PharmaSUG 2012 - Paper HO01

Multiple Techniques for Scoring Quality of Life Questionnaires

Brandon Welch, Rho®, Inc., Chapel Hill, NC
Seungshin Rhee, Rho, Inc., Chapel Hill, NC

ABSTRACT

In the clinical trials computing environment, data sets come in a variety of shapes and sizes. From laboratory data to electrocardiogram (ECG) measurements, transforming raw data to analysis-ready SAS® data sets is often complicated. New challenges arise when we receive data collected from quality of life (QOL) questionnaires. With these data we often compute scores that measure underlying scales – such as mental or social well-being. There are many different types of questionnaires, and it is advantageous to have an arsenal of programming tools when calculating the appropriate scores. In this article, we present a mock questionnaire and common techniques to achieve appropriate calculations. Depending on the input data structure, we illustrate how to calculate scores using various techniques including ARRAY processing, PROC SQL, and simple SAS functions. The techniques we present offer a good overview of basic data step programming and SAS procedures that will educate SAS users at all levels.

INTRODUCTION

Analyzing QOL questionnaire data is commonplace in clinical trials research. The results give investigators a quantitative assessment of a patient's well-being on multiple dimensions. We estimate these dimensions or scales by performing calculations on the survey response items, and each questionnaire is designed with particular rules for scoring. The variation across each questionnaire and the methods for arriving at the final scores offers a unique challenge for SAS programmers.

In this article, we present common ways of computing scores from a mock questionnaire. These techniques developed from our experience in computing scores on some of the more popular QOL questionnaires. In our experience, it isn't the actually scoring that provides the challenge, but the actual imputation methods outlined for each questionnaire. Therefore we show how to perform a mean imputation for missing values, which is a standard imputation method across questionnaires. Depending on the input data structure, we illustrate how to compute a PHYSICAL, MENTAL, and TOTAL score using a variety of SAS programming techniques depending on the input data structure. In the first scenario we demonstrate techniques for tackling a vertically structured data set. We use PROC SQL to reverse score, find the number of missing values, and calculate the mean response on the non-missing values. We complete the calculations in a data step in which we impute and calculate the PHYSICAL, MENTAL, and TOTAL scores. In the second scenario we illustrate how to arrive at the same calculations when the input data set is horizontal using ARRAY processing and SAS functions.

MOCK QUESTIONNAIRE

Below is the Welch-Rhee Headache Indicator (WRHIND), which is a mock questionnaire that captures data measuring physical and mental well-being. The physical and mental scores are calculated by summing across items 1, 4, 6, 8, 10 and 2, 3, 5, 7, 9 respectively. A mean imputation is performed prior to summation if the number of non-missing values is greater than two in each scale. Higher numbers indicate poor QOL, and items 3, 8, and 10 are reverse scored to keep the ordinal direction the same as the remaining items.

Question	1 = Strongly Disagree	2 = Disagree	3 = Neutral	4 = Agree	5 = Strongly Agree
1. My headaches have gotten worse with age.					
2. My headaches interfere with my abilities to socialize with others.					
3. I tend to worry less because of my reduced number of headaches.					
4. My headaches are more severe in the mornings.					
5. I'm less willing to speak in groups because of my headaches.					

6. My allergies adversely affect my headaches.					
7. Life seems more difficult because of my headaches.					
8. The frequency of headaches decline the more I exercise.					
9. My headaches affect my self-confidence.					
10. Prescription pain medication relieves my headaches.					

Table 1. Mock headache questionnaire

COMPUTING THE SCORES

VERTICAL STRUCTURE

IMPUTE BY MEAN SUBSTITUTION

In this scenario we receive data in a vertical structure. Below is a snapshot (*n = 10*) of the SAS data set (WRHIND) keyed from the WRHIND:

Questionnaire Item	Subject ID	Treatment Group (Numeric)	Questionnaire Response
P1	001850	2	2
M2	001850	2	.
M3	001850	2	2
P4	001850	2	1
M5	001850	2	3
P6	001850	2	3
M7	001850	2	4
P8	001850	2	2
M9	001850	2	4
P10	001850	2	3

Output 1. Vertically structured data set

For each questionnaire item, P denotes those questions related to physical attributes; whereas, M denotes those for mental.

