A SAS TESTSCORING AND ITEM ANALYSIS PROGRAM

Charles E. Rich, Ohio University

Introduction
This SAS program, referred to as SCOR, scores multiple choice tests and provides item analyses with the goal of improving classroom tests and minimizing the routine record-keeping work of instructors. SCOR is based on the SAS PROC ITEM, a version of which was first presented at SIGI (Deitz & Smith, 1980) and was later published as SAS Technical Report No. S-126 (Smith, 1980). SCOR is currently the standard program used by the Testscoring Office at Ohio University to perform testscoring functions and produce various raw data files. The staff operator uses CMS Exec's to build and edit SCOR according to features requested by each user. Typically five to ten tests are processed in one multi-step job, run in batch on an IBM 370-158. In each job step SCOR is tailored according to requested features and a particular data file. The data files are created by processing NCS computer answer sheets on an NCS 7003 optical scanner. For each test a maximum of 200 questions with two to five possible choices per question may be processed.

Summary Statistics and Item Analysis
PROC ITEM produces such statistics as the mean score, standard error, reliability indexes, and point biserial correlations to name a few. Beyond the summary statistics provided by PROC ITEM, SCOR provides the median, mode, and measures of skewness and kurtosis by using an output data set from PROC UNIVARIATE. The item analysis is expanded by providing raw frequency counts per choice per question for the total class and for the upper and lower 25% scoring sub-groups. The indexes of difficulty and discrimination are then calculated using these sub-groups, as recommended by Ebel (1979). A five-page descriptive text is also optionally printed, defining SCOR terminology and including suggestions from Ebel (1979) for using portions of the output to improve tests.

Scoring and Record Keeping
While the Grade Report from PROC ITEM provides the scores and other record-keeping information desired by many instructors, SCOR additionally produces a listing which includes the responses which were marked wrong for each student tested. This feature permits a closer analysis of individual responding, which may aid in counseling as well as in test improvement. Individual grade reports which include scores and wrong responses may also be produced, with an optional display of the answer key. The individual reports may be handed back to those who took the test.

Raw Data Files
Instructors may elect to have SCOR produce raw data files as well as, or instead of, the scoring and item analysis output from SCOR. The SAS File Punch statement is used to produce data cards, and it is also used to produce virtual raw data files which may be sent to instructors' virtual machines. Regarding the latter, a batch job is created using SCOR to create all data files to be sent to virtual machines. By defining logical unit 13 (PT13FO01) as Sysout=1, these punch files are converted to print files which are then returned to the virtual reader of the staff virtual machine. This print file is edited using a CMS Exec, and the resulting files are punched to their respective virtual machine destinations. Converting the punch files to print files avoids creating blank records which could exist due to blocking the output in pages.

Error Checks
As mentioned earlier, CMS Exec's are used to produce a multi-step batch job, processing several tests at once. Since errors may occur in this process or in the mainframe system, measures have been programmed to help in debugging and/or in avoiding the failure of the whole multi-step job. An error will result when an instructor inadvertently specifies a range of possible choices which is actually narrower than what is found in the answer key. This discrepancy may also occur when the user marks more than one answer for a question on the answer key. SCOR uses the SAS character function VERIFY to abort the job step if this discrepancy is found. Another error could occur from using single quotes in the user-specified title in the PROC ITEM portion of SCOR; however, when the SAS system option DQUOTE is used in conjunction with the double quote mark (not two adjacent single quotes) this error is avoided. When these or other errors occur, due to temporary system quirks or space problems, the SAS system option ERRWARN stops execution of that particular job step. This along with COND=ERROR in the JCL EXEC statement, allows the next job step to begin...
execution despite failure of the preceding step.

Future Plans

Although SCOR has been very well received by our users, SAS system options and other measures are being explored in order to reduce execution time. Efficiency would doubtless increase if SCOR were adapted to interactive SAS. Two other enhancements are being considered in hopes of reducing staff operator time and human error. First, a procedure may be developed in which the user would specify desired features on an NCS answer sheet, which would become part of the raw input data file along with the answer key. Secondly, the user might similarly specify on an NCS sheet the items to be used in generating subtests. These two procedures would reduce the number of data encoding steps and focus more responsibility on the user for input accuracy.

Acknowledgements

The author wishes to thank Professor McGree, School of Education, Ohio University, who served as statistical consultant for this project. Thanks are due also the Computer Center staff for many valuable suggestions.

