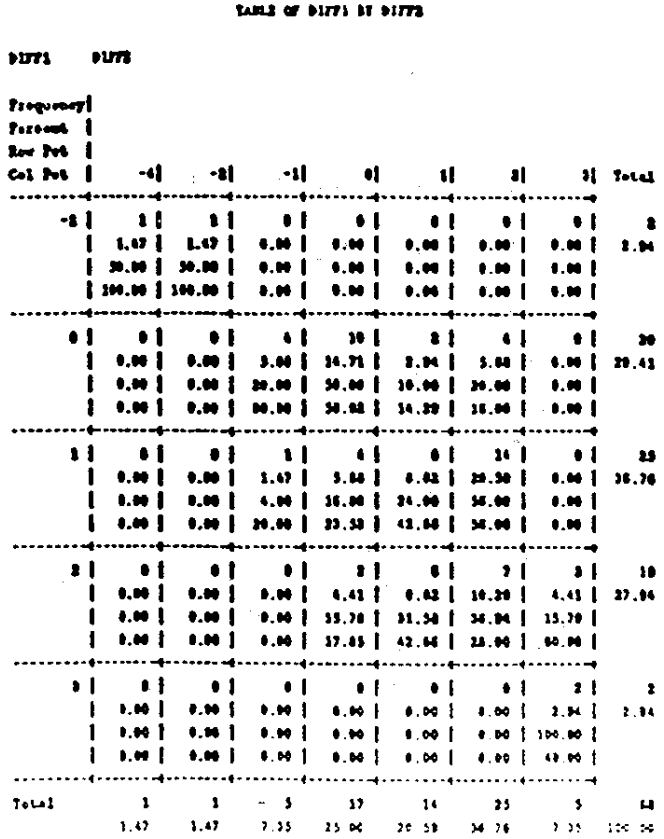
References

Bartko, J. J. (1966). The Intraclass Correlatio Coefficient as a Measure of Reliability. Psychological Reports, 19, 3-11.

Cohen, J. (1968). Weighted Kappa: Nominal Scale Agreement With Provision for Scaled Disagreement for Partial Credit. Psycholocical Bulletin, 70, 213-220.

Spitzer, R. L., Cohen, J., Fleiss, J., and Endicott, J. (1967). Quantification of Agreement in Psychiatric Diagnosis. Arch. Gen. Psychiatry, 17, 83-87.

APPENDIX 1



APPENDIX 2

OBS	DIFF1	DIFF2	DIFF12	COUNT	COUNT1	OSCORE	OBSSCORE
1	3	1	2	1	1	2	2
2	3	3	0	1	2	0	2
3	5	4	1	4	4	4	6
4	5	5	0	10	14	0	6
5	5	6	1	2	16	2	8
6	5	7	2	4	20	8	16
7	6	4	2	1	1	2	18
8	6	5	1	4	5	4	22
9	6	6	0	6	11	0	22
10	6	7	1	14	25	14	36
11	7	5	2	3	3	6	42
12	7	6	1	6	9	6	48
13	7	7	0	7	16	0	48
14	7	8	1	3	19	3	51
15	8	8	0	2	2	0	51

APPENDIX 3

OBS	DIFF1	DIFF2	ROW3	ROW5	ROW6	ROW7	COLUMN1	COLUMN3	COLUMN4	COLUMN5	COLUMN6	COLUMN7	COLUMN8	EXPECT
5	3	1	2	20	25	19	1	1	5	17	14	25	5	.02941
15	3	4	2	20	25	19	1	1	5	17	14	25	5	.14706
20	3	5	2	20	25	19	1	1	5	17	14	25	5	.50000
25	3	6	2	20	25	19	1	1	5	17	14	25	5	.41176
30	3	7	2	20	25	19	1	1	5	17	14	25	5	.73529
35	3	8	2	20	25	19	1	1	5	17	14	25	5	.14706
39	5	1	2	20	25	19	1	1	5	17	14	25	5	.29412

APPENDIX 4

OBS	EXPECT	COUNT	OBSSCORE	DIFF12	EXPSCORE	KAPPA
1	0.02941	1	2	2	0.0588	-33.0000
2	0.02941	1	2	0	0.0588	-33.0000
3	1.47059	4	6	1	9.3529	0.3585
4	5.00000	10	6	0	9.3529	0.3585
5	4.11765	2	8	1	13.4706	0.4061
6	7.35294	4	16	2	28.1765	0.4322
7	1.83824	1	18	2	39.2059	0.5409
8	6.25000	4	22	1	45.4559	0.5160

9	5.14706	6	22	0	45.4559	0.5160
10	9.19118	14	36	1	54.6471	0.3412
11	4.75000	3	42	2	74.8088	0.4386
12	3.91176	6	48	1	78.7206	0.3902
13	6.98529	7	48	0	78.7206	0.3902
14	1.39706	3	51	1	80.1176	0.3634
15	0.14706	2	51	0	84.1176	0.3937