The first step prior to summation is to reverse score items 3, 8, and 10 and calculate the mean on the non-missing values per scale. Here we use PROC SQL to calculate the number of missing values, reverse score, and mean of coded values per subject ID and type of scale:

```
*IMPUTATION METHOD 1 - CALCULATE MEAN OF RECODED VARIABLE, IMPUTE, THEN SUM;
PROC SQL noprint;
  create table getmean as
  select *,
```

```
  /*CREATE GROUPING FOR TWO CONSTRUCTS*/
  case
    when index(quesc,'P') then 'PHYS'
    when index(quesc,'M') then 'MENT'
    else ''
  end
  as type,

  /*GET NUMBER OF MISSINGS PER CONSTRUCT*/
  nmiss(respn) as missresp,

  /*REVERSE SCORE ITEMS*/
  case
    when quesc in ('P8' 'P10' 'M3') then 6 - respn
    else respn
  end
  as r_respn,

  /*GET MEAN OF RECODED VARIABLE*/
  mean(calculated r_respn) as meanresp

  from wrhind
  group by id, type
  order by id, type
  ;
QUIT;
```

OUTPUT

Here is a snapshot of the resulting output data set (*n = 5*) from PROC SQL:

Questionnaire Item	Subject ID	Treatment Group (Numeric)	Questionnaire Response	Scale	Number of Missings	Reversed Scored Item	Mean Response
M2	001850	2		MENT	1		3.75
M3	001850	2	2	MENT	1	4	3.75
M5	001850	2	3	MENT	1	3	3.75
M7	001850	2	4	MENT	1	4	3.75
M9	001850	2	4	MENT	1	4	3.75

Output 2. Output from PROC SQL

Now we have all the necessary pieces to calculate the PHYSICAL, MENTAL and TOTAL scores. We do this in a data step by using RETAIN and summation.

```
DATA vscore1;
  set getmean;
  by id type;

  retain phys ment;
```

```
    *IF NUMBER OF MISSINGS < 3 THEN IMPUTE;
    if      missing(r_respn) and missresp < 3  then r_respn = meanresp;
    else if missing(r_respn) and missresp >= 3 then r_respn = .;

    if first.id then do;
      phys = .;
      ment = .;
    end;

    if index(quesc,'P') and not missing(r_respn) then phys = sum(phys, r_respn);
    if index(quesc,'M') and not missing(r_respn) then ment = sum(ment, r_respn);

    if last.id then do;
      if nmiss(phys, ment) = 0 then total = sum(of phys, ment);
      output;
    end;

    keep id phys ment total trtn;

  RUN;
```

OUTPUT

First five patients:

Subject ID	Treatment Group (Numeric)	Physical Score	Mental Score	Total Score
001651	1	13.75	10.00	23.75
001850	2	13.00	18.75	31.75
002240	1	15.00	14.00	29.00
002244	1	13.00	5.00	18.00
002746	1	15.00	15.00	30.00

Output 3. Output for vertical data set summary

PROC MEANS or PROC SUMMARY similarly calculates the means on the non-missing values prior to the data step. Consequently we merge on to VSCORE1.

```
  PROC MEANS data = premeans noprint;
    var respn;
    class id type;
    types id*type;
    output out = getmean_ n     = n
                          mean  = mean
                          nmiss = nmiss;
  RUN;
```

We prefer PROC SQL because the PROC MEANS/SUMMARY method requires we create the TYPE variable and perform the reverse scoring in a previous data step. PROC SQL allows us to calculate all in one step.

IMPUTE BY ADJUSTING COMPUTED SCORE

In the above example we compute the mean of the non-missing values and substitute its value for the missing values. Alternatively we arrive at the same summation by modifying the resulting computed score. This allows us to supersede using a SAS PROC to compute the mean.

Let *m = number of missing values* and *n = number of non-missing values*:

$$SCORE = (x_1 + x_2 + \cdots + x_n) + (m)\frac{(x_1+x_2+\cdots+x_n)}{n}.$$

This (above) is the mean substitution method and equates to

$$SCORE = \frac{n(x_1+x_2+\cdots+x_n)}{n} + (m)\frac{(x_1+x_2+\cdots+x_n)}{n}.$$

Substituting $\overline{x}$ we have

$$SCORE = n\overline{x} + m\overline{x}$$

Simple algebra yields

$$SCORE = \frac{(n+m)}{n}(x_1 + x_2 + \cdots + x_n).$$

Therefore, calculating the mean value and imputing the missing values is equivalent to adding up the non-missing values and multiplying by $\frac{(n+m)}{n}$. For example, suppose one patient has values 1, ., 3, ., 5. Using the mean imputation method we would calculate the mean of the non-missing values ($\overline{x} = 3$) and *SCORE = 1 + 3 + 3 + 3 + 5 = (1+3+5) + 2(3) = 15*. Using the derivation above we use