References


************** THE SCOR PROGRAM **************

//17FO8 EXEC SAS, OPTIONS='NOSOURCE NONOTES DQUOTE ERRORABEND',
// SORT=8,COND=EVEN
/ FT13FDO1 DD SYSCOUT=B
//DATA08 DD USN=GU.A alone,MCG4A,NCISDATA,DISP=SHR
DATA BEGIN; INFILE DATA08 END=FIN 095=5 ;
CHBS= 1 ; ITMNA=- 1 ; VM7 = 2 ;
MACBC MAXQ QIO % VARS = 10 ;
LENGTH Q1-MAXQ 3 I 2 ; TITLE;
INPUT #1 R M $CHAR20 ; #21 MONTH $1 ; #22 DAY $CHAR2 ; #24 YEAR $CHAR2 ;
#26E5 $CHAR $1 . #27 (SP1-SP6) (#1.) #28 SEX $1 ; #29 GRADE $1 . #30 (Q1-MAXQ) (#1.)
#31 TESTKEY $1.1 ;
ARRAY ITEMKEY (I) $ Q1-MAXQ;
DO OVER ITEMKEY; IF ITEMKEY EQ ' ' THEN ITEMKEY=' ' ; END;
*** PHYSICAL OR VIRTUAL RAW DATA FILES MAY BE CREATED, THE LATTER
*** IF DEFINING FT13FDO1 AT SYSCOUT=A;
FILE PUNCH; CARD=1; NAMEUSE= 1 ;
IF NAMEUSE EQ 1 THEN DO;
IF N_EQ 1 AND CARD EQ 1 THEN PUT 80 '08T39 ' 88 73**/
/ 81 'TESTSCORE DATA, TITLE: Ecn 534 QUIZ SMITH ' /
/ 81 80***;
PUT 81 ID $10 . #11 NAME $20. #31 MONTH $1. #32 DAY $2. #33 YEAR $2.
/ 836 (SP1-SP6) ($1.) #42 SEX $1 . #43 GRADE $1 . #44 (Q1-MAXQ) ($1.)
#45 TESTKEY $1.1 ;
ARRAY NAMEITEM (I) $ Q1-MAXQ;
DO OVER NAMEITEM; IF TESTKEY EQ ' ' THEN NAMEITEM=' ' ; END;
/ 81 'TESTSCORE DATA, TITLE: Ecn 534 QUIZ SMITH ' /
/ 81 80***;
PUT 81 ID $10 . #11 NAME $20. #31 MONTH $1. #32 DAY $2. #33 YEAR $2.
/ 836 (SP1-SP6) ($1.) #42 SEX $1 . #43 GRADE $1 . #44 (Q1-MAXQ) ($1.)
#45 TESTKEY $1.1 ;
ARRAY ITEMKEY (I) $ Q1-MAXQ;
DO OVER ITEMKEY; IF ITEMKEY EQ ' ' THEN ITEMKEY=' ' ; END;
*** PHYSICAL OR VIRTUAL RAW DATA FILES MAY BE CREATED, THE LATTER
*** IF DEFINING FT13FDO1 AT SYSCOUT=A;
FILE PUNCH; CARD=1; NAMEUSE= 1 ;
IF NAMEUSE EQ 1 THEN DO;
IF N_EQ 1 AND CARD EQ 1 THEN PUT 80 '08T39 ' 88 73***/
/ 81 'TESTSCORE DATA, TITLE: Ecn 534 QUIZ SMITH ' /
/ 81 80***;
PUT 81 ID $10 . #11 NAME $20. #31 MONTH $1. #32 DAY $2. #33 YEAR $2.
/ 836 (SP1-SP6) ($1.) #42 SEX $1 . #43 GRADE $1 . #44 (Q1-MAXQ) ($1.)
#45 TESTKEY $1.1 ;
ARRAY ITEMKEY (I) $ Q1-MAXQ;
DO OVER ITEMKEY; IF ITEMKEY EQ ' ' THEN ITEMKEY=' ' ; END;
*** PHYSICAL OR VIRTUAL RAW DATA FILES MAY BE CREATED, THE LATTER
*** IF DEFINING FT13FDO1 AT SYSCOUT=A;
FILE PUNCH; CARD=1; NAMEUSE= 1 ;
IF NAMEUSE EQ 1 THEN DO;
IF N_EQ 1 AND CARD EQ 1 THEN PUT 80 '08T39 ' 88 73***/
/ 81 'TESTSCORE DATA, TITLE: Ecn 534 QUIZ SMITH ' /
/ 81 80***;
PUT 81 ID $10 . #11 NAME $20. #31 MONTH $1. #32 DAY $2. #33 YEAR $2.
/ 836 (SP1-SP6) ($1.) #42 SEX $1 . #43 GRADE $1 . #44 (Q1-MAXQ) ($1.)
#45 TESTKEY $1.1 ;
ARRAY ITEMKEY (I) $ Q1-MAXQ;
DO OVER ITEMKEY; IF ITEMKEY EQ ' ' THEN ITEMKEY=' ' ; END;