$$SCORE = \frac{3+2}{3}(1+3+5) = \frac{5}{3}(9) = 15$$

We illustrate this approach in the following data step:

```
*IMPUTATION METHOD 2 - ADJUST SUMMATION AT THE END;
PROC SORT data = wrhind; by id;
DATA vscore2;
  set wrhind;
  by id;

  retain phys ment p_nonmis m_nonmis p_miss m_miss;

  if first.id then do;
    phys     = .; ment     = .;
    p_nonmis = 0; m_nonmis = 0;
    p_miss  = 0;  m_miss   = 0;
  end;

  if index(quesc,'P') then do;

    *REVERSE SCORE;
    if not missing(respn) and quesc in ('P8' 'P10') then r_respp = 6 - respn;
    else                                                 r_respp = respn;

    if not missing(r_respp) then do;
      p_nonmis + 1;
      phys = sum(phys, r_respp);
    end;
    else p_miss + 1;
```

```
    end;

    else if index(quesc,'M') then do;

      *REVERSE SCORE;
      if not missing(respn) and quesc in ('M3') then r_respm = 6 - respn;
      else                                           r_respm = respn;

      if not missing(r_respm) then do;
        m_nonmis + 1;
        ment = sum(ment, r_respm);
      end;
      else m_miss + 1;

    end;

    if last.id then do;

      *ADJUST IF NON-MISSINGS > 2;
      if p_nonmis > 2 then phys = ((p_nonmis + p_miss) / p_nonmis) * phys;
      if m_nonmis > 2 then ment = ((m_nonmis + m_miss) / m_nonmis) * ment;

      if nmiss(phys, ment) = 0 then total = sum(of phys, ment);
      output;
    end;

    keep id phys ment total trtn;

  RUN;
```

OUTPUT

First five patients

Subject ID	Treatment Group (Numeric)	Physical Score	Mental Score	Total Score
001651	1	13.75	10.00	23.75
001850	2	13.00	18.75	31.75
002240	1	15.00	14.00	29.00
002244	1	13.00	5.00	18.00
002746	1	15.00	15.00	30.00

Output 4. Output for vertical data set summary (adjusted score method)

HORIZONTAL STRUCTURE

ARRAY PROCESSING

Now suppose we receive our input data set in the following structure (first five patients)

Subject ID	Treatment Group (Numeric)	P1	P4	P6	P8	P10	M2	M3	M5	M7	M9
001651	1	.	3	3	4	3	1	1	1	1	2
001850	2	2	1	3	2	3	.	2	3	4	4
002240	1	2	2	3	1	3	2	1	2	2	3
002244	1	3	2	1	1	4	.	.	3	.	2
002746	1	4	.	1	2	.	.	2	.	2	3

Output 5. Horizontally structured data set

Upon transposing these data, we use the same methods described above. We arrive at the same calculations with ARRAY processing without modifying the data structure:

```
*ARRAY APPROACH;
DATA hscore1;
  set t_wrhind;

  array pvars (5) p1 p4 p6 p8 p10;
  array mvars (5) m2 m3 m5 m7 m9;
  do i = 1 to 5;

    *REVERSE SCORE;
    if upcase(vname(pvars{i})) in ('P8' 'P10') and not missing(pvars{i}) then
    pvars{i} = 6 - pvars{i};

    phys     = sum (of pvars(*));
    p_miss   = nmiss(of pvars(*));
    p_nonmis = n(of pvars(*));

    *REVERSE SCORE;
    if upcase(vname(mvars{i})) in ('M3') and not missing(mvars{i}) then mvars{i} =
    6 - mvars{i};

    ment     = sum (of mvars(*));
    m_miss   = nmiss(of mvars(*));
    m_nonmis = n(of mvars(*));

  end;

  *IF NUMBER OF MISSINGS < 3 THEN ADJUST;
  if p_miss < 3 then phys = ((p_nonmis + p_miss) / p_nonmis) * phys;
  if m_miss < 3 then ment = ((m_nonmis + m_miss) / m_nonmis) * ment;

  if nmiss(phys, ment) = 0 then total = sum(phys, ment);

  keep id phys ment total;

RUN;
```