*** PHYSICAL OR VIRTUAL RAW DATA FILES MAY BE CREATED, THE LATTER
*** IF DEFINING FT13FDO1 AT SYSCOUT=A;
FILE PUNCH; CARD=1; NAMEUSE= 1 ;
IF NAMEUSE EQ 1 THEN DO;
IF N_EQ 1 AND CARD EQ 1 THEN PUT 80 '08T39 ' 88 73***/
/ 81 'TESTSCORE DATA, TITLE: Ecn 534 QUIZ SMITH ' /
/ 81 80***;
PUT 81 ID $10 . #11 NAME $20. #31 MONTH $1. #32 DAY $2. #33 YEAR $2.
SUDSEA~ "ECON 534 QUIZ SMITH"; SDBE "ACRO"; SUSIES C6-Q10 "MICRO EBM";
DATA CNEA; SET ONE; IF _N_ EQ 1 THEN DELETE; KOUNT=1;
DATA TWO; IF _N_ EQ 1 THEN DO; SET XYZ; RETAIN NOBS; END;
SET TESTOUT; IF _N_ EQ 1 THEN DELETE; KOUNT=1;
DATA SCORES; SET TOTAL; IF _N_ EQ 1 THEN DELETE; KOUNT=1;
DATA SCORES2; TITLE XMNR;
IF _N_ EQ 1 THEN DELETE; KOUNT=1;
ARRAY COMPARE(A) $ Q1-MAXQ; TRANSFER ARRAY XSAVE(Y) X1-MAXQ;
DATA TWO; IF _N_ EQ 1 THEN DO; RETAIN NOBS; END;
SET TESTOUT; IF _N_ EQ 1 THEN DELETE; KOUNT=1;
DATA SCORES; SET TOTAL; IF _N_ EQ 1 THEN DELETE; KOUNT=1;
DATA SCORES2; TITLE XMNR;
IF _N_ EQ 1 THEN DELETE; KOUNT=1;
ARRAY COMPARE(A) $ Q1-MAXQ; TRANSFER ARRAY XSAVE(Y) X1-MAXQ;
DATA TWO; IF _N_ EQ 1 THEN DO; RETAIN NOBS; END;
SET TESTOUT; IF _N_ EQ 1 THEN DELETE; KOUNT=1;
DATA; SET COMB1; BY GROUP; PECC FREQ; BY GROUP;
DATA TRANSKEY: SET ONE(KEEP=NK1-MAXN); IF _N_ NE 1 THEN DELETE;
DATA S1MMABY1; SET ROC;
DATA SUMMARY1; SET TESTOUT;
*** PRINT SUMMARY STATISTICS FROM PROC UNIVARIATE OUTPUT DATA SET;
DATA SUMMARY; SET SUMMARY;
XTRASUM=1; IF XTRASUM=1 THEN DO; TITLE "ECGN 534 QUIZ SMITH ";
FILE PRINT HEADER=STRC LINE=LNCNT LINESLEFT=LF; PUT //;
@57 'NAME: ' NAME2 //; ELSE NAME2=NAME;
VARS=10; ARRAY INIT(S) Q1-MAXQ; ARRAY COMPARE(A) $ Q1-MAXQ; TRANSFER
DO OVER COMPARE; S=A; INIT=0; INIT=COMPARE; END; ARRAY XSAVE(Y) XI-MAXX;
DO OVER SAVKEY; A=B; Y=B; XSAVE=COMPARE;
IF COMPARE EQ SAVKEY THEN COMPARE=1; ELSE COMPARE=0; END;
CORRECT=SUM(OF Q1-MAXQ); NUMWRG=VARS-CORRECT;
*** PRINT INSTRUCTOR'S LIST OF SCORES AND WRONG ANSWERS;***
FILE PRINT HEADER=STRC LINE=LNCNT LINESLEFT=LF; I=1;
DO OVER COMPARE; Y=A; IF COMPARE EQ 0 THEN DO;
VARS=XSAVE; J=I; DEPAX=Y; IF I NE VARS THEN I=I+1; END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;
WRONGS=0; PLACE=33; PUT @5 NAME2 20. @26 ID 10. @37 CORRECT 3. 8;
IF NUMWRG EQ 0 THEN DO; PUT //; RETURN; END;
DO I=1 TO NUMWRG; J=I; WRONGS=WRONGS+1;
IF WRONGS EQ 10 OR I EQ NUMWRG THEN DO;
PUT ALLPLACE DRAY 3. (' ALLPLACE + WRGS 1.) *;
IF I EQ NUMWRG THEN DO; LINES=VARS/10; LINCM=LF;
IF LINES GT INT(LINES) THEN DC; LINES=LINES+1;
IF LF LT LINES THEN DC; LINES=LINES+1;
END;
END;