SAS FUNCTIONS – NO ARRAYS

```
*SAS FUNCTION APPROACH;
DATA hscore2;
  set t_wrhind;

  *REVERSE SCORE;
  if not missing(p8)  then p8  = 6 - p8;
  if not missing(p10) then p10 = 6 - p10;
  if not missing(m3)  then m3  = 6 - m3;

  phys = sum(of p1, p4, p6, p8, p10);
  ment = sum(of m2, m3, m5, m7, m9);

  *GET NUMBER OF MISSINGS;
  p_miss = nmiss(p1, p4, p6, p8, p10);
  m_miss = nmiss(m2, m3, m5, m7, m9);

  *GET NUMBER OF NON-MISSINGS;
  p_nonmis = n(p1, p4, p6, p8, p10);
  m_nonmis = n(m2, m3, m5, m7, m9);

  if p_miss < 3 then phys = ((p_nonmis + p_miss) / p_nonmis) * phys;
  if m_miss < 3 then ment = ((m_nonmis + m_miss) / m_nonmis) * ment;

  if nmiss(phys, ment) = 0 then total = sum(phys, ment);

  keep id phys ment total trtn;

RUN;
```

GENERALIZED METHOD

In practice, QOL questionnaires may have more than two scales of interest. It is often necessary to generalize the code to expect more than two scales. The SAS macro language provides ways of making programs more flexible. In the following example, we account for more scales, and also add a parameter for an imputation cutoff. For all examples above, we use a 50% cutoff. For brevity we build a macro around the ‘SAS Function’ example. Note we transpose the data since the original structure is vertical:

```
%macro Score(Items  =,
          Scale  =,
          Cutoff =);

  %*GET NUMBER OF ITEMS;
  %let INum = 0;
  %let Scroll = %scan(&Items,%eval(&INum + 1));
  %do %while(&Scroll ne);
    %let INum = %eval(&INum + 1);
    %let Scroll = %scan(&Items,&INum + 1);
  %end;

  %put;
  %put NUMBER OF ITEMS: &INum;
  %put;
```

```
  &Scale.num = &INum;

  %*MAKE COMMA DELIMITED FOR NMISS FUNCTION;
  %let List = ;
  %do i = 1 %to &INum;
    %let List = &List%str(,)%scan(&Items,&i);
  %end;

  %let List = %substr(&List,2); %*STRIP OFF LEADING COMMA;

  &Scale = sum(of &Items);

  *GET NUMBER OF MISSINGS;
  &Scale.ms  = nmiss(&List);

  *GET NUMBER OF NON-MISSINGS;
  &Scale.nms = n(&List);

  %*ADJUST SCORE IF IMPUTATION NECESSARY;
  if &Scale.ms <= %sysevalf(&Cutoff * &INum) then &Scale = ((&Scale.nms +
  &Scale.ms) / &Scale.nms) * &Scale;

%mend;

*SAS FUNCTION APPROACH - MACROTIZED;
DATA hscore3;
  set t_wrhind;

  *REVERSE SCORE;
  if not missing(p8)  then p8  = 6 - p8;
  if not missing(p10) then p10 = 6 - p10;
  if not missing(m3)  then m3  = 6 - m3;

  %Score(Items  = p1 p4 p6 p8 p10,
         Scale  = phys,
         Cutoff = 0.5);

  %Score(Items  = m2 m3 m5 m7 m9,
         Scale  = ment,
         Cutoff = 0.5);

  %Score(Items  = p6 p8 m2 m3 m5 m7 m9,
         Scale  = mix,
         Cutoff = 0.2);

  if nmiss(phys, ment) = 0 then total = sum(phys, ment);

RUN;
```

Notice we apply reverse scoring and compute the totals outside of the macro. In addition, we derive a new scale called MIX in which we combine both physical and mental attributes.

CONCLUSIONS

As this article illustrates, the SAS system provides many ways to derive summed scores for QOL questionnaires. How to attack the problem is a question of preference. PROC SQL or RETAIN in a data step are useful when the data set is vertically structured. Conversely, if one favors a horizontal structure (or receives a horizontal data set), ARRAYs or SAS functions are available. For generalizing methods, macro processing provides flexibility in automating iterative steps. In conclusion, a variety of approaches are available with the many SAS tools at our disposal.

ACKNOWLEDGMENTS

Steve Noga
Eva J. Welch

CONTACT INFORMATION

Brandon Welch
Rho®, Inc.
6330 Quadrangle Dr., Ste. 500
Chapel Hill, NC 27517
Phone: 919-595-6339
Fax: 919-408-0999
Email: Brandon_Welch@rhoworld.com
Web: www.rhoworld.com